Detection and quantification of events in stochastic systems

Hugo Bazille

▶ **To cite this version:**

Hugo Bazille. Detection and quantification of events in stochastic systems. Machine Learning [cs.LG]. Université de Rennes, 2019. English. NNT : 2019REN1S107 . tel-02954814

HAL Id: tel-02954814
https://theses.hal.science/tel-02954814

Submitted on 1 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITE BRETAGNE LOIRE MATHSTIC

UNIVERSITÉ DE RENNES 1

THÈSE DE DOCTORAT DE

L'UNIVERSITE DE RENNES 1
COMUE **U**NIVERSITE **B**RETAGNE **L**OIRE

Ecole Doctorale N°601
Mathématique et Sciences et Technologies de l'Information et de la Communication
Spécialité : Informatique

Par

« Hugo BAZILLE »

« Detection and Quantification of Events in Stochastic Systems »

Thèse présentée et soutenue à Rennes , le 2 décembre 2019

Unité de recherche : INRIA, Equipe SUMO

Composition du jury :

Président :	
Rapporteurs :	Christoforos HADJICOSTIS, Professor, University of Cyprus.
	Stefan KIEFER, Associate professor, Oxford university.
Examinateurs :	Béatrice BERARD, Professeur, Université Paris 6.
	Benoit CAILLAUD, Directeur de recherche, Université Rennes 1.
	Nathanael FIJALKOW, Chargé de recherche, CNRS, LABRI.
Dir. de thèse :	Eric FABRE, Directeur de recherche, INRIA Rennes.
Co-dir. de thèse :	Blaise GENEST, Directeur de Recherche, CNRS, IRISA.

Résumé

Notre dépendance à l'égard des processus automatisés prend de plus en plus d'importance dans chaque aspect de notre vie: finances, transports, robotique, communication, sécurité, systèmes médicaux... De plus, cette croissance s'accélère: chaque objet a maintenant une version "connectée". En classe, les devoirs sont donnés et faits sur des plateformes en ligne. Des IA spécifiques sont formées pour effectuer des diagnostics médicaux à partir d'imageries médicales. L'argent liquide tend à disparaître des transactions financières. Les exemples d'irruption de la technologie dans tous les domaiens sont (presque) infinis.

Que se passe-t-il quand quelque chose ne fonctionne pas comme prévu? Dans le pire des cas, le coût est comptabilisé en vies humaines et en millions/milliards d'euros. Parmi les évènements les plus tristement célèbres, citons le crash de la sonde spatiale Mars Climate Orbiter, où un sous-traitant avait conçu un système de navigation utilisant le système impérial au lieu du système métrique, le lancement échoué d'Ariane V en 1995, ou encore en 1983 un satellite d'alerte précoce soviétique capta les reflets du soleil sur les nuages et les interpréta à tort comme un lancement de missiles aux Etats-Unis, provoquant presque le début de la troisième guerre mondiale. En outre, il existe une grande variété de problèmes. [WHK18] montre comment le changement d'un pixel d'une image modifie le résultat d'un logiciel de reconnaissance: un feu rouge était classé comme vert, un autre feu était maintenant un four... Il est facile de voir l'importance de ces problèmes pour la conduite automatisée.

Bien que les exemples les plus célèbres concernent des systèmes critiques, il existe des problèmes sous-optimaux apparemment bénins qui, sans être un danger, peuvent coûter quelques euros à chaque fois, multipliés par des milliers ou des millions d'usagers. Un exemple est le système Orion développé par UPS pour leurs chauffeurs de camion [Hol+17]. Il optimise leurs déplacements en limitant les virages à gauche: traverser la route entraîne une plus longue période d'attente, entraînant une perte de temps et d'essence. Un autre domaine est celui des télécommunications. Pour des raisons physiques, de nombreux protocoles (codes, répétitions, etc.) essaient d'assurer qu'un message n'est pas perdu. Une question est “quelles garanties peuvent être données sur un réseau selon différents scénarios”. Ces garanties peuvent porter sur la couverture de réseau, les ressources utilisées... De nombreux travaux récents visent à vérifier les réseaux, en particulier les réseaux de

capteurs (tels que [Law+03; Tob+07]...) pour des propriétés telles que la couverture et la résistance à la défaillance d'un composant.

