Qualitative analysis of synchronizing probabilistic systems

Mahsa Shirmohammadi

UNIVERSITÉ LIBRE DE BRUXELLES
Service des méthodes formelles
et vérification

ÉCOLE NORMALE SUPÉRIEURE
DE CACHAN
Laboratoire spécification et vérification

Qualitative Analysis of Probabilistic Synchronizing Systems

Mahsa Shirmohammadi

Dissertation submitted for the joint degree of
Doctor of Philosophy in computer science
from
Université Libre de Bruxelles, Belgium
and
Ecole Normale Supérieure de Cachan, France

Committee in charge:

Laurent Doyen: Supervisor
Thierry Massart: Supervisor
Stefan Kiefer: Reviewer
Jeremy Sproston: Reviewer
Joost-Pieter Katoen: Examiner
Jean François Raskin: Examiner
Béatrice Bérard: Examiner
Emmanuel Filiot: Examiner

Defense date: 10th December 2014

To *Maryam*, my mother.

Acknowledgment

I gratefully acknowledge all who supported me to accomplish my wish. Being so lucky to have supports from several countries and in several languages; I try to mention all, however I am surely bound to forget a few.

Laurent, I admire the way you turn science into art. Your seriousness, patience, insights, enthusiasm and not to forget your expectations motivated me to work hard, to be thirsty to learn how to carry out deep research, write clear proofs and give good talks. I am so grateful that you made me rewrite proofs several times, for some more than there are fingers in a hand, to reformulate and simplify my ideas and to turn a problem up and down, and look from one angle to another. Wishing that I had filled my backpack with more of your knowledge and advices, I am at the same time grateful for the freedom I had while writing my thesis; I found my academic personality those days, thanks Laurent.

Thierry, we have heard "doubt your beliefs and believe in your doubts" more than once from Laurent; however the one who taught it to me is you. I will always remain amazed by your capability of finding mistakes in proofs and counter examples for conjectures. You taught me to question and challenge myself, to not be easily convinced, thanks for all of those, and much more for the social support you never forgot to offer me. Thanks for welcoming me at first in Brussels, for helping me integrate in my new life, for simply being 'there' when the sensitivity of a girl overlapped with her professionalism. Thanks Thierry.

I am so pleased that Stefan Kiefer, Jeremy Sproston, Jean François Raskin, Joost-Pieter Katoen, Béatrice Bérard and Emmanuel Filiot accepted to be the other jury members of my thesis. A further thank for all asked questions during my private defense, some of which spark new directions of research.

I am grateful to the Belgian national fund for scientific research (F.N.R.S.) and International relation office of ENS de Cachan which provided me with two scholarships during my study: one allowed me to fully concentrate on my thesis, and the other one facilitated my dual life between France and Belgium over three years.

I should not forget any member of LSV (ENS de Cachan) and the Verification group at ULB, with a special mention to Benedikt, Frédéric, Gilles, Dietmar, Luc, Cristina, Thida, Thomas and Julien who smiled at me and made refreshing small talks every time I met them during coffee breaks. I also appreciate all the helps I received from the secretaries of both labs: Pascaline, Maryka, Véronique and Virgine several times took care of administrative papers instead of the lost-in-French Mahsa.

I wish to say thanks to Nicolas Markey (a big thanks), Stéphanie Delaune and Alain Finkel for constructive advice; to Kim Larsen and Line for welcoming me in Aalborg university while collaborating; to Alexander for teaching me how to teach; to Sylvain for saving my introduction; to Nathann for knocking the door of my office in a gloomy winter day; to all adorable girls who surrounded both me and my surviving feminine trait: Sinem jan janam, dear Aina, delicate Marie, small mommy Sandie, strong Raffaella, happiest ever Barbara, lively Aiswarya, organized Laure, fun Juliette and the other fun Juliette, kind Eli,

sportive Line, creative Carla, hard-working Atefeh, backgammon-champion Öykü, eastern-soul Xiajou, sweet Agathe, the only-musician-friend Nona and far-away Eleni (girls, I lived moments with you, your weddings, your 25th, 30th birthdays, your pregnancy worried days, your graduations, your love or hate days, simply normal days, thank you, thank you for all); to Gabriel and Tristan to be my first and so good friends in Europe; to Guillermo, Romain and the other Romain for being such unique charming office-mates, and for enjoying summer BBQs where I have to mention Ocan and Mathieu, the delicious-sauce maker; to Chris, Hernan, Vince and César for never rejecting to try what I cook and even complimenting it, and more than that for the friendship and respect between us; to Ali and Hani my best ever friends that keep our relation fresh and deep from such a long distance; to my soul-mates Mina, Mona and Somayeh for not letting me disconnect from the other world of mine by long and late phone conversations; to Mani for lifting spirit words; and to dear Hossein for encouraging me to pursue my post-graduate study and be more independent and more honest.

Vincent and Christoph, I am indebted to you guys. Ask me why, and I need to think minutes before taking out one of the hundreds cards from the deck of friendly supports, scientific advices, stressful critical times, relaxed laughs and smiles, sunny and rainy but all memorable days. Vince, you have been so generous to me, so more than what others can guess, so perfectly careful with the fragile glass of my soul, thanks. Chris, you make me look into myself and hunt clues of peace and braveness whose existences were lost over years, thanks. Special thanks to Vince for going through all my thesis, and to Chris for commenting my introduction and my post-doctorate application to Oxford. Thanks Vince, thanks Chris.

Madaram Maryam, my mother Maryam, one tip of a finger from your kindness and devotion was sufficient to make me who I am, though you lived your life in each of ours, your children. My eyes have always felt shy to look up at the height of the shadow of your support and to say *thanks*; I can only write it down as down as my short mount of achievements. Thanks Maryam, Madaram.

Mahsa so thankful,
after her first Thanksgiving dinner at Tony's place
Paris, FMSH library
December 2014.

Abstract

Markov decision processes (MDPs) are finite-state probabilistic systems with both strategic and random choices, hence well-established to model the interactions between a controller and its randomly responding environment. An MDP can be mathematically viewed as a $1\frac{1}{2}$-player stochastic game played in rounds when the controller chooses an action, and the environment chooses a successor according to a fixed probability distribution.

There are two incomparable views on the behavior of an MDP, when the strategic choices are fixed. In the traditional view, an MDP is a generator of sequence of states, called the *state-outcome*; the winning condition of the player is thus expressed as a set of desired sequences of states that are visited during the game, e.g. Borel condition such as reachability. The computational complexity of related decision problems and memory requirement of winning strategies for the state-outcome conditions are well-studied.

Recently, MDPs have been viewed as generators of sequences of probability distributions over states, called the *distribution-outcome*. We introduce synchronizing conditions defined on distribution-outcomes, which intuitively requires that the probability mass accumulates in some (group of) state(s), possibly in limit. A probability distribution is p-synchronizing if the probability mass is at least p in some state, and a sequence of probability distributions is always, eventually, weakly, or strongly p-synchronizing if respectively all, some, infinitely many, or all but finitely many distributions in the sequence are p-synchronizing. For each synchronizing mode, an MDP can be (*i*) *sure* winning if there is a strategy that produces a 1-synchronizing sequence; (*ii*) *almost-sure* winning if there is a strategy that produces a sequence that is, for all $\epsilon > 0$, a (1-ϵ)-synchronizing sequence; (*iii*) *limit-sure* winning if for all $\epsilon > 0$, there is a strategy that produces a (1-ϵ)-synchronizing sequence. We consider the problem of deciding whether an MDP is winning, for each synchronizing and winning mode: we establish matching upper and lower complexity bounds of the problems, as well as the memory requirement for optimal winning strategies.

As a further contribution, we study synchronization in probabilistic automata (PAs), that are kind of MDPs where controllers are restricted to use only word-strategies; i.e. no ability to observe the history of the system execution, but the number of choices made so far. The synchronizing languages of a PA is then the set of all synchronizing word-strategies: we establish the computational complexity of the emptiness problems for all synchronizing languages in all winning modes.

We carry over results for synchronizing problems from MDPs and PAs to two-player turn-based games and non-deterministic finite state automata. Along with the main results, we establish new complexity results for alternating finite automata over a one-letter alphabet. In addition, we study different variants of synchronization for timed and weighted automata, as two instances of infinite-state systems.

Résumé

Les *Markov Decision Process* (MDP) sont des systèmes finis probabilistes avec à la fois des choix aléatoires et des stratégiques, et sont ainsi reconnus comme de puissants outils pour modéliser les interactions entre un contrôleur et les réponses aléatoires de l'environment. Mathématiquement, un MDP peut être vu comme un jeu stochastique à un joueur et demi où le contrôleur choisit à chaque tour une action et l'environment répond en choisissant un successeur selon une distribution de probabilités fixée.

Il existe deux représentations incomparables du comportement d'un MDP une fois la stratégie fixée. Dans la représentation classique, un MDP est un générateur de séquences d'états, appelées *state-outcome* ; les conditions gagnantes du joueur sont ainsi exprimées comme des ensembles de séquences désirables d'états qui sont visités pendant le jeu, e.g. les conditions de Borel. La complexité des problèmes de décision et de mémoire requise des stratégies gagnantes pour les conditions dites *state-outcome* ont été langement étudiées.

Depuis peu, les MDPs sont aussi considérés comme des générateurs de séquences de distributions de probabilités sur les états, appelées *distribution-outcome*. Nous introduisons des conditions de synchronisation sur les distributions-outcome, qui intuitivement demandent à ce que la masse de probabilité s'accumule dans un (ensemble d') état, potentiellement de façon asymptotique. Une distribution de probabilités est p-synchrone si la masse de probabilité est d'au moins p dans un état ; et la séquence de distributions de probabilités est toujours, éventuellement, faiblement, ou fortement p-synchrone si, respectivement toutes, au moins une, une infinité ou toutes sauf un nombre fini de distributions dans la séquence sont p-synchrones. Pour chaque type de synchronisation, un MDP peut être (i) *sûrement* gagnant si il existe une stratégie qui génère une séquence 1-synchrone ; (ii) *presque-sûrement* gagnant si il existe une stratégie qui génère une séquence $(1-\epsilon)$-synchrone et cela pour tout $\epsilon > 0$; (iii) *asymptotiquement* gagnant si pour tout $\epsilon > 0$, il existe une stratégie produisant une séquence $(1-\epsilon)$-synchrone. Nous considérons le problème consistant à décider si un MDP est gagnant, pour chaque type de synchronisation et chaque mode gagnant : nous établissons les bornes supérieures et inférieures de la complexité de ces problèmes et de la mémoire requise pour une stratégie gagnante optimale.

En outre, nous étudions les problèmes de synchronisation pour les *automates probabilistes* (PAs) qui sont en fait des instances de MDP où les contrôleurs sont restreints à utiliser uniquement des *stratégies-mots* ; c.à.d. qu'ils ne peuvent pas observer l'historique de l'exécution du système et ne peuvent connaître que le nombre de choix effectués. Les langages synchrones d'un PA sont donc l'ensemble des stratégies-mots synchrones : nous établissons la complexité des problèmes des langages synchrones vides pour chaque mode gagnant. Nous répercutons nos résultats de synchronisation obtenus sur les MDPs et PAs aux jeux *à tours* à deux joueurs ainsi qu'aux automates finis non-déterministes. Nous établissons de nouveaux résultats de complexité sur les automates finis alternants avec des alphabets à une lettre. Enfin, nous étudions plusieurs variations de synchronisation sur deux instances de systèmes infinis que sont les automates temporisés et pondérés.

Contents

1 Introduction

"We are like dice thrown on the plains of destiny."

- Rita Mae Brown

The last slice of a pizza is usually too tempting for a child to think of sharing with a sibling, this more delicious than all other slices. The parents' solution to avoid possible arguments could be as easy as tossing a coin and letting fate decide. The two siblings learn what is the *chance* of *winning* that last slice: one half. If not this exact story, a familiar story is a player in the backgammon game [1] wishing for a double when getting mars; no matter how much faith he has, he knows within himself that there is only one-sixth of a chance for that wish to come true. Why is that belief vividly carved in each of our minds? Simply, we know that there are thirty six possible pairs of faces for two dice, the desired *event* of a double is a set of only six pairs, and finally we assume that dice are *fair*, meaning that when rolling a dice, there is an equally likely chance for each face to come up. In other words, if one repeats the *experiment* of rolling a dice for a sufficiently large number of *rounds*, each face will *statistically* show up in one-sixth of the rounds.

Gambling casino chips on a *random* combination of *outcomes* in a game is an entertaining activity for players and a business for the house. The simplest popular gaming

1. One of the oldest board game; excavations in Shahr-e Sukhteh ("The Burnt City") in Iran have shown that the game existed there around 3000 BC. The game is played on a board with black and white checkers for two players. In each round, players in turn throw two dice and move the checkers on the board as many cells as the numbers shown on the dice. A player wins after taking out all her checkers from board. When the loser has no checkers out, he is *mars* and misses two points. A *double* is when both dice have the same face up. A player who rolls doubles plays the numbers on the faces twice.

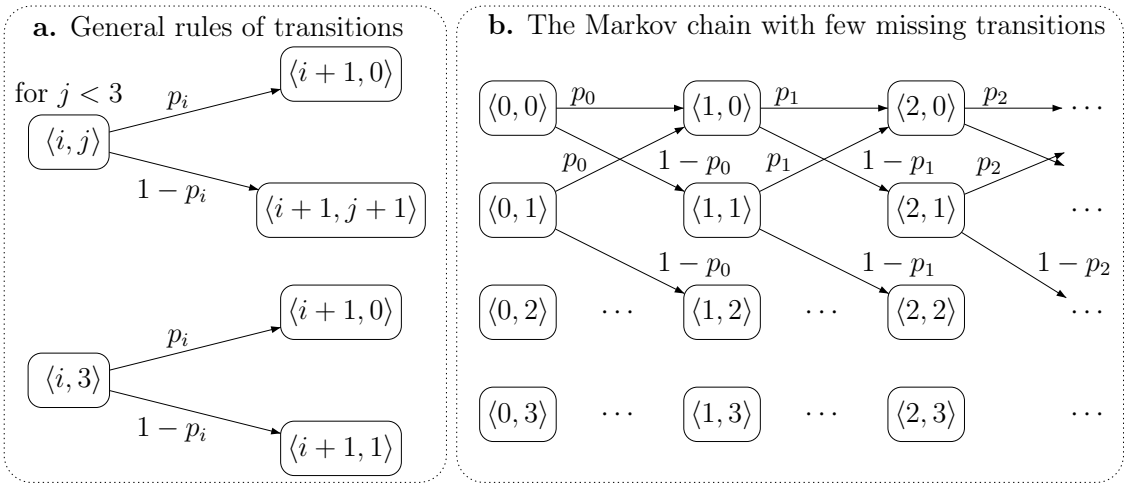

Figure 1.1: A Markov chain that models a simplified Futurity slot machine where a state $\langle i, j\rangle$ indicates the internal mode with the i-component: the modes 0 and 2 are mapped to tight and mode 1 is mapped to loose. From a state with mode i, the chain moves to a state with mode $i+1 \pmod 3$. The j-component of states indicates the number of consecutive losses. The probability p_i is the probability for the player to win in mode i.

machines are *slot machines* that take coins and randomly bring up on their reels three symbols out of a set of symbols, and if the outcome consists of three identical symbols, the player wins the bet. The slot machines are usually designed to be slightly favorable to the casino while still letting players trust the fairness of the game. The Futurity slot machine designed by the Mills Novelty Company of Chicago in 1936 [EL10], for instance, has a special feature that is returning back all coins bet if the player loses consecutively for a certain number of times; this feature gives the player the impression of having more chances to win. However, there is another feature invisible to the player: the machine has a cyclic sequence of internal modes consisting of some *loose* and some *tight* modes. The machine is programmed to have an expected payoff in favor of the player in the loose mode, and in favor of the casino in the tight mode. The sequence of invisible internal modes for the machine has more tight modes, and so is to the advantage of the casino. To delicately hide this trick and at the same time ensure the profit of the casino in the long run, casino owners wish to analyze the functionality of slot machines before investment. To this aim, a slot machine can be modeled by a (discrete-time) *Markov chain*, that is a mathematical object to model systems following a sequence of linked events. In the model of a slot machine, the linked events are the consecutive rounds of plays with no win.

A simplified version of a Futurity slot machine is graphically described in Figure 1.1. The Markov chain models a machine with the cyclic sequence of internal modes "tight, loose, tight" and the misleading feature of returning all coins after three consecutive losses. The *states* of the machine are drawn as boxes and labeled with two components $\langle i, j\rangle$,

see Figure 1.1.b. The i-component encodes the "internal mode" (i.e., the index of internal modes in the cyclic sequence) and j-component stores "the number of consecutive losses". *Transitions*, drawn as arrows between states, encode the feeding of the machine with a coin and the resulting win or loss that changes the internal mode and may also change the number of consecutive losses. Thus, transitions move the chain from one state to another one. Figure 1.1.a depicts the general rules of transitions. Transitions are augmented with probabilities that indicate how likely the player is to win or lose at states of the chain. A transition starting in some state $\langle i, j\rangle$ is augmented with probability p_i if it encodes a win; it is thus redirected to $\langle i+1, 0\rangle$ where the number of consecutive losses is reset. Otherwise, such transition is augmented with probability $1 - p_i$ and is redirected to $\langle i + 1, j + 1\rangle$ to store one more loss. The machine must meet a set of requirements and conditions called a *specification*. For instance, abiding the law, slot machines must be programmed to pay out as winnings at least 75% of the money wagered by players, a *safety condition*.

Most gamblers find gambling in a casino entertaining as long as the loss is roughly comparable to the gain. Stock brokers though might not feel entertained when losing even a small amount of money in the equity markets. Since markets are usually unstable and subject to dramatic and sudden changes, *optimal decisions* must be made instantaneously in order to exchange, buy and sell, securities that increase their profits. To predict the sudden changes or predict the effect of a placed exchange in the market, traders must carefully monitor and analyze the history of all trades and fluctuations, as statistical experiments. Extracting information from such big amounts of data in a short time is error-prone and tiring for humans. On the other hand, enthusiastic human traders commonly rely for their decisions unconsciously on their instincts, in particular after getting exhausted. Automated traders or *trading robots* are pieces of software designed to perform various calculations and analyze large amounts of data quickly in order to make optimal decisions almost instantly and more precisely than human traders. To design a trading robot, the market is usually modeled as a mathematical object such as a *Markov decision process*, which is a generalization of Markov chains. In addition to the stochastic aspects of systems that Markov chains can model, Markov decision processes model a kind of non-determinism to show the variety of the robot's choices. A Markov decision process consists of several states and transitions. States encode the financial status of the market such as the price of companies' shares on the stock market. Transitions encode placing orders that may cause changes in the market, thus the current state of the market will be changed to another state. Transitions are augmented with labels [2] and probabilities: labels vary between different possible orders, and the probabilities are computed by statistical analysis. The probability of a transition from one state to a *successor* state is showing how likely the placed order changes the prices in the market into what is indicated in the successor. A robot trader is programmed to place electronic orders in each state of the model. The robot must meet some specification, e.g., the dealings of a robot of a trading company must never cause dropping the value of

2. It is the main difference between Markov chains and Markov decision processes: transitions in Markov chains are only augmented with probabilities, though they are augmented with both labels and probabilities in Markov decision process.

the shares of the company in the market (*safety condition*). Otherwise, the wealth of the company is automatically decreased and not to forget the bad reputation.

The shareholders of a trading company would not gamble in replacing the human element with automatic traders without guarantees that the trading robots perform as well as advertised. Such guarantees ought to promise that the robots make optimal decisions to fulfill the specification. A way to ensure the *correctness* of the designed robot is *testing*, which is broadly used in commerce and industry. Testing usually consists of running the robot on artificial markets (or simulations of the market) and observing the outcomes of the robot's decisions. Having a set of correct behaviors, testing thus checks whether the outcome of a simulation contains those correct behaviors. The drawback of the testing approach is its non-completeness, which may cause missing some unexpected situations where the decisions of the robot breach a safety condition and could cause the shareholders to lose some money. Another approach is *formal verification*, a technique that promises the *absolute guarantee* for the correctness of a system by providing a mathematical proof (if such a proof exists [3]).

System verification also comes to prominence outside the financial sphere. It becomes even more crucial when any failure even a small deviation from the required behavior for the system is life threatening. In December 2013, it was reported that the first implanted artificial heart stopped beating. This misfortune caused the death of a seventy-six years old man who had received the heart, and the disappointment of French surgeons as well as many patients in need of a heart transplant. The artificial heart was designed to beat for at least five years, though an unexpected circuit shortcut happened after 74 days and caused the dysfunction. A mathematical proof to verify the correctness of the artificial heart would have ideally prevented the previous failure, and hopefully all dysfunctions of the second implanted heart.

1.1 Formal verification of reactive systems

Reactive systems, such as artificial organs and stock markets, are non-terminating systems that maintain an ongoing interaction with the environment rather than returning some value upon termination. In the field of *formal methods and verification of reactive systems*, we abstract reactive systems with mathematical models such as *transition systems*, *finite state automata*, Markov chains, Markov decision processes, etc., in order to take out some complexities in the system while retaining enough information to verify the *correctness* of the design. The behavior of a system is usually viewed as a sequence of interactions between the system and the environment; roughly speaking, the set of all behaviors is the language of the model. Computer-aided verification develops systematic approaches that verify whether all behaviors of the model correctly meet some required specification, that are expressed by mathematical objects such as some formula in some logic (e.g. a temporal logic such as LTL, CTL) or some regular (or ω-regular) language or automata. The cor-

3. The verification problem for some cases is undecidable.

rectness relation is usually chosen based on the model of the system and the mathematical object expressing the specification. For the case where both the system and the specification are modeled by automata, language inclusion ensures that all sequences of reactions in the system are desired behaviors in the specification language. Considering language inclusion as the correctness relation, the system behaviors are distinctly partitioned into "correct" and "incorrect" ones, and the verification seeks to guarantee that all behaviors are correct without any nuance. This *Boolean* and *qualitative* view on the correctness is classical and is well-suited for modelling systems where no fault can be tolerated, such as artificial organs.

The system model and the specification can be augmented with some *quantitative* aspects to be suitable for real-life applications. Real-time systems as an example are effectively modeled by *timed automata* where the model is enriched with some clocks and the transitions are fired with respect to time constraints. Specifications, thus, in this setting may express time operational deadlines or constrain the longest response time [AD94]. The battery consumption of mobile systems is described by *weighted automata* where transitions carry weights to reflect the amount of power recharged or discharged on taking each transition, and the evolution of system is subject to constraints on the accumulated weight (that is the power level). Specifications may then describe the power-life resistance in different scenarios [FJLS11]. As we have seen in systems such as stock markets, probabilities are assigned to possible responses or reactions in an abstracted model of the system in order to statistically predict the behavior of such complex systems; *probabilistic automata* and *Markov* models are well-established syntaxes for describing such systems [dA97, BK08]. Despite quantitative aspects in the system models and specifications, the correctness may follow the Boolean view, and impose a yes/no answer to the verification problem; the *quantitative verification* though seeks a graduated correctness for systems.

1.2 Probabilistic systems

Markov chains and Markov decision processes (MDPs) are widely used in the quantitative and qualitative verification of probabilistic systems [Var85, CY95, FV97, dA97, BK08]. Discrete-time Markov chains (or simply Markov chains) are arguably most popular model for verifying the evolution of a self-sufficient random process that has no interaction with its environment. As an extension, MDPs are a well-established syntax for open systems in which interactions between a controller and its stochastic environment happen. In MDPs, both non-deterministic and probabilistic choices thus coexist. We have seen how these Markov models can be used to model casino machines and stock markets. There are also applications in planning, randomized algorithms, communication protocols and biology [AH90, FP06, BBS06, KNP08].

The *model-checking problem* seeks the verification of a given property which can be expressed by qualitative or quantitative properties. The *controller synthesis problem* asks for computing a control strategy that ensures a specification, and thus is relevant only for MDPs (and not for Markov chains).

1.2.1 State-outcome and distribution-outcome views

There are two incomparable views on Markov models: *state-based* and *distribution-based* views.

Markov chains. In the state-based view, Markov chains are treated as transition systems with states to which a fixed probability distribution, that determines the successor states, is assigned. The transition thus happens from one state to another, and a Markov chain is then a generator of sequence of states, called the behavior or the *state-outcome* of the Markov chain.

In the distribution-based view, Markov chains are treated as transformers of probabilistic distributions over the nodes of the chains. A Markov chain can thus be encoded in a deterministic transition system whose states are probability distributions from which there is a unique outgoing transition to the successor state, i.e., the successor probability distribution. The transition thus happens from one probability distribution to another, and a Markov chain is then a generator of probability distributions, called the *distribution-outcome* or *symbolic-outcome* of the Markov chain. The distribution-outcome is an infinite sequence of probability distributions obtained by iteratively applying the probabilistic transition matrix of the chain to the unique initial distribution (over nodes of the chain).

MDPs. The controller in an MDP can be instructed by a *strategy* (sometimes called *scheduler* [BK08]). When a strategy is fixed in an MDP, the system induced under the strategy is a Markov chain, and as expected the two views can carry over. The MDP under a fixed strategy thus can be viewed as a state-outcome or distribution-outcome generator.

Comparison of state-outcomes and distribution-outcomes. The state-outcome of stochastic systems is classical and well-studied, for example see [Var85, CY95, FV97, BK98a, BK98b, dAH00, KNP05, BKHW05, BCH$^+$07, BK08, EKVY08, CD11, BHK$^+$12, BGB12, CJS12, CHJS13]. Typical state-based properties for Markov models are safety, reachability, Büchi, and coBüchi, which require the system to visit a target state always, once, infinitely often, and ultimately always, respectively. The quantitative and qualitative analysis of such state-based properties and model checking of probabilistic temporal logics such as PCTL, PCTL* has been well-studied for Markov models.

The distribution-outcomes of stochastic systems has recently been used in several works on design and verification of reactive systems [KVAK10, CKV$^+$11, AAGT12]. This semantics is adequate in many applications, such as systems biology, sensor networks, robot planning, etc. [BBMR08, DMS14a, HMW09]. Such system consist of several copies of the same process (molecules, sensors, robots, etc.), and the relevant information along the execution of the system is the number of processes in each state, or the relative frequency (i.e., the probability) of each state.

The logics defined on the two semantics, state-outcome and distribution-outcome, are incomparable, i.e., there are properties that are expressible in one but not the other. In particular, the approaches to decide model-checking problem in one cannot be reduced to

the other [KVAK10]. In one hand, intuitively, logics like PCTL and PCTL* do not allow for reasoning about the probability of being at different (groups of) states at the same time, whereas we will see an example of such properties expressed on the distribution-outcome. On the other hand, properties expressible with the distribution-based semantics do not take into account the branching structure of the system. In addition, for probabilistic systems a natural bisimulation relation on the distribution-outcome has recently been introduced [HKK14], and has been proved to be incomparable with the standard bisimulation relation (defined on the state-outcome).

1.2.2 Analysis of state-outcomes in Markov chains

In the state-based view, a Markov chain is viewed as a generator of sequences of states. A σ-algebra can be defined on the probability space produced by the sequence of states, and the probability of *events* that are sets of sequences of states, can be measured. Probabilistic linear time properties and probabilistic temporal logics such as PCTL, PCTL* assert quantitative or qualitative properties as the specification.

Verifying a qualitative property that usually asks whether a certain event will happen almost-surely (i.e., with probability 1) or with non-zero probability, is called the qualitative analysis of Markov chains. An instance of a qualitative property is the PCTL formula

$$\mathsf{Pr}_{=1}(\Box\Diamond\langle 0,3\rangle) \wedge \mathsf{Pr}_{=1}(\Box\Diamond\langle 1,3\rangle) \wedge \mathsf{Pr}_{=1}(\Box\Diamond\langle 2,3\rangle)$$

where $\mathsf{Pr}_{=1}$ requires that the events $\Box\Diamond\langle i,3\rangle$ happens with probability 1, $\Box\Diamond\langle i,3\rangle$ requires that the state $\langle i,3\rangle$ is infinitely often visited along the system execution. This formula asserts the following almost-surely property for the model of Futurity slot machine drawn in Figure 1.1 on page 2:

"all states encoding the three consecutive losses in the game, are almost-surely visited infinitely often."

Quantitative analysis of Markov chains is similar to model-checking problem of qualitative properties, but constrain the probability or expectation of certain events with regards to any value, not only the extreme bounds [4]. An instance of a quantitative property is

$$\mathsf{Pr}_{<0.05}\big((\langle 0,3\rangle \vee \langle 1,3\rangle \vee \langle 2,3\rangle) \rightarrow \bigcirc\bigcirc\bigcirc(\langle 0,3\rangle \vee \langle 1,3\rangle \vee \langle 2,3\rangle)\big)$$

where $\bigcirc$ is the temporal operator to assert a property on the next state visited along the system execution. This formula asserts the following quantitative property for the model of Futurity slot machine drawn in Figure 1.1 on page 2:

"the scenario where a player loses three times consecutively and gets all his coins returned, must only happen two times in a row with the probability less than 0.05."

4. In qualitative analysis, the only allowed probability bounds are the extreme bounds, one and zero: $\mathsf{Pr}_{=1}$ (almost-surely) and $\mathsf{Pr}_{>0}$ (non-zero).

In order to model-check qualitative state-based properties, graph-based techniques are employed [BK08]. However, the quantitative analysis usually is reduced to solving a system of linear equations.

1.2.3 Analysis of distribution-outcomes in Markov chains

In the distribution-based view, a Markov chain is viewed as a generator of sequence of probability distributions. The specification can thus assert properties on the current probability of a set of nodes that is not expressible in the state-based logics.

An instance of such a property is

$$\exists t \text{ such that } X_t(q) \geq 1$$

where X is, roughly speaking, the chance for the Markov chain to be in q at time t. The above formula asserts that

"there is a time t when the Markov chain is almost surely in the state q."

Qualitative and quantitative analysis differ in the distribution-based view like they do in the state-based semantics. Model checking of qualitative limit distribution-based properties in Markov chains usually relies on the computation of stationary distributions and periodicity of the chains, which has been well-studied, e.g. see [KS83, Nor98, Ser13]. In recent works, the verification of quantitative properties of the distribution-based semantics was shown undecidable [KVAK10]. Decidability is obtained for special subclasses [CKV$^+$11], or through approximations in which distributions are discretized using intervals [AAGT12].

1.2.4 Analysis of state-outcomes in MDPs

In the state-based semantics, when resolving non-determinism by strategies, there can be different probability spaces in an MDP, and thus the probability of a certain event might be measured unequally [Var85]. The quantitative analysis of an MDP against a specification, expressed for example by PCTL* or ω-regular properties, amounts to calculating the minimal and maximal probabilities that can be guaranteed when ranging over all strategies. The qualitative analysis simply asks whether there exists a strategy ensuring that the specification holds almost-surely (with probability 1) or limit-surely (with probability arbitrary close to 1); and the synthesis problem seeks the computation of such winning strategies. The calculation of minimal and maximal probabilities is well-studied and is usually achieved by graph algorithms operating on the graph underlying the MDP accompanied with some linear constraints. The obtained linear programs can be solved by means of an iterative approximation algorithm (called value iteration) [BK08]. For traditional ω-regular properties, in particular safety and reachability conditions, memoryless strategies are sufficient as optimal strategies [CH12]. To solve the synthesis problem, such optimal strategies can be constructed by some complementary analysis while computing the probability bounds for model checking.

1.2.5 Analysis of distribution-outcomes in MDPs

Compared to the state-outcome in MDPs, there has been little work on the analysis of the distribution-outcome view. Since the Skolem problem, which has been open for over eighty years, can be reduced [5] to the quantitative analysis, studying the qualitative analysis of the distribution-based semantics is more promising.

There is only one logic defined on the distribution-outcomes in MDPs. The logic is defined by propositions using linear inequalities over probability distributions, and using modal operators to reason about temporal behavior [KVAK10]. The verification of the defined logic is in general undecidable even for a restricted class of strategies that determine their choice of actions based on the current state of the system and the number of choices made so far, called *counting strategies* (or *Markovian schedulers*). The decidability results are only obtained by imposing some restrictions either on the class of counting strategies, or on the propositions (where the inequality in the proposition is non-zero).

As we will see, the main results of this thesis lie in the direction of the qualitative analysis of the distribution-outcomes in MDPs. We introduce different variants of synchronizing properties on distribution-outcomes of MDPs. In contrast with previous work, we consider the qualitative analysis with the extreme bound 1, and we prove that synchronizing properties are decidable for a general class of strategies that select actions depending on the full history of the system execution, called *perfect-information* strategies.

1.3 Synchronizing problems in MDPs

The main contribution of this thesis is a novel approach to the qualitative analysis and the synthesis problem of the distribution-outcomes in MDPs. We introduce synchronizing properties defined on the distribution-outcomes, which require that the winning strategy brings the MDP in some (group of) state(s) with a large probability, possibly in limit. In other words, the probability mass (say at least probability $p > 0$) in the distribution, that indicates the likelihood of the system to be in different states at specific states, is accumulated in some target state. We define variants of p-synchronizing conditions on the symbolic-outcomes of an MDP, and study the qualitative analysis of such properties where $p = 1$ or p tends to 1.

We consider the symbolic-outcome of the behaviors of MDPs as sequences $X_0X_1X_2\cdots$ of probability distributions $X_i : Q \to [0, 1]$ over the finite state space Q of the system, where $X_i(q)$ is the probability that the MDP is in state $q \in Q$ after i steps. For $0 \leq p \leq 1$, a distribution is *p-synchronized* if the probability mass accumulated in a single state (or a group of states) is at least p. An MDP is always (resp., eventually, weakly or strongly) p-synchronizing if the probability p is always (resp., ultimately once, infinitely often, or ultimately at every step) synchronized along the execution. More precisely, a sequence $X_0X_1\cdots$ of probability distributions is

5. The results also holds for Markov chains, where the quantitative verification of the distribution-outcome semantics is in general already undecidable.

(a) always p-synchronizing if X_i is p-synchronized for all i;

(b) eventually p-synchronizing if X_i is p-synchronized for some i;

(c) weakly p-synchronizing if X_i is p-synchronized for infinitely many i's;

(d) strongly p-synchronizing if X_i is p-synchronized for all but finitely many i's.

An MDP is thus eventually, always, weakly or strongly p-synchronizing if there exists some strategy whose symbolic-outcome is accordingly eventually, always, weakly or strongly p-synchronizing. It is easy to see that always p-synchronizing implies strongly p-synchronizing, which implies weakly p-synchronizing. Moreover, weakly p-synchronizing implies eventually p-synchronizing. We show that, in general, this hierarchy is strict. We study the qualitative analysis of synchronizing conditions in such quantitative models. The qualitative synchronizing properties, corresponding to the case where either $p = 1$ or p tends to 1 are analogous to the traditional safety, reachability, Büchi, and coBüchi conditions (in the state-based semantics). We consider the following qualitative (winning) modes:

(i) sure winning, if there is a strategy that generates an {always, eventually, weakly, strongly} 1-synchronizing sequence;

(ii) almost-sure winning, if there is a strategy that generates a sequence that is, for all $\epsilon > 0$, {always, eventually, weakly, strongly} $(1 - \epsilon)$-synchronizing;

(iii) limit-sure winning, if for all $\epsilon > 0$, there is a strategy that generates a {always, eventually, weakly, strongly} $(1 - \epsilon)$-synchronizing sequence.

The definitions are summarized in Table 3.1 on page 47. These three winning modes are classically considered in stochastic games; MDPs can be viewed as $1\frac{1}{2}$-player stochastic games. Some games in casinos can be analyzed with such $1\frac{1}{2}$-player stochastic games. Until recently, slot machines were not elaborate enough to have any human interaction other than feeding the machine with the betting coins. However, with microprocessors nowadays ubiquitous, a more interactive gambling game has become widely popular, that is *video slot machines*. This game can in fact be viewed as a computer game where players interact only via a touch screen and where there is no mechanical constraints. It allows the designers to display more virtual reels and also add some interaction with the player through bonus features. For instance, when the player hits some specific combination on the reels, he may choose to either obtain free spins or hold some reels while the others are re-spun, or bet for another prize, etc. Due to this human interaction in the bonus features, a video slot machine cannot be modeled anymore by a Markov chain but requires an MDP where the random responses of the environment is predicted by a statistical study on human-player psychology. The aim is to synthesize a controller microprocessor for slot machines such that the specifications are met. Thus, the machine is modeled by an MDP, the choices of the microprocessor are the strategic choices and the human choices are resolved randomly based on the probability distributions obtained by statistical experiments.

Moreover, having a distributed network of connected slot machines, designers may offer a new exciting large prize, usually called "jackpot". For instance, in a group of connected machines, the jackpot can be triggered when all connected slot machines bring the same

combination of symbols on their reels at the same time. The exciting news is that when jackpot happens, every player wins the total amount bet by all players. When analyzing the MDP of the distributed machines, the global state represent the internal states of all distributed machines, and the jackpot is a kind of synchronization when the MDP is synchronized into the set of states representing identical symbols on the reels of all machines. The aim of the designer is usually to synthesize an optimal strategy that allows for such jackpots infinitely often, and only when the amount of the total bet is smaller than a threshold.

In this thesis, we study the introduced synchronizing conditions in MDPs. We establish the computational complexity of deciding whether a given MDP is always, eventually, weakly or strongly synchronizing by providing matching upper and lower complexity bounds: for each winning mode we show that the problems are in PTIME for the always synchronizing condition, PSPACE-complete for eventually and weak synchronization, and PTIME-complete for strong synchronization. Moreover, we prove that the three winning modes coincide for the always synchronizing condition, though those modes form a strict hierarchy for eventually synchronizing. In particular, there are limit-sure winning MDPs that are not almost-sure winning; this result is unlike the analogue reachability condition in the state-based semantics where the two almost-sure and limit-sure winning modes coincide [dAHK07]. One of the difficult result in this thesis is to show that for weak and strong synchronization the almost-sure and limit-sure modes coincide. It implies that the strong and weak synchronizing conditions are more robust than the eventually synchronizing, and provide conservative approximations of the eventually synchronizing.

We complete the picture by providing optimal memory bounds for winning strategies. We will see that all solutions, for the problems of deciding whether an MDP is synchronizing, are constructive, and provide the construction of the winning strategies that answer the synthesis problems. In always synchronizing MDPs, we show that memoryless strategies are sufficient for all three winning modes. In eventually synchronizing MDPs, for sure winning strategies, exponential memory is sufficient and may be necessary, and that in general infinite memory is necessary for almost-sure winning, and unbounded memory is necessary for limit-sure winning. Moreover, exponential memory is sufficient and may be necessary for sure winning in weak synchronization, infinite memory is necessary for almost-sure winning in weak synchronization, and linear memory is sufficient for strong synchronization in all winning modes.

Regarding the symbolic-outcomes in MDPs, the p-synchronizing conditions defined on the outcome sequence are using a sum function, i.e., the sum of probabilities assigned to target states is at least p. We present a variant of synchronization for which the used function is taking the maximum probability assigned to the set of target states. We also establish some reductions for eventually and weakly synchronization such that the tight complexity bounds for sum functions are carried over to the max variant. For strong synchronization, we see that deciding whether the MDP is winning is again PTIME-complete though memoryless strategies are sufficient.

The obtained results on synchronization in MDPs are summarized in Tables 4.1, 5.1 and 6.1 on pages 67, 90 and 116, respectively.

1.3.1 Relations to games and to alternating finite automata

Some results in this thesis rely on insights related to games and alternating automata that are of independent interest. First, the sure-winning problem for eventually synchronizing in MDPS is equivalent to a two-player game with a synchronized reachability condition, where the goal for the first player is to ensure that a target state is reached after a number of steps that is independent of the strategy of the opponent (and thus this number can be fixed in advance by the first player). This condition is stronger than plain reachability, and while the winner in two-player reachability games can be decided in polynomial time, deciding the winner for synchronized reachability is PSPACE-complete. This result is obtained by turning the synchronized reachability game into a one-letter alternating automaton for which the emptiness problem (i.e., deciding if there exists a word accepted by the automaton) is PSPACE-complete [Hol95, JS07]. Second, the PSPACE lower bound for the limit-sure winning problem in eventually synchronizing uses a PSPACE-completeness result that we establish for the universal finiteness problem, which is to decide, given a one-letter alternating automata, whether from every state the accepted language is finite.

1.4 Synchronizing problems in probabilistic automata

Synchronizing problems were first considered for finite state automata where a synchronizing word is a finite sequence of control actions that can be executed from any unknown or unobservable state of an automaton and brings the automaton to a known specific state (see [Vol08] for a survey of results and applications). The notion of synchronizing automata is motivated by the following natural problem: *how can we regain control over a device if we do not know its current state?* Since losing the control over a device may happen due to missing the observation on the outputs produced by the system, *blind strategies*, which are finite sequences (or words) of input letters, are considered while synchronizing systems. As an example think of remote systems connected to a wireless controller that emits the command via wireless waves but expects the observations via physical connectors (it might be excessively expensive to mount wireless senders on the remote systems), and consider that the physical connection to controller is lost due to technical failure. The wireless controller can therefore not observe the current states of the distributed subsystems. In this setting, emitting a synchronizing word as the command leaves the remote system (as a whole) in one particular state no matter in which state each distributed subsystem started; and thus the controller can again regain control.

Synchronizing automata are well-studied in the setting of complete deterministic finite-state automata [Čer64, Pin78, Epp90, IS95, IS99, San04, Vol08, Mar10, AGV10, OU10, Mar12]. While the existence of a synchronizing word is NLOGSPACE-complete for complete deterministic finite state automata, extensive research efforts have been devoted to estab-

lishing tight bounds on the length of the shortest synchronizing word, which is conjectured to be $(n-1)^2$ for automata with n states [Čer64, Pin78, Vol08]. Various extensions of the notion of synchronizing word have been proposed for not-complete or non-deterministic finite state automata and were proved to be PSPACE-complete [IS95, IS99, Mar10, Mar12]. We introduce infinite synchronizing words for non-deterministic finite state automata, where the always, eventually, weakly and strongly synchronization require that the automaton is always, once, infinitely often and ultimately always synchronized (into a given target set).

Probabilistic automata (PAs) are a generalization of non-deterministic finite state automata where the non-deterministic choice of successors is resolved with a probabilistic distribution; at the same time, a PA can also be interpreted as an MDP with a blind controller. In this context, an input word for the PA corresponds to the special case of a blind strategy (sometimes called *word-strategy*) that chooses the control actions in advance. In the example of remote systems, a blind strategy cannot depend on the current states of the distributed subsystems, but only depend on the sequence of wireless commands emitted so far. The set of all winning word-strategies for an always, eventually, weakly and strongly synchronizing PA is respectively the always, eventually, weakly and strongly synchronizing language of the PA. Similar to the MDPs, we study three qualitative modes, sure, almost-sure and limit-sure for synchronizing languages in PAs.

A different definition of synchronizing words for probabilistic automata was proposed by Kfoury, but the associated decision problem is undecidable [Kfo70]. In contrast, we show that the emptiness problem of almost-sure strongly synchronizing languages is PSPACE-complete, while the emptiness problem of almost-sure eventually and weakly synchronizing languages are undecidable. Moreover, we show that the emptiness problem of sure languages, for all three eventually, weakly and strongly synchronizations, are PSPACE-complete, though the emptiness problem of limit-sure languages are undecidable. There are easy arguments to establish the complexity bounds for the always synchronizing languages. The obtained results are summarized in Tables 4.2, 5.2 and 6.2 on pages 81, 103 and 124, respectively.

We also shortly discuss the universality problems for the synchronizing languages in PAs: we show that the universality problems of sure and almost-sure [6] always synchronizing languages are in PTIME. The universality problems of sure eventually, weakly and strongly synchronizing languages are in PSPACE, while for almost-sure eventually, weakly and strongly synchronizing languages, the universality problems are PSPACE-complete. In addition, we show that the results from synchronizing in PAs can carry over to the non-deterministic finite state automata.

6. The universality problem of synchronizing languages in PAs cannot be considered for limit-sure winning mode.

1.5 Synchronizing problems in timed and weighted automata

A further contribution of this thesis is introducing synchronization in systems whose behaviors depend on quantitative constraints. We study two classes of such systems, timed automata and weighted automata. We introduce the synchronization problem for timed and weighted automata in several variants to include the quantitative aspects of those models as well as some safety condition while synchronizing. The main challenge is that we are facing automata with infinite state-spaces and infinite branching (e.g. delays in a timed automaton).

For timed automata the synchronization problems are shown to be PSPACE-complete in the deterministic case, and undecidable in the non-deterministic case. The obtained results are summarized in Table 7.1 on 142. In the deterministic case, synchronizing the classical region abstraction is not sound. We propose an algorithm which reduces the (uncountably) infinite set of configurations into a finite set (with at most the number of locations in the automaton), and then pairwise synchronizes the obtained finite set of states.

For weighted automata, states are composed of locations and quantitative values as weights. As weights are merely accumulated in this setting, it is impossible to synchronize to a single state. Instead we search for a *location-synchronizing word*, that is a word after which all states will agree on the location. In addition, we add a safety condition insisting that during synchronization the accumulated weight (energy) is safe, e.g. a non-negative safety condition (or energy constraints) that requires a system to never run out of power while synchronizing. Considering the safety condition is what distinguishes our setting from the one presented in [FV13]; moreover, in that work weighted automata are restricted to only have non-negative weights on transitions. For deterministic weighted automata, we provide PSPACE-completeness of the synchronization problem under energy constraints, and the membership in 3-EXPSPACE under general safety constraints. The obtained results are summarized in Table 8.1 on 158.

1.6 Structure of this manuscript

This thesis is organized as follows.

Chapter 2 *Preliminaries.* We introduce definitions needed throughout the thesis such as MDPs and PAs. We also recall some classical problems and results on such models.

The results of the following four chapters on synchronization in MDPs and PAs are mostly disseminated in these publications [DMS11a, DMS11b, DMS12, DMS14a, DMS14b]:

Chapter 3 *Synchronizing Problems.* We recall the definition of finite synchronizing words for finite state automata, and survey decision problems and conjectures about such words. We provide definitions of infinite synchronizing words for non-deterministic finite state automata. We introduce variants of synchronizations in MDPs and PAs by considering synchronizing strategies and synchronizing words. The relations to two-player games and to one-letter alternating finite automata are also discussed.

Chapter 4, Chapter 5 and Chapter 6 *Always and Eventually Synchronizing Condition*, *Weakly Synchronizing Condition* and *Strongly Synchronizing Condition.* We show tight complexity bounds of the membership problem for always, eventually, weakly and strongly synchronizing conditions in MDPs, and provide the memory requirement of winning strategies. Moreover, we establish tight complexity bounds for the emptiness problem of all synchronizing languages for PAs while the universality problem is also shortly discussed.

The results of the following two chapters on synchronization in timed and weighted automata has been published in [DJL+14]:

Chapter 7 *Synchronization in Timed Automata.* We introduce synchronizing and location-synchronizing words for timed automata. We focus on deciding the existence of a synchronizing and location-synchronizing word for timed automata, proving tight complexity bounds of the problems for deterministic timed automata, and proving undecidability for non-deterministic timed automata.

Chapter 8 *Synchronization in Weighted Automata.* We introduce synchronizing and location-synchronizing words for weighted automata. We define safe synchronization for weighted automata where the aim is to synchronize the set of safe states while forbidding the automaton to visit states outside the safety-set during synchronization. We establish complexity results for deciding the existence of synchronizing and location-synchronizing words for weighted automata under different kind of safety conditions.

Chapter 9 *Conclusion.*

Selected publications by the author Most of the contributions in this manuscript have been already published in international conferences. The following publications partially contain the results of this thesis:

[DJL+14] Laurent Doyen, Line Juhl, Kim G. Larsen, Nicolas Markey, and Mahsa Shirmohammadi. Synchronizing words for weighted and timed automata. In *Proceedings of the 34th Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2014*, Leibniz International Proceedings in Informatics. Leibniz-Zentrum für Informatik, 2014. To appear.

[DMS14b] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Robust synchronization in Markov decision processes. In *CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings*, volume 8704 of *Lecture Notes in Computer Science*, pages 234–248. Springer, 2014.

[DMS14a] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Limit synchronization in Markov decision processes. In *Foundations of Software Science and Computation Structures - 17th International Conference, FOSSACS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8412 of *Lecture Notes in Computer Science*, pages 58–72. Springer, 2014.

[DMS12] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Infinite synchronizing words for probabilistic automata (erratum). *CoRR*, abs/1206.0995, 2012.

[DMS11a] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Infinite synchronizing words for probabilistic automata. In *Mathematical Foundations of Computer Science 2011 - 36th International Symposium, MFCS 2011, Warsaw, Poland, August 22-26, 2011. Proceedings*, volume 6907 of *Lecture Notes in Computer Science*, pages 278–289. Springer, 2011.

[DMS11b] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Synchronizing objectives for Markov decision processes. In *Proceedings International Workshop on Interactions, Games and Protocols, iWIGP 2011, Saarbrücken, Germany, 27th March 2011.*, volume 50 of *EPTCS*, pages 61–75, 2011.

2 Preliminaries

First sight. In this chapter we introduce definitions and recall results needed throughout the thesis.

We first introduce finite automata, Markov decision processes and probabilistic automata. We then mention classical acceptance and winning conditions to define languages and winning regions. We conclude the chapter with the last section devoted to the classical problems and results for each model.

Contents

We denote by $\mathbb{Z}$ the set of integer numbers and by $\mathbb{N} \stackrel{def}{\equiv} \{n \geq 0 \mid n \in \mathbb{Z}\}$ the set of natural numbers. We denote by $\mathbb{R}$ the set of real numbers and by $\mathbb{R}_{\geq 0} \stackrel{def}{\equiv} \{x \geq 0 \mid x \in \mathbb{R}\}$ the set of non-negative real numbers. We assume that the reader is familiar with the basic concepts of sets, relations, graphs theories and automata theory [Ros93, Die12, Sip97].

2.1 Finite automata and words

Finite automata are defined over finite alphabets to accept input words which are sequences of concatenated letters of the alphabet. Such words could be finite, such as $w_1 = a \cdot b \cdot a \cdot b \cdot b$, or could be infinite such as $w_2 = a \cdot b \cdot b \cdot b \cdots$ where a and b are some input letters. Given an alphabet A, we denote by A^* the set of all finite words over this alphabet, and by A^ω the set of all infinite words. The length of a finite word $w \in \mathsf{A}^*$ is denoted by $|w|$; for example $|w_3| = 3$ for the prefix $w_3 = a \cdot b \cdot b$ of w_2. The empty word ϵ is of length zero, and A^+ is the set of all non-empty finite words.

Regular and ω-regular languages. A language of finite words over the alphabet A is a subset of A^* and a language of infinite words is a subset of A^ω. A language (of finite words) is *regular* if there is a regular expression to express it. Given a finite alphabet A, the following constants are defined as regular expressions:

- $\emptyset$ denotes the empty language,
- ϵ denotes the language $\{\epsilon\}$ which contains only the empty word ϵ of length zero,
- $a \in \mathsf{A}$ denotes the language $\{a\}$ containing only the word a of length 1.

Given two regular expressions r_1 and r_2, the expressions $r_1 + r_2$, $r_1 \cdot r_2$ and r_1^* are regular expressions where

- (union operator) $r_1 + r_2$ denotes the language that is obtained by the union of languages expressed by r_1 and r_2,
- (concatenation operator) $r_1 \cdot r_2$ denotes the language of all words made by concatenation of a word from the language expressed by r_1 followed by a word from r_2,
- (Kleene star operator) r_1^* denotes the smallest language that includes ϵ and r_1, and that is closed under concatenation operator.

A language (of infinite words) is *ω-regular* if there is an ω-regular expression to express it. Given a regular expression r expressing a regular language not containing the empty word ϵ, the expression r^ω that denotes a language of infinite words made by concatenating consecutively the words of r infinitely many times, is ω-regular. Given a regular expression r_1 and two ω-regular expressions r_2 and r_3, the following expressions are ω-regular:

- (concatenation operator) $r_1 \cdot r_2$ denotes the language of all infinite words made by concatenation of a finite word from the language expressed by r_1 followed by an infinite word from r_2.
- (union operator) $r_2 + r_3$ denotes the language that is obtained by the union of languages expressed by r_2 and r_3.

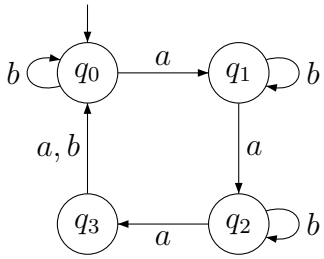

Figure 2.1: The NFA described in Example 2.2 with initial state q_0.

Example 2.1. *The regular expression $a \cdot b^* \cdot a$ is expressing the language $\{a \cdot b^n \cdot a \mid n \in \mathbb{N}\}$ where b^n is the word consists of n consecutive b. The ω-regular expression $a \cdot b^* \cdot a^\omega$ is expressing the language $\{a \cdot b^n \cdot a \cdot a \cdot a \cdots \mid n \in \mathbb{N}\}$.*

◁

Finite automata. A finite automaton recognizes a regular or an ω-regular language:

Definition 2.1 (Finite automata)**.** *A non-deterministic finite automaton (NFA) $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$ consists of a finite set Q of states, a finite alphabet A of input letters and a transition function $\Delta : Q \times \mathsf{A} \to 2^Q$.*

Given a set of initial states Q_0 and an input word $w = a_0 \cdot a_1 \cdots$, the behavior of an NFA is as follows. One of the initial states $q_0 \in Q$ is chosen non-deterministically. The automaton starts in that initial state q_0 and reads the first letter a_0 of the input word w. Inputting the letter a at the time the automaton is in q (that is letter a_0 when it initially is in q_0), two cases happen: (1) either $\Delta(q, a) \neq \emptyset$ when the automaton moves to the next state q', that is chosen non-deterministically among the states of $\Delta(q, a)$; (2) or $\Delta(q, a) = \emptyset$ when the automaton "fails" to move and becomes blocked. As long as the automaton is not blocked, in each move the next letter of $w = a_0 a_1 \cdots$, from left to right, is read.

Example 2.2. *An NFA is depicted in Figure 2.1, which is a running example used throughout the thesis. The automaton has four states q_0, q_1, q_2 and q_3, and two letters a and b. The transition function Δ is defined as follows. For all states, the a-transitions shift once the automaton in the loop $q_0 q_1 q_2 q_3$; formally,*

$$\Delta(q_i, a) = \{q_j\} \text{ where } j \equiv i + 1 \pmod 4 \text{ for all } 0 \leq i \leq 3.$$

The b-transitions in the states q_0, q_1, q_2 are self-loops, $\Delta(q_i, b) = \{q_i\}$ for all $0 \leq i < 3$; the next successor on b-transition in q_3, however, is q_0.

◁

In Example 2.2, the transition function maps each pair of state q and letter a only to sets of size at most 1, that implies this automaton is in fact *deterministic*. A deterministic

finite automaton (DFA) is a finite automaton [1] where each state q and input letter a have at most one successor state: $|\Delta(q,a)| \leq 1$. On the other hand, all states and letters of the automaton of Example 2.2 have at least one successor that means this automaton is *complete* too. A complete finite automaton is an automaton where the function $\Delta(q,a)$ is non-empty for each state q and input letter a. A state q is *absorbing* if $\Delta(q,a) = \{q\}$ for all letters $a \in \mathsf{A}$ (all transitions are self-loops on q).

We denote by $q \xrightarrow{a} q'$ an a-transition that goes from q to q': $q' \in \Delta(q,a)$. Let the relation $\rightarrow \subseteq Q \times Q$ be such that $q \rightarrow q'$ if there exists a letter a where $q \xrightarrow{a} q'$. We thus define the transitive closure $\rightarrow^+$ of this relation, that is $\cup_{1 \leq i} \rightarrow^i$ where $\rightarrow^i$ is the i-th power of $\rightarrow$ by the composition operation in relations. The notation $q \rightarrow^+ q'$ thus means that there exists a finite word $w = a_0 \cdots a_{n-1}$ such that after inputting w, the automaton initiating in q ends to move in q'. The word w, actually, induces a *finite run* over the automaton; a finite run is a sequence of states $q_0 q_1 \cdots q_n$ where $q_0 = q$, $q_n = q'$ and $q_i \xrightarrow{a_i} q_{i+1}$ for all $0 \leq i < n$. An input word w may induce several runs over a (non-deterministic) automaton; it induces however only one run over a DFA. We denote by $q \xrightarrow{w} q'$ such cases when w induces some run from q to q'. This notation can also be generalized for sets of states $S, S' \subseteq Q$: $S \xrightarrow{w} S'$ that is $S' = \{q' \in Q \mid \text{ there exists } q \in S \text{ such that } q \xrightarrow{w} q'\}$. In Example 2.2, one can verify that $q_0 \xrightarrow{b \cdot a \cdot b} q_1$, where the run is $q_0 q_0 q_1 q_1$, and $\{q_0, q_3\} \xrightarrow{b \cdot a} \{q_1\}$.

The relation $\rightarrow$ when defined as a subset of $Q \times Q$ relates q to a successor state; however, when the relation $\rightarrow$ is a subset of $2^Q \times 2^Q$ it relates $\{q\}$ to the set of all its successors:

$$q \xrightarrow{a} q' \text{ if } q' \in \Delta(q,a) \qquad \text{and} \qquad \{q\} \xrightarrow{a} S \text{ if } S = \Delta(q,a)$$

A finite automaton can also input infinite words where the words induce *infinite runs* over the automaton; an infinite run for an infinite word $w = a_0 a_1 \cdots$ is a sequence of states $q_0 q_1 \cdots$ where $q_0 \in Q_0$ and $q_i \xrightarrow{a_i} q_{i+1}$ for all $i \in \mathbb{N}$.

We refer the reader to [Sip97] and [BK08] for more details.

2.2 Markov decision processes and strategies

Probability distribution. A *probability distribution* over a finite set S is a function $d : S \to [0,1]$ such that $\sum_{s \in S} d(s) = 1$. The *support* of d is the set $\mathsf{Supp}(d) = \{s \in S \mid d(s) > 0\}$. For example, given the alphabet $\mathsf{A} = \{a, b\}$, the function $d(a) = \frac{1}{4}$ and $d(b) = \frac{3}{4}$ is a probability distribution over A; we usually denote probability distributions over an alphabet with d. Another example is the function $X(q_1) = \frac{1}{3}$, $X(q_2) = \frac{2}{3}$ and $X(q_3) = 0$ that is a probability distribution X over set $S = \{q_1, q_2, q_3\}$ of states, where $\mathsf{Supp}(X) = \{q_1, q_2\}$. In this thesis, we usually denote probability distributions over states with X. We denote by $\mathcal{D}(S)$ the set of all probability distributions over S. Given a set $T \subseteq S$ and a probability distribution X over S, let $X(T) = \sum_{s \in T} X(s)$.

For $T \neq \emptyset$, the *uniform distribution* on T assigns probability $\frac{1}{|T|}$ to every state in T. For example, the distribution $X(q_1) = X(q_2) = \frac{1}{2}$ is a uniform distribution on the set $\{q_1, q_2\}$.

1. A DFA is equipped with only one initial state q_0.

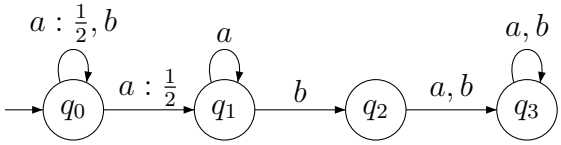

Figure 2.2: The MDP described in Example 2.3 with the initial Dirac distribution q_0.

Given $s \in S$, the *Dirac distribution* on s assigns probability 1 to s, and by a slight abuse of notation, we denote it simply by s.

2.2.1 Markov decision processes

Markov decision processes are a kind of stochastic games:

Definition 2.2 (Markov decision processes)**.** *A Markov decision process (MDP)* $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ *consists of a finite set* Q *of states, a finite set* A *of actions and a probabilistic transition function* $\delta : Q \times \mathsf{A} \to \mathcal{D}(Q)$.

Given an initial distribution $X_0 \in \mathcal{D}(Q)$, the behavior of an MDP is described as a one-player stochastic game played in rounds. The game starts in the state q with probability $X_0(q)$. In all rounds, the player chooses an action $a \in \mathsf{A}$, and if the game is in state q, the next round starts in the successor state q' with probability $\delta(q, a)(q')$.

Example 2.3. *An MDP is depicted in Figure 2.2 where the states are* $Q = \{q_0, q_1, q_2, q_3\}$ *and the alphabet is* $\mathsf{A} = \{a, b\}$*. The probabilistic transition function* δ *is defined as follows. Both* a*-transitions and* b*-transitions in* q_1, q_2, q_3 *are* deterministic*, that is* $\delta(q_i, a)$ *and* $\delta(q_i, b)$ *are Dirac distributions: for* $0 < i \leq 3$*,*

$$\delta(q_1, a) = q_1, \quad \delta(q_1, b) = q_2 \quad \text{and} \quad \delta(q_2, c) = \delta(q_3, c) = q_3 \text{ for } c \in \{a, b\}.$$

The only probabilistic transition is the a*-transition in* q_0 *where* $\delta(q_0, a)$ *is a uniform distribution on the set* $\{q_0, q_1\}$*. The* b*-transition in* q_0 *is a deterministic self-loop:* $\delta(q_0, b) = q_0$*.*

$\triangleleft$

In Example 2.3, we say that the state q_3 is *absorbing* since $\delta(q_3, c)$ is the Dirac distribution on q_3 for all actions $c \in \mathsf{A}$. Given $q \in Q$ and $a \in \mathsf{A}$, denote by $\mathsf{post}(q, a)$ the set $\mathsf{Supp}(\delta(q, a))$ which contains all states that can be chosen with a positive probability as the next successor of the a-transition in q. In Example 2.3, $\mathsf{post}(q_0, a) = \{q_0, q_1\}$. Let $S \subseteq Q$ and $\mathsf{A}' \subseteq \mathsf{A}$. We denote by $\mathsf{post}(S, a)$ the set of successors $\cup_{q \in S} \mathsf{post}(q, a)$ of all states q in S and the action a; and similarly by $\mathsf{post}(S, \mathsf{A}')$ the set of all successors $\cup_{a \in \mathsf{A}'} \mathsf{post}(S, a)$ of states in S and actions in A'.

Given $T \subseteq Q$, let $\mathsf{Pre}(T) = \{q \in Q \mid \exists a \in \mathsf{A} : \mathsf{post}(q, a) \subseteq T\}$ be the set of states from which the player has an action to ensure that the successor state is in T. For $k > 0$, let $\mathsf{Pre}^k(T) = \mathsf{Pre}(\mathsf{Pre}^{k-1}(T))$ with $\mathsf{Pre}^0(T) = T$. In Example 2.3, for $T = \{q_0, q_3\}$ one

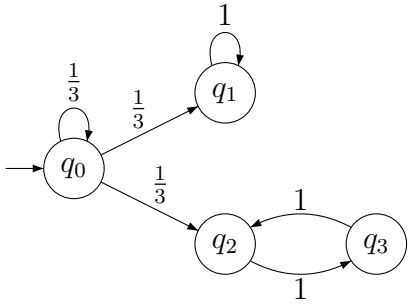

Figure 2.3: The Markov chain of Example 2.4 with the initial Dirac distribution q_0.

can verify that $\mathsf{Pre}(T) = \{q_0, q_2, q_3\}$ and $\mathsf{Pre}^2(T) = \{q_0, q_1, q_2, q_3\}$. Moreover, $\mathsf{Pre}^k(T) = \mathsf{Pre}^2(T)$ for all $k \geq 2$ showing that the sequence of predecessors is ultimately periodic for T. Since the set of states Q is finite, the sequence of predecessors $\mathsf{Pre}^0(T), \mathsf{Pre}^1(T), \mathsf{Pre}^2(T), \cdots$ is, indeed, ultimately periodic for all sets $T \subseteq Q$.

A *play* (or simply a path) in an MDP $\mathcal{M}$ is an infinite sequence $\pi = q_0 q_1 q_2 \dots$ such that for all $i \geq 0$ there exists an action a such that $q_{i+1} \in \mathsf{post}(q_i, a)$. A finite prefix $\rho = q_0 q_1 \dots q_n$ of a path has length $|\rho| = n$ and last state $\mathsf{Last}(\rho) = q_n$. In Example 2.3, the play $\pi = q_0 q_1 q_2 (q_3)^\omega$ has the prefix $\rho = q_0 q_1 q_2$ where $|\rho| = 2$ and $\mathsf{Last}(\rho) = q_2$. We denote by $\mathsf{Plays}(\mathcal{M})$ and $\mathsf{Pref}(\mathcal{M})$ the set of all plays and finite paths in $\mathcal{M}$, respectively.

2.2.2 Markov chains

A finite-state *Markov chain* is a special kind of MDPs where the player's choice has no role.

Definition 2.3 (Markov chains)**.** *A finite-state Markov chain $\mathcal{M} = \langle Q, \delta \rangle$ consists of a finite set Q of states and a probabilistic transition function $\delta : Q \to \mathcal{D}(Q)$.*

We see that the probability $\delta(q)(q')$ of moving from one state q to the successor q' is fixed independently of the actions and the states that the Markov chain went through. The Markov chains defined in Definition 2.3 are sometimes referred as homogeneous (discrete time) Markov chains [KS83, Nor98, Ser13]. In this manuscript, we only consider the homogeneous Markov chains and thus omit the word homogeneous as a qualifier.

Example 2.4. *A Markov chains is depicted in Figure 2.3 where the states are $Q = \{q_0, q_1, q_2, q_3\}$ and the probabilistic transition function δ is defined as follows. The next successor in q_0 is either q_0, q_1 or q_2, each with probability $\frac{1}{3}$. The state q_1 is an absorbing state: $\delta(q_1)(q_1) = 1$. The transitions in q_2 and q_3 are deterministic: $\delta(q_2)(q_3) = 1$ and $\delta(q_3)(q_2) = 1$.*

$\triangleleft$

Since Markov chains are a subset of MDPs, we analogously define paths and finite prefixes here.

The states of finite Markov chains can be classified in *recurrent* and *transient* states; for that, we define a *bidirectional reachability relation* $\sim$ on their state space Q: given the Markov chain $\mathcal{M}$, we say that q and q' are related, denoted by $q \sim q'$, if there are two finite paths in $\mathcal{M}$, one from q to q' and another from q' to q. In Example 2.4, we see that $q_0 \sim q_0$, $q_2 \sim q_3$ and $q_1 \not\sim q_2$. The relation $\sim$ is an equivalence relation since it is reflexive, symmetric and transitive. We denote by $[Q]_\sim$ the partition (equivalence classes) induced by the relation $\sim$. For the Markov chain described in Example 2.4, the equivalence classes are $\{q_0\}, \{q_1\}$ and $\{q_2, q_3\}$. Intuitively, each of these classes is a strongly connected component (SCC) in the digraph induced by the Markov chain. A state q is recurrent if from q there is no path to a state outside the class containing q; formally, a state $q \in c$ where $c \in [Q]_\sim$ is recurrent if there is no finite path to a state q' where $q' \notin c$. Otherwise the state q is transient. The definition results in the states of a class being all recurrent or all transient. The Markov chain of Example 2.4 has two recurrent classes $\{q_1\}$ and $\{q_2, q_3\}$ and one transient class $\{q_0\}$. In the graph view, a recurrent class is a bottom SCC in the digraph induced by the Markov chain.

Roughly speaking, from a recurrent state q there is no possibility of going to a state q' from which there can be no return to q. Therefore, if a Markov chain ever enters in the state q of a recurrent class c, it always stays in c and eventually returns to q with probability 1, and thus keeps returning in q infinitely often. To formalize this observation, let $\mathcal{M}$ be a Markov chain and X_0 be an initial distribution; for technical reason, we assume that for all recurrent states q, there is a path from some initial state $q_0 \in \mathsf{Supp}(X_0)$ to q (otherwise we omit the recurrent state q). For all $n \in \mathbb{N}$, let the probability distribution $\mathcal{M}_n$ over the states of the Markov chain $\mathcal{M}$ give the probability of being in various states at step n. Thus, $\mathcal{M}_0 = X_0$ and $\mathcal{M}_{n+1}(q) = \sum_{q' \in Q} \mathcal{M}_n(q') \cdot \delta(q')(q)$ for all $n \in \mathbb{N}$. Considering the sequence $\mathcal{M}_0\mathcal{M}_1\mathcal{M}_2 \cdots$ in a Markov chain $\mathcal{M}$, a distribution $X \in \mathcal{D}(Q)$ is *stationary* if there is an infinite sequence $n_1 n_2 n_3 \cdots$ such that $\lim_{k \to \infty} \mathcal{M}_{n_k}$ exists and is equal to X. It is known that all finite homogeneous (discrete time) Markov chains always have a non-empty set of stationary distributions [Nor98]. Given a Markov chain with m stationary distributions $X_1, \cdots, X_m$, let $S_i = \mathsf{Supp}(X_i)$ be the support of the stationary distribution X_i where $1 \leq i \leq m$. For all recurrent states q, there exists at least one support S_i such that $q \in S_i$; and moreover, none of the supports S_i contains some transient state. We refer to the sets $S_1, \cdots, S_m$ as *periodic supports of recurrent states*. For instance, the periodic supports of recurrent states for the Markov chain of Example 2.4 are $S_1 = \{q_1, q_2\}$ and $S_2 = \{q_1, q_3\}$. As a result, the probability $\mathcal{M}_n(q)$ in all transient states q vanishes when $n \to \infty$, and all recurrent states are visited infinitely often with probability 1.

2.2.3 Strategies

For an MDP, a strategy is a recipe (a program) that determines the player's choice of action in all different situations of the game played on the MDP. The strategy observes the finite sequence of states visited so far in the game and suggests the player a probability distribution over a designated set of good actions.

Definition 2.4 (Strategies)**.** *A randomized strategy for an MDP $\mathcal{M}$ (or simply a strategy) is a function $\alpha : \mathsf{Pref}(\mathcal{M}) \to \mathcal{D}(\mathsf{A})$ that, given a finite path ρ, returns a probability distribution $\alpha(\rho)$ over the action set, used to select a successor state q' of ρ with probability $\sum_{a \in \mathsf{A}} \alpha(\rho)(a) \cdot \delta(q, a)(q')$ where $q = \mathsf{Last}(\rho)$.*

A strategy α is *pure* if for all $\rho \in \mathsf{Pref}(\mathcal{M})$, there exists an action $a \in \mathsf{A}$ such that $\alpha(\rho)(a) = 1$; and *memoryless* if $\alpha(\rho) = \alpha(\rho')$ for all ρ, ρ' such that $\mathsf{Last}(\rho) = \mathsf{Last}(\rho')$. We view pure strategies as functions $\alpha : \mathsf{Pref}(\mathcal{M}) \to \mathsf{A}$, and memoryless strategies as functions $\alpha : Q \to \mathcal{D}(\mathsf{A})$.

Example 2.5. *Let $\mathcal{M}$ be the MDP described in Example 2.3. The strategy α is a randomized strategy for $\mathcal{M}$, that is, given a prefix $\rho = p_0 p_1 \cdots p_n$ in $\mathsf{Pref}(\mathcal{M})$, defined by*

$$\alpha(\rho) = \begin{cases} \text{the Dirac distribution on } a & \text{if } n < 4 \text{ and } p_n \in \{q_0, q_1\} \\ \text{the uniform distribution on } \{a, b\} & \text{if } n < 4 \text{ and } p_n \in \{q_2, q_3\} \\ \text{the Dirac distribution on } b & \text{otherwise,} \end{cases}$$

The strategy α purely plays a for the first four rounds, if the MDP is in q_0 or q_1; otherwise in those rounds, it plays either a or b, each with probability $\frac{1}{2}$. For all next rounds, α purely plays b. The strategy β, defined for a given prefix $\rho \in \mathsf{Pref}(\mathcal{M})$ by

$$\beta(\rho) = \begin{cases} a & \text{if } n < 4, \\ b & \text{otherwise,} \end{cases}$$

is a pure strategy for $\mathcal{M}$; and γ is a memoryless strategy where $\gamma(q_0) = a$ and $\gamma(q_i) = b$ for all $0 < i \leq 3$.

$\triangleleft$

A strategy α uses *finite memory* if it can be represented by a finite-state transducer $T = \langle \mathsf{Memory}, m_0, \alpha_u, \alpha_n \rangle$ where Memory is a finite set of modes (the memory of the strategy), $m_0 \in \mathsf{Memory}$ is the initial mode, $\alpha_u : \mathsf{Memory} \times Q \to \mathsf{Memory}$ is an update function, that given the current memory and last state updates the memory, and $\alpha_n : \mathsf{Memory} \times Q \to \mathcal{D}(\mathsf{A})$ is a next-move function that selects the probability distribution $\alpha_n(m, q)$ over actions when the current mode is m and the current state of $\mathcal{M}$ is q. For pure strategies, we assume that $\alpha_n : \mathsf{Memory} \times Q \to \mathsf{A}$. Formally, the strategy α defined by T is such that $\alpha(\rho \cdot q) = \alpha_n(\hat{\alpha}_u(m_0, \rho), q)$ for all $\rho \in \mathsf{Pref}(\mathcal{M})$ and $q \in Q$, where $\hat{\alpha}_u$ extends α_u to finite paths in the usual way. The *memory size* of the strategy is the number $|\mathsf{Memory}|$ of modes.

For an MDP $\mathcal{M}$ and a finite-memory strategy α represented by the finite-state transducer $T = \langle \mathsf{Memory}, m_0, \alpha_u, \alpha_n \rangle$, we denote by $\mathcal{M}(\alpha) = \langle Q', \delta' \rangle$ the Markov chain obtained as the product of $\mathcal{M}$ with the transducer defining α where $Q' = \mathsf{Memory} \times Q$ and a transition from the state $\langle m, q \rangle$ to a successor $\langle m', q' \rangle$ with $m' = \alpha_u(m, q)$ has probability $\sum_{a \in \mathsf{A}} \alpha_n(m, q)(a) \cdot \delta(q, a)(q')$.

For instance, we see that the strategy α described in Example 2.5 is actually a finite-memory strategy that can be represented by the transducer $\langle \mathsf{Memory}, m_0, \alpha_u, \alpha_n \rangle$ where $\mathsf{Memory} = \{m_0, m_1, \cdots, m_4\}$. The update function is $\alpha_u(m_4, q) = m_4$ and $\alpha_u(m_i, q) = m_{i+1}$ for all $0 \leq i < 4$ and states $q \in Q$. The next-move function α_n is defined as follows.

$$\alpha_n(m_i, q) = \begin{cases} \text{the Dirac distribution on } a & \text{if } 0 \leq i < 4 \text{ and } q \in \{q_0, q_1\} \\ \text{the uniform distribution on } \{a, b\} & \text{if } 0 \leq i < 4 \text{ and } q \in \{q_2, q_3\} \\ \text{the Dirac distribution on } b & \text{otherwise.} \end{cases}$$

For the finite-memory strategy α, the obtained Markov chain is $\mathcal{M}(\alpha) = \langle Q', \delta' \rangle$ where $Q' = \{m_0, \cdots, m_4\} \times \{q_0, \cdots, q_3\}$ and the transition function is defined as follows. The transitions in $\langle m_i, q_0 \rangle$ is a uniform distribution on $\{\langle m_{i+1}, q_0 \rangle, \langle m_{i+1}, q_1 \rangle\}$ for all $0 \leq i < 4$ and the transition in $\langle m_4, q_0 \rangle$ is a self-loop. The transitions in $\langle m_i, q_1 \rangle$ are deterministic and go to $\langle m_{i+1}, q_1 \rangle$ for all $0 \leq i < 4$ while the transition in $\langle m_4, q_1 \rangle$ is the Dirac distribution on $\langle m_4, q_2 \rangle$. Finally, for all modes m_i, the transitions in two states $\langle m_i, q_2 \rangle$ and $\langle m_i, q_3 \rangle$ are directed to the same state: to $\langle m_{i+1}, q_3 \rangle$ if $0 \leq i < 4$, and to $\langle m_4, q_3 \rangle$ otherwise.

2.3 Probabilistic Automata and randomized word

A natural extension for non-deterministic automata are *probabilistic automata* where the non-deterministic choice of successors for a pair of states and actions is resolved by randomness.

Reactive systems, by nature, respond to environmental actions. Finite-state automata and probabilistic automata are used to model a kind of reactive systems where the reactions to the environment are fixed by a controller before the system execution. An automaton inputs an infinite word where the letters are, one-by-one, read as the controller's choice among possible reactions. Observing the system execution by controllers is a way to enhance modeling reactive systems; in this case, the controller provides a strategy that is able to choose an action (or a letter) according to the sequence of states visited along the system execution. Considering the power that controller gains by observing the execution, MDPs can be viewed as a generalization of probabilistic automata. An infinite word can be viewed as a pure *blind strategy* $\alpha : \mathbb{N} \to \mathsf{A}$ that chooses actions (or letters) independently of any observation on the system execution but only depending on the number of previous actions that it has chosen so far. We sometimes consider *randomized words* that are sequences $d_0 d_1 d_2 \cdots$ of probability distributions $d_i \in \mathcal{D}(\mathsf{A})$ over the letters, i.e., randomized blind strategies $\alpha : \mathbb{N} \to \mathcal{D}(\mathsf{A})$.

Definition 2.5 (Probabilistic automata)**.** *A probabilistic automaton (PA) $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ consists of a finite set Q of states, a finite alphabet A of input letters and a probabilistic transition function $\delta : Q \times \mathsf{A} \to \mathcal{D}(Q)$.*

As we may compare the definitions 2.2 and 2.5, the syntax of PAs and MDPs are the same, but the semantics (the behaviors) are different. Given an initial distribution

$X_0 \in \mathcal{D}(Q)$, the behavior of a PA is as follows. The automaton starts in the state q with probability $X_0(q)$ and reads the first letter of the input pure word w (or if the input word $w = d_0 d_1 d_2 \cdots$ is randomized, it reads the letter a that is chosen among $a \in \mathsf{Supp}(d_0)$ with probability $d_0(a)$). The automaton moves to the next state q' that is chosen with probability $\delta(q, a)(q')$ if the automaton is in the state q and the inputted letter is a. Then, automaton reads the next letter (or the next probability distribution) of w.

Since PAs are an extension of NFAs, and since input infinite words for PAs are indeed blind strategies for MDPs, all definitions such as runs and prefixes are analogously defined for PAs.

Example 2.6. *Consider the automaton depicted in Figure 2.2 on page 21 as an instance of a PA. This automaton is described in Example 2.3 as an MDP. Let $w = (a \cdot b)^\omega$ be an input pure word that can be described as the pure blind strategy α as follows. For all prefixes ρ with length n,*

$$\alpha(\rho) = \begin{cases} a & \text{if } n \equiv 0 \pmod 2, \\ b & \text{if } n \equiv 1 \pmod 2. \end{cases}$$

The word w results in the following set of infinite runs over the PA: $q_0^\omega + q_0(q_0q_0)^* q_1 q_2 (q_3)^\omega$.

$\triangleleft$

2.4 Winning and acceptance conditions

A *Borel* set [2] $\varphi \subseteq Q^\omega$ is a set of infinite sequences of states (an infinite run or a play) in the Cantor topology on Q^ω. For an infinite sequence of states $\pi = q_0 q_1 q_2 \cdots$, define $\mathsf{Inf}(\pi) = \{q \in Q \mid \text{ for all } i \geq 0 \text{ there exists } j \geq i \text{ such that } q_j = q\}$ which is the set of states visited infinitely often along π. An important subclass of the Borel sets are the ω-regular ones, that are ω-regular languages over Q. We mention reachability, safety, Büchi and coBüchi condition from this subclass:

Reachability and safety conditions. A reachability set $T \subseteq Q$ for a system requires that some target state in T be visited along the system execution. The set $\Diamond T = \{q_0 q_1 q_2 \cdots \in Q^\omega \mid \text{there exists } i \geq 0 \text{ such that } q_i \in T\}$ is, thus, a set of winning plays or accepting infinite runs for the system with the reachability condition on the target set T. In the same way, a safety set $T \subseteq Q$ for a system requires that only safe states in T be visited along the system execution. The set $\Box T = \{q_0 q_1 q_2 \cdots \in Q^\omega \mid \text{for all } i \geq 0 \text{ we have } q_i \in T\}$ is, thus, a set of winning plays or accepting infinite runs for the system with the safety condition on T.

2. Generally speaking, a Borel set is any set in a topological space that is built on open sets (or equivalently on closed sets) by countable unions, countable intersections and relative complement. We refer the reader to [Mar90, Mar98] for more details.

Büchi and coBüchi conditions. A Büchi set $T \subseteq Q$ for a system requires that some state in T be visited along the system execution infinitely often. The set $\Box\Diamond T = \{\pi \in Q^\omega \mid \mathsf{Inf}(\pi) \cap T \neq \emptyset\}$ is, thus, a set of winning plays or accepting infinite runs for the system with the Büchi condition on the target set T. In the same way, a coBüchi set $T \subseteq Q$ for a system requires that only states in T be visited infinitely often along the system execution; all states outside T can only be visited finitely many times. The set $\Diamond\Box T = \{\pi \in Q^\omega \mid \mathsf{Inf}(\pi) \subseteq T\}$ is, thus, a set of winning plays or accepting infinite runs for the system with the coBüchi condition on T.

Note that the set of winning plays of safety and reachability conditions are dual, that is $\Diamond T = Q^\omega \setminus \Box(Q \setminus T)$; the same holds for the set of winning plays of Büchi and coBüchi conditions: $\Box\Diamond T = Q^\omega \setminus \Diamond\Box(Q \setminus T)$.

Example 2.7. *Consider the target set $T_1 = \{q_2\}$ and $T_2 = \{q_1, q_2\}$ for the MDP $\mathcal{M}$ in Example 2.3 on page 21. The set of winning plays (or accepting runs if $\mathcal{M}$ behaves as a PA) is $\Diamond T_1 = Q^* \cdot q_2 \cdot Q^\omega$ and is $\Box T_1 = q_2^\omega$ for reachability and safety condition on T_1, respectively. As a result, the set of infinite plays in $\mathcal{M}$ that achieves the reachability condition $\Diamond T_1$ is $q_0^* q_1^* q_2 (q_3)^\omega$. This set is an empty set for the safety, Büchi and coBüchi conditions. Similarly, the set of plays in $\mathcal{M}$ that win $\Diamond T_2$ is $q_0^*(q_1)^\omega + q_0^* q_1^* q_2 (q_3)^\omega$ and it is empty for the safety condition. The sets of plays in $\mathcal{M}$ that achieves Büchi condition $\Box\Diamond T_2$ and coBüchi condition $\Diamond\Box T_2$ is the same set: $q_0^*(q_1)^\omega$.*

$\triangleleft$

2.4.1 Languages of NFAs

A regular language (a subset of $\mathcal{L} \in \mathsf{A}^*$) is recognizable by a finite automaton $\mathcal{N}$ defined over the alphabet A and augmented with an initial set Q_0 of states and a *finite acceptance condition* $\mathcal{F} \subseteq Q$, if for all finite words w of the language, there is an accepting run over $\mathcal{N}$. A finite run $q_0 q_1 \cdots q_n$ is *accepting* for the automaton $\mathcal{N}$ if it starts in some state $q_0 \in Q_0$ of the initial set and if the last state of the run visits the acceptance conditions $q_n \in \mathcal{F}$; the regular language of the automaton is, thus,

$$\mathcal{L}(\mathcal{N}) = \{w \in \mathsf{A}^* \mid \text{there exists } q_0, q \text{ such that } q_0 \xrightarrow{w} q \text{ where } q_0 \in Q_0 \text{ and } q \in \mathcal{F}\}.$$

An ω-regular language (a subset of $\mathcal{L} \in \mathsf{A}^\omega$) is recognizable by a finite automaton $\mathcal{N}$ defined over the alphabet A and augmented with an initial set Q_0 of states and an ω-regular acceptance condition with the set Ω of accepting runs (acceptance conditions such as reachability with $\Omega = \Diamond T$ and Büchi condition with $\Omega = \Box\Diamond T$) if for all infinite words w of the language, there is an accepting infinite run in $\mathcal{N}$. Thus,

$$\mathcal{L}_\Omega(\mathcal{N}) = \{w \in \mathsf{A}^\omega \mid w \text{ induces an infinite run } q_0 q_1 \cdots \text{ such that } q_0 \in Q_0 \text{ and } q_0 q_1 \cdots \in \Omega\}.$$

Example 2.8. *Let q_0 be the unique initial state of the automaton described in Example 2.2 on page 19. For the finite acceptance condition $\mathcal{F} = \{q_0\}$, we have*

$$\mathcal{L}(\mathcal{N}) = b^* \cdot (a \cdot b^* \cdot a \cdot b^* \cdot a \cdot (a + b) \cdot b^*)^*$$

that we name $\mathcal{L}_1$. Let $T = \{q_0\}$ be the target set as well, then $\mathcal{L}_{\Box T}(\mathcal{N}) = b^\omega$ that we name $\mathcal{L}_2$. The language for the coBüchi condition of T is $\mathcal{L}_{\Diamond\Box T}(\mathcal{N}) = \mathcal{L}_1 \cdot \mathcal{L}_2$.

◁

A Finite automaton with a Büchi or coBüchi acceptance condition is said to be a *Büchi* or *coBüchi* automata, respectively.

2.4.2 Path-outcomes of MDPs

Given an initial distribution $X_0 \in \mathcal{D}(Q)$ and a strategy α in an MDP $\mathcal{M}$, a *path-outcome* is an infinite path $\pi = q_0 q_1 \dots$ in $\mathcal{M}$ such that $q_0 \in \mathsf{Supp}(X_0)$ and for all $i \geq 0$, the state $q_{i+1} \in \mathsf{post}(q_i, a)$ for some action $a \in \mathsf{Supp}(\alpha(q_0 \dots q_i))$. The probability of a finite prefix $\rho = q_0 q_1 \dots q_n$ of the play π is

$$X_0(q_0) \cdot \prod_{j=0}^{n-1} \sum_{a \in \mathsf{A}} \alpha(q_0 \dots q_j)(a) \cdot \delta(q_j, a)(q_{j+1}).$$

We denote by $\mathit{Outcomes}(X_0, \alpha)$ the set of all path-outcomes from X_0 under strategy α. As an example, let X_0 be the Dirac distribution on q_0 for the MDP $\mathcal{M}$ in Example 2.3. Consider the pure memoryless strategy γ that plays a in q_0, and that plays b in all states $q \neq q_0$ (presented in Example 2.5). Thus, $\mathit{Outcomes}(q_0, \gamma) = (q_0)^\omega + q_0^* q_0 q_1 q_2 (q_3)^\omega$, and the probability of the finite prefix $q_0 q_0$ is $\frac{1}{4}$ and the probability of $q_0 q_0 q_0 q_1 q_2$ is $\frac{1}{8}$.

The winning sets Ω of plays for the ω-regular conditions (the set Ω is also called an event in this context) are measurable sets of paths, and thus given an initial distribution X_0 and a strategy α, the probabilities $\mathsf{Pr}^\alpha(\Omega)$ of events Ω are uniquely defined [Var85]. Hence, $\mathsf{Pr}^\alpha(\Diamond T)$ is the probability to reach T under strategy α. For the MDP $\mathcal{M}$ of Example 2.3, the Dirac initial distribution q_0 and the strategy γ: the probability of reaching q_2 is $\mathsf{Pr}^\gamma(\Diamond\{q_2\}) = 1$ whereas the probability of the safety condition $\mathsf{Pr}^\gamma(\Box\{q_0\}) = 0$.

Definition 2.6 (Winning modes for the ω-regular conditions)**.** *Given an initial distribution X_0 and an event Ω, we say that an MDP $\mathcal{M}$ is:*

- sure winning *if there exists a strategy α such that all path-outcomes are winning: $\mathit{Outcomes}(X_0, \alpha) \subseteq \Omega$;*
- almost-sure winning *if there exists a strategy α such that $\mathsf{Pr}^\alpha(\Omega) = 1$;*
- limit-sure winning *if* $\sup_\alpha \mathsf{Pr}^\alpha(\Omega) = 1$.

For the MDP $\mathcal{M}$ of Example 2.3, even though $\mathsf{Pr}^\gamma(\Diamond\{q_2\}) = 1$ (from q_0) implies that $\mathcal{M}$ is almost-sure winning for the reachability condition $\Diamond\{q_2\}$, the infinite play $(q_0)^\omega$ is

a counterexample witness proving that $\mathcal{M}$ is not sure-winning for this condition. The definitions result in $\mathcal{M}$ being limit-sure winning for the reachability condition $\Diamond\{q_2\}$ since $\mathcal{M}$ is almost-sure winning for that condition. Actually, almost-sure and limit-sure winning modes coincide for reachability in MDPs [dAHK07].

2.4.3 Languages of PAs

The classical languages of PAs are defined on pure words: since pure words for probabilistic automata can be viewed as pure blind strategies for MDPs, the set of runs of a finite word in a PA define a measurable event. Given an initial distribution X_0 for a PA $\mathcal{P}$, the probability that a finite run $\rho = q_0 q_1 \cdots q_n$ of a finite pure word $w = a_0 \cdot a_1 \cdots a_{n-1}$ in $\mathcal{P}$ happens, is

$$\mathsf{Pr}^w(\rho) = X_0(q_0) \cdot \prod_{j=0}^{n-1} \delta(q_j, a_j)(q_{j+1}).$$

For the finite acceptance condition $\mathcal{F}$, the probability $\mathsf{Pr}(w)$ of the word w to be accepted is thus $\sum_{\rho \in \mathsf{Runs}(w,\mathcal{F})} \mathsf{Pr}^w(\rho)$ where $\mathsf{Runs}(w, \mathcal{F})$ is the set of accepting runs over the finite word w in $\mathcal{P}$; recall that a run $\rho = q_0 q_1 \cdots q_n$ is accepting if $q_n \in \mathcal{F}$. Given a threshold $0 \leq \lambda \leq 1$, the language of a PA $\mathcal{P}$ over the finite words is

$$\mathcal{L}(\mathcal{P}) = \{w \in \mathsf{A}^* \mid \mathsf{Pr}(w) \geq \lambda\}.$$

For infinite words, we follow all definitions presented for the MDPs. Given an initial distribution $X_0 \in \mathcal{D}(Q)$ and the infinite word w in the PA $\mathcal{P}$, the set of infinite runs over w in $\mathcal{P}$ is $\mathit{Outcomes}(X_0, w)$ in the MDP $\mathcal{M}$ where the behavior of $\mathcal{P}$ is interpreted as an MDP. This way, the probabilities $\mathsf{Pr}^w(\Omega)$ of ω-regular acceptance conditions with the set Ω of accepting runs are uniquely defined [Var85]. The ω-language

$$\mathcal{L}_\Omega(\mathcal{P}) = \{w \in \mathsf{A}^\omega \mid \mathsf{Pr}^w(\Omega) > 0\}.$$

is recognizable by a PA $\mathcal{P}$ augmented with the initial distribution X_0 and the ω-regular acceptance condition with the set Ω of accepting runs.

2.5 Decision Problems

2.5.1 Problems for NFAs

We recall two classical decision problems for NFAs.

Decision problem(s). The *emptiness problem* asks, given an NFA $\mathcal{N}$ with an acceptance condition, whether the accepted language of $\mathcal{N}$ is empty; and *universality problem* asks whether the accepted language is A^* for regular languages or is A^ω for ω-regular languages.

Finite acceptance condition. The emptiness problem for NFAs with finite acceptance conditions $\mathcal{F}$ is reducible to $s-t$ *connectivity* problem in digraphs, that can be determined in NLOGSPACE in the size of the graph (and thus the automaton) [SVW87, Sip97]. To decide the emptiness problem, while performing a depth-first search algorithm that visits all *accessible* states for which there is a directed path from some initial states, one can check whether one of those accessible states is in $\mathcal{F}$.

The universality problem for an NFA $\mathcal{N}$ is PSPACE-complete [IRS76]. The proof of the PSPACE-hardness is by a reduction from the *space-bounded tiling problem* that is known to be PSPACE-complete; and since co-PSPACE is equal to PSPACE, the (N)PSPACE upper bound follows from a non-deterministic algorithm that decides co-universal problem: guess one-by-one letters of a finite word w (with $|w| \leq 2^n$ where n is the number of states in $\mathcal{N}$), and check whether w is not accepted by the automaton. The algorithm uses polynomial space to keep track of visited states along the runs induced by inputting w to see if the last state of some of those runs do not visit $\mathcal{F}$.

ω-regular acceptance condition. The emptiness problem for an NFA $\mathcal{N}$ with a reachability condition $\Diamond T$ reduces to the same problem with the Büchi condition $\Box\Diamond T$ where the states $q \in T$ are transformed to absorbing states. On the other hand, safety and coBüchi conditions are accordingly duals of reachability and Büchi conditions, thus we only mention the complexity of classical decision problems for NFA with a Büchi acceptance condition.

The emptiness problem for NFA with a Büchi condition $\Box\Diamond T$ is solved by means of graph algorithms. Such algorithms first run a SCC decomposition, and then see if there is a directed path from an initial state to some SCC that contains at least one cycle and that contains at least one state $q \in T$. It is proved that the emptiness problem for Büchi automata is NLOGSPACE-complete. The universality problem for Büchi automata is though PSPACE-complete [SVW87].

2.5.2 Problems for MDPs

We recall following decision problems for MDPs.

Decision problem(s). The *membership problem* of {sure, almost-sure, limit-sure} modes for ω-regular winning condition Ω asks, given an MDP $\mathcal{M}$ and given an initial distribution X_0, whether $\mathcal{M}$ is {sure, almost-sure, limits-sure} winning for Ω from X_0, respectively.

For the MDP $\mathcal{M}$ and the winning condition Ω, the set of all initial distributions from which $\mathcal{M}$ is winning, is the *winning region.* It is known that for reachability, safety, Büchi and coBüchi winning conditions, and for all three winning modes, the membership problems are decidable in PTIME [dAH00, dAHK07, CH12]. For all these winning conditions, it is shown that the randomization is not necessary. Moreover, for all distributions X_0 in the winning region of all these conditions, all Dirac distributions on all the states $q \in \mathsf{Supp}(X_0)$

are winning too. We say that such states are *winning states*. Moreover for reachability and safety conditions, it is known that memoryless strategies are sufficient for all three winning modes.

The membership problems of almost-sure and limit-sure modes for the winning conditions in MDPs, are mostly studied as *qualitative analysis of MDPs* [BK98b, BK08, CDH10, CHJS13]. In the sequel, we give some hints on how to decide the membership problem for the different winning modes of all reachability, safety, Büchi and coBüchi winning conditions.

Reachability condition. Let $\mathcal{M}$ be an MDP with state space Q and let $T \subseteq Q$. The membership problem of sure winning for reachability condition $\Diamond T$ is reducible to reachability in a *turn-based two-player game*, where the probabilistic choice of the environment is simulated by an *adversary* called Player-2 (since the exact values of transitions do not matter). Initially, Player-2 chooses a state $q \in \mathsf{Supp}(X_0)$ where the game starts in. For each round, Player-1 chooses an action a; and then the adversary chooses a successor state among the states in $\mathsf{post}(q, a)$. The problem of deciding whether Player-1 has a strategy to ensure reaching a target set T, is known to be PTIME-complete [CH12]. The following fix-point computation is a naive algorithm to compute the set of winning states for sure reachability. For all $n \geq 1$, compute $\mathsf{win}_n = \mathsf{Pre}(\mathsf{win}_{n-1}) \cup \mathsf{win}_{n-1}$ where $\mathsf{win}_0 = T$. We see that $\mathsf{win}_0 \subseteq \mathsf{win}_1 \subseteq \mathsf{win}_2 \subseteq \cdots$ and since Q is finite (and all $\mathsf{win}_i \subseteq Q$) then this sequence will stabilize (in at most $|Q|$ iterations) into a set win that is the set of winning states.

The membership problem for almost-sure winning is more tricky than what it is for sure winning mode, see [dAH00, dAHK07, CH12]. We give few hints on how the set of winning states is computed for almost-sure reachability condition $\Diamond T$ in an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$. The computation of winning states for almost-sure reachability condition has the solution of *positive reachability* in the main core: positive reachability problem asks whether the probability $\mathsf{Pr}(\Diamond T)$ is (strictly) positive. The positive reachability problem can be solved by a simple fix-point computation. To use this problem for reachability though we define a variant of positive reachability that is reaching a target set with some positive probability while keeping the remaining part of probability mass in the winning region (for positive probability condition). Given $X, Y \subseteq Q$, let $\mathsf{APre}(X, Y) : \{q \in Q \mid \exists a \in \mathsf{A} : (\mathsf{post}(q, a) \cap X \neq \emptyset$ and $\mathsf{post}(q, a) \subseteq Y)\}$ be the set of states from which the player has some action a to ensure reaching X with some positive probability, and moreover the set Y is not left with probability 1. A nested fix-point computation gives us the set of winning states win. Let $\mathsf{win}_0 = Q$, and for all $m \geq 0$, iterate the following two steps:

(1) for all $n \geq 1$ compute $X_n = \mathsf{APre}(X_{n-1}, \mathsf{win}_m) \cup X_{n-1}$ where $X_0 = T$, until the computation stabilizes into X (i.e., $X_n = X_{n+1} = X$ for some n), and then as the next step let

(2) $\mathsf{win}_{m+1} = X$.

Since Q is finite and $\mathsf{win}_{m+1} \subseteq \mathsf{win}_m$ then this sequence will stabilize (in at most $|Q|$ iterations) into a set win that is the set of winning states from which there is a positive probability to reach T in at most $|Q|$ rounds, and since win is never left, the probability of

eventually reaching T is 1. Thus, win is the set of winning states for almost-sure reachability condition $\Diamond T$.

We study another approach, which might be more intuitive, to compute the winning set for almost-sure reachability. Given $S \subseteq Q$, let $\mathsf{Pre}_{\mathsf{some}}(S) = \{q \in Q \mid \exists a \in \mathsf{A} : \mathsf{post}(q,a) \cap S \neq \emptyset\}$ be the set of states from which the player has some action to ensure reaching S with some (strictly) positive probability; note that $\mathsf{Pre}_{\mathsf{some}}(S) = \mathsf{APre}(S,Q)$. Let $\mathsf{Pre}_{\mathsf{all}}(S) = \{q \in Q \mid \forall a \in \mathsf{A} : \mathsf{post}(q,a) \cap S \neq \emptyset\}$ be the set of states from which the player has no choice unless going to S with some positive probability.

An algorithm to compute the set of winning states for positive reachability condition $\Diamond T$ is the following fix-point computation. For all $n \geq 1$, compute $\mathsf{win}_n = \mathsf{Pre}_{\mathsf{some}}(\mathsf{win}_{n-1}) \cup \mathsf{win}_{n-1}$ where $\mathsf{win}_0 = T$. We see that $\mathsf{win}_0 \subseteq \mathsf{win}_1 \subseteq \mathsf{win}_2 \subseteq \cdots$ and since Q is finite (and all $\mathsf{win}_i \subseteq Q$) then this sequence will stabilize (in at most $|Q|$ iterations) into a set $\mathsf{win}_{\mathsf{some}}(\Diamond T)$ that is the set of winning states for positive reachability $\Diamond T$; in other words, the set of states from which the player has some strategy to ensure positive reachability $\Diamond T$. In a similar way, the following fix-point computation, $\mathsf{win}_n = \mathsf{Pre}_{\mathsf{all}}(\mathsf{win}_{n-1}) \cup \mathsf{win}_{n-1}$ for all $n \geq 1$ where $\mathsf{win}_0 = T$, gives the set of states $\mathsf{win}_{\mathsf{all}}(\Diamond T)$ where all strategies of the player are winning for the positive reachability $\Diamond T$.

We are now equipped to solve almost-sure reachability $\Diamond T$: let $\mathcal{M}_0 = \mathcal{M}$ be the MDP and $Q_0 = Q$ be the state space. For all $i \geq 0$, compute the set

$$Q_{i+1} = Q_i \setminus \mathsf{win}_{\mathsf{all}}(\Diamond(Q_i \setminus \mathsf{win}_{\mathsf{some}}(\Diamond T)));$$

where $\mathsf{win}_{\mathsf{some}}$ and $\mathsf{win}_{\mathsf{all}}$ are computed on the MDP $\mathcal{M}_i$. For all $i \geq 1$, the MDP $\mathcal{M}_i$ is obtained from $\mathcal{M}_{i-1}$ where the state space is $Q_i \cup \{\mathsf{sink}\}$ and the transition function δ_i is defined as follows. For all states q and actions a such that $\mathsf{post}(q,a) \subseteq Q_i$, let $\delta_i(q,a) = \delta_{i-1}(q,a)$. Otherwise, $\delta_i(q,a)$ is the Dirac distribution on sink. The state sink is an absorbing state.

Intuitively, the algorithm first computes all states $\mathsf{win}_{\mathsf{some}}(\Diamond T)$ from which $\mathcal{M}_i$ has a strategy to reach T with some positive probability. Thus, the set $Q_i \setminus \mathsf{win}_{\mathsf{some}}(\Diamond T)$ contains all states from which there is no chance to reach T, even with a very small probability. Another important point is, as soon as $\mathcal{M}_i$ enters in some state of the set $Q_i \setminus \mathsf{win}_{\mathsf{some}}(\Diamond T)$, regardless of the player strategy, it always remains in that set with some positive probability. Thus, all states from which player has no choice unless going to $Q_i \setminus \mathsf{win}_{\mathsf{some}}(\Diamond T)$ with some positive probability, must be avoided. We remove such states, that are states in $\mathsf{win}_{\mathsf{all}}(\Diamond(Q_i \setminus \mathsf{win}_{\mathsf{some}}(\Diamond T))$, from $\mathcal{M}_i$ (and all in-going or out-going transitions) to obtain $\mathcal{M}_{i+1}$. We repeat this computation until there is no such state to be removed from the MDP, and as a result the set of winning states for almost-sure reachability $\Diamond T$ is computed. All the computations can be done in PTIME. We refer the reader to [BK08] for more details and the correctness of both algorithms.

The set of limit-sure winning states for reachability condition coincide with the set of almost-sure winning states [dA97]. The PTIME-hardness results of the membership problems for reachability condition, is by a reduction from *monotone Boolean circuit problem*, that is known to be PTIME-complete [GR88].

Safety condition. Let $\mathcal{M}$ be an MDP with state space Q and let $T \subseteq Q$. In analogy to reachability, the membership problem of sure winning for safety condition $\Box T$ is reducible to safety in a *turn based two-player game*. A naive algorithm to compute the set of winning states for sure safety condition $\Box T$ in the MDP $\mathcal{M}$ is the following fix-point computation. For all $n \geq 1$, compute $\mathsf{win}_n = \mathsf{Pre}(\mathsf{win}_{n-1}) \cap \mathsf{win}_{n-1}$ where $\mathsf{win}_0 = T$ and $\mathsf{Pre}(S) = \{q \in Q \mid \exists a \in \mathsf{A} : \mathsf{post}(q, a) \subseteq S\}$. We see that $\mathsf{win}_n \subseteq \mathsf{win}_{n-1}$, therefore this sequence will stabilize (in at most $|T|$ iterations) into a *safe* set win that is the set of winning states. It is known for safety conditions $\Box T$ in MDPs that the three winning modes coincide [CH12].

Büchi and coBüchi conditions. Let $\mathcal{M}$ be an MDP with state space Q and let $T \subseteq Q$. Analogous to reachability, the membership problem of sure winning for Büchi condition $\Box\Diamond T$ and coBüchi condition $\Diamond\Box T$ are reducible to Büchi and coBüchi in a *turn based two-player game*. A well-known iterative algorithm to compute the set of winning states for Büchi condition $\Box\Diamond T$ in a turn-based two player games, has the following logic. For all $i = 1, 2, \cdots, |Q|$, the algorithm performs two steps:

(1) computes the set win_i of states where the Player-1 has a strategy to ensure a visit to T. Next,

(2) it computes the set lose_i of states where the adversary has a strategy to ensure a visit to $Q \setminus \mathsf{win}_i$. The set lose_i is removed from the game.

The states that are remaining in the game after the computation are all sure winning for Büchi condition $\Box\Diamond T$. We refer the reader to [CHP08] for the detailed algorithm.

For an MDP $\mathcal{M}$, a set $C \subseteq Q$ is *closed* if for every state $q \in C$, there exists $a \in \mathsf{A}$ such that $\mathsf{post}(q, a) \subseteq C$. For each $q \in C$, let $D_C(q) = \{a \in \mathsf{A} \mid \mathsf{post}(q, a) \subseteq C\}$. The digraph induced by C is $\mathcal{M} \restriction C = (C, E)$ where E is the set of edges $\langle q, q' \rangle \in C \times C$ such that $q' \in \mathsf{post}(q, a)$ for some $a \in D_C(q)$. An *end component* is a closed set U such that the digraph $\mathcal{M} \restriction U$ is strongly connected. An end component $U \subseteq Q$ is *terminal* if $\mathsf{post}(U, \mathsf{A}) \subseteq U$. Let $\mathsf{endComp}(\mathcal{M})$ be the set of all end components. For the MDP $\mathcal{M}$ of Example 2.3, the set of all end components is $\mathsf{endComp}(\mathcal{M}) = \{\{q_0\}, \{q_1\}, \{q_3\}\}$.

The following remarks are folklore results about end components:

Remark 1. Given an MDP $\mathcal{M}$, for all end components $U \in \mathsf{endComp}(\mathcal{M})$ and all initial distributions X_0 where $\mathsf{Supp}(X_0) \subseteq U$, there exists a (finite-memory) strategy such that $\mathsf{Pr}^\alpha(\Box U) = 1$ and $\mathsf{Pr}^\alpha(\Box\Diamond\{q\}) = 1$ for all $q \in U$.

Roughly speaking, this remark says that for all end components U there is a (finite-memory) strategy that almost surely ensures that $\mathcal{M}$ stays forever in the end component U and ensures that all states of the end component U are visited, infinitely often. The second folk remark is:

Remark 2. For an MDP $\mathcal{M}$, let $\Omega = \{\pi \in \mathsf{Plays}(\mathcal{M}) \mid \mathsf{Inf}(\pi) = U \text{ for some } U \in \mathsf{endComp}(\mathcal{M})\}$. Then $\Pr^\alpha(\Omega) = 1$ under all strategies α and from all initial distributions X_0.

Proof (sketch). Consider an initial distribution X_0 and a strategy α for the MDP $\mathcal{M}$. For all infinite paths $\pi = q_0 q_1 q_2 \cdots$ of the MDP $\mathcal{M}$ such that $q_0 \in \mathsf{Supp}(X_0)$, and all $q \in \mathsf{Inf}(\pi)$,

there is some action a that is played by α, infinitely often, for the prefixes ρ of the play with $\mathsf{Last}(\rho) = q$ (prefixes that ends in q). Given the condition that the a-transition in q is taken infinitely often along the play π, for all $q' \in \mathsf{post}(q, a)$, the conditional probability that $\mathcal{M}$ moves from q to q' for only finitely many times is zero. Therefore, given that the a-transition in q is taken infinitely often along the play π, all successors $\mathsf{post}(q, a)$ of the a-transition in q are almost-surely visited infinitely often along π. Hence, for all states $q \in \mathsf{Inf}(\pi)$ there is an action a such that $\mathsf{post}(q, a) \in \mathsf{Inf}(\pi)$ meaning that $\mathsf{Inf}(\pi)$ almost-surely is (strongly connected and) included in an end component.

□

These two remarks enable us to reduce the membership problem of almost-sure mode for Büchi and coBüchi conditions, to the membership problem of almost-sure for reachability. For the Büchi condition $\Box\Diamond T$, let U_{good} be the union of all end components $U \in \mathsf{endComp}(\mathcal{M})$ such that $U \cap T \neq \emptyset$; the membership problem of almost-sure for Büchi condition $\Box\Diamond T$ is reducible to the membership problem of almost-sure mode for reachability condition $\Diamond U_{\mathsf{good}}$. The reduction is correct since as soon as the MDP $\mathcal{M}$ enters the end component U where $U \cap T \neq \emptyset$, there is a strategy that ensures, infinitely often, the visits to T.

The following modifications are necessary when reducing the membership problem of almost-sure coBüchi condition $\Diamond\Box T$ to the membership problem of almost-sure reachability condition. The reason for the following modifications is because an end component $U \in \mathsf{endComp}(\mathcal{M})$ of an MDP$\mathcal{M}$ where $T \subseteq U$, may include some bad state $q \notin T$ where there is no strategy to avoid visiting that bad state whereas it may have such a state but there is some strategy which visits only the states of $U \cap T$, infinitely often. Let $\mathcal{M}'$ be a copy of $\mathcal{M}$ where all states $q' \notin T$ are replaced with an absorbing state sink, and for all states $q \in T$ and all actions $a \in \mathsf{A}$, if the successor set of the a-transition in q is not included in T, i.e., $\mathsf{post}(q, a) \not\subseteq T$, then the a-transition is redirected to sink. Let U_{good} be the union of all end components $U' \in \mathsf{endComp}(\mathcal{M}')$ of the $\mathcal{M}'$ such that $U' \subseteq T$. The membership problem of almost-sure for coBüchi condition $\Diamond\Box T$ is reducible to the membership problem of almost-sure mode for reachability condition $\Diamond U_{\mathsf{good}}$ in the same MDP $\mathcal{M}$.

As the almost-sure and limit-sure winning states for reachability condition coincide in MDPs, the sets of winning states of almost-sure and limit-sure are the same for Büchi condition too. The same result holds for coBüchi condition.

2.5.3 Problems for PAs

Similar to NFAs, the emptiness problems is a classical decision problem for PAs. We recall that the classical languages of PAs are defined on pure words.

Decision problem(s). The *emptiness problem* asks, given a PA $\mathcal{P}$ with an acceptance condition, whether the accepted language of $\mathcal{P}$ is empty.

The language of finite words for a PA $\mathcal{P}$ is defined for a given threshold $0 \leq \lambda \leq 1$; recall that $\mathcal{L}^{\lambda}(\mathcal{P}) = \{w \in \mathsf{A}^* \mid \mathsf{Pr}(w) \geq \lambda\}$. The emptiness problem asks, thus, the existence of a finite word $w \in \mathsf{A}^*$ such that $\mathsf{Pr}(w) \geq \lambda$. The *strict emptiness problem* is to decide the existence of a finite word $w \in \mathsf{A}^*$ such that $\mathsf{Pr}(w) > \lambda$. For the extreme value $\lambda = 0$, the answer to the emptiness problem is always yes; and the strict emptiness problem reduces to emptiness problem for NFAs. For the extreme value $\lambda = 1$, emptiness problem is reducible to the universality problem for NFAs; and the answer to the strict emptiness problem is always no. For $0 < \lambda < 1$, both emptiness problem and strict emptiness problem are undecidable [GO10].

It is known [BBG08, CT12] that the emptiness problem for PAs with positive Büchi acceptance conditions is undecidable, whereas it is PSPACE-complete for almost-sure Büchi acceptance conditions. The emptiness problem for PAs with almost-sure safety and reachability conditions are PSPACE-complete, too.

An interesting problem for PAs with a finite acceptance condition is the *value* 1 *problem*, which is used to establish several complexity results throughout this thesis. The value of a PA $\mathcal{P}$ augmented with the finite acceptance condition $\mathcal{F}$, is $\mathsf{val}(\mathcal{P}) = \sup_{w \in \mathsf{A}^*} \mathsf{Pr}(w)$.

Decision problem(s). The value 1 problem asks, given a PA $\mathcal{P}$ and a finite acceptance condition, whether $\mathsf{val}(\mathcal{P}) = 1$.

Intuitively, it asks whether some words exist with the acceptance probability arbitrarily close to 1. The value 1 problem for PAs is proved undecidable in [GO10]. Subsection 6.2.5 on page 134 provides some details about this problem.

3 Synchronizing Problems

First sight. In this chapter we define the synchronizing problems we are mainly concerned about in this thesis. We recall the definition of finite synchronizing words for NFAs, and we survey decision problems and conjectures about finite synchronizing words.

We then provide definitions of infinite synchronizing words for NFAs. We introduce variants of synchronizations in MDPs and PAs by considering synchronizing strategies and (finite and infinite) synchronizing words. We study the relation between one-letter finite alternating automata and MDPs. We also briefly discuss the synchronization in two-player turn-based games.

Contents

3.1 Synchronization in NFAs

The finite *synchronizing* words are well-studied for finite automata, for example see [Čer64, Pin78, Epp90, IS95, IS99, San04, Vol08, Mar10, AGV10, OU10, Mar12]; such a word drives the automaton from an unknown or unobservable state to a known specific state. As an example think of remote systems connected to a wireless controller that emits the command via wireless waves but expects the observations via physical connectors (it might be excessively expensive to mount wireless senders on the remote systems), and consider that the physical connection to controller is lost due to some technical failure. The wireless controller can therefore not observe the current states of the distributed subsystems. In this setting, emitting a *synchronizing word* as the command leaves the remote system (as a whole) in one particular state no matter in which state each distributed subsystem started; and thus the controller again regain control.

For synchronizing words, called directable or reset words as well, applications in planning, control of discrete event systems, bio-computing, and robotics [BAPE+03, Vol08, DMS11a].

3.1.1 Finite synchronization in NFAs

A finite automaton is *synchronizing* if there exists an input word, called synchronizing word, that brings the automaton from every state to the same state.

Definition 3.1 (Finite synchronizing words)**.** *For a complete NFA $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$, a finite synchronizing word is a word $w \in \mathsf{A}^*$ for which there exists a state $q \in Q$ such that $Q \xrightarrow{w} \{q\}$.*

The natural decision problem for finite synchronization in NFAs is as follows.

Decision problem(s). The *finite synchronizing problem* asks, given an NFA, whether a finite synchronizing word exists.

A straightforward approach to decide finite synchronizing problem is a reduction to the emptiness problem in the *subset construction* of the NFA. For a complete NFA $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$, the subset construction which is usually defined to determinize the automaton, consists in $\langle Q', \mathsf{A}, \Delta' \rangle$ where the state space $Q' = 2^Q \setminus \{\emptyset\}$ is the set of all *cells*, i.e., subsets of states in $\mathcal{N}$, and the transition function $\Delta' : Q' \times \mathsf{A} \to Q'$ defined as $\Delta'(c, a) = c'$ where $c \xrightarrow{a} c'$ and $a \in \mathsf{A}$. The finite synchronizing problem for $\mathcal{N}$ can be encoded as the emptiness problem for the deterministic subset construction of $\mathcal{N}$ equipped with the initial cell $c_0 = Q$ and the set $\mathcal{F} = \{\{q\} \mid q \in Q\}$ of accepting cells. By this reduction, one can prove that the language of finite synchronizing words for an NFA $\mathcal{N}$, that is $\mathcal{L}^{\mathsf{synch}} = \{w \in \mathsf{A}^* \mid w \text{ is a finite synchronizing word for } \mathcal{N}\}$, is a regular language.

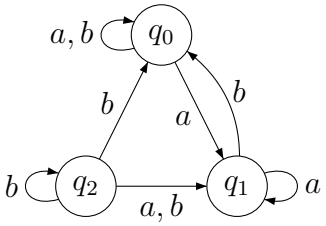

Figure 3.1: The NFA described in Example 3.1 with synchronizing word $a \cdot a \cdot b$.

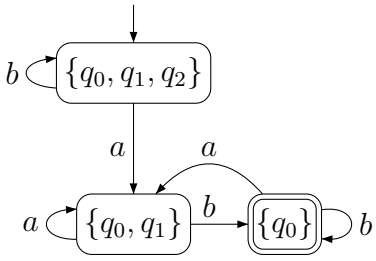

Figure 3.2: The subset construction of the NFA in Example 3.1 (restricted to the part showing the run over $a \cdot a \cdot b$).

Example 3.1. *The NFA $\mathcal{N}$ depicted in Figure 3.1 has three states q_0, q_1, q_2 and has two letters a and b. The a-transitions in q_0 is non-deterministic and bring $\mathcal{N}$ into q_0 or q_1: $\{q_0\} \xrightarrow{a} \{q_0, q_1\}$; the a-transitions in q_1 and q_2 however deterministically go to q_1. The b-transition in q_2 spreads the current state of the automaton into the state space: $\{q_2\} \xrightarrow{b} \{q_0, q_1, q_2\}$ whereas the b-transitions in q_0 and q_1 are deterministic and moves the automaton into q_0. The word $a \cdot a \cdot b$ is synchronizing since $\{q_0, q_1, q_2\} \xrightarrow{a \cdot a \cdot b} \{q_0\}$; it can be verified by Figure 3.2 that illustrates some part of the subset construction to show the run from the initial cell $\{q_0, q_1, q_2\}$ to the accepting cell $\{q_0\}$ over the word $a \cdot a \cdot b$. We see that all words $b^* \cdot a \cdot a^* \cdot b \cdot a^*$ are synchronizing too.*

◁

The reduction to emptiness problem for subset construction provides the PSPACE membership of the finite synchronizing problem for NFAs. A reduction from the *finite automata intersection* that is PSPACE-complete [Koz77, GJ79, San04], has established the matching lower bound to prove that finite synchronizing problem is indeed that costly, even if only one transition in the NFA is non-deterministic [IS99, Mar10, Mar12]. The same complexity results hold for the finite synchronizing problem of not-complete DFAs, called *partial finite automata* (PFAs), even if the transition function is undefined only for one state-action pair [Mar10, Mar12]. When the DFA is complete, the finite synchronizing problem lies in NLOGSPACE. Below, we see that the reason of the lower complexity for DFAs is the *pair-wise synchronizing* technique whose idea comes from a Lemma proved by Černý in the early 1960's [Čer64].

Finite synchronizing word for DFAs. For a given complete deterministic finite automaton $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$, there exists a finite synchronizing word if, and only if, for all pairs of states $q_1, q_2 \in Q$, there exists a state q and a word w such that $\{q_1, q_2\} \xrightarrow{w} \{q\}$. This result is by a Lemma from Černý [Čer64, Epp90, Vol08]; one direction of the statement is trivial, by monotonicity for all states q, if $Q \xrightarrow{w} \{q\}$ implying that w is a synchronizing word for $\mathcal{N}$ then $\{q_1, q_2\} \xrightarrow{w} \{q\}$ for all pairs of states q_1 and q_2. The reverse direction is provable by an easy observation about DFAs: when runs (over any arbitrarily word) starting in two states q_1 and q_2 cross each other in some state q, those runs stay merged/synchronized forever. In other words, all words $w \cdot v \in \mathsf{A}^*$ are synchronizing when w is a finite synchronizing

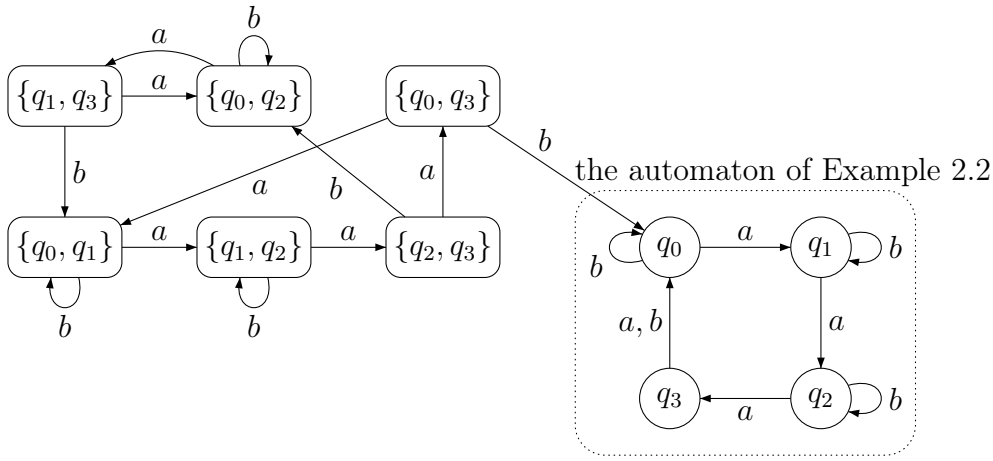

Figure 3.3: The subset construction of the NFA in Example 2.2 on page 19 (where the state space is restricted only to cells of size at most 2).

word followed by $v \in \mathsf{A}^*$. This observation does not hold in a non-deterministic setting. As a result in DFAs, the state space can be synchronized by consecutive synchronizations of pairs of states. The NLOGSPACE upper bound for DFAs follows from an algorithm repeating *pair-wise synchronization* as follows:

(1) choose a pair of states q_1 and q_2 from the set S of *must-be-synchronized* states, where S is initially set to the state space Q.

(2) find a word w for which there exists q such that $\{q_1, q_2\} \xrightarrow{w} \{q\}$; if such a word does not exists, return "no synchronizing word exists". Otherwise, update the set of must-be-synchronized states: compute the successor set S' after inputting the word w: $S \xrightarrow{w} S'$ and let $S = S'$.

(3) if S is a singleton, return "the automaton is synchronizing".

The termination of this algorithm is guaranteed by Černý's Lemma. In fact by each iteration, the set S shrinks to a strictly smaller set and the algorithm terminates in at most $|Q|$ iterations. The algorithm is thus in NLOGSPACE that comes from the second step which is finding a word w that merges runs from two states q_1 and q_2; this can be reduced to the membership problem in the subset construction of the automaton where the state space (of the subset construction) is restricted only to cells of size at most 2 (equivalently to the reachability problem in the product of two copies of the DFA). Figure 3.3 shows such subset construction, i.e., restricted to cells of size at most 2, for the deterministic automaton described in Example 2.2 on page 19. We see that for all pairs of states, there is a run to the accepting cell $\{q_0\}$. A finite synchronizing word for this automaton is $b \cdot a^3 \cdot b \cdot a^3 \cdot b$. This algorithm constructs a synchronizing word for the input DFA, if any, that is not always the shortest one.

The famous Černý conjecture claims that the length of shortest synchronizing word for a DFA with $n > 1$ states is at most $(n-1)^2$. Despite all attempts in last decades, this conjecture has not been proved or disproved. Černý provided a family of the automaton (over the alphabet $\{a, b\}$) where the shortest synchronizing word is exactly the extreme value $(n-1)^2$. An n-states automaton in this family has following structure: there is a loop going through $q_0 q_1 \cdots q_{n-1}$ and closing in q_0. For all states, the a-transitions shift

once the automaton in this loop $q_0q_1\cdots q_{n-1}q_0$. For all states $q \neq q_{n-1}$, the b-transition is a self-loop on q; the b-transition in q_{n-1} takes the automaton to q_0. See Example 2.2 for a formal description for the transition function of the 4-state automaton instance of this family. The shortest synchronizing word for the automaton in Example 2.2 is $b \cdot a^3 \cdot b \cdot a^3 \cdot b$ and for the n-state automaton in this family is $(b \cdot a^{n-1})^{n-2} \cdot b$. This family is the only known family satisfying the extreme bound on the length of shortest synchronizing word [Vol08, AGV10]. The best known upper bound for the shortest synchronizing words of an n-state automaton is $\frac{1}{6}(n^3 - n - 6)$, proved by Pin [Pin78, Vol08]. Trahtman recently tried to provide a better upper bound $\frac{1}{48}(7n^3+6n^2-16n)$ where the proof was unfortunately erroneous [Tra11, Tra14]. The related decision problem which asks, given a DFA and the bound k, whether the DFA has a synchronizing word with length less than k, is complete for the class DP [OU10], the closure of NP∪coNP under finite intersections.

Synchronization from a subset in NFAs. A variant of finite synchronization in NFAs is synchronization from a subset S of states.

Decision problem(s). Given an NFA $\mathcal{N} = \langle Q, \mathsf{A}, \Delta\rangle$ and a subset $S \subseteq Q$, the *finite synchronizing problem from a subset* S asks whether there exists a finite word $w \in \mathsf{A}^*$ and some state q such that $S \xrightarrow{w} \{q\}$.

The finite synchronizing problem from a subset is PSPACE-complete, even if the automaton is complete and deterministic [San04]. In DFAs, the *pair-wise synchronizing* technique provides a NLOGSPACE algorithm for the synchronizing finite problem. This technique fails when synchronizing from a subset is asked because a pair of states $q_1, q_2 \in S$ are not guaranteed to be merged in a state q from which (and the states of the must-be-synchronized set) there is a synchronizing word. Thus, the order in which the pairs of state are chosen for pair-wise synchronization matters. Note that, an NFA might not have a synchronizing word but synchronizing words from S exist for several subsets S of the states.

The PSPACE upper bound for the synchronization from a subset comes trivially from the membership problem of the subset construction. To find a synchronizing word from S, we equip the subset construction with the initial cell $c_0 = S$ and the set $\mathcal{F} = \{\{q\} \mid q \in Q\}$ of accepting cells. The PSPACE-hardness is by a reduction from *finite automata intersection* [Koz77, GJ79, San04]; the finite automata intersection asks, given n DFAs equipped with initial states and accepting sets, is there a common finite word that is accepting by all DFAs. For n automaton $\mathcal{N}_1, \mathcal{N}_2, \cdots, \mathcal{N}_n$ over the common alphabet A, let $q_1, q_2, \cdots, q_n$ be the initial states and $\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_n$ be the accepting sets. Let $\mathcal{N}$ be the automaton that has one copy of each automaton $\mathcal{N}_i$ $(1 \leq i \leq n)$ and two new absorbing states synch and sink. For a new letter #, let #-transitions in all accepting states $q \in \mathcal{F}_1 \cup \cdots \cup \mathcal{F}_n$ of all automata go to the state synch whereas all #-transitions in all non-accepting states $q \notin \mathcal{F}_1 \cup \cdots \cup \mathcal{F}_n$ go to the state sink. From the subset $S = \{q_1, q_2, \cdots, q_n, \mathsf{synch}\}$, a synchronizing word cannot escape synch because synch is an absorbing state. Let w be the shortest synchronizing word from S. As a result w must end with #: $w = v \cdot \#$ where

$v \in \mathsf{A}^*$; it holds since the only way that runs starting in the initial states $q_1, q_2, \cdots, q_n$ arrives in synch is inputting $\#$ at the right time when those runs are in some accepting states. One can verify that v is an accepting word for all $\mathcal{N}_i$ $(1 \leq i \leq n)$.

3.1.2 Infinite synchronization in NFAs

We provide four definitions of infinite synchronizing words for NFAs to define languages of infinite synchronizing words, in analogy to safety, reachability, Büchi and coBüchi acceptance conditions. For an infinite word $w = a_0 \cdot a_1 \cdot a_2 \cdots$, let $w_0 = \epsilon$ be the empty word and for all $i \geq 1$, let w_i denote the prefix $a_0 \cdots a_{i-1}$ of w consisting in the first i letters.

Definition 3.2 (Infinite synchronizing words)**.** *For a complete NFA $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$, an infinite word $w \in \mathsf{A}^\omega$ is*

- *always synchronizing if for all $i \in \mathbb{N}$ there exists some state q such that $Q \xrightarrow{w_i} \{q\}$,*
- *eventually (or event) synchronizing if there exists $i \in \mathbb{N}$ and some state q such that $Q \xrightarrow{w_i} \{q\}$,*
- *weakly synchronizing if for all $n \in \mathbb{N}$ there exists $i \geq n$ and some state q such that $Q \xrightarrow{w_i} \{q\}$,*
- *strongly synchronizing if there exists $n \in \mathbb{N}$ such that for all $i \geq n$ there exists some state q where $Q \xrightarrow{w_i} \{q\}$*

where $w_0 = \epsilon$ and for $i \geq 1$, the word w_i is the prefix of w consisting in the first i letters.

Intuitively, the always and eventually synchronization condition require that the NFA is always synchronized or at least once; though the weakly requires infinitely often synchronization, and strongly requires always synchronization except for finitely many times. We say that an NFA $\mathcal{N}$ is {always, event, strongly, weakly} synchronizing if there exists a(n) {always, event, strongly, weakly} synchronizing infinite word for $\mathcal{N}$.

For an NFA $\mathcal{N}$, let the language $\mathcal{L}^{always}$, $\mathcal{L}^{event}$, $\mathcal{L}^{strongly}$ and $\mathcal{L}^{weakly}$ be the set of all always, eventually, strongly and weakly synchronizing words, respectively. As an immediate result of definitions:

Remark 3. For an NFA $\mathcal{N}$, we have $\mathcal{L}^{always} \subseteq \mathcal{L}^{strongly} \subseteq \mathcal{L}^{weakly} \subseteq \mathcal{L}^{event}$.

These inclusions, in general, are strict; see Example 3.2. Note that the language $\mathcal{L}^{always}$ is an empty set for all NFAs that have at least two states. In the sequel, we present variations of these definitions where the always synchronization is possible. For a DFA defined over an alphabet A, we have $\mathcal{L}^{strongly} = \mathcal{L}^{weakly} = \mathcal{L}^{event}$ where $\mathcal{L}^{event}$ is the set of all words $w.v$ constructed by a finite synchronizing word w concatenated with an infinite word $v \in \mathsf{A}^\omega$.

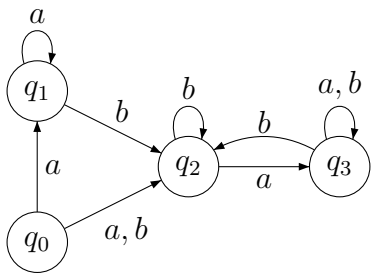

Figure 3.4: The NFA $\mathcal{N}_1$ of Example 3.2 with the strongly and eventually synchronizing words $b \cdot a^\omega$ and $b \cdot a \cdot b^\omega$.

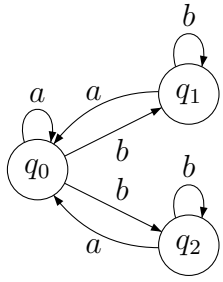

Figure 3.5: The NFA $\mathcal{N}_2$ of Example 3.2 with weakly synchronizing word $(a \cdot b)^\omega$ and strongly synchronizing word a^ω.

Example 3.2. *The NFA $\mathcal{N}_1$ in Figure 3.4 has four states q_0, q_1, q_2, q_3 and two letters a and b. The a-transition in q_0 is non-deterministic $\{q_0\} \xrightarrow{a} \{q_0, q_1\}$; but it is deterministic in other states: $q_2 \xrightarrow{a} q_3$, $q_3 \xrightarrow{a} q_2$ and it is a self-loop on q_1. The b-transitions in q_0 and q_1 take the automaton to q_2, and it is a self-loop on q_2 itself. The b-transition in q_3 is non-deterministic: $\{q_3\} \xrightarrow{b} \{q_2, q_3\}$. The word $w = b \cdot a$ is a finite synchronizing word for $\mathcal{N}_1$, thus any infinite word initiated with w is an eventually synchronizing word, such as $b \cdot a \cdot b^\omega$. The word $b \cdot a \cdot b^\omega$ is a witness to show that $\mathcal{L}^{weakly} \subset \mathcal{L}^{event}$ for $\mathcal{N}_1$. A strongly and weakly synchronizing word, here, is $b \cdot a^\omega$.*

The NFA $\mathcal{N}_2$ depicted in Figure 3.5 is an example to show that there are NFAs for which the inclusion $\mathcal{L}^{strongly} \subset \mathcal{L}^{weakly}$ is strict. The a-transitions in this automaton synchronizes $\mathcal{N}$ in q_0: $q_i \xrightarrow{a} q_0$ for all $0 \leq i \leq 2$. The b-transitions in q_1 and q_2 are self-loops; and $\{q_0\} \xrightarrow{b} \{q_1, q_2\}$. The languages of strongly and weakly synchronizing words are

$$\mathcal{L}^{strongly} = (a+b)^* a^\omega \text{ and } \mathcal{L}^{weakly} = \mathcal{L}^{strongly} + (a+b)^*(b \cdot b^* \cdot a \cdot a^*)^\omega.$$

◁

Decision problem(s). For all $\lambda \in \{\text{always, event, strongly, weakly}\}$, the *emptiness problem* for $\mathcal{L}^\lambda$ asks, given an NFA $\mathcal{N}$, whether no infinite λ-synchronizing words exist for $\mathcal{N}$ meaning that $\mathcal{L}^\lambda = \emptyset$.

Similar to finite synchronization, we can define λ-synchronization from an initial subset Q_0 of states. For example, a strongly synchronizing word from Q_0 is a word $w \in \mathsf{A}^\omega$ for which there exists $n \in \mathbb{N}$ such that for all $i \geq n$ there exists some state $q \in Q$ such that $Q_0 \xrightarrow{w_i} \{q\}$. Note that if Q_0 is a singleton, all infinite words are trivially eventually synchronizing. In this case, we are mostly interested in nontrivial synchronization (when synchronization after at least one input is of interest). For the automaton $\mathcal{N}_2$ described in Example 3.2, since there is a weakly synchronizing word from Q, there are weakly synchronizing words from all subsets Q_0 of Q.

For an NFA $\mathcal{N}$ with an initial set Q_0, the set of all infinite runs over an infinite word $w = a_0 \cdot a_1 \cdot a_2 \cdots$ can be visualized as an acyclic graph for which there is $|Q_0|$ roots: all states $q_{0,j}$

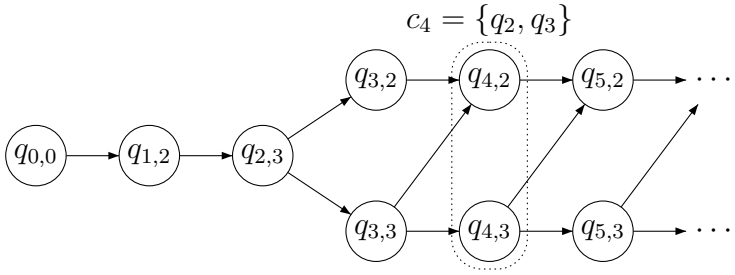

Figure 3.6: The acyclic graph of runs induced by the word $b \cdot a \cdot b^\omega$ over the automaton $\mathcal{N}_1$ described in Example 3.2.

where $q_j \in Q_0$. The set of vertices is

$$V = \{q_{0,j} \mid q_j \in Q_0\} \cup \bigcup_{i \in \mathbb{N}} \{q_{i+1,j} \mid q_j \in S \text{ where } Q_0 \xrightarrow{w_i} S\}$$

where $w_i = a_0 \cdots a_i$ is the prefix of w consisting in the first $i+1$ letters. There is an edge from $q_{i,j}$ towards $q_{i+1,k}$, denoted by $q_{i,j} \to q_{i+1,k}$, if $q_j \xrightarrow{a_i} q_k$, $q_j, q_k \in Q$. For all $i \in \mathbb{N}$, define $c_i = \{q_j \mid q_{i,j} \in V\}$, that is the set of states where the automaton may arrive in after inputting the first i letters. Thus, $c_0 = Q_0$ and $c_1 = \Delta(Q_0, a_0)$. In fact, the sequence $c_0 c_1 c_2 \cdots$ is nothing else than the sequence of sets where $c_0 = Q_0$ and $Q_0 \xrightarrow{w_i} c_{i+1}$. Each infinite path starting in a root, on this acyclic graph is indeed an infinite run induced by the word w. As an example, consider the infinite word $b \cdot a \cdot b^\omega$ over the NFA $\mathcal{N}_1$ in Example 3.2. The acyclic graph of runs starting from $Q_0 = \{q_0\}$ is drawn in Figure 3.6.

Now, we can re-formalize the definitions of infinite synchronizing words for the NFAs. Let $\mathcal{N}$ be an NFA and Q_0 be an initial set. For example, a strongly synchronizing word from Q_0 is a word $w \in \mathsf{A}^\omega$ for which there exists $n \in \mathbb{N}$ such that for all $i \geq n$ we have $|c_i| = 1$; a weakly synchronizing word from Q_0 is a word $w \in \mathsf{A}^\omega$ such that for all $n \in \mathbb{N}$ there exists $i \geq n$ where $|c_i| = 1$. For the NFA $\mathcal{N}_1$ in Example 3.2 the word $w = b \cdot a \cdot b^\omega$ is neither strongly nor weakly synchronizing from $Q_0 = \{q_0\}$: because $|c_i| \geq 2$ for all $i \geq 3$ (the acyclic graph induced by w is drawn in Figure 3.6). This word w is however a (nontrivial) eventually synchronizing word: $|c_1| = 1$.

We introduce two other variants of infinite synchronization for an NFA. In these variants, the states where the automaton arrives in at synchronization's time is under question. For a target set T, we define the Boolean functions $sub_T : 2^Q \to \{\mathsf{true}, \mathsf{false}\}$ and $mem_T : 2^Q \to \{\mathsf{true}, \mathsf{false}\}$ where for all $c \subseteq Q$, we have $sub_T(c) = \mathsf{true}$ if $\emptyset \neq c \subseteq T$ and $mem_T(c) = \mathsf{true}$ if $|c| = 1$ and $c \subseteq T$. Intuitively, the function sub_T only decides if the input set is a non-empty subset of T (subset inclusion of c) though the function mem_T decides if c is a singleton such as $c = \{q\}$, and if $q \in T$ (membership of q).

Definition 3.3 (Infinite synchronizing words into target sets)**.** *For a complete NFA $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$ with an initial set Q_0, a target set T and $f \in \{sub_T, mem_T\}$, an infinite word $w \in \mathsf{A}^\omega$ from Q_0 according to f is*

- *always synchronizing if $f(c_i)$ for all $i \in \mathbb{N}$,*
- *eventually (or event) synchronizing if $f(c_i)$ for some $i \in \mathbb{N}$,*
- *weakly synchronizing if for all $n \in \mathbb{N}$ there exists $i \geq n$ such that $f(c_i)$,*
- *strongly synchronizing if there exists $n \in \mathbb{N}$ such that for all $i \geq n$ we have $f(c_i)$.*

As an instance, consider the automaton $\mathcal{N}_1$ described in Example 3.2 and the unique initial state q_0. The word $b \cdot a \cdot b^\omega$ where the induced graph of runs is drawn in Figure 3.6, is not strongly synchronizing from q_0 according to the function $mem_{\{q_1,q_2,q_3\}}$ but it is strongly synchronizing according to $sub_{\{q_1,q_2,q_3\}}$.

Given an NFA $\mathcal{N}$ with the initial set Q_0 of states and the target set T, define the λ-synchronizing ω-language for $\mathcal{N}$ according to f to be

$$\mathcal{L}^\lambda(f) = \{\text{the set of all infinite } \lambda\text{-synchronizing words for } \mathcal{N} \text{ from } Q_0 \text{ according to } f\}$$

where $\lambda \in \{\text{always, event, strongly, weakly}\}$ and $f \in \{sub_T, mem_T\}$.

Decision problem(s). The *emptiness problem* of the λ-synchronizing ω-language according to f asks, given an NFA $\mathcal{N}$ and an initial set of states Q_0, whether $\mathcal{L}^\lambda(f) = \emptyset$.

For the automaton $\mathcal{N}_2$ of Example 3.2 with all possible initial set $Q_0 \subseteq Q$ and for all $\lambda \in \{\text{event, strongly, weakly}\}$, we have $\mathcal{L}^\lambda(mem_{\{q_1,q_2\}}) = \emptyset$ whereas $\mathcal{L}^\lambda(sub_{\{q_1,q_2\}}) \neq \emptyset$. For the *emptiness problem* of λ-synchronizing ω-languages according to a function f, without loss of generality we can assume that the initial set is a singleton set: given an NFA $\mathcal{N}$ and an initial set Q_0, we construct a copy of $\mathcal{N}$ with a new initial state q_{init} from which the successor set on all actions is Q_0 (for eventually synchronizing, we need to consider nontrivial synchronization and for always synchronizing, we need to consider the target set $T = T \cup \{q_{\mathsf{init}}\}$). We, thus, always equip the automaton with an initial state q_{init} and study synchronization from that unique initial state.

For all singleton target sets T, we have $sub_T = mem_T$, thus we have $\mathcal{L}^\lambda(sub_T) = \mathcal{L}^\lambda(mem_T)$. For the sake of simplicity we usually denote by $\mathcal{L}^\lambda(q)$ those sets when $T = \{q\}$.

3.2 Synchronization in MDPs

For probabilistic systems, a natural extension of words is the notion of strategy that reacts and chooses actions according to the sequence of states visited along the system execution. In this context, we define synchronizing problems in MDPs regarding to strategies (and not words). To this aim, we consider a symbolic-outcome of MDPs viewed as generators of sequences of probability distributions over states [KVAK10]. In contrast to the

traditional path-outcomes presented in subsection 2.4.2, the symbolic-outcome of MDPs recently has received attention [KVAK10, CKV+11, AAGT12, HKK14].

Let us first define two variants of the reachability condition for MDPs: given a set $T \subseteq Q$ of target states and $k \in \mathbb{N}$, the event of reaching T after exactly k steps is $\Diamond^k T = \{q_0 q_1 q_2 \cdots \in \mathsf{Plays}(\mathcal{M}) \mid q_k \in T\}$, and the event of reaching T within k steps is $\Diamond^{\leq k} T = \bigcup_{j \leq k} \Diamond^j T$. In the same way of reachability condition, these events are measurable provided the strategy α and the initial distribution X_0; thus, the probabilities $\mathsf{Pr}^\alpha(\Diamond^k T)$ and $\mathsf{Pr}^\alpha(\Diamond^{\leq k} T)$ of the events are uniquely defined. As an example, consider the MDP $\mathcal{M}$ described in Example 2.3 on page 21 and the memoryless strategy γ where $\gamma(q_0) = a$ and $\gamma(q) = b$ for all states $q \neq q_0$ (defined in Example 2.5 on page 24). Let X_0 be the Dirac distribution on q_0, then from X_0

- $\mathsf{Pr}^\gamma(\Diamond^k \{q_1\}) = \frac{1}{2^k}$ for all $k \geq 1$: it holds since $\Diamond^k \{q_1\} = (q_0)^{k-1} q_1 Q^\omega$ and since on the action a, the probability that $\mathcal{M}$ stays in q_0 or leaves q_0 to move into q_1 is $\frac{1}{2}p$ if $\mathcal{M}$ is in q_0 with probability p.
- $\mathsf{Pr}^\gamma(\Diamond^{\leq k} \{q_3\}) = 1 - \frac{1}{2^{k-2}}$ for all $k \geq 3$. Since q_3 is absorbing, when $\mathcal{M}$ arrives in q_3 it can never leave out q_3. Hence, $\mathsf{Pr}^\gamma(\Diamond^{\leq k} \{q_3\}) = \mathsf{Pr}^\gamma(\Diamond^k \{q_3\})$.

Symbolic-outcome of MDPs. Let $\mathcal{M}$ be an MDP and $X_0 \in \mathcal{D}(Q)$ be an initial distribution. Under a strategy α, the *symbolic-outcome* of $\mathcal{M}$ from X_0 is the infinite sequence $\mathcal{M}_0^\alpha \mathcal{M}_1^\alpha \mathcal{M}_2^\alpha \cdots$ of probability distributions defined by $\mathcal{M}_k^\alpha(q) = Pr^\alpha(\Diamond^k \{q\})$ for all $k \geq 0$ and $q \in Q$. Hence, $\mathcal{M}_k^\alpha$ is the probability distribution over states after k steps under strategy α. Note that $\mathcal{M}_0^\alpha = X_0$. As an example, the infinite sequence

$$\begin{array}{c} q_0 \\ q_1 \\ q_2 \\ q_3 \end{array} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{2} \\ 0 \end{pmatrix} \begin{pmatrix} \frac{1}{8} \\ \frac{1}{8} \\ \frac{1}{4} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{1}{16} \\ \frac{1}{16} \\ \frac{1}{8} \\ \frac{3}{4} \end{pmatrix} \begin{pmatrix} \frac{1}{32} \\ \frac{1}{32} \\ \frac{1}{16} \\ \frac{7}{8} \end{pmatrix} \cdots$$

is the symbolic-outcome for the MDP $\mathcal{M}$ described in Example 2.3 under the memoryless strategy γ where $\gamma(q_0) = a$ and $\gamma(q) = b$ for all states $q \neq q_0$ (mentioned in Example 2.5).

Synchronization. Informally, synchronizing conditions require that the probability of some target state (or some group of target states) tends to 1 in the sequence $\mathcal{M}_0^\alpha \mathcal{M}_1^\alpha \mathcal{M}_2^\alpha \cdots$, either always, once, infinitely often, or always after some point.

Given a target set $T \subseteq Q$, we define the functions $sum_T : \mathcal{D}(Q) \to [0,1]$ and $max_T : \mathcal{D}(Q) \to [0,1]$ that compute $sum_T(X) = \sum_{q \in T} X(q)$ and $max_T(X) = \max_{q \in T} X(q)$. The function $max_T(X)$ is the quantitative version of the function $mem(\mathsf{Supp}(X))$, and $sum_T(X)$ is the quantitative version of $sub_T(\mathsf{Supp}(X))$.

	Always			Eventually		
Sure	$\exists\alpha$	$\forall n$	$\mathcal{M}_n^\alpha(T) = 1$	$\exists\alpha$	$\exists n$	$\mathcal{M}_n^\alpha(T) = 1$
Almost-sure	$\exists\alpha$	$\inf_n$	$\mathcal{M}_n^\alpha(T) = 1$	$\exists\alpha$	$\sup_n$	$\mathcal{M}_n^\alpha(T) = 1$
Limit-sure	$\sup_\alpha$	$\inf_n$	$\mathcal{M}_n^\alpha(T) = 1$	$\sup_\alpha$	$\sup_n$	$\mathcal{M}_n^\alpha(T) = 1$
	Weakly			**Strongly**		
Sure	$\exists\alpha$	$\forall N\, \exists n \geq N$	$\mathcal{M}_n^\alpha(T) = 1$	$\exists\alpha$	$\exists N\, \forall n \geq N$	$\mathcal{M}_n^\alpha(T) = 1$
Almost-sure	$\exists\alpha$	$\limsup_{n\to\infty}$	$\mathcal{M}_n^\alpha(T) = 1$	$\exists\alpha$	$\liminf_{n\to\infty}$	$\mathcal{M}_n^\alpha(T) = 1$
Limit-sure	$\sup_\alpha$	$\limsup_{n\to\infty}$	$\mathcal{M}_n^\alpha(T) = 1$	$\sup_\alpha$	$\liminf_{n\to\infty}$	$\mathcal{M}_n^\alpha(T) = 1$

Table 3.1: Winning modes and synchronizing conditions (where $\mathcal{M}_n^\alpha(T)$ denotes the probability that under strategy α, after n steps the MDP $\mathcal{M}$ is in a state of T).

Definition 3.4 (Synchronized probability distributions)**.** *For the function $f \in \{sum_T, max_T\}$ and $p \in [0,1]$, we say that a probability distribution X is p-synchronized according to f if $f(X) \geq p$.*

We say that a sequence $\bar{X} = X_0 X_1 \ldots$ of probability distributions is:
(*a*) *always p-synchronizing* if X_i is p-synchronized for all $i \geq 0$;
(*b*) *event* (or *eventually*) *p-synchronizing* if X_i is p-synchronized for some $i \geq 0$;
(*c*) *weakly p-synchronizing* if X_i is p-synchronized for infinitely many i's;
(*d*) *strongly p-synchronizing* if X_i is p-synchronized for all but finitely many i's.

For $p = 1$, these definitions are analogous to the traditional safety, reachability, Büchi, and coBüchi conditions [dAH00].

Similar to traditional semantics, the following winning modes can be considered:

Definition 3.5 (Synchronizing MDPs)**.** *Let $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ be an MDP with the initial distribution X_0 and let $f \in \{sum_T, max_T\}$. For the {always, eventually, weakly, strongly} synchronizing condition, $\mathcal{M}$ is:*

- sure winning *if there exists a strategy α such that the symbolic-outcome of α from X_0 is {always, eventually, weakly, strongly} 1-synchronizing according to f;*
- almost-sure winning *if there exists a strategy α such that for all $\epsilon > 0$ the symbolic-outcome of α from X_0 is {always, eventually, weakly, strongly} $(1-\epsilon)$-synchronizing according to f;*
- limit-sure winning *if for all $\epsilon > 0$, there exists a strategy α such that the symbolic-outcome of α from X_0 is {always, eventually, weakly, strongly} $(1-\epsilon)$-synchronizing according to f.*

We often use $X(T)$ instead of $sum_T(X)$, as in Tables 3.1 where the definitions of the various winning modes and synchronizing conditions for $f = sum_T$ are summarized, and we use $\|X\|_T$ instead of $max_T(X)$.

For $f \in \{sum_T, max_T\}$ and $\lambda \in \{\text{always, event(ually), weakly, strongly}\}$, the *winning region* $\langle\langle 1\rangle\rangle^{\lambda}_{sure}(f)$ is the set of initial distributions such that $\mathcal{M}$ is sure winning for λ-synchronization (we assume that $\mathcal{M}$ is clear from the context). We define analogously the winning regions $\langle\langle 1\rangle\rangle^{\lambda}_{almost}(f)$ and $\langle\langle 1\rangle\rangle^{\lambda}_{limit}(f)$. For a singleton $T = \{q\}$, we have $sum_T = max_T$, and we simply write $\langle\langle 1\rangle\rangle^{\lambda}_{\mu}(q)$ where $\mu \in \{\text{sure, almost, limit}\}$.

Example 3.3. *Consider the MDP $\mathcal{M}$ described in Example 2.3 on page 21 and the memoryless strategy γ where $\gamma(q_0) = a$ and $\gamma(q) = b$ for all states $q \neq q_0$ (defined in Example 2.5 on page 24). The outcome from the initial Dirac distribution q_0 is as follows*

$$\begin{array}{c} q_0 \\ q_1 \\ q_2 \\ q_3 \end{array} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{2} \\ 0 \end{pmatrix} \begin{pmatrix} \frac{1}{8} \\ \frac{1}{8} \\ \frac{1}{4} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} \frac{1}{16} \\ \frac{1}{16} \\ \frac{1}{8} \\ \frac{3}{4} \end{pmatrix} \begin{pmatrix} \frac{1}{32} \\ \frac{1}{32} \\ \frac{1}{16} \\ \frac{7}{8} \end{pmatrix} \cdots \begin{pmatrix} \frac{1}{2^k} \\ \frac{1}{2^k} \\ \frac{1}{2^{k-1}} \\ 1 - \frac{1}{2^{k-2}} \end{pmatrix} \cdots$$

where $k \geq 3$ is the number of steps. We have $\sup_{k\in\mathbb{N}} \mathcal{M}^{\gamma}_k(q_3) = \sup_{k\in\mathbb{N}}(1 - \frac{1}{2^{k-2}}) = 1$ showing that $q_0 \in \langle\langle 1\rangle\rangle^{event}_{almost}(q_3)$.

Consider the pure memoryless strategy β that always plays b in all states: $\beta(q) = b$ for all states $q \in Q$. The outcome is such that for all $k \in \mathbb{N}$, the probability $\mathcal{M}^{\beta}_k(q_0) = 1$ from q_0 which trivially gives $q_0 \in \langle\langle 1\rangle\rangle^{always}_{sure}(q_0)$. The MDP $\mathcal{M}$ is sure winning for always synchronizing condition since in all steps k, there is a deterministic transition ensuring the existence of a unique successor, and thus ensuring that the probability distribution $\mathcal{M}^{\beta}_{k+1}$ over states remains a Dirac distribution. By definition, if q_0 is sure winning for always synchronizing condition in $\{q_0\}$, it is sure winning for eventually synchronizing condition too. Thus, $\langle\langle 1\rangle\rangle^{event}_{sure}(max_T) \neq \emptyset$ for the target set $T = \{q_0, q_1\}$.

Now, let us modify the MDP $\mathcal{M}$ as follows. The b-transition in q_0 is not deterministic anymore: $\delta(q_0, b)(q_0) = \delta(q_0, b)(q_1) = \frac{1}{2}$ (shown in Figure 3.7). We can easily verify that $q_0 \notin \langle\langle 1\rangle\rangle^{always}_{sure}(q_0)$ and $q_0 \notin \langle\langle 1\rangle\rangle^{always}_{sure}(q_1)$ neither. This two results are sufficient to prove that $q_0 \notin \langle\langle 1\rangle\rangle^{always}_{sure}(max_T)$ for the target set $T = \{q_0, q_1\}$. However, a pure memoryless strategy that always plays a for all states $q \in Q$, is a witness to prove $q_0 \in \langle\langle 1\rangle\rangle^{always}_{sure}(sum_T)$.

◁

The definitions of synchronization result in following hierarchies:

Remark 4. Given an MDP $\mathcal{M}$, for all functions $f \in \{sum_T, max_T\}$, for all $\lambda \in \{\text{always, event, weakly, strongly}\}$, and $\mu \in \{\text{sure, almost, limit}\}$, we have:

– $\langle\langle 1\rangle\rangle^{\lambda}_{sure}(f) \subseteq \langle\langle 1\rangle\rangle^{\lambda}_{almost}(f) \subseteq \langle\langle 1\rangle\rangle^{\lambda}_{limit}(f)$,

– $\langle\langle 1\rangle\rangle^{always}_{\mu}(f) \subseteq \langle\langle 1\rangle\rangle^{strongly}_{\mu}(f) \subseteq \langle\langle 1\rangle\rangle^{weakly}_{\mu}(f) \subseteq \langle\langle 1\rangle\rangle^{event}_{\mu}(f)$

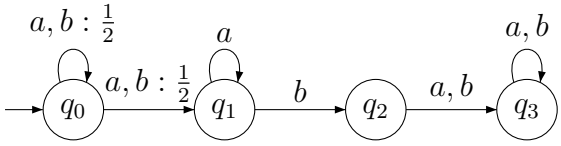

Figure 3.7: An MDP $\mathcal{M}$ such that $\langle\langle 1\rangle\rangle^{event}_{almost}(q_2) \neq \langle\langle 1\rangle\rangle^{event}_{limit}(q_2)$ and $\langle\langle 1\rangle\rangle^{\lambda}_{sure}(q_1) \neq \langle\langle 1\rangle\rangle^{\lambda}_{almost}(q_1)$ for all $\lambda \in \{\text{event, weakly, strongly}\}$.

For the always synchronizing condition we will provide robustness result saying that the three winning modes coincide: $\langle\langle 1\rangle\rangle^{always}_{sure}(f) = \langle\langle 1\rangle\rangle^{always}_{almost}(f) = \langle\langle 1\rangle\rangle^{always}_{limit}(f)$ (See Lemma 4.1 on page 64). For the eventually synchronizing condition as the least robust synchronization, we will prove that the inclusions $\langle\langle 1\rangle\rangle^{event}_{sure}(f) \subset \langle\langle 1\rangle\rangle^{event}_{almost}(f) \subset \langle\langle 1\rangle\rangle^{event}_{limit}(f)$ are in general strict (See Lemma 3.1 on page 49). The strong and weak synchronization are more robust than eventually synchronization, but less than always synchronization: we see that $\langle\langle 1\rangle\rangle^{\lambda}_{sure}(f) \subset \langle\langle 1\rangle\rangle^{\lambda}_{almost}(f)$ where $\lambda \in \{\text{strongly,weakly}\}$; and we will prove that the almost-sure and limit-sure winning modes coincide (See Lemma 3.1 , Theorem 5.3 and Corollaries 6.1 and 6.2 on pages 49, 100, 119 and 123, respectively.).

Lemma 3.1. *For all $\lambda \in$ {event, weakly, strongly}, there exists an MDP $\mathcal{M}$ and two states q_1, q_2 such that:*

(i) $\langle\langle 1\rangle\rangle^{\lambda}_{sure}(q_1) \subsetneq \langle\langle 1\rangle\rangle^{\lambda}_{almost}(q_1)$ *and,*

(ii) $\langle\langle 1\rangle\rangle^{event}_{almost}(q_2) \subsetneq \langle\langle 1\rangle\rangle^{event}_{limit}(q_2)$.

Proof. Consider the MDP $\mathcal{M}$ with states q_0, q_1, q_2, q_3 and actions a, b as shown in Figure 3.7. All transitions are deterministic except from q_0 where on all actions, the successor is q_0 or q_1 with probability $\frac{1}{2}$. The deterministic a and b-transitions in q_1, q_2, q_3 are as follows,

$$\delta(q_1, a) = q_1, \quad \delta(q_1, b) = q_2 \quad \text{and} \quad \delta(q_2, c) = \delta(q_3, c) = q_3 \text{ for } c \in \{a, b\}.$$

Let the initial distribution X_0 be the Dirac distribution on q_0.

To establish (i), since $\langle\langle 1\rangle\rangle^{strongly}_{\mu}(q_1) \subseteq \langle\langle 1\rangle\rangle^{weakly}_{\mu}(q_1) \subseteq \langle\langle 1\rangle\rangle^{event}_{\mu}(q_1)$ for all $\mu \in \{\text{sure, almost}\}$, it is sufficient to prove that $X_0 \in \langle\langle 1\rangle\rangle^{strongly}_{almost}(q_1)$ and $X_0 \notin \langle\langle 1\rangle\rangle^{event}_{sure}(q_1)$, as it implies that $X_0 \in \langle\langle 1\rangle\rangle^{\lambda}_{almost}(q_1)$ and $X_0 \notin \langle\langle 1\rangle\rangle^{\lambda}_{sure}(q_1)$ for all $\lambda \in \{\text{event, weakly, strongly}\}$. To prove that $X_0 \in \langle\langle 1\rangle\rangle^{strongly}_{almost}(q_1)$, consider the pure strategy that always plays a. The outcome is such that the probability to be in q_1 after k steps is $1 - \frac{1}{2^k}$, showing that $\mathcal{M}$ is almost-sure winning for the strongly synchronizing condition in the target set $\{q_1\}$ (from X_0). On the other hand, $X_0 \notin \langle\langle 1\rangle\rangle^{event}_{sure}(q_1)$ because for all strategies α, the probability in q_0 remains always positive, and thus in q_1 we have $\mathcal{M}^{\alpha}_n(q_1) < 1$ for all $n \geq 0$, showing that $\mathcal{M}$ is not sure winning for the eventually synchronizing condition in q_1 (from X_0).