Lorsqu'un problème a été identifié, comment peut-on le corriger? Dans certains cas, le changement à effectuer est facilement identifiable, comme dans le cas de la sonde Mars Climate Orbiter ("utilisez simplement le système métrique!"). Dans d'autres cas, la question est beaucoup plus difficile. Reprenant l'exemple du logiciel de reconnaissance, il est complexe de comprendre pourquoi deux images identiques à de plus de $99,99\%$ sont classées de manière aussi différente. En général, nous voulons trouver des moyens de certifier que le comportement observé d'un système est conforme au comportement prévu.

C'est là que la vérification formelle entre en jeu. La vérification formelle est définie comme la vérification de l'exactitude d'un conception/produit à l'aide de techniques mathématiques. Cela peut par exemple être fait en prouvant mathématiquement la correction. Une réponse négative peut également être apportée grâce à un contre-exemple. Une question de vérification peut également demander une réponse plus détaillée que "oui/non". Récemment, plusieurs travaux ont élargi la vérification formelle aux questions quantitatives [HK97], en prenant en compte des quantités telles que le temps [Che+09] ou les probabilités [Bri+13]. Ces quantités permettent d'obtenir des résultats plus précis, mais les techniques associées ne sont pas encore matures et peuvent être améliorées. C'est une direction que regarde cette thèse et dans laquelle nous allons pousser nos recherches.

Concernant la vérification formelle, différentes techniques apparaissent. Dans ce qui suit, nous discutons des principales techniques, ainsi que des objets mathématiques sur lesquels la vérification peut être effectuée.

Nous pouvons distinguer deux cadres principaux dans la vérification: soit nous travaillons directement sur un système, soit sur une abstraction de ce système, appelée modèle. Une première question est "qu'est-ce qu'un bon modèle?". C'est une question délicate: pour modéliser le système, il faut décider ce qui est important et ce qui ne l'est pas, et formaliser les différentes intéractions, réactions et tout ce qui peut se produire lors de l'exécution du système. Ainsi, certaines informations seront perdues lors de la création d'un modèle. Cependant, cette perte est nécessaire pour obtenir un modèle de taille raisonnable. Dans certains cas, différents modèles avec différentes précisions peuvent être conçus. Ces modèles peuvent être comparés les uns aux autres. Par exemple, si A est un raffinement de B, on peut souhaiter que ce qui se passe dans B se produise également dans A, éventuellement avec des détails supplémentaires. La nécessité d'obtenir un modèle approprié est bien exprimée dans [BK08] (chapitre 1), "toute vérification utilisant des techniques basées sur un modèle n'est pas meilleure que le modèle du système". Cela

met en évidence le fait que la finalité des techniques basées sur des modèles n'est pas de certifier la "perfection" du système mais plutôt de gagner en confiance. En effet, certains problèmes peuvent être masqués par la modélisation. Il est donc nécessaire d'obtenir des garanties formelles sur ces modèles pour renforcer la confiance. C'est une direction que nous allons explorer dans cette thèse.

Sur la modélisation: Des modèles peuvent être générés à partir d'un système existant afin de vérifier son exactitude. Pour cela, une représentation précise et non ambiguë du système et des propriétés à vérifier doit être créée. A titre d'exemple, un sous-ensemble des propriétés d'un protocole de communication,le protocole ISDN (Integrated Services Digital Network), a été formalisé. Cette formalisation a montré qu'une grande partie (55%) des spécifications était incohérente [Hol92]. Ainsi, la modélisation formelle permet de rechercher des bugs sur les systèmes existants ou sur leurs spécifications. Un autre exemple est le satellite Deep Space 1. La vérification basée sur un modèle a montré plusieurs défauts de conception [Hav+00] sous la forme de problèmes de concurrence. Un possible blocage qui n'avait pas été détecté lors des centaines d'heures de test fut créé au cours des 24 premières heures de fonctionnement par une suite d'instructions qui était pourtant peu probable.

D'autre part, on peut d'abord travailler sur un modèle jusqu'à obtenir quelque chose de "satisfaisant", puis développer un système correspondant à ce modèle. Un exemple en est l'évaluation de la performance d'un système de train urbain [Ade+17]. Au lieu d'effectuer des mesures sur une exécution du système (regarder les trains et mesurer certains indicateurs tels que le retard, la ponctualité, etc.), un modèle de simulation efficace pouvant représenter un réseau est conçu et les mesures sont effectuées sur ce modèle de simulation. Alors que le premier doit être fait en temps réel, une simulation des heures de mouvements de train peut être faite en quelques secondes. Cela met en évidence un autre point des techniques basées sur un modèle, à savoir des performances plus élevées pour de nombreux problèmes.