To establish (ii), for all $k \geq 0$ consider a strategy that plays a for k steps, and then plays b. Then the probability to be in q_2 after $k+1$ steps is $1-\frac{1}{2^k}$, showing that this strategy is eventually $(1 - \frac{1}{2^k})$-synchronizing in the target set $\{q_2\}$. Hence, $\mathcal{M}$ is limit-sure winning for the eventually synchronizing condition in q_2 (from X_0). Second, for all strategies, since

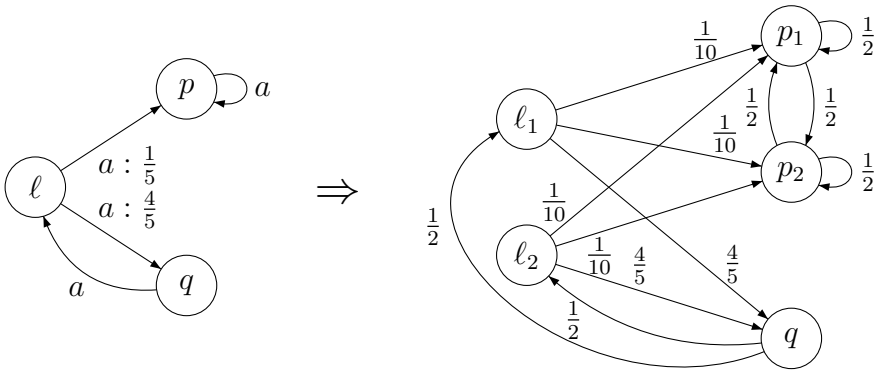

Figure 3.8: State duplication ensures that the probability mass can never be accumulated in a single state except in q (we omit action a for readability).

the probability in q_0 remains always positive, the probability in q_2 is always smaller than 1. Moreover, if the probability p in q_2 is positive after n steps ($p > 0$), then after any number $m > n$ of steps, the probability in q_2 is bounded by $1 - p$. It follows that the probability in $\{q_2\}$ is never equal to 1 and cannot tend to 1 for $m \to \infty$, showing that $\mathcal{M}$ is not almost-sure winning for the eventually synchronizing condition in $\{q_2\}$ (from X_0).

□

Decision problem(s). For $\lambda \in \{$always, event, strongly,weakly$\}$ and $\mu \in \{$sure, almost, limit$\}$, the *membership problem* of λ-synchronization is to decide, given an MDP $\mathcal{M}$ and an initial probability distribution X_0, whether $X_0 \in \langle\langle 1 \rangle\rangle^{\lambda}_{\mu}(f)$.

Remark 5. For $\lambda \in \{$eventually, weakly$\}$ synchronizing condition and each winning mode, we show that the membership problem with function max_T is polynomial-time equivalent to the membership problem with function $sum_{T'}$ with a singleton T'. First, for $\mu \in \{$sure, almost, limit$\}$, we have $\langle\langle 1 \rangle\rangle^{\lambda}_{\mu}(max_T) = \bigcup_{q \in T} \langle\langle 1 \rangle\rangle^{\lambda}_{\mu}(q)$, showing that the membership problems for max are polynomial-time reducible to the corresponding membership problem for $sum_{T'}$ with singleton T'. The reverse reduction is as follows. Given an MDP $\mathcal{M}'$, a singleton $T' = \{q\}$ and an initial distribution X'_0, we can construct an MDP $\mathcal{M}$ and initial distribution X_0 such that $X'_0 \in \langle\langle 1 \rangle\rangle^{\lambda}_{\mu}(q)$ if and only if $X_0 \in \langle\langle 1 \rangle\rangle^{\lambda}_{\mu}(max_T)$ where $T = Q$ is the state space of $\mathcal{M}$. The idea is to construct $\mathcal{M}$ and X_0 as a copy of $\mathcal{M}'$ and X'_0 where all states except q are duplicated, and the initial and transition probabilities are evenly distributed between the copies (see Figure 3.8). Therefore if the probability tends to 1 in some state, it has to be in the state q.

Remark 6. For $\lambda \in \{$always, event, strongly, weakly$\}$, for the *membership problem* of λ-synchronization it is sufficient to consider Dirac initial distributions (i.e., assuming that

MDPs have a single initial state) because the answer to the general membership problem for an MDP $\mathcal{M}$ with initial distribution X_0 can be obtained by solving the membership problem for a copy of $\mathcal{M}$ with a new initial state q_{init} from which the successor distribution on all actions is X_0. We, thus, always equip the MDP with an initial Dirac distribution q_{init} and study synchronization from q_{init}. Note that in the case of always synchronizing MDPs, the target set is replaced with $T \cup \{q_{\mathsf{init}}\}$, and in the case of eventually synchronizing MDPs the non-trivial synchronization is of interest.

3.3 Synchronization in PAs

Since words for probabilistic automata can be viewed as blind strategies for MDPs, we define symbolic-runs of PAs as the symbolic outcomes of MDPs with blind strategies. In the context of synchronization, we consider randomized words for PAs.

For an infinite randomized word $w = d_0 d_1 d_2 \cdots$ where $d_i \in \mathcal{D}(\mathsf{A})$ for all $i \in \mathbb{N}$, the symbolic-run $\mathcal{P}_0^w \mathcal{P}_1^w \mathcal{P}_2^w \cdots$ of the PA $\mathcal{P}$ from the initial distribution X_0 is defined by $\mathcal{P}_0^w = X_0$, and for all $n \in \mathbb{N}$ and $q \in Q$, by

$$\mathcal{P}_{n+1}^w(q) = \sum_{a \in \mathsf{A}} \sum_{q' \in Q} \mathcal{P}_n^w(q') \cdot d_n(a) \cdot \delta(q', a)(q).$$

By replacing the path-outcomes with run-outcomes and strategies with randomized words, PAs inherit all definitions of synchronizations (such as sure weakly synchronizing condition) from MDPs. For all $\lambda \in \{\text{always, eventually, strongly, weakly}\}$ and $f \in \{max_T, sum_T\}$, we thus define {sure, almost} λ-synchronizing languages according to f for PAs $\mathcal{P}$, as $\mathcal{L}^\lambda_{sure}(f)$ and $\mathcal{L}^\lambda_{almost}(f)$, that are the set of all randomized words (blind strategies) that $\mathcal{P}$ is sure and almost-sure winning for the λ-synchronizing condition. We also define the limit-sure $\mathcal{L}^\lambda_{limit}(f)$ λ-synchronizing languages, denoted by $\mathcal{L}^\lambda_{limit}(f)$, that is the set of all families of limit-sure words such that in each family, for all ϵ there is a word w over which $\mathcal{P}$ is $(1-\epsilon)$-synchronizing for λ-synchronizing condition.

Decision problem(s). For $\lambda \in \{\text{always, event, weakly, strongly}\}$ and $f \in \{sum_T, max_T\}$, the *emptiness problem* for {sure, almost, limit} λ-synchronizing languages according to f asks, given a PA $\mathcal{P}$ and an initial distribution X_0, whether the language is empty.

By Remark 6, without loss of generality we assume that X_0 is a Dirac distribution on the state q_{init}.

Example 3.4. *Consider the MDP drawn in Figure 3.9 with the initial state q_{init}. All transitions are deterministic except in q_{init} where on all actions, the successors are q_1 and q_2 each with probability $\frac{1}{2}$. In q_1 and q_2, the b-transitions are self-loops whereas the a-transitions leave the states to q_2 and q_3, respectively. The state q_3 is an absorbing state.*

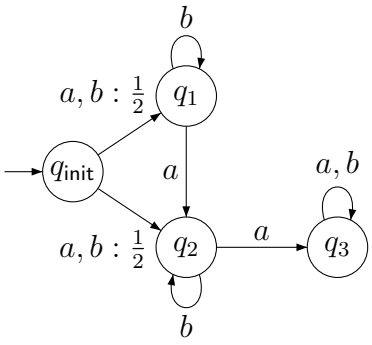

Figure 3.9: An MDP such that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(q_2)$ implying that the MDP is sure eventually synchronizing in $\{q_2\}$ by some strategy. However, there is no blind strategy under which the MDP is sure eventually synchronizing in $\{q_2\}$, and thus $\mathcal{L}^{event}_{sure}(q_2) = \emptyset$ from q_{init} when interpreting the model as a PA.

We prove that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(q_2)$ implying that there is some strategy for $\mathcal{M}$ to win sure eventually synchronizing condition in $\{q_2\}$, and we prove that there is no blind strategy to win this condition. This implies that $\mathcal{L}^{event}_{sure}(q_2) = \emptyset$ from q_{init} when interpreting the model as a PA.

Consider the pure memoryless strategy α that plays a in all states except q_2 where it plays b: $\alpha(q_2) = b$ and $\alpha(q) = a$ for all states $q \neq q_2$. The symbolic-outcome of $\mathcal{M}$ under α is

$$\begin{array}{c} q_{\mathsf{init}} \\ q_1 \\ q_2 \\ q_3 \end{array} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \dots$$

meaning that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(q_2)$ in the MDP $\mathcal{M}$.

Considering the model as the PA $\mathcal{P}$, the first input letter brings the automaton in q_1 or q_2 with equal probability $\frac{1}{2}$. From now on, if $\mathcal{P}$ always inputs b, the probability $\frac{1}{2}$ in q_1 never leaks down to q_2, and $\mathcal{P}$ can never be in q_2 with probability 1. On the other hand, as soon as $\mathcal{P}$ inputs once a, the automaton moves from q_2 to q_3 with probability $\frac{1}{2}$. Since q_3 is absorbing, the automaton never comes back from q_3 to q_2; thus, the probability in q_2 is never more than $\frac{1}{2}$ and $\mathcal{L}^{event}_{sure}(q_2) = \emptyset$ from q_{init}.

◁

3.4 Relation to one-letter alternating finite automata

Let $\mathsf{B}^+(Q)$ be the set of positive Boolean formulas over Q, i.e., Boolean formulas built from elements in Q using $\wedge$ and $\vee$. A set $S \subseteq Q$ *satisfies* a formula $\varphi \in \mathsf{B}^+(Q)$ (denoted $S \models \varphi$) if φ is satisfied when replacing in φ the elements in S by true, and the elements in $Q \setminus S$ by false.

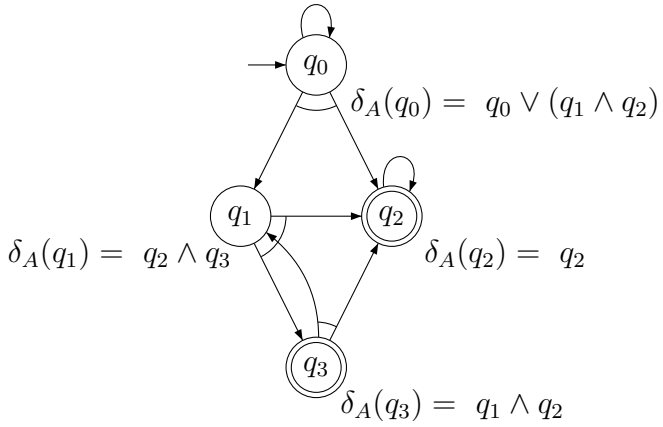

Figure 3.10: An example of 1L-AFA where the language $\mathcal{L}(\mathcal{A}_{q_0})$ is not finite.

Definition 3.6 (One-letter alternating finite automaton)**.** *A* one-letter alternating finite automaton *(1L-AFA) is a tuple $\mathcal{A} = \langle Q, \delta_{\mathcal{A}}, \mathcal{F} \rangle$ where Q is a finite set of states, $\delta_{\mathcal{A}} : Q \to \mathsf{B}^+(Q)$ is the transition function, and $\mathcal{F} \subseteq Q$ is the set of accepting states.*

We assume that the formulas in transition function are in disjunctive normal form. Note that the alphabet of the automaton is omitted, as it has a single letter. In the language of a 1L-AFA, only the length of words is relevant. For all $n \geq 0$, define the set $Acc_{\mathcal{A}}(n, \mathcal{F}) \subseteq Q$ of states from which the word of length n is accepted by $\mathcal{A}$ as follows:

- $Acc_{\mathcal{A}}(0, \mathcal{F}) = \mathcal{F}$;
- $Acc_{\mathcal{A}}(n, \mathcal{F}) = \{q \in Q \mid Acc_{\mathcal{A}}(n-1, \mathcal{F}) \models \delta(q)\}$ for all $n > 0$.

The set $\mathcal{L}(\mathcal{A}_q) = \{n \in \mathbb{N} \mid q \in Acc_{\mathcal{A}}(n, \mathcal{F})\}$ is the *language* accepted by $\mathcal{A}$ from initial state q.

Example 3.5. *Consider the 1L-AFA depicted in Figure 3.10 which has four states q_0, q_1, q_2 and q_3. The transition function is defined as follows. In q_0, there is a self-loop and another transition going to q_1 and q_2: $\delta_{\mathcal{A}}(q_0) = q_0 \vee (q_1 \wedge q_2)$. The state q_2 has a self-loop. In q_1 and q_3, there are only $\wedge$-transitions: $\delta_{\mathcal{A}}(q_1) = q_2 \wedge q_3$ and $\delta_{\mathcal{A}}(q_3) = q_1 \wedge q_2$. Let $\mathcal{F} = \{q_2, q_3\}$ be the set of accepting states. By definition, we have $Acc_{\mathcal{A}}(0, \{q_2, q_3\}) = \{q_2, q_3\}$ and*

$$Acc_{\mathcal{A}}(1, \mathcal{F}) = \{q_1, q_2\} \text{ and } Acc_{\mathcal{A}}(2, \mathcal{F}) = \{q_0, q_2, q_3\}.$$

Since q_0 has a self-loop, then $q_0 \in Acc_{\mathcal{A}}(n, \mathcal{F})$ for all next iterations $n \geq 2$. It implies that the language $\mathcal{L}(\mathcal{A}_{q_0})$ of $\mathcal{A}$ starting in q_0 is not finite.

$\triangleleft$

For fixed n, we view $Acc_{\mathcal{A}}(n, \cdot)$ as an operator on 2^Q that, given a set $\mathcal{F} \subseteq Q$ computes the set $Acc_{\mathcal{A}}(n, \mathcal{F})$. Note that $Acc_{\mathcal{A}}(n, \mathcal{F}) = Acc_{\mathcal{A}}(1, Acc_{\mathcal{A}}(n-1, \mathcal{F}))$ for all $n \geq 1$. Denote by $\mathsf{Pre}_{\mathcal{A}}(\cdot)$ the operator $Acc_{\mathcal{A}}(1, \cdot)$. Then for all $n \geq 0$ the operator $Acc_{\mathcal{A}}(n, \cdot)$

coincides with $\mathsf{Pre}^n_{\mathcal{A}}(\cdot)$, the n-th iterate of $\mathsf{Pre}_{\mathcal{A}}(\cdot)$. For the 1L-AFA of Example 3.5, we have $\mathsf{Pre}^3_{\mathcal{A}}(\{q_2, q_3\}) = \{q_0, q_1, q_2\}$ and $\mathsf{Pre}^4_{\mathcal{A}}(\{q_2, q_3\}) = \{q_0, q_2, q_3\}$. For the next iterations, we see that for all $n \geq 1$

$$\mathsf{Pre}^{2n+1}_{\mathcal{A}}(\{q_2, q_3\}) = \mathsf{Pre}^3_{\mathcal{A}}(\{q_2, q_3\}) \text{ and } \mathsf{Pre}^{2n+2}_{\mathcal{A}}(\{q_2, q_3\}) = \mathsf{Pre}^2_{\mathcal{A}}(\{q_2, q_3\}).$$

Thus, the infinite sequence $\mathsf{Pre}^0_{\mathcal{A}}(\mathcal{F})\mathsf{Pre}^1_{\mathcal{A}}(\mathcal{F})\mathsf{Pre}^2_{\mathcal{A}}(\mathcal{F})\cdots$ is ultimately periodic. This sequence is ultimately periodic for all 1L-AFAs: there exists $n, k \in \mathbb{N}$ such that $\mathsf{Pre}^{n+k}_{\mathcal{A}}(\mathcal{F}) = \mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F})$.

3.4.1 Problems for 1L-AFA

We present classical decision problems for alternating automata, namely the emptiness and finiteness problems, and we introduce a variant of the finiteness problem that will be useful to give the complexity of synchronizing problems for (two player games and) MDPs.

Decision problem(s). Given a 1L-AFA $\mathcal{A}$ and an initial state q, the *emptiness problem* for 1L-AFAs is to decide whether $\mathcal{L}(\mathcal{A}_q) = \emptyset$, and the *finiteness problem* is to decide whether $\mathcal{L}(\mathcal{A}_q)$ is finite. The *universal finiteness problem* is to decide, given a 1L-AFA $\mathcal{A}$, whether $\mathcal{L}(\mathcal{A}_q)$ is finite for all states q.

In sequel, we discuss all of these three decision problems.

Emptiness problem. Given a 1L-AFA $\mathcal{A}$, the sequence $\mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F})$ is ultimately periodic: for all $n > 2^{|Q|}$ there exists $k \leq 2^{|Q|}$ such that $\mathsf{Pre}^k_{\mathcal{A}}(\mathcal{F}) = \mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F})$. Therefore, a simple PSPACE algorithm to decide the emptiness problem is to compute on the fly the sets $\mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F})$ for $n \leq 2^{|Q|}$, and check membership of q. The emptiness problem can be solved by checking whether $q \in \mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F})$ for some $n \geq 0$. It is known that the emptiness problem is PSPACE-complete, even for transition functions in disjunctive normal form [Hol95, JS07].

Finiteness problem. This problem can be solved in (N)PSPACE by guessing $n, k \leq 2^{|Q|}$ such that $\mathsf{Pre}^{n+k}_{\mathcal{A}}(\mathcal{F}) = \mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F})$ and $q \in \mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F})$. The finiteness problem is PSPACE-complete by a simple reduction from the emptiness problem: from an instance $(\mathcal{A}, q)$ of the emptiness problem, construct $(\mathcal{A}', q')$ where $q' = q$ and $\mathcal{A}' = \langle Q, \delta', \mathcal{F}\rangle$ is a copy of $\mathcal{A} = \langle Q, \delta, \mathcal{F}\rangle$ with a self-loop on q (formally, $\delta'(q) = q \vee \delta(q)$ and $\delta'(r) = \delta(r)$ for all $r \in Q \setminus \{q\}$). It is easy to see that $\mathcal{L}(\mathcal{A}_q) = \emptyset$ if and only if $\mathcal{L}(\mathcal{A}'_{q'})$ is finite.

Universal finiteness problem. This problem can be solved in PSPACE by checking whether $\mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F}) = \emptyset$ for some $n \leq 2^{|Q|}$. Note that if $\mathsf{Pre}^n_{\mathcal{A}}(\mathcal{F}) = \emptyset$, then $\mathsf{Pre}^m_{\mathcal{A}}(\mathcal{F}) = \emptyset$ for all $m \geq n$. Given the PSPACE-hardness proofs of the emptiness and finiteness problems, it is not easy to see that the universal finiteness problem is PSPACE-hard.

Lemma 3.2. *The universal finiteness problem for 1L-AFA is* PSPACE-hard.

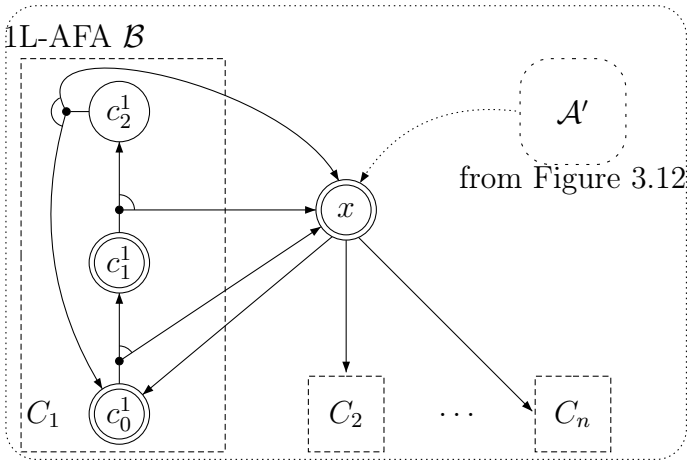

Figure 3.11: Sketch of reduction to show PSPACE-hardness of the universal finiteness problem for 1L-AFA.

Proof. The proof is by a reduction from the emptiness problem for 1L-AFA, which is PSPACE-complete [Hol95, JS07]. The language of a 1L-AFA $\mathcal{A} = \langle Q, \delta, \mathcal{F} \rangle$ is non-empty if $q_0 \in \mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$ for some $i \geq 0$. Since the sequence $\mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$ is ultimately periodic, it is sufficient to compute $\mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$ for all $i \leq 2^{|Q|}$.

From $\mathcal{A}$, we construct a 1L-AFA $\mathcal{B} = \langle Q', \delta', \mathcal{F}' \rangle$ with set $\mathcal{F}'$ of accepting states such that the sequence $\mathsf{Pre}^i_{\mathcal{B}}(\mathcal{F}')$ in $\mathcal{B}$ mimics the sequence $\mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$ in $\mathcal{A}$ for $2^{|Q|}$ steps. The automaton $\mathcal{B}$ contains the state space of $\mathcal{A}$, i.e., $Q \subseteq Q'$. The goal is to have $\mathsf{Pre}^i_{\mathcal{B}}(\mathcal{F}') \cap Q = \mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$ for all $i \leq 2^{|Q|}$, as long as $q_0 \notin \mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$. Moreover, if $q_0 \in \mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$ for some $i \geq 0$, then $\mathsf{Pre}^j_{\mathcal{B}}(\mathcal{F}')$ will contain q_0 for all $j \geq i$ (the state q_0 has a self-loop in $\mathcal{B}$), and if $q_0 \notin \mathsf{Pre}^i_{\mathcal{A}}(\mathcal{F})$ for all $i \geq 0$, then $\mathcal{B}$ is constructed such that $\mathsf{Pre}^j_{\mathcal{B}}(\mathcal{F}') = \emptyset$ for sufficiently large j (roughly for $j > 2^{|Q|}$). Hence, the language of $\mathcal{A}$ is non-empty if and only if the sequence $\mathsf{Pre}^j_{\mathcal{B}}(\mathcal{F}')$ is not ultimately empty, that is if and only if the language of $\mathcal{B}$ is infinite from some state (namely q_0).

The key is to let $\mathcal{B}$ simulate $\mathcal{A}$ for exponentially many steps, and to ensure that the simulation stops if and only if q_0 is not reached within $2^{|Q|}$ steps. We achieve this by defining $\mathcal{B}$ as the gadget in Figure 3.11 connected to a modified copy $\mathcal{A}'$ of $\mathcal{A}$ with the same state space. The transitions in $\mathcal{A}'$ are defined as follows, where x is the entry state of the gadget (see Figure 3.12): for all $q \in Q$ let (i) $\delta_{\mathcal{B}}(q) = x \wedge \delta_{\mathcal{A}}(q)$ if $q \neq q_0$, and (ii) $\delta_{\mathcal{B}}(q_0) = q_0 \vee (x \wedge \delta_{\mathcal{A}}(q_0))$.

Thus, q_0 has a self-loop, and given a set $S \subseteq Q$ in the automaton $\mathcal{A}$, if $q_0 \notin S$, then $\mathsf{Pre}_{\mathcal{A}}(S) = \mathsf{Pre}_{\mathcal{B}}(S \cup \{x\})$ that is $\mathsf{Pre}_{\mathcal{B}}$ mimics $\mathsf{Pre}_{\mathcal{A}}$ when x is in the argument (and q_0 has not been reached yet). Note that if $x \notin S$ (and $q_0 \notin S$), then $\mathsf{Pre}_{\mathcal{B}}(S) = \emptyset$, that is unless q_0 has been reached, the simulation of $\mathcal{A}$ by $\mathcal{B}$ stops. Since we need that $\mathcal{B}$ mimics $\mathcal{A}$ for $2^{|Q|}$ steps, we define the gadget and the set $\mathcal{F}'$ to ensure that $x \in \mathcal{F}'$ and if $x \in \mathsf{Pre}^i_{\mathcal{B}}(\mathcal{F}')$, then $x \in \mathsf{Pre}^{i+1}_{\mathcal{B}}(\mathcal{F}')$ for all $i \leq 2^{|Q|}$.

In the gadget, the state x has non-deterministic transitions $\delta_{\mathcal{B}}(x) = c_0^1 \vee c_0^2 \vee \cdots \vee c_0^n$ to n

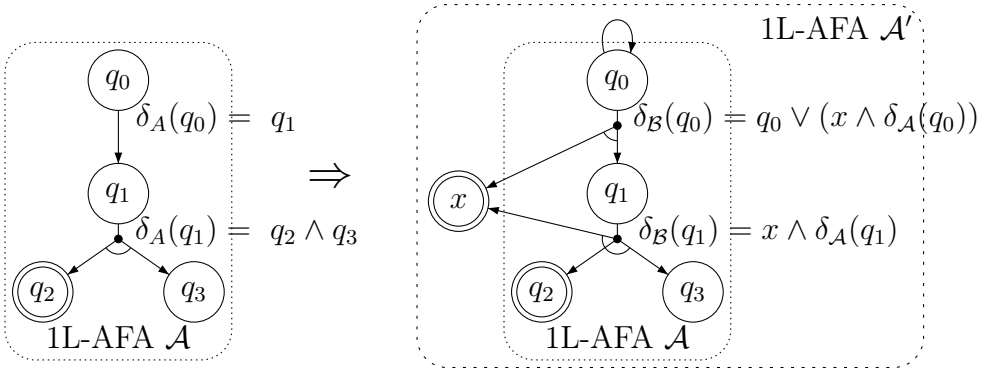

Figure 3.12: Detail of the copy $\mathcal{A}'$ obtained from $\mathcal{A}$ in the reduction of Figure 3.11.

components with state space $C_i = \{c_0^i, \ldots, c_{p_i-1}^i\}$ where p_i is the $(i+1)$-th prime number, and the transitions [1] $\delta_{\mathcal{B}}(c_j^i) = x \wedge c_{j+1}^i$ form a loop in each component $(i = 1, \ldots, n)$. We choose n such that $p_n^{\#} = \prod_{i=1}^{n} p_i > 2^{|Q|}$ (take $n = |Q|$). Note that the number of states in the gadget is $1 + \sum_{i=1}^{n} p_i \in O(n^2 \log n)$ [BS96] and hence the construction is polynomial in the size of $\mathcal{A}$.

By construction, for all sets S, we have $x \in \mathsf{Pre}_{\mathcal{B}}(S)$ whenever the first state c_0^i of some component C_i is in S, and if $x \in S$, then $c_j^i \in S$ implies $c_{j-1}^i \in \mathsf{Pre}_{\mathcal{B}}(S)$. Thus, if $x \in S$, the operator $\mathsf{Pre}_{\mathcal{B}}(S)$ 'shifts' backward the states in each component; and, x is in the next iteration (i.e., $x \in \mathsf{Pre}_{\mathcal{B}}(S)$) as long as $c_0^i \in S$ for some component C_i.

Now, define the set of accepting states $\mathcal{F}'$ in $\mathcal{B}$ in such a way that all states c_0^i disappear simultaneously only after $p_n^{\#}$ iterations. Let $\mathcal{F}' = \mathcal{F} \cup \{x\} \cup \bigcup_{1 \le i \le n}(C_i \setminus \{c_{p_i-1}^i\})$, thus $\mathcal{F}'$ contains all states of the gadget except the last state of each component. It is easy to check that, irrespective of the transition relation in $\mathcal{A}$, we have $x \in \mathsf{Pre}_{\mathcal{B}}^i(\mathcal{F}')$ if and only if $0 \le i < p_n^{\#}$. Therefore, if $q_0 \in \mathsf{Pre}_{\mathcal{A}}^i(\mathcal{F})$ for some i, then $q_0 \in \mathsf{Pre}_{\mathcal{B}}^j(\mathcal{F}')$ for all $j \ge i$ by the self-loop on q_0. On the other hand, if $q_0 \notin \mathsf{Pre}_{\mathcal{A}}^i(\mathcal{F})$ for all $i \ge 0$, then since $x \notin \mathsf{Pre}_{\mathcal{B}}^i(\mathcal{F}')$ for all $i > p_n^{\#}$, we have $\mathsf{Pre}_{\mathcal{B}}^i(\mathcal{F}') = \emptyset$ for all $i > p_n^{\#}$. This shows that the language of $\mathcal{A}$ is non-empty if and only if the language of $\mathcal{B}$ is infinite from some state (namely q_0), and establishes the correctness of the reduction.

□

3.4.2 Connection with MDPs.

The underlying structure of a Markov decision process $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ is an alternating graph, where the successor q' of a state q is obtained by an existential choice of an action a and a universal choice of a state $q' \in \mathsf{Supp}(\delta(q, a))$. Therefore, it is natural that some questions related to MDPs have a corresponding formulation in terms of alternating automata. We show that such connections exist between synchronizing problems for

1. In expression c_j^i, we assume that j is interpreted modulo p_i.

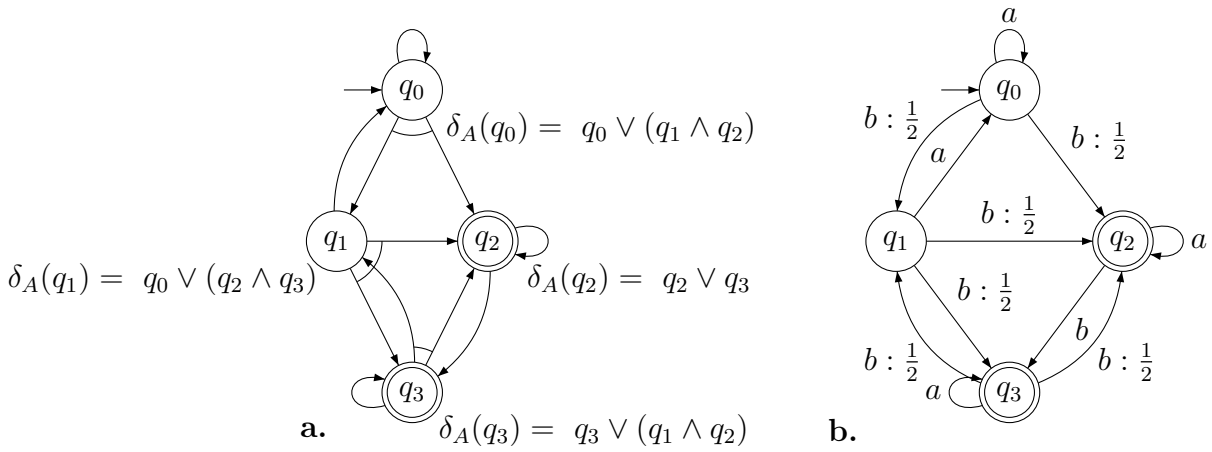

Figure 3.13: An example of 1L-AFA $\mathcal{A}$ with the connected MDP $\mathcal{M}_{\mathcal{A}}$.

MDPs and language-theoretic questions for alternating automata, such as emptiness and universal finiteness. Given a 1L-AFA $\mathcal{A} = \langle Q, \delta_{\mathcal{A}}, \mathcal{F} \rangle$, assume without loss of generality that the transition function $\delta_{\mathcal{A}}$ is such that $\delta_{\mathcal{A}}(q) = c_1 \vee \cdots \vee c_m$ has the same number m of conjunctive clauses for all $q \in Q$. From $\mathcal{A}$, construct the MDP $\mathcal{M}_{\mathcal{A}} = \langle Q, \mathsf{A}, \delta_{\mathcal{M}} \rangle$ where $\mathsf{A} = \{a_1, \ldots, a_m\}$ and $\delta_{\mathcal{M}}(q, a_k)$ is the uniform distribution over the states occurring in the k-th clause c_k in $\delta_{\mathcal{A}}(q)$, for all $q \in Q$ and $a_k \in \mathsf{A}$. Then for all $n \geq 0$, we have $Acc_{\mathcal{A}}(n, \mathcal{F}) = \mathsf{Pre}^n(\mathcal{F})$ in the MDP $\mathcal{M}_{\mathcal{A}}$. Similarly, from an MDP $\mathcal{M}$ and a set T of states, we can construct a 1L-AFA $\mathcal{A} = \langle Q, \delta_{\mathcal{A}}, \mathcal{F} \rangle$ with $\mathcal{F} = T$ such that for all $n \geq 0$ we have $Acc_{\mathcal{A}}(n, \mathcal{F}) = \mathsf{Pre}^n(T)$ in the MDP $\mathcal{M}$ (let $\delta_{\mathcal{A}}(q) = \bigvee_{a \in \mathsf{A}} \bigwedge_{q' \in \mathsf{post}(q,a)} q'$ for all $q \in Q$).

Example 3.6. *Consider the 1L-AFA $\mathcal{A}$ depicted in Figure 3.13.a which has four states q_0, q_1, q_2 and q_3. The transition function is defined as follows. In all states, the transitions have 2 conjunctive clauses. In q_0, there is a self-loop and another transition going to q_1 and q_2: $\delta_{\mathcal{A}}(q_0) = q_0 \vee (q_1 \wedge q_2)$. In q_1, the transition function is $\delta_{\mathcal{A}}(q_1) = q_0 \vee (q_2 \wedge q_3)$. The states q_2 and q_3, both, have a self-loop and a second transition: $\delta_{\mathcal{A}}(q_2) = q_2 \vee q_3$ and $\delta_{\mathcal{A}}(q_3) = q_3 \vee (q_1 \wedge q_2)$. From $\mathcal{A}$, we construct the MDP $\mathcal{M}_{\mathcal{A}}$ with two letters a, b and the same four states (see Figure 3.13.b). The a-transitions in q_0, q_2, q_3 are all self-loops (the Dirac distribution on itself); and a-transition in q_1 is the Dirac distribution on q_0. The b-transition in q_2 deterministically goes to q_3. The b-transitions in q_0, q_1, q_3 are respectively uniform distributions over the successors sets $\{q_1, q_2\}$, $\{q_2, q_3\}$ and $\{q_1, q_2\}$.*

$\triangleleft$

Several decision problems for 1L-AFA can be solved by computing the sequence of sets $Acc_{\mathcal{A}}(n, \mathcal{F})$ of states from which the word of length n is accepted, and we show that some synchronizing problems for MDPs require the computation of the sequence $\mathsf{Pre}^n(\mathcal{F})$ in the MDP $\mathcal{M}$. Therefore, the above relation between 1L-AFA and MDPs establishes bridges that we use in Chapter 4 to transfer complexity results from 1L-AFA to MDPs.

3.5 Relation to two-player turn-based games.

Remark 7 follows from the characterizations and reductions presented in Chapters 4, 5 and 6 to establish the upper bound of computational complexity for the synchronizing problems in MDPs .

Remark 7. For all three winning modes and all membership problems of {always, event, weakly, strongly} synchronization in MDPs, only the support of the probability distributions in the transition function of the system is relevant (i.e., the exact value of the positive probabilities does not matter).

As a result, we can encode an MDP as an A-labelled transition system (Q, R) with $R \subseteq Q \times \mathsf{A} \times Q$ such that $(q, a, q') \in R$ is a transition if $q' \in \mathsf{post}(q, a)$.

The A-labeled transition system (Q, R) of an MDP can be interpreted as a game arena. A turn-based two player game $\mathcal{G}$, with Player-1 and adversary Player-2, can be played in rounds. The game starts in an initial state $q_{\mathsf{init}} \in Q$. For each round, Player-1 chooses an action a to play in the current state q of the game. Next, Player-2 chooses a successor state among the states q' where $(q, a, q') \in R$, and the game moves to q'.

The definitions of paths and prefixes follow from the definitions in MDPs. A path $q_0 q_1 \cdots$ is an infinite sequence of states where $q_0 = q_{\mathsf{init}}$ and for all $i \in \mathbb{N}$, there exists an action a such that $(q_i, a, q_{i+1}) \in R$; the finite prefix of a path is a *prefix* or simply a *finite path*. Let $\mathsf{Pref}(\mathcal{G})$ be the set of all finite paths in the game. A strategy for Player-1 is a function $\alpha : \mathsf{Pref}(\mathcal{G}) \to \mathsf{A}$ that, given a finite path ρ, returns an action. A strategy for adversary Player-2 is a function $\beta : \mathsf{Pref}(\mathcal{G}) \times \mathsf{A} \to Q$ that, given a finite path ρ and an action a, returns the successor state. Given the strategy α for Player-1, we say that a finite path $q_0 q_1 \cdots q_n$ is *feasible* if there exists some strategy β for the adversary such that $q_0 = q_{\mathsf{init}}$ and $(q_i, a, q_{i+1}) \in R$ where $a = \alpha(q_0 q_1 \cdots q_i)$ and $q_{i+1} = \beta(q_0 q_1 \cdots q_i, a)$ for all $i \in \mathbb{N}$.

Given a strategy α for Player-1, the sequence $c_0 c_1 \cdots$ of *feasible current sets* is defined as follows. For all $i \in \mathbb{N}$, let

$$c_i = \{\mathsf{Last}(\rho) \mid \rho = q_0 q_1 \cdots q_i \in \mathsf{Pref}(\mathcal{G}) \text{ is a feasible path of length } i \text{ for } \alpha\}$$

where $\mathsf{Last}(\rho) = q_i$ is the last state q_i visited along the path $\rho = q_0 q_1 \cdots q_i$. Then, $c_0 = \{q_{\mathsf{init}}\}$ and $c_1 = \mathsf{post}(q_{\mathsf{init}}, a)$ where $a = \alpha(q_{\mathsf{init}})$. Now, we can define synchronizing winning conditions of Player-1 in games $\mathcal{G}$.

Definition 3.7 (Synchronizing games)**.** *Let $\mathcal{G}$ be a two-player turn-based game played on $\langle Q, R\rangle$, q_{init} be the initial state, T a target set and $f \in \{sub_T, mem_T\}$. The game $\mathcal{G}$ from q_{init} according to f is*

- *always synchronizing if $f(c_i)$ for all $i \in \mathbb{N}$,*
- *eventually (or event) synchronizing if $f(c_i)$ for some $i \in \mathbb{N}$,*
- *weakly synchronizing if for all $n \in \mathbb{N}$ there exists $i \geq n$ such that $f(c_i)$,*

- *strongly synchronizing if there exists $n \in \mathbb{N}$ such that for all $i \geq n$ we have $f(c_i)$*

where $c_0 c_1 c_2 \cdots$ is the sequence of feasible current sets for some strategy α of the good player.

The following decision problems are of interest.

Decision problem(s). For all $\lambda \in$ {always, event, weakly, strongly} and the target set T, the λ-synchronizing problem with function $f \in \{max_T, sum_T\}$ in games asks, given a game $\mathcal{G}$ and an initial state q_{init}, whether $\mathcal{G}$ is λ-synchronizing from q_{init} according to f.

As a result of Remark 7, we have Corollary 3.1.

Corollary 3.1. *For $\lambda \in$ {always, event, weakly, strongly}, the λ-synchronizing problem in turn-based two player games is polynomial-time equivalent with the membership problem of sure λ-synchronization in MDPs.*

We thus focus on establishing the computation complexities of synchronizing problems only in MDPs, and carry up the results to turn-based two player games.

3.6 Discussions

Unlike for Remark 7 stating that the exact value of transitions does not matter while synchronizing in an MDP, this result in general does not hold for emptiness problem of synchronizing languages in PAs. For instance in Chapter 4, we will establish undecidability result of the emptiness problem for almost-sure eventually synchronizing languages in PAs by a reduction from value 1 problem in PAs: the undecidability of value 1 is obtained by providing a PA (see Example 6.2 on page 135) where the PA has value 1 if and only if the probability of some transition is strictly greater than $\frac{1}{2}$. However, Remark 8 follows from the characterizations and reductions presented in Chapters 4, 5 and 6 to establish the upper bound of computational complexity of emptiness problem for sure eventually synchronizing languages in PAs.

Remark 8. For the emptiness problems of sure {always, event, weakly, strongly} synchronizing languages in PAs, only the support of the probability distributions in the transition function of the system is relevant (i.e., the exact value of the positive probabilities does not matter).

Since PAs are a generalization of NFAs where non-deterministic choices of successor are resolved with probabilistic distributions and by Remark 8, Corollary 3.2 follows.

Corollary 3.2. *For $\lambda \in$ {always, event, weakly, strongly}, the emptiness problem for λ-synchronizing languages in NFAs is polynomial-time equivalent with the emptiness problem of sure λ-synchronizing languages in PAs.*

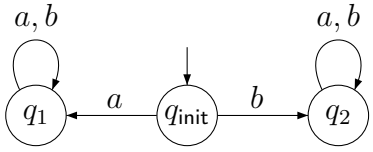

Figure 3.14: Randomization is necessary to decide the universality problems of synchronizing languages according to *max* in PAs.

We thus focus on establishing only the computation complexities of emptiness problems for synchronizing languages in PAs, and carry up the results to NFAs.

A further observation that can be deduced from the constructions and reductions presented in Chapters 4, 5 and 6 is that in order to decide emptiness problems of synchronizing languages in PAs it is sufficient to consider only pure words, and the computational complexities would not be more costly than considering randomized words.

The second classical decision problem for languages that one can study is the universality problem.

Decision problem(s). For $\lambda \in \{\text{always, event, weakly, strongly}\}$ and $f \in \{sum_T, max_T\}$, the *universality problem* for $\{\text{sure, almost}\}$ λ-synchronizing languages according to f asks, given a PA $\mathcal{P}$ and an initial distribution X_0, whether the language is universal, e.g. whether $\mathcal{L}^{\lambda}_{sure}(f) = \mathcal{D}(\mathsf{A})^{\omega}$.

This problem cannot be considered for limit-sure winning mode in PAs. The universality problems of synchronizing languages in PAs are mostly obtained by easy arguments, we will discuss these problems shortly, in Sections 4.4, 5.3 and 6.3, and give the main ideas and intuitions.

Unlike for the emptiness problem, in general, it is not sufficient to consider only pure words to decide universality of synchronizing languages. For the *max* function, Lemma 3.3 provides a PA and target sets where all pure words are {always, eventually, weakly, strongly} synchronizing whereas there is a randomized word that is not {always, eventually, weakly, strongly} synchronizing. An infinite randomized word is the uniformly randomized word over the alphabet A denoted by $w_u = d_0 d_1 d_2 \cdots$ where d_i is the uniform distribution over A for all $i \in \mathbb{N}$.

Lemma 3.3. *For all $\lambda \in$ {event, weakly, strongly} and $\mu \in$ {sure, almost}, there exists a PA $\mathcal{P}$ with an initial state q_{init} and two target sets $T = \{q_{\mathsf{init}}, q_1, q_2\}$ and $T' = \{q_1, q_2\}$ such that:*

(i) $\mathsf{A}^{\omega} \subseteq \mathcal{L}^{always}_{\mu}(max_T)$ *but* $w_u \notin \mathcal{L}^{always}_{\mu}(max_T)$ *and,*

(ii) $\mathsf{A}^{\omega} \subseteq \mathcal{L}^{\lambda}_{\mu}(max_{T'})$ *but* $w_u \notin \mathcal{L}^{\lambda}_{\mu}(max_{T'})$.

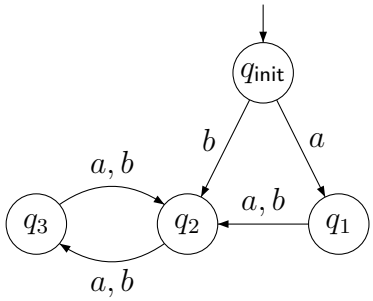

Figure 3.15: Randomization is necessary to decide the universality problems of {event, weakly} synchronizing languages according to *sum* in PAs.

Proof. Consider the PA $\mathcal{P}$ with states q_{init}, q_1 and q_2 and actions a, b as shown in Figure 3.14. All transitions are deterministic; two states q_1 and q_2 are absorbing and each is reached deterministically from q_{init} by playing one of the actions: $\mathsf{post}(q_{\mathsf{init}}, a) = \{q_1\}$ and $\mathsf{post}(q_{\mathsf{init}}, b) = \{q_2\}$.

To establish (i), since sure always synchronizing implies almost-sure always synchronizing, it is sufficient to show that all pure words are sure always synchronizing whereas the uniformly randomized word is not almost-sure always synchronizing. After reading the first letter of an input pure word w, since $\mathsf{post}(q_{\mathsf{init}}, a) = \{q_1\}$ and $\mathsf{post}(q_{\mathsf{init}}, b) = \{q_2\}$, if $w \in a \cdot \{a, b\}^\omega$ then $\mathcal{P}$ is in q_1 with probability 1; and $\mathcal{P}$ is in q_2 with probability 1 otherwise. Since the initial distribution is Dirac and since both states q_1 and q_2 are absorbing, we see that all pure words are sure always synchronizing according to max_T with $T = \{q_{\mathsf{init}}, q_1, q_2\}$. On the other hand, by inputting the uniformly randomized word w_u both states q_1 and q_2 would be reached with probability $\frac{1}{2}$ showing that $\|\mathcal{P}_n^{w_u}\| = \frac{1}{2}$ for all $n > 1$. It proves that $w_u \notin \mathcal{L}_{almost}^{always}(max_T)$.

To establish (ii), since $\mathcal{L}_{sure}^{strongly}(max_{T'}) \subseteq \mathcal{L}_{sure}^{weakly}(max_{T'}) \subseteq \mathcal{L}_{sure}^{eventually}(max_{T'})$ and since sure λ-synchronization implies almost-sure λ-synchronization, it is sufficient to show that $\mathsf{A}^\omega \subseteq \mathcal{L}_{sure}^{strongly}(max_{T'})$ and $w_u \notin \mathcal{L}_{almost}^{event}(max_{T'})$, as it implies that $\mathsf{A}^\omega \subseteq \mathcal{L}_\mu^\lambda(max_{T'})$ but $w_u \notin \mathcal{L}_\mu^\lambda(max_{T'})$ for all $\lambda \in \{\text{event, weakly, strongly}\}$ and all $\mu \in \{\text{sure, almost}\}$. To prove $\mathsf{A}^\omega \subseteq \mathcal{L}_{sure}^{strongly}(max_{T'})$, the same argument used to proved $\mathsf{A}^\omega \subseteq \mathcal{L}_{sure}^{always}(max_T)$ is valid: we only need to relax the condition on the initial distribution. To complete the proof, we see that the uniformly randomized word w_u is not almost-sure eventually synchronizing according to $max_{T'}$ again due to the fact that both absorbing states q_1 and q_2 are reached with probability $\frac{1}{2}$ after first input letter. The proof is complete.

□

For the *sum* function, a similar result holds for eventually and weakly synchronization. Lemma 3.4 provides a PA and some target state where all pure words are {eventually, weakly} synchronizing whereas there is a randomized word that is not {eventually, weakly} synchronizing.

Lemma 3.4. *For all $\lambda \in$ {event, weakly} and $\mu \in$ {sure, almost}, there exists a PA $\mathcal{P}$ with an initial state q_{init} such that:*

(*i*) $\mathsf{A}^\omega \subseteq \mathcal{L}^\lambda_\mu(q_3)$, *but*

(*ii*) $w_u \notin \mathcal{L}^\lambda_\mu(q_3)$.

Proof. Consider the PA $\mathcal{P}$ depicted in Figure 3.15 with states q_{init}, q_1, q_2 and q_3 and actions a, b. All transitions are deterministic; in the state q_{init} the a-transition goes to q_1 and the b-transition goes to q_2, both deterministically. All transitions in q_1 are redirected to q_2. Two states q_2 and q_3 make a deterministic loop: $\mathsf{post}(q_2, c) = \{q_3\}$ and $\mathsf{post}(q_3, c) = \{q_2\}$ for all actions $c \in \{a, b\}$.

To establish (*i*), since all sure weakly synchronizing words are sure eventually synchronizing, and since sure λ-synchronization implies almost-sure λ-synchronization, it is sufficient to show that all pure words are sure weakly synchronizing in $\{q_3\}$. Since all transitions are deterministic, by inputting a pure word the probability mass would stay in a single state in all steps. For all words $w \in a \cdot \{a, b\}^\omega$, the PA $\mathcal{P}$ is 1-synchronized in $\{q_3\}$ in all even steps; and similarly, it is 1-synchronized in $\{q_3\}$ in all odd steps for all words $w \in b \cdot \{a, b\}^\omega$. Thus, all pure words are sure weakly synchronizing in $\{q_3\}$.

To establish (*ii*), it is sufficient to show that the uniformly randomized word w_u is not almost-sure eventually synchronizing in $\{q_3\}$. The symbolic-outcome of $\mathcal{P}$ over w_u is

$$\begin{array}{c} q_{\mathsf{init}} \\ q_1 \\ q_2 \\ q_3 \end{array} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} \cdots$$

One can verify that $\mathcal{P}^{w_u}_n(q_3) \leq \frac{1}{2}$ for all $n \in \mathbb{N}$, and thus w_u is neither almost-sure (and thus sure) eventually nor almost-sure (and thus sure) weakly synchronizing in $\{q_3\}$.

□

This result, in general, does not hold for the universality of always and strongly synchronizing languages according to *sum*. *Discussions* of Chapters 4, 5 and 6 are devoted to the universality problems of synchronizing languages, where we will show that the universality problem of sure and almost-sure always synchronizing languages, are in PTIME. The universality problems of sure eventually, weakly and strongly synchronizing languages are in PSPACE, while for almost-sure eventually, weakly and strongly synchronizing languages, the universality problems are PSPACE-complete.

4 Always and Eventually Synchronizing Condition

First sight. In this chapter we study the always and eventually synchronizing conditions in MDPs and PAs. We provide tight complexity bounds of membership problems for each winning modes {sure, almost-sure and limit-sure} in MDPs, and the memory requirement of winning strategies. We also establish the tight complexity bounds for the emptiness problems of always synchronizing languages for PAs, and for the emptiness problems of sure eventually synchronizing language. We prove undecidability results for the emptiness problem of {almost-sure, limit-sure} synchronizing languages in PAs. Table 4.1 and Table 4.2 on pages 67 and 81 summarize the results presented in this chapter.

Contents

4.1 Always synchronizing condition in MDPs and PAs

We first provide a lemma stating that for always synchronizing condition, the winning regions of the three winning modes {sure, almost-sure, limit-sure} coincide in MDPs. The result also holds in PAs meaning that from an initial state and a target set T, the sure always-synchronizing language is empty if, and only if, the almost-sure (and limit-sure) always-synchronizing language is empty. We also observe that for always synchronization in probabilistic settings, it only matters whether the value of a transition is non-zero (for the *sum* function) or whether a transition is deterministic (for the *max* function); deterministic here means there exists a unique successor with probability 1 for the transition. By this observation and provided Lemma 4.1 and Corollary 4.1, the solutions for MDPs and PAs carry up to their counterpart non-probabilistic settings, i.e. two player games and NFAs. Results presented in this section are listed in Tables 4.1 and 4.2.

Lemma 4.1. *Let T be a set of states for an MDP $\mathcal{M}$. For all functions $f \in \{max_T, sum_T\}$, we have $\langle\langle 1\rangle\rangle^{always}_{sure}(f) = \langle\langle 1\rangle\rangle^{always}_{almost}(f) = \langle\langle 1\rangle\rangle^{always}_{limit}(f)$.*

Proof. It follows from the definition of winning modes that $\langle\langle 1\rangle\rangle^{always}_{sure}(f) \subseteq \langle\langle 1\rangle\rangle^{always}_{almost}(f) \subseteq \langle\langle 1\rangle\rangle^{always}_{limit}(f)$. Hence it suffices to show that $\langle\langle 1\rangle\rangle^{always}_{limit}(f) \subseteq \langle\langle 1\rangle\rangle^{always}_{sure}(f)$, that is for all initial distributions X_0, if $\mathcal{M}$ is limit-sure always synchronizing from X_0 (i.e. $X_0 \in \langle\langle 1\rangle\rangle^{always}_{limit}(f)$), then $\mathcal{M}$ is sure always synchronizing from X_0 (i.e. $X_0 \in \langle\langle 1\rangle\rangle^{always}_{sure}(f)$). For $f = max_T$, consider ϵ smaller than the smallest positive probability in the initial distribution X_0 and in the transitions of the MDP $\mathcal{M}$. Then, given an always $(1-\epsilon)$-synchronizing strategy α (consider a blind strategy for the PA), it is easy to show by induction on k that the distributions $\mathcal{M}^{\alpha}_k$ are Dirac for all $k \geq 0$. Thus, α is an always 1-synchronizing strategy that proves the statement.

A similar argument for $f = sum_T$ shows that for sufficiently small ϵ, an always $(1-\epsilon)$-synchronizing strategy β must produce a sequence $\mathcal{M}^{\beta}_0\mathcal{M}^{\beta}_1\cdots$ of distributions with support contained in T, meaning that $S_k \subseteq T$ where $S_k = \mathsf{Supp}(\mathcal{M}^{\beta}_k)$ for all $k \geq 0$, until some support repeats in the sequence. Such supports exist because Q is finite, and so there must be two numbers $m > n$ such that $S_n = S_m$. This naturally induces an always 1-synchronizing strategy proving the statement.

□

The proof of Lemma 4.1 is valid even if we restrict the controller in the MDPs to use only blind strategies.

Corollary 4.1. *Let T be a set of states for a PA $\mathcal{P}$. For all functions $f \in \{max_T, sum_T\}$ and all initial state q_{init}, the following three propositions are equivalent:*

(i) $\mathcal{L}^{always}_{sure}(f) = \emptyset$,

(ii) $\mathcal{L}^{always}_{almost}(f) = \emptyset$,

(iii) $\mathcal{L}^{always}_{limit}(f) = \emptyset$.

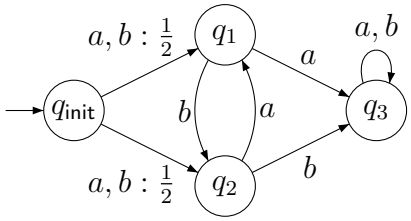

Figure 4.1: The MDP described in Example 4.1 where for the target set $T = \{q_{\mathsf{init}}, q_1, q_2\}$, we have $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{always}_{sure}(sum_T)$ but $q_{\mathsf{init}} \notin \langle\langle 1 \rangle\rangle^{always}_{sure}(max_T)$.

Example 4.1. *Consider the MDP depicted in Figure 4.1 that has four states $q_{\mathsf{init}}, q_1, q_2, q_3$ and two actions a, b. From the initial state q_{init}, both a-transitions and b-transitions have two successors q_1 and q_2 each with probability $\frac{1}{2}$. The state q_3 is an absorbing state and it is deterministically reachable from q_1 by the action a and from q_2 by b. The b-transition from q_1 goes to q_2, and the a-transition moves the MDP back from q_2 to q_1. Let $T = \{q_{\mathsf{init}}, q_1, q_2\}$. Consider the memoryless strategy α that always plays b in q_1 and a in q_2; this strategy is a witness to prove that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{always}_{sure}(sum_T)$. On the other hand, we see that $\|M_i^\alpha\|_T \leq \frac{1}{2}$ for all steps $i \geq 1$ and under all strategies α. Thus, $q_{\mathsf{init}} \notin \langle\langle 1 \rangle\rangle^{always}_{sure}(max_T)$.*

For the likewise structured PA, after inputting the first letter both states q_1 and q_2 are assigned with probability $\frac{1}{2}$. However, by the second letter, no matter if it is a or b, the PA will be $\frac{1}{2}$-synchronized in $\{q_3\}$. This observation and the fact that q_3 is absorbing imply that both the languages $\mathcal{L}^{always}_{sure}(sum_T)$ and $\mathcal{L}^{always}_{sure}(max_T)$ are empty from q_{init}.

◁

Always synchronizing condition in MDPs. Consider an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ and the set T of target states. Let α be an always 1-synchronizing strategy according to max_T and $\mathcal{M}_0^\alpha \mathcal{M}_1^\alpha \cdots$ be the outcome. For all $k \geq 0$, let $q_k \in T$ be such that $\mathcal{M}_k^\alpha(q_k) = 1$. For all k, since $\mathcal{M}_k^\alpha$ and $\mathcal{M}_{k+1}^\alpha$ are both Dirac distributions, then there is a deterministic transition from q_k to q_{k+1} (there exists an action a such that the unique successor of q_k and a is q_{k+1}; formally $\mathsf{post}(q_k, a) = \{q_{k+1}\}$). In particular X_0 is Dirac, and let $q_{\mathsf{init}} \in T$ be such that $X_0(q_{\mathsf{init}}) = 1$. It follows that there is an infinite path from q_{init} in the digraph $\langle T, E \rangle$ where $(q, q') \in E$ if there exists an action $a \in \mathsf{A}$ such that $\mathsf{post}(q, a) = \{q'\}$. The existence of this path entails that there is a loop reachable from q_{init} in the digraph $\langle T, E \rangle$, and this naturally defines a sure-winning always synchronizing strategy in $\mathcal{M}$. Thus, to decide the membership problem for always synchronizing according to max_T (for all winning modes), it is sufficient to check the existence of an infinite path staying in T in the digraph $\langle T, E \rangle$, which can be decided in PTIME. Note that for the functions max_T, pure memoryless strategies are sufficient.

It follows from the proof of Lemma 4.1 that the winning region for always synchronizing according to sum_T coincides with the set of winning initial distributions for the safety condition $\Box T$ in the traditional semantics in MDPs, which can also be computed in PTIME [CH12]. Thus, for the functions sum_T, pure memoryless strategies are sufficient.

Theorem 4.1. *For always synchronizing condition in MDPs and for all functions* $f \in \{max_T, sum_T\}$*:*

1. *(Complexity). The membership problem is in* PTIME.
2. *(Memory). Pure memoryless strategies are sufficient.*

Always synchronizing condition in PAs. We have just seen that in MDPs, to decide always synchronizing condition according to max_T, it is sufficient to find an infinite path $q_0q_1q_2\cdots$ such that there is a deterministic transition from q_k to q_{k+1} for all $k \geq 0$ and q_0 is the state where the initial distribution is the Dirac distribution on q_0 ($q_{\mathsf{init}} = q_0$). Such transitions give an action a_k such that $\mathsf{post}(q_k, a_k) = \{q_{k+1}\}$ for all $k \geq 0$. A blind strategy α that plays a_k for all $k \in \mathbb{N}$ is thus always synchronizing according to max_T. It means that pure blind strategies are sufficient to decide the membership problem of always synchronizing condition according to max_T. It follows that the always synchronizing language according to max_T from an initial state q_{init} for the PA $\mathcal{P}$ is not empty if and only if q_{init} is winning for the always synchronizing condition in the likewise structured MDP $\mathcal{M}$. As a result, the emptiness problem of always synchronizing language in PAs according to max_T can be decided in PTIME.

By a similar argument to the case for MDPs, the always synchronizing language according to sum_T in PAs is equal to the ω-language with safety condition $\Box T$ in the traditional semantics in PAs, where the emptiness problem is PSPACE-complete [CT12].

Theorem 4.2. *For always synchronizing language in PAs:*

1. *The emptiness problem for* $f = max_T$ *is in* PTIME.
2. *The emptiness problem for* $f = sum_T$ *is* PSPACE-complete.

4.2 Eventually synchronizing condition in MDPs

In this section, we show the PSPACE-completeness of the membership problem for eventually synchronizing conditions in the three winning modes. By Remarks 5 and 6 (on page 50), we consider the membership problem with function *sum* and Dirac initial distributions (i.e., single initial state). The presented algorithms to solve the membership problem for $\langle\langle 1\rangle\rangle_\mu^{event}(sum_T)$ for arbitrary T and $\mu \in \{\text{sure, almost, limit}\}$, can be used for the special case of singleton T, which solves the membership problem for $\langle\langle 1\rangle\rangle_\mu^{event}(max_T)$. Furthermore, we establish PSPACE-completeness of the problems, even in the special case when T is a singleton, which implies PSPACE-completeness of the membership problems for $\langle\langle 1\rangle\rangle_\mu^{event}(max_T)$ too. Results presented in this section are listed in Table 4.1.

	Always		Eventually	
	Complexity	**Required memory**	**Complexity**	**Required memory**
Sure	PTIME	memoryless	PSPACE-complete	exponential
Almost-sure			PSPACE-complete	infinite
Limit-sure			PSPACE-complete	unbounded

Table 4.1: Computational complexity of the membership problem of always and eventually synchronization in MDPs, and memory requirement for the winning strategies (for always synchronizing, the three modes coincide).

4.2.1 Sure eventually synchronization

Given a target set T in an MDP, the membership problem for sure-winning eventually synchronizing condition in T [1] can be solved by computing the sequence $\mathsf{Pre}^n(T)$ of iterated predecessors. A state q_{init} is sure-winning for eventually synchronizing in T if and only if $q_{\mathsf{init}} \in \mathsf{Pre}^n(T)$ for some $n \geq 0$.

Lemma 4.2. *Let $\mathcal{M}$ be an MDP and T be a target set. For all states q_{init}, we have $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_T)$ if and only if there exists $n \geq 0$ such that $q_{\mathsf{init}} \in \mathsf{Pre}^n_{\mathcal{M}}(T)$.*

Proof. We prove the following equivalence by induction (on the length i): for all initial states q_{init}, there exists a strategy α sure-winning in i steps from q_{init} (i.e., such that $\mathcal{M}^\alpha_i(T) = 1$) if and only if $q_{\mathsf{init}} \in \mathsf{Pre}^i(T)$. The case $i = 0$ trivially holds since for all strategies α, we have $\mathcal{M}^\alpha_0(T) = 1$ if and only if $q_{\mathsf{init}} \in T$.

Assume that the equivalence holds for all $i < n$. For the induction step, show that $\mathcal{M}$ is sure eventually synchronizing from q_{init} (in n steps) if and only if there exists an action a such that $\mathcal{M}$ is sure eventually synchronizing (in $n-1$ steps) from all states $q' \in \mathsf{post}(q_{\mathsf{init}}, a)$ (equivalently, $\mathsf{post}(q_{\mathsf{init}}, a) \subseteq \mathsf{Pre}^{n-1}(T)$ by the induction hypothesis, that is $q_{\mathsf{init}} \in \mathsf{Pre}^n(T)$). First, if all successors q' of q_{init} under some action a are sure eventually synchronizing, then so is q_{init} by playing a followed by a winning strategy from each successor q'. For the other direction, assume towards contradiction that $\mathcal{M}$ is sure eventually synchronizing from q_{init} (in n steps), but for each action a, there is a state $q' \in \mathsf{post}(q_{\mathsf{init}}, a)$ that is not sure eventually synchronizing. Then, from q' there is a positive probability to reach a state not in T after $n-1$ steps, no matter the strategy played. Hence from q_{init}, for all strategies, the probability mass in T cannot be 1 after n steps, in contradiction with the fact that $\mathcal{M}$ is sure eventually synchronizing from q_{init} in n steps. It follows that the induction step holds, and the proof is complete.

□

1. We sometimes refer to synchronizing condition according to sum_T by synchronizing condition in T.

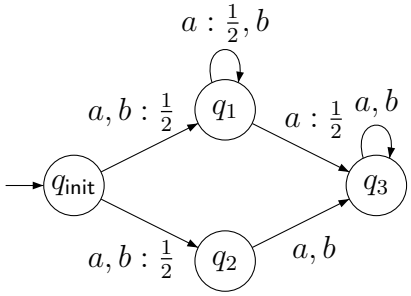

Figure 4.2: The MDP described in Example 4.2 where for the target set $T = \{q_1, q_3\}$, we have $q_{\mathsf{init}} \in \langle\!\langle 1 \rangle\!\rangle^{event}_{sure}(sum_T)$ but $q_{\mathsf{init}} \notin \langle\!\langle 1 \rangle\!\rangle^{event}_{sure}(q_2)$.

Example 4.2. *Consider the MDP $\mathcal{M}$ shown in Figure 4.2 that has four states $q_{\mathsf{init}}, q_1, q_2, q_3$ and two actions a, b. From the initial state q_{init}, both a-transitions and b-transitions have two successors q_1 and q_2 each with probability $\frac{1}{2}$. The b-transition on the state q_1 is a self-loop, and the a-transition is a uniform distribution on $\{q_1, q_3\}$. The a and b-transitions in q_2 are both the Dirac distribution on q_3; and q_3 is an absorbing state. For the target set $\{q_2\}$, the predecessor is $\mathsf{Pre}^n(\{q_2\}) = \emptyset$ for all $n \geq 1$. The MDP $\mathcal{M}$ is not sure eventually synchronizing in $\{q_2\}$ from q_{init}. For the target set $T = \{q_1, q_3\}$, the predecessor sequence is*

$$\{q_1, q_3\}\{q_1, q_2, q_3\}(\{q_{\mathsf{init}}, q_1, q_2, q_3\})^\omega.$$

We see that $q_{\mathsf{init}} \in \mathsf{Pre}^2(T)$ which gives $q_{\mathsf{init}} \in \langle\!\langle 1 \rangle\!\rangle^{event}_{sure}(sum_T)$.

◁

By Lemma 4.2, the membership problem for sure eventually synchronizing is equivalent to the emptiness problem of 1L-AFA, and thus PSPACE-complete. Moreover if $q_{\mathsf{init}} \in \mathsf{Pre}^n_{\mathcal{M}}(T)$, a finite-memory strategy with n modes that at mode i in a state q plays an action a such that $\mathsf{post}(q, a) \subseteq \mathsf{Pre}^{i-1}(T)$ is sure winning for eventually synchronizing.

There exists a family of MDPs $\mathcal{M}_n$ ($n \in \mathbb{N}$) over alphabet $\{a, b\}$ that are sure winning for eventually synchronization, and where the sure winning strategies require exponential memory. The MDP $\mathcal{M}_2$ is shown in Figure 4.3. The structure of $\mathcal{M}_n$ consists of an initial state q_{init}, n components $H_1, \ldots, H_n$ and two states q_T and $q_\perp$. The goal of construction is to be sure eventually synchronizing in the target set $\{q_T\}$. Each component H_i is a cycle of length p_i, the i-th prime number. We name the states of H_i with $q^i_1 q^i_2 \cdots q^i_{p_i}$ where the initial state in this cycle is q^i_1 and the last state is $q^i_{p_i}$. On all actions in q_{init}, the probabilistic transition is a uniform distribution on the initial states q^i_1 of the n components $H_1, \ldots, H_n$. On action a, the next state in the cycle H_n is reached: $\mathsf{post}(q^i_j, a) = \{q^i_{j+1}\}$ for all $1 \leq j < p_i$ and $\mathsf{post}(q^i_{p_i}, a) = \{q^i_1\}$. On action b the target state q_T is reached but only from the last state $q^i_{p_i}$ in the cycles. From other states, the action b leads to $q_\perp$ (transitions not depicted). A sure winning strategy for eventually synchronization in $\{q_T\}$ is to play a in the first $p^\#_n = \prod_{i=1}^n p_i$ steps, and then play b. This requires memory of size $p^\#_n > 2^n$ while the size of $\mathcal{M}_n$ is in $O(n^2 \log n)$ [BS96]. It can be proved by standard pumping arguments that no strategy of size smaller than $p^\#_n$ is sure winning.

The following theorem summarizes the results for sure eventually synchronizing in MDPs.

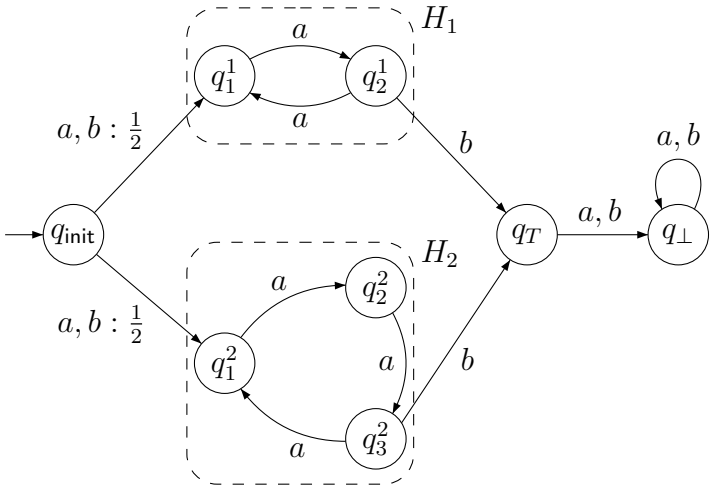

Figure 4.3: The MDP $\mathcal{M}_2$. All not-drawn transitions are directed to $q_\perp$, for example b-transitions in q_1^1 and q_1^2.

Theorem 4.3. *For sure eventually synchronizing condition in MDPs and for all functions $f \in \{max_T, sum_T\}$:*

1. *(Complexity). The membership problem is* PSPACE-complete.
2. *(Memory). Exponential memory is necessary and sufficient for both pure and randomized strategies, and pure strategies are sufficient.*

4.2.2 Almost-sure eventually synchronization

This subsection starts with an example of almost-sure eventually synchronizing MDPs where infinite memory is necessary to win the almost-sure eventually synchronizing condition. Example 4.3 describes this MDP and provide a winning strategy that uses infinite memory to win almost-sure eventually synchronizing condition; next in Lemma 4.3, we prove that no finite-memory strategy wins this condition.

Example 4.3. *Consider the MDP in Figure 4.4 with three states $q_{\mathsf{init}}, q_1, q_2$ and two actions a, b. The only probabilistic transitions are in the initial state q_{init} where on both actions a and b, the next successor is itself or q_1 each with probability $\frac{1}{2}$. The a-transition in q_1 is a self-loop and the b-transition goes to q_2 where all transitions are deterministically directed to q_{init}. We are interested to study whether $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(q_2)$. To this aim, we construct a strategy that is almost-sure eventually synchronizing in $\{q_2\}$, showing that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(q_2)$ and suggesting that winning strategies for almost-sure eventually synchronization may require infinite memory. First, observe that for all $\epsilon > 0$ we can have*

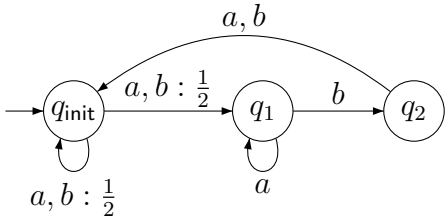

Figure 4.4: An MDP where infinite memory is necessary for almost-sure eventually synchronizing strategies.

probability at least $1 - \epsilon$ *in* q_2 *after finitely many steps: playing* n *times* a *and then* b *leads to probability* $1 - \frac{1}{2^n}$ *in* q_2*. Thus the MDP is limit-sure eventually synchronizing in the target set* $\{q_2\}$*. Moreover the remaining probability mass is in* q_{init} *and* q_1*; which we recall with the MDP being limit-sure eventually synchronizing in* $\{q_2\}$ *with support* $\{q_{\mathsf{init}}, q_1, q_2\}$*. It turns out that from any (initial) distribution with support* $\{q_{\mathsf{init}}, q_1, q_2\}$*, the MDP is again limit-sure eventually synchronizing in the target set* $\{q_2\}$*, and with support in* $\{q_{\mathsf{init}}, q_1, q_2\}$ *(by strategies with the same logic: playing finitely many consecutive* a *and then* b*). Therefore we can take a smaller value of* ϵ *and play a strategy to have probability at least* $1 - \epsilon$ *in* q_2*, and repeat this for* $\epsilon \to 0$*. This strategy ensures almost-sure eventually synchronizing in the target set* $\{q_2\}$*.*

$\triangleleft$

The next Lemma shows that infinite memory is indeed necessary for almost-sure eventually synchronizing strategies for the MDP of Example 4.3.

Lemma 4.3. *There exists an almost-sure eventually synchronizing MDP for which all almost-sure eventually synchronizing strategies require infinite memory.*

Proof. Consider the MDP $\mathcal{M}$ of Example 4.3 which is shown in Figure 4.4. We argued in Example 4.3 that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(q_2)$ and we now show that infinite memory is necessary for almost-sure eventually synchronization in $\{q_2\}$ from q_{init}.

Assume towards contradiction that there exists a finite-memory strategy α that is almost-sure eventually synchronizing in the target set $\{q_2\}$. Consider the Markov chain $\mathcal{M}(\alpha)$ obtained by the product of the MDP $\mathcal{M}$ with the finite-state transducer defining α as explained in Section 2.2.3. To simply the notation, by $X_0\ X_1\ X_2 \cdots$ we denote the sequence of probability distributions generated by $\mathcal{M}(\alpha)$. We know that the probability $X_k((m, q))$ in transient states (m, q) vanishes for $k \to \infty$ [FV97, Nor98, Ser13]. A state (m, q) in $\mathcal{M}(\alpha)$ is called a *q-state*. Since α is almost-sure eventually synchronizing (but is not sure eventually synchronizing) in q_2, there is a q_2-state in the set of recurrent states of $\mathcal{M}(\alpha)$. Since on all actions q_{init} is a successor of q_2, and q_{init} is a successor of itself, it follows that there is a recurrent q_{init}-state in $\mathcal{M}(\alpha)$, and that all periodic supports of recurrent states in $\mathcal{M}(\alpha)$ contain a q_{init}-state. Hence, in each stationary distribution there

is a q_{init}-state with a (strictly) positive probability, and therefore the probability mass in q_{init} is bounded away from zero. It follows that the probability mass in q_2 is bounded away from 1 thus α is not almost-sure eventually synchronizing in q_2, a contradiction.

□

It turns out that in general, almost-sure eventually synchronizing strategies can be constructed from a family of limit-sure eventually synchronizing strategies if we can also ensure that the probability mass remains in the winning region (as in the MDP of Example 4.3).

We present a characterization of the winning region for almost-sure winning based on an extension of the limit-sure eventually synchronizing condition *with exact support*. This condition requires to ensure probability arbitrarily close to 1 in the target set T, and moreover that after the same number of steps the support of the probability distribution is contained in a given set U. Formally, given an MDP $\mathcal{M}$, let $\langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$ for $T \subseteq U$ be the set of all initial distributions such that for all $\epsilon > 0$ there exist a strategy α and $n \in \mathbb{N}$ such that $\mathcal{M}^\alpha_n(T) \geq 1 - \epsilon$ and $\mathcal{M}^\alpha_n(U) = 1$ (i.e. the support of $\mathcal{M}^\alpha_n$ is contained in U). We say that α is limit-sure eventually synchronizing in T with support in U.

We will present an algorithmic solution to limit-sure eventually synchronizing condition with exact support in Section 4.2.3. Our characterization of the winning region for almost-sure winning is as follows.

Lemma 4.4. *Let $\mathcal{M}$ be an MDP and T be a target set. For all states q_{init}, we have $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{event}_{almost}(sum_T)$ if and only if there exists a set U such that:*

- *$q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{event}_{sure}(sum_U)$, and*
- *$X_U \in \langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$ where X_U is the uniform distribution over U.*

Proof. First, if $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{event}_{almost}(sum_T)$, then there is a strategy α such that $\sup_{n\in\mathbb{N}} \mathcal{M}^\alpha_n(T) = 1$. Then either $\mathcal{M}^\alpha_n(T) = 1$ for some $n \geq 0$, or $\limsup_{n\to\infty} \mathcal{M}^\alpha_n(T) = 1$. If $\mathcal{M}^\alpha_n(T) = 1$, then q_{init} is sure winning for eventually synchronizing in T, thus $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{event}_{sure}(sum_T)$ and we can take $U = T$. Otherwise, for all $i > 0$ there exists $n_i \in \mathbb{N}$ such that $\mathcal{M}^\alpha_{n_i}(T) \geq 1 - 2^{-i}$, and moreover $n_{i+1} > n_i$ for all $i > 0$. Let $s_i = \mathsf{Supp}(\mathcal{M}^\alpha_{n_i})$ be the support of $\mathcal{M}^\alpha_{n_i}$. Since the state space is finite, there is a set U that occurs infinitely often in the sequence $s_0 s_1 \ldots$, thus for all $k > 0$ there exists $m_k \in \mathbb{N}$ such that $\mathcal{M}^\alpha_{m_k}(T) \geq 1 - 2^{-k}$ and $\mathcal{M}^\alpha_{m_k}(U) = 1$. It follows that α is sure eventually synchronizing in U from q_{init}, hence $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{event}_{sure}(sum_U)$. Moreover $\mathcal{M}$ with initial distribution $X_1 = \mathcal{M}^\alpha_{m_1}$ is limit-sure eventually synchronizing in T with exact support in U. Since $\mathsf{Supp}(X_1) = U = \mathsf{Supp}(X_U)$, it follows by Corollary 4.3 that $X_U \in \langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$.

To establish the converse, note that since $X_U \in \langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$, it follows from Corollary 4.3 that from all initial distributions with support in U, for all $\epsilon > 0$ there exists a strategy α_ϵ and a position n_ϵ such that $\mathcal{M}^{\alpha_\epsilon}_{n_\epsilon}(T) \geq 1 - \epsilon$ and $\mathcal{M}^{\alpha_\epsilon}_{n_\epsilon}(U) = 1$. We construct an almost-sure limit eventually synchronizing strategy α as follows. Since $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{event}_{sure}(sum_U)$, play according to a sure eventually synchronizing strategy from q_{init} until all the probability mass is in U. Then for $i = 1, 2, \ldots$ and $\epsilon_i = 2^{-i}$, repeat the

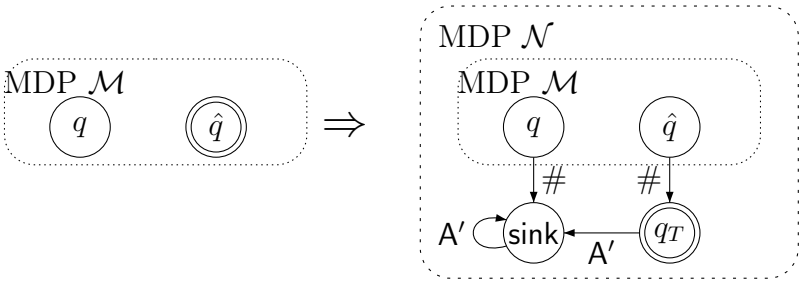

Figure 4.5: Sketch of the reduction to show PSPACE-hardness of the membership problem for almost-sure eventually synchronizing in MDPs.

following procedure: given the current probability distribution, select the corresponding strategy α_{ϵ_i} and play according to α_{ϵ_i} for n_{ϵ_i} steps, ensuring probability mass at least $1 - 2^{-i}$ in T, and since after that the support of the probability mass is again in U, play according to $\alpha_{\epsilon_{i+1}}$ for $n_{\epsilon_{i+1}}$ steps, etc. This strategy α ensures that $\sup_{n\in\mathbb{N}} \mathcal{M}_n^\alpha(T) = 1$ from q_{init}, hence $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(sum_T)$.

□

Note that from Lemma 4.4, it follows that counting strategies are sufficient to win almost-sure eventually synchronizing condition (a strategy is *counting* if $\alpha(\rho) = \alpha(\rho')$ for all prefixes ρ, ρ' with the same length and $\mathsf{Last}(\rho) = \mathsf{Last}(\rho')$).

As we show in Section 4.2.3 that the membership problem for limit-sure eventually synchronizing with exact support can be solved in PSPACE, it follows from the characterization in Lemma 4.4 that the membership problem for almost-sure eventually synchronizing is in PSPACE, using the following (N)PSPACE algorithm: guess the set U, and check that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_U)$, and that $X_U \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_T, U)$ where X_U is the uniform distribution over U (this can be done in PSPACE by Theorem 4.3 and Theorem 4.5). We present a matching lower bound.

Lemma 4.5. *For MDPs, the membership problem for $\langle\langle 1 \rangle\rangle^{event}_{almost}(sum_T)$ is* PSPACE-hard *even if T is a singleton.*

Proof. The proof is by a reduction from the membership problem for sure eventually synchronization, which is PSPACE-complete by Theorem 4.3. Given an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$, an initial state $q_{\mathsf{init}} \in Q$, and a state $\hat{q} \in Q$, we construct an MDP $\mathcal{N} = \langle Q', \mathsf{A}', \delta' \rangle$ and a state $q_T \in Q'$ such that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(\hat{q})$ in $\mathcal{M}$ if and only if $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(q_T)$ in $\mathcal{N}$. The MDP $\mathcal{N}$ is a copy of $\mathcal{M}$ with two new states q_T and sink reachable only by a new action # (see Figure 4.5). Formally, $Q' = Q \cup \{q_T, \mathsf{sink}\}$ and $\mathsf{A}' = \mathsf{A} \cup \{\#\}$, and the transition function δ' is defined as follows, for all $q \in Q$: $\delta'(q, a) = \delta(q, a)$ for all $a \in \mathsf{A}$, $\delta'(q, \#)(\mathsf{sink}) = 1$ if $q \neq \hat{q}$, and $\delta'(\hat{q}, \#)(q_T) = 1$; finally, for all $a \in \mathsf{A}'$, let $\delta'(q_T, a)(\mathsf{sink}) = \delta'(\mathsf{sink}, a)(\mathsf{sink}) = 1$.

The goal is that $\mathcal{N}$ simulates $\mathcal{M}$ until the action # is played in $\hat{q}$ to move the probability mass from $\hat{q}$ to q_T, ensuring that if $\mathcal{M}$ is sure-winning for eventually synchronizing in $\{\hat{q}\}$,

then $\mathcal{N}$ is also sure-winning (and thus almost-sure winning) for eventually synchronizing in $\{q_T\}$. Moreover, the only way to be almost-sure eventually synchronizing in $\{q_T\}$ is to have probability 1 in q_T at some point, because the state q_T is transient under all strategies, thus the probability mass cannot accumulate and tend to 1 in q_T in the long run. It follows that (from all initial states q_{init}) $\mathcal{M}$ is sure-winning for eventually synchronizing in $\{\hat{q}\}$ if and only if $\mathcal{N}$ is almost-sure winning for eventually synchronizing in $\{q_T\}$. It follows from this reduction that the membership problem for almost-sure eventually synchronizing condition is PSPACE-hard.

□

The results of this section are summarized as follows.

Theorem 4.4. *For almost-sure eventually synchronizing condition in MDPs and for all functions $f \in \{max_T, sum_T\}$:*

1. *(Complexity). The membership problem is* PSPACE-complete.
2. *(Memory). Infinite memory is necessary in general for both pure and randomized strategies, and pure strategies are sufficient.*

4.2.3 Limit-sure eventually synchronization

In this subsection, we present the algorithmic solution for limit-sure eventually synchronizing with exact support in MDPs. Note that the limit-sure eventually synchronizing condition is a special case where the support is the state space of the MDP.

Example 4.4. *Consider the MDP in Figure 3.7 which is limit-sure eventually synchronizing in $\{q_2\}$, as shown in Lemma 3.1. For $i = 0, 1, \ldots$, the sequence $\mathsf{Pre}^i(T)$ of predecessors of $T = \{q_2\}$ is ultimately periodic: $\mathsf{Pre}^0(T) = \{q_2\}$, and $\mathsf{Pre}^i(T) = \{q_1\}$ for all $i \geq 1$. Given $\epsilon > 0$, a strategy to get probability $1 - \epsilon$ in q_2 first accumulates probability mass in the* periodic *subsequence of predecessors (here $\{q_1\}$), and when the probability mass is greater than $1-\epsilon$ in q_1, the strategy injects the probability mass in q_2 (through the aperiodic prefix of the sequence of predecessors).*

◁

In general, the typical shape of a limit-sure eventually synchronizing strategy is as explained in Example 4.4. To have probability arbitrarily close to 1 in a target set T, a limit-sure eventually synchronizing strategy accumulates the probability mass in the *periodic* subsequence of predecessors until the probability mass is as large as demanded; and then the strategy synchronously moves the probability mass into the target set T via the prefix subsequence of predecessors. Note that in this scenario, the MDP is also limit-sure eventually synchronizing in every set $\mathsf{Pre}^i(T)$ of the sequence of predecessors. A special

case is when it is possible to get probability 1 in the sequence of predecessors after finitely many steps. In this case, the probability mass injected in T is 1 and the MDP is even sure-winning. The algorithm for deciding limit-sure eventually synchronization relies on the above characterization, generalized in Lemma 4.6 to limit-sure eventually synchronizing with exact support, saying that limit-sure eventually synchronization in T with support in U is equivalent to either limit-sure eventually synchronization in $\mathsf{Pre}^k(T)$ with support in $\mathsf{Pre}^k(U)$ (for arbitrary k), or sure eventually synchronizing in T (and therefore also in U).

Lemma 4.6. *Given an MDP $\mathcal{M}$, for all $T \subseteq U$ and all $k \geq 0$, we have $\langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U) = \langle\langle 1\rangle\rangle^{event}_{sure}(sum_T) \cup \langle\langle 1\rangle\rangle^{event}_{limit}(sum_R, Z)$ where $R = \mathsf{Pre}^k(T)$ and $Z = \mathsf{Pre}^k(U)$.*

Proof. The proof is in two parts. First we show that $\langle\langle 1\rangle\rangle^{event}_{sure}(sum_T) \cup \langle\langle 1\rangle\rangle^{event}_{limit}(sum_R, Z) \subseteq \langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$: since $T \subseteq U$, it follows from the definitions that $\langle\langle 1\rangle\rangle^{event}_{sure}(sum_T) \subseteq \langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$; to show that $\langle\langle 1\rangle\rangle^{event}_{limit}(sum_R, Z) \subseteq \langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$ in an MDP $\mathcal{M}$, let $\epsilon > 0$ and consider an initial distribution X_0 and a strategy α such that for some $i \geq 0$ we have $\mathcal{M}^\alpha_i(R) \geq 1 - \epsilon$ and $\mathcal{M}^\alpha_i(Z) = 1$. We construct a strategy β that plays like α for the first i steps, and then since $R = \mathsf{Pre}^k(T)$ and $Z = \mathsf{Pre}^k(U)$ plays from states in R according to a sure eventually synchronizing strategy with target T, and from states in $Z \setminus R$ according to a sure eventually synchronizing strategy with the target set U (such strategies exist by the proof of Lemma 4.2). The strategy β ensures from X_0 that $\mathcal{M}^\beta_{i+k}(T) \geq 1-\epsilon$ and $\mathcal{M}^\beta_{i+k}(U) = 1$, showing that $\mathcal{M}$ is limit-sure eventually synchronizing in T with support in U.

Second we show the converse inclusion, namely that $\langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U) \subseteq \langle\langle 1\rangle\rangle^{event}_{sure}(sum_T) \cup \langle\langle 1\rangle\rangle^{event}_{limit}(sum_R, Z)$. Consider an initial distribution $X_0 \in \langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$ in the MDP $\mathcal{M}$ and for $\epsilon_i = \frac{1}{i}$ ($i \in \mathbb{N}$) let α_i be a strategy and $n_i \in \mathbb{N}$ such that $\mathcal{M}^{\alpha_i}_{n_i}(T) \geq 1 - \epsilon_i$ and $\mathcal{M}^{\alpha_i}_{n_i}(U) = 1$. We consider two cases. (a) If the set $\{n_i \mid i \geq 0\}$ is bounded, then there exists a number n that occurs infinitely often in the sequence $(n_i)_{i\in\mathbb{N}}$, and such that for all $i \geq 0$, there exists a strategy β_i such that $\mathcal{M}^{\beta_i}_n(T) \geq 1 - \epsilon_i$ and $\mathcal{M}^{\beta_i}_n(U) = 1$. Since n is fixed, we can assume w.l.o.g. that the strategies β_i are pure, and since there is a finite number of pure strategies over paths of length at most n, it follows that there is a strategy β that occurs infinitely often among the strategies β_i and such that for all $\epsilon > 0$ we have $\mathcal{M}^\beta_n(T) \geq 1 - \epsilon$, hence $\mathcal{M}^\beta_n(T) = 1$, showing that $\mathcal{M}$ is sure winning for eventually synchronization in T, that is $X_0 \in \langle\langle 1\rangle\rangle^{event}_{sure}(sum_T)$. ($b$) otherwise, the set $\{n_i \mid i \geq 0\}$ is unbounded and we can assume w.l.o.g. that $n_i \geq k$ for all $i \geq 0$. We claim that the family of strategies α_i ensures limit-sure synchronization in $R = \mathsf{Pre}^k(T)$ with support in $Z = \mathsf{Pre}^k(U)$. Essentially this is because if the probability in T is close to 1 after n_i steps, then k steps before the probability in $\mathsf{Pre}^k(T)$ must be close to 1 as well. Formally, we show that α_i is such that $\mathcal{M}^{\alpha_i}_{n_i-k}(R) \geq 1 - \frac{\epsilon}{\eta^k}$ and $\mathcal{M}^{\alpha_i}_{n_i-k}(Z) = 1$ where η is the smallest positive probability in the transitions of $\mathcal{M}$. Towards contradiction, assume that $\mathcal{M}^{\alpha_i}_{n_i-k}(R) < 1 - \frac{\epsilon}{\eta^k}$, then $\mathcal{M}^{\alpha_i}_{n_i-k}(Q \setminus R) > \frac{\epsilon}{\eta^k}$ and from every state $q \in Q \setminus R$, no matter which sequence of actions is

played by α_i for the next k steps, there is a path from q to a state outside of T, thus with probability at least η^k. Hence the probability in $Q \setminus T$ after n_i steps is greater than $\frac{\epsilon}{\eta^k} \cdot \eta^k$, and it follows that $\mathcal{M}_{n_i}^{\alpha_i}(T) < 1 - \epsilon$, in contradiction with the definition of α_i. This shows that $\mathcal{M}_{n_i-k}^{\alpha_i}(R) \geq 1 - \frac{\epsilon}{\eta^k}$, and an argument analogous to the proof of Lemma 4.2 shows that $\mathcal{M}_{n_i-k}^{\alpha_i}(Z) = 1$. It follows that $X_0 \in \langle\langle 1 \rangle\rangle_{limit}^{event}(sum_R, Z)$ and the proof is complete.

□

Thanks to Lemma 4.6, since membership problem of sure eventually synchronization is already solved in Section 4.2.1, it suffices to solve the membership problem of limit-sure eventually synchronization in the target set $R = \mathsf{Pre}^k(T)$ and support $Z = \mathsf{Pre}^k(U)$ with arbitrary k, instead of T and U. We can choose k such that both $\mathsf{Pre}^k(T)$ and $\mathsf{Pre}^k(U)$ lie in the periodic part of the sequence of pairs of predecessors $(\mathsf{Pre}^i(T), \mathsf{Pre}^i(U))$. We can assume that $k \leq 3^{|Q|}$ since $\mathsf{Pre}^i(T) \subseteq \mathsf{Pre}^i(U) \subseteq Q$ for all $i \geq 0$. For such value of k the limit-sure problem is conceptually simpler: once some probability is injected in $R = \mathsf{Pre}^k(T)$, it can loop through the sequence of predecessors and visit R infinitely often (every r steps, where $r \leq 3^{|Q|}$ is the period of the sequence of pairs of predecessors). It follows that if a strategy ensures with probability 1 that the set R can be reached by finite paths whose lengths are congruent modulo r, then the whole probability mass can indeed synchronously accumulate in R in the limit.

Therefore, limit-sure eventually synchronization in R reduces to standard limit-sure reachability condition $\Diamond R$ and the additional requirement that the numbers of steps at which the target set is reached be congruent modulo r. In the case of limit-sure eventually synchronization with support in Z, we also need to ensure that no mass of probability leaves the sequence $\mathsf{Pre}^i(Z)$. In a state $q \in \mathsf{Pre}^i(Z)$, we say that an action $a \in \mathsf{A}$ is *Z-safe* at position i if[2] $\mathsf{post}(q, a) \subseteq \mathsf{Pre}^{i-1}(Z)$. In states $q \notin \mathsf{Pre}^i(Z)$ there is no Z-safe action at position i.

To encode the above requirements, we construct an MDP $\mathcal{M}_Z \times [r]$ that allows only Z-safe actions to be played (and then mimics the original MDP), and tracks the position (modulo r) in the sequence of predecessors, thus simply decrementing the position on each transition since all successors of a state $q \in \mathsf{Pre}^i(Z)$ on a Z-safe action are in $\mathsf{Pre}^{i-1}(Z)$.

Formally, if $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ then $\mathcal{M}_Z \times [r] = \langle Q', \mathsf{A}, \delta' \rangle$ where

- $Q' = Q \times \{r-1, \ldots, 1, 0\} \cup \{\mathsf{sink}\}$; intuitively, we expect that $q \in \mathsf{Pre}^i(Z)$ in the reachable states $\langle q, i \rangle$ consisting of a state q of $\mathcal{M}$ and a *position* i in the predecessor sequence;
- δ' is defined as follows (assuming an arithmetic modulo r on positions) for all $\langle q, i \rangle \in Q'$ and $a \in \mathsf{A}$: if a is a Z-safe action in q at position i, then $\delta'(\langle q, i \rangle, a)(\langle q', i-1 \rangle) = \delta(q, a)(q')$, otherwise $\delta'(\langle q, i \rangle, a)(\mathsf{sink}) = 1$ (and sink is absorbing).

Note that the size of the MDP $\mathcal{M}_Z \times [r]$ is exponential in the size of $\mathcal{M}$ (since r is at most $3^{|Q|}$).

2. Since $\mathsf{Pre}^r(Z) = Z$ and $\mathsf{Pre}^r(R) = R$, we assume a modular arithmetic for exponents of Pre, that is $\mathsf{Pre}^x(\cdot)$ is defined as $\mathsf{Pre}^{x \bmod r}(\cdot)$. For example $\mathsf{Pre}^{-1}(Z)$ is $\mathsf{Pre}^{r-1}(Z)$.

Lemma 4.7. *Let $\mathcal{M}$ be an MDP and $R \subseteq Z$ be two sets of states such that $\mathsf{Pre}^r(R) = R$ and $\mathsf{Pre}^r(Z) = Z$ where $r > 0$. Then a state q_{init} is limit-sure eventually synchronizing in R with support in Z ($q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_R, Z)$) if and only if there exists $0 \leq t < r$ such that $\langle q_{\mathsf{init}}, t \rangle$ is limit-sure winning for the reachability condition $\Diamond(R \times \{0\})$ in the MDP $\mathcal{M}_Z \times [r]$.*

Proof. For the first direction of the lemma, assume that q_{init} is limit-sure eventually synchronizing in R with support in Z, and for $\epsilon > 0$ let β be a strategy such that $\mathcal{M}^\beta_k(Z) = 1$ and $\mathcal{M}^\beta_k(R) \geq 1 - \epsilon$ for some number k of steps. Let $0 \leq t \leq r$ such that $t = k \mod r$ and let $R_0 = R \times \{0\}$. We show that from initial state (q_{init}, t) the strategy α in $\mathcal{M}_Z \times [r]$ that mimics (copies) the strategy β is limit-sure winning for the reachability condition $\Diamond R_0$: it follows from Lemma 4.2 that α plays only Z-safe actions, and since $Pr^\alpha(\Diamond R_0) \geq Pr^\alpha(\Diamond^k R_0) = \mathcal{M}^\beta_k(R) \geq 1 - \epsilon$, the result follows.

For the converse direction, let $R_0 = R \times \{0\}$ and assuming that there exists $0 \leq t < r$ such that $\langle q_{\mathsf{init}}, t \rangle$ is limit-sure winning for the reachability condition $\Diamond R_0$ in $\mathcal{M}_Z \times [r]$, show that q_{init} is limit-sure synchronizing in target set R with exact support in Z. Since the winning region of limit-sure and almost-sure reachability coincide for MDPs [dAHK07], there exists a (pure) strategy α in $\mathcal{M}_Z \times [r]$ with initial state $\langle q, t \rangle$ such that $\Pr^\alpha(\Diamond R_0) = 1$.

Given $\epsilon > 0$, we construct from α a pure strategy β in $\mathcal{M}$ that is $(1 - \epsilon)$-synchronizing in R with support in Z. Given a finite path $\rho = q_0 a_0 q_1 a_1 \dots q_n$ in $\mathcal{M}$ (with $q_0 = q_{\mathsf{init}}$), there is a corresponding path $\rho' = \langle q_0, k_0 \rangle a_0 \langle q_1, k_1 \rangle a_1 \dots \langle q_n, k_n \rangle$ in $\mathcal{M}_Z \times [r]$ where $k_0 = t$ and $k_{i+1} = k_i - 1$ for all $i \geq 0$. Since the sequence $k_0, k_1, \dots$ is uniquely determined from ρ, there is a clear bijection between the paths in $\mathcal{M}$ and the paths in $\mathcal{M}_Z \times [r]$ that we often omit to apply and mention. Define the strategy β as follows: if $q_n \in \mathsf{Pre}^{k_n}(R)$, then there exists an action a such that $\mathsf{post}(q_n, a) \subseteq \mathsf{Pre}^{k_n - 1}(R)$ and we define $\beta(\rho) = a$, otherwise let $\beta(\rho) = \alpha(\rho')$. Thus β mimics α (thus playing only Z-safe actions) unless a state q is reached at step n such that $q \in \mathsf{Pre}^{t-n}(R)$, and then β switches to always playing actions that are R-safe (and thus also Z-safe since $R \subseteq Z$). We now prove that β is limit-sure eventually synchronizing in target set R with support in Z. First since β plays only Z-safe actions, it follows for all k such that $t - k = 0$ (modulo r), all states reached from q_{init} with positive probability after k steps are in Z. Hence $\mathcal{M}^\beta_k(Z) = 1$ for all such k.

Second, we show that given $\epsilon > 0$ there exists k such that $t - k = 0$ and $\mathcal{M}^\beta_k(R) \geq 1 - \epsilon$, thus also $\mathcal{M}^\beta_k(Z) = 1$ and β is limit-sure eventually synchronizing in target set R with support in Z. To show this, recall that $\Pr^\alpha(\Diamond R_0) = 1$, and therefore $\Pr^\alpha(\Diamond^{\leq k} R_0) \geq 1 - \epsilon$ for all sufficiently large k. Without loss of generality, consider such a k satisfying $t - k = 0$ (modulo r). For $i = 1, \dots, r-1$, let $R_i = \mathsf{Pre}^i(R) \times \{i\}$. Then trivially $\Pr^\alpha(\Diamond^{\leq k} \bigcup_{i=0}^r R_i) \geq 1 - \epsilon$ and since β agrees with α on all finite paths that do not (yet) visit $\bigcup_{i=0}^r R_i$, given a path ρ that visits $\bigcup_{i=0}^r R_i$ (for the first time), only R-safe actions will be played by β and thus all continuations of ρ in the outcome of β will visit R after k steps (in total). It follows that $\Pr^\beta(\Diamond^{=k} R_0) \geq 1 - \epsilon$, that is $\mathcal{M}^\beta_k(R) \geq 1 - \epsilon$. Note that we used the same strategy β for all $\epsilon > 0$ and thus β is also almost-sure eventually synchronizing in R.

□

The proof of Lemma 4.7 immediately gives the following corollary:

Corollary 4.2. *Let $\mathcal{M}$ be an MDP and R be a set of states such that $\mathsf{Pre}^r(R) = R$ for some $r > 0$. Then, $\langle\langle 1\rangle\rangle^{event}_{almost}(sum_R) = \langle\langle 1\rangle\rangle^{event}_{limit}(sum_R)$.*

Since deciding limit-sure reachability is PTIME-complete, it follows from Lemma 4.7 that limit-sure synchronization (with exact support) can be decided in EXPTIME. We show that the problem can be solved in PSPACE by exploiting the special structure of the exponential MDP in Lemma 4.7. We conclude this subsection by showing that limit-sure synchronization with exact support is PSPACE-complete (even in the special case of a trivial support).

Lemma 4.8. *For MDPs, the membership problem for limit-sure eventually synchronization with exact support is in* PSPACE.

Proof. We present a (non-deterministic) PSPACE algorithm to decide, given an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta\rangle$, an initial state q_{init}, and two sets $T \subseteq U \subseteq Q$, whether q_{init} is limit-sure eventually synchronizing in T with support in U.

First, the algorithm computes numbers $k \geq 0$ and $r > 0$ such that for $R = \mathsf{Pre}^k(T)$ and $Z = \mathsf{Pre}^k(U)$ we have $\mathsf{Pre}^r(R) = R$ and $\mathsf{Pre}^r(Z) = Z$. As discussed before, this can be done by guessing $k, r \leq 3^{|Q|}$. By Lemma 4.6, we have $\langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U) = \langle\langle 1\rangle\rangle^{event}_{limit}(sum_R, Z) \cup \langle\langle 1\rangle\rangle^{event}_{sure}(sum_T)$, and since sure eventually synchronization in T can be decided in PSPACE (by Theorem 4.3), it suffices to decide limit-sure eventually synchronizing in R with support in Z in PSPACE. According to Lemma 4.7, it is therefore sufficient to show that deciding limit-sure winning for the (standard) reachability condition $\Diamond(R\times\{0\})$ in the MDP $\mathcal{M}_Z\times[r]$ can be done in polynomial space. As we cannot afford to construct the exponential-size MDP $\mathcal{M}_Z \times [r]$, the algorithm relies on the following characterization of the limit-sure winning set for reachability conditions in MDPs. It is known that the winning region for limit-sure and almost-sure reachability coincide [dAHK07], and pure memoryless strategies are sufficient. Therefore, we can see that the almost-sure winning set W for the reachability condition $\Diamond(R \times \{0\})$ satisfies the following property: there exists a memoryless strategy $\alpha : W \to \mathsf{A}$ such that (1) W is closed, that is $\mathsf{post}(q, \alpha(q)) \subseteq W$ for all $q \in W$, and (2) in the graph of the Markov chain $M(\alpha)$, for every state $q \in W$, there is a path (of length at most $|W|$) from q to $R \times \{0\}$.

This property ensures that from every state in W, the target set $R \times \{0\}$ is reached within $|W|$ steps with positive (and bounded) probability, and since W is closed it ensures that $R \times \{0\}$ is reached with probability 1 in the long run. Thus any set W satisfying the above property is almost-sure winning.

Our algorithm will guess and explore on the fly a set W to ensure that it satisfies this property, and contains the state $\langle q_{\mathsf{init}}, t\rangle$ for some $t < r$. As we cannot afford to explicitly guess W (remember that W could be of exponential size), we decompose W into *slices* $W_0, W_1, \ldots$ such that $W_i \subseteq Q$ and $W_i \times \{-i \mod r\} = W \cap (Q \times \{-i \mod r\})$. We start by guessing W_0, and we use the property that in $\mathcal{M}_Z \times [r]$, from a state (q, j) under all

Z-safe actions, all successors are of the form $(\cdot, j-1)$. It follows that the successors of the states in $W_i \times \{-i\}$ should lie in the slice $W_{i+1} \times \{-i-1\}$, and we can guess on the fly the next slice $W_{i+1} \subseteq Q$ by guessing for each state q in a slice W_i an action a_q such that $\bigcup_{q \in W_i} \mathsf{post}(q, a_q) \subseteq W_{i+1}$. Moreover, we need to check the existence of a path from every state in W to $R \times \{0\}$. As W is closed, it is sufficient to check that there is a path from every state in $W_0 \times \{0\}$ to $R \times \{0\}$. To do this we guess along with the slices $W_0, W_1, \ldots$ a sequence of sets $P_0, P_1, \ldots$ where $P_i \subseteq W_i$ contains the states of slice W_i that belong to the guessed paths. Formally, $P_0 = W_0$, and for all $i \geq 0$, the set P_{i+1} is such that $\mathsf{post}(q, a_q) \cap P_{i+1} \neq \emptyset$ for all $q \in P'_i$ (where $P'_i = P_i \setminus R$ if i is a multiple of r, and $P'_i = P_i$ otherwise), that is P_{i+1} contains a successor of every state in P_i that is not already in the target R (at position 0 modulo r).

We need polynomial space to store the first slice W_0, the current slice W_i and the set P_i, and the value of i (in binary). As $\mathcal{M}_Z \times [r]$ has $|Q| \cdot r$ states, the algorithm runs for $|Q| \cdot r$ iterations and then checks that (1) $W_{|Q| \cdot r} \subseteq W_0$ to ensure that $W = \bigcup_{i \leq |Q| \cdot r} W_i \times \{i \bmod r\}$ is closed, (2) $P_{|Q| \cdot r} = \emptyset$ showing that from every state in $W_0 \times \{0\}$ there is a path to $R \times \{0\}$ (and thus also from all states in W), and (3) the state q_{init} occurs in some slice W_i. The correctness of the algorithm follows from the characterization of the almost-sure winning set for reachability in MDPs: if some state $\langle q_{\mathsf{init}}, t \rangle$ is limit-sure winning, then the algorithm accepts by guessing (slice by slice) the almost-sure winning set W and the paths from $W_0 \times \{0\}$ to $R \times \{0\}$ (at position 0 modulo r), and otherwise any set (and paths) correctly guessed by the algorithm would not contain q_{init} in any slice.

□

It follows from the proof of Lemma 4.7 that all winning modes for eventually synchronizing are independent of the numerical value of the positive transition probabilities.

Corollary 4.3. *Let $\mathcal{M}$ be an MDP and $T \subseteq U$ be two sets of states. For all winning modes $\mu \in \{sure, almost, limit\}$, and for all two pairs of distributions d, d' with $\mathsf{Supp}(d) = \mathsf{Supp}(d')$, we have $d \in \langle\langle 1 \rangle\rangle_{\mu}^{event}(sum_T, U)$ if and only if $d' \in \langle\langle 1 \rangle\rangle_{\mu}^{event}(sum_T, U)$.*

To establish the PSPACE-hardness for limit-sure eventually synchronizing in MDPs, we use a reduction from the universal finiteness problem for 1L-AFAs.

Lemma 4.9. *For MDPs, the membership problem for $\langle\langle 1 \rangle\rangle_{limit}^{event}(sum_T)$ is* PSPACE-hard *even if T is a singleton.*

Proof. The proof is by a reduction from the universal finiteness problem for one-letter alternating automata (1L-AFA), which is PSPACE-complete (by Lemma 3.2). It is easy to see that this problem remains PSPACE-complete even if the set T of accepting states of the 1L-AFA is a singleton, and given the tight relation between 1L-AFA and MDP (see Subsection 3.4.2), it follows from the definition of the universal finiteness problem that

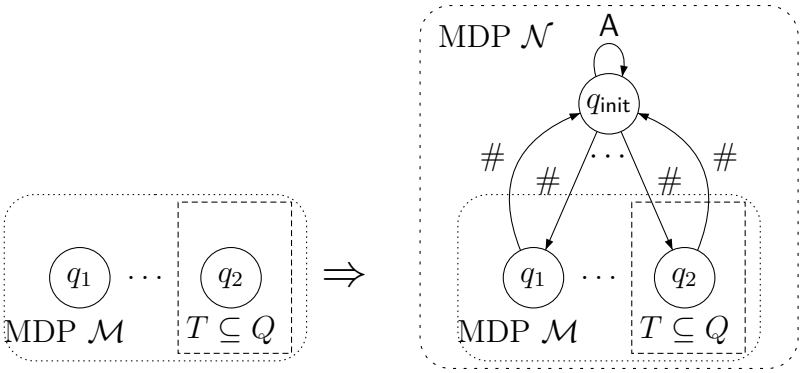

Figure 4.6: Sketch of reduction to show PSPACE-hardness of the membership problem for limit-sure eventually synchronizing.

deciding, in an MDP $\mathcal{M}$, whether the sequence $\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$ is PSPACE-complete.

The reduction is as follows (see also Figure 4.6). Given an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ and a singleton $T \subseteq Q$, we construct an MDP $\mathcal{N} = \langle Q', \mathsf{A}', \delta' \rangle$ with state space $Q' = Q \uplus \{q_{\mathsf{init}}\}$ such that $\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$ if and only if q_{init} is limit-sure eventually synchronizing in T. The MDP $\mathcal{N}$ is essentially a copy of $\mathcal{M}$ with alphabet $\mathsf{A} \uplus \{\#\}$ and the transition function on action $\#$ is the uniform distribution on Q from q_{init}, and the Dirac distribution on q_{init} from the other states $q \in Q$. There are self-loops on q_{init} for all other actions $a \in \mathsf{A}$. Formally, the transition function δ' is defined as follows, for all $q \in Q$:

- $\delta'(q, a) = \delta(q, a)$ for all $a \in \mathsf{A}$ (copy of $\mathcal{M}$), and $\delta'(q, \#)(q_{\mathsf{init}}) = 1$;
- $\delta'(q_{\mathsf{init}}, a)(q_{\mathsf{init}}) = 1$ for all $a \in \mathsf{A}$, and $\delta'(q_{\mathsf{init}}, \#)(q) = \frac{1}{|Q|}$.

We establish the correctness of the reduction as follows. For the first direction, assume that $\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$. Then since $\mathcal{N}$ embeds a copy of $\mathcal{M}$ it follows that $\mathsf{Pre}^n_{\mathcal{N}}(T) \neq \emptyset$ for all $n \geq 0$ and there exist numbers $k_0, r \leq 2^{|Q|}$ such that $\mathsf{Pre}^{k_0+r}_{\mathcal{N}}(T) = \mathsf{Pre}^{k_0}_{\mathcal{N}}(T) \neq \emptyset$. Using Lemma 4.6 with $k = k_0$ and $R = \mathsf{Pre}^{k_0}_{\mathcal{N}}(T)$ (and $U = Z = Q'$ is the trivial support), it is sufficient to prove that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{limit}(R)$ to get $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{limit}(T)$ (in $\mathcal{N}$). We show the stronger statement that q_{init} is actually almost-sure eventually synchronizing in R with the pure strategy α defined as follows, for all play prefix ρ (let $m = |\rho| \bmod r$):

- if $\mathsf{Last}(\rho) = q_{\mathsf{init}}$, then $\alpha(\rho) = \#$;
- if $\mathsf{Last}(\rho) = q \in Q$, then
  - if $q \in \mathsf{Pre}^{r-m}_{\mathcal{N}}(R)$, then $\alpha(\rho)$ plays a R-safe action at position $r - m$;
  - otherwise, $\alpha(\rho) = \#$.

The strategy α ensures that the probability mass that is not (yet) in the sequence of predecessors $\mathsf{Pre}^n_{\mathcal{N}}(R)$ goes to q_{init}, where by playing $\#$ at least a fraction $\frac{1}{|Q|}$ of it would reach the sequence of predecessors (at a synchronized position). It follows that after $2i$ steps, the probability mass in q_{init} is $(1 - \frac{1}{|Q|})^i$ and the probability mass in the sequence of

predecessors is $1-(1-\frac{1}{|Q|})^i$. For $i \to \infty$, the probability in the sequence of predecessors tends to 1 and since $\mathsf{Pre}^n_{\mathcal{N}}(R) = R$ for all positions n that are a multiple of r, we get $\sup_n \mathcal{M}^\alpha_n(R) = 1$ and $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(R)$.

For the converse direction, assume that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{limit}(T)$ is limit-sure eventually synchronizing in T. By Lemma 4.6, either (1) q_{init} is limit-sure eventually synchronizing in $\mathsf{Pre}^n_{\mathcal{N}}(T)$ for all $n \geq 0$, and then it follows that $\mathsf{Pre}^n_{\mathcal{N}}(T) \neq \emptyset$ for all $n \geq 0$, or (2) q_{init} is sure eventually synchronizing in T, and then since only the action $\#$ leaves the state q_{init} (and $\mathsf{post}(q_{\mathsf{init}}, \#) = Q$), the characterization of Lemma 4.2 shows that $Q \subseteq \mathsf{Pre}^k_{\mathcal{N}}(T)$ for some $k \geq 0$, and since $Q \subseteq \mathsf{Pre}_{\mathcal{N}}(Q)$ and $\mathsf{Pre}_{\mathcal{N}}(\cdot)$ is a monotone operator, it follows that $Q \subseteq \mathsf{Pre}^n_{\mathcal{N}}(T)$ for all $n \geq k$ and thus $\mathsf{Pre}^n_{\mathcal{N}}(T) \neq \emptyset$ for all $n \geq 0$. We conclude the proof by noting that $\mathsf{Pre}^n_{\mathcal{M}}(T) = \mathsf{Pre}^n_{\mathcal{N}}(T) \cap Q$ and therefore $\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$.

□

The example in the proof of Lemma 4.3 can be used to show that the memory needed by a family of strategies to win limit-sure eventually synchronizing condition (in the target set $T = \{q_2\}$) is unbounded.

The following theorem summarizes the results for limit-sure eventually synchronization in MDPs.

Theorem 4.5. *For limit-sure eventually synchronizing condition (with or without exact support) in MDPs and for all functions $f \in \{max_T, sum_T\}$:*

1. *(Complexity). The membership problem is* PSPACE-complete.
2. *(Memory). Unbounded memory is required for both pure and randomized strategies, and pure strategies are sufficient.*

4.3 Eventually synchronization in PAs

As discussed in the beginning of Section 4.2, to show the complexity results of the membership problem for eventually synchronizing conditions in PAs, we only consider the emptiness problem of the eventually synchronizing languages according to function *sum* and from Dirac initial distributions too. Results presented in this section are listed in Table 4.2.

	Always		Eventually
	sum	*max*	
Sure	PSPACE-complete	PTIME	PSPACE-complete
Almost-sure			undecidable
Limit-sure			undecidable

Table 4.2: Computational complexity of the emptiness problem of always and eventually synchronizing languages in PAs.

4.3.1 Sure eventually synchronization

Given a target set T in a PA, the emptiness problem for sure eventually synchronizing language in T can be solved by means of *subset construction*, as it is solved for NFAs. For a PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$, the subset construction consists in $\mathcal{N}_\mathcal{P} = \langle Q', \mathsf{A}, \Delta \rangle$ where the state space $Q' = 2^Q \setminus \{\emptyset\}$ is the set of all *cells*, i.e. subsets of states in $\mathcal{P}$, and the transition function $\Delta : Q' \times \mathsf{A} \to Q'$ defined as $\Delta'(c, a) = c'$ where $\mathsf{post}(c, a) = c'$ and $a \in \mathsf{A}$. We equip the subset construction of $\mathcal{P}$ with the set $\mathcal{F} = \{S \in Q' \mid S \subseteq T\}$ of accepting cells. A state q_{init} is sure-winning for eventually synchronizing in T if and only if the subset construction $\mathcal{N}_\mathcal{P}$, equipped with initial cell $\{q_{\mathsf{init}}\}$ and $\mathcal{F}$, accepts a word.

Lemma 4.10. *Let $\mathcal{P}$ be a PA and T be a target set. From all states q_{init}, we have $\mathcal{L}^{event}_{sure}(sum_T) \neq \emptyset$ if and only if the language of the subset construction $\mathcal{N}_\mathcal{P}$, equipped with the initial cell $\{q_{\mathsf{init}}\}$ and set $\mathcal{F} = \{S \in Q' \mid S \subseteq T\}$ of accepting cells, is non-empty.*

Proof. We prove the following equivalence by induction (on the length i of accepting words): for all initial distributions X_0, there exists a word w such that $\mathcal{P}^w_i(T) = 1$ (we say that the word w is sure-winning in i steps from X_0) if and only if there exists an accepting word v with length $|v| = i$ for the subset construction $\mathcal{N}_\mathcal{P}$ equipped with the initial cell $\mathsf{Supp}(X_0)$ and set $\mathcal{F} = \{S \in Q' \mid S \subseteq T\}$ of accepting cells. The case $i = 0$ trivially holds since for all words w, we have $\mathcal{P}^w_0(T) = 1$ if and only if $X_0(T) = 1$ implying that the initial cell $\mathsf{Supp}(X_0)$ is an accepting cell in the subset construction.

Assume that the equivalence holds for all $i < n$. For the induction step, show that $\mathcal{P}$ is sure eventually synchronizing from X_0 (in n steps) if and only if there exists some letter a such that $\mathcal{P}$ is sure eventually synchronizing from the distribution d where $d(q) = \sum_{q' \in Q} X_0(q') \cdot \delta(q', a)(q)$, meaning that there exists an accepting word v with length $n-1$ for $\mathcal{P}$ equipped with the initial cell $\mathsf{Supp}(d)$, and thus the word $a \cdot v$ with length n is accepting for $\mathcal{P}$ starting in $\mathsf{Supp}(X_0)$. First, if $\mathcal{P}$ is sure eventually synchronizing from the distribution d by some synchronizing word w', then so is by the input word $a \cdot w'$ from X_0. For the other direction, assume towards contradiction that $\mathcal{P}$ is sure eventually synchronizing from X_0 (in n steps), but for all letters a, the distribution d is not sure eventually synchronizing. Thus, no matter the input words, there is no way that the probability mass in T is 1 after n steps from X_0, in contradiction with the fact that $\mathcal{P}$ is

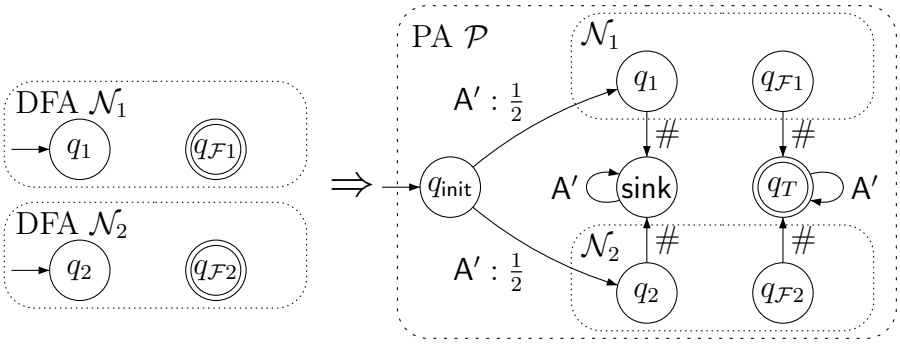

Figure 4.7: Sketch of reduction to show PSPACE-hardness of the emptiness problem for sure eventually synchronizing language in PAs.

sure eventually synchronizing from X_0 in n steps. It follows that the induction step holds, and thus the statement of lemma holds for all initial distributions.

□

By Proof of Lemma 4.10, a PA $\mathcal{P}$ is sure eventually synchronizing in the target set T by a pure word w if and only if there exists a finite word v such that $w \in v \cdot \mathsf{A}^\omega$ where the subset construction $\mathcal{N}_\mathcal{P}$ equipped with the initial cell $\{q_{\mathsf{init}}\}$ and set $\mathcal{F} = \{S \in Q' \mid S \subseteq T\}$ of accepting cells, accepts the finite word v. It implies that the set of all pure sure eventually synchronizing words in a PA is ω-regular. Moreover, the PSPACE upper bounds follows for the emptiness problem of sure eventually synchronizing languages in PAs. The PSPACE-hardness is by a reduction from *finite automata intersection* based on an idea close to the reduction provided to show PSPACE-hardness for finite synchronization from a subset in NFAs (in Subsection 3.1.1).

Lemma 4.11. *For PAs, the emptiness problem for $\mathcal{L}_{sure}^{event}(sum_T)$ is* PSPACE-hard *even if T is a singleton.*

Proof. The proof is by a reduction from the finite automata intersection for DFAs, which is known to be PSPACE-complete [Koz77, GJ79, San04]. The finite automata intersection asks, given n DFAs equipped with initial states and accepting sets, whether there exists an accepting finite word for all DFAs. The reduction is as follows: Given n DFAs $\mathcal{N}_1, \mathcal{N}_2, \cdots, \mathcal{N}_n$ over the common alphabet A where $\mathcal{N}_i = \langle Q_i, \mathsf{A}, \Delta_i \rangle$ for all $1 \leq i \leq n$, and where each $\mathcal{N}_i$ is equipped with the initial state q_i and the set $\mathcal{F}_i$ of accepting states, we construct the PA $\mathcal{P} = \langle Q, \mathsf{A}', \delta \rangle$ as sketched in Figure 4.7 for the case $n = 2$. We augment the alphabet with a new letter: $\mathsf{A}' = \mathsf{A} \cup \{\#\}$. We add a new initial state q_{init}, one copy of the state space of each DFA $\mathcal{N}_i$, and two new absorbing states q_T and sink: $Q = Q_1 \cup \cdots \cup Q_n \cup \{q_{\mathsf{init}}, q_T, \mathsf{sink}\}$. For the new letter #, let #-transitions in all accepting states $q \in \mathcal{F}_1 \cup \cdots \cup \mathcal{F}_n$ of all automata go to the state q_T whereas all #-transitions in all non-accepting states $q \notin \mathcal{F}_1 \cup \cdots \cup \mathcal{F}_n$ go to the state sink. On all letters in initial state q_{init}, the next successor is randomly chosen from the n initial state $q_1, q_2, \cdots, q_n$,

each with probability $\frac{1}{n}$. For all $1 \leq i \leq n$ and all states $q \in Q_i$ and actions $a \in \mathsf{A}$, let $\delta(q,a) = \Delta_i(q,a)$. Let q_T be the target state.

We establish the correctness of the reduction as follows. First direction of proof is straightforward. By construction, for all finite words w accepted by all DFAs, the PA $\mathcal{P}$ is sure eventually synchronizing in $\{q_T\}$ from q_{init}, by the infinite word $\# \cdot w \cdot \#^\omega$.

Second, assume that $\mathcal{L}^{event}_{sure}(q_T) \neq \emptyset$ from q_{init}, then there exists w that is a sure eventually synchronizing word from q_{init} in $\{q_T\}$. After the first input, regardless the choice of letter, we have probability $\frac{1}{n}$ in each of the states $q_1, q_2, \cdots, q_n$. Since q_T is only reachable via #-transitions, then w has at least one #. Let $w = a \cdot v \cdot \# \cdot v'$ where $a \in \mathsf{A} \cup \{\#\}$, v is the subword after the first letter and up to the first following occurrence of # and $v' \in (\mathsf{A} + \#)^\omega$. We show that v is an accepting word for all DFAs. Towards contradiction, assume that v is not accepted by at least one of the automaton, say $\mathcal{N}_1$. Thus, the run over v from q_1 is not ending in an accepting state and the probability to be in sink would be $\frac{1}{n}$ by inputting the letter # right after v. It means that the probability in q_T will always be bounded by $1 - \frac{1}{n}$ contradicting with w being a sure eventually synchronizing word from q_{init} in $\{q_T\}$.

□

The following theorem summarizes the results for sure eventually synchronizing in PAs.

Theorem 4.6. *For sure eventually synchronizing languages in PAs:*

1. *(Complexity). For all functions $f \in \{max_T, sum_T\}$, the emptiness problem is* PSPACE-complete.
2. *(Regularity). For the function sum_T, the set of all pure sure eventually synchronizing words is ω-regular.*

4.3.2 Almost-sure eventually synchronization

In this subsection, we provide undecidability result for the emptiness problem of almost-sure eventually synchronizing languages in PAs.

Theorem 4.7. *For all functions $f \in \{max_T, sum_T\}$, the emptiness problem of almost-sure eventually synchronizing language in PAs is undecidable.*

Proof. The proof is by a reduction from the value 1 problem for PAs, which is known to be undecidable (See Subsection 2.5.3). From a PA $\mathcal{A} = \langle Q, \mathsf{A}, \delta \rangle$ equipped with an initial state q_{init} and a set $\mathcal{F}$ of accepting states, we construct another PA $\mathcal{P} = \langle Q', \mathsf{A}', \delta' \rangle$ and a

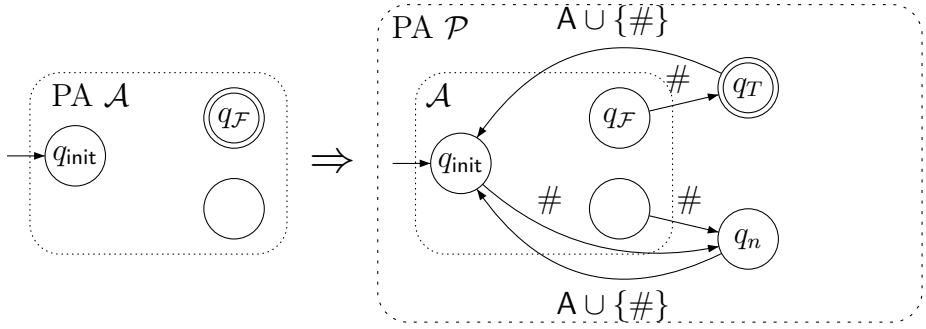

Figure 4.8: Sketch of reduction to show undecidability of the emptiness problem for almost-sure eventually synchronizing languages in PAs.

target state q_T such that the PA $\mathcal{A}$ has value 1 if and only if $\mathcal{L}^{event}_{almost}(q_T) \neq \emptyset$ from q_{init} in the PA $\mathcal{P}$.

The PA $\mathcal{P}$ is a copy of $\mathcal{A}$ with two new states q_n and q_T that are reachable only by a new action $\#$. Formally, $Q' = Q \cup \{q_T, q_n\}$ and $\mathsf{A}' = \mathsf{A} \cup \{\#\}$. The construction is depicted in Figure 4.8. The transition function δ' is defined as follows: $\delta'(q, a) = \delta(q, a)$ for all states $q \in Q$ and $a \in \mathsf{A}$. The $\#$-transition in all accepting states $q \in \mathcal{F}$ goes to the target state q_T: $\delta'(q, \#)(q_T) = 1$ where $q \in \mathcal{F}$. The $\#$-transition in all not-accepting states $q \notin \mathcal{F}$ goes to q_n: $\delta'(q, \#)(q_n) = 1$ where $q \notin \mathcal{F}$. All transitions, in the two new states q_T and q_n, reset the PA $\mathcal{P}$ into the initial state q_{init} meaning that $\delta'(q_n, a)(q_{\mathsf{init}}) = \delta'(q_T, a)(q_{\mathsf{init}}) = 1$ for all actions $a \in \mathsf{A} \cup \{\#\}$.

We establish the correctness of the reduction as follows. First, assume that the PA $\mathcal{A}$ has value 1. Since the definition of value in PAs is $\mathsf{val}(\mathcal{A}) = \sup_{w \in \mathsf{A}^*} \mathsf{Pr}(w)$, if $\mathsf{val}(\mathcal{A}) = 1$ then for all $\epsilon > 0$ there exists a finite word w_ϵ such that $\mathsf{Pr}(w_\epsilon) \geq 1 - \epsilon$. Consider the decreasing sequence $\{\frac{1}{n}\}_{n \geq 2}$ and the finite words $\{w_{\frac{1}{n}}\}_{n \geq 2}$; and construct the infinite word $v = w_{\frac{1}{2}} \cdot \# \cdot \# \cdot w_{\frac{1}{3}} \cdot \# \cdot \# \cdots w_{\frac{1}{n}} \cdot \# \cdot \# \cdots$ represented as $v = \{w_{\frac{1}{n}} \cdot \# \cdot \#\}_{n \geq 2}$. We show that v is an almost-sure eventually synchronizing word in $\{q_T\}$ from q_{init}. We repeat the following argument for all $n = 2, 3, \cdots$: the initial (current) distribution is the Dirac distribution on q_{init}, next we input the word $w_{\frac{1}{n}}$ where $\Pr(w_{\frac{1}{n}}) \geq 1 - \frac{1}{n}$. We thus have probability $1 - \frac{1}{n}$ in the set $\mathcal{F}$, and inputting the following $\#$ leads the mass of probability $1 - \frac{1}{n}$ to the target set $\{q_T\}$ while the remaining mass of probability is in $\{q_n\}$. Now, the second $\#$-transitions reset the PA $\mathcal{P}$, and the current distribution will be the initial Dirac distribution q_{init}. This argument proves that $\mathcal{L}^{event}_{almost}(q_T) \neq \emptyset$ from q_{init}.

For the reverse direction, assume that $\mathcal{L}^{event}_{almost}(q_T) \neq \emptyset$ from q_{init} in the PA $\mathcal{P}$. Consider an infinite word w such that $\sup_n \mathcal{P}^w_n(q_T) = 1$. Let $\epsilon > 0$ and let n be such that $\mathcal{P}^w_n(q_T) \geq 1 - \epsilon$. Since q_T is only reachable via $\#$-transitions then the n-th letter of w must be $\#$. We pick the subword v of w as follows,

- if there is no occurrence of $\#$ before the $(n-1)$-th letter, let v be the subword from the beginning up to $(n-1)$-th letter.

– otherwise, if the last occurrence of # before the $(n-1)$-th letter happens as the m-th letter, check whether $\mathsf{post}(q_{\mathsf{init}}, w_m) = \{q_{\mathsf{init}}\}$:
  1. if yes, let v be the subword after the m-th letter up to $(n-1)$-th letter
  2. otherwise, let v be the subword after the $(m+1)$-th letter up to the $(n-1)$-th letter

  where $w_m = a_0 \cdot a_1 \cdots a_m$ is the prefix of w consisting in the first $m+1$ letters.

So, the finite word v does not contain # and it is thus a valid word for $\mathcal{A}$. We show that $\mathsf{Pr}(v) \geq 1-\epsilon$ in the PA $\mathcal{A}$ (equipped with the set $\mathcal{F}$ of accepting states). The prefix subword of w before v (the first m or $m+1$ letters, if there is any #) resets the automaton into q_{init}; note that if $\mathsf{post}(q_{\mathsf{init}}, w_m) \neq \{q_{\mathsf{init}}\}$, then we surely have $\mathsf{post}(q_{\mathsf{init}}, w_m) = \{q_n, q_T\}$ and thus $\mathsf{post}(q_{\mathsf{init}}, w_{m+1}) = \{q_{\mathsf{init}}\}$. Next, inputting the word v followed by # results in having probability $1-\epsilon$ in the target state q_T. Since q_T is only reachable from accepting states $q \in \mathcal{F}$, so the probability in $\mathcal{F}$ must have been $1-\epsilon$ after reading v and right before the last #. This argument proves that for all $\epsilon \geq 0$, we are able to find a subword v of w that is valid for $\mathcal{A}$ and for which $\mathsf{Pr}(v) \geq 1-\epsilon$. This completes the proof.

□

4.3.3 Limit-sure eventually synchronization

The emptiness problem for limit-sure eventually synchronizing language in PAs is polynomial-time equivalent with value 1 problem in PAs. If a PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$, equipped with the initial state q_{init} and the set $\mathcal{F} \subseteq Q$ of accepting states, has value 1, then there exists a family of finite words w_i for which the acceptance probability is arbitrarily close to 1. By concatenating any infinite word $v \in \mathsf{A}^\omega$ at the end of each finite word w_i in this family, a family of limit-sure eventually synchronizing words in the target set $\mathcal{F}$ from q_{init} is constructed. In the same way, from a family of limit-sure eventually synchronizing infinite words in the target set $\mathcal{F}$ from initial state q_{init} we can construct a family of finite words with acceptance probability arbitrarily close to 1. It happens since the probability in the target set $\mathcal{F}$ will be $1-\epsilon$ after finitely many steps for the limit-sure eventually synchronizing condition.

The following theorem follows from the above observation.

Theorem 4.8. *For all functions $f \in \{max_T, sum_T\}$, the emptiness problem of limit-sure eventually synchronizing language in PAs is undecidable.*

4.4 Discussion

In this section, we shortly discuss the universality problems for sure and almost-sure {always, eventually} synchronizing languages in PA.

Universality problems of always synchronizing languages. For the function max_T, by Proof of Lemma 4.1, we have established that the symbolic-run of a PA over a sure and almost-sure always synchronizing word w is an always 1-synchronizing sequence. Thus, the states where the probability mass $\|\mathcal{P}_i^w\|$ is accumulated form an infinite sequence $q_0q_1q_2\cdots$ where there is a deterministic transition from q_i to q_{i+1} for all $i \in \mathbb{N}$. For a PA $\mathcal{P}$ with universal sure (and thus almost-sure) always synchronizing language, we claim that there is a unique sequence $q_0q_1q_2\cdots$ of states where the probability mass is accumulated along the symbolic runs of $\mathcal{P}$ over all words. Toward contradiction, assume that for all deterministic sequences $q_0q_1q_2\cdots$ in $\mathcal{P}$, there exists a word w and some n where $\|\mathcal{P}_n^w\| < 1$. The number of deterministic sequences in $\mathcal{P}$ are finite, thus we can enumerate all pair of such words and steps with $\langle w_1, n_1\rangle, \cdots \langle w_n, n_k\rangle$ where k is the number of deterministic sequences in $\mathcal{P}$. Thus, the symbolic run of the automaton over the randomized word $v = \frac{1}{n}w_1 + \cdots + \frac{1}{n}w_n$ cannot be 1-synchronized at steps $n_1, \cdots, n_k$ implying that v is not sure (and almost-sure) always synchronizing, a contradiction with the fact that the sure (and almost-sure) always synchronizing languages are universal. Thus, to decide the universality problems of always synchronizing languages according to max_T, it is sufficient to check for all states q reachable from q_{init}, whether $q \in T$ and $|\mathsf{post}(q, \mathsf{A})| = 1$. It implies that the graph underling the PA must be lasso-shaped, which can be checked in PTIME.

A similar argument for the function sum_T shows that to decide the universality problem of sure (and almost-sure) always synchronizing language, it is sufficient to check whether $q \in T$ for all states q reachable from q_{init}, which can be checked in PTIME.

Universality problem of sure eventually synchronizing language. Consider a PA $\mathcal{P}$ such that the sure eventually synchronizing language according to max_T is universal. By definition, for all words w the symbolic run $\mathcal{P}_0^w\mathcal{P}_1^w\mathcal{P}_2^w\cdots$ is eventually 1-synchronized in a singleton $\{q\}$ where $q \in T$, i.e. there exists n such that $\mathcal{P}_n^w(q) = 1$, and by Lemma 4.10, we know that $n \leq 2^{|Q|}$. We claim that there exists some step $n \leq 2^{|Q|}$ such that for all words w, we have $\|\mathcal{P}_n^w\|_T = 1$. Toward contradiction, assume that for all steps $n \leq 2^{|Q|}$, there exists a word w_n such that $\|\mathcal{P}_n^{w_n}\|_T \neq 1$. Then the symbolic run of the automaton over the randomized word $v = \frac{1}{k}w_1 + \cdots + \frac{1}{k}w_k$ where $k = 2^{|Q|}$ is never 1-synchronized, implying that v is not sure eventually synchronizing according to max_T, a contradiction with the fact that $\mathcal{P}$ has a universal sure eventually synchronizing language according to max_T. A similar argument proves that the state q where the probability mass is accumulated at step n is unique for all words too. Thus, to decide the universality problem of sure eventually synchronizing language according to max_T, one can compute the sequence $s_0s_1s_2\cdots$ where $s_0 = \{q_{\mathsf{init}}\}$ and $s_i = \mathsf{post}(s_{i-1}, \mathsf{A})$ for all $i \in \mathbb{N}$, and check whether there exists some n such that $s_n = \{q\}$ is a singleton and $q \in T$. The sequence $s_0s_1s_2\cdots$ is ultimately periodic meaning that there are $k, n \leq 2^{|Q|}$ such that $s_n = s_{n+k}$; as a result universality problem of sure eventually synchronizing language according to max_T can be decided in PSPACE.

A similar argument for the function sum_T shows that to decide the universality problem of sure eventually synchronizing language, the sequence $s_0s_1s_2\cdots$ where $s_0 = \{q_{\mathsf{init}}\}$ and

$s_i = \mathsf{post}(s_{i-1}, \mathsf{A})$ for all $i \in \mathbb{N}$, can be computed and then one can check whether there exists some n such that $s_n \subseteq T$, which can be done in PSPACE.

We do not provide a matching lower bound for the universality problem of sure eventually synchronizing languages. However, there is a reduction from *tautology problem* proving that deciding whether all pure words are sure eventually synchronizing according to the function sum_T, is co-NP-hard.

Universality problem of almost-sure eventually synchronizing language. We recall that if $x = \sup_n x_n$ for an infinite sequence $x_0x_1x_2\cdots$ of bounded values, then x is equal or greater than all x_i and either there exists a certain x_n such that $x = x_n$ or $x = \limsup_{n\to\infty} x_n$,

For the function max_T, consider a PA $\mathcal{P}$ that the almost-sure eventually synchronizing language is universal. By definition, $\sup_{n\in\mathbb{N}}\|\mathcal{P}_n^w\|_T = 1$ for all words w. There are three cases:

- either for all words w, there exists n such that $\|\mathcal{P}_n^w\|_T = 1$ meaning that the universality problem of almost-sure eventually synchronizing language reduces to the universality problem of sure eventually synchronizing language.
- or $\limsup_{n\to\infty}\|\mathcal{P}_n^w\|_T = 1$ for all words w, meaning that the universality problem of almost-sure eventually synchronizing language reduces to the universality problem of almost-sure weakly synchronizing language, which is discussed in Section 5.3.
- or there must be two words w and w' such that there exists n such that $\|\mathcal{P}_n^w\|_T = 1$ but $\limsup_{n\to\infty}\|\mathcal{P}_n^w\|_T < 1$, and $\|\mathcal{P}_i^{w'}\|_T \neq 1$ for all $i \in \mathbb{N}$ but $\limsup_{n\to\infty}\|\mathcal{P}_n^{w'}\|_T = 1$. We see that for the randomized word $v = \frac{1}{2}w + \frac{1}{2}w'$, there exists no n such that $\|\mathcal{P}_n^v\|_T = 1$ and also $\limsup_{n\to\infty}\|\mathcal{P}_n^v\|_T < 1$, a contradiction with the fact that $\mathcal{P}$ has a universal almost-sure eventually synchronizing language. It proves that the third case never happens.

A similar argument for the function sum_T shows that to decide the universality problem for almost-sure eventually synchronizing language, it is sufficient to check if the sure eventually synchronizing language of the PA, or if the almost-sure weakly synchronizing language is universal.

Since for both function sum_T and max_T, the universality problem of almost-sure weakly synchronizing languages is in PSPACE, by Lemma 5.11 on page 111, the PSPACE membership follows for the universality problem of almost-sure eventually synchronizing languages in PAs. Lemma 4.12 provides a matching lower bound.

Lemma 4.12. *The universality problem of almost-sure eventually synchronizing language according to sum_T in PAs is* PSPACE-hard*, even if T is a singleton.*

Proof. We present a proof using a reduction from a PSPACE-complete problem so called *initial state problem*. Given an NFA $\mathcal{N} = \langle L, \mathsf{A}, \Delta\rangle$ and a set of states $\mathcal{F}$, the *initial state problem* is to decide whether there exists an initial state $\ell_0 \in L$ and an infinite word $w \in \mathsf{A}^\omega$ such that all runs $\rho = r_0r_1r_2\cdots$ of $\mathcal{N}$ starting in ℓ_0, over w avoid $\mathcal{F}$, i.e. $r_i \notin \mathcal{F}$ for all $i \in \mathbb{N}$. From the results of [CSV08, TBG09], it follows that the initial state problem is

PSPACE-complete. We present a polynomial-time reduction from the initial state problem to the universality problem, establishing the PSPACE-hardness result.

Let $\mathcal{N} = \langle L, \mathsf{A}, \Delta \rangle$ be an NFA with the set $\mathcal{F} \neq \emptyset$ of states. We assume, without loss of generality, that $\mathcal{N}$ is complete and all states in $\mathcal{N}$ are all reachable (from some state ℓ_0). From $\mathcal{N}$, we construct a PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ where the set of states is augmented with a new absorbing state synch, formally $Q = L \cup \{\mathsf{synch}\}$. The transition function δ is defined as follows, for all $q \in L$ and $a \in \mathsf{A}$,

– if $q \notin \mathcal{F}$, then $\delta(q, a)$ is the uniform distribution over $\Delta(q, a)$,
– and if $q \in \mathcal{F}$, $\delta(q, a)(q') = \frac{1}{2|\Delta(q,a)|}$ for all $q' \in \Delta(q, a)$ and $\delta(q, a)(\mathsf{synch}) = \frac{1}{2}$.

Let $q_{\mathsf{init}} = \ell_0$ be the state from which all states are reachable. We show that the answer to the initial state problem for $\mathcal{N}$ is Yes if and only if the almost-sure eventually synchronizing language of $\mathcal{P}$ in $\{\mathsf{synch}\}$ is not universal.

First, assume that the answer to the initial state problem for $\mathcal{N}$ is Yes. Thus, there exists some state $\hat{q}$ and a word $w \in \mathsf{A}^\omega$ satisfying the initial state problem. We construct a word that is not almost-sure eventually synchronizing for $\mathcal{P}$. First, consider the $|Q|$-times repetition of the uniform distribution d_u over A. Then, with positive probability the state synch is reached, and also with positive probability the state $\hat{q}$ is reached, say after k steps. Let $w' \in \mathsf{A}^\omega$ be such that $w = v \cdot w'$ and $|v| = |Q| - k$. Note that from state $\hat{q}$ the finite word v is played with positive probability by the repetition of uniform distribution d_u. Therefore, on the word $(d_u)^{|Q|} \cdot w'$, with some positive probability the state synch is never reached, and thus $\mathcal{P}$ is not almost-sure eventually synchronizing in $\{\mathsf{synch}\}$, and the almost-sure eventually synchronizing language of $\mathcal{P}$ is not universal.

Second, assume that the almost-sure eventually synchronizing language of $\mathcal{P}$ is not universal. Since synch is an absorbing state the eventually and weakly synchronization coincide, then by the construction in Lemma 5.10 on page 108, there exists a pure word $w \in \mathsf{A}^\omega$ such that from the initial state, all runs over w avoid the terminal end component $\{\mathsf{synch}\}$, and therefore also avoid $\mathcal{F}$. Hence, the answer to the initial state problem for $\mathcal{N}$ is Yes.

□

From the above arguments and Lemma 4.12, it appears that for all functions $f \in \{max_T, sum_T\}$, the universality problem of almost-sure eventually synchronizing language in PAs is PSPACE-complete.

5 Weakly Synchronizing Condition

First sight. In this chapter, we establish the complexity and memory requirement for weakly synchronizing conditions. We show that the membership problem for MDPs is PSPACE-complete for sure and almost-sure winning, that exponential memory is necessary and sufficient for sure winning while infinite memory is necessary for almost-sure winning, and we show that limit-sure and almost-sure winning coincide.

Moreover, we show that while the emptiness problem for sure weakly synchronizing languages in PAs is PSPACE-complete, the emptiness decision problems for almost-sure and limit-sure languages are undecidable. The results presented in this chapter are summarized in Table 5.1 and Table 5.2 on pages 90 and 103, respectively.

Contents

	Weakly	
	Complexity	**Memory requirement**
Sure	PSPACE-complete	exponential
Almost-sure	PSPACE-complete	infinite
Limit-sure	PSPACE-complete	infinite

Table 5.1: Computational complexity of the membership problem of weak synchronization in MDPs, and memory requirement for the winning strategies.

5.1 Weak synchronization in MDPs

By Remarks 5 and 6 (on pages 50 and 50), we consider the membership problem according to function *sum* and from Dirac initial distributions (i.e., single initial state q_{init}). The results presented in this section are summarized in Table 5.1.

5.1.1 Sure weak synchronization

In this subsection, we study the membership problem for sure weakly synchronizing condition and show that it is PSPACE-complete . The PSPACE upper bound of the membership problem for sure weak synchronization is obtained by the following characterization.

Lemma 5.1. *Let $\mathcal{M}$ be an MDP and T be a target set. For all states q_{init}, we have $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{sure}(sum_T)$ if and only if there exists a set $S \subseteq T$ such that $q_{\mathsf{init}} \in \mathsf{Pre}^m(S)$ for some $m \geq 0$ and $S \subseteq \mathsf{Pre}^n(S)$ for some $n \geq 1$.*

Proof. We recall that for all initial distribution X_0 with support S, we have $X_0 \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_T)$ if and only if there exists $k \geq 0$ such that $S \subseteq \mathsf{Pre}^k_{\mathcal{M}}(T)$ (See Lemma 4.2).

First, if $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{sure}(sum_T)$, then let α be a sure winning weakly synchronizing strategy. Then there are infinitely many positions n such that $\mathcal{M}^\alpha_n(T) = 1$, and since the state space is finite, there is a set S of states such that for infinitely many positions n we have $\mathsf{Supp}(\mathcal{M}^\alpha_n) = S$ and $\mathcal{M}^\alpha_n(T) = 1$, and thus $S \subseteq T$. By Lemma 4.2, it follows that $q_{\mathsf{init}} \in \mathsf{Pre}^m(S)$ for some $m \geq 0$, and by considering two positions $n_1 < n_2$ where $\mathsf{Supp}(\mathcal{M}^\alpha_{n_1}) = \mathsf{Supp}(\mathcal{M}^\alpha_{n_2}) = S$, it follows that $S \subseteq \mathsf{Pre}^n(S)$ for $n = n_2 - n_1 \geq 1$.

The reverse direction is straightforward by considering a strategy α ensuring that $\mathcal{M}^\alpha_m(S) = 1$ for some $m \geq 0$, and then ensuring that the probability mass from all states in S remains in S after every multiple of n steps where $n > 0$ is such that $S \subseteq \mathsf{Pre}^n(S)$, showing that α is a sure winning weakly synchronizing strategy in S (and thus in T) from q_{init}, thus $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{sure}(sum_T)$.

□

The PSPACE upper bound follows from the characterization in Lemma 5.1. A (N)PSPACE algorithm is to guess the set $S \subseteq T$, and the numbers m, n (with $m, n \leq$

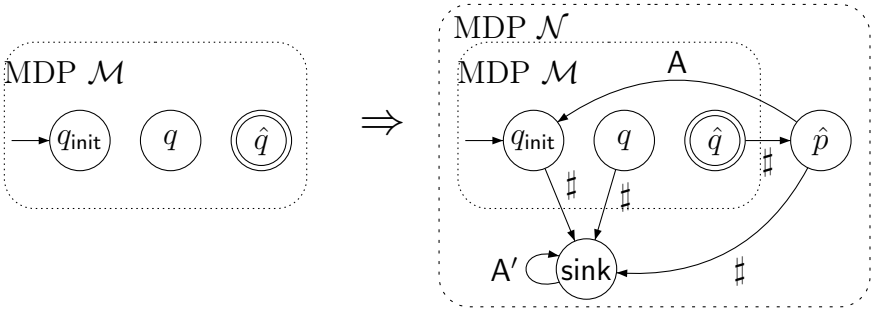

Figure 5.1: The reduction sketch to show PSPACE-hardness of the membership problem for sure weak synchronization in MDPs.

$2^{|Q|}$ since the sequence $\mathsf{Pre}^n(S)$ of predecessors is ultimately periodic), and check that $q_{\mathsf{init}} \in \mathsf{Pre}^m(S)$ and $S \subseteq \mathsf{Pre}^n(S)$. The PSPACE lower bound follows from the PSPACE-completeness of the membership problem for sure eventually synchronization. Lemma 5.2 gives the PSPACE lower bound.

Lemma 5.2. *The membership problem for* $\langle\langle 1 \rangle\rangle^{weakly}_{sure}(sum_T)$ *is* PSPACE-hard *even if* T *is a singleton.*

Proof. The proof is by a reduction from the membership problem for $\langle\langle 1 \rangle\rangle^{event}_{sure}(sum_T)$ with a singleton T, which is PSPACE-complete (See Theorem 4.3). From an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ with initial state q_{init} and target state $\hat{q}$, we construct another MDP $\mathcal{N} = \langle Q', \mathsf{A}', \delta' \rangle$ and a target state $\hat{p}$ such that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(\hat{q})$ in $\mathcal{M}$ if and only if $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{sure}(\hat{p})$ in $\mathcal{N}$.

The MDP $\mathcal{N}$ is a copy of $\mathcal{M}$ with two new states $\hat{p}$ and sink that are reachable only by a new action $\sharp$ (see Figure 5.1). Formally, $Q' = Q \cup \{\hat{p}, \mathsf{sink}\}$ and $\mathsf{A}' = \mathsf{A} \cup \{\sharp\}$. The transition function δ' is defined as follows: $\delta'(q, a) = \delta(q, a)$ for all states $q \in Q$ and $a \in \mathsf{A}$, $\delta(q, \sharp)(\mathsf{sink}) = 1$ for all $q \in Q' \setminus \{\hat{q}\}$ and $\delta(\hat{q}, \sharp)(\hat{p}) = 1$. The state sink is absorbing and from state $\hat{p}$ all other transitions lead to the initial state, i.e., $\delta(\mathsf{sink}, a)(\mathsf{sink}) = 1$ and $\delta(\hat{p}, a)(q_{\mathsf{init}}) = 1$ for all $a \in \mathsf{A}$.

We establish the correctness of the reduction as follows. First, if $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(\hat{q})$ in $\mathcal{M}$, then let α be a sure winning strategy in $\mathcal{M}$ for eventually synchronization in $\{\hat{q}\}$. A sure winning strategy in $\mathcal{N}$ for weak synchronization in $\{\hat{p}\}$ is to play according to α until the whole probability mass is in $\hat{q}$, then play $\sharp$ followed by some $a \in \mathsf{A}$ to visit $\hat{p}$ and get back to the initial state q_{init}, and then repeat the same strategy from q_{init}. Hence $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{sure}(\hat{p})$ in $\mathcal{N}$.

Second, if $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{sure}(\hat{p})$ in $\mathcal{N}$, then consider a strategy α such that $\mathcal{N}^{\alpha}_{n}(\hat{p}) = 1$ for some $n \geq 0$. By construction of $\mathcal{N}$, it follows that $\mathcal{N}^{\alpha}_{n-1}(\hat{q}) = 1$, that is all path-outcomes of α of length $n-1$ reach $\hat{q}$, and α plays $\sharp$ in the next step. If α never plays $\sharp$ before position $n-1$, then α is a valid strategy in $\mathcal{M}$ up to step $n-1$ and it shows that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(\hat{q})$ is sure winning in $\mathcal{M}$ for eventually synchronization in $\{\hat{q}\}$. Otherwise let m be the largest number such that there is a finite path-outcome ρ of α of length $m < n-1$ such that

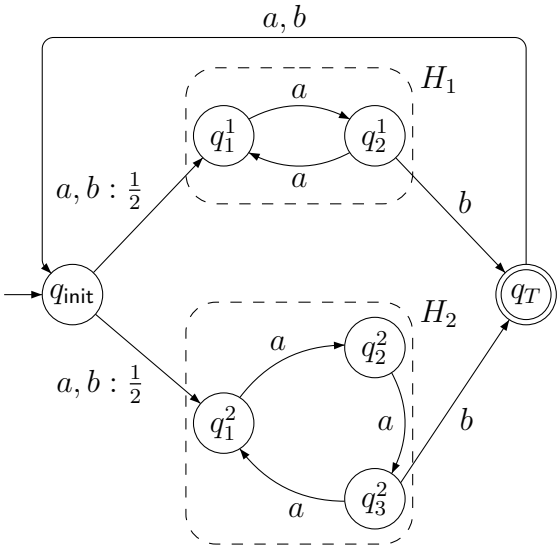

Figure 5.2: The MDP $\mathcal{M}_2$.

$\sharp \in \mathsf{Supp}(\alpha(\rho))$. Note that the action $\sharp$ can be played by α only in the state $\hat{q}$, and thus the initial state is reached again after one more step. It follows that in some path-outcome ρ' of α of length $m+2$, we have $\mathsf{Last}(\rho') = q_{\mathsf{init}}$, and by the choice of m, the action $\sharp$ is not played by α until position $n-1$ where all the probability mass is in $\hat{q}$. Hence the strategy that plays like α from ρ' in $\mathcal{N}$ is a valid strategy from q_{init} in $\mathcal{M}$, and is a witness that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(\hat{q})$.

□

The proof of Lemma 5.1 suggests an exponential-memory strategy for sure weakly synchronization that in $q \in \mathsf{Pre}^n(S)$ plays an action a such that $\mathsf{post}(q,a) \subseteq \mathsf{Pre}^{n-1}(S)$, which can be realized with exponential memory since $n \leq 2^{|Q|}$. It can be shown that exponential memory is necessary in general. The argument is very similar to the proof of exponential memory lower bound for sure eventually synchronization (See Section 4.2). For the sake of completeness, we present a family of MDPs $\mathcal{M}_n$ ($n \in \mathbb{N}$) over alphabet $\{a, b\}$ that are sure winning for weak synchronization, and where the sure winning strategies require exponential memory. The MDP $\mathcal{M}_2$ is shown in Figure 5.2. The structure of $\mathcal{M}_n$ is an initial uniform probabilistic transition to n components $H_1, \ldots, H_n$ where H_i is a cycle of length p_i the i-th prime number. On action a, the next state in the cycle is reached, and on action b the target state q_T is reached, only from the last state in the cycles. From other states, the action b leads to an absorbing sink state (transitions not depicted). A sure winning strategy from q_{init} for weak synchronization in $\{q_T\}$ is to play a in the first $p_n^{\#} = \prod_{i=1}^{n} p_i$ steps, and then play bb to reach q_{init} again, through q_T. This requires memory of size $p_n^{\#} > 2^n$ while the size of $\mathcal{M}_n$ is in $O(n^2 \log n)$ [BS96]. It can be proved that all winning strategies for weak synchronization need to be, from q_{init}, sure

eventually synchronizing in $\{q_T\}$ (consider the last occurrence of q_{init} along a play before all the probability mass is in q_T) and this requires memory of size at least $p_n^{\#}$ by standard pumping arguments as in the sure eventually case.

Theorem 5.1. *For sure weak synchronization in MDPs and for all functions $f \in \{sum_T, max_T\}$:*

1. *(Complexity). The membership problem is* PSPACE-complete.
2. *(Memory). Exponential memory is necessary and sufficient for both pure and randomized strategies, and pure strategies are sufficient.*

5.1.2 Almost-sure weak synchronization

Let $x = \sup_{n\in\mathbb{N}} x_n$ for a bounded sequence $(x_n)_{n\in\mathbb{N}}$; then x is equal or greater than all x_i and either there exists a certain n such that $x_n = x$ or $x = \limsup_{n\to\infty} x_n$. Since all entries in a probability distribution are bounded by 1, we can see that for an MDP $\mathcal{M}$ and a target set T,

– $\langle\langle 1\rangle\rangle^{event}_{almost}(f) = \langle\langle 1\rangle\rangle^{event}_{sure}(f) \cup \langle\langle 1\rangle\rangle^{weak}_{almost}(f)$

where $f \in \{max_T, sum_T\}$. Despite this mathematical connection, in the sequel, we give the intuitive characterization to decide the membership problem of almost-sure weak synchronization.

We present a characterization of almost-sure weak synchronization that gives a PSPACE upper bound for the membership problem. Our characterization uses the limit-sure eventually synchronizing conditions *with exact support*. Recall that this condition requires that the probability mass tends to 1 in a target set T, and moreover that after the same number of steps the support of the probability distribution is contained in a given set U. Formally, given an MDP $\mathcal{M}$, let $\langle\langle 1\rangle\rangle^{event}_{limit}(sum_T, U)$ for $T \subseteq U$ be the set of all initial distributions such that for all $\epsilon > 0$ there exists a strategy α and $n \in \mathbb{N}$ such that $\mathcal{M}^{\alpha}_n(T) \geq 1 - \epsilon$ and $\mathcal{M}^{\alpha}_n(U) = 1$.

We show that an MDP is almost-sure weakly synchronizing in target T if (and only if), for some set U, there is a sure eventually synchronizing strategy in target U, and from the probability distributions with support U there is a limit-sure winning strategy for eventually synchronizing in $\mathsf{Pre}(T)$ with support in $\mathsf{Pre}(U)$. This ensures that from the initial state we can have the whole probability mass in U, and from U have probability $1 - \epsilon$ in $\mathsf{Pre}(T)$ (and in T in the next step), while the whole probability mass is back in $\mathsf{Pre}(U)$ (and in U in the next step), allowing to repeat the strategy for $\epsilon \to 0$, thus ensuring infinitely often probability at least $1 - \epsilon$ in T (for all $\epsilon > 0$).

Lemma 5.3. *Let $\mathcal{M}$ be an MDP and T be a target set. For all states q_{init}, we have $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{weakly}_{almost}(sum_T)$ if and only if there exists a set U such that*

– *$q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{event}_{sure}(sum_U)$, and*

– $X_U \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_{\mathsf{Pre}(T)}, \mathsf{Pre}(U))$ *where* X_U *is the uniform distribution over* U.

Proof. First, if $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{almost}(sum_T)$, then there exists a strategy α such that for all $i \geq 0$ there exists $n_i \in \mathbb{N}$ such that $\mathcal{M}^{\alpha}_{n_i}(T) \geq 1 - 2^{-i}$, and moreover $n_{i+1} > n_i$ for all $i \geq 0$. Let $s_i = \mathsf{Supp}(\mathcal{M}^{\alpha}_{n_i})$ be the support of $\mathcal{M}^{\alpha}_{n_i}$. Since the state space is finite, there is a set U that occurs infinitely often in the sequence $s_0 s_1 \ldots$, thus for all $k > 0$ there exists $m_k \in \mathbb{N}$ such that $\mathcal{M}^{\alpha}_{m_k}(T) \geq 1 - 2^{-k}$ and $\mathcal{M}^{\alpha}_{m_k}(U) = 1$. It follows that α is sure eventually synchronizing in U from q_{init}, i.e., $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_U)$. Moreover, we can assume that $m_{k+1} > m_k$ for all $k > 0$ and thus $\mathcal{M}$ is also limit-sure eventually synchronizing in $\mathsf{Pre}(T)$ with exact support in $\mathsf{Pre}(U)$ from the initial distribution $X_1 = \mathcal{M}^{\alpha}_{m_1}$. By Corollary 4.3, since $\mathsf{Supp}(X_1) = U = \mathsf{Supp}(X_U)$ and since only the support of the initial probability distributions is relevant for the limit-sure eventually synchronizing condition, it follows that $X_U \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_{\mathsf{Pre}(T)}, \mathsf{Pre}(U))$.

To establish the converse, note that since $X_U \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_{\mathsf{Pre}(T)}, \mathsf{Pre}(U))$, it follows from Corollary 4.3 that from all initial distributions with support in U, for all $\epsilon > 0$ there exists a strategy α_ϵ and a position n_ϵ such that $\mathcal{M}^{\alpha_\epsilon}_{n_\epsilon}(T) \geq 1 - \epsilon$ and $\mathcal{M}^{\alpha_\epsilon}_{n_\epsilon}(U) = 1$. We construct an almost-sure weakly synchronizing strategy α as follows. Since $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_U)$, play according to a sure eventually synchronizing strategy from q_{init} until all the probability mass is in U. Then for $i = 1, 2, \ldots$ and $\epsilon_i = 2^{-i}$, repeat the following procedure: given the current probability distribution, select the corresponding strategy α_{ϵ_i} and play according to α_{ϵ_i} for n_{ϵ_i} steps, ensuring probability mass at least $1 - 2^{-i}$ in $\mathsf{Pre}(T)$ and support of the probability mass in $\mathsf{Pre}(U)$. Then from states in $\mathsf{Pre}(T)$, play an action to ensure reaching T in the next step, and from states in $\mathsf{Pre}(U)$ ensure reaching U. Continue playing according to $\alpha_{\epsilon_{i+1}}$ for $n_{\epsilon_{i+1}}$ steps, etc. Since $n_{\epsilon_i} + 1 > 0$ for all $i \geq 0$, this strategy ensures that $\limsup_{n\to\infty} \mathcal{M}^{\alpha}_{n}(T) = 1$ from q_{init}, hence $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weak}_{almost}(sum_T)$.