Que peut-on garantir? Toutes les propriétés ne peuvent pas être vérifiées sur tous les systèmes. Par exemple, considérons la question "le programme se termine-t-il?". Sur les programmes C, cela mène à un problème indécidable. Cependant, limiter cette question à des classes spécifiques de modèles peut assurer la décidabilité. Un défi consiste alors à trouver un cadre suffisamment expressif pour coder des propriétés intéressantes tout en conduisant à une décidabilité en un temps raisonnable. Des sous-ensembles de "questions"

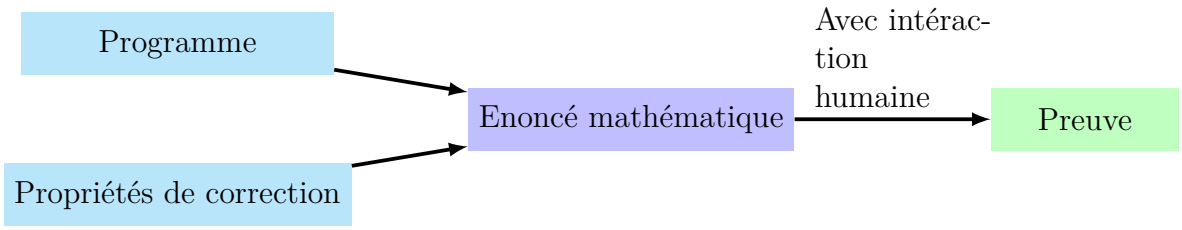

Figure 1: La vérification déductive.

possibles ont été étudiés et des techniques pour ces sous-ensembles spécifiques ont été développées, comme par exemple les logiques LTL et CTL.

Vérification déductive: Un premier moyen de s'assurer qu'un système se comporte comme prévu est de coder l'exactitude de ce système en tant qu'énoncé mathématique, puis de prouver cet énoncé. Parmi d'autres possibilités, ces preuves peuvent être réalisées à l'aide d'un assistant de preuves (tel que Coq, Isabelle, Why3 ...) ou de solveur basé sur la *satisfiability modulo theories* (tels que CVC4, OpenSMT...). Cette approche a l'avantage d'être très puissante: on doit "juste" exprimer l'exactitude en tant qu'énoncé mathématique. Cependant, elle présente un inconvénient: elle a besoin d'un expert, non seulement pour définir l'énoncé mathématique, mais également pour trouver une stratégie permettant de prouver cet énoncé, par exemple sous la forme d'une suite de théorèmes.

Tests: Etant donné un système et une spécification, le test consiste à exécuter le système avec différentes valeurs d'entrée et à vérifier si le comportement souhaité est observé (*i.e.*, la spécification est validée). Les tests visent à montrer que les comportements attendus et réels d'un système diffèrent, ou à prendre confiance qu'ils ne le font pas. Ils peuvent être effectués sur un système (eg "essayons cette voiture sur un circuit et effectuer des mesures") ou sur un modèle de ce système [UPL12; GS18] (*e.g.*, "voici un modèle de cette voiture et un logiciel de simulation de flux d'air, étudions l'aérodynamique sur différents réglages"). Bien que l'idée des tests soit ancienne, des travaux récents visaient à les formaliser et à les rendre plus efficaces [Pel13; Ber07]. Un des avantages des tests est qu'ils sont simples et peuvent être effectués en un temps souvent raisonnable. Cependant, l'efficacité des tests dépend de leur pertinence : cette méthode manque de complétude. Une question clé est donc la couverture des tests, c'est-à-dire déterminer si les cas vérifiés sont suffisants pour obtenir des garanties robustes.