□

Note that from Lemma 5.3, it follows that counting strategies are sufficient to win almost-sure weakly synchronizing condition (a strategy is *counting* if $\alpha(\rho) = \alpha(\rho')$ for all prefixes ρ, ρ' with the same length and $\mathsf{Last}(\rho) = \mathsf{Last}(\rho')$).

Since the membership problems for sure eventually synchronizing and for limit-sure eventually synchronizing with exact support are PSPACE-complete (See Theorem 4.4 and Theorem 4.5), the membership problem for almost-sure weak synchronization is in PSPACE by guessing the set U, and checking that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_U)$, and checking that $X_U \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_{\mathsf{Pre}(T)}, \mathsf{Pre}(U))$. We establish a matching PSPACE lower bound.

Lemma 5.4. *The membership problem for* $\langle\langle 1 \rangle\rangle^{weakly}_{almost}(sum_T)$ *is* PSPACE-hard *even if* T *is a singleton.*

Proof. The proof is by the same reduction from the universal finiteness problem as presented in Lemma 4.9 where from an MDP $\mathcal{M}$ and a singleton T, we constructed $\mathcal{N}$ and q_{init}, and showed that $\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$ if and only if q_{init} is limit-sure eventually synchronizing in T (with support trivially in the state space Q' of $\mathcal{N}$). We show that

$\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$ if and only if q_{init} is almost-sure weakly synchronizing in T, proving that the membership problem for almost-sure weakly synchronization is PSPACE-hard.

First, if q_{init} is almost-sure weakly synchronizing in T, then q_{init} is also limit-sure eventually synchronizing in T, and therefore $\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$ by the arguments in the proof of Lemma 4.9.

Second, if $\mathsf{Pre}^n_{\mathcal{M}}(T) \neq \emptyset$ for all $n \geq 0$, then q_{init} is limit-sure eventually synchronizing in T (with support in the state space $Q' = Q \cup \{q_{\mathsf{init}}\}$ of $\mathcal{N}$), and since the state q_{init} is reached from all states in Q by playing the action $\sharp$ and from q_{init} by playing any action in A, it follows that the uniform distribution over Q' is also limit-sure eventually synchronizing in T (with support in Q') by first going to q_{init} and then playing a limit-sure eventually synchronizing strategy. By the characterization of Lemma 5.3 and since q_{init} is trivially sure eventually synchronizing in Q', it follows that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{almost}(sum_T)$.

□

We provide an example of an MDP such that all strategies to win almost-sure weakly synchronizing conditions require infinite memory. The example and argument are analogous to the proof that infinite memory is necessary for almost-sure eventually synchronizing, see Lemma 4.3.

Lemma 5.5. *There exists an almost-sure weakly synchronizing MDP for which all almost-sure weakly synchronizing strategies require infinite memory.*

Proof. Consider the MDP $\mathcal{M}$ shown in Figure 4.4 with three states $q_{\mathsf{init}}, q_1, q_2$ and two actions a, b. The only probabilistic transition is in q_{init} on action a that has successors q_{init} and q_1 with probability $\frac{1}{2}$. The other transitions are deterministic. Let q_{init} be the initial state. We construct a strategy that is almost-sure weakly synchronizing in $\{q_2\}$, showing that $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{almost}(q_2)$. First, observe that for all $\epsilon > 0$ we can have probability at least $1 - \epsilon$ in q_2 after finitely many steps from q_{init}: playing n times a and then b leads to probability $1 - \frac{1}{2^n}$ in q_2. Note that after that, the current probability distribution has support $\{q_{\mathsf{init}}, q_2\}$ and that from such a distribution, we can as well ensure probability at least $1 - \epsilon$ in q_2. Thus for a fixed ϵ, the MDP is $(1 - \epsilon)$-synchronizing in $\{q_2\}$ (after finitely many steps), and by taking a smaller value of ϵ, we can continue to play a strategy to have probability at least $1 - \epsilon$ in q_2, and repeat this for $\epsilon \to 0$. This strategy ensures almost-sure weak synchronization in $\{q_2\}$. Below, we show that infinite memory is necessary for almost-sure winning in this MDP.

Assume towards contradiction that there exists a finite-memory strategy α that is almost-sure weakly synchronizing in $\{q_2\}$. Consider the Markov chain $\mathcal{M}(\alpha)$ (the product of the MDP $\mathcal{M}$ with the finite-state transducer defining α). A state (q, m) in $\mathcal{M}(\alpha)$ is called a *q-state*. Since α is almost-sure weakly synchronizing in $\{q_2\}$, there is a q_2-state in the recurrent states of $\mathcal{M}(\alpha)$. Since on all actions q_{init} is a successor of q_2, and q_{init} is a successor of itself, it follows that there is a recurrent q_{init}-state in $\mathcal{M}(\alpha)$, and that all periodic supports of recurrent states in $\mathcal{M}(\alpha)$ contain a q_{init}-state. Hence, in each

stationary distribution there is a q_{init}-state with a positive probability, and therefore the probability mass in q_{init} is bounded away from zero. It follows that the probability mass in q_2 is bounded away from 1 thus α is not almost-sure weakly synchronizing in $\{q_2\}$, a contradiction.

□

From previous lemmas, we obtain Theorem 5.2 stating that the membership problem for almost-sure weakly synchronizing, is PSPACE-complete.

Theorem 5.2. *For almost-sure weak synchronization in MDPs and all functions $f \in \{max_T, sum_T\}$:*

1. *(Complexity). The membership problem is PSPACE-complete.*
2. *(Memory). Infinite memory is necessary in general for both pure and randomized strategies, and pure strategies are sufficient.*

5.1.3 Limit-sure weak synchronization

We show that the winning regions for almost-sure and limit-sure weak synchronization coincide. The result is not intuitively obvious (recall that it does not hold for eventually synchronizing) and requires a careful analysis of the structure of limit-sure winning strategies to show that they always induce the existence of an almost-sure winning strategy. The construction of an almost-sure winning strategy from a family of limit-sure winning strategies is illustrated in the following example.

Example 5.1. *Consider the MDP $\mathcal{M}$ in Figure 5.3 with initial state q_{init} and target set $T = \{q_4\}$. Note that there is a relevant strategic choice only in q_3, and that q_{init} is limit-sure winning for eventually synchronization in $\{q_4\}$ since we can inject a probability mass arbitrarily close to 1 in q_3 (by always playing a in q_3), and then switching to playing b in q_3 gets probability $1-\epsilon$ in T (for arbitrarily small ϵ). Moreover, the same holds from state q_4. These two facts are sufficient to show that q_{init} is limit-sure winning for weak synchronization in $\{q_4\}$: given $\epsilon > 0$, play from q_{init} a strategy to ensure probability at least $p_1 = 1 - \frac{\epsilon}{2}$ in q_4 (in finitely many steps), and then play according to a strategy that ensures from q_4 probability $p_2 = p_1 - \frac{\epsilon}{4}$ in q_4 (in finitely many, and at least one step), and repeat this process using strategies that ensure, if the probability mass in q_4 is at least p_i, that the probability in q_4 is at least $p_{i+1} = p_i - \frac{\epsilon}{2^{i+1}}$ (in at least one step). It follows that $p_i = 1 - \frac{\epsilon}{2} - \frac{\epsilon}{4} - \cdots - \frac{\epsilon}{2^i} > 1 - \epsilon$ for all $i \geq 1$, and thus $\limsup_{i\to\infty} p_i \geq 1 - \epsilon$ showing that q_{init} is limit-sure weakly synchronizing in target $\{q_4\}$.*

It follows from the result that we establish in this section (Theorem 5.3) that q_{init} is actually almost-sure weakly synchronizing in target $\{q_4\}$. To see this, consider the sequence $\mathsf{Pre}^i(T)$ for $i \geq 0$: $\{q_4\}, \{q_3\}, \{q_2\}, \{q_3\}, \ldots$ is ultimately periodic with period $r = 2$ and

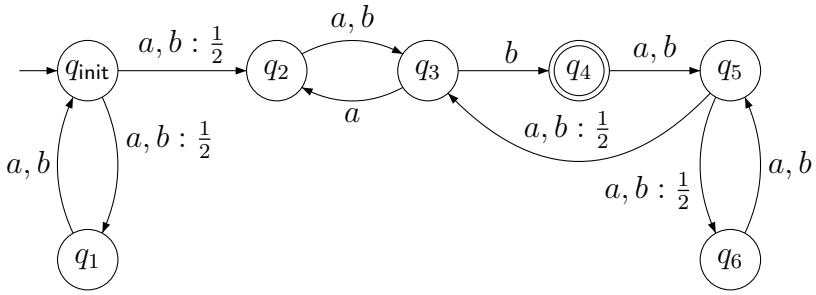

Figure 5.3: An example to show $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{limit}(q_4)$ implies $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{weakly}_{almost}(q_4)$.

$R = \{q_3\} = \mathsf{Pre}(T)$ *is such that* $R = \mathsf{Pre}^2(R)$*. The period corresponds to the loop* q_2q_3 *in the MDP. It turns out that* limit-sure *eventually synchronizing in* T *implies* almost-sure *eventually synchronizing in* R*, see Corollary 4.2, thus from* q_{init} *a* single *strategy ensures that the probability mass in* R *is* 1*, either in the limit or after finitely many steps. Note that in both cases since* $R = \mathsf{Pre}^r(R)$ *this even implies almost-sure weakly synchronizing in* R*. The same holds from state* q_4*.*

Moreover, note that all distributions produced by an almost-sure weakly synchronizing strategy are themselves almost-sure weakly synchronizing. An almost-sure winning strategy for weak synchronization in $\{q_4\}$ *consists in playing from* q_{init} *an* almost-sure *eventually synchronizing strategy in target* $R = \{q_3\}$*, and considering a decreasing sequence* ϵ_i *such that* $\lim_{i\to\infty} \epsilon_i = 0$*, when the probability mass in* R *is at least* $1 - \epsilon_i$*, inject it in* $T = \{q_4\}$*. Then the remaining probability mass defines a distribution (with support* $\{q_1, q_2\}$ *in the example) that is still almost-sure eventually synchronizing in* R*, as well as the states in* T*. Note that in the example, (almost all) the probability mass in* $T = \{q_4\}$ *can move to* q_3 *in an even number of steps, while from* $\{q_1, q_2\}$ *an odd number of steps is required, resulting in a* shift *of the probability mass. However, by repeating the strategy two times from* q_4 *(injecting large probability mass in* q_3*, moving to* q_4*, and injecting in* q_3 *again), we can make up for the shift and reach* q_3 *from* q_4 *in an even number of steps, thus in synchronization with the probability mass from* $\{q_1, q_2\}$*.*

$\triangleleft$

This idea behind Example 5.1, which is constructing an almost-sure winning strategy from a family of limit-sure winning strategies by annihilating the shifts at proper times, is formalized in the rest of this subsection, and we prove that we can always make up for the shifts, which requires a carefully analysis of the allowed amounts of shifting.

The result is easier to prove when the target T is a singleton, as in Example 5.1. For an arbitrary target set T, we need to get rid of the states in T that do not contribute a significant (i.e., bounded away from 0) probability mass in the limit, that we call the *vanishing states*. We show that they can be removed from T without changing the winning region for limit-sure winning. When the target set has no vanishing state, we can construct an almost-sure winning strategy as in the case of a singleton target set.

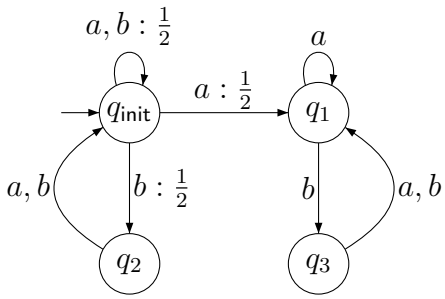

Figure 5.4: The state q_2 is a vanishing state for the target set $T = \{q_2, q_3\}$ and the family of stagiest $(\alpha)_{i\in\mathbb{N}}$ described in Example 5.2.

Definition 5.1. *Given an MDP $\mathcal{M}$ with initial state $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{weakly}_{limit}(sum_T)$ that is limit-sure winning for the weakly synchronizing condition in target set T, let $(\alpha_i)_{i\in\mathbb{N}}$ be a family of limit-sure winning strategies such that $\limsup_{n\to\infty} \mathcal{M}^{\alpha_i}_n(T) \geq 1 - \epsilon_i$ where $\lim_{i\to\infty} \epsilon_i = 0$. Hence by definition of $\limsup$, for all $i \geq 0$ there exists a strictly increasing sequence $k_{i,0} < k_{i,1} < \cdots$ of positions such that $\mathcal{M}^{\alpha_i}_{k_{i,j}}(T) \geq 1 - 2\epsilon_i$ for all $j \geq 0$. A state $q \in T$ is* vanishing *if $\liminf_{i\to\infty} \liminf_{j\to\infty} \mathcal{M}^{\alpha_i}_{k_{i,j}}(q) = 0$ for some family of limit-sure weakly synchronizing strategies $(\alpha_i)_{i\in\mathbb{N}}$.*

Intuitively, the contribution of a vanishing state q to the probability in T tends to 0 and therefore $\mathcal{M}$ is also limit-sure winning for the weakly synchronizing condition in target set $T \setminus \{q\}$.

Example 5.2. *Consider the MDP in Figure 5.4 with four states and two actions a, b. All transitions are deterministic except in q_{init}: the a-transitions have two successors q_{init} and q_1 each with probability $\frac{1}{2}$, as well as the b-transitions with two successors q_{init} and q_2. The a-transition in q_1 is a self-loop whereas the b-transition deterministically goes to q_3. All transitions in q_2 are directed to q_{init}, and all in q_3 are directed to q_1. Let $T = \{q_2, q_3\}$ be the target set and for all $i \in \mathbb{N}$, let α_i be the strategy that repeats the following template forever: playing i times a and then playing two times b. The family of strategies $(\alpha_i)_{i\in\mathbb{N}}$ is a witness to show that $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{weakly}_{limit}(sum_T)$ where the state q_2 is a vanishing state. We see that the contribution of q_2 in accumulating the probability mass in $\{q_2, q_3\}$ tends to 0 when $i \to \infty$. Thus, $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{weakly}_{limit}(q_3)$ too.*

◁

Lemma 5.6. *If an MDP $\mathcal{M}$ is limit-sure weakly synchronizing in target set T, then there exists a set $T' \subseteq T$ such that $\mathcal{M}$ is limit-sure weakly synchronizing in T' without vanishing states.*

Proof. If there is no vanishing state for $(\alpha_i)_{i\in\mathbb{N}}$, then take $T' = T$ and the proof is complete. Otherwise, let $(\alpha_i)_{i\in\mathbb{N}}$ be a family of limit-sure winning strategies such that

$\limsup_{n\to\infty} \mathcal{M}_n^{\alpha_i}(T) \geq 1-\epsilon_i$ where $\lim_{i\to\infty} \epsilon_i = 0$ and let q be a vanishing state for $(\alpha_i)_{i\in\mathbb{N}}$. We show that $(\alpha_i)_{i\in\mathbb{N}}$ is limit-sure weakly synchronizing in $T \setminus \{q\}$. For every $i \geq 0$ let $k_{i,0} < k_{i,1} < \cdots$ be a strictly increasing sequence such that (a) $\mathcal{M}_{k_{i,j}}^{\alpha_i}(T) \geq 1-2\epsilon_i$ for all $i, j \geq 0$, and (b) $\liminf_{i\to\infty} \liminf_{j\to\infty} \mathcal{M}_{k_{i,j}}^{\alpha_i}(q) = 0$.

It follows from (b) that for all $\epsilon > 0$ and all $x > 0$ there exists $i > x$ such that for all $y > 0$ there exists $j > y$ such that $\mathcal{M}_{k_{i,j}}^{\alpha_i}(q) < \epsilon$, and thus

$$\mathcal{M}_{k_{i,j}}^{\alpha_i}(T \setminus \{q\}) \geq 1 - 2\epsilon_i - \epsilon$$

by (a). Since this holds for infinitely many i's, we can choose i such that $\epsilon_i < \epsilon$ and we have

$$\limsup_{j\to\infty} \mathcal{M}_{k_{i,j}}^{\alpha_i}(T \setminus \{q\}) \geq 1 - 3\epsilon$$

and thus

$$\limsup_{n\to\infty} \mathcal{M}_n^{\alpha_i}(T \setminus \{q\}) \geq 1 - 3\epsilon$$

since the sequence $(k_{i,j})_{j\in\mathbb{N}}$ is strictly increasing. This shows that $(\alpha_i)_{i\in\mathbb{N}}$ is limit-sure weakly synchronizing in $T \setminus \{q\}$.

By repeating this argument as long as there is a vanishing state (thus at most $|T|-1$ times), we can construct the desired set $T' \subseteq T$ without vanishing state.

□

For a limit-sure weakly synchronizing MDP in target set T (without vanishing states), we show that from a probability distribution with support T, a probability mass arbitrarily close to 1 can be injected synchronously back in T (in at least one step), that is $X_T \in \langle\langle 1\rangle\rangle_{limit}^{event}(sum_{\mathsf{Pre}(T)})$. The same holds from the initial state q_{init} of the MDP. This property is the key to construct an almost-sure weakly synchronizing strategy.

Lemma 5.7. *If an MDP $\mathcal{M}$ with initial state q_{init} is limit-sure weakly synchronizing in a target set T without vanishing states, then $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle_{limit}^{event}(sum_{\mathsf{Pre}(T)})$ and $X_T \in \langle\langle 1\rangle\rangle_{limit}^{event}(sum_{\mathsf{Pre}(T)})$ where X_T is the uniform distribution over T.*

Proof. Since $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle_{limit}^{weakly}(sum_T)$ and $\langle\langle 1\rangle\rangle_{limit}^{weakly}(sum_T) \subseteq \langle\langle 1\rangle\rangle_{limit}^{event}(sum_T)$, we have $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle_{limit}^{event}(sum_T)$ and thus it suffices to prove that $X_T \in \langle\langle 1\rangle\rangle_{limit}^{event}(sum_{\mathsf{Pre}(T)})$. This is because then from q_{init}, probability arbitrarily close to 1 can be injected in $\mathsf{Pre}(T)$ through a distribution with support in T (since by Corollary 4.3 only the support of the initial probability distribution is important for limit-sure eventually synchronizing).

Let $(\alpha_i)_{i\in\mathbb{N}}$ be a family of limit-sure winning strategies such that $\limsup_{n\to\infty} \mathcal{M}_n^{\alpha_i}(T) \geq 1-\epsilon_i$ where $\lim_{i\to\infty} \epsilon_i = 0$, and such that there is no vanishing state. For every $i \geq 0$ let $k_{i,0} < k_{i,1} < \cdots$ be a strictly increasing sequence such that $\mathcal{M}_{k_{i,j}}^{\alpha_i}(T) \geq 1-2\epsilon_i$ for all $i, j \geq 0$, and let $B = \min_{q\in T} \liminf_{i\to\infty} \liminf_{j\to\infty} \mathcal{M}_{k_{i,j}}^{\alpha_i}(q)$. Note that $B > 0$ since there is no vanishing state. It follows that there exists $x > 0$ such that for all $i > x$ there exists $y_i > 0$ such that for all $j > y_i$ and all $q \in T$ we have $\mathcal{M}_{k_{i,j}}^{\alpha_i}(q) \geq \frac{B}{2}$.

Given $\nu > 0$, let $i > x$ such that $\epsilon_i < \frac{\nu B}{4}$, and for $j > y_i$, consider the positions $n_1 = k_{i,j}$ and $n_2 = k_{i,j+1}$. We have $n_1 < n_2$ and $\mathcal{M}^{\alpha_i}_{n_1}(T) \geq 1 - 2\epsilon_i$ and $\mathcal{M}^{\alpha_i}_{n_2}(T) \geq 1 - 2\epsilon_i$, and $\mathcal{M}^{\alpha_i}_{n_1}(q) \geq \frac{B}{2}$ for all $q \in T$. Consider the strategy β that plays like α_i plays from position n_1 and thus transforms the distribution $\mathcal{M}^{\alpha_i}_{n_1}$ into $\mathcal{M}^{\alpha_i}_{n_2}$. For all states $q \in T$, from the Dirac distribution on q under strategy β, the probability to reach $Q \setminus T$ in $n_2 - n_1$ steps is thus at most

$$\frac{\mathcal{M}^{\alpha_i}_{n_2}(Q \setminus T)}{\mathcal{M}^{\alpha_i}_{n_1}(q)} \leq \frac{2\epsilon_i}{B/2} < \nu.$$

Therefore, from an arbitrary probability distribution with support T (over set T) we have $\mathcal{M}^{\beta}_{n_2-n_1}(T) > 1-\nu$, showing that X_T is limit-sure eventually synchronizing in T and thus in $\mathsf{Pre}(T)$ since $n_2 - n_1 > 0$ (it is easy to show that if the mass of probability in T is at least $1 - \nu$, then the mass of probability in $\mathsf{Pre}(T)$ one step before is at least $1 - \frac{\nu}{\eta}$ where η is the smallest positive probability in $\mathcal{M}$).

□

To show that limit-sure and almost-sure winning coincide for weakly synchronizing conditions, from a family of limit-sure winning strategies we construct an almost-sure winning strategy that uses the eventually synchronizing strategies of Lemma 5.7. The construction consists in using successively strategies that ensure probability mass $1 - \epsilon_i$ in the target T, for a decreasing sequence $\epsilon_i \to 0$. Such strategies exist by Lemma 5.7, both from the initial state and from the set T. However, the mass of probability that can be guaranteed to be synchronized in T by the successive strategies is always smaller than 1, and therefore we need to argue that the remaining masses of probability (of size ϵ_i) can also get synchronized in T, and despite their possible shift with the main mass of probability.

Two main key arguments are needed to establish the correctness of the construction: (1) eventually synchronizing implies that a finite number of steps is sufficient to obtain a probability mass of $1-\epsilon_i$ in T, and thus the construction of the strategy is well defined, and (2) by the finiteness of the period r (such that $R = \mathsf{Pre}^r(R)$ where $R = \mathsf{Pre}^k(T)$ for some k) we can ensure to eventually make up for the shifts, and every piece of the probability mass can contribute (synchronously) to the target infinitely often.

Theorem 5.3. *$\langle\langle 1\rangle\rangle^{weakly}_{limit}(sum_T) = \langle\langle 1\rangle\rangle^{weakly}_{almost}(sum_T)$ for all MDPs and target sets T.*

Proof. Since $\langle\langle 1\rangle\rangle^{weakly}_{almost}(sum_T) \subseteq \langle\langle 1\rangle\rangle^{weakly}_{limit}(sum_T)$ holds by the definition, it is sufficient to prove that $\langle\langle 1\rangle\rangle^{weakly}_{limit}(sum_T) \subseteq \langle\langle 1\rangle\rangle^{weakly}_{almost}(sum_T)$ and by Lemma 5.6 it is sufficient to prove that if $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{weakly}_{limit}(sum_T)$ is limit-sure weakly synchronizing in T without vanishing state, then q_{init} is almost-sure weakly synchronizing in T. If T has vanishing states, then consider $T' \subseteq T$ as in Lemma 5.6 and it will follows that q_{init} is almost-sure weakly synchronizing in T' and thus also in T. We proceed with the proof that $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{weakly}_{limit}(sum_T)$ implies $q_{\mathsf{init}} \in \langle\langle 1\rangle\rangle^{weakly}_{almost}(sum_T)$.

For $i = 1, 2, \ldots$ consider the sequence of predecessors $\mathsf{Pre}^i(T)$, which is ultimately periodic: let $1 \leq k, r \leq 2^{|Q|}$ such that $\mathsf{Pre}^k(T) = \mathsf{Pre}^{k+r}(T)$, and let $R = \mathsf{Pre}^k(T)$. Thus $R = \mathsf{Pre}^{k+r}(T) = \mathsf{Pre}^r(R)$.

Claim 1. *We have* $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(sum_R)$ *and* $X_T \in \langle\langle 1 \rangle\rangle^{event}_{almost}(sum_R)$.

Proof of Claim 1. By Lemma 5.7, since there is no vanishing state in T we have $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_{\mathsf{Pre}(T)})$ and $X_T \in \langle\langle 1 \rangle\rangle^{event}_{limit}(sum_{\mathsf{Pre}(T)})$, and it follows from the characterization of Lemma 4.6 and Corollary 4.2 that:

either (1) $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_{\mathsf{Pre}(T)})$ or (2) $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(sum_R)$, and
either (a) $X_T \in \langle\langle 1 \rangle\rangle^{event}_{sure}(sum_{\mathsf{Pre}(T)})$ or (b) $X_T \in \langle\langle 1 \rangle\rangle^{event}_{almost}(sum_R)$.

Note that (a) implies (b) (and thus (b) holds) since (a) implies $T \subseteq \mathsf{Pre}^i(T)$ for some $i \geq 1$ (by Lemma 4.2) and thus $T \subseteq \mathsf{Pre}^{n \cdot i}(T)$ for all $n \geq 0$ by monotonicity of $\mathsf{Pre}^i(\cdot)$, which entails for $n \cdot i \geq k$ that $T \subseteq \mathsf{Pre}^m(R)$ where $m = (n \cdot i - k) \mod r$ and thus X_T is sure (and almost-sure) winning for the eventually synchronizing condition in target R.

Note also that (1) implies (2) since by (1) we can play a sure-winning strategy from q_{init} to ensure in finitely many steps probability 1 in $\mathsf{Pre}(T)$ and in the next step probability 1 in T, and by (b) play an almost-sure winning strategy for eventually synchronizing in R. Hence $q_{\mathsf{init}} \in \langle\langle 1 \rangle\rangle^{event}_{almost}(sum_R)$ and thus (2b) holds, which concludes the proof of Claim 1.

We now show that there exists an almost-sure winning strategy for the weakly synchronizing condition in target T.

Recall that $\mathsf{Pre}^r(R) = R$ and thus once some probability mass p is in R, it is possible to ensure that the probability mass in R after r steps is at least p, and thus that (with period r) the probability in R does not decrease. By the result of Lemma 4.7, almost-sure winning for eventually synchronizing in R implies that there exists a strategy α such that the probability in R tends to 1 at periodic positions: for some $0 \leq h < r$ the strategy α is *almost-sure eventually synchronizing in R with shift h*, that is $\forall \epsilon > 0 \cdot \exists N \cdot \forall n \geq N : n \equiv h \mod r \implies \mathcal{M}^\alpha_n(R) \geq 1 - \epsilon$. We also say that the initial distribution $X_0 = \mathcal{M}^\alpha_0$ is almost-sure eventually synchronizing in R with shift h.

Claim 2.

- *($\star$) If $\mathcal{M}^\alpha_0$ is almost-sure eventually synchronizing in R with some shift h, then $\mathcal{M}^\alpha_i$ is almost-sure eventually synchronizing in R with shift $h - i \mod r$.*
- *($\star\star$) Let t such that X_T is almost-sure eventually synchronizing in R with shift t. If a distribution is almost-sure eventually synchronizing in R with some shift h, then it is also almost-sure eventually synchronizing in R with shift $h + k + t \mod r$ (where we chose k such that $R = \mathsf{Pre}^k(T)$).*

Proof of Claim 2. The result ($\star$) immediately follows from the definition of shift, and we prove ($\star\star$) as follows. We show that almost-sure eventually synchronizing in R with shift h implies almost-sure eventually synchronizing in R with shift $h + k + t \mod r$. Intuitively, the probability mass that is in R with shift h can be injected in T in k steps, and then from T we can play an almost-sure eventually synchronizing strategy in target R with shift t, thus a total shift of $h + k + t \mod r$. Precisely, an almost-sure winning strategy α is constructed as follows: given a finite prefix of play ρ, if there is no state $q \in R$ that occurs in ρ at a position $n \equiv h \mod r$, then play in ρ according to the almost-sure winning strategy α_h for eventually synchronizing in R with shift h. Otherwise, if there is no $q \in T$ that occurs in ρ at a position $n \equiv h + k \mod r$, then we play according to a sure winning strategy α_{sure} for eventually synchronizing in T, and otherwise we play according to an almost-sure winning strategy α_t from T for eventually synchronizing in R with shift t. To show that α is almost-sure eventually synchronizing in R with shift $h + k + t$, note that α_h ensures with probability 1 that R is reached at positions $n \equiv h \mod r$, and thus T is reached at positions $h + k \mod r$ by α_{sure}, and from the states in T the strategy α_t ensures with probability 1 that R is reached at positions $h + k + t \mod r$. This concludes the proof of Claim 2.

Construction of an almost-sure winning strategy. We construct strategies α_ϵ for $\epsilon > 0$ that ensure, from a distribution that is almost-sure eventually synchronizing in R (with some shift h), that after finitely many steps, a distribution d' is reached such that $d'(T) \geq 1 - \epsilon$ and d' is almost-sure eventually synchronizing in R (with some shift h'). Since q_{init} is almost-sure eventually synchronizing in R (with some shift h), it follows that the strategy α_{as} that plays successively the strategies (each for finitely many steps) $\alpha_{\frac{1}{2}}$, $\alpha_{\frac{1}{4}}$, $\alpha_{\frac{1}{8}}, \ldots$ is almost-sure winning from q_{init} for the weakly synchronizing condition in target T.

We define the strategies α_ϵ as follows. Given an initial distribution that is almost-sure eventually synchronizing in R with a shift h and given $\epsilon > 0$, let α_ϵ be the strategy that plays according to the almost-sure winning strategy α_h for eventually synchronizing in R with shift h for a number of steps $n \equiv h \mod r$ until a distribution d is reached such that $d(R) \geq 1 - \epsilon$, and then from d it plays according to a sure winning strategy α_{sure} for eventually synchronizing in T from the states in R (for k steps), and keeps playing according to α_h from the states in $Q \setminus R$ (for k steps). The distribution d' reached from d after k steps is such that $d'(T) \geq 1 - \epsilon$ and we claim that it is almost-sure eventually synchronizing in R with shift t. This holds by definition from the states in $\mathsf{Supp}(d') \cap T$, and by ($\star$) the states in $\mathsf{Supp}(d') \setminus T$ are almost-sure eventually synchronizing in R with shift $h - (h + k) \mod r$, and by ($\star\star$) with shift $h - (h + k) + k + t = t$.

It follows that the strategy α_{as} is well-defined and ensures, for all $\epsilon > 0$, that the probability mass in T is infinitely often at least $1 - \epsilon$, thus is almost-sure weakly synchronizing in T. This concludes the proof of Theorem 5.3.

□

From previous lemmas and Theorem 5.3, we obtain the following result.

	Weakly
Sure	PSPACE-complete
Almost-sure	undecidable
Limit-sure	undecidable

Table 5.2: Computational complexity of the emptiness problem of weakly synchronizing languages in PAs.

Theorem 5.4. *For limit-sure weak synchronization in MDPs and all functions* $f \in \{max_T, sum_T\}$*:*

1. *(Complexity). The membership problem is PSPACE-complete.*
2. *(Memory). Infinite memory is necessary in general for both pure and randomized strategies, and pure strategies are sufficient.*

5.2 Weak synchronization in PAs

Again by Remarks 5 and 6 (on pages 50 and 50), we consider the emptiness problem for weakly synchronizing languages in PAs only according to function *sum* and initialized from Dirac distributions. The results presented in this section are summarized in Table 5.2.

5.2.1 Sure weak synchronization

We show that the emptiness problem for sure weakly synchronizing languages in PAs, is PSPACE-complete. The PSPACE upper bound is an immediate result from the following intuitive Lemma.

Lemma 5.8. *Let* $\mathcal{P}$ *be a PA and* T *be a target set. For all initial state* q_{init}*, we have* $\mathcal{L}_{sure}^{weakly}(sum_T) \neq \emptyset$ *if and only if there are two words* $w, v \in \mathsf{A}^\omega$ *and a set* $S \subseteq T$ *such that*

- *w is sure eventually synchronizing in S from q_0,*
- *v is non-trivially sure eventually synchronizing in S from an arbitrary distribution with support S.*

Proof. First, if $\mathcal{L}_{sure}^{weakly}(sum_T) \neq \emptyset$ from q_{init}, then let w be a sure weakly synchronizing word. Then, there are infinitely many positions n such that $\mathcal{P}_n^w(T) = 1$, and since the state space is finite, there is a set S of states such that for infinitely many positions n we have $\mathsf{Supp}(\mathcal{P}_n^w) = S$ and $\mathcal{P}_n^w(T) = 1$, and thus $S \subseteq T$. It thus follows that w is sure eventually synchronizing in S from q_0 too. Moreover by considering two positions $n_1 < n_2$ where $\mathsf{Supp}(\mathcal{P}_{n_1}^w) = \mathsf{Supp}(\mathcal{P}_{n_2}^w) = S$, it follows that there is a non-empty word v' such

that all words $v \in v' \cdot \mathsf{A}^\omega$ are non-trivially sure eventually synchronizing in S from the distribution $\mathcal{P}^w_{n_1}$ with support S.

The reverse direction is straightforward. Let $w_n = a_0 \cdots a_{n-1}$ be the prefix of w, which consists of the first n-th letters, where n is such that $\mathcal{P}^w_n(S) = 1$ from q_0. Let $v_m = b_0 \cdots b_{m-1}$ be the non-empty prefix of v, which consists of the first m-th letters, where m is such that $\mathcal{P}^v_m(S) = 1$ from an arbitrary distribution with support S. By characterization presented in Lemma 4.10, we know that only the support of an initial distribution matters for sure eventually synchronizing languages in PAs, then $w_n \cdot (v_m)^\omega \in \mathcal{L}^{weak}_{sure}(sum_T)$ from q_0.

□

The PSPACE upper bound follows from the characterization in Lemma 5.8 and the fact that emptiness problem for sure eventually synchronizing languages in PAs, is PSPACE-complete (See Theorem 4.6). The PSPACE-hardness is by a reduction from *finite automata intersection* that is the same reduction to obtain hardness result for the emptiness problem of sure eventually synchronizing languages in PAs (See Lemma 4.11).

Lemma 5.9. *The emptiness problem for $\mathcal{L}^{weakly}_{sure}(sum_T)$ in PAs, is* PSPACE-hard *even if T is a singleton.*

Proof. The proof is by the same reduction from finite automata intersection problem that is presented in Lemma 4.11 where from n DFAs $\mathcal{N}_1, \mathcal{N}_2, \cdots, \mathcal{N}_n$ we constructed a PA $\mathcal{P}$, an initial state q_{init} and a target state q_T and showed that all n DFAs accept a common word if, and only if, $\mathcal{L}^{event}_{sure}(q_T) \neq \emptyset$ from q_{init}.

Now we show that, in the constructed PA $\mathcal{P}$, we have $\mathcal{L}^{event}_{sure}(q_T) = \mathcal{L}^{weakly}_{sure}(q_T)$ from q_{init}, proving that the emptiness problem for sure weakly synchronizing languages is PSPACE-hard.

First direction is straightforward since by definition, $\mathcal{L}^{weakly}_{sure}(q_T) \subseteq \mathcal{L}^{event}_{sure}(q_T)$ from all initial states.

Second, if $w \in \mathcal{L}^{event}_{sure}(q_T)$, then there is $i \in \mathbb{N}$ such that $\mathcal{P}^w_i(q_T) = 1$. Let $w_i = a_0 a_1 \cdots a_{i-1}$ be the prefix of w that consists of the first i-th letter. Since all transitions in q_T are self-loops, then all ω-words $w_i \cdot (\mathsf{A}')^\omega$ are sure weakly synchronizing from q_{init} (See Lemma 5.8) As a result, $w \in \mathcal{L}^{weakly}_{sure}(q_T)$ from q_{init} that completes the proof.

□

From the previous lemmas, we obtain the following Theorem.

Theorem 5.5. *For all functions $f \in \{sum_T, max_T\}$ the emptiness problem for sure weakly synchronizing languages in PAs is* PSPACE-complete.

5.2.2 Almost-sure weak synchronization

In this subsection, we provide undecidability result for the emptiness problem of almost-sure weakly synchronizing languages in PAs.

Theorem 5.6. *For all functions $f \in \{max_T, sum_T\}$, the emptiness problem of almost-sure weakly synchronizing language in PAs is undecidable.*

Proof. The proof is by the same reduction from value 1 problem in PAs that is presented in Theorem 4.7 where from a PA $\mathcal{A}$ equipped with an initial state q_{init} and a set $\mathcal{F}$ of accepting states, we constructed another PA $\mathcal{P}$ and a target state q_T; and we showed that the PA $\mathcal{A}$ has value 1 if and only if $\mathcal{L}^{event}_{almost}(q_T) \neq \emptyset$ from q_{init} in the PA $\mathcal{P}$.

Now we prove that the PA $\mathcal{A}$ has value 1 if and only if $\mathcal{L}^{weak}_{almost}(q_T) \neq \emptyset$ from q_{init} in the PA $\mathcal{P}$ too. First, assume that $\mathcal{A}$ has value 1. In the proof of Theorem 4.7, we have argued that the infinite word $v = w_{\frac{1}{2}} \cdot \# \cdot \# \cdot w_{\frac{1}{3}} \cdot \# \cdot \# \cdots w_{\frac{1}{n}} \cdot \# \cdot \# \cdots$ is almost-sure eventually synchronizing in $\{q_T\}$ from q_{init}. The same argument showing that for all $n \geq 2$, the mass of probability $1 - \frac{1}{n}$ is accumulated in q_T after reading the prefix of v (from beginning) up to the subword $w_{\frac{1}{n}} \cdot \sharp$, is valid here and implies that v is also almost-sure weakly synchronizing in q_T.

The reverse direction is straightforward since we know $\mathcal{L}^{weak}_{almost}(q_T) \subseteq \mathcal{L}^{event}_{almost}(q_T)$ from all initial states, and since we have proven that if $\mathcal{L}^{event}_{almost}(q_T) \neq \emptyset$ from q_{init}, then $\mathcal{A}$ has value 1.

□

5.2.3 Limit-sure weak synchronization

In this subsection, we provide undecidability result for the emptiness problem of limit-sure weakly synchronizing languages in PAs.

Theorem 5.7. *For all functions $f \in \{max_T, sum_T\}$, the emptiness problem of limit-sure weakly synchronizing language in PAs is undecidable.*

Proof. The proof is by the same reduction from value 1 problem in PAs that is presented in Theorem 4.7 and Theorem 5.6 where from a PA $\mathcal{A}$ equipped with an initial state q_{init} and a set $\mathcal{F}$ of accepting states, we constructed another PA $\mathcal{P}$ and a target state q_T; and we showed that the PA $\mathcal{A}$ has value 1 if and only if $\mathcal{L}^{event}_{almost}(q_T) \neq \emptyset$ from q_{init} if and only if $\mathcal{L}^{weak}_{almost}(q_T) \neq \emptyset$ from q_{init} in the PA $\mathcal{P}$.

Now we prove that the PA $\mathcal{A}$ has value 1 if and only if $\mathcal{L}^{weak}_{limit}(q_T) \neq \emptyset$ from q_{init} in the PA $\mathcal{P}$ too. First, assume that $\mathcal{A}$ has value 1. In the proof of Theorem 5.6, we have shown that there exists some almost-sure weakly synchronizing word $v \in \mathcal{L}^{weak}_{almost}(q_T)$ from q_{init}. By definition, since v is almost-sure weakly synchronizing, so is limit-sure weakly synchronizing. Thus, $\mathcal{L}^{weak}_{limit}(q_T) \neq \emptyset$ from q_{init}.

Second, assume that $\mathcal{L}^{weak}_{limit}(q_T) \neq \emptyset$ from q_{init}. Therefore, for all $\epsilon > 0$ there exists an infinite word w such that $\limsup_{n\to\infty} \mathcal{P}^w_n(q_T) \geq 1 - \epsilon$. For $\epsilon > 0$, let the word w and the number n be such that $\mathcal{P}^w_n(q_T) \geq 1 - 2\epsilon$. Since q_T is only reachable via #-transitions then the n-th letter of w must be #.

We pick the subword v of w as we have done in the proof of Theorem 4.7:

- if there is no occurrence of # before the $(n-1)$-th letter, let v be the subword from the beginning up to $(n-1)$-th letter.
- otherwise, if the last occurrence of # before the $(n-1)$-th letter happens as the m-th letter, check whether $\mathsf{post}(q_{\mathsf{init}}, w_m) = \{q_{\mathsf{init}}\}$:
  1. if yes, let v be the subword after m-th letter up to $(n-1)$-th letter
  2. otherwise, let v be the subword after $(m+1)$-th letter up to $(n-1)$-th letter

  where $w_m = a_0 \cdot a_1 \cdots a_m$ is the prefix of w consisting in the first $m+1$ letters.

By a similar argument (as one in the proof of Theorem 4.7), we have $\mathsf{Pr}(v) \geq 1 - 2\epsilon$ implying that there is a family of words whose accepting probability in $\mathcal{A}$ is arbitrarily close to 1. This means $\mathcal{A}$ has value 1 and the proof is complete.

□

5.3 Discussion

In this section, we briefly discuss the universality problems for sure and almost-sure weakly synchronizing languages in PAs.

Universality problems of sure weakly synchronizing language. Consider a PA $\mathcal{P}$ with the universal sure weakly synchronizing language according to max_T. By definition, for all words w the symbolic run $\mathcal{P}^w_0\mathcal{P}^w_1\mathcal{P}^w_2\cdots$ is infinitely often 1-synchronized in a singleton $\{q\}$ where $q \in T$. We claim that there exist $n, m \leq 2^{|Q|}$ such that for all words w, and all $i \in \mathbb{N}$ we have $\|\mathcal{P}^w_k\| = 1$ where $k = n + i \cdot m$. Toward contradiction, assume that for all $n, m \leq 2^{|Q|}$ there exists some word $w_{n,m}$ such that for some $i \in \mathbb{N}$, we have $\|\mathcal{P}^w_k\| < 1$ where $k = n + i \cdot m$. Then, the symbolic run of the automaton over the randomized word $v = \frac{1}{k^2}w_{1,1} + \cdots + \frac{1}{k^2}w_{k,k}$ where $k = 2^{|Q|}$ is never weakly 1-synchronized implying that v is not sure weakly synchronizing according to max_T, a contradiction with the fact that $\mathcal{P}$ has a universal sure weakly synchronizing language according to max_T. A similar argument proves that the state q where the probability mass is accumulated at steps of the sequence $n, n+m, n+2m, \cdots$ is unique for all words too. In order to decide the universality problem of sure weakly synchronizing language according to max_T, one can compute the sequence $s_0s_1s_2\cdots$ where $s_0 = \{q_{\mathsf{init}}\}$ and $s_i = \mathsf{post}(s_{i-1}, \mathsf{A})$ for all $i \in \mathbb{N}$. The sequence $s_0s_1s_2\cdots$ is ultimately periodic meaning that there are $n, m \leq 2^{|Q|}$ such that $s_n = s_{n+m}$; thus it is sufficient to check if there is some s_i in the periodic part of the

sequence $s_0 s_1 s_2 \cdots$ such that $s_i = \{q\}$ is a singleton with $q \in T$, which can be decided in PSPACE.

A similar argument for the function sum_T shows that to decide the universality problem of sure weakly synchronizing language, the sequence $s_0 s_1 s_2 \cdots$ where $s_0 = \{q_{\mathsf{init}}\}$ and $s_i = \mathsf{post}(s_{i-1}, \mathsf{A})$ for all $i \in \mathbb{N}$, can be computed and then one can check whether there exists some s_i in the periodic part of the sequence $s_0 s_1 s_2 \cdots$ such that $s_i \subseteq T$, which can be done in PSPACE.

We do not provide a matching lower bound for the universality problem of sure weakly synchronizing languages. However, there is a reduction from *tautology problem* proving that deciding whether all pure words are sure weakly synchronizing according to the function sum_T, is co-NP-hard.

Universality problems of almost-sure weakly synchronizing language. Recall that for a PA $\mathcal{P}$, an end component $U \subseteq Q$ is *terminal* if $\mathsf{post}(U, \mathsf{A}) \subseteq U$; and an infinite randomized word is the uniformly randomized word over the alphabet A denoted by $w_u = d_0 d_1 d_2 \cdots$ if d_i is the uniform distribution over A for all $i \in \mathbb{N}$.

For the function max_T, a PA $\mathcal{P}$ with a universal almost-sure weakly synchronizing language, must only have a unique terminal end component. Otherwise, the uniformly randomized word w_u would reach all terminal end components with some positive probability, and thus $\mathcal{P}$ would not be almost-sure weakly synchronizing by w_u. Example 5.3 provides an instance of a PA to show that even though having only one terminal end component is necessary for universality of almost-sure weakly synchronizing language, it is not sufficient.

Example 5.3. *The PA $\mathcal{P}$ depicted in Figure 5.5 has four states $q_{\mathsf{init}}, q_1, q_2$ and q_3 and two actions a, b. All transitions in q_{init} have two successors q_1 and q_2, each with probability $\frac{1}{2}$. Two states q_1 and q_2 are somewhat symmetric: both have one transition directed to another, and one transition directed to the absorbing state q_3; however, the a-transition of one acts like the b-transition of the other.*

The PA $\mathcal{P}$ has only one terminal end component $\{q_3\}$, but the almost-sure weakly synchronizing language is not universal. As a counterexample word, consider $w = a \cdot (a \cdot b)^\omega$ where $\mathcal{P}$ is weakly $\frac{1}{2}$-synchronizing in two sets $\{q_1, q_2\}$ and $\{q_3\}$ (according to max). It implies that having one terminal end component is not sufficient to have a universal almost-sure weakly synchronizing language.

$\triangleleft$

In order to have a universal almost-sure weakly synchronizing language, a PA must ensure that for all randomized words, the probability mass tends to accumulate in a unique terminal end component; we express this property for a terminal end component as being *absorbing*. A terminal end component U is *absorbing* if $\lim_{n\to\infty} \mathcal{P}_n^w(U) = 1$ for all infinite randomized words $w \in \mathcal{D}(\mathsf{A})^\omega$. Example 5.4 shows an instance of a PA where the unique terminal end component is absorbing and the almost-sure weakly synchronizing language is universal.

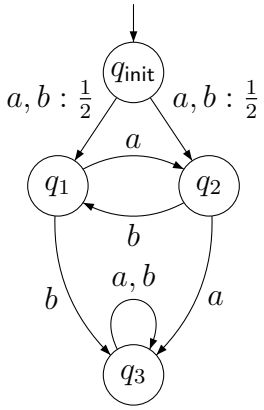

Figure 5.5: Non-absorbing end component.

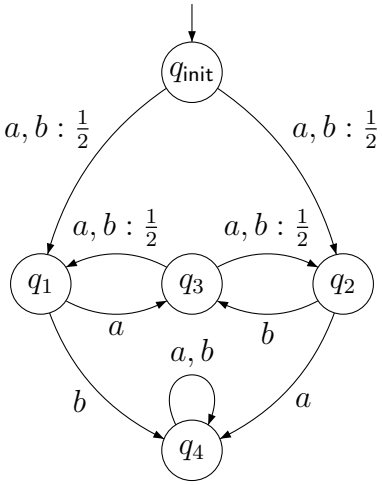

Figure 5.6: Absorbing end component.

Example 5.4. *The PA $\mathcal{P}$ shown in Figure 5.6 has five states $q_{\mathsf{init}}, q_1, q_2, q_3$ and q_4 and two actions a, b. All transitions in q_{init} and q_3 have two successors q_1 and q_2, each with probability $\frac{1}{2}$. Two states q_1 and q_2 are somewhat symmetric: both have one transition directed to q_3, and one transition directed to the absorbing state q_4; however, the a-transition of one acts like the b-transition of the other.*

The PA $\mathcal{P}$ has only one terminal end component $\{q_4\}$, which is absorbing: $\lim_{n\to\infty} \mathcal{P}_n^w(q_4) = 1$ for all infinite randomized words w. The end component $\{q_4\}$ is absorbing because for the other end components $\{q_1, q_2, q_3\}$, even though there are strategies to ensure that once the PA enters in the end component, it stays there (the requirement to be an end component), there is no blind strategy (word) to ensure this property with non-zero probability. We see that after the first input, $\mathcal{P}$ arrives either in q_1 or q_2 each with probability $\frac{1}{2}$. Next, after every two consecutive inputs some positive probability leaks down to the absorbing end component $\{q_4\}$ while the remaining probability is in the end components $\{q_1, q_2, q_3\}$. Repeating this forever, the probability mass is gradually accumulated in $\{q_4\}$ implying that the almost-sure weakly synchronizing language of $\mathcal{P}$ is universal.

◁

Decision problem(s). The *absorbing* decision problem asks, given a PA and an end component, whether the end component is absorbing for the PA.

Lemma 5.10 provides the PSPACE membership of the absorbing problem.

Lemma 5.10. *The absorbing problem for PAs is decidable in* PSPACE.

Proof. Given a terminal end component U of a PA $\mathcal{P}$, we construct an NFA $\mathcal{N}$ equipped with an initial state and a coBüchi acceptance condition such that U is absorbing if and

only if the language of $\mathcal{N}$ is empty. The automaton $\mathcal{N}$ is exponential in the size of $\mathcal{P}$, and since the emptiness problem of NFAs with coBüchi conditions is NLOGSPACE-complete, the PSPACE membership of absorbing problem follows.

Intuitively, the NFA $\mathcal{N}$ guesses a word such that the symbolic-outcome of $\mathcal{P}$ over that word, from some point on, keeps assigning a (strictly) positive probability $p > 0$ to sets of states which are disjoint with U. The alphabet of $\mathcal{N}$ is 2^{A} where A is the alphabet of $\mathcal{P}$. An infinite word over 2^{A} is a sequence of sets of letters which can be viewed as the sequence of supports of a randomized word. The states of $\mathcal{N}$ are pairs (s, b) consisting of a set $s \subseteq Q$ of states of $\mathcal{P}$ and a Boolean flag $b \in \{0, 1\}$. A pair (s, b) is 0-*flagged* if $b = 0$; and it is 1-*flagged* otherwise. The flag b is set to 1 when $\mathcal{N}$ guesses that from now on, $\mathcal{P}$ is always outside of U with some positive probability p. The NFA $\mathcal{N}$ begins in the pair $(\{q_{\mathsf{init}}\}, 0)$ where q_{init} is the initial state of $\mathcal{P}$. On an input $\mathsf{A}' \subseteq \mathsf{A}$, $\mathcal{N}$ not-deterministically moves from a 0-flagged state $(s, 0)$ into

- either another 0-flagged state $(s', 0)$ where $s' = \mathsf{post}(s, \mathsf{A}')$ is the set of successors of s and the input,
- or a 1-flagged state $(c, 1)$ where c is a non-empty subset of $\mathsf{post}(s, \mathsf{A}') \setminus U$; note that c is disjoint with U, i.e., $c \cap U = \emptyset$.

After reaching the first 1-flagged pair $(c, 1)$, $\mathcal{N}$ only checks whether the successors of c (where the probability p was assigned) always has an empty intersection with U. The coBüchi acceptance condition requires that $\mathcal{N}$ visits only 1-flagged pairs, infinitely often.

From the PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ with the initial state q_{init} and the end component U, we construct $\mathcal{N} = \langle L, 2^{\mathsf{A}} \setminus \{\emptyset\}, \Delta \rangle$ where $L = 2^Q \times \{0, 1\}$ is the set of pairs and $2^{\mathsf{A}} \setminus \{\emptyset\}$ is the alphabet. The transition function $\Delta : L \times 2^{\Sigma} \to 2^L$ is defined as follows. For all subsets $s \subseteq Q$ and $\mathsf{A}' \subseteq \mathsf{A}$, let $s' = \mathsf{post}(s, \mathsf{A}')$, and

- define $\Delta((s, 0), \mathsf{A}') = \{(s', 0)\} \cup \{(c, 1) \mid c \neq \emptyset \text{ and } c \subseteq s' \setminus U\}$,
- define $\Delta((s, 1), \mathsf{A}') = \{(s', 1)\}$ if $s' \cap U = \emptyset$, and $\Delta((s, 1)) = \emptyset$ otherwise.

We equip $\mathcal{N}$ with the initial pair $(\{q_{\mathsf{init}}\}, 0)$ and the coBüchi condition $\Diamond\Box\mathcal{F}$ where $\mathcal{F} = 2^Q \times \{1\}$.

To establish the correctness of the reduction, we show that the coBüchi language of $\mathcal{N}$ is empty if and only if the terminal end component U is absorbing. First, assume that the terminal end component U is absorbing for the PA $\mathcal{P}$. Toward contradiction, assume that the coBüchi language of $\mathcal{N}$ is not empty. Therefore, there exists some infinite word v and $n \in \mathbb{N}$ such that for the run $\rho = r_0 r_1 \dots$ of $\mathcal{N}$ over v and for all $i > n$, we have $r_i \in \mathcal{F}$. From the accepting word $v = \mathsf{A}_0 \mathsf{A}_1 \dots$ for coBüchi condition $\Diamond\Box\mathcal{F}$, we construct a randomized word w and claim that the symbolic-run of $\mathcal{P}$ over w tends to accumulate some positive probability outside U. Define $w = d_0 d_1 \dots$ such that d_i is the uniform distribution over A_i, thus $\mathsf{Supp}(d_i) = \mathsf{A}_i$. Let $\mathcal{P}_0^w \mathcal{P}_1^w \cdots$ be the symbolic-run of $\mathcal{P}$ over w. Since $\mathcal{F} = 2^Q \times \{1\}$ and by the construction of $\mathcal{N}$, we know that for all $i > n$, the pair $r_n = (s_i, 1)$ is 1-flagged and $s_{i+1} = \mathsf{post}(s_i, \mathsf{A}_i)$. By construction of w, we see that $s_n \subseteq \mathsf{Supp}(\mathcal{P}_n^w)$; therefore, each s_i is a subset of the support $\mathsf{Supp}(\mathcal{P}_i^w)$ of the symbolic-outcome at step i. Since $\emptyset \neq s_n \subseteq \mathsf{Supp}(\mathcal{P}_n^w)$, we have $\mathcal{P}_n^w(s_n) = p > 0$. Since $s_{i+1} = \mathsf{post}(s_i, \mathsf{A}_i)$ for all $i > n$, the probability $\mathcal{P}_i^w(s_i)$ is always at least p. On the other hand, we have $s_i \subseteq Q \setminus U$ which gives

$\mathcal{P}_i^w(Q \setminus U) \geq p > 0$. Therefore $\lim_{i\to\infty} \mathcal{P}_i^w(U) < 1 - p < 1$, a contradiction with the fact that U is an absorbing end component.

Second, assume that the coBüchi language of $\mathcal{N}$ is empty. We prove that U is absorbing by showing that $\lim_{n\to\infty} \mathcal{P}_n^w(U) = 1$ for all randomized words w. We claim that for all randomized words $w = d_0 d_1 \cdots$, for all $n \in \mathbb{N}$ and for all states $q \in \mathsf{Supp}(\mathcal{P}_n^w)$ where $\mathcal{P}_0^w \mathcal{P}_1^w \cdots$ is the symbolic-run of $\mathcal{P}$ over w, there exists a *w-path* from q to U with length $k \leq 2^{|Q|}$; a *w-path* from $q \in \mathsf{Supp}(\mathcal{P}_n^w)$ to U is a path $\ell_0 \ell_1 \cdots \ell_k$ such that $\ell_0 = q$, $\ell_k \in U$ and for all $0 \leq i < k$, there exists $a \in \mathsf{Supp}(d_{n+i})$ such that $\ell_{i+1} \in \mathsf{post}(\ell_i, a)$. Towards contradiction, assume that there exists w and $m \in \mathbb{N}$, and some state $q_m \in \mathsf{Supp}(\mathcal{P}_m^w)$ such that all w-paths starting in q_m and with length less than $2^{|Q|}$ do not end in U. From the randomized word $w = d_0 d_1 \cdots$, we construct an ultimately periodic run r of the automaton $\mathcal{N}$ which is accepting. For the first $m-1$ transitions, this run ρ visits the 0-flagged states $(s_i, 0)$ where $s_i = \mathsf{Supp}(\mathcal{P}_i^w)$ for all $i < m$. At step m, the NFA $\mathcal{N}$ guesses that the positive probability assigned to q_m will never contribute to the accumulated probability in U, thus the accepting run visits the 1-flagged state $(\{q_m\}, 1)$. Within the next transitions, this run deterministically visits 1-flagged states $(c_{m+1}, 1)(c_{m+2}, 1) \ldots (c_{m+k}, 1)$ where $k = 2^{|Q|}$ and $c_{m+i} = \mathsf{post}(c_{m+i-1}, \mathsf{Supp}(d_{m+i-1}))$ for all $i \leq k$. Since from q_m there is no w-path to U with length less than k, all sets c_{m+i} with $i \leq k$ are disjoint with U. Since the automaton $\mathcal{N}$ has only k 1-flagged states, therefore based on pigeonhole principle, at least one state $(c, 1)$ has been visited twice. This means there is a cycle from $(c, 1)$ to itself which includes only accepting states. Next, the run ρ only visits the states of this cycle which implies that ρ is an accepting run, a contradiction with the fact that the coBüchi language of $\mathcal{N}$ is empty.

Above, we have shown that for all randomized words $w = d_0 d_1 \cdots$, for all $n \in \mathbb{N}$ and for all states $q \in \mathsf{Supp}(\mathcal{P}_n^w)$ where $\mathcal{P}_0^w \mathcal{P}_1^w \cdots$ is the symbolic-run of $\mathcal{P}$ over w, there exists a *w-path* from q to U with length $k \leq 2^{|Q|}$. Therefore, for all words w and $i \in \mathbb{N}$, from all states $q \in \mathsf{Supp}(\mathcal{P}_i^w)$, the end component U is reached within $k = 2^{|Q|}$ steps:

$$\mathsf{Pr}(\Diamond^{\leq k} U) \geq \eta^k \text{ from all states } q \in \mathsf{Supp}(\mathcal{P}_i^w)$$

where η is the smallest probability of taking a transition in $\mathcal{P}$. For all randomized words w, thus,

$$\mathcal{P}_k^w(Q \setminus U) \leq 1 \cdot (1 - \eta^k),$$

because 1 is an upper bound of $\mathcal{P}_0^w(Q \setminus U)$ and $(1-\eta^k)$ is an upper bound for the probability mass which does not move to U within k steps. Similarly,

$$\mathcal{P}_{2\cdot k}^w(Q \setminus U) \leq (1 - \eta^k) \cdot (1 - \eta^k)$$

where $(1 - \eta^k)$ is an upper bound of $\mathcal{P}_{2\cdot k}^w(Q \setminus U)$ and $(1 - \eta^k)$ is an upper bound for the probability mass which does not move to U within k steps. After j repetitions, we have

$$\mathcal{P}_{j\cdot k}^w(Q \setminus U) \leq (1 - \eta^k)^j.$$

Since $0 \leq 1 - \eta^k < 1$, we have

$$\lim_{j\to\infty} \mathcal{P}_{j\cdot k}^w(Q \setminus U) = 0.$$

The above arguments immediately prove that $\lim_{n\to\infty} \mathcal{P}_n^w(U) = 1$ for all words w, meaning that U is an absorbing end component. The proof is complete.

□

Another necessary condition for a PA $\mathcal{P}$ to have a universal almost-sure weakly synchronizing language is that $\mathcal{P}$ is almost-sure weakly synchronizing by the uniformly randomized word. For instance, the automaton presented in Lemma 3.4 has the absorbing end component $\{q_2, q_3\}$, but since the $\mathcal{P}$ is not almost-sure weakly synchronizing in $\{q_3\}$ by the uniformly randomized word, the almost-sure weakly synchronizing language is not universal.

Lemma 5.11. *The universality problem of almost-sure weakly synchronizing languages according to max_T in PAs is in* PSPACE*.*

Proof. Given a PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ with the initial state q_{init} and the function max_T, the almost-sure weakly synchronizing language is universal if and only if

(i) there is a (then necessarily unique) absorbing end component U, and

(ii) $\mathcal{P}$ is almost-sure weakly synchronizing by the uniformly randomized word w_u.

One direction is straightforward. For the reverse direction, assume that both of the conditions (i) and (ii) are fulfilled in the PA $\mathcal{P}$. We show that the almost-sure weakly synchronizing language (according to max_T) of $\mathcal{P}$ is universal. Since $\mathcal{P}$ is almost-sure weakly synchronizing according to max_T by the uniformly randomized word w_u, there exists $\hat{q} \in T \cap U$ such that for all $\epsilon > 0$ and all $m \in \mathbb{N}$, there exists $n > m$ where $\mathcal{P}_n^{w_u}(\hat{q}) > 1 - \epsilon$. Let $c_0 = \{\hat{q}\}$ and define the sequence of sets $c_1 c_2 \cdots$ as follows; $c_{i+1} = \mathsf{post}(c_i, \mathsf{A})$ for all $i \in \mathbb{N}$. Let n be the smallest number such that the absorbing end component $U = \cup_{0 \le i < n} c_i$ is covered with the first n subsets of the sequence $c_0 c_1 \cdots$. We claim that these sets $c_0, c_1, \cdots, c_{n-1}$ are disjoint. There are two cases (1) $c_n = \{\hat{q}\}$ or (2) $c_n \neq \{\hat{q}\}$.

(1) Assume that $c_n = \{\hat{q}\}$. Suppose that there exists some state q and $0 \le i < j < n$ such that $q \in c_i$ and $q \in c_j$. Thus $\hat{q} \in c_{i+n-j}$. Since n is the smallest number such that $U = \cup_{0 \le i < n} c_i$, so $c_0 \subset c_{i+n-j}$. Since the set of all successors reached from c_{i+n-j} in $j - i$ steps is $c_n = \{\hat{q}\}$, so is the set of successors from $c_0 \subset c_{i+n-j}$. Hence, $c_{j-i} = \{\hat{q}\}$ which contradicts with the fact that n is the smallest number.

(2) Assume that $c_n \neq \{\hat{q}\}$. There are two cases: either $\hat{q} \in c_n$ or there exists some $0 < i < n$ such that $\hat{q} \in c_i$. In both cases, there exists $0 < i \le n$ such that $\hat{q} \in c_i$. Since $c_0 \subset c_i$, then $c_j \subseteq c_{i+j}$ for all $j \in \mathbb{N}$. Hence, there exists some $k \ge i$ such that $\{\hat{q}\} \subset c_k$ and $c_k = c_{k+n}$. This argument also implies that there exists no $k' \ge i$ where $c_{k'} = \{\hat{q}\}$. Thus, there exists another state $q \neq \hat{q}$ such that $\{\hat{q}, q\} \subseteq c_k$. Let $\epsilon < \frac{\eta^n}{1+\eta^n}$ where η is the smallest probability of taking a transition in $\mathcal{P}$. Let $n_0 > 0$ be such that $\mathcal{P}_{n_0}^{w_u}(\hat{q}) > 1 - \epsilon$ and $c_k \subseteq \mathsf{Supp}(\mathcal{P}_{n_0}^{w_u})$. Thus, $\mathcal{P}_{n_0+n}^{w_u}(q) \ge \eta^n(1-\epsilon)$ and $\mathcal{P}_{n_0+n}^{w_u}(\hat{q}) < 1 - \eta^n(1-\epsilon)$. We can repeat the arguments for all $j \in \mathbb{N}$ and show that $\mathcal{P}_{n_0+jn}^{w_u}(\hat{q}) < 1 - \eta^n(1-\epsilon) < 1 - \epsilon$ which is a contradiction with the fact that w_u is almost-sure weakly synchronizing according to max_T.

We have shown that the sets $c_0, c_1, \cdots, c_{n-1}$ are disjoints and $c_n = c_0 = \{\hat{q}\}$ where $\hat{q} \in T$. Towards contradiction with the universality of the almost-sure weakly synchronizing language of $\mathcal{P}$, assume that there exists an infinite word w that is not almost-sure weakly synchronizing according to max_T. Let $\mathcal{P}_0^w \mathcal{P}_1^w \cdots$ be the symbolic-run of $\mathcal{P}$ over w; and $\mathcal{P}_0^{w_u} \mathcal{P}_1^{w_u} \cdots$ be the symbolic-run over the uniformly randomized word w_u. Since U is absorbing and it consists of n disjoint sets $c_0, c_1, \cdots, c_{n-1}$ where $c_0 \subseteq T$ is a singleton, there exists $m \in \mathbb{N}$ such that for all $n > m$ there are two subsets c and c' (from the sets $c_0, c_1, \cdots, c_{n-1}$ covering U) where $c \cup c' \subseteq \mathsf{Supp}(\mathcal{P}_n^w)$; otherwise w would be almost-sure weakly synchronizing. By assumption, the uniformly randomized word w_u is almost-sure weakly synchronizing. All states reachable by w are also reachable by w_u: $\mathsf{Supp}(\mathcal{P}_n^w) \subseteq \mathsf{Supp}(\mathcal{P}_n^{w_u})$. Hence, for all $n > m$ there are two subsets c and c' (from the sets $c_0, c_1, \cdots, c_{n-1}$ covering U) such that $c \cup c' \subseteq \mathsf{Supp}(\mathcal{P}_n^{w_u})$ too. Since the probabilities assigned to these disjoint sets c and c' always loop through the sequence $c_0 c_1 \cdots c_n$, the probability mass could never accumulate in the singleton c_0 (or any of singletons $c \subseteq T$ in the sequence $c_0 c_1 \cdots c_n$), a contradiction with condition (ii) stating that w_u is almost-sure weakly synchronizing.

Condition (i) can be checked in PSPACE by Lemma 5.10, and condition (ii) can be checked in PTIME by steady state analysis of the Markov chain induced by the PA under the uniformly randomized word. The PSPACE bound follows.

□

We have proved that, given a PA $\mathcal{P}$ with the initial state q_{init} and the function max_T, the almost-sure weakly synchronizing language (according to max_T) of $\mathcal{P}$ is universal if and only if (i) there is an absorbing end component U, and (ii) $\mathcal{P}$ is almost-sure weakly synchronizing by the uniformly randomized word w_u. As we have shown in the proof of Lemma 5.11, these two conditions enforce a special shape to the absorbing end component U; the end component U is partitioned into n disjoints sets $c_0, c_1 \cdots, c_n$ where one of them is singleton, and for the symbolic-run of the PA over the uniformly randomized word, all supports can only contain one of these disjoint sets c_i $(0 \leq i < n)$. One can observe that these disjoint sets are indeed the periodic supports of recurrent states in the Markov chain induced by the PA under the uniformly randomized word.

For the function sum_T, a similar argument shows that the universality of almost-sure weakly synchronizing language (according to sum_T) of $\mathcal{P}$ is decidable in PSPACE. The requirement of having an absorbing end component is not necessary for the function sum_T; in fact, states in all end components U can contribute in accumulating the probability mass in the target set T. Each end component U can be treated as an absorbing end component by removing all outgoing transitions, and the periodic supports of recurrent states $c_0, c_1, \cdots, c_n$ can be computed in PSPACE. From those supports, a randomized word w can be constructed such that it is a linear combinations from all pure words where the PA stays in U with some positive probability, once it has entered in U (by those pure words); and check whether $\mathcal{P}$ is almost-sure weakly synchronizing by w. After repeating these arguments for all end components U, the PA must be almost-sure weakly synchronizing by the uniformly randomized word w_u too.

The above arguments provided the PSPACE membership of the the universality problem of almost-sure weakly synchronizing languages for both function sum_T and max_T, a matching lower bound can be proved by the same reduction presented in Lemma 4.12 for eventually synchronizing languages. In the presented reduction, the constructed PA is almost-sure eventually synchronizing in the singleton $\{\mathsf{synch}\}$ if and only if it is almost-sure weakly synchronizing.

From the above arguments and Lemma 5.11, it turns out that for all functions $f \in \{max_T, sum_T\}$, the universality problem of almost-sure weakly synchronizing language in PAs is PSPACE-complete.

6 Strongly Synchronizing Condition

First sight. In this chapter, we show that the membership problem for strongly synchronizing conditions in MDPs is PTIME-complete, for all winning modes, and both with function max_T and function sum_T. We show that linear-size memory is necessary in general for max_T, and memoryless strategies are sufficient for sum_T. It follows from our results that the limit-sure and almost-sure winning modes coincide for strong synchronization.

Moreover, we study the emptiness problem for strongly synchronizing languages in PAs. We prove that for sure and almost-sure languages according to both functions max_T and sum_T, the emptiness problem is PSPACE-complete whereas the problems for limit-sure languages are undecidable. The results presented in this chapter are summarized in Table 6.1 and Table 6.2 on pages 116 and 124, respectively.

Contents

	Strongly			
	max		*sum*	
	Complexity	**Required memory**	**Complexity**	**Required memory**
Sure	PTIME-complete	linear	PTIME-complete	memoryless
Almost-sure	PTIME-complete	linear	PTIME-complete	memoryless
Limit-sure				

Table 6.1: Computational complexity of the membership problem of strongly synchronization in MDPs, and memory requirement for the winning strategies (Two winning modes almost-sure and limit-sure coincide).

6.1 Strong synchronization in MDPs

For strong synchronization the membership problem with function max_T reduces to the membership problem with function max_Q where Q is the entire state space, by a construction similar to the proof of Remark 5: states in $Q \setminus T$ are duplicated, in such a way that a state $q \in Q \setminus T$ contains some probability p if and only if the duplication of that state q contain the exact probability p too. This construction ensures that only states in T are used to accumulate the probability mass tending to 1.

The results presented in this section are summarized in Table 6.1.

6.1.1 Strong synchronization with function *max*

In this subsection, to prove the PTIME-completeness for the membership problem of strong synchronization with function max_T, we solve the problem with function max_Q that establishes the upper bound. We provide the matching lower bound by proving PTIME-hardness for the membership problem of strong synchronization with function max_T where T is a singleton.

The strongly synchronizing condition with function *max* requires that from some point on, almost all the probability mass is at every step in a single state. The sequence of states that contain almost all the probability corresponds to a sequence of deterministic transitions in the MDP, and thus eventually to a cycle of deterministic transitions.

The *graph of deterministic transitions* of an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ is the directed graph $G = \langle Q, E \rangle$ where $E = \{\langle q_1, q_2 \rangle \mid \exists a \in \mathsf{A} : \delta(q_1, a)(q_2) = 1\}$. For $\ell \geq 1$, a *deterministic cycle in* $\mathcal{M}$ of length ℓ is a finite path $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ in G (that is, $\langle \hat{q}_i, \hat{q}_{i-1} \rangle \in E$ for all $1 \leq i \leq \ell$) such that $\hat{q}_0 = \hat{q}_\ell$. The cycle is *simple* if $\hat{q}_i \neq \hat{q}_j$ for all $1 \leq i < j \leq \ell$.

We show that sure (resp., almost-sure and limit-sure) strong synchronization is equivalent to sure (resp., almost-sure and limit-sure) reachability to a state in such a cycle, with the requirement that it can be reached in a synchronized way, that is by finite paths whose lengths are congruent modulo the length ℓ of the cycle. To check this, we keep track of a modulo-ℓ counter along the play.

Define the MDP $\mathcal{M} \times [\ell] = \langle Q', \mathsf{A}, \delta' \rangle$ where $Q' = Q \times \{0, 1, \cdots, \ell - 1\}$ and $\delta'(\langle q, i \rangle, a)(\langle q', i-1 \rangle) = \delta(q, a)(q')$ (where $i-1$ is $\ell - 1$ for $i = 0$) for all states $q, q' \in Q$, actions $a \in \mathsf{A}$, and $0 \leq i \leq \ell - 1$.

Lemma 6.1. *Given an MDP $\mathcal{M}$, let η be the smallest positive probability in the transitions of $\mathcal{M}$ and let $\frac{1}{1+\eta} < p \leq 1$. There exists a strategy α such that $\liminf_{n\to\infty} \|\mathcal{M}_n^\alpha\| \geq p$ from an initial state q_{init} if and only if there exists a simple deterministic cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ in $\mathcal{M}$ and a strategy β in $\mathcal{M} \times [\ell]$ such that $\Pr^\beta(\Diamond\{\langle \hat{q}_0, 0 \rangle\}) \geq p$ from $\langle q_{\mathsf{init}}, 0 \rangle$.*

Proof. First, assume that there exists a simple deterministic cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ with length ℓ and a strategy β in $\mathcal{M} \times [\ell]$ that ensures the target set $\Diamond\{\langle \hat{q}_0, 0 \rangle\}$ is reached with probability at least p from the state $\langle q_{\mathsf{init}}, 0 \rangle$. Since randomization is not necessary for reachability conditions, we can assume that β is a pure strategy. We show that there exists a strategy α such that $\liminf_{n\to\infty} \|\mathcal{M}_n^\alpha\| \geq p$ from q_{init}. From β, we construct a pure strategy α in $\mathcal{M}$. Given a finite path $\rho = q_0 a_0 q_1 a_1 \dots q_n$ in $\mathcal{M}$ (with $q_0 = q_{\mathsf{init}}$), there is a corresponding path $\rho' = \langle q_0, k_0 \rangle a_0 \langle q_1, k_1 \rangle a_1 \dots \langle q_n, k_n \rangle$ in $\mathcal{M} \times [\ell]$ where $k_i = -i \ mod \ \ell$. Since the sequence $k_0 k_1 \dots$ is uniquely determined from ρ, there is a clear bijection between the paths in $\mathcal{M}$ and the paths in $\mathcal{M} \times [\ell]$ that we often omit to apply and mention. For ρ, we define α as follows: if $q_n = \hat{q}_{k_n}$, then there exists an action a such that $\mathsf{post}(\hat{q}_{k_n}, a) = \{\hat{q}_{k_{n+1}}\} = \{\hat{q}_{n+1}\}$ and we define $\alpha(\rho) = a$, otherwise let $\alpha(\rho) = \beta(\rho')$. Thus α mimics β unless a state q is reached at step n such that $q = \hat{q}_k$ where $k = -n \ mod \ \ell$, and then α switches to always playing actions that keeps $\mathcal{M}$ in the simple deterministic cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$. Below, we prove that given $\epsilon > 0$ there exists k such that for all $n \geq k$, we have $\|\mathcal{M}_n^\alpha\| \geq p - \epsilon$. It follows that $\liminf_{n\to\infty} \|\mathcal{M}_n^\alpha\| \geq p$ from q_{init}. Since $\Pr^\beta(\Diamond\{\langle \hat{q}_0, 0 \rangle\}) \geq p$, there exists k such that $\Pr^\beta(\Diamond^{\leq k}\{\langle \hat{q}_0, 0 \rangle\}) \geq p - \epsilon$. We assume w.l.o.g. that $k \ mod \ \ell = 0$. For $i = 0, 1, \dots, \ell - 1$, let $R_i = \{\langle \hat{q}_i, i \rangle\}$. Then trivially $\Pr^\beta(\Diamond^{\leq k} \bigcup_{i=0}^{\ell-1} R_i) \geq p - \epsilon$ and since α agrees with β on all finite paths that do not (yet) visit $\bigcup_{i=0}^{\ell-1} R_i$, given a path ρ that visits $\bigcup_{i=0}^{\ell-1} R_i$ (for the first time), only actions that keep $\mathcal{M}$ in the simple cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ are played by α and thus all continuations of ρ in the outcome of α will visit $\hat{q}_0$ after k steps (in total). It follows that $\Pr^\beta(\Diamond^k\{\langle \hat{q}_0, 0 \rangle\}) \geq p - \epsilon$, that is $\mathcal{M}_k^\alpha(\hat{q}_0) \geq p - \epsilon$. Thus, $\|\mathcal{M}_k^\alpha\| \geq p - \epsilon$. Since next, α will always play actions that keeps $\mathcal{M}$ looping through the cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$, we have $\|\mathcal{M}_n^\alpha\| \geq p - \epsilon$ for all $n \geq k$.

Second, assume that there exists a strategy α such that $\liminf_{n\to\infty} \|\mathcal{M}_n^\alpha\| \geq p$ from q_{init}. Thus, for all $\epsilon > 0$ there exists $k \in \mathbb{N}$ such that for all $n \geq k$ we have $\|\mathcal{M}_n^\alpha\| \geq p - \epsilon$. Fix $\epsilon < p - \frac{1}{1+\eta}$. Let k be such that for all $n \geq k$, there exists a unique state $\hat{p}_n$ such that $\mathcal{M}_n^\alpha(\hat{p}_n) \geq p - \epsilon$. Below, we prove that for all $n \geq k$, there exists some action $a \in \mathsf{A}$ such that $\mathsf{post}(\hat{p}_n, a) = \{\hat{p}_{n+1}\}$. Assume towards contradiction that there exists $j > k$ such that for all a there exists $q \neq \hat{p}_{j+1}$ such that $\{q, \hat{p}_{j+1}\} \subseteq \mathsf{post}(\hat{p}_j, a)$. Therefore, $\mathcal{M}_{j+1}^\alpha(q) \geq \mathcal{M}_j^\alpha(\hat{p}_j) \cdot \eta \geq (p - \epsilon) \cdot \eta$. Hence,

$$\mathcal{M}_{j+1}^\alpha(\hat{p}_{j+1}) \leq 1 - \mathcal{M}_{j+1}^\alpha(q) \leq 1 - (p - \epsilon) \cdot \eta.$$

Thus, $p - \epsilon \leq \|\mathcal{M}_{j+1}^\alpha\| \leq 1 - (p - \epsilon) \cdot \eta$ that gives $\epsilon \geq p - \frac{1}{1+\eta}$, a contradiction. This argument proves that for all $n \geq k$, there exists an action $a \in \mathsf{A}$ such that $\mathsf{post}(\hat{p}_n, a) = \{\hat{p}_{n+1}\}$. The

finiteness of the state space Q entails that in the sequence $\hat{p}_k\hat{p}_{k+1}\cdots$, some state and thus some simple deterministic cycle occur infinitely often. Let $\hat{q}_\ell\hat{q}_{\ell-1}\cdots\hat{q}_0$ be a cycle that occurs infinitely often in the sequence $\hat{p}_k\hat{p}_{k+1}\cdots$ (in the right order). For all j, let i_j be the position of $\hat{q}_0$ in all occurrences of the cycle $\hat{q}_\ell\hat{q}_{\ell-1}\cdots\hat{q}_0$ in the sequence $\hat{p}_k\hat{p}_{k+1}\cdots$; and let $t_j = i_j \ mod \ \ell$. In the sequence $t_0t_1\cdots$, there exists $0 \leq t < \ell$ that appears infinitely often. Let the cycle $r_\ell r_{\ell-1}\cdots r_0$ be such that $r_{(i+t) \ mod \ \ell} = \hat{q}_i$. Then, the cycle $r_\ell r_{\ell-1}\cdots r_0$ happens infinitely often in the sequence $\hat{p}_k\hat{p}_{k+1}\cdots$ such that the positions of r_0 are infinitely often 0 (modulo ℓ). Therefore, the probability of $\mathcal{M}$ to be in r_0 in positions (modulo ℓ) equals to 0, is infinitely often equal or greater than p. Hence, for a strategy β in $\mathcal{M} \times [\ell]$ that copies all the plays of the strategy α, we have $\Pr^\beta(\Diamond\{\langle r_0, 0\rangle\}) \geq p$ from $\langle q_{\mathsf{init}}, 0\rangle$.

□

It follows directly from Lemma 6.1 with $p = 1$ that almost-sure strong synchronization is equivalent to almost-sure reachability to a deterministic cycle in $\mathcal{M} \times [\ell]$. The same equivalence holds for the sure and limit-sure winning modes.

Lemma 6.2. *Given an MDP $\mathcal{M}$, a state q_{init} is sure (resp., almost-sure or limit-sure) winning for the strongly synchronizing condition (according to max_Q) if and only if there exists a simple deterministic cycle $\hat{q}_\ell\hat{q}_{\ell-1}\cdots\hat{q}_0$ such that $\langle q_{\mathsf{init}}, 0\rangle$ is sure (resp., almost-sure or limit-sure) winning for the reachability condition $\Diamond\{\langle\hat{q}_0, 0\rangle\}$ in $\mathcal{M} \times [\ell]$.*

Proof. The proof is organized in three sections:

(1) sure winning mode: First, assume that there exists a simple deterministic cycle $\hat{q}_\ell\hat{q}_{\ell-1}\cdots\hat{q}_0$ with length ℓ such that $\langle q_{\mathsf{init}}, 0\rangle$ is sure winning for the reachability condition $\Diamond\{\langle\hat{q}_0, 0\rangle\}$. Thus, there exists a pure memoryless strategy β such that $Outcomes(\langle q_{\mathsf{init}}, 0\rangle, \beta) \subseteq \Diamond\{\langle\hat{q}_0, 0\rangle\}$. Since β is memoryless, there must be $k \leq |Q| \times \ell$ such that $Outcomes(\langle q_{\mathsf{init}}, 0\rangle, \beta) \subseteq \Diamond^{\leq k}\{\langle\hat{q}_0, 0\rangle\}$ meaning that all infinite paths starting in $\langle q_{\mathsf{init}}, 0\rangle$ and following β reach $\langle\hat{q}_0, 0\rangle$ within k steps. From β, we construct a pure finite-memory strategy α in $\mathcal{M}$ that is represented by $T = \langle \mathsf{Memory}, i, \alpha_u, \alpha_n\rangle$ where $\mathsf{Memory} = \{0, \cdots, \ell - 1\}$ is the set of modes. The idea is that α simulates what β plays in the state $\langle q, i\rangle$, in the state q of $\mathcal{M}$ and mode i of T (there is only one exception). Thus, the initial mode is 0. The update function only decreases modes by 1 ($\alpha_u(i, a, q) = (i - 1) \ mod \ \ell$ for all states q and actions a) since by taking any transition the mode is decreased by 1. The next-move function $\alpha_n(i, q)$ is defined as follows: $\alpha_n(i, q) = \beta(\langle q, i\rangle)$ for all states q and modes $0 \leq i < \ell$, except when $q = \hat{q}_i$, in this case let $\alpha_n(i, q) = a$ where $\mathsf{post}(\hat{q}_i, a) = \{q_{i-1}\}$. Thus β mimics α unless a state q is reached at step n such that $q = \hat{q}_{-n \ mod \ \ell}$, and then α switches to always playing actions that keeps $\mathcal{M}$ in the simple deterministic cycle $\hat{q}_\ell\hat{q}_{\ell-1}\cdots\hat{q}_0$. Now we prove that q_{init} is sure winning for the strongly synchronizing condition according to max_Q. Let $j \geq k$ be such that $j \ mod \ \ell = 0$. Let $R = \{\langle\hat{q}_i, i\rangle \mid 0 \leq i < \ell\}$. Thus obviously $Outcomes(\langle q_{\mathsf{init}}, 0\rangle, \beta) \subseteq \Diamond R$. and since α agrees with β on all finite paths that do not (yet) visit R, given a path ρ that visits R (for the first time), only actions that keep $\mathcal{M}$ in the simple cycle $\hat{q}_\ell\hat{q}_{\ell-1}\cdots\hat{q}_0$ are played by α and thus all continuations of ρ in the outcome of α will visit $\hat{q}_0$ after j steps. It follows

that $\Pr^{\beta}(\Diamond^{j}\{\langle \hat{q}_0, 0\rangle\}) = 1$, that is $\mathcal{M}_j^{\alpha}(q_0) = 1$. Thus, $\|\mathcal{M}_j^{\alpha}\| = 1$. Since next, α will always play actions that keeps $\mathcal{M}$ looping through the cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$, we have $\|\mathcal{M}_n^{\alpha}\| = 1$ for all $n \geq j$.

Second, assume that there exists a strategy α and k such that for all $n \geq k$ we have $\|\mathcal{M}_n^{\alpha}\| = 1$ from the initial state q_{init}. For all $n \geq k$, let $\hat{p}_n$ be a state such that $\mathcal{M}_n^{\alpha}(\hat{p}_n) = 1$. The finiteness of the state space Q entails that in the sequence $\hat{p}_k \hat{p}_{k+1} \cdots$, some state and thus some simple deterministic cycle occur infinitely often. Let $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ be a cycle that occurs infinitely often in the sequence $\hat{p}_k \hat{p}_{k+1} \cdots$ (in the right order). For all j, let i_j be the position of $\hat{q}_0$ in all occurrences of the cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ in the sequence $\hat{p}_k \hat{p}_{k+1} \cdots$; and let $t_j = i_j \ mod \ \ell$. In the sequence $t_0 t_1 \cdots$, there exists $0 \leq t < \ell$ that appears infinitely often. Let the cycle $r_\ell r_{\ell-1} \cdots r_0$ be such that $r_{(i+t) \ mod \ \ell} = \hat{q}_i$. Then, the cycle $r_\ell r_{\ell-1} \cdots r_0$ happens infinitely often in the sequence $\hat{p}_k \hat{p}_{k+1} \cdots$ such that the positions of r_0 are infinitely often 0 (modulo ℓ). Hence, for a strategy β in $\mathcal{M} \times [\ell]$ that copies all the plays of the strategy α, we have $\mathit{Outcomes}(\langle q_{\mathsf{init}}, 0\rangle, \beta) \subseteq \Diamond\{\langle r_0, 0\rangle\}$ from the initial state $\langle q_{\mathsf{init}}, 0\rangle$.

(2) almost-sure winning mode: This case is an immediate result from Lemma 6.1, by taking $p = 1$.

(3) limit-sure winning mode: First, assume that there exists a simple deterministic cycle $\hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ with length ℓ such that $\langle q_{\mathsf{init}}, 0\rangle$ is limit-sure (and thus almost-sure) winning for the reachability condition $\Diamond\{\langle \hat{q}_0, 0\rangle\}$). Since $\langle \hat{q}_{\mathsf{init}}, 0\rangle$ is almost-sure for reachability condition, then q_{init} is almost-sure (and thus limit-sure) for strongly synchronizing condition. Second, assume that q_{init} is limit-sure winning for the strongly synchronizing condition (according to max_Q). It means that for all i there exists a strategy α_i such that $\liminf_{n\to\infty} \|\mathcal{M}_n^{\alpha_i}\| \geq 1 - 2^{-i}$. Let k be such that $1 - 2^{-k} \geq \frac{1}{1+\eta}$. By Lemma 6.1, for all $i \geq k$ there exists a simple deterministic cycle $c_i = \hat{p}_{\ell_i} \hat{p}_{\ell_i - 1} \cdots \hat{p}_0$ with length ℓ_i and a strategy β_i in $\mathcal{M} \times [\ell_i]$ such that $\Pr^{\beta_i}(\Diamond\{\langle \hat{q}_0, 0\rangle\}) \geq 1 - 2^{-i}$ from $\langle q_{\mathsf{init}}, 0\rangle$. Since the number of simple deterministic cycle is finite, there exists some simple cycle c that occurs infinitely often in the sequence $c_k c_{k+1} c_{k+2} \cdots$. We see that for the cycle $c = \hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$, the states $\langle \hat{q}_{\mathsf{init}}, 0\rangle$ is limit-sure winning for the reachability condition $\Diamond\{\langle \hat{q}_0, 0\rangle\}$).

□

Since the winning regions of almost-sure and limit-sure winning coincide for reachability conditions in MDPs [dAHK07], the next corollary follows from Lemma 6.2.

Corollary 6.1. $\langle\!\langle 1 \rangle\!\rangle_{limit}^{strongly}(max_T) = \langle\!\langle 1 \rangle\!\rangle_{almost}^{strongly}(max_T)$ *for all MDPs* $\mathcal{M}$ *and target sets* T.

If there exists a cycle c satisfying the condition in Lemma 6.2, then all cycles reachable from c in the graph G of deterministic transitions also satisfies the condition. Hence it is sufficient to check the condition for an arbitrary simple cycle in each strongly connected component (SCC) of G. It follows that strong synchronization can be decided in PTIME (SCC decomposition can be computed in PTIME, as well as sure, limit-sure, and almost-sure reachability in MDPs). The length of the cycle gives a linear bound on the memory needed to win, and the bound is tight.

Lemma 6.3. *For all sure, almost-sure and limit-sure winning modes, the membership problem for strongly synchronizing condition according to max_T in MDPs is in* PTIME.

Proof. Given an MDP $\mathcal{M} = \langle Q, \mathsf{A}, \delta \rangle$ and a state q_{init}, we say a simple deterministic cycle $c = \hat{q}_\ell \hat{q}_{\ell-1} \cdots \hat{q}_0$ is sure winning (resp., almost-sure and limit-sure) for strong synchronization from q_{init} if $\langle q_{\mathsf{init}}, 0 \rangle$ is sure winning (resp., almost-sure and limit-sure) for the reachability condition $\Diamond\{\langle \hat{q}_0, 0 \rangle\}$ in $\mathcal{M} \times [\ell]$. We show that if c is sure winning (resp., almost-sure and limit-sure) for strong synchronization from q_{init}, then so are all simple cycles $c' = \hat{p}_{\ell'} \hat{p}_{\ell'-1} \cdots \hat{p}_0$ reachable from c in the deterministic digraph induced by $\mathcal{M}$.

(1) sure winning: Since c is sure winning for strong synchronization from q_{init}, $\mathcal{M}$ is 1-synchronized in $\hat{q}_0$. Since there is a path via deterministic transitions from $\hat{q}_0$ to $\hat{p}_0$, $\mathcal{M}$ is 1-synchronized in $\hat{p}_0$ too. So the cycle c' is sure winning for strong synchronization from q_{init}, too.

(2) limit-sure winning: Assume that c is limit-sure winning for strong synchronization from q_{init}. By definition, for all $i \in \mathbb{N}$, there exists n such that for all $j > n$ we have $\mathcal{M}$ is $1 - 2^{-i-j}$ in $\hat{q}_0$. It implies that for all i there exists n such that $\mathcal{M}$ is $1 - 2^{-2i}$-synchronized in $\hat{q}_0$. Since there is a path via deterministic transitions from $\hat{q}_0$ to $\hat{p}_0$, then $\mathcal{M}$ is $1 - 2^{-2i}$-synchronized in $\hat{p}_0$ for all i. So the cycle c' is limit-sure winning for strong synchronization from q_{init}, too.

(3) almost-sure winning: By corollary 6.1, since a cycle is almost-sure winning for strong synchronization from q_{init} if and only if it is limit-sure winning, the results follows.

The above arguments prove that if a simple deterministic cycle c is sure winning (resp., almost-sure and limit-sure) for strongly synchronizing condition from q_{init}, then all simple cycles reachable from c in the graph of deterministic transitions G induced by $\mathcal{M}$, are sure winning (resp., almost-sure and limit-sure). In particular, it holds for all simple cycles in the bottoms SCCs reachable from c in G. Therefore, to decide membership problem for strongly synchronizing condition, it suffices to only check whether one cycle in each bottom SCC of G is sure winning (resp., almost-sure and limit-sure). Since the SCC decomposition for a digraph is in PTIME, and since the number of bottom SCCs in a connected digraph is at most the size of the digraph (the number of states $|Q|$), the PTIME upper bound follows.

□

Since memoryless strategies are sufficient for reachability conditions in MDPs, it follows from the proof of Lemma 6.1 and Lemma 6.2 that the (memoryless) winning strategies in $\mathcal{M} \times [\ell]$ can be transferred to winning strategies with memory $\{0, 1, \cdots, \ell - 1\}$ in $\mathcal{M}$. Since $\ell \leq |Q|$, linear-size memory is sufficient to win strongly synchronizing conditions. We present a family of MDPs $\mathcal{M}_n$ ($n \in \mathbb{N}$) that are sure winning for strongly synchronization (according to $\max_Q$), and where the sure winning strategies require linear memory. The MDP $\mathcal{M}_2$ is shown in Figure 6.1, and the MDP $\mathcal{M}_n$ is obtained by replacing the cycle $q_2 q_3$ of deterministic transitions by a simple cycle of length n. Note that only in q_1 there is a real strategic choice. Since after the first action the two states q_1 and q_2 always contain some probability, we need to wait in q_1 (by playing b) until we play a when the probability

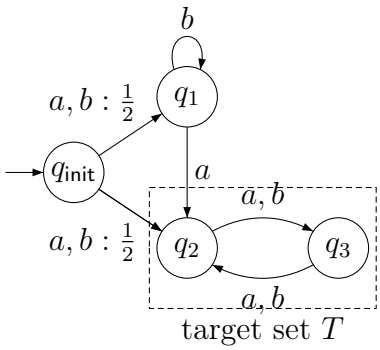

Figure 6.1: An MDP that all strategies to win sure strongly synchronizing with function $max_{\{q_2,q_3\}}$ require memory.

in q_2 comes back in q_2 through the cycle. We need to play $n-1$ times b, and then a, thus linear memory is sufficient and it is easy to show that it is necessary to ensure strongly synchronization.

Lemma 6.4. *For $\mu \in \{sure, almost, limit\}$, the membership problem for $\langle\langle 1\rangle\rangle_{\mu}^{strongly}(max_T)$ in MDPs is* PTIME-hard *even for a singleton T.*

Proof. For all $\mu \in \{sure, almost, limit\}$ the proof is by a reduction from monotone Boolean circuit problem (MBC). Given an MBC with an underlying binary tree, the value of leaves are either 0 or 1 (called 0 or 1-leaves), and the value of other vertices, labeled with $\wedge$ or $\vee$, are computed inductively. It is known that deciding whether the value of the *root* is 1 for a given MBC, is PTIME-complete [GR88]. From an MBC, we construct an MDP $\mathcal{M}$ where the states are the vertices of the tree with three new absorbing states synch, q_1 and q_2, and two actions L, R. On both actions L and R, the next successor of the 1-leaves is only synch, and the next successor of the 0-leaves is q_1 or q_2 with probability $\frac{1}{2}$. The next successor of a $\wedge$-state is each of their children with equal probability, on all actions. The next successor of a $\vee$-state is its left (resp., right) child by action L (resp., action R). We can see that $\mathcal{M}$ can be synchronized only in synch.

We call a subtree *complete* if (1) *root* is in the subtree, (2) at least one child of all $\vee$-vertices is in the subtree, (3) both children of all $\wedge$-vertices are in the subtree. There is a bijection between a complete subtree and a strategy in $\mathcal{M}$. The value of *root* is 1 if and only if there is a complete subtree such that it has no 0-leaves (all leaves are 1-leaves). For such subtrees, all plays under the corresponding strategy reach some 1-leaf and thus are synchronized in synch. It means that $root \in \langle\langle 1\rangle\rangle_{\mu}^{strongly}(\mathsf{synch})$ if and only if the value of *root* is 1. The proof is complete and the PTIME-hardness result is established.

□

From previous lemmas and arguments, Theorem 6.1 follows.

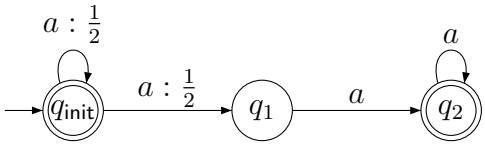

Figure 6.2: An MDP such that q_{init} is sure-winning for coBüchi condition in $T = \{q_{\mathsf{init}}, q_2\}$ but not for sure strong synchronization according to sum_T.

Theorem 6.1. *For the three winning modes of strong synchronization according to max_T in MDPs:*

1. *(Complexity). The membership problem is* PTIME-complete.
2. *(Memory). Linear memory is necessary and sufficient for both pure and randomized strategies, and pure strategies are sufficient.*

6.1.2 Strong synchronization with function *sum*

The strongly synchronizing condition with function sum_T requires that eventually all the probability mass remains in T. We show that this is equivalent to a traditional reachability condition with target defined by the set of sure winning initial distributions for the safety condition $\Box T$.

It follows that almost-sure (and limit-sure) winning for strong synchronization is equivalent to almost-sure (or equivalently limit-sure) winning for the coBüchi condition $\Diamond\Box T$. However, sure strong synchronization is not equivalent to sure winning for the coBüchi condition. Example 6.1 provides an MDP and a target set T such that the MDP is sure winning for coBüchi condition $\Diamond\Box T$, but not for sure strongly synchronizing with sum_T.

Example 6.1. *Consider the MDP $\mathcal{M}$ depicted in Figure 6.2 that has three states $q_{\mathsf{init}}, q_1, q_2$ and one letter a. All a-transitions are deterministic except in q_{init} where the a-transition has two successors q_{init} and q_1 each with probability $\frac{1}{2}$. The MDP $\mathcal{M}$ is sure winning for the coBüchi condition $\Diamond\Box\{q_{\mathsf{init}}, q_2\}$ from q_{init}, but not sure winning for the reachability condition $\Diamond S$ where $S = \{q_2\}$ is the winning region for the safety condition $\Box\{q_{\mathsf{init}}, q_2\}$ and thus not sure strongly synchronizing.*

The MDP $\mathcal{M}$ in Example 6.1 is almost-sure strongly synchronizing in target $T = \{q_{\mathsf{init}}, q_2\}$ from q_{init}, and almost-sure winning for the coBüchi condition $\Diamond\Box T$, as well as almost-sure winning for the reachability condition $\Diamond S$ where $S = \{q_2\}$.

Lemma 6.5. *Given a target set T, an MDP $\mathcal{M}$ is sure (resp., almost-sure or limit-sure) winning for the strongly synchronizing condition according to sum_T if and only if $\mathcal{M}$ is sure (resp., almost-sure or limit-sure) winning for the reachability condition $\Diamond S$ where S is the sure winning region for the safety condition $\Box T$.*

Proof. First, assume that a state q_{init} of $\mathcal{M}$ is sure (resp., almost-sure or limit-sure) winning for the strongly synchronizing condition according to sum_T, and show that q_{init} is sure (resp., almost-sure or limit-sure) winning for the reachability condition $\Diamond S$.

(i) *Limit-sure winning.* For all $\epsilon > 0$, let $\epsilon' = \frac{\epsilon}{|Q|} \cdot \eta^{|Q|}$ where η is the smallest positive probability in the transitions of $\mathcal{M}$. By the assumption, from q_{init} there exists a strategy α and $N \in \mathbb{N}$ such that for all $n \geq N$, we have $\mathcal{M}_n^\alpha(T) \geq 1 - \epsilon'$. We claim that at step N, all non-safe states have probability at most $\frac{\epsilon}{|Q|}$, that is $\mathcal{M}_N^\alpha(q) \leq \frac{\epsilon}{|Q|}$ for all $q \in Q \setminus S$. Towards contradiction, assume that $\mathcal{M}_N^\alpha(q) > \frac{\epsilon}{|Q|}$ for some non-safe state $q \in Q \setminus S$. Since $q \notin S$ is not safe, there is a path of length $\ell \leq |Q|$ from q to a state in $Q \setminus T$, thus with probability at least $\eta^{|Q|}$. It follows that after $N + \ell$ steps we have $\mathcal{M}_{N+\ell}^\alpha(Q \setminus T) > \frac{\epsilon}{|Q|} \cdot \eta^{|Q|} = \epsilon'$, in contradiction with the fact $\mathcal{M}_n^\alpha(T) \geq 1 - \epsilon'$ for all $n \geq N$. Now, since all non-safe states have probability at most $\frac{\epsilon}{|Q|}$ at step N, it follows that $\mathcal{M}_N^\alpha(Q \setminus S) \leq \frac{\epsilon}{|Q|} \cdot |Q| = \epsilon$ and thus $\Pr^\alpha(\Diamond S) \geq 1 - \epsilon$. Therefore $\mathcal{M}$ is limit-sure winning for the reachability condition $\Diamond S$ from q_{init}.

(ii) *Almost-sure winning.* Since almost-sure strongly synchronizing implies limit-sure strongly synchronizing, it follows from (i) that $\mathcal{M}$ is limit-sure (and thus also almost-sure) winning for the reachability condition $\Diamond S$, as limit-sure and almost-sure reachability coincide for MDPs [dAHK07].

(iii) *Sure winning.* From q_{init} there exists a strategy α and $N \in \mathbb{N}$ such that for all $n \geq N$, we have $\mathcal{M}_n^\alpha(T) = 1$. Hence α is sure winning for the reachability condition $\Diamond \mathsf{Supp}(\mathcal{M}_N^\alpha)$, and from all states in $\mathsf{Supp}(\mathcal{M}_N^\alpha)$ the strategy α ensures that only states in T are visited. It follows that $\mathsf{Supp}(\mathcal{M}_N^\alpha) \subseteq S$ is sure winning for the safety condition $\Box T$, and thus α is sure winning for the reachability condition $\Diamond S$ from q_{init}.

For the converse direction of the lemma, assume that a state q_{init} is sure (resp., almost-sure or limit-sure) winning for the reachability condition $\Diamond S$. We construct a winning strategy for strong synchronization in T as follows: play according to a sure (resp., almost-sure or limit-sure) winning strategy for the reachability condition $\Diamond S$, and whenever a state of S is reached along the play, then switch to a winning strategy for the safety condition $\Box T$. The constructed strategy is sure (resp., almost-sure or limit-sure) winning for strong synchronization according to sum_T because for sure winning, after finitely many steps all paths from q_{init} end up in $S \subseteq T$ and stay in S forever, and for almost-sure (or equivalently limit-sure) winning, for all $\epsilon > 0$, after sufficiently many steps, the set S is reached with probability at least $1 - \epsilon$, showing that the outcome is strongly $(1 - \epsilon)$-synchronizing in $S \subseteq T$, thus the strategy is almost-sure (and also limit-sure) strongly synchronizing.

□

Corollary 6.2. *For an MDP $\mathcal{M}$ and all target sets T, we have $\langle\langle 1 \rangle\rangle_{limit}^{strongly}(sum_T) = \langle\langle 1 \rangle\rangle_{almost}^{strongly}(sum_T)$.*

The following result follows from Lemma 6.5 and the fact that the winning region for sure safety, sure reachability and almost-sure reachability can be computed in PTIME for MDPs [dAHK07]. Moreover, memoryless strategies are sufficient for these conditions.

	Strongly	
	max	*sum*
Sure	PSPACE-complete	PSPACE-complete
Almost-sure	PSPACE-complete	PSPACE-complete
Limit-sure	undecidable	

Table 6.2: Computational complexity of the emptiness problem of strongly synchronizing languages in PAs.

Theorem 6.2. *For the three winning modes of strong synchronization according to sum_T in MDPs:*

1. *(Complexity). The membership problem is* PTIME-complete.
2. *(Memory). Pure memoryless strategies are sufficient.*

6.2 Strong synchronization in PAs

The same reduction mentioned to reduce the membership problem of strong synchronization with function max_T to membership problem with function max_Q in MDPs (see the beginning of Section 6.1), is valid for the emptiness problem of strongly synchronizing languages according to *max* in PAs. In the sequel, we thus study the emptiness problem of strongly synchronizing languages only according to max_Q. The results presented in this section are summarized in Table 6.2.

6.2.1 Sure strong synchronization with function *max*

In this subsection, we prove that the emptiness problem of sure strongly synchronizing languages with function max_Q is PSPACE-complete. The PSPACE upper bound of the problem is obtained by the following intuitive characterization.

Lemma 6.6. *For a PA $\mathcal{P}$, we have $\mathcal{L}_{sure}^{strongly}(max_Q) \neq \emptyset$ from an initial state q_{init} if and only if there exists some state q such that*

- *$\mathcal{L}_{sure}^{event}(q) \neq \emptyset$ from q_{init},*
- *and $\mathcal{L}_{sure}^{always}(max_Q) \neq \emptyset$ from q.*

Proof. First, we assume that the sure strongly synchronizing language $\mathcal{L}_{sure}^{strongly}(max_Q)$ for the PA $\mathcal{P}$ is not empty. By definition, there exists a word w and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, the probability mass $\|\mathcal{P}_n^w\| = 1$ is 1 at every step n. For all steps $n \geq n_0$, let $\hat{q}_n$ be the state in which the probability mass is accumulated: $\mathcal{P}_n^w(\hat{q}_n) = 1$. So we see

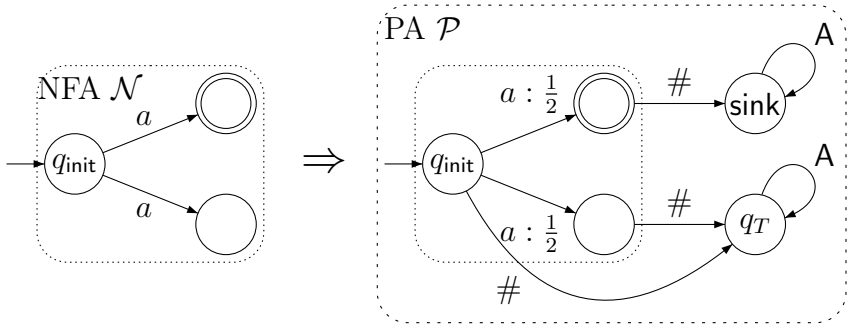

Figure 6.3: The reduction sketch to show PSPACE-hardness of the emptiness problem for sure strongly synchronizing languages in PAs.

that $\mathcal{L}_{sure}^{event}(q_{n_0}) \neq \emptyset$ from q_{init}. On the other hand, let $v = a_{n_0}a_{n_0+1}\cdots$ be the subword of $w = a_0a_1\cdots$ obtained by cutting the first n_0 letters. Starting in q_{n_0}, we see that for all $i \in \mathbb{N}$ there is always a unique state $\hat{q}_{n_0+i}$ such that $\mathcal{P}_i^v(\hat{q}_{n_0+i}) = 1$, and thus $\mathcal{L}_{sure}^{always}(max_Q) \neq \emptyset$ from q_{n_0}.

Second, assume that there exists some state q such that $\mathcal{L}_{sure}^{event}(q) \neq \emptyset$ from q_{init} and $\mathcal{L}_{sure}^{always}(max_Q) \neq \emptyset$ from q. Let w be the word that is sure eventually synchronizing in $\{q\}$ from q_{init}, and n be such that $\mathcal{P}_n^w(q) = 1$. Take the word $v = a_0a_1\cdots a_{n-1}$ that is the prefix of $w = a_0a_1\cdots$ consisting of the first n letters. Let v' be the word that is sure always synchronizing according to max_Q from q. By definition, for all $i \in \mathbb{N}$ there is always a unique state $\hat{q}_i \in Q$ such that $\mathcal{P}_i^{v'}(\hat{q}_i) = 1$. Concatenating the infinite word v' at the end of v, we have $\mathcal{P}_{n+i}^{v\cdot v'}(\hat{q}_i) = 1$ for all $i \geq n$ meaning that $\mathcal{L}_{sure}^{strongly}(max_Q) \neq \emptyset$ from the initial state q_{init}.

□

As we show in Theorem 4.6, the emptiness problem for sure eventually synchronizing languages in PAs is in PSPACE and as we show in Theorem 4.2, the emptiness problem for sure always synchronizing languages with function max is in PTIME, it follows from the characterization in Lemma 6.6 that the membership problem for sure strongly synchronizing language is in PSPACE, using the following (N)PSPACE algorithm: guess the state q, and check that $\mathcal{L}_{sure}^{event}(q) \neq \emptyset$ from q_{init} and $\mathcal{L}_{sure}^{always}(max_Q) \neq \emptyset$ from q. Lemma 6.7 presents a matching lower bound by a reduction from the non-universality problem in NFAs. As an alternative to the reduction from the non-universality problem in NFAs, we could use the same reduction from the finite automata intersection problem established to prove PSPACE-hardness of the emptiness problem of sure eventually synchronizing languages in Lemma 4.11.

Lemma 6.7. *The emptiness problem of $\mathcal{L}_{sure}^{strongly}(max_T)$ in PAs is* PSPACE-hard *even if T is a singleton.*

Proof. We present a proof using a reduction from the universality problem for NFAs that is known to be PSPACE-complete [IRS76]. Given a NFA $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$ equipped with an initial state q_{init} and a set $\mathcal{F}$ of accepting states, we construct a PA $\mathcal{P}$ and a singleton target set T such that the language of $\mathcal{N}$ is universal if and only if $\mathcal{L}_{sure}^{strongly}(max_T) = \emptyset$ from q_{init}. The reduction is illustrated in Figure 6.3.

The non-deterministic transitions of $\mathcal{N}$ become probabilistic in $\mathcal{P}$ with a uniform distribution over successors. The target set is the absorbing state $T = \{q_T\}$, a singleton. Therefore, a sure strongly synchronizing word needs to inject all that probability into q_T. This can be done with the new letter # from the non-accepting states of the NFA. From the accepting states, the #-transitions lead to a sink state sink from which there is no way to synchronize into q_T

The construction of the PA $\mathcal{P} = \langle Q', \mathsf{A}', \delta \rangle$ is such that the state space $Q' = Q \cup \{q_T, \mathsf{sink}\}$ has two new states which are only accessible with a new letter #, and $\mathsf{A}' = \mathsf{A} \cup \{\#\}$. For all $q \in Q$ and $a \in \mathsf{A}$, the probabilistic transition function $\delta(q,a)(q') = \frac{1}{|\Delta(q,a)|}$ if $q' \in \Delta(q,a)$, and $\delta(q,a)(q') = 0$ otherwise. For all $a \in \mathsf{A} \cup \{\#\}$, the a-transition in both states q_T and sink are self-loops meaning that both these states are absorbing. The #-transitions in all non-accepting states $q \in Q \setminus \mathcal{F}$ bring the PA deterministically into q_T, while the #-transitions in all accepting states $q \in \mathcal{F}$ is redirected to sink.

To establish the correctness of the reduction, we first assume that the language of $\mathcal{N}$ is not universal. Thus, there exists some finite word w such that all runs of $\mathcal{N}$ over w end in a non-accepting state, and inputting $w \cdot \#$ to $\mathcal{P}$ the state q_T is reached with probability 1. Therefore, $w \cdot (\#)^\omega$ is a sure strongly synchronizing word in $\{q_T\}$ for $\mathcal{P}$ from q_{init}.

Second, assume that there exists some infinite word v such that $v \in \mathcal{L}_{sure}^{strongly}(q_T)$ from q_{init} in $\mathcal{P}$. Let $\mathcal{P}_0^v \mathcal{P}_1^v \mathcal{P}_2^v \cdots$ be the symbolic run of $\mathcal{P}$ over v and let n_0 be such that $\mathcal{P}_n^v(q_T) = 1$ for all $n > n_0$. The word v must contain #, as this is the only transition to the target state q_T. Let $w \in \mathsf{A}^*$ be the prefix of v before the first occurrence of #. We claim that w is not accepted by the NFA $\mathcal{N}$. By contradiction, if there is an accepting run ρ of $\mathcal{N}$ over w, then positive probability is injected in sink by the finite word $w \cdot \#$ and stays there forever, in contradiction with the fact that $\mathcal{P}_n^v(q_T) = 1$ for all $n > n_0$. Therefore w is not accepted by the NFA $\mathcal{N}$, and the language of the NFA is not universal.

□

From previous Lemmas and argument, Theorem 6.3 follows.

Theorem 6.3. *The emptiness problem for sure strongly synchronizing languages with function max_T in PAs is* PSPACE-complete.

6.2.2 Almost-sure strong synchronization with function max

We present a construction to reduce the emptiness problem for almost-sure strongly synchronizing languages according to max_Q in PAs to the emptiness problem for ω-automata

with Büchi condition. The construction is exponential; however thanks to NLOGSPACE-completeness of the emptiness problem for Büchi automata, PSPACE upper bound follows. We also prove a matching lower bound.

Lemma 6.8. *The emptiness problem of almost-sure strongly synchronization with function max_Q in PAs, is in* PSPACE*.*

Proof. Given a PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ with an initial state q_{init}, we construct an automaton $\mathcal{N}$ equipped with an initial state ℓ_0 and a Büchi condition $\Box\Diamond\mathcal{F}$ such that $\mathcal{L}^{strongly}_{almost}(max_Q) = \emptyset$ from q_{init} if and only if $\mathcal{L}_{\Box\Diamond\mathcal{F}}(\mathcal{N}) = \emptyset$. The automaton $\mathcal{N}$ is exponential in the size of $\mathcal{P}$, and thus the PSPACE bound follows from the NLOGSPACE-completeness of the emptiness problem for Büchi automata.

The construction of $\mathcal{N}$ relies on the following intuition. The symbolic run of the PA $\mathcal{P}$ under an almost-sure strongly synchronizing word w is the sequence $\mathcal{P}^w_0\mathcal{P}^w_1\mathcal{P}^w_2\cdots$ of probability distributions $\mathcal{P}^w_i$ ($i \in \mathbb{N}$) in which the probability mass tends to accumulate in a single state $\hat{q}_i$ at step i. We prove that for all sufficiently large i, there exists a deterministic transition from $\hat{q}_i$ to $\hat{q}_{i+1}$ meaning that there exists some $a \in \mathsf{A}$ such that $\mathsf{post}(\hat{q}_i, a) = \{\hat{q}_{i+1}\}$ in the PA $\mathcal{P}$. The Büchi automaton $\mathcal{N}$ will guess the *witness sequence* $\hat{q}_i\hat{q}_{i+1}\ldots$ and check that the probability mass is *injected* into this sequence. The automaton $\mathcal{N}$ keeps track of the support $s_i = \mathsf{Supp}(\mathcal{P}^w_i)$ of the symbolic run, and at some point guesses that the witness sequence $\hat{q}_i\hat{q}_{i+1}\ldots$ starts. Then, using an *obligation* set $o_i \subseteq s_i$, it checks that every state in s_i eventually *injects* some probability mass in the witness sequence. When the obligation set gets empty, it is recharged with the current support s_i.

We construct the Büchi automaton $\mathcal{N} = \langle L, \mathsf{A}, \Delta \rangle$ as follows. The set $L = 2^Q \cup (2^Q \times 2^Q \times Q)$ is the set of states. A state $s \subseteq Q$ is the support of the current probability distribution. A state $(s, o, \hat{q}) \in 2^Q \times 2^Q \times Q$ consists of the support s, the obligation set $o \subseteq s$, and a state $\hat{q} \in s$ of the witness sequence. The alphabet A is in common with the PA $\mathcal{P}$. The transition function $\Delta : L \times \Sigma \to 2^L$ is defined as follows.

- For all $s \in 2^Q$ and $a \in \mathsf{A}$, let $s' = \mathsf{post}(s, a)$, and define

$$\Delta(s, a) = \{s'\} \cup \{(s', s', \hat{q}) \mid \hat{q} \in s'\}.$$

  A transition in the state s which leads to a state $(s', s', \hat{q})$, guesses $\hat{q}$ as the initial state of the witness sequence.
- For all $(s, o, \hat{q}) \in 2^Q \times 2^Q \times Q$ and $a \in \mathsf{A}$, let $s' = \mathsf{post}(s, a)$. If $\mathsf{post}(\hat{q}, a)$ is not a singleton, then $\Delta((s, o, \hat{q}), a) = \emptyset$, otherwise let $\{\hat{q}'\} = \mathsf{post}(\hat{q}, a)$, and:
  - If $o \neq \emptyset$, then

$$\Delta((s, o, \hat{q}), a) = \{(s', o' \setminus \{\hat{q}'\}, \hat{q}') \mid \forall q \in o : o' \cap \mathsf{post}(q, a) \neq \emptyset\}.$$

    These transitions deterministically choose the next state $\hat{q}'$ of the witness sequence, and in addition, non-deterministically take care of paths from the obligation set o to the witness sequence. For this sake, the constraint $o' \cap \mathsf{post}(q, a) \neq \emptyset$ is required for all $q \in o$.

– If $o = \emptyset$, then $\Delta((s, o, \hat{q}), a) = \{(s', s', \hat{q}')\}$. This transition is to recharge the obligation set with the current support s' when it gets empty.

The automaton $\mathcal{N}$ is equipped with the initial state $\ell_0 = \{q_{\mathsf{init}}\}$ and with the Büchi condition $\Box\Diamond\mathcal{F}$ where $\mathcal{F} = \{(s, o, \hat{q}) \in 2^Q \times 2^Q \times Q \mid o = \emptyset\}$. is the set of states with an empty obligation set o.

To establish the correctness of the reduction, we first assume that $\mathcal{L}_{\Box\Diamond\mathcal{F}}(\mathcal{N}) \neq \emptyset$ and show that $\mathcal{L}^{strongly}_{almost}(max_Q) \neq \emptyset$ from q_{init}. Since the Büchi language of $\mathcal{N}$ is not empty, there exists some infinite word $w = a_0a_1a_2\cdots$ inducing an ultimately periodic run $\rho = r_0r_1\dots r_{n-1}(r_n\dots r_{m-1})^\omega$ over $\mathcal{N}$ such that $r_n \in \mathcal{F}$. Let $r_m = r_n$ and $\kappa = m - n$ be the size of the cycle from r_n to itself. From the construction of Büchi automaton, we know that all r_i (where $n \leq i \leq m$) are states of the form $(s_i, o_i, \hat{q}_i)$ where $s_i = \mathsf{Supp}(\mathcal{P}^w_i)$ is the support of the symbolic run at step i (and also at steps $i + (j \cdot \kappa)$ for all $j \in \mathbb{N}$). As a result, $\mathcal{P}^w_i(q) > 0$ for all states $q \in s_i$. Since $\hat{q}_n \in s_n$, the state $\hat{q}_n$ is reached at step n with some positive probability $p > 0$. Since for all $n \leq i < m$, the state $\hat{q}_{i+1}$ is the unique successor of $\hat{q}_i$ and a_i, the sequence $\hat{q}_n\hat{q}_{n+1}\dots\hat{q}_{m-1}\hat{q}_m$ forms a deterministic cycle in $\mathcal{P}$. Hence, $\mathcal{P}^w_{i+1}(\hat{q}_{i+1}) \geq \mathcal{P}^w_i(\hat{q}_i) \geq p$, and we will prove that $\liminf_{i\to\infty} \mathcal{P}^w_i(\hat{q}_i) = 1$.

Since $r_n \in \mathcal{F}$, this state is of the form $(s_n, \emptyset, \hat{q}_n)$ where the obligation set is empty. Consequently $r_{n+1} = (s_{n+1}, s_{n+1}, \hat{q}_{n+1})$ where the obligation set $o_{n+1} = s_{n+1}$ is recharged. Let $o'_i = o_i \cup \{\hat{q}_i\}$ for all $n \leq i \leq m$. By a backward induction, we show that there are paths going through each state of o_i and ending in $\hat{q}_m$. The base of induction holds because $o'_m = \{\hat{q}_m\}$ and $\mathsf{post}(q, a_{m-1}) \cap o'_m \neq \emptyset$ for all $q \in o_{m-1}$. The induction holds for all $n \leq i \leq m$ too, thanks to the fact that $\mathsf{post}(q, a_{i-1}) \cap o'_i \neq \emptyset$ for all $q \in o_{i-1}$. Therefore, the state $\hat{q}_m$ is reached from all states $q \in s_{n+1}$ within $m - n$ steps, with some positive probability that is at least η^κ where $\kappa = m - n$ and η is the smallest probability of taking a transition in $\mathcal{P}$ [1]. Recall that $\mathcal{P}^w_n(\hat{q}_n) = p$ and $\mathcal{P}^w_n(Q \setminus \{\hat{q}_n\}) \leq 1 - p$. We thus have

$$\mathcal{P}^w_m(Q \setminus \{\hat{q}_m\}) \leq (1 - p) \cdot (1 - \eta^\kappa).$$

Visiting the states in the periodic part of the run ρ for j times leads to

$$\mathcal{P}^w_{m+j\cdot\kappa}(Q \setminus \{\hat{q}_{m+j\cdot\kappa}\}) \leq (1 - p)(1 - \eta^\kappa)^j.$$

The probability outside the witness sequence then tends to 0 when $j \to \infty$, i.e., $\lim_{j\to\infty} \mathcal{P}^w_{m+j\cdot\kappa}(Q \setminus \{\hat{q}_{m+j\cdot\kappa}\}) = 0$ implying that $\lim_{i\to\infty} \|\mathcal{P}^w_i\| = 1$.

Second, assume that $w \in \mathcal{L}^{strongly}_{almost}(max_Q)$ from q_{init}. We assume that $w = a_0a_1a_2\dots$ is pure [2]. Let $\mathcal{P}^w_0\mathcal{P}^w_1\dots$ be the symbolic run of w over $\mathcal{P}$. Since w is almost-sure strongly synchronizing, then $\forall\epsilon > 0$ there exists $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$ there exists some state $\hat{q}$ such that $\mathcal{P}^w_n(\hat{q}) > 1 - \epsilon$. By assuming $\epsilon < \frac{1}{2}$, the state $\hat{q}$ is unique in each step

1. Formally, $\eta = \min_{q,q'\in Q, a\in\mathsf{A}}(\delta(q, a)(q'))$.

2. To generalize the results to consider randomized words, the alphabet of the Büchi automaton $\mathcal{N}$ could be defined 2^A where A is the alphabet of $\mathcal{P}$. An infinite word over 2^A is a sequence of sets of letters which can be viewed as the sequence of supports of a randomized word. For the sake of simplicity, we prove the results for pure words.

$n \geq n_0$; we therefore denote this state by $\hat{q}_n$. Moreover, it is easy to see that the state $\hat{q}_n$ is independent of ϵ.

We prove that for all $n \geq n_0$, the state $\hat{q}_{n+1}$ is the unique successor of q_n and a_n: $\mathsf{post}(\hat{q}_n, a_n) = \{\hat{q}_{n+1}\}$. Let $\epsilon < \frac{\eta}{1+\eta}$ where η is the smallest probability in the transitions of $\mathcal{P}$. Towards contradiction, assume that there exists some $n > n_0$ such that $\mathsf{post}(\hat{q}_n, a_n) \neq \{\hat{q}_{n+1}\}$; in other words, there exists another state $q \neq \hat{q}_{n+1}$ such that $\{q, \hat{q}_{n+1}\} \subseteq \mathsf{post}(\hat{q}_n, a_n)$. By next transition, the probability in q then is

$$\mathcal{P}^w_{n+1}(q) \geq \delta(\hat{q}_n, \sigma_n)(q) \cdot \mathcal{P}^w_n(\hat{q}_n) \geq \eta \cdot (1 - \epsilon).$$

The probability $\mathcal{P}^w_{n+1}(\hat{q}_{n+1})$ in the witness state $\hat{q}_{n+1}$ is thus at most $1 - \eta \cdot (1 - \epsilon)$. The computed upper bound for the probability in the witness state $\hat{q}_{n+1}$ and the lower bound coming from the definition of almost-sure strongly synchronizing gives

$$1 - \epsilon \leq \|\mathcal{P}^w_{n+1}\| \leq 1 - \eta \cdot (1 - \epsilon)$$

that results in $\epsilon \geq \frac{\eta}{1+\eta}$, a contradiction. Therefore for all $n \geq n_0$, the state $\hat{q}_{n+1}$ is the unique successor of $\hat{q}_n$ and a_n.

For a given state q and word $v \in \mathsf{A}^+$, we define $\mathsf{post}(q, v)$ as the natural extension of $\mathsf{post}(q, a)$ over words: $\mathsf{post}(q, v) = \mathsf{post}(\mathsf{post}(q, v'), a)$ where $v = v' \cdot a$ and $v' \in \mathsf{A}^+$. To complete the proof, we construct an accepting run ρ of $\mathcal{N}$ over w. For the first n_0 transitions, the run ρ visits s_i where $s_i = \mathsf{Supp}(\mathcal{P}^w_i)$ and $0 \leq i < n_0$. At step n_0, the run ρ visits $(s_{n_0}, s_{n_0}, \hat{q}_{n_0})$ where $\hat{q}_{n_0}$ is the unique state in which the mass of probability $\|\mathcal{P}^w_{n_0}\|$ is accumulated at step n_0. Below, we prove that for all $n \geq n_0$ and all states in $q \in s_n$, there exists some path $p_0 p_1 \cdots p_k$ with length k that is from q to the witness state $\hat{q}_{n+k}$ where $p_0 = q$ and $p_q = \hat{q}_{n+k}$. Towards contradiction, assume that for some $n \geq n_0$, there exists $q \in s_n$ from which there is no path to the witness sequence. Since $q \in s_n$, this state contains some strictly positive probability p at step n where $p = \mathcal{P}^w_n(q) > 0$. Since the probability mass accumulated in the witness sequence would be bounded by $1 - p$ for all next steps after n, a contradiction arises with the fact that w is almost-sure strongly synchronizing. As a result of above argument, for all $n \geq n_0$ and all states in $q \in s_n$, there exists some path $p_0 p_1 \cdots p_k$ that is from q to the witness sequence; let $k(q) = k$ denote the length of this path and let $p(q, i) = p_i$ denote the i-th state visited along this path. Let $K = \max_{q \in s_n} k(q)$ be the length of the longest paths starting from some state $q \in s_n$ to the witness sequence. The accepting run r after the state $(s_{n_0}, o_{n_0}, \hat{q}_{n_0})$ visits the following K states $(s_{n_0+1}, o_{n_0+1}, \hat{q}_{n_0+1}) \cdots (s_{n_0+K}, o_{n_0+K}, \hat{q}_{n_0+K})$ consecutively, where for all $0 < i \leq K$,

- the set $s_{n_0+i} = \mathsf{Supp}(\mathcal{P}^w_{n_0+i})$ is the support of the symbolic run at step i,
- the obligation set $o_{n_0+i} = O \setminus \{\hat{q}_{n_0+i}\}$ with $O = \bigcup_{q \in s_{n_0} : k(q) \geq i} \{p(q, i)\}$, and
- $\hat{q}_{n_0+i}$ is the witness state at step $n_0 + i$.

By the choice of obligation sets, we see that $o_{n_0+K} = \emptyset$ and $r_{n_0+K} \in \mathcal{F}$. The automaton $\mathcal{N}$, by next transition, deterministically, reset the obligation set with s_{n_0+K+1}. With a similar construction, the run r visits the accepting states infinitely often and w is accepting.

□

Lemma 6.9. *The emptiness problem of* $\mathcal{L}^{strongly}_{almost}(max_T)$ *in PAs is* PSPACE-hard *even if* T *is a singleton.*

Proof. The proof is by the same reduction from the non-universality problem for NFAs that is presented in Lemma 6.7 where from a NFA $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$ equipped with an initial state q_{init} and a set $\mathcal{F}$ of accepting states, we construct a PA $\mathcal{P}$ and a singleton target state q_T such that the language of $\mathcal{N}$ is universal if and only if $\mathcal{L}^{strongly}_{sure}(q_T) = \emptyset$ from q_{init}.

We now argue that in the constructed PA $\mathcal{P}$, the sure and almost-sure strongly synchronizing languages with function max_{q_T} are equal: $\mathcal{L}^{strongly}_{sure}(q_T) = \mathcal{L}^{strongly}_{almost}(q_T)$. By definition, we know that all sure strongly synchronizing words are almost-sure strongly synchronizing too. To complete the proof, we thus assume that w is an almost-sure strongly synchronizing word in $\{q_T\}$, and prove that $w \in \mathcal{L}^{strongly}_{sure}(q_T)$ too. Let $\epsilon > 0$. Since w is almost-sure strongly synchronizing, there exists some $n \in \mathbb{N}$ such that $\mathcal{P}^w_n(q_T) \geq 1-\epsilon$. All #-transitions in $\mathcal{P}$ are redirected to either sink or q_T: $\mathsf{post}(Q', \#) = \{\mathsf{sink}, q_T\}$. This and the fact that q_T is only reachable by #-transitions give that $\mathsf{Supp}(\mathcal{P}^w_n) \subseteq \{q_T, \mathsf{sink}\}$. Hence, the remaining probability mass at step n is accumulated in either of those states q_T, sink. On the other hand since sink is a self-loop, if $\mathcal{P}^w_n(\mathsf{sink}) > 0$ then $\mathcal{P}^w_i(\mathsf{sink}) > 0$ for all next steps $i > n$. It means that in the case sink contains some positive probability p at step n, the probability in q_T is bounded away from 1 for all next steps $i > n$: $\mathcal{P}^w_i(q_T) < 1 - p$, a contradiction with the fact that w is almost-sure strongly synchronizing in q_T. Hence, the probability in q_T at step n is 1, and we see that it remains 1 forever irrespective to the next inputs. It proves that w is a sure strongly synchronizing word in q_T too.

□

From the Lemma 6.8 and Lemma 6.9, we conclude Theorem 6.4.

Theorem 6.4. *The emptiness problem for almost-sure strongly synchronizing languages with function* max_T *in PAs is* PSPACE-complete.

In subsection 6.2.5, we provide undecidability result for the emptiness problem of limit-sure strong synchronization with both functions *max* and *sum*.

6.2.3 Sure strong synchronization with function *sum*

In the sequel, we prove that the emptiness problem of sure strongly synchronizing languages with function sum_T is PSPACE-complete. The PSPACE upper bound of the problem is obtained by Lemma 6.10. Since the hardness result in Lemma 6.7 is proved for singleton target set T, the result holds for both functions max_T and sum_T.

Lemma 6.10. *Let* $\mathcal{P}$ *be a PA and* T *be a target set. We have* $\mathcal{L}^{strongly}_{sure}(sum_T) \neq \emptyset$ *from an initial state* q_{init} *if and only if there exists some set* $S \subseteq T$ *such that*

– $\mathcal{L}^{event}_{sure}(sum_S) \neq \emptyset$ *from* q_{init},

– *and* $\mathcal{L}_{sure}^{always}(sum_T) \neq \emptyset$ *from an arbitrary distribution over* S.

Proof. First, we assume that the sure strongly synchronizing language $\mathcal{L}_{sure}^{strongly}(sum_T)$ for the PA $\mathcal{P}$ is not empty. By definition, there exists a word w and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, the probability mass $\mathcal{P}_n^w(T)$ is 1 at every step n. For all steps $n \geq n_0$, let $S_n \subseteq T$ be the set of states $q \in T$ with strictly positive probability: $S_n = \mathsf{Supp}(\mathcal{P}_n^w)$. Let $S = S_{n_0}$. Then $\mathcal{P}_{n_0}^w(S) = 1$ implying that $\mathcal{L}_{sure}^{event}(sum_S) \neq \emptyset$ from q_{init}. On the other hand, let $v = a_{n_0}a_{n_0+1}\cdots$ be the subword of $w = a_0a_1\cdots$ obtained by cutting the first n_0 letters. Starting in the distribution d where $d = \mathcal{P}_{n_0}^w$ with support S, we see that for all $i \in \mathbb{N}$ there is always a subset $S_{n_0+i} \subseteq T$ such that $\mathcal{P}_i^v(S_{n_0+i}) = 1$, and thus $\mathcal{L}_{sure}^{always}(sum_T) \neq \emptyset$ from d.

Second, assume that there exists some set $S \subseteq T$ such that $\mathcal{L}_{sure}^{event}(sum_S) \neq \emptyset$ from q_{init} and $\mathcal{L}_{sure}^{always}(sum_T) \neq \emptyset$ from some distribution d over S. Let w be the word that is sure eventually synchronizing in S from q_{init}, and n be such that $\mathcal{P}_n^w(S) = 1$. Let $d' = \mathcal{P}_n^w$ with support S. By Remark 8, we know that $d \in \mathcal{L}_{sure}^{always}(sum_T)$ if and only if $d' \in \mathcal{L}_{sure}^{always}(sum_T)$ Take the word $v = a_0a_1\cdots a_{n-1}$ that is the prefix of $w = a_0a_1\cdots$ consisting of the first n_0 letters. Let v' be the word that is sure always synchronizing in S from d'. By definition, for all $i \in \mathbb{N}$ there is always a set $S_i \subseteq T$ such that $\mathcal{P}_i^{v'}(S_i) = 1$. Concatenating the infinite word v' at the end of v, we have $\mathcal{P}_{n+i}^{v\cdot v'}(T) = 1$ for all $i \geq n$ meaning that $\mathcal{L}_{sure}^{strongly}(sum_T) \neq \emptyset$ from the initial state q_{init}.

□

As we show in Theorem 4.6 and Theorem 4.2 that the emptiness problems for both sure eventually synchronizing languages and sure always synchronizing languages with function *sum* are in PSPACE, it follows from the characterization in Lemma 6.10 that the membership problem for sure strongly synchronizing language is in PSPACE, using the following (N)PSPACE algorithm: guess the subset $S \subseteq T$, and check that $\mathcal{L}_{sure}^{event}(sum_S) \neq \emptyset$ from q_{init} and $\mathcal{L}_{sure}^{always}(sum_S) \neq \emptyset$ from the uniform distribution on S.

From previous Lemma and discussion, Theorem 6.5 follows.

Theorem 6.5. *The emptiness problem for sure strongly synchronizing languages with function* sum_T *in PAs is* PSPACE-complete.

6.2.4 Almost-sure strong synchronization with function *sum*

We present a reduction from the emptiness problem for almost-sure strongly synchronizing languages with function sum_T to the emptiness problem for Büchi automata. The construction is similar to Proof of Lemma 6.8 and it is exponential in the size of the PA. The (N)PSPACE membership thus follows from the NLOGSPACE-completeness of the emptiness problem for Büchi automata.

Lemma 6.11. *The emptiness problem of almost-sure strongly synchronization with function* sum_T *in PAs, is in* PSPACE.

Proof. Given a PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ with an initial state q_{init}, we construct an automaton $\mathcal{N}$ equipped with an initial state ℓ_0 and a Büchi condition $\Box\Diamond\mathcal{F}$ such that $\mathcal{L}^{strongly}_{almost}(sum_T) = \emptyset$ from q_{init} if and only if $\mathcal{L}_{\Box\Diamond\mathcal{F}}(\mathcal{N}) = \emptyset$. The automaton $\mathcal{N}$ is exponential in the size of $\mathcal{P}$; however thanks to the NLOGSPACE-completeness of the emptiness problem, the PSPACE membership follows.

The construction of $\mathcal{N}$ relies on the following intuition (similarly to Proof of Lemma 6.8 where the Büchi automaton guesses the single state $\hat{q}$ in which the mass of probability is accumulated, here the automaton guesses the set c of target states which contribute in accumulating the probability mass). The symbolic run of the PA $\mathcal{P}$ under an almost-sure strongly synchronizing word w is the sequence $\mathcal{P}_0^w \mathcal{P}_1^w \mathcal{P}_2^w \cdots$ of probability distributions $\mathcal{P}_i^w$ ($i \in \mathbb{N}$) in which the probability mass tends to accumulate in some subset $c_i \subseteq T$ of the target set T at step i. We prove that for all sufficiently large i, there exists some a-transition from c_i to c_{i+1} such that $c_{i+1} = \mathsf{post}(c_i, a)$ in the PA $\mathcal{P}$. The Büchi automaton $\mathcal{N}$ will guess the *witness sequence* $c_i c_{i+1} \ldots$ and check that the probability mass is *injected* into this sequence. The automaton $\mathcal{N}$ keeps track of the support $s_i = \mathsf{Supp}(\mathcal{P}_i^w)$ of the symbolic run, and at some point guesses that the witness sequence $c_i c_{i+1} \ldots$ starts. Then, using an *obligation* set $o_i \subseteq s_i$, it checks that every state in s_i eventually *injects* some probability mass in the witness sequence. When the obligation set gets empty, it is recharged with the current support s_i.

We construct the Büchi automaton $\mathcal{N} = \langle L, \mathsf{A}, \Delta \rangle$ as follows. The set $L = 2^Q \cup (2^Q \times 2^Q \times 2^Q)$ is the set of states. A state $s \subseteq Q$ is the support of the current probability distribution. A state $(s, o, c) \in 2^Q \times 2^Q \times 2^Q$ consists of the support s, the obligation set $o \subseteq s$, and the subset $c \subseteq s \cap T$ of the witness sequence. The alphabet A is in common with the PA $\mathcal{P}$. The transition function $\Delta : L \times \Sigma \to 2^L$ is defined as follows.

- For all $s \in 2^Q$ and $a \in \mathsf{A}$, let $s' = \mathsf{post}(s, a)$, and define
$$\Delta(s, a) = \{s'\} \cup \{(s', s', c) \mid c \subseteq T \cap s'\}.$$
A transition in the state s which leads to a state (s', s', c), guesses c as the initial state of the witness sequence.
- For all $(s, o, c) \in 2^Q \times 2^Q \times 2^Q$ and $a \in \mathsf{A}$, let $s' = \mathsf{post}(s, a)$. If $\mathsf{post}(c, a) \not\subseteq T$ is not included in T, then $\Delta((s, o, c), a) = \emptyset$, otherwise let $c' = \mathsf{post}(c, a)$, and:
  - If $o \neq \emptyset$, then
$$\Delta((s, o, c), a) = \{(s', o' \setminus c', c') \mid \forall q \in o : o' \cap \mathsf{post}(q, a) \neq \emptyset\}.$$
These transitions deterministically choose the next subset c' of the witness sequence, and in addition, non-deterministically take care of paths from the obligation set o to the witness sequence. For this sake, the constraint $o' \cap \mathsf{post}(q, a) \neq \emptyset$ is required for all $q \in o$.
  - If $o = \emptyset$, then $\Delta((s, o, c), a) = \{(s', s', c')\}$. This transition is to recharge the obligation set with the current support s' when it gets empty.

The automaton $\mathcal{N}$ is equipped with the initial state $\ell_0 = \{q_{\mathsf{init}}\}$ and with the Büchi condition $\Box\Diamond\mathcal{F}$ where $\mathcal{F} = \{(s, o, c) \in 2^Q \times 2^Q \times 2^Q \mid o = \emptyset\}$ is the set of states with an empty obligation set o.

To establish the correctness of the reduction, we first assume that $\mathcal{L}_{\Box\Diamond\mathcal{F}}(\mathcal{N}) \neq \emptyset$ and show that $\mathcal{L}^{strongly}_{almost}(sum_T) \neq \emptyset$ from q_{init}. Since the Büchi language of $\mathcal{N}$ is not empty, there exists some infinite word $w = a_0a_1a_2\cdots$ inducing an ultimately periodic run $\rho = r_0r_1\dots r_{n-1}(r_n\dots r_{m-1})^\omega$ over $\mathcal{N}$ such that $r_n \in \mathcal{F}$. Let $r_m = r_n$ and $\kappa = m - n$ be the size of the cycle from r_n to itself. From the construction of Büchi automaton, we know that all r_i (where $n \leq i \leq m$) are states of the form (s_i, o_i, c_i) where $s_i = \mathsf{Supp}(\mathcal{P}^w_i)$ is the support of the symbolic run at step i (and also at steps $i + (j \cdot \kappa)$ for all $j \in \mathbb{N}$). As a result, $\mathcal{P}^w_i(q) > 0$ for all states $q \in s_i$. Since $c_n \subseteq s_n$, all the states $q \in c_n$ are reached at step n with some positive probability $p > 0$. Since $c_{i+1} = \mathsf{post}(c_i, a_i)$, we have $\mathcal{P}^w_{i+1}(c_{i+1}) \geq \mathcal{P}^w_i(c_i) \geq p$, and we will prove that $\liminf_{i\to\infty} \mathcal{P}^w_i(c_i) = 1$.

Since $r_n \in \mathcal{F}$, this state is of the form $(s_n, \emptyset, c_n)$ where the obligation set is empty. Consequently $r_{n+1} = (s_{n+1}, s_{n+1}, c_{n+1})$ where the obligation set $o_{n+1} = s_{n+1}$ is recharged. Let $o'_i = o_i \cup c_i$ for all $n \leq i \leq m$. By a backward induction, we show that there are paths going through each state of o_i and ending in some state $q \in c_m$. The base of induction holds because $o'_m = c_m$ and $\mathsf{post}(q', a_{m-1}) \cap o'_m \neq \emptyset$ for all $q' \in o_{m-1}$. The induction holds for all $n \leq i \leq m$ too, thanks to the fact that $\mathsf{post}(q', a_{i-1}) \cap o'_i \neq \emptyset$ for all $q' \in o_{i-1}$. Therefore, from all states $q' \in s_{n+1}$ some state $q \in c_m$ is reached within $m - n$ steps, with some positive probability that is at least η^κ where $\kappa = m - n$ and η is the smallest probability of taking a transition in $\mathcal{P}$. Recall that $\mathcal{P}^w_n(c_n) = p$ and $\mathcal{P}^w_n(Q \setminus c_n) \leq 1 - p$. We thus have

$$\mathcal{P}^w_m(Q \setminus c_m) \leq (1 - p) \cdot (1 - \eta^\kappa).$$

Visiting the states in the periodic part of the run ρ for j times leads to

$$\mathcal{P}^w_{m+j\cdot\kappa}(Q \setminus c_{m+j\cdot\kappa}) \leq (1 - p)(1 - \eta^\kappa)^j.$$

The probability outside the witness sequence then tends to 0 when $j \to \infty$. That means $\lim_{j\to\infty} \mathcal{P}^w_{m+j\cdot\kappa}(Q \setminus c_{m+j\cdot\kappa}) = 0$ implying that $\lim_{i\to\infty} \mathcal{P}^w_i(T) = 1$.

Second, assume that the almost-sure strongly synchronizing language with function sum_T of $\mathcal{P}$ is not empty. Let S be the set of initial states q such that $\mathcal{L}^{always}_{sure}(sum_T) \neq \emptyset$. We know that the set S is equivalently the set of all initial states from which the ω-language for the safety condition $\Box T$ in $\mathcal{P}$ is non-empty. Let $w \in \mathcal{L}^{strongly}_{almost}(sum_T)$ from q_{init}. We assume that $w = a_0a_1a_2\dots$ is pure[3]. Let $\mathcal{P}^w_0\mathcal{P}^w_1\dots$ be the symbolic run of w over $\mathcal{P}$. Consider $\epsilon > \frac{1}{2}$ and let $\epsilon' = \frac{\epsilon}{|Q|} \cdot \eta^{|Q|}$ where η is the smallest positive probability in the transitions of $\mathcal{P}$. By definition, there exists $N \in \mathbb{N}$ such that for all $n \geq N$, we have $\mathcal{P}^w_n(T) \geq 1 - \epsilon'$. We claim that at step N, all non-safe states $q \in Q \setminus S$ have probability at most $\frac{\epsilon}{|Q|}$, that is $\mathcal{P}^w_N(q) \leq \frac{\epsilon}{|Q|}$ for all $q \in Q \setminus S$. Towards contradiction, assume that $\mathcal{P}^w_N(q) > \frac{\epsilon}{|Q|}$ for some non-safe state $q \in Q \setminus S$. Since $q \notin S$ is not safe, there is a path of length $\ell \leq |Q|$ from q to a state in $Q \setminus T$, thus with probability at least $\eta^{|Q|}$. It follows that after $N + \ell$ steps we have $\mathcal{P}^w_{N+\ell}(Q \setminus T) > \frac{\epsilon}{|Q|} \cdot \eta^{|Q|} = \epsilon'$, in contradiction with the fact

3. To generalize the results to consider randomized words, the alphabet of the Büchi automaton $\mathcal{N}$ could be defined 2^{A} where A is the alphabet of $\mathcal{P}$. An infinite word over 2^{A} is a sequence of sets of letters which can be viewed as the sequence of supports of a randomized word. For the sake of simplicity, we prove the results for pure words.

$\mathcal{P}_n^w(T) \geq 1-\epsilon'$ for all $n \geq N$. Now, since all non-safe states have probability at most $\frac{\epsilon}{|Q|}$ at step N, it follows that $\mathcal{P}_N^w(Q \setminus S) \leq \frac{\epsilon}{|Q|} \cdot |Q| = \epsilon$. Thus $\mathcal{P}_N^w(S) \geq 1-\epsilon$, and $\mathcal{P}_n^w(S) \geq 1-\epsilon$ for all $n \geq N$.

For a given state q and word $v \in \mathsf{A}^+$, we define $\mathsf{post}(q, v)$ as the natural extension of $\mathsf{post}(q, a)$ over words: $\mathsf{post}(q, v) = \mathsf{post}(\mathsf{post}(q, v'), a)$ where $v = v' \cdot a$ and $v' \in \mathsf{A}^+$. To complete the proof, we construct an accepting run ρ of $\mathcal{N}$ over w. For the first N transitions, the run ρ visits s_i where $s_i = \mathsf{Supp}(\mathcal{P}_i^w)$ and $0 \leq i < N$. At step N, the run ρ visits (s_N, s_N, S). Let $c_N c_{N+1} c_{N+2} \cdots$ be the witness sequence where $c_N = S$ and $c_{N+i} = \mathsf{post}(S, v)$ where $v = a_N \cdots a_{N+i-1}$ is the subword of $w = a_0 \cdot a_1 \cdots$ from the N-th letter up to $(N+i)$-th letter. Below, we prove that for all $n \geq N$ and all states in $q \in s_n$, there exists some path $p_0 p_1 \cdots p_k$ with length k that is from q to the witness state c_{n+k} where $\ell_0 = q$ and $\ell_q \in c_{n+k}$. Towards contradiction, assume that for some $n \geq n_0$, there exists $q \in s_n$ from which there is no path to the witness sequence. Since $q \in s_n$, this state contains some strictly positive probability p at step n where $p = \mathcal{P}_n^w(q) > 0$. Since the probability mass accumulated in the witness sequence would be bounded by $1-p$ for all next steps after n, a contradiction arises with the fact that w is almost-sure strongly synchronizing. As a result of above argument, for all $n \geq n_0$ and all states in $q \in s_n$, there exists some path $p_0 p_1 \cdots p_k$ that is from q to the witness sequence; let $k(q) = k$ denote the length of this path and let $p(q, i) = p_i$ denote the i-th state visited along this path. Let $K = \max_{q \in s_n} k(q)$ be the length of the longest paths starting from some state $q \in s_n$ to the witness sequence. The accepting run r after the state (s_N, o_N, c_N) visits the following K states $(s_{N+1}, o_{N+1}, c_{N+1}) \cdots (s_{N+K}, o_{N+K}, c_{N+K})$ consecutively, where for all $0 < i \leq K$,

- the set $s_{N+i} = \mathsf{Supp}(\mathcal{P}_{N+i}^w)$ is the support of the symbolic run at step i,
- the obligation set $o_{n_0+i} = O \setminus \{\hat{q}_{N+i}\}$ with $O = \bigcup_{q \in s_N : k(q) \geq i} \{p(q, i)\}$, and
- c_{N+i} is the witness state at step $N+i$.

By the choice of obligation sets, we see that $o_{N+K} = \emptyset$ and $r_{N+K} \in \mathcal{F}$. The automaton $\mathcal{N}$, by next transition, deterministically, reset the obligation set with s_{N+K+1}. With a similar construction, the run r visits the accepting states infinitely often and w is accepting.

□

Lemma 6.11 gives the (N)PSPACE upper bound of the emptiness problem for almost-sure strongly synchronizing languages with function sum_T, the matching lower bound is an immediate result of Lemma 6.9. Thus, Theorem 6.6 follows.

Theorem 6.6. *The emptiness problem for almost-sure strongly synchronizing languages with function sum_T in PAs is* PSPACE-complete.

6.2.5 Limit-sure strong synchronization

We prove that the emptiness problems for limit-sure strongly synchronizing languages according to both functions max_T and sum_T are undecidable, by a similar construction used in [GO10] to establish the undecidability result for the value 1 problem in PAs.

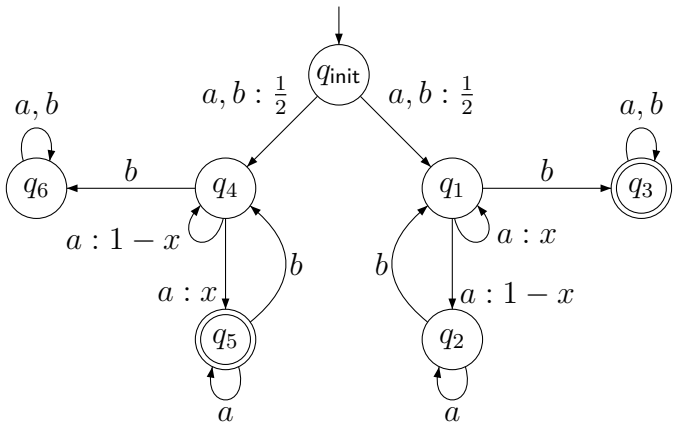

Figure 6.4: An automaton presented in [GO10], that has value 1 if and only if $x > \frac{1}{2}$.

Recall that, given a PA $\mathcal{N}$ and a threshold $0 < \lambda < 1$, the strict emptiness problem is undecidable: this problem asks whether the language $\mathcal{L}^{\lambda}(\mathcal{N}) = \{w \in \mathsf{A}^* \mid \mathsf{Pr}(w) > \lambda\}$ is empty. To prove undecidability of value 1 problem, a PA $\mathcal{P}$ is presented where the probability of some transitions are parametrized with x; and it is proved that $\mathcal{P}$ has value 1 if and only if $x > \frac{1}{2}$; see Example 6.2. The undecidability of value 1 problem follows from combining $\mathcal{P}$ and another PA $\mathcal{N}$ such that $x > \frac{1}{2}$ if and only if $\mathcal{L}^{\lambda}(\mathcal{N}) = \emptyset$ with $\lambda = \frac{1}{2}$. We refer the reader to [GO10] for details.

Example 6.2. *Let $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ be the PA depicted in Figure 6.4 where $\mathsf{A} = \{a, b\}$ and $Q = \{q_{\mathsf{init}}, q_1, \cdots, q_6\}$. On all letters, the successors of q_{init} are q_1 and q_4 each with probability $\frac{1}{2}$. Two states q_1 and q_4 are in some sense symmetric. The a-transitions have two successors one with probability x and another with probability $1 - x$: in q_1, the next successor is q_1 with probability x and is q_2 with probability $1 - x$ whereas in q_4, the next successor is itself with probability $1 - x$ and is q_5 with probability x. The b-transitions are redirected to q_3 and q_6, respectively. Two states q_2 and q_5 are also somehow symmetric: the a-transitions are self-loops while the b-transitions are deterministically redirected to q_1 and q_4, accordingly. Two states q_3 and q_6 are symmetrically absorbing states.*

Starting in q_1 the probability of the word $a^n \cdot b$ to be accepted in $\{q_3\}$ is x^n, and similarly starting in q_4 the probability of the word $a^n \cdot b$ to be accepted in $\{q_6\}$ is $(1 - x)^n$. By construction of the automaton, all accepting words are in the shape $w = (a + b) \cdot (a + b) \cdot (a^* \cdot b)^* \cdot a^*$. For a sequence $n_0 n_1 n_2 \cdots$ and for all $i \in \mathbb{N}$, let $w_i = a^{n_0} \cdot b \cdot a^{n_1} \cdot b \cdots a^{n_i}$ be a finite word. Starting in q_1 the probability of the word w_i to be accepted in $\{q_3\}$ is

$$(1 - \prod_{0 \leq j < i} (1 - x^{n_j})).$$

Starting in q_4 the probability of the word w_i to be accepted in $\{q_6\}$ is

$$(1 - \prod_{0 \leq j \leq i} (1 - (1 - x)^{n_j}) < \sum_{0 \leq j \leq i} (1 - x^{n_j}).$$

Let $\mathcal{F} = \{q_3, q_5\}$ be the accepting states and q_{init} be the initial state. We see that after the first input letter, two states q_1 and q_4 both contains probability $\frac{1}{2}$. Starting from q_{init}, the probability of the words $v_i \in (a+b)\cdot w_i$ to be accepted in $\mathcal{F}$ thus is

$$\mathsf{Pr}(v_i) = \frac{1}{2}(1 - \prod_{0\leq j<i}(1-x^{n_j})) + \frac{1}{2}(\prod_{0\leq j\leq i}(1-(1-x)^{n_j})).$$

There are two cases:

(i.) *Let $x \leq \frac{1}{2}$. By the above computation, considering the accepting set $\{q_3\}$, after inputting a word the probability in q_3 is less than the probability in q_6. As a result, the probability of accepting all words is less than $\frac{1}{2}$.*

(ii.) *Let $x > \frac{1}{2}$. It is proved that for all $\epsilon > 0$ there exists some increasing sequence $n_0 < n_1 < n_2 < \cdots$ such that*

$$\prod_{0\leq j<i}(1-x^{n_j}) \to 0 \text{ and } \sum_{0\leq j\leq i}(1-x^{n_j}) < \epsilon$$

when $i \to \infty$. It means that there is a careful way of choosing the sequence $n_1 n_2 n_3 \cdots$ where after reading each subword $a^*\cdot b$ of v_i (for sufficiently large i) q_3 contains larger probability than what is lost in q_6. Hence, the probability of the words v_i to be accepted in $\mathcal{F}$ is at least $1-\epsilon$.

It proves that $\mathcal{P}$ has value 1 if and only if $x > \frac{1}{2}$.

This example is presented in [GO10] to establish the undecidability result of the value 1 problem.

◁

We briefly recall the reduction from the strict emptiness problem to the value 1 problem [GO10].

Remark 9. The value 1 problem in PAs is undecidable.

Proof (sketch). Let $\mathcal{P} = \langle Q, \mathsf{A}, \delta\rangle$ be the PA described in Example 6.2 and let $\mathcal{N}$ be a PA whose alphabet A' has no intersection with A. The automaton $\mathcal{N}$ is equipped with an initial state and the set $\mathcal{F}$ of accepting states.

Let $\mathcal{N}_1 = \langle Q_1, \mathsf{A}', \delta_1\rangle$ and $\mathcal{N}_4 = \langle Q_4, \mathsf{A}', \delta_4\rangle$ be two copies of $\mathcal{N}$. From $\mathcal{N}_1$, $\mathcal{N}_4$ and $\mathcal{P}$ we construct another PA $\mathcal{B} = \langle Q_{\mathcal{B}}, \mathsf{A}_{\mathcal{B}}, \delta_{\mathcal{B}}\rangle$ such that $\mathcal{L}^{\frac{1}{2}}(\mathcal{N}) = \emptyset$ if and only if $\mathcal{B}$ has value 1. The sketch of constriction is as follows.

- Let the alphabet of $\mathcal{B}$ be $\mathsf{A}_{\mathcal{B}} = \mathsf{A}' \cup \{b, \star\}$.
- Copy all states of $\mathcal{P}$ into $\mathcal{B}$. Add all b-transitions in $\mathcal{P}$ to $\mathcal{B}$.
- Copy all states of $\mathcal{N}_1$ and $\mathcal{N}_4$ into $\mathcal{B}$ such that the initial state of $\mathcal{N}_1$ is the (copy of) state q_1 of $\mathcal{P}$ and the initial state of $\mathcal{N}_4$ is the (copy of) state q_4 of $\mathcal{P}$. For all $a \in \mathsf{A}'$, add all a-transitions in $\mathcal{N}_1$ and $\mathcal{N}_4$ to $\mathcal{B}$.
- For all $a \in \mathsf{A}'$, add a self-loop a-transition in all states $q \in \{q_{\mathsf{init}}, q_2, q_3, q_5, q_6\}$ to $\mathcal{B}$.

– From all copies of states $q \in Q_1 \cup Q_4$ of $\mathcal{N}_1$ and $\mathcal{N}_4$ except initial states q_1 and q_4, add a b-transition directed to the absorbing state q_6.
– For all copies of states $q \in Q_1$ of $\mathcal{N}_1$, the $\star$-transition goes to the (initial) state q_1 if q is an accepting state for $\mathcal{N}_1$, and goes to the state q_2 otherwise.
– For all copies of states $q \in Q_4$ of $\mathcal{N}_4$, the $\star$-transition goes to the state q_5 if q is an accepting state for $\mathcal{N}_4$, and goes to the (initial) state q_4 otherwise.

To establish the correctness of reduction. Assume that there exists some word w for $\mathcal{N}$ such that $\mathsf{Pr}(w) > \frac{1}{2}$. To prove that $\mathcal{B}$ has value 1, let $\epsilon > 0$ and let $v = b \cdot a^{n_0} \cdot b \cdot a^{n_2} \cdots a^{n_i} \cdot b$ be a word that is accepted by $\mathcal{P}$ with probability $1 - \epsilon$, see Example 6.2. By construction of $\mathcal{B}$, one can verify that the word $b \cdot (w \cdot \star)^{n_0} \cdot b \cdot (w \cdot \star)^{n_2} \cdots (w \cdot \star)^{n_i} \cdot b$ is accepted by $\mathcal{B}$ with probability $1 - \epsilon$. Thus, $\mathcal{B}$ has value 1.

Second, assume that $\mathcal{B}$ has value 1. Towards contradiction, assume that for all words w the probability to be accepted by $\mathcal{N}$ is equal or less than one half: $\mathsf{Pr}(w) \leq \frac{1}{2}$. Let $w' \in \mathsf{A}' \cup \{b, \star\}$ be an arbitrarily finite accepting word for $\mathcal{B}$. By construction of $\mathcal{B}$, the word w' can be written as $u_0 \cdot v_0 \cdot \star \cdot u_1 \cdot v_1 \cdot \star \cdots$ where $u_i \in b^*$ and $v_i \in \mathsf{A}'^*$. By construction of $\mathcal{B}$, the automaton $\mathcal{B}$ accepts w with the probability less than the acceptance probability of $w = u_0 \cdot a \cdot u_1 \cdot a \cdots$ by $\mathcal{P}$ where $x \leq \frac{1}{2}$, that is less than $\frac{1}{2}$. It gives a contradiction with the fact that $\mathcal{B}$ has value 1.

□

Theorem 6.7 provides the undecidability result of the emptiness problems for limit-sure strongly synchronizing languages with both functions max_T and sum_T.

Theorem 6.7. *For all functions $f \in \{max_T, sum_T\}$, the emptiness problem of limit-sure strongly synchronizing language in PAs is undecidable.*

Proof. The proof is by a similar construction used in [GO10] to establish the undecidability result for the value 1 problem in PAs. Consider the PA $\mathcal{P}$ described in Example 6.2. From the PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$, we construct another PA $\mathcal{P}'$ and a target set q_T such that $\mathcal{P}$ has value 1 if and only if $\mathcal{P}'$ is limit-sure strongly synchronizing in $\{q_T\}$.

The construction of $\mathcal{P}' = \langle Q', \mathsf{A}', \delta' \rangle$ is as follows. The PA $\mathcal{P}'$ is a copy of $\mathcal{P}$ where two new absorbing states q_T and sink are added that are only accessible by a new letter $\#$. Formally, $Q' = Q \cup \{q_T, \mathsf{sink}\}$ and $\mathsf{A}' = \mathsf{A} \cup \{\#\}$. All transitions in $\mathcal{P}$ are copied to $\mathcal{P}'$, so $\delta'(q, a) = \delta(q, a)$ for all $q \in Q$ and $a \in \mathsf{A}$. The following transitions are added: all $\#$-transitions in accepting states q_3 and q_5 are directed to q_T; all $\#$-transitions in non-accepting states $q \in Q \setminus \{q_3, q_5\}$ are directed to sink. Both new states q_T and sink are absorbing.

Below, we prove that $\mathcal{P}$ has value 1 if and only if $\mathcal{P}'$ is limit-sure strongly synchronizing in $\{q_T\}$. First, assume that $\mathcal{P}$ has value 1. By definition, for all $\epsilon > 0$ there exists a word w such that $\mathsf{Pr}(w) \geq 1 - \epsilon$. Taking the $\#$-transition after such words w, we see that

$v \in w \cdot (\#)^\omega$ is $(1-\epsilon)$-synchronized in q_T. Thus, $\mathcal{P}'$ is limit-sure strongly synchronizing in $\{q_T\}$.

Now, assume that $\mathcal{L}_{sure}^{strongly}(q_T) \neq \emptyset$. Let $\epsilon > 0$, and let w be such that w is $(1-\epsilon)$-synchronized in q_T. Since the only in-going transitions to q_T are #-transitions, then w must have some #. Let v be the prefix of w from the beginning up to the first occurrence of #, so that v does not contain any # and is a valid word for $\mathcal{P}$.

Since $\mathsf{post}(Q', \#) = \{q_T, \mathsf{sink}\}$, after reading the prefix $v \cdot \#$ of w the only states assigned with some possibly positive probability are sink and q_T. Since both of these states are absorbing, then the assigned probabilities would never changed, no matter what is the next input letters. Thus, if w is $(1-\epsilon)$-synchronized in q_T, then v is $(1-\epsilon)$-synchronized in $\{q_3, q_5\}$ implying that $\mathsf{Pr}(v) \geq 1-\epsilon$.

The above argument proves that $\mathcal{P}$ has value 1 if and only if $\mathcal{P}'$ is limit-sure strongly synchronizing in $\{q_T\}$. The same reduction, presented in Remark 9, from the strict emptiness problem to the value 1 problem is used to complete the proof.

□

6.3 Discussion

In this section, we briefly discuss the universality problems for sure and almost-sure strongly synchronizing languages in PAs.

Universality problems of sure strongly synchronizing language. Consider a PA $\mathcal{P}$ with the universal sure strongly synchronizing language according to max_T. By definition, for all words w the symbolic run $\mathcal{P}_0^w \mathcal{P}_1^w \mathcal{P}_2^w \cdots$ is ultimately always 1-synchronized in a singleton $\{q\}$ where $q \in T$, i.e., there exists some $n_0 \leq 2^{|Q|}$ (where Q is the state space of $\mathcal{P}$) such that for all $n > n_0$ there exists some state $q \in T$ where $\mathcal{P}_n^w(q) = 1$. We claim that for all $n \geq 2^{|Q|}$ there exists a unique state $\hat{q}_n$ such that for all words w we have $\mathcal{P}_n^w(\hat{q}_n) = 1$. Toward contradiction, assume that there exists n, two states q and q' and two words w and w' such that $\mathcal{P}_n^w(q) = 1$ and $\mathcal{P}_n^w(q') = 1$. Thus, the symbolic run of the automaton over the randomized word $v = \frac{1}{2}w + \frac{1}{2}w'$ is not 1-synchronized in any state; i.e., $\|\mathcal{P}_n^v\|_T \neq 1$ implying that v is not sure strongly synchronizing according to max_T, a contradiction with the fact that $\mathcal{P}$ has a universal sure strongly synchronizing language according to max_T. By above arguments, we see that to decide the universality problem of sure strongly synchronizing language according to max_T, one can check whether the graph underlying the PA has a unique bottom SCC consisting of a simple cycle, and whether the periodic supports of recurrent states in the Markov chain induced by the PA under the uniformly randomized word, are all singletons sets. This can be done in PTIME.

A similar argument for the function sum_T shows that to decide the universality problem of sure strongly synchronizing language, the sequence $s_0 s_1 s_2 \cdots$ where $s_0 = \{q_{\mathsf{init}}\}$ and $s_i = \mathsf{post}(s_{i-1}, \mathsf{A})$ for all $i \in \mathbb{N}$, can be computed and then one can check whether $s_i \subseteq T$ for all $n \leq i \leq n+m$, which can be done in PSPACE.

We do not provide a matching lower bound for the universality problem of sure strongly synchronizing languages according to the function sum_T.

Universality problems of almost-sure strongly synchronizing language. Given a PA $\mathcal{P}$ with the initial state q_{init} and the function max_T, Lemma 6.12 proves that the almost-sure strongly synchronizing language of $\mathcal{P}$ is universal if and only if (i) there is an absorbing end component U, and (ii) $\mathcal{P}$ is almost-sure strongly synchronizing by the uniformly randomized word w_u, which provides the PSPACE membership of this problem.

Lemma 6.12. *The universality problem of almost-sure strongly synchronizing languages according to max_T in PAs is in* PSPACE.

Proof. Given a PA $\mathcal{P} = \langle Q, \mathsf{A}, \delta \rangle$ with the initial state q_{init} and the function max_T, the almost-sure strongly synchronizing language is universal if and only if

(i) there is a (then necessarily unique) absorbing end component U, and

(ii) $\mathcal{P}$ is almost-sure strongly synchronizing by the uniformly randomized word w_u.

One direction is straightforward. For the reverse direction, assume that both of the conditions (i) and (ii) are fulfilled in the PA $\mathcal{P}$. We show that the almost-sure strongly synchronizing language (according to max_T) of $\mathcal{P}$ is universal. Let $U = \{q_0, q_1, \ldots, q_{|U|-1}\}$, for convenience also let $q_{|U|} = q_0$. We claim that $\mathsf{post}(q_i, \mathsf{A}) = \{q_{i+1}\}$ is singleton for all $0 \leq i < |U|$, which means U consists of a simple cycle. Let $\epsilon = \frac{\eta}{(1+\eta)}$ where η is the smallest probability of taking a transition in $\mathcal{P}$. Since w_u is almost-sure strongly synchronizing, there exists $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, there exists some state q where $\mathcal{P}_n^{w_u}(q) > 1 - \epsilon$. Let $n_1 \in \mathbb{N}$ be such that for all $n > n_1$, the total probability to be in states $Q \setminus U$ is smaller than ϵ: $\mathcal{P}_n^{w_u}(Q \setminus U) < \epsilon$. Then, for all $k > \max(n_0, n_1)$, there exists a unique state $q \in U$ such that $\mathcal{P}_k^{w_u}(q) \geq 1 - \epsilon$. Assume towards contradiction that q has two successors $q_1, q_2 \in U$ with $q_1 \neq q_2$, that is $\{q_1, q_2\} \subseteq \mathsf{post}(q, \mathsf{A})$. Then,

$$\mathcal{P}_{k+1}^{w_u}(q_1) \geq \mathcal{P}_k^{w_u}(q) \cdot \delta(q, a)(q_1) > (1 - \epsilon) \cdot \eta = \epsilon$$

where $q_1 \in \mathsf{post}(q, a)$. By a symmetric argument, we have $\mathcal{P}_{k+1}^{w_u}(q_2) > \epsilon$, showing that $\|\mathcal{P}_{k+1}^{w_u}\| < 1 - \epsilon$, a contradiction. This arguments proves that the absorbing end component U consists of a simple cycle.

We have shown that the absorbing end component $U = \{q_0, q_1, \ldots, q_{|U|-1}\}$ consists of a simple cycle. Towards contradiction with the universality of the almost-sure strongly synchronizing language of $\mathcal{P}$, assume that there exists an infinite word w that is not almost-sure strongly synchronizing according to max_T. Let $\mathcal{P}_0^w \mathcal{P}_1^w \cdots$ be the symbolic-run of $\mathcal{P}$ over w; and $\mathcal{P}_0^{w_u} \mathcal{P}_1^{w_u} \cdots$ be the symbolic-run over the uniformly randomized word w_u. Since U is absorbing and consists of a simple cycle, there exists $n \in \mathbb{N}$ such that for all $i > n$ the support $\mathsf{Supp}(\mathcal{P}_i^w)$ contains more than one state $\hat{q} \in U$, say two states q, q'; otherwise w would be almost-sure strongly synchronizing. By assumption, the uniformly randomized word w_u is almost-sure strongly synchronizing. All states reachable by w are also reachable by w_u: $\mathsf{Supp}(\mathcal{P}_i^w) \subseteq \mathsf{Supp}(\mathcal{P}_i^{w_u})$. Hence, for all $i > n$ there are two states q

and q' such that $\{q, q'\} \subseteq \mathsf{Supp}(\mathcal{P}_i^{w_u})$ too. Since the probabilities assigned to these state q and q' always loop through the sequence $q_0 q_1 \cdots q_{|U|}$, the probability mass could never accumulate in a singleton anymore, a contradiction with condition (ii) stating that w_u is almost-sure strongly synchronizing.

Condition (i) can be checked in PSPACE by Lemma 5.10, and condition (ii) can be checked in PTIME by steady state analysis of the Markov chain induced by the PA under the uniformly randomized word. The PSPACE bound follows.

□

We have proved that, given a PA $\mathcal{P}$ with the initial state q_{init} and the function max_T, the almost-sure strongly synchronizing language (according to max_T) of $\mathcal{P}$ is universal if and only if (i) there is an absorbing end component U, and (ii) $\mathcal{P}$ is almost-sure strongly synchronizing by the uniformly randomized word w_u. As we have shown in the proof of Lemma 6.12, these two conditions enforce a special shape to the absorbing end component U; the end component U consists of a simple cycle, and for the symbolic-run of the PA over the uniformly randomized word, all supports can only contain one of the states of U. One can observe that these conditions indeed imply that the periodic supports of recurrent states in the Markov chain induced by the PA under the uniformly randomized word, are all singletons sets.

For the function sum_T, a similar argument shows that the almost-sure strongly synchronizing language (according to sum_T) of $\mathcal{P}$ is decidable in PSPACE. The requirement of having an absorbing end component is not necessary for the function sum_T; in fact, states in all end components U can contribute in accumulating the probability mass in the target set T. Each end component U can be treated as an absorbing end component by removing all outgoing transitions, and the periodic supports of recurrent states $c_0, c_1, \cdots, c_n$ can be computed in PSPACE. From those supports, a randomized word w can be constructed such that it is a linear combinations from all pure words where the PA stays in U with some positive probability, once it has entered in U (by those pure words); and check whether $\mathcal{P}$ is almost-sure strongly synchronizing by w. After repeating these arguments for all end components U, the PA must be almost-sure strongly synchronizing by the uniformly randomized word w_u too.

The above arguments provided the PSPACE membership of the universality problem of almost-sure strongly synchronizing languages for both function sum_T and max_T, a matching lower bound can be proved by the same reduction presented in Lemma 4.12 for eventually synchronizing languages. In the presented reduction, the constructed PA is almost-sure eventually synchronizing in the singleton $\{\mathsf{synch}\}$ if and only if it is almost-sure strongly synchronizing.

From the above arguments and Lemma 6.12, it turns out that for all functions $f \in \{max_T, sum_T\}$, the universality problem of almost-sure strongly synchronizing language in PAs is PSPACE-complete.

7 Synchronization in Timed Automata

First sight. In this chapter, we introduce variants of synchronizing words for timed automata. Timed automata are finite automata enriched with some clocks and the transitions are fired with respect to some time constraints. The behavior of a timed automaton thus depends on such quantitative constraints, and this is one of the challenges when considering synchronization. Moreover, the automaton must be brought into the same state no matter what is the initial state as it does when synchronizing a finite automaton, however the initial state here ranges among possibly infinite set of states.

We focus on deciding the existence of different variants of synchronizing word for timed automata, proving PSPACE-completeness of the problems for deterministic timed automata, and proving undecidability for non-deterministic timed automata. The results presented in this chapter are summarized in Table 7.1.

Contents

	Synchronization	Location-synchronization
Deterministic TAs	PSPACE-complete	PSPACE-complete
Non-deterministic TAs	undecidable	

Table 7.1: Computational complexity of the synchronizing and location-synchronizing problem in TAs.

7.1 Preliminaries

We first introduce labeled transitions systems with possibly infinitely many states:

Definition 7.1 (Labeled transition systems)**.** *A* labeled transition system *over a (possibly infinite) alphabet* Γ *is a pair* $\langle Q, R\rangle$ *where* Q *is a set of states and* $R \subseteq Q \times \Gamma \times Q$ *is a transition relation. The state space* $Q = L \times X$ *consists of a finite set* L *of locations and a possibly infinite set* X *of quantitative values.*

Given a state $q = (\ell, x)$, let $\mathsf{loc}(q) = \ell$ be the location of q, and for $a \in \Gamma$, let $\mathsf{post}(q, a) = \{q' \mid (q, a, q') \in R\}$. For $P \subseteq Q$, let $\mathsf{loc}(P) = \{\mathsf{loc}(q) \mid q \in P\}$ and $\mathsf{post}(P, a) = \bigcup_{q \in P} \mathsf{post}(q, a)$. For nonempty words $w, w' \in \Gamma^+$ where $w = a \cdot w'$, define inductively $\mathsf{post}(q, w) = \mathsf{post}(\mathsf{post}(q, a), w')$. A *run* (or *path*) in a labeled transition system $\langle Q, R\rangle$ over Γ is a finite sequence $q_0 q_1 \cdots q_n$ such that there exists a word $a_0 a_1 \cdots a_{n-1} \in \Gamma^*$ for which $(q_i, a_i, q_{i+1}) \in R$ for all $0 \leq i < n$.

In the sequel, we consider labeled transition systems induced by timed automata.

7.1.1 Timed automata and timed words

Let $C = \{x_1, \ldots, x_{|C|}\}$ be a finite set of *clocks*. A (clock) valuation is a mapping $v \colon C \to \mathbb{R}_{\geq 0}$ that assigns to each clock a non-negative real number. We denote by $\mathbf{0}_C$ (or $\mathbf{0}$ when the set of clocks is clear from the context) the valuation that assigns 0 to every clock.

Let $\mathcal{I}$ be the set of intervals with integer or infinite endpoints. A *guard* $g = (I_1, \ldots, I_{|C|})$ over C is a tuple of $|C|$ intervals $I_i \in \mathcal{I}$. A valuation v satisfies g, denoted $v \models g$, if $v(x_i) \in I_i$ for all $1 \leq i \leq |C|$. For $t \in \mathbb{R}_{\geq 0}$, we denote by $v + t$ the valuation defined by $(v + t)(x) = v(x) + t$ for all $x \in C$, and for a set $r \subseteq C$ of clocks, we denote by $v[r]$ the valuation such that $v[r](x) = 0$ for all $x \in r$, and $v[r](x) = v(x)$ for all $x \in C \setminus r$.

Definition 7.2 (Timed automata)**.** *A* timed automaton *(TA) over a finite alphabet* A *is a tuple* $\langle L, C, E\rangle$ *consisting of a finite set* L *of locations, a finite set* C *of clocks, and a set* $E \subseteq L \times \mathcal{I}^{|C|} \times \mathsf{A} \times 2^C \times L$ *of edges.*

When E is clear from the context, we denote by $\ell \xrightarrow{g,a,r} \ell'$ the edge $(\ell, g, a, r, \ell') \in E$, which represents a transition on the letter a from location ℓ to ℓ' with the guard g and set r of clocks to reset.

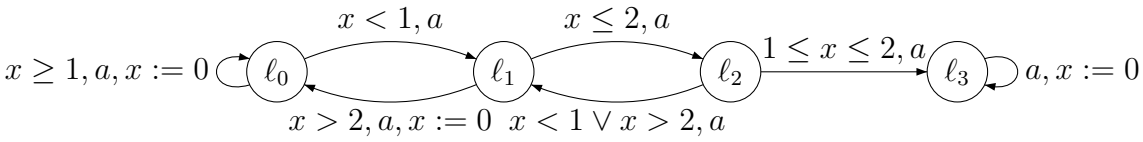

Figure 7.1: A 1-letter TA with synchronizing word $\mathsf{d}(3) \cdot a^3 \cdot \mathsf{d}(1) \cdot a^3$ where d is a special letter indicating quantitative delays.

The semantics of a TA $\mathcal{A} = \langle L, C, E \rangle$ is the labeled transition system $[\![\mathcal{A}]\!] = \langle Q, R \rangle$ over the alphabet $\Gamma = \mathbb{R}_{\geq 0} \cup \mathsf{A}$ [1] where $Q = L \times (C \to \mathbb{R}_{\geq 0})$, and $((\ell, v), \gamma, (\ell', v')) \in R$ if
- either $\gamma \in \mathbb{R}_{\geq 0}$, and $\ell = \ell'$ and $v' = v + \gamma$;
- or $\gamma \in \mathsf{A}$, and there is an edge $(\ell, g, \gamma, r, \ell') \in E$ such that $v \models g$ and $v' = v[r]$.

A *timed word* is a sequence $w = a_0 a_1 \cdots a_n$ with $a_i \in \mathsf{A} \cup \mathbb{R}_{\geq 0}$ for all $0 \leq i \leq n$. For a timed word, the *length* is the number of letters in A it contains, and the *granularity* is infinite if the word involves non-rational delays, and it is the largest denominator if the timed word only involves rational delays. As an example consider the timed word $w = \mathsf{d}(3) \cdot a^3 \cdot \mathsf{d}(1) \cdot a$ where $\mathsf{d}(t)$ with $t \in \mathbb{R}_{\geq 0}$ denotes the quantitative delays, and the length of w is 4.

Example 7.1. *The TA drawn in Figure 7.1 has four locations $\ell_0, \ell_1, \ell_2, \ell_3$, one letter a and one clock x. In the location ℓ_0, if the letter a is read when the clock valuation $x < 1$ is still less than 1, the automaton moves to ℓ_1; otherwise, the automaton stays in ℓ_0 but resets the clock x. In the location ℓ_1, if the letter a is input when the clock valuation $x \leq 2$ is less than or equal to 2, the automaton moves to ℓ_2; otherwise it goes to ℓ_0 and resets the clock x too. In the location ℓ_2, reading the letter a leads the automaton to end up in ℓ_3 or move to ℓ_1 depending on the guards: $\ell_2 \xrightarrow{1 \leq x \leq 2, a} \ell_3$ and $\ell_2 \xrightarrow{x < 1 \vee x > 2, a} \ell_1$ where the disjunction in the guard of a transition is obtained by several interval guarded transitions. The location ℓ_3 is a sink state since no transitions leave it. Consider the timed word $w = \mathsf{d}(3) \cdot a^3 \cdot \mathsf{d}(1) \cdot a$ where $\mathsf{d}(t)$ with $t \in \mathbb{R}_{\geq 0}$ denotes the quantitative delays. A valid run of the automaton from the state $(\ell_0, 0)$ to the state $(\ell_3, 1)$ over w is*

$$(\ell_0, 0) \xrightarrow{\mathsf{d}(3)} (\ell_0, 3) \xrightarrow{a} (\ell_0, 0) \xrightarrow{a} (\ell_1, 0) \xrightarrow{a} (\ell_2, 0) \xrightarrow{\mathsf{d}(1)} (\ell_2, 1) \xrightarrow{a} (\ell_3, 1).$$

◁

The TA $\mathcal{A}$ is *deterministic* if for all states $(\ell, v) \in Q$, for all edges $(\ell, g_1, a, r_1, \ell_1)$ and $(\ell, g_2, b, r_2, \ell_2)$ in E, if $a = b$, and $v \models g_1$ and $v \models g_2$, then $r_1 = r_2$ and $\ell_1 = \ell_2$; it is *complete* if for all $(\ell, v) \in Q$ and all $a \in \mathsf{A}$, there exists an edge $(\ell, g, a, r, \ell') \in E$ such that $v \models g$. For instance, consider the TA in Example 7.1. Since for all locations ℓ and all valuations for the only clock x, there is only one a-transition the TA is deterministic.

1. We assume that $\mathsf{A} \cap \mathbb{R}_{\geq 0} = \emptyset$.

7.1.2 Problems in TAs

The decidability of several classical problems in TAs are proved by the classical notion of region equivalence, see [AD94]. The region equivalence partitions the set of clock valuations into finitely many classes called *regions* where two states in the same location and same region are time-abstract bisimilar.

For a TA $\langle L, C, E\rangle$ over the finite alphabet A, let M be the maximal constant that appears in the guards in absolute value. The region equivalence $\cong$ on valuations is defined as follows. For all pairs of clock valuations v and v', we define $v \cong v'$ if and only if the three following conditions hold for all pairs of clocks $x, y \in C$:

1. the clock x in valuation v is greater than M if and only if it is in v'

$$v(x) > M \Leftrightarrow v'(x) > M,$$

2. if the clock x in one of the valuations is less than M, then the integer part of the clock x in both valuations is equal, and moreover the fractional part is zero only if it is zero in both valuations v and v'

$$\text{if } v(x) \leq M \text{ then } (\lfloor v(x)\rfloor = \lfloor v'(x)\rfloor) \text{ and } (\mathsf{frac}(v(x)) = 0 \Leftrightarrow \mathsf{frac}(v'(x)) = 0),$$

3. if both clocks x, y in one of the valuations are less than M, then the value of clock x is more than clock y in v if and only if the same holds in the valuation v'

$$\text{if } v(x), v(y) \leq M \text{ then } (\mathsf{frac}(v(x)) > \mathsf{frac}(v(y)) \Leftrightarrow \mathsf{frac}(v'(x)) > \mathsf{frac}(v'(y))),$$

where $\mathsf{frac}(v(x)) = v(x) - \lfloor v(x)\rfloor$ denotes the fractional part of the clock valuation. The equivalence class of a valuation v for the relation $\cong$ is a *region* of $\mathcal{A}$. By $\mathcal{R}$ we denote the set of regions where its size $|\mathcal{R}|$ is $O(2^{|C|} \times |C|! \times (2M+1)^{|C|})$, i.e. exponential in the size of the TA $\mathcal{A}$. The *region automaton* is then a finite-state automaton obtained by quotienting the original TA with the region equivalence (where delay transitions are abstracted by a d-transition). We refer to [AD94] for more details on the region abstraction for TAs.

Example 7.2. *The TA depicted in Figure 7.2 has two locations p, q, one letter and one clock x. The a-transitions in the location p either stay there and reset the clock x; or if the clock x is less than 1, the a-transition brings the automaton from p to q. The state q is a self-loop that the only transition is always available. The region automaton of the TA has eight states as it is shown in Figure 7.2. The d-transitions abstract away the exact values of delays. For instance, from the state $(p, 0 < x < 1)$ the d-transition is directed to $(p, x = 1)$ and from there, an a-transition is directed to $(p, x = 0)$. However, we can see that for each state (p, x) with $0 < x < 1$ in the TA, the exact value of delay that allows the transition $p \xrightarrow{x\geq 1, a, x:=0} p$ to be fired and moves the TA to the state $(p, 0)$ is different from delays for the other states (p, x') with $x \neq x'$.*

◁

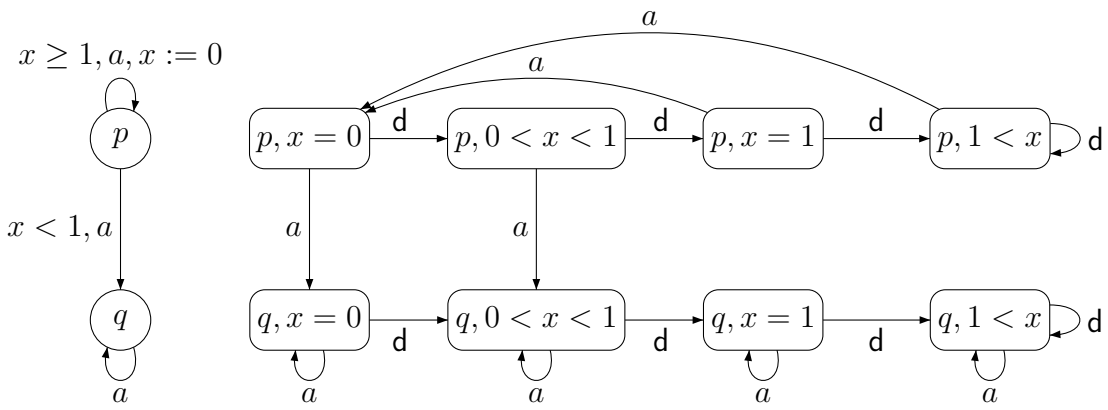

Figure 7.2: A TA and its region automaton (d is a special letter indicating delay transitions). The region automaton is synchronized by the word $a \cdot a \cdot \mathsf{d} \cdot \mathsf{d} \cdot \mathsf{d}$, but the TA cannot be synchronized (because there is no way to reset the clock x when starting in the location q).

Decision problem(s). The *reachability* problem in TAs asks, given a TA and two locations ℓ_i and ℓ_f, whether there exists a run from $(\ell_i, \mathbf{0})$ to some state (ℓ_f, v) (with arbitrary valuation v).

The reachability problem is PSPACE-complete where the PSPACE membership follows from reachability in the region automata which has exponentially many states in the size of the input TA. The PSPACE-hardness is obtained by a reduction from halting problem for linear bounded automata (Turing machines that use only a polynomial number of cells on their tape).

Given a TA equipped with an initial location ℓ_i and a set $\mathcal{F}$ of accepting locations, the timed language of the automaton is the set of timed words for which there exists some run from the initial state $(\ell_i, \mathbf{0})$ reaching some accepting state (ℓ_f, v) where $\ell_f \in \mathcal{F}$ and v is an arbitrary clock valuation.

Decision problem(s). The *universality* problem in TAs asks, given a TA equipped with an initial state and a set of accepting states, whether the timed language contains all timed words.

The universality problem is known to be undecidable in (non-deterministic) TAs by encoding the computation of a Minsky Machine [AD94].

7.2 Synchronization in TA

We generalize the finite synchronizing words in NFAs, presented in Definition 3.1 on page 38, to TAs.

Definition 7.3 (Synchronizing and location-synchronizing words for TAs)**.** *Given a complete TA $\mathcal{A} = \langle L, C, E\rangle$ over the alphabet A and with the semantic (the labeled transition system) $[\![\mathcal{A}]\!] = \langle Q, R\rangle$, a timed word w is* synchronizing *if $\mathsf{post}(Q, w)$ is a singleton, and it is* location-synchronizing *if $\mathsf{loc}(\mathsf{post}(Q, w))$ is a singleton.*

Thus a synchronizing word can be read from every state and bring the TA to a single state, and a location-synchronizing word brings the TA to a (set of states where all agree on a) single location with possibly different clock valuations. A TA is *(location-)synchronizing* if there exists some (location-)synchronizing word for it.

Decision problem(s). The *(location-)synchronizing problem* asks, given a TA $\mathcal{A}$, whether it is (location-)synchronizing.

For instance, consider the TA described in Example 7.1. We have infinitely many states to synchronize using the letter a and quantitative delays $\mathsf{d}(t)$ ($t \in \mathbb{R}_{\geq 0}$). One can verify that the timed word $\mathsf{d}(3) \cdot a^3 \cdot \mathsf{d}(1) \cdot a^3$ synchronizes the TA into the state $(\ell_3, 0)$. Note that a synchronizing word, in general, is location-synchronizing too. However, in this example, since the a-transition in the location ℓ_3 is always available and always resets the clock x ($\ell_3 \xrightarrow{a, x:=0} \ell_3$), concatenating one more a at the end of each location-synchronizing word gives a synchronizing word, and thus synchronization and location-synchronization coincide.

In following, Example 7.3 illustrates a TA where the region equivalence is not sound to find a synchronizing word: the region automata is synchronizing while the TA has no synchronizing word. This unsoundness is due to the fact that region equivalence abstracts away the exact value of the clocks (as explained in Example 7.2), while synchronizing needs to keep track of them.

Example 7.3. *Consider the TA described in Example 7.2. Since the a-transition in the location q never reset the clock x, so two runs starting from two states (q, v_1) and (q, v_2) where $v_1(x) \neq v_2(x)$ are never able to end up in the same state. The automaton thus has no synchronizing word whereas the region automata is synchronizing by the word $a \cdot a \cdot \mathsf{d} \cdot \mathsf{d} \cdot \mathsf{d}$.*

◁

There are simple examples showing that corner point abstraction, another known abstraction in TAs [BBL08], is not sound to find synchronizing words in TAs too.

7.3 Synchronization in deterministic TAs

In this section, we prove that deciding whether there exists a synchronizing word for a given complete deterministic TA, is PSPACE-complete. To establish the membership in

PSPACE, we first prove the existence of a *short witness* (in the sequel, a timed word is *short* when its length and granularity are in $O(2^{|C|} \times |L| \times |\mathcal{R}|)$). The built short witness starts with a *finitely-synchronizing* word, a word that brings the infinite set of states of the automaton to a finite set, and continues by synchronizing the states of this finite set pairwise.

Example 7.4. *Consider the TA described in Example 7.1 that is a complete automaton. To find a synchronizing word for the TA, our proposed algorithm consists in two phases: first reducing the (uncountably) infinite set of configurations into a finite set (with at most the number of locations in the TA) by finding a finitely-synchronizing word, and then pairwise synchronizing the obtained finite set of states. The word* $\mathsf{d}(3)\cdot a\cdot a$ *is a* finitely synchronizing word *that synchronizes the infinite set of states into a finite set: whatever the initial state, inputting the word* $\mathsf{d}(3)\cdot a\cdot a$ *the automaton ends up in one of the states* $(\ell_0, 0)$, $(\ell_1, 0)$ *or* $(\ell_3, 0)$. *On the other hand, since* ℓ_3 *is an absorbing state (all transitions are self-loop on* ℓ_3*), all synchronizing words must synchronize the automaton in a state* (ℓ_3, x) *with location* ℓ_3 *and* $x \in \mathbb{R}_{\geq 0}$. *It then suffices to input* $a \cdot \mathsf{d}(1) \cdot a \cdot a \cdot a$ *to end up in* $(\ell_3, 0)$*, whatever the initial state. Our algorithm thus returns the synchronizing word* $\mathsf{d}(3) \cdot a^3 \cdot \mathsf{d}(1) \cdot a^3$ *for the TA, which always leads to the state* $(\ell_3, 0)$.

◁

We fix a complete deterministic TA $\mathcal{A} = \langle L, C, E\rangle$ with the maximal constant M appearing in the guards in absolute value. We begin with two folklore remarks on TAs. For all locations ℓ, we denote by $L_\ell = \{(\ell, v) \mid v(x) > M \text{ for all clocks } x \in C\}$ the set of states with location ℓ and where all clocks are unbounded. Then the clock valuations in L_ℓ form a region (and L_ℓ is one of the states in the region automata of $\mathcal{A}$).

Remark 10. For all locations ℓ and for all timed words w, the set $\mathsf{loc}(\mathsf{post}(L_\ell, w))$ is a singleton and $\mathsf{post}(L_\ell, w)$ is included in a single region.

Proof. The proof is by an induction on the length of the timed words w. We reinforce the statement of the remark as follows. For all locations $\ell \in L$, for all pairs of states $q_1, q_2 \in L_\ell$ and all timed words w, let us write $\mathsf{post}(q_1, w) = \{(\ell_1, u_1)\}$ and $\mathsf{post}(q_2, w) = \{(\ell_2, u_2)\}$. After inputting the timed word w from both states q_1 and q_2, the automaton

- ends up in the same location: $\ell_1 = \ell_2$, and
- for all clocks $x \in C$, either the clock in both valuations is unbounded

$$u_1(x) > M \Leftrightarrow u_2(x) > M,$$

  or the clocks have the same value: $u_1(x) = u_2(x)$.

The base of induction clearly holds: since after inputting the empty word ϵ we have $u_1(x) > M$ and $u_2(x) > M$ for all clocks $x \in C$. Assuming that the statement holds for all timed words w, one can easily see that it still holds after a delay transition. In case of a letter-transition (such as a-transition where $a \in \mathsf{A}$), the clock values are preserved, except for the clocks x that are reset by taking the transition. Both those clocks then have value zero in both valuations (both $u_1(x)$ and $u_2(x)$).

□

Notice that Remark 10 is a special property of the state L_ℓ, and in general: elapsing the same delay from two region-equivalent valuations may lead to non-equivalent valuations.

Remark 11 is technical and provides the length and granularity of timed words that are considered when solving reachability problem in TAs.

Remark 11. For all locations $\ell \in L$, let $r \in \mathcal{R}$ where $L_\ell = (\ell, r)$ in the region automaton of $\mathcal{A}$. For all locations $\ell' \in L$ and regions $r' \in \mathcal{R}$ such that (ℓ', r') is reachable from the state (ℓ, r) in the region automaton, there exists a short timed word w of length at most $|L| \times |\mathcal{R}|$ and two valuations $v \in r$ and $v' \in r'$ such that $\mathsf{post}((\ell, v), w) = \{(\ell', v')\}$.

Proof. Let $\rho = (\ell_0, r_0)(\ell_1, r_1) \cdots (\ell, n, r_n)$ be a simple path in the region automaton from (ℓ, r) to (ℓ', r') where $(\ell_0, r_0) = (\ell, r)$ and $(\ell_n, r_n) = (\ell', r')$. Since the path ρ is simple (there is no cycles), it has size less than the number of states in the region automaton, namely $n \leq |L| \times |\mathcal{R}|$. Let $v \in r$ be a valuation in the region r, then $(\ell, v) \in L_\ell$. We denote the sequence of letters read along the run ρ with $a_0 \cdot a_1 \cdots a_m$ where $a_i \in \mathsf{A}$ for all $0 \leq i \leq m$, and $m \leq n$. Considering the timestamps t_i (for all $0 \leq i \leq m$) and letting $t_{-1} = 0$, we define the timed word $w = b_0 \cdot b_1 \cdots b_{2m+1}$ as follows: let $b_{2i} = (t_i - t_{i-1})$ and let $b_{2i+1} = a_i$. The guards that have to be satisfied along the path ρ entail conditions of the form $t_n - t_m \sim c$, which defines a convex zone of possible timestamps. This zone has at most $|\rho|$ integer vertices (which may not belong to the zone itself, but only to its closure). The center of these vertices belongs to the zone, and has the expected granularity, which proves the result.

□

Lemma 7.1. *All synchronizing deterministic TAs have a short finitely-synchronizing word.*

Proof. Let $\mathcal{A} = \langle L, C, E \rangle$ be a complete deterministic TA with the maximal constant M appearing in the guards in absolute value. Assume that the TA $\mathcal{A}$ has a synchronizing word, we build a short finitely-synchronizing word w_f for $\mathcal{A}$. The built word has a key property: for all clocks $x \in C$, irrespective of the starting state, the run over w_f takes some transition resetting x. We first argue that for all clocks $x \in C$, from all states with the clock valuations $v(x) \neq 0$ not equal to zero, there exists a reachable x-resetting transition. Towards contradiction, assume that there exist some state (ℓ, v) and clock x such that x will never be reset along any run from (ℓ, v). Runs starting from states with the same location ℓ but different clock valuations, say (ℓ, v') with $v'(x) \neq v(x)$, over a synchronizing word w, may either (1) reset x, and thus the final values of x on two runs from (ℓ, v) and (ℓ, v') are different, or (2) not reset x, so that the difference between $v(x)$ and $v'(x)$ is preserved along the runs over w. Both cases give contradiction, and thus for all clocks $x \in C$, from all states with $v(x) \neq 0$, there exists a reachable x-resetting transition.

Let $\ell \in L$ be some location and $x \in C$ be some clock. Let time word $w_{\ell,x}$ be a word built by applying the argument presented above to an arbitrary state q of L_ℓ and the clock x.

By Remark 10, inputting the same timed word from any state of L_ℓ always leads to the same transition resetting x. Moreover, all such runs end up in the same region. Note that by Remark 11, $w_{\ell,x}$ can be chosen to have length and granularity at most $|L| \times |\mathcal{R}|$.

Below, we construct the short finitely synchronizing word w_f for $\mathcal{A}$ where S is the infinite set of states to be (finitely) synchronized (meaning that $\mathsf{post}(S, w_f)$ must be a finite set). Repeat the following procedure until S is synchronized to a finite set.

1. Choose a location $\ell \in \mathsf{loc}(S)$ such that there is an infinite set $S_\ell \subseteq S$ of states (ℓ, v) with that location ℓ and some clock valuation v, included in S.
2. For all clocks $x \in C$, iteratively, input a word that consists of a $(M+1)$-time-unit delay followed by the word $w_{\ell,x}$. The timed word of $M+1$ delay brings the infinite set S_ℓ to the unbounded region L_ℓ. Next, inputting the word $w_{\ell,x}$ makes the runs starting from S_ℓ end up in a single region where clock x has the same value for all runs (since it has been reset). The word $w_\ell = (\mathsf{d}(M+1) \cdot w_{\ell,x})_{x \in C}$ synchronizes the infinite set S_ℓ to a single state by resetting all clocks, one-by-one, and it also shrinks S.
3. Apply the word w_ℓ to S and update this set by $S = \mathsf{post}(S, w_\ell)$. Among all reached locations, choose the next location $\ell' \in \mathsf{loc}(S)$ such that there is an infinite set $S_\ell \subseteq S$ of states (ℓ, v) with that location ℓ and some clock valuation v, included in S. Repeat the second and third step until S is synchronized to a finite set.

Note that for all locations ℓ, the word w_ℓ has length at most $|C| \times |L| \times (|\mathcal{R}| + 1)$ and granularity at most $|L| \times |\mathcal{R}|$. Thus the finitely-synchronizing word w_f, obtained by concatenating the successive words w_ℓ, has length bounded by $|C| \times |L|^2 \times (|\mathcal{R}| + 1)$ and granularity at most $|L| \times |\mathcal{R}|$, so that it is short. By construction, the word w_f finitely-synchronizes $\mathcal{A}$, which concludes our proof.

□

From the proof of Lemma 7.1, we see that for all synchronizing TAs, there exists a finitely-synchronizing word which, in a sense, synchronizes the clock valuations. Precisely:

Corollary 7.1. *For all synchronizing deterministic TAs, there exists a short finitely-synchronizing word w_f such that for all locations ℓ, this word w_f synchronizes the set $\{\ell\} \times (C \to \mathbb{R}_{\geq 0})$ into a single state.*

Lemma 7.2 uses Corollary 7.1 to construct a short synchronizing word for a synchronizing TA. Short synchronizing words consist of a *finitely-synchronizing* word followed by a *pairwise synchronizing* word (i.e., a word that iteratively synchronizes pairs of states).

Lemma 7.2. *All synchronizing deterministic TAs have a short synchronizing word.*

Proof. Let $\mathcal{A} = \langle L, C, E \rangle$ be a complete deterministic TA with the maximal constant M appearing in the guards in absolute value. By Corollary 7.1, we know that for $\mathcal{A}$ there exists a short finitely synchronizing word w_f that synchronizes the clock valuations. Thus, to synchronize $\mathcal{A}$ it is sufficient (and necessary) to synchronize the set $S = \mathsf{post}(L \times (C \to$

$\mathbb{R}_{\geq 0}), w_f)$ obtained by shrinking the infinite state space $L \times (C \to \mathbb{R}_{\geq 0})$ after inputting w_f. We assume, w.l.o.g., that all clock valuations of states in S are in the unbounded region (since otherwise we can concatenate the timed word of $M+1$ delay at the end of w_f). Since S has finite cardinality at most $n = |L|$, we enumerate its elements and denote it by

$$S_n = \{(\ell_1, v_1), (\ell_2, v_2), \cdots, (\ell_n, v_n)\}.$$

Consider the timed automaton $\mathcal{A}^2$ obtained as the product of two copies of $\mathcal{A}$: this automaton has $2|C|$ clocks and $|L|^2$ locations where we denote the clocks in $\mathcal{A}^2$ by $(x^j)_{j\in\{1,2\},x\in C}$. To synchronize S_n to a single state, we repeat the following procedure for all $i = n, n-1, \cdots, 1$. Take a pair of states in S_i, say (ℓ_1, v_1) and (ℓ_2, v_2). Consider the state (L, V) of $\mathcal{A}^2$ defined by these states where $L = (\ell_1, \ell_2)$ and $V(x^j) = v_j(x)$ for $j \in \{1, 2\}$ and $x \in C$. Since $\mathcal{A}$ has a synchronizing word, there is a run from (L, V) in $\mathcal{A}^2$ to a *symmetric* state of the form (L', V') with $L' = (\ell', \ell')$ for some ℓ' and $V(x^1) = V(x^2)$ for all $x \in C$. Write w_i for the corresponding word, and let $S_{i-1} = \mathsf{post}(S_i, w_i)$. Repeat the same procedure for $i-1$. We see that S_i contains at most i states, thus the word $w = w_n \cdot w_{n-1} \cdots w_1$ synchronizes S_n to a single state. For all i, the word w_i is chosen to have length at most $2|C| \times |L|^2 \times (|\mathcal{R}_2| + 1)$ and granularity at most $|L| \times |\mathcal{R}_2|$, where $|\mathcal{R}_2| \leq (4M+1)^{2|C|} \times (2|C|)! \leq |\mathcal{R}|^{|C|+1}$, so that w has length and granularity polynomial in $|L|$ and M, and exponential in $|C|$. Thus, $w_f \cdot w$ is a synchronizing word for $\mathcal{A}$, and it is short.

□

A naive algorithm for deciding the existence of a synchronizing word would consist in non-deterministically picking a short timed word, and checking whether it is synchronizing. However, the latter cannot be done easily, because we have infinitely many states to check, and the region automaton is not sound for this.

Lemma 7.3. *The synchronizing problem in deterministic TAs is in* PSPACE.

Proof. To have a PSPACE algorithm to decide the synchronizing problem in TAs, we cannot simply guess a short timed word and check whether it is synchronizing since we have infinitely many states to check if the runs starting from those states end up in a single state after reading the short witness.

Let $\mathcal{A} = \langle L, C, E\rangle$ be a complete deterministic TA with the maximal constant M appearing in the guards in absolute value. We thus first consider the set $S_0 = \{(\ell, \mathbf{0}) \mid \ell \in L\}$ and compute the set of successors reached from S_0 after reading a finitely-synchronizing word w_f (built in the proof of Lemma 7.1); that is, we simply compute the set $\mathsf{post}(S_0, w_f)$. This can be achieved using polynomial space, since S_0 contains polynomially many states and w_f can be guessed on-the-fly. The important point here is that since w_f begins with a delay of $M+1$ time unit, the set $\mathsf{post}(S_0, w_f)$ is equal to the set $\mathsf{post}(Q, w_f)$ where $Q = L \times \mathbb{R}^C_{\geq 0}$ is the state space of the semantic $[\![\mathcal{A}]\!]$ of the TA $\mathcal{A}$.

The set $\mathsf{post}(S_0, w_f)$ contains at most $|L|$ states, and this set $\mathsf{post}(S_0, w_f)$ can now be synchronized pairwise. This pairwise-synchronization phase can be achieved by computing

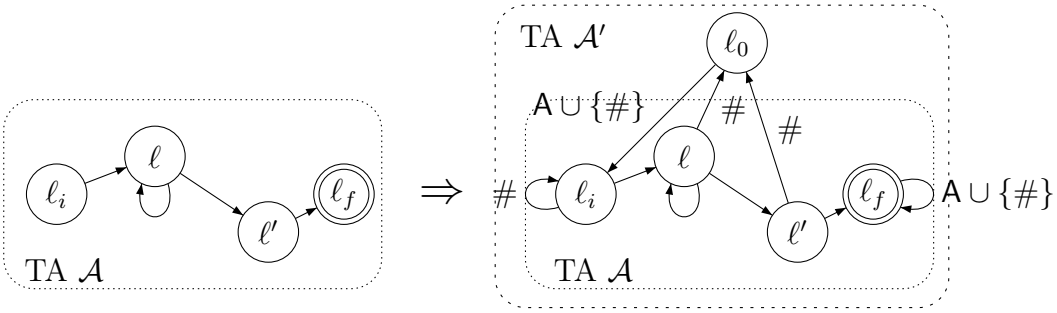

Figure 7.3: The reduction sketch to show PSPACE-hardness of the (location-)synchronizing problem in TAs.

the product automaton $\mathcal{A}^2$ and solving reachability problems in that automaton (similar to the proof of Lemma 7.2). This algorithm runs in polynomial space, and successfully terminates if, and only if, $\mathcal{A}$ has a synchronizing word.

□

Using similar arguments, we obtain the following result stating the PSPACE membership of the location-synchronizing problem in TAs.

Lemma 7.4. *The location-synchronizing problem in deterministic TAs is in* PSPACE.

Proof. Let $\mathcal{A} = \langle L, C, E \rangle$ be a complete deterministic TA with the maximal constant M appearing in the guards in absolute value. To location-synchronize the TA $\mathcal{A}$, we propose the following.

- The location-synchronizing word w starts with $(M+1)$-time-unit delay, in such a way that after this delays all the clocks $x \in C$ end up with values above the maximal constant M.
- We see that for all locations ℓ, all timed words w and all pairs of region-equivalent valuations v and v', the states in $\mathsf{post}((\ell, v), w)$ and $\mathsf{post}((\ell, v'), w)$ still have the same location and region-equivalent valuations. Hence location-synchronization can be achieved by just considering the set $\{(\ell, v_{M+1}) \mid \ell \in L\}$, where the valuation v_{M+1} maps all clocks to $M+1$. These states can be location-synchronized pairwise.

There is one important point to take into account: two states q and q' that have been location-synchronized might later de-synchronize, since they do not end up in the same region. However, this problem is easily avoided by letting $M+1$ time unit elapse after location-synchronizing each pair of states. Using the same arguments as for synchronizing word the above procedure runs in polynomial space. The membership of location-synchronizing problem in PSPACE follows.

□

Lemma 7.5. *The (location-)synchronizing problem in deterministic TAs is* PSPACE-hard.

Proof. The proof is by a reduction from the reachability problem in TA, which is known to be PSPACE-complete [AD94].

Given a deterministic TA $\mathcal{A} = \langle L, C, E\rangle$ (without loss of generality, we assume that $\mathcal{A}$ is complete) defined over a finite alphabet A and equipped with two locations ℓ_i and ℓ_f, we construct another TA $\mathcal{A}'$ such that there is some clock valuation v such that there exists a run in $\mathcal{A}$ starting from $(\ell_i, \mathbf{0})$ reaching the state (ℓ_f, v) if and only if the automaton $\mathcal{A}'$ has a (location-)synchronizing word.

The TA $\mathcal{A}' = \langle L', C, E'\rangle$ is a copy of $\mathcal{A}$ with one new location ℓ_0: $L' = L \cup \{\ell_0\}$. The automaton $\mathcal{A}'$ is defined on the alphabet $\mathsf{A}' = \mathsf{A} \cup \{\#\}$ where $\# \notin \mathsf{A}$ is a new letter. See Figure 7.3. All transitions in $\mathcal{A}$ are copied to $\mathcal{A}'$ except the transitions in the location ℓ_f. The following transitions are added, that are always available and reset all the clocks,

- an a-transition in ℓ_f for all letters $a \in \mathsf{A} \cup \{\#\}$ which is a self-loop:

$$(\ell_f, \texttt{true}, a, C, \ell_f) \text{ for all } a \in \mathsf{A}.$$

- an a-transition from the new location ℓ_0 to ℓ_i for all letters $a \in \mathsf{A} \cup \{\#\}$:

$$(\ell_0, \texttt{true}, a, C, \ell_i) \text{ for all } a \in \mathsf{A}.$$

- a $\#$-transition in ℓ_i which is a self-loop: $(\ell_i, \texttt{true}, \#, C, \ell_i)$.
- from all locations $\ell \in L' \setminus \{\ell_0, \ell_i, \ell_f\}$ except ℓ_0, ℓ_i and ℓ_f, a $\#$-transition to ℓ_0:

$$(\ell, \texttt{true}, \#, C, \ell_0).$$

The resulting automaton $\mathcal{A}'$ is deterministic and complete.

We establish the correctness of the reduction as follows. First assume that $\mathcal{A}'$ has a synchronizing word w. We prove that there exists some clock valuation v such that $\mathcal{A}$ has a run from $(\ell_i, \mathbf{0})$ to (ℓ_f, v). On reading the synchronizing word w by the automaton $\mathcal{A}'$, the final location necessarily is (ℓ_f, v) where v is some clock valuation, as ℓ_f has no outgoing transition. Thus, the run starting in $(\ell_i, \mathbf{0})$ over this word w also reaches (ℓ_f, v); however this run might possibly take some $\#$-transitions, which are not valid transitions in $\mathcal{A}$. Consider the shortest subrun going from $(\ell_i, \mathbf{0})$ to (ℓ_f, v). That run does not contain any $\#$-transition, since any such transition either corresponds to the self-loop on ℓ_f or ℓ_i, or leads to ℓ_0 and will be followed by another visit to $(\ell_i, \mathbf{0})$, both options contradicting the fact that we have considered the shortest subrun from $(\ell_i, \mathbf{0})$ to (ℓ_f, v).

Second, assume that there exists a clock valuation v such that there is a run in $\mathcal{A}$ from $(\ell_i, \mathbf{0})$ to (ℓ_f, v) over some timed word w. Then the word $\# \cdot \# \cdot w \cdot \#$ necessarily synchronizes the whole state space of the TA $\mathcal{A}'$ into the state $(\ell_f, \mathbf{0})$: indeed,

- since there are only self-loops in the location ℓ_f, we have $\mathsf{post}((\ell_f, v'), w) = (\ell_f, \mathbf{0})$ for all clock valuations v'.
- moreover for all clock valuations v', from (ℓ_i, v') reading two times $\#$ brings the automaton into $(\ell_i, \mathbf{0})$, and then following the word w the state (ℓ_f, v) is reached. The last $\#$ resets all the clocks and the automaton end up in $(\ell_f, \mathbf{0})$.

– for all clock valuations v', from (ℓ_0, v') the automaton end up in $(\ell_i, \mathbf{0})$ by taking the #-transitions. Next, inputting the word $w \cdot \#$ synchronizes the automaton into $(\ell_f, \mathbf{0})$ as discussed in the former case.
– for all clock valuations v', from (ℓ, v) where $\ell \notin \{\ell_0, \ell_i, \ell_f\}$, the first #-transition leads to $(\ell_0, \mathbf{0})$, and the second one goes to $(\ell_i, \mathbf{0})$. Inputting the word $w \cdot \#$ afterward synchronizes the automaton into $(\ell_f, \mathbf{0})$ as discussed in the former case.

The proof is complete and the PSPACE-hardness follows. Note that the same reduction is correct to establish the result for the location-synchronizing problem: in fact since all transitions in ℓ_f (the only possible location to synchronize) always reset all clocks. Therefore, $\mathcal{A}'$ is synchronizing if and only if it is location-synchronizing.

□

From previous Lemmas, the following Theorem 7.1 is obtained.

Theorem 7.1. *The synchronizing and location-synchronizing problems in deterministic TAs are* PSPACE-complete.

7.4 Synchronization in non-deterministic TAs

We now show the undecidability of the synchronizing problem for non-deterministic TAs. The proof is by a reduction from the non-universality problem of timed language for non-deterministic TAs, which is known to be undecidable [AD94].

Theorem 7.2. *The (location-)synchronizing problem in non-deterministic TAs is undecidable.*

Proof. Let $\mathcal{A} = \langle L, C, E \rangle$ be a non-deterministic TA over an alphabet A equipped with an initial location ℓ_i and a set $\mathcal{F}$ of accepting locations (w.l.o.g. we assume that $\mathcal{A}$ is complete). From $\mathcal{A}$, we construct another TA $\mathcal{A}'$ over A' such that the language of $\mathcal{A}$ is not universal if and only if $\mathcal{A}'$ has a location-synchronizing word.

The construction of the TA $\mathcal{A}' = \langle L', C, E' \rangle$ defined over a new alphabet A' is as follows (See Figure 7.4). There are two new states s and d and the alphabet is augmented with two new letters # and $\star$. Formally, $L' = L \cup \{d, s\}$ (assuming $d, s \notin L$) and $\mathsf{A}' = \mathsf{A} \cup \{\#, \star\}$. All transitions in $\mathcal{A}$ are copied to $\mathcal{A}'$ and the following new transitions are added.

– Location s is a sink location, carrying a self-loop for all letters $a \in \mathsf{A} \cup \{\#, \star\}$. Location d is a *departure* location: it also carries a self-loop for all letters, except for $\star$, which leads to ℓ_i. All those transitions reset all the clocks.
– From all locations $\ell \in L$, there is a $\star$-transition to ℓ_i along which all the clocks are reset. From the states not in $\mathcal{F}$, there is a #-transition to s along which all clocks are reset. From the states in $\mathcal{F}$, the #-transition goes to d resetting all clocks.

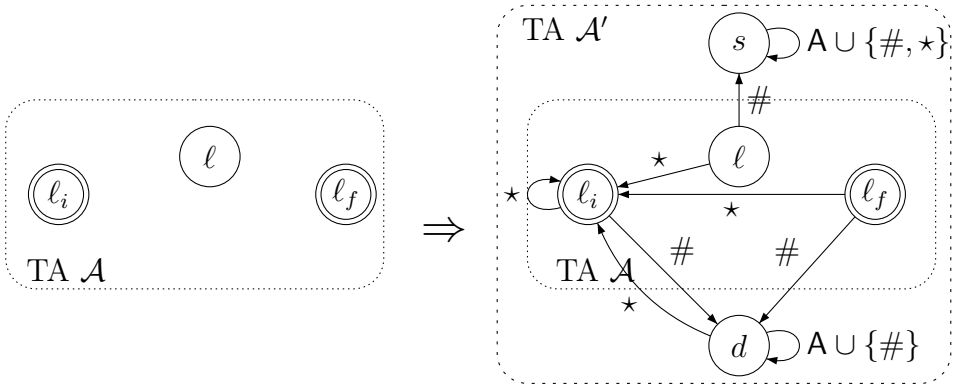

Figure 7.4: The reduction sketch to show undecidability of the (location-)synchronizing problem in TAs where in this example: $\{\ell_i, \ell_f\} \subseteq F$.

To establish the correctness of the reduction, we first assume that the timed language of $\mathcal{A}$ is not universal. So there exists a word w that is not accepted by $\mathcal{A}$. Then all runs of $\mathcal{A}$ over w starting in $(\ell_i, \mathbf{0})$ end in copies of non-accepting states. Hence in $\mathcal{A}'$, the word $\star \cdot w \cdot \#$ reaches the state $(s, \mathbf{0})$, whatever the starting state. It shows that $\mathcal{A}'$ has a synchronizing word.

Second, assume that $\mathcal{A}'$ has a synchronizing word. Let w be one of the shortest synchronizing words (in terms of the number of transitions). As s has no outgoing transitions, $\mathcal{A}'$ can only be synchronized in s. Since entering s is only possible by reading the letter $\#$, it must be the case that w has at least one occurrence of $\#$. Similarly, the states with the location d are also able to synchronize into s, it must be the case that w contains at least one occurrence of $\star$ which is followed by one occurrence of $\#$. Let w_1 be the subword of w between the last $\star$ that is followed by an occurrence of $\#$, up to the first subsequent occurrence of $\#$. So w_1 contains neither $\#$ nor $\star$, and w can be written as $w_0 \cdot \star \cdot w_1 \cdot \# \cdot w_2$, where w_2 contains no $\star$ (otherwise w would not be one of the shortest synchronizing word). We show that the word w_1 is not accepting by $\mathcal{A}$ (and thus, the timed language of $\mathcal{A}$ is not universal). Towards a contradiction, assume that w_1 is accepted by $\mathcal{A}$. It cannot be the case that w_0 is synchronizing, as w was chosen to be one of the shortest such words. Hence there must be two states (ℓ, v) and (ℓ', v') where $\ell' \neq s$, such that there is a run from (ℓ, v) to (ℓ', v') over the word w_0. Reading $\star$ from (ℓ', v') leads to $(\ell_i, \mathbf{0})$, from which there is a run over the word w_1 that goes to a state (ℓ'', v'') with $\ell'' \in F$. From (ℓ'', v''), reading $\#$ leads to $(d, \mathbf{0})$. Since w_2 contains no $\star$, there is no path from $(d, \mathbf{0})$ to location s by w_2. This means that we have found a state (ℓ, v) from which reading w does not lead to s, contradicting the fact that w is synchronizing in $\mathcal{A}'$. Hence w_1 is not accepted by $\mathcal{A}$ and $\mathcal{A}$ is thus not synchronizing.

The same reduction is used to show undecidability of the location-synchronizing problem; note that all transitions going to s (the only possible location to synchronize) always

reset all clocks. Therefore, the TA $\mathcal{A}'$ is synchronizing if and only if it is location synchronizing.

□

8 Synchronization in Weighted Automata

First sight. In this chapter, we introduce variants of synchronizing words for weighted automata. Weighted automata are finite automata where transitions are augmented with some weights that are mostly considered as the resource (or energy) consumptions. We define safe synchronization for weighted automata where the aim is to synchronize the set of safe states while forbidding the automaton to visit states outside the safety-set during synchronization. One of the challenges to synchronize weighted automata is having a possibly infinite set of states to synchronize, as it is for timed automata.

We focus on deciding the existence of different variants of synchronizing words for weighted automata (with safety conditions). For deterministic weighted automata, the synchronizing problem is proven PSPACE-complete under energy constraints, and in 3-EXPSPACE under general safety constraints. The results presented in this chapter are summarized in Table 8.1.

Contents

	Synchronization		Location-synchronization	
	No cond.	**Safety cond.**	**No cond.**	**Safety cond.**
Deterministic	Trivial (always `false`)	PSPACE-complete	NLOGSPACE-complete	3-EXPSPACE energy condition: PSPACE-complete
Non-deterministic	Trivial (always `false`)	PSPACE-complete	PSPACE-complete	?

Table 8.1: Computational complexity of the synchronizing and location-synchronizing problem in WAs.

8.1 Preliminaries

In this section, we provide the definitions of weighted automata, Minsky machine and vector addition systems with states.

8.1.1 Weighted automata

We present the definition of weighted automata as an instance of a labeled transition system with possibly infinite states.

Let us first recall the definition of labeled transition systems, which we have provided in Definition 7.1. A labeled transition system over an alphabet Γ is a pair $\langle Q, R\rangle$ where Q is a set of states and $R \subseteq Q \times \Gamma \times Q$ is a transition relation. The state space $Q = L \times X$ consists of a finite set L of locations and a possibly infinite set X of quantitative values. Given a state $q = (\ell, x)$, let $\mathsf{loc}(q) = \ell$ be the location of q, and for $a \in \Gamma$, let $\mathsf{post}(q, a) = \{q' \mid (q, a, q') \in R\}$. For $P \subseteq Q$, let $\mathsf{loc}(P) = \{\mathsf{loc}(q) \mid q \in P\}$ and $\mathsf{post}(P, a) = \bigcup_{q \in P} \mathsf{post}(q, a)$. For nonempty words $w \in \Gamma^+$, define inductively $\mathsf{post}(q, aw) = \mathsf{post}(\mathsf{post}(q, a), w)$. A *run* (or *path*) in a labeled transition system $\langle Q, R\rangle$ over Γ is a finite sequence $q_0 q_1 \cdots q_n$ such that there exists a word $a_0 a_1 \cdots a_{n-1} \in \Gamma^*$ for which $(q_i, a_i, q_{i+1}) \in R$ for all $0 \leq i < n$.

In the sequel, we consider labeled transition systems induced by weighted automata.

Definition 8.1 (Weighted automata)**.** *A* weighted automaton *(WA) over a finite alphabet* A *is a tuple* $\mathcal{A} = \langle L, E\rangle$ *consisting of a finite set* L *of locations, and a set* $E \subseteq L \times \mathsf{A} \times \mathbb{Z} \times L$ *of edges.*

When E is clear from the context, we denote by $\ell \xrightarrow{a:z} \ell'$ the edge $(\ell, a, z, \ell') \in E$, which represents a transition on letter a from location ℓ to ℓ' with weight z. We view the weights as the resource (or energy) consumption of the system.

The semantics of a WA $\mathcal{A} = \langle L, E\rangle$ is the labeled transition system $[\![\mathcal{A}]\!] = \langle Q, R\rangle$ on the alphabet A where $Q \subseteq L \times \mathbb{Z}$ and $((\ell, e), a, (\ell', e')) \in R$ if $(\ell, a, e' - e, \ell') \in E$. In a state (ℓ, e), we call e the *energy level*. The WA $\mathcal{A}$ is *deterministic* if for all edges (ℓ, a, z_1, ℓ_1),

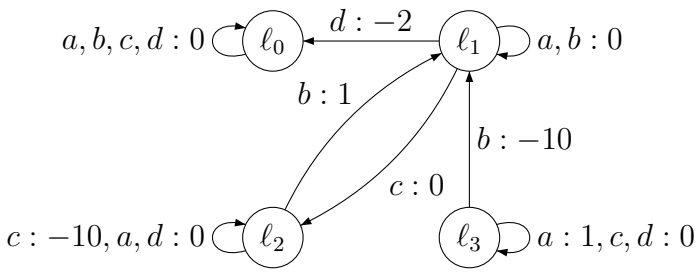

Figure 8.1: A complete deterministic WA with the synchronizing word $a^{10} \cdot b \cdot (c \cdot b)^2 \cdot d$ under the energy safety condition.

$(\ell, b, z_2, \ell_2) \in E$, if $a = b$, then $z_1 = z_2$ and $\ell_1 = \ell_2$; it is *complete* if for all $\ell \in L$ and all $a \in \mathsf{A}$, there exists an edge $(\ell, a, z, \ell') \in E$.

Example 8.1. *The WA drawn in Figure 8.1 has four locations $\ell_0, \ell_1, \ell_2, \ell_3$ and four letters a, b, c, d. The state ℓ_0 is an absorbing state where all transitions have weight 0. In the state ℓ_1, the d-transition brings the automaton to the absorbing state ℓ_0 with weight -2, while the c-transition is redirected to ℓ_2. The a and b-transitions are self-loops in ℓ_1. The b-transition in ℓ_2 goes to ℓ_1: $\ell_2 \xrightarrow{b:+1} \ell_1$; other transitions in ℓ_2 are however self-loops with weight 0 except the self-loop of the c-transition that has weight -10. All transitions in ℓ_3 are self-loops with weights 0 or 1; except the b-transition that leaves this state and goes to ℓ_1 by decreasing the energy level by -10.*

One can verify that the WA in Figure 8.1 is complete and deterministic. Thus, for all words w there is only one possible run of the automaton over w. Starting in the location ℓ_3 with the energy level 1, the run of the WA over the word $w = a^{10} \cdot b \cdot (c \cdot b)^2 \cdot d$ is

$$(\ell_3, 1) \xrightarrow{a:+1} (\ell_3, 2) \xrightarrow{a:+1} \cdots \xrightarrow{a:+1} (\ell_3, 11) \xrightarrow{b:-10} (\ell_1, 1) \xrightarrow{c:0} (\ell_2, 1)$$

$$\xrightarrow{b:+1} (\ell_1, 2) \xrightarrow{c:0} (\ell_2, 2) \xrightarrow{b:+1} (\ell_1, 3) \xrightarrow{d:-2} (\ell_0, 1).$$

◁

8.1.2 Minsky machine and vector addition systems with states

We recall the definitions of *Minsky machine* and *vector addition systems*, plus two decision problems that are used in this chapter.

Minsky machine

A Minsky machine has two non-negative counters c_0 and c_1, and it consists of a sequence of numbered instructions $1 : \mathsf{inst}_1$; $2 : \mathsf{inst}_2$; $\ldots$ $n : \mathsf{inst}_n$; where $\mathsf{inst}_n = \mathsf{halt}$ and other

instructions inst_i are an *increment* or a *guarded decrement*:

$$\texttt{(increment)}\ \ i : c_j := c_j + 1;\ \ \texttt{goto}\ k;$$
$$\texttt{(guarded decrement)}\ \ i :\ \ \texttt{if}\ c_j = 0\ \texttt{goto}\ k\ \texttt{else}\ (c_j := c_j - 1;\ \ \texttt{goto}\ k');$$

where $0 < i < n$, $0 < k, k' \leq n$ and $j \in \{0, 1\}$. A configuration is a tuple (i, v) where i is the current instruction and $v = (e_0, e_1)$ stores the values of the counters c_0, c_1 in e_0, e_1, accordingly. Minsky machines start in configuration $(1, (0, 0))$ and halt in halt.

Decision problem(s). The *halting problem* asks, given a Minsky machine, whether it halts.

It is well-known that the halting problem for (two-counter) Minsky machines is undecidable [Min67].

Vector addition systems with states

A *Vector addition system with states* (VASS) is a finite-state machine $\mathsf{VASS} = \langle Q, T \rangle$ where the transitions carry vectors of integers of some fixed dimension d. A configuration of a VASS is (s, v) where $s \in Q$ is a state and v is a d-dimension vector of non-negative integers. A transition $t : s \xrightarrow{v'} s'$ can be taken from the configuration (s, v) to $(s', v + v')$ if $v + v'$ is bigger than the $\mathbf{0}$-vector (the vector with 0 for all d dimensions).

Decision problem(s). The *coverability problem* asks, given a VASS with an initial configuration $(s_{\mathsf{init}}, v_{\mathsf{init}})$ and final configuration $(s_{\mathsf{end}}, v_{\mathsf{end}})$, whether starting from $(s_{\mathsf{init}}, v_{\mathsf{init}})$, a configuration (s, v) with $s = s_{\mathsf{end}}$ and $v \geq v_{\mathsf{end}}$ is reachable.

If the answer to coverability problem is positive and in particular such configuration (s, v) exists, we say that $(s_{\mathsf{end}}, v_{\mathsf{end}})$ is covered by $(s_{\mathsf{init}}, v_{\mathsf{init}})$. It is known that the coverability problem in VASSs is $\mathsf{EXPSPACE\text{-}complete}$ [Lip76, Rac78].

8.2 Synchronization in WA

We generalize the finite synchronizing words in NFAs, presented in Definition 3.1 on page 38, to WAs.

Definition 8.2 (Synchronizing and location-synchronizing words for WAs)**.** *Given a complete WA $\mathcal{A} = \langle L, E \rangle$ over the alphabet A and with the semantic (the labeled transition system) $[\![\mathcal{A}]\!] = \langle Q, R \rangle$, a word $w \in \mathsf{A}^+$ is* synchronizing *in the labeled transition system $\langle Q, R \rangle$ if $\mathsf{post}(Q, w)$ is a singleton, and it is* location-synchronizing *if $\mathsf{loc}(\mathsf{post}(Q, w))$ is a singleton.*

A synchronizing word can be read from every state and bring the WA to a single state. As weights are merely accumulated in WA, it is impossible to synchronize to a single state: consider two states (ℓ, e_1) and (ℓ, e_2) involving the same location ℓ but different initial energies $e_2 > e_1$. At least two of the runs starting from these two states go through the states involving the same location and the energy levels preserving the difference $e_2 - e_1$ (for deterministic WAs there are only two runs, but for non-deterministic WAs there might be more than two runs starting from those states). So two states (ℓ, e_1) and (ℓ, e_2) can never be synchronized, and thus WAs trivially never have a synchronizing word. Instead we define location-synchronizing words; inputting such words result in the WA reaching states that agree only on the location.

A location-synchronizing word brings the WA to a (set of states where all have a) single location with possibly different energy levels. In this setting, the location-synchronization in WAs is equivalent to synchronization of finite-state automata (i.e., weights play no role). Consequently, deciding whether, a given WA, has a location-synchronizing word is NLOGSPACE-complete if WA is deterministic, and is PSPACE-complete if WA is non-deterministic. A more realistic extension is to consider *safe synchronization* where we add a safety condition insisting that during synchronization the accumulated weight (energy) must be *safe*, e.g. a non-negative safety condition that requires the system to never run out of power while synchronizing. Considering the safety condition is what distinguishes our setting from the one presented in [FV13]; moreover, in that work WAs are restricted to have only non-negative weights on transitions.

Given $U \subseteq Q$, a word w is synchronizing (resp., location-synchronizing) in $\langle Q, R\rangle$ *with safety condition* U if

- $\mathsf{post}(U, w)$ is a singleton (resp., $\mathsf{loc}(\mathsf{post}(U, w))$ is a singleton), and
- $\mathsf{post}(U, v) \subseteq U$ for all prefixes v of w.

The safety condition U requires that the states in $Q \setminus U$ are never visited while reading the synchronizing word. Let $\mathcal{I}$ be the set of intervals with integer or infinite endpoints. We consider safety conditions of the form $\mathsf{Safe}\colon L \to \mathcal{I}$, therefore the safe set is $U = \{(\ell, x) \in Q \mid x \in \mathsf{Safe}(\ell)\}$. We denote an interval $[y, z]$ by $y \leq e \leq z$, an interval $[z, +\infty)$ by $e \geq z$, etc. where e is an energy variable.

A WA $\mathcal{A}$ is *(location-)synchronizing* under the safety condition if there exists some (location-)synchronizing word under the safety condition for it.

Decision problem(s). The *(location-)synchronizing problem* with a safety condition asks, given a WA $\mathcal{A}$ and a safety condition Safe, whether $\mathcal{A}$ is (location-)synchronizing under Safe.

Example 8.2. *Consider the WA $\mathcal{A}$ described in Example 8.1, and let the safety condition Safe be non-negative $e \geq 0$ for all four locations. We sometimes call such safety conditions an* energy condition.

In order to location-synchronize the automaton, we have to deal with infinitely many states (ℓ_i, e) where $e \in \mathbb{R}_{\geq 0}$. The only way to location-synchronize some state (ℓ_3, e) with states (ℓ, e') involving other locations $\ell \neq \ell_3$ is to input b. However, if b is provided initially, this will drop the energy level by -10 violating the non-negative safety condition for $(\ell_3, 0)$. Fortunately, the letter a recharges the energy level at ℓ_3 and has no negative effect at other locations. After reading $a^{10} \cdot b$, all states are location-synchronized in ℓ_0 and ℓ_1 with energy at least 0. Next, a d-transition can location-synchronize states involving ℓ_0 and ℓ_1, but it drops the energy level at ℓ_1 by -2. Again, we try to find a word that recharges the energy at ℓ_1. Supplying $c \cdot b$ twice makes a d-transition safe to be taken to location-synchronize safe states involving ℓ_0 and ℓ_1. So, the word $a^{10} \cdot b \cdot (c \cdot b)^2 \cdot d$ location-synchronizes the automaton with non-negative safety condition.

◁

As mentioned, it is impossible to synchronize WAs in the absence of a safety condition. In the presence of safety conditions, synchronization is also most-often impossible, for the same reason: the only exception is when safety condition is punctual. A *punctual safety condition* assigns at most one safe energy level to each location, thus there would not be two safe states (ℓ, e_1) and (ℓ, e_2) with the same location ℓ but different energy levels $e_1 \neq e_2$. The synchronization with a punctual safety condition in WAs is equivalent to synchronizing partial (not-complete) finite-state automata, which is PSPACE-complete [Mar10]. We thus focus on location-synchronizing problem with safety conditions; we establish complexity bounds for this problem for deterministic WAs. The location-synchronizing problem with safety conditions for non-deterministic WAs is left open, however we provide an example to explain why the used techniques for deterministic WA cannot be directly applied to the non-deterministic WAs (See Example 8.4).

8.3 Location-synchronization in deterministic WAs

We first solve the location-synchronizing problem for deterministic WAs with *lower-bounded safety condition*, which are safety conditions that assign conditions of the form $e \geq n$ with $n \in \mathbb{Z}$, to all locations. We provide tight complexity bounds for the location-synchronizing problem for deterministic WAs with lower-bounded safety condition. Next, we study this problem with *general safety condition* (with no constraints on the condition).

8.3.1 Location-synchronization under lower-bounded safety condition

In this subsection we assume that the safety condition is lower bounded where all the locations have some condition of the form $e \geq n$ with $n \in \mathbb{Z}$. This is equivalent to having only safety conditions of the form $e \geq 0$: it suffices to add $-n$ to the weight of all incoming transitions and to add $+n$ to the weight of outgoing transitions. As an instance, consider the WA $\mathcal{A}$ described in Example 8.1 and the transition $\ell_3 \xrightarrow{b:-10} \ell_1$. Let the safety constraint

for ℓ_1 be $e \geq 3$, and for ℓ_3 be $e \geq -7$. We change the weight of the transition to -20 when considering energy condition for both locations.

Below, we thus study the location-synchronizing problem with safety conditions of the form $e \geq 0$, which we call *non-negative safety conditions* or energy condition.

Lemma 8.1. *The location-synchronizing problem under non-negative safety condition in WAs, is in* PSPACE.

Proof. Let $\mathcal{A} = \langle L, E \rangle$ be a complete deterministic WA over the alphabet A where Z is the maximum value appearing as weight in transitions, in absolute. Runs starting from two states (ℓ, e_1) and (ℓ, e_2) with same location ℓ but two different energy levels $e_2 > e_1$, always go through the states involving the same locations and the energy levels preserving the difference $e_2 - e_1$. Therefore, to decide whether $\mathcal{A}$ is location-synchronizing under non-negative safety condition, it suffices to check if there is a word that synchronizes all locations with the initial energy $\mathbf{0}$, into a single location. We show that deciding whether such word w exists is in PSPACE by providing an upper bound for the length of w.

Below, we assume that $\mathcal{A}$ has a location-synchronizing word. For all subsets $S \subseteq L$ with cardinality $m > 2$, there is a word that synchronizes S into some strictly smaller set. To characterize the properties of such words, we consider the weighted digraph G_m where $m = |S|$. The digraph G_m is induced by the product between m copies of $\mathcal{A}$, where all vertices with at least two identical locations, are replaced with a new vertex synch. All ingoing transitions to some vertice $(\ell_1, \ldots, \ell_m)$ with at least two identical locations are redirected to synch. There is only a self-loop transition in synch. Formally, $G_m = \langle V_m, E_m \rangle$ is a weighted digraph where the set of vertices

$$V_m = \{(\ell_1, \ell_2, \ldots, \ell_m) \in L^m \mid \ell_i \neq \ell_j \leftrightarrow i \neq j\} \cup \{\mathsf{synch}\}$$

includes all m-tuples of distinct locations and another vertex synch.

- For all pairs of vertices $x, y \in V_m$, there is an edge from $x = (\ell_1, \ldots, \ell_m)$ to $y = (\ell'_1, \ldots, \ell'_m)$ with weight $\langle z_1, \ldots, z_m \rangle$ if there exists some letter a such that for all $0 < i \leq m$ we have $\ell_i \xrightarrow{a:z_i} \ell'_i$ in $\mathcal{A}$.
- For all vertices $x = (\ell_1, \ldots, \ell_m) \in V_m$, there is an edge from x to synch with weight $\langle z_1, \ldots, z_m \rangle$ if there exists some location ℓ and letter a such that for all $0 < i \leq m$ we have $\ell_i \xrightarrow{a:z_i} \ell$.

An edge with weight $\langle z_1, \ldots, z_m \rangle$ is *non-negative* (resp., *zero-effect*) if $z_i \geq 0$ for all dimensions $1 \leq i < m$ (resp., $z_i = 0$); and it is *negative* otherwise. A non-negative edge is *positive* if z_i is positive for some dimension i. There is a one-to-one correspondence between a path $x_0 x_1 \cdots x_n$ in G_m and a group of m runs $\rho^1 \ldots \rho^m$ in $\mathcal{A}$ such that all runs ρ^i are in shape of $\rho^i = \ell_0^i \cdots \ell_n^i$ where $x_j = (\ell_j^1, \ldots, \ell_j^m)$ for all $0 \leq j \leq n$. A path is *safe* if all corresponding m runs ρ^i starting from ℓ_0^i with energy level $\mathbf{0}$, always keep a non-negative energy level while going through all the locations $\ell_1^i \cdots \ell_n^i$ along the run.

The following claim is a key to compute an upper bound for the length of location-synchronizing words. Roughly speaking, it states that for all subsets S of locations, either there is a *short* word w that synchronizes S into a strictly smaller set, or there exists a

family of words $w_0 \cdot (w_1)^i$ ($i \in \mathbb{N}$) such that inputting the word $w_0 \cdot (w_1)^i$ accumulates energy i for the run starting in some location $\ell \in S$, while having non-negative effects along the runs starting from the other locations in S.

As an example consider the WA described in Examples 8.1 and 8.2. Since in the digraph G_2, there is no safe path from (ℓ_0, ℓ_2) to synch, there is a family of words $(b \cdot c)^i$ such that each iteration of $b \cdot c$ increase the energy level in ℓ_2 by 1.

Claim 1. *For all $1 < m \leq |L|$, for all vertices x of the digraph G_m, there is either a safe simple path from x to synch, or a simple cycle where all edges are non-negative and at least one is positive, which is reachable from x via a safe path.*

Proof of Claim 1. Since $\mathcal{A}$ has a location-synchronizing word, then from all vertices of the digraph G_m, there must be a safe path to synch. Take a vertex $x \in V_m$ and assume that all simple paths from x to synch are unsafe. Write G for the digraph obtained from G_m by removing all negative edges. Thus, there is no path from x to synch in G. Consider one of the bottom SCCs reached from x in G. Since there is no path from the bottom SCC to synch in G, we see that one of the edges in this bottom SCC must be positive. Otherwise, if all edges in this bottom SCC are zero-effect then for all vertices y of the bottom SCC, there is no way to synchronize the m-locations of y with initial energy 0. Thus the statement of claim holds, and the proof of Claim 1 is complete.

The next claim states that $\mathcal{A}$ has a location-synchronizing word if it has a *short* one, of length at most $\mathsf{Z}^{|L|} \times |L|^{3+|L|^2}$.

Claim 2. *For the synchronizing WA $\mathcal{A}$, there exists a short location-synchronizing word.*

Proof of Claim 2. The proof is by an induction: we prove that for all $2 \leq k \leq |L|$ and all subsets S of locations with $|S| = k$, there is a word w_S of $\mathsf{length}(k) \leq \mathsf{Z}^k \, |L|^{3+k^2}$ that location-synchronizes (under non-negative safety condition) the subset S into a single location.

Base case. We prove that for all subsets S of locations with $|S| = 2$, the length of the word w_S is at most $4\mathsf{Z}^2 \, |L|^6$. Let $S = \{\ell_1, \ell_2\}$ and consider the digraph G_2. If there is a safe simple path from (ℓ_1, ℓ_2) to synch, the base of induction trivially holds. Otherwise by Claim 1, for one of the locations in S, say ℓ_1, for all $i > 0$ there exists an *i-recharging* word $w_0 \cdot (w_1)^i$. Recall that an i-recharging word that recharges energy in ℓ_1 is a word such that inputting this word keeps the energy levels non-negative along the runs starting from the states in S; however it accumulates energy i along the run starting from ℓ_1. Let ℓ_1' and ℓ_2' be the locations reached from ℓ_1 and ℓ_2, accordingly, after inputting the i-recharging word with $i = \mathsf{Z}(2 \, |L|^2 + \mathsf{Z} \, |L|^4)$. A slightly different argument from the one used to prove Claim 1 gives us another word $w_2 \cdot (w_3)^j$ that recharges energy in run starting in ℓ_2' at least

to to an arbitrary value j by not considering the negative effect it may cause on the other run.

Let ℓ_1'' and ℓ_2'' be the locations reached from ℓ_1' and ℓ_2', accordingly, after inputting the word $w_2 \cdot (w_3)^j$ with $j = \mathsf{Z}\,|L|^2$. Let w_4 be a shortest synchronizing word for ℓ_1'' and ℓ_2'' in $\mathcal{A}$ by interpreting it as a DFA. We argue that the word $w = w_0 \cdot (w_1)^i \cdot w_2 \cdot (w_3)^j \cdot w_4$ is a location-synchronizing word for the subset S. By reading the word $w_0 \cdot (w_1)^i$, two states $(\ell_1, 0)$ and $(\ell_2, 0)$ go to $(\ell_1', e_1 + i)$ and (ℓ_2', e_2) where $e_1, e_2 \geq 0$. Since there is a high energy at ℓ_1', it can tolerate the negative effect caused after reading the word $w_2 \cdot (w_3)^j$. Next, reading $w_2 \cdot (w_3)^j$ leads two states $(\ell_1', e_1 + i)$ and (ℓ_2', e_2) to the states $(\ell_1'', e_3 + j)$ and $(\ell_2'', e_4 + j)$ where $e_3, e_4 \geq 0$. Both states $(\ell_1'', e_3 + j)$ and $(\ell_2'', e_4 + j)$ can tolerate the word w_4, and get synchronized while maintaining the energy level positive meaning that w is a location-synchronizing word for S. Hence, $\mathsf{length}(2) \leq 4\mathsf{Z}^2\,|L|^6$ and the base of the induction holds.

Inductive step. We prove the inductive step for n. Assume that for all $k < n$ and all subsets S' of locations with cardinality k, the statement of the induction holds, meaning that there is a word $w_{S'}$ of size $\mathsf{length}(k) \leq \mathsf{Z}^k\,|L|^{3+k^2}$ that location-synchronizes the subset S' into a single location. Let $S = \{\ell_1, \ell_2, \cdots, \ell_n\}$ and consider the digraph $G(n)$. If there is a safe simple path from $(\ell_1, \ell_2, \cdots, \ell_n)$ to synch, the step of the induction trivially holds. Otherwise by Claim 1, for one of the locations in S, say ℓ_n, for all $i > 0$ there exists an *i-recharging* word $w_0 \cdot (w_1)^i$. A location-synchronizing word can start with a $[\mathsf{Z} \cdot \mathsf{length}(n-1)]$-recharging word to accumulate at least $\mathsf{Z} \cdot \mathsf{length}(n-1)$ energy in the run starting from ℓ_n. For all $0 < i \leq n$, let ℓ_i' be the location reached from ℓ_i after reading the $[\mathsf{Z} \cdot \mathsf{length}(n-1)]$-recharging word. Then, the location-synchronizing word can be followed with a word $w_{S'}$ that location-synchronizing word $S' = \{\ell_1', \cdots, \ell_{n-1}'\}$ because it has already accumulated enough energy at ℓ_n' to tolerate the negative effect caused while synchronizing the other states, even if all taken transitions decrease the energy level of the run starting in ℓ_n' by the maximum negative weight Z. Let the subset S' get synchronized in the location x, and y be the location reached from ℓ_n' by reading $w_{S'}$. At last, the location-synchronizing word must only synchronize x and y. Thus,

$$\mathsf{length}(n) \leq |L|^n\,[2 + \mathsf{Z} \cdot \mathsf{length}(n-1)] + \ \mathsf{length}(2).$$

So, $\mathsf{length}(n) \leq \mathsf{Z}^n\,|L|^{3+n^2}$ and the induction holds. Thus $\mathcal{A}$ has a location-synchronizing word with length at most $\mathsf{length}(|L|)$. This concludes the proof of Claim 2.

Claim 2 proves that $\mathcal{A}$ has a location-synchronizing word if it has one of length at most $\mathsf{Z}^{|L|} \times |L|^{3+|L|^2}$. Since this value can be stored in polynomial space, an (N)PSPACE algorithm can decide whether the given WA $\mathcal{A}$ is location-synchronizing.

□

We use an intuitive reduction from synchronizing problem for partial finite automaton to establish the matching lower bound.

Lemma 8.2. *The location-synchronizing problem under non-negative safety condition in WAs, is* PSPACE-hard.

Proof. To show PSPACE-hardness, we use a reduction from synchronizing word problem for deterministic finite automata with partially defined transition function that is PSPACE-complete [Mar10].

From a partial finite state automaton $\mathcal{N} = \langle Q, \mathsf{A}, \Delta \rangle$, we construct a WA $\mathcal{A}$ over the alphabet A such that $\mathcal{N}$ has a synchronizing word if and only if $\mathcal{A}$ has a location-synchronizing word under an energy condition.

The construction of $\mathcal{A} = \langle L, E \rangle$ is as follows.

- All defined transitions of $\mathcal{N}$ are augmented with the weight 0 in $\mathcal{A}$: for all states $q \in Q$ and letters $a \in \mathsf{A}$, let $q \xrightarrow{a:0} q'$ if $\Delta(q, a) = q'$ in the automaton $\mathcal{N}$.
- To complete $\mathcal{A}$, all non-defined transitions are added as self-loops with weight -1: for all states $q \in Q$ and letters $a \in \mathsf{A}$, let $q \xrightarrow{a:-1} q$ if the a-transition in q is not defined in $\mathcal{N}$.

Since the safety condition is non-negative in all locations, none of the transitions with weight -1 are allowed to be used along synchronization in $\mathcal{A}$. So $\mathcal{N}$ has a synchronizing word if and only if $\mathcal{A}$ has a location-synchronizing word.

□

From previous Lemmas and arguments, Theorem 8.1 follows.

Theorem 8.1. *The location-synchronizing problem under lower-bounded safety conditions in WAs, is* PSPACE-complete.

We generalize the location-synchronizing problem to *location-synchronization from a subset*, where the aim is to synchronize a given subset of locations. This variant is used to decide location-synchronization under general safety condition. Given a subset $S \subseteq L$ of locations, we prove Lemma 8.3 and Lemma 8.4 using reductions to and from the coverability problem in vector-addition systems.

Lemma 8.3. *The location-synchronizing problem from a subset S of locations under non-negative safety conditions in WAs, is decidable in* 2-EXPSPACE.

Proof. We present a construction to reduce location-synchronizing problem from a subset S of locations under non-negative safety conditions in WAs, to the coverability problem in vector addition systems. The construction is exponential in the size of the WA, and by the fact that the coverability problem is in EXPSPACE, the 2-EXPSPACE membership of location-synchronizing problem from a subset S of locations under non-negative safety conditions follows.

Given a WA $\mathcal{A} = \langle L, E\rangle$ over an alphabet A, a subset $S \subseteq L$ of locations and a non-negative safety condition Safe, we construct a VASS with two configurations $(s_{\mathsf{init}}, v_{\mathsf{init}})$ and $(s_{\mathsf{end}}, v_{\mathsf{end}})$ such that the automaton $\mathcal{A}$ under Safe condition has a location-synchronizing word from S if and only if the configuration $(s_{\mathsf{end}}, v_{\mathsf{end}})$ is covered from $(s_{\mathsf{init}}, v_{\mathsf{init}})$ in VASS.

The encoding is straightforward. Let the set S have cardinality $|S| = m$, we thus write $S = \{p_1, \cdots, p_m\}$. We construct the vector addition system $\mathsf{VASS} = \langle Q, T\rangle$ with vectors of dimension m as follows.

- The state space $Q = L^m \cup \{s_{\mathsf{end}}\}$ where L^m is all m-tuples of locations.
- From a state $s = (\ell_1, \cdots, \ell_m)$ there is a transition to $s' = (\ell'_1, \cdots, \ell'_m)$ carrying the vector $v = (z_1, \cdots, z_m)$, if for some $a \in \mathsf{A}$ and all $0 < i \leq m$ we have $\ell_i \xrightarrow{a,z_i} \ell'_i$.
- From all states $s = (\ell, \cdots, \ell)$ with the identical location ℓ in all components, there is a transition to s_{end} with $\mathbf{0}$-vector.

Let $s_{\mathsf{init}} = (p_1, \cdots, p_m)$ and both vectors v_{init} and v_{end} be $\mathbf{0}$-vectors. There is a one-to-one corresponding between a sequence of transitions $t_0 t_1 \cdots t_n$ in VASS with a word $w = a_0 a_1 \cdots a_n$ in $\mathcal{A}$. Since the only way to reach s_{end} is via states $(\ell, \cdots, \ell)$ with the identical components ℓ and positive energy level, we can easily see that the configuration $(s_{\mathsf{end}}, v_{\mathsf{end}})$ is covered from $(s_{\mathsf{init}}, v_{\mathsf{init}})$ if and only if there is a location-synchronizing word w from S in $\mathcal{A}$. This construction uses exponential space in size of $\mathcal{A}$, then it proves that location-synchronizing problem is decidable in $\mathsf{2\text{-}EXPSPACE}$.

□

Lemma 8.4. *The location-synchronizing problem from a subset S of locations under lower-bounded safety conditions in WAs, is* $\mathsf{EXPSPACE\text{-}hard}$.

Proof. The $\mathsf{EXPSPACE\text{-}hardness}$ proof is by a reduction from the coverability problem in vector addition systems. From a given $\mathsf{VASS} = \langle Q, T\rangle$ equipped with two configurations $(s_{\mathsf{init}}, v_{\mathsf{init}})$ and $(s_{\mathsf{end}}, v_{\mathsf{end}})$, we construct a WA $\mathcal{A}$, a lower-bounded safety condition Safe and a set S of locations such that $(s_{\mathsf{end}}, v_{\mathsf{end}})$ is covered from $(s_{\mathsf{init}}, v_{\mathsf{init}})$ if and only if the automaton $\mathcal{A}$ under the Safe condition has a location-synchronizing word from S.

The intuition behind the reduction is that we add d copies of VASS to $\mathcal{A}$ such that the weights in the i-th copy is taken from the i-th dimension of vectors in VASS. An absorbing location called synch is added that is only reachable from the locations s_{end} of each copy. The set S contains synch and the locations s_{init} of all copies, $\mathcal{A}$ thus can only be location-synchronized in synch. Therefore, to location-synchronize S all d copies must run in parallel and try to reach copies of s_{end}.

We assume that all states s of VASS have exactly the same number m of outgoing transitions (otherwise we add self-loops with $\mathbf{0}$-vectors). We consider m letters $a_1, a_2, \cdots, a_m$ and label each outgoing transition t in s with a unique letter a_i where $0 < i \leq m$ (all pairs of outgoing transitions have different labels). The construction of $\mathcal{A} = \langle L, E\rangle$ over A is as follows.

- The alphabet is $\mathsf{A} = \{a_1, \cdots, a_m, \star, \#\}$.

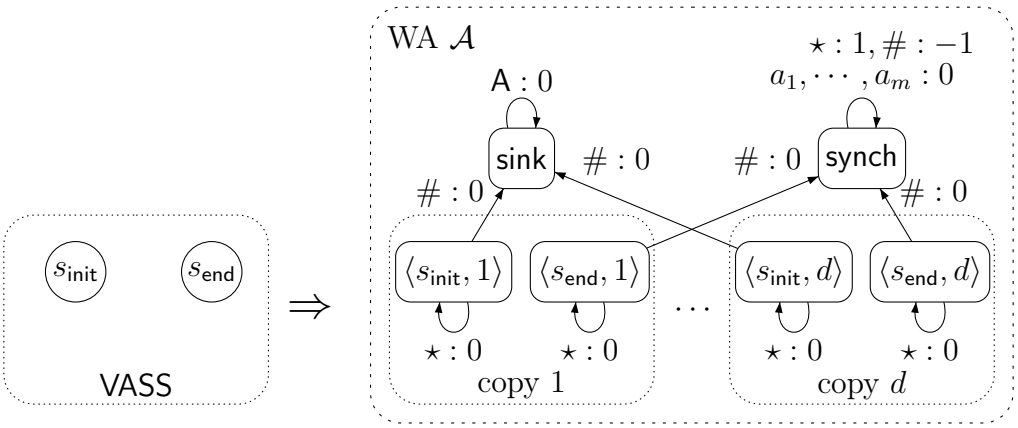

Figure 8.2: (Schematic) reduction from the coverability problem in vector addition systems with states to location-synchronizing problem from a subset S of locations under lower-bounded safety conditions in WAs.

- The set of locations $L = Q \times \{1, 2, \cdots, d\} \cup \{\mathsf{synch}, \mathsf{sink}\}$ includes d copies of VASS and two new locations synch and sink. A state $\langle s, i\rangle$ is in the i-th copy of VASS.
- For all letters $a \in \mathsf{A}$, the a-transition in both locations synch and sink are self-loops implying that those locations are absorbing. The weights of all those transitions are 0 except # and $\star$-transitions in synch:

$$\mathsf{synch} \xrightarrow{\star:+1} \mathsf{synch} \qquad \text{and} \qquad \mathsf{synch} \xrightarrow{\#:-1} \mathsf{synch}.$$

- For all transitions $s \xrightarrow{v} s'$ in VASS that is labeled by a and the vector $v = \langle z_1, \cdots, z_d\rangle$, there are d transitions in $\mathcal{A}$ such that for all $0 < i \leq d$:

$$\langle s, i\rangle \xrightarrow{a:z_i} \langle s', i\rangle$$

- Let $v = v_{\mathsf{end}} - v_{\mathsf{init}}$ and write $v = \langle z_1, z_2, \cdots, z_d\rangle$. For all $s \in Q$ and $0 < i \leq d$, the #-transition is directed to sink:

$$\langle s, i\rangle \xrightarrow{\#:0} \mathsf{sink}$$

  except in s_{end} where the #-transition goes to synch:

$$\langle s_{\mathsf{end}}, i\rangle \xrightarrow{\#:-z_i+1} \mathsf{synch}.$$

- For all $s \in Q$ and $0 < i \leq d$, the $\star$-transition in $\langle s, i\rangle$ is a self-loop with weight 0.

The construction is depicted in Figure 8.2. Let $S = \{\langle s_{\mathsf{init}}, i\rangle \mid 0 < i \leq d\} \cup \{\mathsf{synch}\}$ be such that the location-synchronizing from S is of interest. The safety condition is non-negative for all locations except in synch where $\mathsf{Safe}(\mathsf{synch}) = \{e \geq 1\}$.

First, assume that $(s_{\mathsf{end}}, v_{\mathsf{end}})$ is covered from $(s_{\mathsf{init}}, v_{\mathsf{init}})$ in VASS. Thus, there is a sequence of transitions $t_0 t_1 \cdots t_n$ that are taken from $(s_{\mathsf{init}}, v_{\mathsf{init}})$ to cover $(s_{\mathsf{end}}, v_{\mathsf{end}})$. Let $w = a_0 \cdot a_1 \cdots a_n$ be the sequence of the labels of those transitions $t_0 t_1 \cdots t_n$ where a_i is the label of t_i for all $0 \leq i \leq n$. The word $w \cdot \star \cdot \#$ reaches the location synch with non-negative energy level, no matter its origin (in S) and the initial energy. So $\mathcal{A}$ has a location-synchronizing word under the condition Safe.

Second, assume that $\mathcal{A}$ has a location-synchronizing word w. Since synch is absorbing, and since synch $\in S$ then $\mathcal{A}$ is location-synchronized in synch. Entering synch is only possible by reading #, so it must be the case that w contains some #. Moreover, reading # in synch is only possible after at least one $\star$; otherwise the safety condition in the location synch is violated (with energy level 1). Let w' be the subword of w that is obtained by omitting all letters $\star$ from the prefix of w until the first #. Thus, the word w' has neither # nor $\star$. The first $\star$ increases the energy level at location synch by 1, and thus the #-transition becomes affordable in synch (to location-synchronize other states in synch). Since from all locations $\langle s, i \rangle$ where $s \neq s_{\mathsf{end}}$, the #-transitions lead to sink where there is no way to synchronize, the first # of w must be read when all copies of VASS are in the locations $\langle s_{\mathsf{end}}, i \rangle$ where $0 < i \leq d$. Let $v = v_{\mathsf{end}} - v_{\mathsf{init}}$ and write $v = \langle z_1, \cdots, z_d \rangle$. The location $\langle s_{\mathsf{end}}, i \rangle$ needs to have at least energy level z_i to afford reading # (and not violate the safety condition in synch). Therefore the sequence of transitions in VASS corresponding to the word w', starting from the configuration $(s_{\mathsf{init}}, v_{\mathsf{init}})$ must end up in a configuration that covers $(s_{\mathsf{end}}, v_{\mathsf{end}})$. The above argument shows the correctness of the presented reduction to establish the EXPSPACE-hardness.

□

Lemma 8.3 and Lemma 8.4 provide reductions to and from the coverability problem in vector addition systems to establish the following result.

Theorem 8.2. *The location-synchronizing problem from a subset S of locations under lower-bounded safety conditions in WAs, is decidable in* 2-EXPSPACE*, and it is* EXPSPACE-hard.

8.3.2 Location-synchronization under general safety condition

We now discuss location-synchronization under the *general* safety condition where the energy constraint for each location can be a bounded interval, lower or upper-bounded, or trivial (always `true`). We proceed in two steps: first, we prove that the PSPACE-completeness results in case of non-negative safety condition is preserved in location-synchronization under safety condition with only lower-bounded or trivial constraints. Second, we extend our techniques to establish results for general safety conditions. To obtain results for the general case, we use the variant *location-synchronization from a subset*, that is discussed in all cases as well.

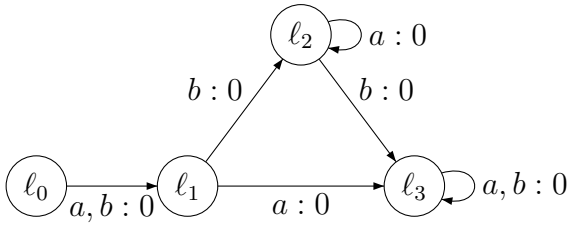

Figure 8.3: To location-synchronize the automaton with $L_{\mapsto} = \{\ell_0, \ell_2\}$ and $L_{\leftrightarrow} = \{\ell_1, \ell_3\}$, taking the back-edge $\ell_1 \xrightarrow{b:0} \ell_2$ is avoidable.

Location-synchronization under lower-bounded or trivial safety conditions.

Let the safety condition Safe assign to each location of L either an interval of the form $[n, +\infty)$ or `true`. Accordingly, we partition the set L of locations into two classes $L_{\mapsto}$ and $L_{\leftrightarrow}$ such that $L_{\leftrightarrow}$ is the set of all locations with `true` safety constraints, and $L_{\mapsto}$ is the set of all locations with a lower-bounded safety constraints. A *back-edge* is a transition $\ell \to \ell'$ that goes from some location $\ell \in L_{\leftrightarrow}$ with `true` safety constraints to some location $\ell' \in L_{\mapsto}$ with lower-bounded safety constraints. An important observation is that while location-synchronizing, all runs starting from some location $\ell \in L_{\leftrightarrow}$ with `true` safety constraints can never pass through some location $\ell' \in L_{\mapsto}$; otherwise, the runs starting in some state (ℓ, e) where $e < 0$ is a sufficiently small integer, even after taking several transitions with positive weights to increase the energy level, arrive in the location ℓ' with negative energy level violating the safety condition. Those back-edge transitions though might be fired along the runs starting from locations with lower-bounded safety constraints. See Example 8.3.

Example 8.3. *Consider the WA drawn in Figure 8.3 with four locations and two letters. All transitions have weight* 0*, and the WA is deterministic. From the state ℓ_0, all transitions bring the WA into ℓ_1. The a-transition in the state ℓ_1 goes to ℓ_3; and the b-transition goes to ℓ_2. The b-transition leaves ℓ_2 and moves to ℓ_3 while the a-transition stays there. The state ℓ_3 is absorbing. The safety condition is non-negative in ℓ_0 and ℓ_2, and is trivial in ℓ_1 and ℓ_3: $L_{\mapsto} = \{\ell_0, \ell_2\}$ and $L_{\leftrightarrow} = \{\ell_1, \ell_3\}$. Thus, the transition $\ell_1 \xrightarrow{b:0} \ell_2$ is a back-edge.*

The word $a \cdot b \cdot b$ is a location-synchronizing word that takes the back-edge $\ell_1 \xrightarrow{b,0} \ell_2$ along the following runs starting in ℓ_0:

$$(\ell_0, e) \xrightarrow{a:0} (\ell_1, e) \xrightarrow{b:0} (\ell_2, e) \xrightarrow{b:0} (\ell_3, e)$$

where $e \geq 0$.

While synchronizing by any location-synchronizing word, the back-edge $\ell_1 \xrightarrow{b:0} \ell_2$ cannot be used for the runs starting in ℓ_1; otherwise the run $(\ell_1, -1) \xrightarrow{b:0} (\ell_2, -1)$ would violate the non-negative safety condition at ℓ_2. In this example, there exists an alternative word $a \cdot a \cdot b$ that takes no back-edges and still location-synchronizes the automaton.

$\triangleleft$

In Example 8.3, we see that even though a back-edge might be taken along runs over some location-synchronizing words such as $a \cdot b \cdot b$, there are alternative location-synchronizing words that does not fire a back-edge transition, for instance the word $a \cdot a \cdot b$. We prove, by Lemma 8.5, that such alternative words always exist implying that taking back-edge transitions while synchronizing is avoidable in deterministic WAs.

Lemma 8.5. *Let $\mathcal{A}$ be a deterministic WA and Safe be a safety condition with only lower-bounded or trivial constraints. There is a location-synchronizing word in $\mathcal{A}$ if and only if there is one in the automaton obtained from $\mathcal{A}$ by removing all back-edge transitions.*

Proof. First direction is trivial. To prove the reverse, assume that $\mathcal{A}$ has a location-synchronizing word w under Safe, such that there exists some state starting in which the run over w takes a back-edge, say the back-edge $\ell_1 \xrightarrow{b} \ell_2$ with $\ell_1 \in L_{\leftrightarrow}$ and $\ell_2 \in L_{\mapsto}$. We denote by $[n, +\infty)$ the lower-bounded safety constraint in ℓ_2. Let (ℓ_0, e_0) be some state such that starting from (ℓ_0, e_0), the run over w takes the back-edge $\ell_1 \xrightarrow{b} \ell_2$ earliest (among all the runs starting from all safe states). We split the word $w = w_0 \cdot b \cdot w_1$ such that b is the letter firing the back-edge $\ell_1 \xrightarrow{b} \ell_2$ in the run starting from (ℓ_0, e_0) first. We prove that $\ell_0 \in L_{\mapsto}$ meaning that ℓ_0 has lower-bounded safety condition. Towards contradiction, assume that $\ell_0 \in L_{\leftrightarrow}$. Since w location-synchronizes all safe states, then the runs starting from all states (ℓ_0, e) where $e \in \mathbb{Z}$ would never violate the safety condition. Let $e = n - \mathsf{Z} \cdot |w_0| - 1$ where Z is the maximum value appearing as weight in transitions in absolute, and where n is the minimum allowed energy level at the location ℓ_2. Inputting the word $w_0 \cdot b$ from the state (ℓ_0, e) brings $\mathcal{A}$ to the state (ℓ_2, n') with $n' < n$, violating the safety condition at ℓ_2. It contradicts with the fact that w is a location-synchronizing word under Safe. It proves then $\ell_0 \in L_{\mapsto}$.

We denote by U the set of all safe states: $U = \{(\ell, e) \mid e \in \mathsf{Safe}(\ell)\}$. Let $S = \mathsf{loc}(\mathsf{post}(U, w_0))$ be the set of all reached locations after inputting w_0. Since the runs starting from the location $\ell_0 \in L_{\mapsto}$ are ending in $\ell_1 \in L_{\leftrightarrow}$ by the word w_0, and since w_0 does not fire any back-edge transition, then $S \cap L_{\mapsto} \subset L_{\mapsto}$ meaning that the number of locations with a lower-bounded safety constraint is decreased after reading w_0. Since the word w_0 does not violate the safety condition and keeps the automaton in the safe set $\mathsf{post}(U, w_0) \subseteq U$, then $w_0 \cdot w$ is also a location-synchronizing word. Therefore, $w_0 \cdot w$ is a location-synchronizing word such that there are less runs over it firing a back-edge, compared to the number of runs over w. We can repeat the above argument for all back-edges (at most $|L_{\mapsto}|$ times) that are taken while synchronizing $\mathcal{A}$ by w, and construct a location-synchronizing word $w' \cdot w$ such that no back-edge transition is fired while reading w'. Moreover, all reached locations after reading w' have trivial safety constraints $\mathsf{loc}(\mathsf{post}(U, w') \subseteq L_{\leftrightarrow}$. Hence, the word $w' \cdot w$ is a location-synchronizing word that no runs starting from some safe state over it takes a back-edge transition. It completes the proof.

□

Lemma 8.5 does not hold when synchronizing from a subset S of the locations. Indeed, consider the one-letter automaton drawn in Figure 8.4. The automaton has three states,

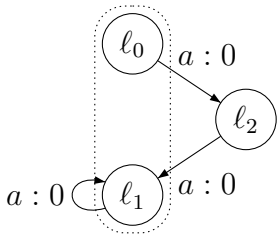

Figure 8.4: To location-synchronize the automaton from $S = \{\ell_0, \ell_1\}$ where $L_{\mapsto} = \{\ell_0, \ell_1\}$ and $L_{\leftrightarrow} = \{\ell_2\}$, taking the back-edge $\ell_2 \xrightarrow{a:0} \ell_1$ is unavoidable.

but we are only interested to location-synchronize it from two locations $S = \{\ell_0, \ell_1\}$. The safety condition for those two locations is non-negative $L_{\mapsto} = S$, however the safety constraint at ℓ_2 is trivial. Obviously, it is possible to location-synchronize from S, and this would not be possible without taking the back-edge $\ell_2 \xrightarrow{a:0} \ell_1$.

Lemma 8.5 also fails for non-deterministic WAs. We provides a non-deterministic WA in Example 8.4 such that for all location-synchronizing word, a back-edge transition must be fired.

Example 8.4. *Consider the non-deterministic WA $\mathcal{A}$ depicted in Figure 8.5.a. The automaton $\mathcal{A}$ has four states and three letters. All transitions have weight* 0*, except two transitions. In the state ℓ_0, the a-transitions non-deterministically go to either ℓ_0 or ℓ_2. The b-transition is a self-loop and the c-transition goes to ℓ_3 that is an absorbing state. In the location ℓ_1, the a-transition is a self-loop whereas b and c-transitions go to ℓ_0: $\ell_1 \xrightarrow{b:0} \ell_0$ and $\ell_1 \xrightarrow{c:-1} \ell_0$. All transitions in ℓ_2 leave the location: $\ell_2 \xrightarrow{a,c:0} \ell_3$ and $\ell_2 \xrightarrow{b:+1} \ell_1$*

The safety condition Safe *for locations in $L_{\mapsto} = \{\ell_0, \ell_1\}$ is non-negative, and for locations in $L_{\leftrightarrow} = \{\ell_2, \ell_3\}$ is trivial. Therefore, the transition $\ell_2 \xrightarrow{b:+1} \ell_1$ is a back-edge; we prove that there is no way to location-synchronize $\mathcal{A}$ unless taking the back-edge $\ell_2 \xrightarrow{b:+1} \ell_1$.*

Assume that $w = a_0 \cdot a_1 \cdots a_n$ is a location-synchronizing word. Since ℓ_3 is an absorbing state, so $\mathcal{A}$ is synchronized into ℓ_3. The first letter a_0 of w

- *cannot be b because $(\ell_2, -2) \xrightarrow{b:+1} (\ell_1, -1)$ violating the non-negative safety condition at the location ℓ_1,*
- *cannot be c because $(\ell_1, 0) \xrightarrow{c:-1} (\ell_0, -1)$ violating the non-negative safety condition at the location ℓ_0.*

Thus, w starts with the letter a (See Figure 8.5.b.). Inputting a shrinks the safe set

$$U = \{(\ell_i, e) \mid 0 \leq i \leq 1, e \in \mathbb{N}\} \cup \{(\ell_i, e) \mid 2 \leq i \leq 3, e \in \mathbb{Z}\}$$

into the set

$$U_1 = \{(\ell_i, e) \mid 0 \leq i \leq 2, e \in \mathbb{N}\} \cup \{(\ell_3, e) \mid e \in \mathbb{Z}\}.$$

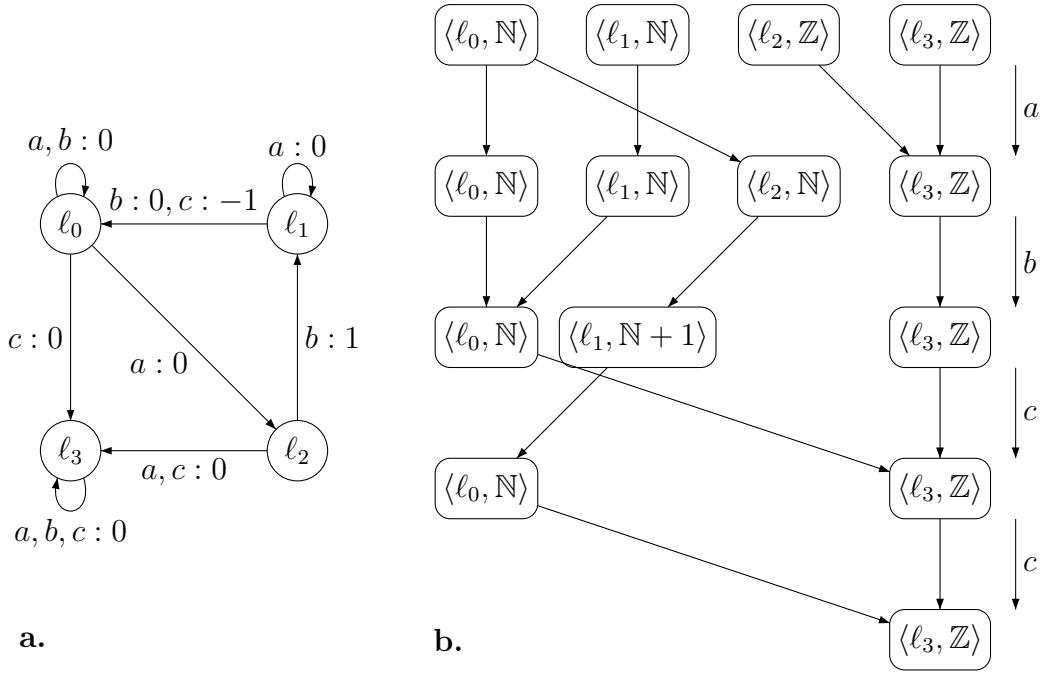

Figure 8.5: Unavoidable back-edges to synchronize non-deterministic WAs.

Inputting another a is safe but it would not shrink the set U_1, and in particular would not help in synchronizing. The same reason as before, the second letter a_1 of w cannot be c. Inputting b shrinks the set U_1 into

$$U_2 = \{(\ell_0, e) \mid e \in \mathbb{N}\} \cup \{(\ell_1, e) \mid e \geq 1\} \cup \{(\ell_3, e) \mid e \in \mathbb{Z}\}$$

and fires the back-edge $\ell_2 \xrightarrow{b:+1} \ell_1$. It remains to ensure that there is indeed a way of location-synchronizing into the location ℓ_3, which is inputting c twice.

◁

By Lemma 8.5, location-synchronizing a given WA $\mathcal{A}$ under a safety condition with only lower-bounded or trivial constraints can be achieved in two steps:

- first location-synchronizing all the states with lower-bounded safety constraints into the set of states with trivial constraints. It can be done in PSPACE by using Theorem 8.1,
- second location-synchronizing the states with trivial safety constraints where the weights play no role.

Lemma 8.6. *The location-synchronizing problem in WAs under lower-bounded or trivial safety condition is* PSPACE-complete.

Proof. Let $\mathcal{A} = \langle L, E \rangle$ be a complete deterministic WA over the alphabet A where Z is the maximum value appearing as weight in transitions in absolute. Let Safe be the safety

condition with only lower-bounded or trivial constraints. By Lemma 8.5, we can assume that $\mathcal{A}$ has no back-edge. We also assume that $L_{\leftrightarrow}$ is not empty.

Since there is no back-edge in $\mathcal{A}$, the automaton is synchronized in some location $\ell \in L_{\leftrightarrow}$ with trivial safety constraint. Thus, there exists some word w such that by reading this word w, the set of states (ℓ, e) with some location $\ell \in L_{\mapsto}$ having a lower-bounded safety constraint end up in the set $L_{\leftrightarrow}$; formally, $\mathsf{loc}(\mathsf{post}(S, w)) \subseteq L_{\leftrightarrow}$ where $S = \{(\ell, e) \mid \ell \in L_{\mapsto}, e \in \mathsf{Safe}(\ell)\}$. After reading such words, weights play no role while location-synchronizing the reached set $\mathsf{post}(S, w)$ of states. Thus, we propose following PSPACE algorithm:

- We obtain the WA $\mathcal{A}'$ from $\mathcal{A}$ by replacing all locations $\ell \in L_{\leftrightarrow}$ with some absorbing location synch. All ingoing-transition to some location $\ell \in L_{\leftrightarrow}$ is redirected to synch. At the location synch, we define the safety constraint $e \geq m - \mathsf{Z}$ where m is the minimum allowed energy level for all locations with a lower-bounded safety constraints:

$$m = \min\{e \mid \text{ there exists } \ell \in L_{\mapsto} \text{ such that } e \in \mathsf{Safe}(\ell)\}.$$

  We choose $m - \mathsf{Z}$ as the minimum allowed energy level at synch since the least energy level for locations in $L_{\mapsto}$ is m, and the energy level cannot be decreased more than Z by taking the transitions going to synch . We find a location-synchronizing word w for $\mathcal{A}'$ by using Lemma 8.1.
- We obtain the DFA $\mathcal{N}$ from $\mathcal{A}$ by removing all locations $\ell \in L_{\mapsto}$ and omitting the weights of transitions. We find a synchronizing word v for the DFA $\mathcal{N}$.

By above arguments, the word $w \cdot v$ is a location-synchronizing word for $\mathcal{A}$. The proof is complete.

□

The proof of Lemma 8.6 carries on for synchronizing from a subset of locations, except using Lemma 8.3 instead of Theorem 8.1, and requiring that the automaton has no back-edge.

Lemma 8.7. *Given a WA $\mathcal{A}$ with no back-edges, the location-synchronizing problem from a subset S of locations under lower-bounded or trivial safety condition is decidable in* 2-EXPSPACE*, and it is* EXPSPACE-hard*.*

Location-synchronization under general safety conditions.

Let us relax the restrictions on the safety condition Safe, and let some of the locations have bounded intervals to indicate the safe range of energy. The set L of locations is partitioned into $L_{\mapsto}$, $L_{\mapsto}$ and $L_{\leftrightarrow}$ where locations in $L_{\mapsto}$ have safety conditions such as $e \in [n_1, n_2]$ where $n_1, n_2 \in \mathbb{Z}$. In this setting, transitions from locations in $L_{\mapsto}$ or $L_{\leftrightarrow}$ to locations in $L_{\mapsto}$ are considered as *back-edge* too. Since taking back-edge transitions while synchronizing from a subset S of locations is not avoidable, we can use bounded

safety conditions to establish a reduction from the halting problem in Minsky machines to provide the following undecidability result.

Theorem 8.3. *The location-synchronizing problem from a set S of locations in WAs under a general safety condition is undecidable.*

Proof. The proof is by a reduction from the halting problem in two-counter Minsky machine. Below, without loss of generality, we assume that the Minsky machine has at least two guarded decrement instructions, one on each counter.

From a Minsky machine, we construct a deterministic WA $\mathcal{A}$, a general safety condition Safe and a subset S such that Minsky machine halts if and only if $\mathcal{A}$ has a location-synchronizing word from S under the condition Safe. The automaton $\mathcal{A}$ is constructed from two disjoint automaton $\mathsf{B_0}$ and $\mathsf{B_1}$ (and some other locations such as synch) with the same number of locations and transitions. A run over automaton $\mathsf{B_j}$ simulate the value of counter c_j along a sequence of configurations in the Minsky machine. The only way to synchronize these two disjoint automata is arriving to their halt, simultaneously, and then play a special letter # to reach the location synch. To let the counters to get freely any non-negative value, the safety condition in all location of $\mathsf{B_0}$ and $\mathsf{B_1}$ are non-negative except some particular locations that are reserved to check the correctness of a guarded decrement. The guarded decrement instructions are simulated by taking a back-edge and visiting locations with the safety condition of form $e = 0$. To synchronize all locations with different kind of safety conditions, we add a gadget that forces all location-synchronizing words for $\mathcal{A}$ to always begin with the letter $\star$.

The construction of $\mathcal{A}$ over the alphabet A is as follows. An instance of reduction is depicted in Figure 8.6.

- The alphabet has four letters $\mathsf{A} = \{a, b, \#, \star\}$.
- For each increment instructions $i : c_j := c_j + 1;$ `goto` k; we have two locations i_j, k_j and the transitions $i_j \xrightarrow{c,1} k_j$ in $\mathsf{B_j}$ and two locations i_{1-j}, k_{1-j} and the transitions $i_{1-j} \xrightarrow{c,0} k_{1-j}$ in $\mathsf{B_{1-j}}$ where $c \in \{a, b\}$. See instructions 1 and 3 in Figure 8.6 and the corresponding locations and transitions in Figure 8.7.
- For each guarded decrement instructions $i :$ `if` $c_j = 0$ `goto` k `else` $(c_j := c_j - 1;$ `goto` $k')$; we have four locations $i_j, k_j, k'_j, f_{i,j}$ and the transitions

$$i_j \xrightarrow{b:0} f_{i,j}\,,\ f_{i,j} \xrightarrow{c:0} k_j\,,\ i_j \xrightarrow{a:-1} k'_j$$

  in $\mathsf{B_j}$ and similarly, four locations $i_j, k_j, k'_j, f_{i,j}$ and following transitions

$$i_{1-j} \xrightarrow{a:0} k_{1-j}\,,\ i_{1-j} \xrightarrow{b:0} f_{i,1-j}\,,\ f_{i,1-j} \xrightarrow{c:0} k'_{1-j}$$

  in $\mathsf{B_{1-j}}$ where $c \in \{a, b\}$. See instruction 4 in Figure 8.6 and the corresponding locations and transitions in Figure 8.7.

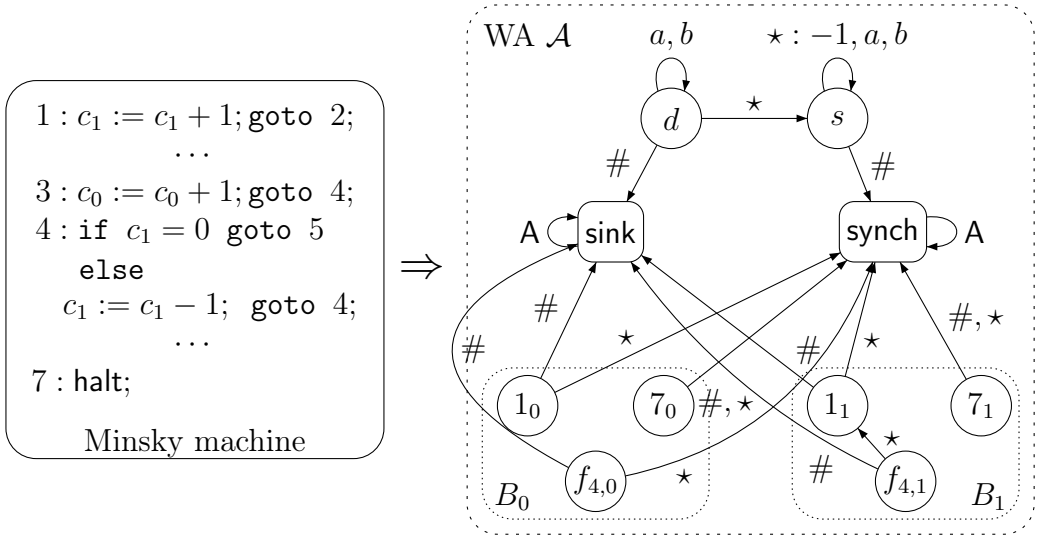

Figure 8.6: (Schematic) reduction from the halting problem in Minsky machines to location-synchronizing problem from a subset S of locations under general safety conditions in WAs. To simplify the figure, the weights of transitions with weight 0 are omitted. See Figure 8.7 for the details of B_0 and B_1.

- We add two new absorbing locations synch and sink meaning that all transitions are self-loops with weight 0.
- We have a gadget with two new locations "departure" d and "stay" s, and following transitions:
$$d \xrightarrow{c:-1} d\,,\ d \xrightarrow{\star:0} s\,,\ s \xrightarrow{\star:-1} s\,,\ s \xrightarrow{c:0} s$$
where $c = \{a, b\}$.
- From the stay location s, n_0 and n_1 where the n-th instruction is halt, the #-transition goes to synch with weight 0, but from all other locations the #-transition goes to the location sink.
- From all locations in $\mathsf{B_j}$, the $\star$-transition leads to synch except in locations $f_{i,j}$ if the instruction i is a guarded decrement on counter c_j. From these locations, the $\star$-transition goes to location 1_j (where $j \in \{0, 1\}$).

The safety condition for synch and sink is trivial, and for all locations $f_{i,j}$ in B_j, it is of the form $e = 0$ if the instruction i is a guarded decrement on the counter c_j (for $0 < i < n$ and $j \in \{0, 1\}$). For all other locations, the safety condition is non-negative. The subset S includes all locations except the stay and sink locations s, sink.

To establish the correctness of the reduction, first assume that Minsky machine halts. Thus, there is a sequence of m configurations starting from $(1, (0, 0))$ and reaching halt:

$$(\mathsf{inst}_1, v_1)(\mathsf{inst}_2, v_2) \cdots (\mathsf{inst}_m, v_m) \text{ where } \mathsf{inst}_1 = 1 \text{ and } \mathsf{inst}_m = \mathsf{halt}.$$

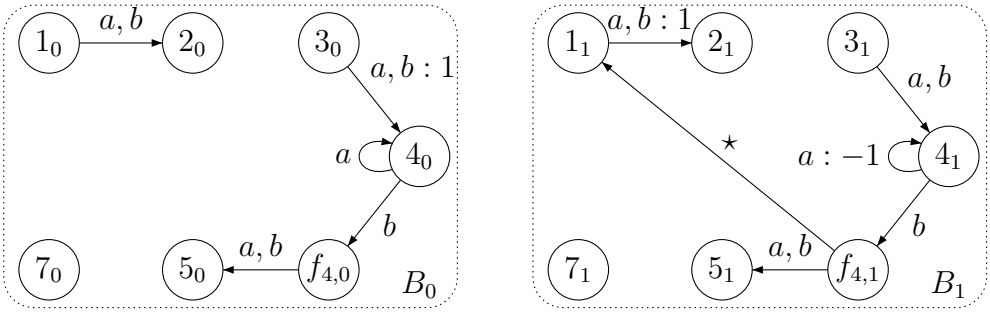

Figure 8.7: The details of B_0 and B_1 in Figure 8.6. To simplify the figure, the weights of transitions with weight 0 are omitted.

Consider the word $w = \star \cdot w_1 \cdots w_m \cdot \sharp$ where

- $w_i = a$ if inst_i is an increment or a guarded decrement on c_j when $e_j > 0$ in the valuation $v_i = (e_0, e_1)$,
- $w_i = b \cdot b$ if instruction i is a guarded decrement on c_j when $e_j = 0$ in $v_i = (e_0, e_1)$.

We see that w is a location-synchronizing word for $\mathcal{A}$.

Second, assume that $\mathcal{A}$ has a location-synchronizing word w. Since in the departure location d the only letter that does not violate the safety condition is $\star$, w must start with the letter $\star$. Reading $\star$ shrinks the set S into the locations $1_0, 1_1$ with energy level 0, s with all non-negative energy levels, and synch with all integer energy levels:

$$S_1 = \{(1_0, 0), (1_1, 0)\} \cup \{(s, e) \mid e \in \mathbb{N}\} \cup \{(\mathsf{synch}, e) \mid e \in \mathbb{Z}\}.$$

Since synch is absorbing, thus $\mathcal{A}$ is location-synchronized in synch; this location is only reachable by $\#$ and $\star$. On the other hand, due to the safety condition at the stay location s, inputting more $\star$'s is impossible. Thus, w must have some occurrence of $\#$ to location-synchronize the set S_1 into synch. Let $w' = a_1 \cdot a_2 \cdots a_n$ be the subword of w after the first $\star$ and up to the first $\#$, w' thus has neither $\#$ nor $\star$. Below, we show that there is a sequence of configurations such that the Minsky machine halts by operating this sequence. Consider $(1, (0, 0))$ to be the first configuration of the Minsky machine. For all $1 < k \leq n$ where n is the length of w':

1. let $(\ell, e_j) = \mathsf{post}((1_j, 0), a_1 \cdots a_k)$ in the automaton B_j for both $j = \{1, 2\}$,
2. if ℓ is of the form $f_{i,j}$ skip the next step,
3. define the next configuration (ℓ, v) of the Minsky machine such that $v = (e_1, e_2)$.

We know that after the subword w', there is an immediate $\#$ in the location-synchronizing word w. The $\#$-transitions in all locations of B_1 and B_2 except halt locations n_1 and n_2, bring $\mathcal{A}$ to the location sink (where there is no way to be synchronized with synch). Thus, after reading w' both automaton must be in their halt; otherwise w is not a location-synchronizing word. It implies that the constructed sequence of configurations for Minsky machine halts. Moreover, that is a valid configuration thanks to the

zero safety constraints at the locations simulating the guarded decrement instructions and non-negative safety constraints in other locations of B_0 and B_1.

□

In the absence of back-edges, we can get rid of the bounded safety constraints by explicitly encoding the energy values in the locations at the expense of an exponential blowup. We thus assign non-negative safety constraints to the encoded location and reduce to Lemma 8.6.

Lemma 8.8. *Given a WA $\mathcal{A}$ with no back-edges, the location-synchronizing problem from a subset S of locations under safety condition with bounded, lower-bounded or trivial constraints is decidable in* 3-EXPSPACE*, and it is* EXPSPACE-hard*.*

Proof. Let the WA $\mathcal{A} = \langle L, E\rangle$ and the safety condition Safe be such that the set of locations are partitioned as follows $L = L_{\mapsto} \cup L_{\mapsto} \cup L_{\leftrightarrow}$. Without loss of generality, assume that $L_{\leftrightarrow} \subseteq S$. We build another WA $\mathcal{A}'$ in which the locations of $L_{\mapsto}$ are augmented with the explicit value of the energy levels.

The construction of $\mathcal{A}' = \langle L', E'\rangle$ is as follows. The set of locations

$$L' = \{\langle \ell, e\rangle \mid \ell \in L_{\mapsto},\ e \in \mathsf{Safe}(\ell)\} \cup L_{\mapsto} \cup L_{\leftrightarrow}$$

contains all locations in $L_{\mapsto} \cup L_{\leftrightarrow}$ and a location $\langle \ell, e\rangle$ for all pairs of locations $\ell \in L_{\mapsto}$ and possible safe energy level for that location ℓ. There are following transitions in $\mathcal{A}'$:

- for all pairs $\langle \ell, e\rangle, \langle \ell', e+z\rangle \in L'$, there is a transition $(\langle \ell, e\rangle, a, 0, \langle \ell', e+z\rangle) \in E'$ from $\langle \ell, e\rangle$ to $\langle \ell', e+z\rangle$ with weight 0 in $\mathcal{A}'$ if $(\ell, a, z, \ell') \in E$,
- for all $\langle \ell, e\rangle \in L'$ and $\ell' \in L_{\mapsto} \cup L_{\leftrightarrow}$, there is a transition $(\langle \ell, e\rangle, a, e+z, \ell') \in E'$ from $\langle \ell, e\rangle$ to ℓ' with weight $e+z$ in $\mathcal{A}'$ if $(\ell, a, z, \ell') \in E$,
- for all pairs of locations $\ell, \ell' \in L_{\mapsto} \cup L_{\leftrightarrow}$, there is a transition $(\ell, a, z, \ell') \in E'$ in $\mathcal{A}'$ if $(\ell, a, z, \ell') \in E$.

We then define the safety condition Safe' over L' by letting $\mathsf{Safe}'((\ell, e)) = [0, +\infty)$, and $\mathsf{Safe}'(\ell) = \mathsf{Safe}(\ell)$ for all $\ell \in L_{\mapsto} \cup L_{\leftrightarrow}$. Finally, given a set of locations $S \subseteq L$ containing $L_{\leftrightarrow}$, we let

$$S' = \{\langle \ell, e\rangle \in L' \mid \ell \in S \cap L_{\mapsto}\} \cup [S \cap (L_{\mapsto} \cup L_{\leftrightarrow})].$$

Assuming that $L_{\mapsto} \cup L_{\leftrightarrow} \neq \emptyset$, we prove that a location-synchronizing word in $\mathcal{A}'$ from S' under safety condition Safe' is also a location-synchronizing word in $\mathcal{A}$ from S under safety condition Safe, and vice-versa.

First, assume that $\mathcal{A}'$ has a location-synchronizing word w. Let $\ell_f \in L'$ be the location where $\mathcal{A}'$ is synchronized in from S': $\ell_f = \mathsf{loc}(\mathsf{post}(S', w))$. Consider a state (ℓ, e) of $\mathcal{A}$ where $\ell \in S$. There are two cases: (i) the location $\ell \in L_{\mapsto} \cup L_{\leftrightarrow}$ has lower-bounded or trivial safety constraint, then (ℓ, e) is a valid state in $\mathcal{A}'$. Moreover, all states and transitions visited along the run of w over $\mathcal{A}'$ are valid states and transitions in $\mathcal{A}$ too. Thus, inputting w from (ℓ, e) in $\mathcal{A}$ brings the automaton in the same location ℓ_f. (ii) the

location $\ell \in L_{\mapsto}$ has bounded safety constraint. The run of $\mathcal{A}$ over the word w from the state (ℓ, e) is mimicked by run of $\mathcal{A}'$ but starting from $(\langle \ell, e \rangle, 0)$. As a consequence, this run ends in ℓ_f too.

The converse implication is similar. Assume that there is a location-synchronizing word w in $\mathcal{A}$ that merges all runs starting from S into the same location ℓ_f. We show that w is location-synchronizing in $\mathcal{A}'$ too. For all locations in $S \cap (L_{\mapsto} \cup L_{\leftrightarrow})$, as well as for the states of the form $((\ell, e), 0)$, it is easy to see that the run of $\mathcal{A}$ over w is mimicked with the run of $\mathcal{A}'$. Moreover, since there are only lower-bound or trivial constraints in Safe' condition for $\mathcal{A}'$, it is the case from states of the form $((\ell, e), f)$ with $f > 0$ too.

We then apply the methods used in Lemma 8.7 to the WA $\mathcal{A}'$. Since the construction of $\mathcal{A}'$ involves an exponential blowup in the number of locations, we end up with a 3-EXPSPACE algorithm. The EXPSPACE lower bound proved in Lemma 8.4 still applies here.

□

Now, we consider general safety conditions where the constraints can be bounded, lower-bounded, upper-bounded or trivial constraints. The set L of locations is partitioned into $L_{\mapsto}$, $L_{\mapsto}$, $L_{\leftarrow}$ and $L_{\leftrightarrow}$ where locations in $L_{\leftarrow}$ have upper-bounded safety constraints such as $e \leq n$ where $n \in \mathbb{Z}$. In this setting, transitions between locations in $L_{\leftarrow}$ and locations in both of $L_{\mapsto}$ and $L_{\leftrightarrow}$ are taken as *back-edge* too.

Lemma 8.9. *The location-synchronizing problem in WAs under a general safety condition is decidable in* 3-EXPSPACE*.*

Proof. Let $\mathcal{A} = \langle L, E \rangle$ be a WA over the alphabet A, and let Safe be a general safety condition. Considering the safety condition, we partition the set of locations into $L = L_{\mapsto} \cup L_{\mapsto} \cup L_{\leftarrow} \cup L_{\leftrightarrow}$. Using similar arguments as presented in the proof of Lemma 8.5, we remove all back-edges. Since all back-edges are removed, for all words w, starting from some location $\ell \in L_{\mapsto}$ the run over w either visit locations in $L_{\leftarrow}$ or location in $L_{\mapsto}$. We thus can turn the upper-bound constraints to the lower-bound constraints by negating the weights of transitions in all locations $\ell \in L_{\leftarrow}$ with such upper-bound constraints.

The following non-deterministic algorithm decides whether the WA $\mathcal{A}$ has a location-synchronizing word w under Safe. It first non-deterministically partitions $L_{\mapsto}$ into two sets $L_{\mapsto}^{\leftarrow}$ and $L_{\mapsto}^{\mapsto}$ such that the locations from which the run over w only visits location in $L_{\leftarrow}$ are guessed and are contained in $L_{\mapsto}^{\leftarrow}$, and the locations from which the run over w only visits location in $L_{\mapsto}$ are contained in $L_{\mapsto}^{\mapsto}$. The algorithm next builds another WA $\mathcal{A}' = \langle L', E' \rangle$ from $\mathcal{A}$ and Safe as follows. The set of locations is

$$L' = (L_{\mapsto} \cup L_{\leftarrow}) \times \{-1\} \cup (L_{\mapsto} \cup L_{\mapsto} \cup L_{\leftrightarrow}) \times \{1\}$$

and

– for all $x \in \{-1, 1\}$, the transition $(\langle \ell, x \rangle, a, x \times z, \langle \ell', x \rangle) \in E'$ is in $\mathcal{A}'$ if and only if $(\ell, a, z, \ell') \in E$;

– for all locations $\ell \in L_{\mapsto} \cup L_{\leftarrow}$ and $\ell' \in L_{\leftrightarrow}$, the transition $(\langle \ell, -1 \rangle, a, -z, \langle \ell', 1 \rangle) \in E'$ is in $\mathcal{A}'$ if and only if (ℓ, a, z, ℓ') is in E.

Now we change the safety conditions and define Safe' such that $e \in \mathsf{Safe}'(\langle \ell, y \rangle)$ if and only if $ye \in \mathsf{Safe}(\ell)$. Notice that Safe' then only has bounded, lower-bounded or trivial safety constraints.

This construction has the following property: a word w is a location-synchronizing word in $\mathcal{A}$ with safety constraint Safe if and only if it is also a location-synchronizing word in $\mathcal{A}'$ from the subset $S = (L_{\mapsto}^{\leftrightarrow} \cup L_{\leftarrow}) \times \{-1\} \cup (L_{\mapsto}^{\mapsto} \cup L_{\mapsto} \cup L_{\leftrightarrow}) \times \{1\}$ with safety constraint Safe'. Both implications are straightforwardly proven, and since $\mathcal{A}'$ has no back-edges the 3-EXPSPACE result follows.

□

From previous Lemmas and arguments, Theorem 8.4, stating the complexity bounds of location-synchronizing word in WAs under a general safety condition, follows. Regarding the PSPACE-hardness result, the reduction from partial transition function used in Lemma 8.2 can be adapted to use lower-bounded and interval safety constraint, which entails the result.

Theorem 8.4. *The location-synchronizing problem in WAs under a general safety condition is decidable in* 3-EXPSPACE*, and it is* PSPACE-hard*.*

9 Conclusion

Summary. The main contribution of this thesis is a novel approach to the qualitative analysis and the synthesis problem of the distribution-outcomes in probabilistic systems. We introduced synchronizing properties defined on the distribution-outcomes of Markov decision processes (MDPs) and probabilistic automata (PAs), that require that the winning strategy (or the input word) brings the system in some (group of) state(s) with a large probability, possibly in limit. In other words, the probability mass in the distribution, that indicates the likelihood of the system to be in different states at specific states, is accumulated in some target state. We defined always, eventually, weakly and strongly synchronizing properties that respectively are in analogy with safety, reachability, Büchi and coBüchi conditions in state-semantics. We considered three qualitative winning modes: sure, almost-sure and limit-sure modes, and we provided matching upper and lower complexity bounds of the existence problem of a winning strategy for synchronizing conditions, as well as the memory requirement for optimal winning strategies. In addition to the existence problem of a winning word-strategy in PAs that has been considered as the emptiness problems of synchronizing languages, we also discussed the universality problems. We carried over results for synchronizing problems from MDPs and PAs to two-player turn-based games and non-deterministic finite state automata. Along with the main results, we established new complexity results for alternating finite automata over a one-letter alphabet.

As a further contribution, we introduced the synchronization for timed automata (TAs) and weighted automata (WAs). We provided results for several variants and combinations of these: including deterministic and non-deterministic timed and weighted automata, synchronization to unique location with possibly different clock valuations or accumulated

weights, as well as synchronization with a safety condition forbidding the automaton to visit states outside a safety-set during synchronization (e.g. energy constraints).

Future works. Partially-observable Markov decision processes (POMDPs) are a generalization of MDPs where the controller, which makes the strategic choices, is not able to distinguish states, but groups of states through an observation [CDH10]. Thus, the states of systems are grouped (or sometimes partitioned) in different observations, and the controller choice only relies on partial information about the history of the system execution, namely, on the past sequence of observations. As one direction for further investigation, synchronizing properties in POMDPs can be considered. In particular, since PAs are a kind of POMDPs where the set of observations is a singleton, the undecidability results for the emptiness problem of some synchronizing languages in PAs, immediately give the undecidability for the existence problem of a winning observation-based strategy in POMDPs. On the other hand, MDPs are a kind of POMDPs where the set of observations is equal to the state space, and we have seen that all synchronizing problems are decidable in MDPs. Going from decidability to undecidability of synchronizing problems in these two extreme kinds of POMDPs, one can see which type of observations functions arise a decidable class of synchronizing properties in POMDPs; particularly, studying the sets of (practically meaningful) restrictions on the class of winning strategies with decidable synchronizing problems is interesting.

MDPs can be viewed as $1\frac{1}{2}$-player stochastic games, and thus a natural generalization of them is $2\frac{1}{2}$-player stochastic games where in addition to strategic choices of a controller (Player-1) and random responses of its environment, strategic choices for an adversary (Player-2) exist [CH12]. Along the results for the synchronizing properties in MDPs, we have established results for turn based two player games, that is kind of $2\frac{1}{2}$-player stochastic games where the environment always responds deterministically. The links between synchronizing properties in MDPs to turn based two player games are established by the fact that the exact value of probabilities in transitions of MDPs do not matter while synchronization. It would be very interesting to see whether this result holds in $2\frac{1}{2}$-player stochastic games, and to study the role of a de-synchronizing adversary and the memory requirement for the strategies of the adversary.

It is known that an irreducible finite Markov chain has a set of stationary distributions [KS83, Nor98, Ser13]. For an MDP and a strategy, a distribution X can be defined as a *stationary distribution* when in the distribution-outcome, there is an infinite subsequence of distributions converging to X. Given a set of target stationary distributions and an MDP, one can study *limit* conditions in analogy to Büchi and coBüchi conditions: does there exists a strategy for the MDP such that some or all stationary distributions are from the target set? The weakly and strongly synchronizing conditions are kind of limit conditions; for example, for the function max_T the set of target distributions in limit condition is the set of Dirac distributions on states of T. Another interesting point could be observing the behavior of the maximal subsequence of the distribution-outcome con-

verging to a stationary distribution X. In particular, taking intervals probabilities and assigning symbols to each interval distribution, the language obtained in the convergence of a maximal subsequence to a stationary distribution X, is irregular for Markov chains and if only one entry of X is of interest [AAGT12]. It is interesting to find the class of strategies such that the MDP shows a regular behavior when converging to a stationary distribution.

In this thesis, we have studied synchronizing and location-synchronizing problems for TAs and WAs. We left an open question in the thesis: decidability of the location-synchronizing problem in non-deterministic WAs. We showed that the technique used to solve location-synchronizing problem in deterministic WAs fails when applying for non-deterministic WAs, that is namely avoiding back-edges. Avoiding back-edges is not also possible when location-synchronizing from a subset of locations in deterministic WAs; though in that case we provided the undecidability result. It would be interesting to fill this gap and complete the extensive study done on synchronization of WAs. Recently, the study of synchronizing strategies under partial observation has been motivated in turn-based games [LLS14]. In this direction, a further topic to explore is defining synchronizing conditions in timed and weighted games, considering perfect-information and partial observation strategies for the players.

The concept of synchronization has been found useful in applications such as planning, control of discrete event systems, bio-computing, and robotics [BAPE$^+$03, Vol08, DMS11a]. The usage of synchronization also shows promises in the verification of *cryptographic protocols*. Cryptographic protocols are small programs that are specifically designed to ensure the security of private communications on public networks such as wireless ad-hoc networks. Contrary to classical networks using routers and access-points, ad-hoc networks consist of communicating members that broadcast a message one to another. Messages sent by a source member of the network is typically forwarded by other members until the message reaches its destination. *Onion protocols* have been specifically developed to ensure the *anonymity* of the members sending messages, meaning that a malicious member of the network should not be able to determine the identity of the sender from the forwarded messages [CD13]. By design, before forwarding a message, a member adds an encrypted layer of routing information to the message which increases the size of the message. The malicious member thus can determine the length of the path the message has gone through from the sender to himself. Using the concept of synchronization in the topology of small examples of networks, one can see that a sender, from which all paths to the malicious member have the same length, can be identified implying a breach on the anonymity property supposedly guaranteed by onion protocols. A further work could be to formally specify the synchronization requirements on the topology of ad-hoc networks and combine them with classic verification techniques on cryptographic protocols [CD13] to ensure the security of onion protocols.

Bibliography

[AAGT12] Manindra Agrawal, S. Akshay, Blaise Genest, and P. S. Thiagarajan. Approximate verification of the symbolic dynamics of Markov chains. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*, pages 55–64. IEEE, 2012.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[AGV10] Dimitry S. Ananichev, Vladimir V. Gusev, and Mikhail V. Volkov. Slowly synchronizing automata and digraphs. In *Mathematical Foundations of Computer Science 2010, 35th International Symposium, MFCS 2010, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6281 of *Lecture Notes in Computer Science*, pages 55–65. Springer, 2010.

[AH90] James Aspnes and Maurice Herlihy. Fast randomized consensus using shared memory. *Journal Algorithm*, 11(3):441–461, 1990.

[BAPE+03] Yaakov Benenson, Rivka Adar, Tamar Paz-Elizur, Zvi Livneh, and Ehud Shapiro. DNA molecule provides a computing machine with both data and fuel. *Proceedings of the National Academy of Sciences, USA*, 100:2191–2196, 2003.

[BBG08] Christel Baier, Nathalie Bertrand, and Marcus Größer. On decision problems for probabilistic Büchi automata. In *Foundations of Software Science and Computational Structures, 11th International Conference, FOSSACS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29 - April 6, 2008. Proceedings*, volume 4962 of *Lecture Notes in Computer Science*, pages 287–301. Springer, 2008.

[BBL08] Patricia Bouyer, Ed Brinksma, and Kim Guldstrand Larsen. Optimal infinite scheduling for multi-priced timed automata. *Formal Methods in System Design*, 32(1):3–23, 2008.

[BBMR08] Roberto Baldoni, François Bonnet, Alessia Milani, and Michel Raynal. On the solvability of anonymous partial grids exploration by mobile robots. In *Principles of Distributed Systems, 12th International Conference, OPODIS 2008, Luxor, Egypt, December 15-18, 2008. Proceedings*, volume 5401 of *Lecture Notes in Computer Science*, pages 428–445. Springer, 2008.

[BBS06] Christel Baier, Nathalie Bertrand, and Ph. Schnoebelen. On computing fixpoints in well-structured regular model checking, with applications to lossy

channel systems. In *Logic for Programming, Artificial Intelligence, and Reasoning, 13th International Conference, LPAR 2006, Phnom Penh, Cambodia, November 13-17, 2006, Proceedings*, volume 4246 of *Lecture Notes in Computer Science*, pages 347–361. Springer, 2006.

[BCH+07] Christel Baier, Lucia Cloth, Boudewijn R. Haverkort, Matthias Kuntz, and Markus Siegle. Model checking Markov chains with actions and state labels. *IEEE Transaction Software Engineering*, 33(4):209–224, 2007.

[BGB12] Christel Baier, Marcus Größer, and Nathalie Bertrand. Probabilistic ω-automata. *Journal ACM*, 59(1):1, 2012.

[BHK+12] Tomás Brázdil, Holger Hermanns, Jan Krcál, Jan Kretínský, and Vojtech Rehák. Verification of open interactive Markov chains. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012, December 15-17, 2012, Hyderabad, India*, volume 18 of *LIPIcs*, pages 474–485. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[BK98a] Christel Baier and Marta Z. Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11(3):125–155, 1998.

[BK98b] Christel Baier and Marta Z. Kwiatkowska. On the verification of qualitative properties of probabilistic processes under fairness constraints. *Inf. Process. Lett.*, 66(2):71–79, 1998.

[BK08] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

[BKHW05] Christel Baier, Joost-Pieter Katoen, Holger Hermanns, and Verena Wolf. Comparative branching-time semantics for Markov chains. *Information and Computation*, 200(2):149–214, 2005.

[BS96] Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory, Vol. 1: Efficient Algorithms*. MIT Press, 1996.

[CD11] Krishnendu Chatterjee and Laurent Doyen. Games and Markov decision processes with mean-payoff parity and energy parity objectives. In *Mathematical and Engineering Methods in Computer Science - 7th International Doctoral Workshop, MEMICS 2011, Lednice, Czech Republic, October 14-16, 2011, Revised Selected Papers*, volume 7119 of *Lecture Notes in Computer Science*, pages 37–46. Springer, 2011.

[CD13] Rémy Chrétien and Stéphanie Delaune. Formal analysis of privacy for routing protocols in mobile ad hoc networks. In *Principles of Security and Trust - Second International Conference, POST 2013, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013, Rome, Italy, March 16-24, 2013. Proceedings*, volume 7796 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2013.

[CDH10] Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. Qualitative analysis of partially-observable Markov decision processes. In *Mathematical Foundations of Computer Science 2010, 35th International Symposium, MFCS 2010, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6281 of *Lecture Notes in Computer Science*, pages 258–269. Springer, 2010.

[Čer64] Ján Černý. Poznámka k homogénnym experimentom s konečnými automatmi. *Matematicko-fyzikálny časopis*, 14(3):208–216, 1964.

[CH12] Krishnendu Chatterjee and Thomas A. Henzinger. A survey of stochastic ω-regular games. *Journal of Computer and System Sciences*, 78(2):394–413, 2012.

[CHJS13] Krishnendu Chatterjee, Monika Henzinger, Manas Joglekar, and Nisarg Shah. Symbolic algorithms for qualitative analysis of Markov decision processes with Büchi objectives. *Formal Methods in System Design*, 42(3):301–327, 2013.

[CHP08] Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Algorithms for Büchi games. *CoRR*, abs/0805.2620, 2008.

[CJS12] Krishnendu Chatterjee, Manas Joglekar, and Nisarg Shah. Average case analysis of the classical algorithm for Markov decision processes with Büchi objectives. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012, December 15-17, 2012, Hyderabad, India*, volume 18 of *LIPIcs*, pages 461–473. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[CKV+11] Rohit Chadha, Vijay Anand Korthikanti, Mahesh Viswanathan, Gul Agha, and YoungMin Kwon. Model checking MDPs with a unique compact invariant set of distributions. In *Eighth International Conference on Quantitative Evaluation of Systems, QEST 2011, Aachen, Germany, 5-8 September, 2011*, pages 121–130. IEEE Computer Society, 2011.

[CSV08] Rohit Chadha, A. Prasad Sistla, and Mahesh Viswanathan. On the expressiveness and complexity of randomization in finite state monitors. In *Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science, LICS 2008, 24-27 June 2008, Pittsburgh, PA, USA*, pages 18–29. IEEE Computer Society, 2008.

[CT12] Krishnendu Chatterjee and Mathieu Tracol. Decidable problems for probabilistic automata on infinite words. In *Proceedings of the 27th Annual IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25-28, 2012*, pages 185–194. IEEE, 2012.

[CY95] Costas Courcoubetis and Mihalis Yannakakis. The complexity of probabilistic verification. *Journal of the ACM*, 42(4):857–907, 1995.

[dA97] Luca de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Stanford University, 1997.

[dAH00] Luca de Alfaro and Thomas A. Henzinger. Concurrent omega-regular games. In *15th Annual IEEE Symposium on Logic in Computer Science, Santa Barbara, California, USA, June 26-29, 2000*, pages 141–154. IEEE Computer Society, 2000.

[dAHK07] Luca de Alfaro, Thomas A. Henzinger, and Orna Kupferman. Concurrent reachability games. *Theoretical Computer Science*, 386(3):188–217, 2007.

[Die12] Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

[DJL+14] Laurent Doyen, Line Juhl, Kim G. Larsen, Nicolas Markey, and Mahsa Shirmohammadi. Synchronizing words for weighted and timed automata. In *Proceedings of the 34th Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS'14*, Leibniz International Proceedings in Informatics. Leibniz-Zentrum für Informatik, 2014. To appear.

[DMS11a] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Infinite synchronizing words for probabilistic automata. In *Mathematical Foundations of Computer Science 2011 - 36th International Symposium, MFCS 2011, Warsaw, Poland, August 22-26, 2011. Proceedings*, volume 6907 of *Lecture Notes in Computer Science*, pages 278–289. Springer, 2011.

[DMS11b] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Synchronizing objectives for Markov decision processes. In *Proceedings International Workshop on Interactions, Games and Protocols, iWIGP 2011, Saarbrücken, Germany, 27th March 2011.*, volume 50 of *EPTCS*, pages 61–75, 2011.

[DMS12] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Infinite synchronizing words for probabilistic automata (erratum). *CoRR*, abs/1206.0995, 2012.

[DMS14a] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Limit synchronization in Markov decision processes. In *Foundations of Software Science and Computation Structures - 17th International Conference, FOSSACS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8412 of *Lecture Notes in Computer Science*, pages 58–72. Springer, 2014.

[DMS14b] Laurent Doyen, Thierry Massart, and Mahsa Shirmohammadi. Robust synchronization in Markov decision processes. In *CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings*, volume 8704 of *Lecture Notes in Computer Science*, pages 234–248. Springer, 2014.

[EKVY08] Kousha Etessami, Marta Z. Kwiatkowska, Moshe Y. Vardi, and Mihalis Yannakakis. Multi-objective model checking of Markov decision processes. *Logical Methods in Computer Science*, 4(4), 2008.

[EL10] Stewart Ethier and Jiyeon Lee. A Markovian slot machine and parrondo's paradox. *Annals of Applied Probability*, 20:1098–1125, 2010.

[Epp90] David Eppstein. Reset sequences for monotonic automata. *SIAM J. Comput.*, 19(3):500–510, 1990.

[FJLS11] Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jirí Srba. Energy games in multiweighted automata. In *Theoretical Aspects of Computing - ICTAC 2011 - 8th International Colloquium, Johannesburg, South Africa, August 31 - September 2, 2011. Proceedings*, volume 6916 of *Lecture Notes in Computer Science*, pages 95–115. Springer, 2011.

[FP06] Wan Fokkink and Jun Pang. Variations on Itai-Rodeh leader election for anonymous rings and their analysis in PRISM. *Journal of Universal Computer Science*, 12(8):981–1006, 2006.

[FV97] Jerzy Filar and Koos Vrieze. *Competitive Markov Decision Processes*. Springer, 1997.

[FV13] Fedor M. Fominykh and Mikhail V. Volkov. P(l)aying for synchronization. *International Journal of Foundations of Computer Science*, 24(6):765–780, 2013.

[GJ79] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[GO10] Hugo Gimbert and Youssouf Oualhadj. Probabilistic automata on finite words: Decidable and undecidable problems. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, LNCS 6199, pages 527–538. Springer, 2010.

[GR88] A. Gibbons and W. Rytter. *Efficient parallel algorithms*. Cambridge University Press, 1988.

[HKK14] Holger Hermanns, Jan Krcál, and Jan Kretínský. Probabilistic bisimulation: Naturally on distributions. In *CONCUR 2014 - Concurrency Theory - 25th International Conference, Rome, Italy, September 2-5, 2014. Proceedings*, volume 8704 of *Lecture Notes in Computer Science*, pages 249–265. Springer, 2014.

[HMW09] Thomas A. Henzinger, Maria Mateescu, and Verena Wolf. Sliding window abstraction for infinite Markov chains. In *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings*, volume 5643 of *Lecture Notes in Computer Science*, pages 337–352. Springer, 2009.

[Hol95] Markus Holzer. On emptiness and counting for alternating finite automata. In *Developments in Language Theory*, pages 88–97, 1995.

[IRS76] Harry B. Hunt III, Daniel J. Rosenkrantz, and Thomas G. Szymanski. On the equivalence, containment, and covering problems for the regular and context-free languages. *Journal of Computer and System Sciences*, 12(2):222–268, 1976.

[IS95] Balázs Imreh and Magnus Steinby. Some remarks on directable automata. *Acta Cybern.*, 12(1):23–35, 1995.

[IS99] Balazs Imreh and Magnus Steinby. Directable nondeterministic automata. *Acta Cybernetica*, 14(1):105–115, 1999.

[JS07] Petr Jancar and Zdenek Sawa. A note on emptiness for alternating finite automata with a one-letter alphabet. *Information Processing Letters*, 104(5):164–167, 2007.

[Kfo70] D.J. Kfoury. Synchronizing sequences for probabilistic automata. *Studies in Applied Mathematics*, 29:101–103, 1970.

[KNP05] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Probabilistic model checking in practice: case studies with PRISM. *SIGMETRICS Performance Evaluation Review*, 32(4):16–21, 2005.

[KNP08] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Using probabilistic model checking in systems biology. *SIGMETRICS Performance Evaluation Review*, 35(4):14–21, 2008.

[Koz77] Dexter Kozen. Lower bounds for natural proofs systems. In *Proceedings of 18th Symposium on the Foundations of Computer Science*, pages 254–266, 1977.

[KS83] John G. Kemeny and James Laurie Snell. *Finite Markov chains*. Springer-Verlag, New York, 1983.

[KVAK10] Vijay Anand Korthikanti, Mahesh Viswanathan, Gul Agha, and YoungMin Kwon. Reasoning about MDPs as transformers of probability distributions. In *QEST 2010, Seventh International Conference on the Quantitative Evaluation of Systems, Williamsburg, Virginia, USA, 15-18 September 2010*, pages 199–208. IEEE Computer Society, 2010.

[Lip76] Richard J. Lipton. The reachability problem requires exponential space. 62, New Haven, Connecticut: Yale University, Department of Computer Science, Research, 1976.

[LLS14] Kim Guldstrand Larsen, Simon Laursen, and Jirí Srba. Synchronizing strategies under partial observability. In *CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings*, volume 8704 of *Lecture Notes in Computer Science*, pages 188–202. Springer, 2014.

[Mar90] Donald A. Martin. An extension of borel determinacy. *Annals of Pure and Applied Logic*, 49(3):279–293, 1990.

[Mar98] Donald A. Martin. The determinacy of blackwell games. *Journal of Symbolic Logic*, 63(4):1565–1581, 1998.

[Mar10] Pavel V. Martyugin. Complexity of problems concerning carefully synchronizing words for PFA and directing words for NFA. In *Computer Science - Theory and Applications, 5th International Computer Science Symposium in*

Russia, CSR 2010, Kazan, Russia, June 16-20, 2010. Proceedings, volume 6072 of *Lecture Notes in Computer Science*, pages 288–302. Springer, 2010.

[Mar12] Pavel V. Martyugin. Synchronization of automata with one undefined or ambiguous transition. In *Implementation and Application of Automata - 17th International Conference, CIAA 2012, Porto, Portugal, July 17-20, 2012. Proceedings*, volume 7381 of *Lecture Notes in Computer Science*, pages 278–288. Springer, 2012.

[Min67] Marvin Lee Minsky. *Computation: Finite and Infinite Machines*. Prentice Hall, Inc., 1967.

[Nor98] James R. Norris. *Markov chains*. Cambridge series in statistical and probabilistic mathematics. Cambridge University Press, 1998.

[OU10] Jörg Olschewski and Michael Ummels. The complexity of finding reset words in finite automata. In *Mathematical Foundations of Computer Science 2010, 35th International Symposium, MFCS 2010, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6281 of *Lecture Notes in Computer Science*, pages 568–579. Springer, 2010.

[Pin78] Jean-Eric Pin. Sur les mots synthronisants dans un automate fini. *Elektronische Informationsverarbeitung und Kybernetik*, 14(6):297–303, 1978.

[Rac78] Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6:223–231, 1978.

[Ros93] Kenneth H. Rosen. *Elementary number theory and its applications (3. ed.)*. Addison-Wesley, 1993.

[San04] Sven Sandberg. Homing and synchronizing sequences. In *Model-Based Testing of Reactive Systems, Advanced Lectures [The volume is the outcome of a research seminar that was held in Schloss Dagstuhl in January 2004]*, volume 3472 of *Lecture Notes in Computer Science*, pages 5–33. Springer, 2004.

[Ser13] Bruno Sericola. *Markov Chains: Theory and Applications*. Wiley-ISTE, 2013.

[Sip97] Michael Sipser. *Introduction to the theory of computation*. PWS Publishing Company, 1997.

[SVW87] A. Prasad Sistla, Moshe Y. Vardi, and Pierre Wolper. The complementation problem for Büchi automata with appplications to temporal logic. *Theoretical Computer Science*, 49:217–237, 1987.

[TBG09] Mathieu Tracol, Christel Baier, and Marcus Größer. Recurrence and transience for probabilistic automata. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2009, December 15-17, 2009, IIT Kanpur, India*, volume 4 of *LIPIcs*, pages 395–406. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2009.

[Tra11] A. N. Trahtman. Modifying the upper bound on the length of minimal synchronizing word. In *Fundamentals of Computation Theory - 18th International*

Symposium, FCT 2011, Oslo, Norway, August 22-25, 2011. Proceedings, volume 6914 of *Lecture Notes in Computer Science*, pages 173–180. Springer, 2011.

[Tra14] A. N. Trahtman. The length of a minimal synchronizing word and the černy conjecture. *CoRR*, abs/1405.2435, 2014.

[Var85] Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 327–338. IEEE Computer Society, 1985.

[Vol08] Mikhail V. Volkov. Synchronizing automata and the Cerny conjecture. In *Language and Automata Theory and Applications, Second International Conference, LATA 2008, Tarragona, Spain, March 13-19, 2008. Revised Papers*, volume 5196 of *Lecture Notes in Computer Science*, pages 11–27. Springer, 2008.