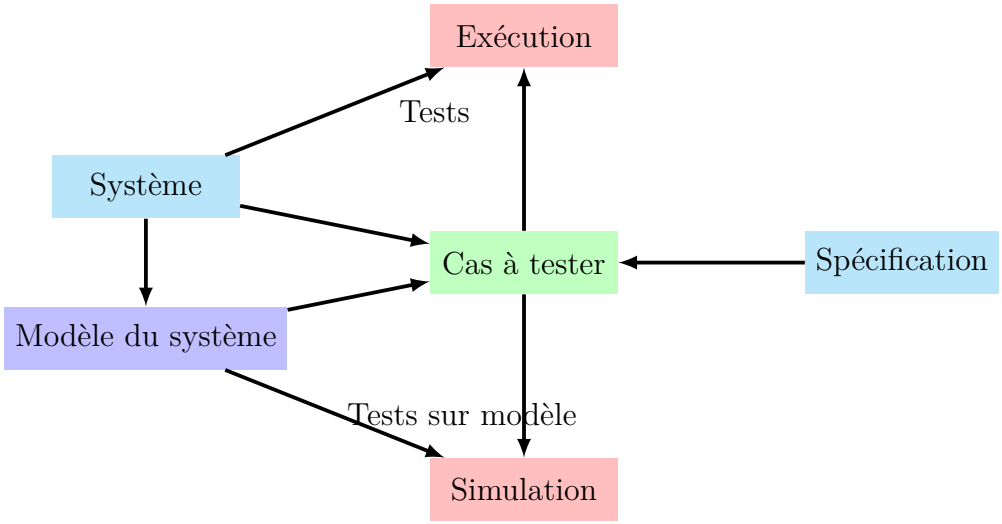

Figure 2: Les tests.

Vérification de modèle: Comme son nom l'indique, la vérification de modèle consiste à vérifier un modèle d'un système, c'est-à-dire vérifier si un modèle satisfaît une spécification donnée grâce à une exploration des états et des transitions du modèle du système. Pour pouvoir effectuer cette vérification, le modèle et la spécification doivent tous deux être exprimés en un langage mathématique. Ensuite, le vérificateur de modèles vérifie si le modèle satisfait la formule. Si ce n'est pas le cas, il fournit un contre-exemple, c'est-à-dire une preuve que la spécification est violée. Ce contre-exemple peut ensuite être analysé pour modifier la spécification si elle révèle un défaut de conception ou pour affiner le modèle si ce contre-exemple est un "faux positif", grâce à des techniques comme le raffinement de l'abstraction guidé par les contre-exemples (CEGAR) [Cla+00].La vérification des modèles a été introduite au début des années 80 [CE80; CE81; QS82] et a été régulièrement développée et étudiée depuis lors, avec l'introduction de langages plus expressifs permettant de décrire des spécifications plus complexes (telles que des logiques plus expressives [AHK02]), et de les associer à des évaluations précises des complexités (*e.g.*, in [SC85]). Pour les applications pratiques, les tests et la vérification des modèles sont en concurrence, chacun présentant des avantages et des inconvénients [BL17]. Un premier inconvénient de la vérification des modèles est qu'en raison de l'exploration exhaustive, les techniques de vérification des modèles peuvent ne pas bien passer à l'échelle. La seconde, bien sûr, est la nécessité d'un modèle. En revanche, un avantage considérable de la vérification basée sur un modèle est que la vérification peut être effectuée de manière systématique et autonome et qu'elle est exhaustive.

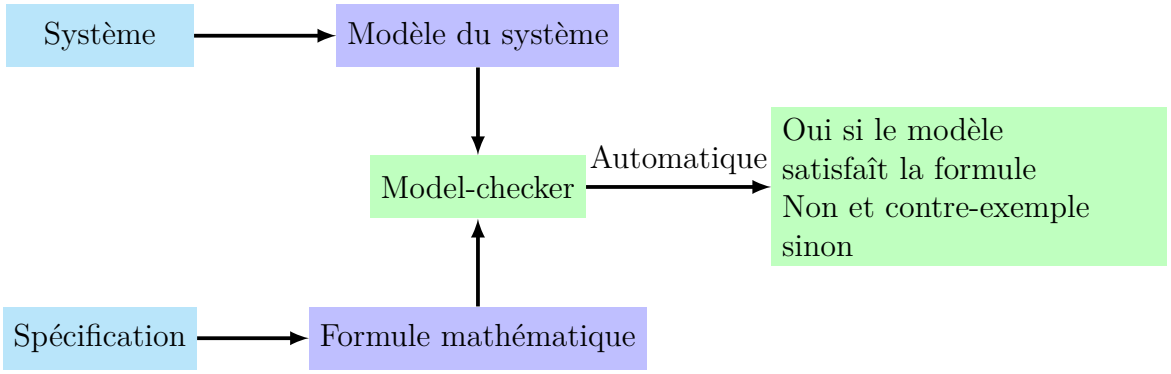

Figure 3: La vérification de modèles.

Dans cette thèse, nous allons nous concentrer sur les techniques de vérification des modèles. Comme indiqué précédemment, nous avons besoin d'un modèle pour pouvoir appliquer ces techniques. Ainsi, nous détaillons certaines caractéristiques importantes des modèles que nous allons considérer.

Systèmes stochastiques: Comme mentionné précédemment, les modèles ne représentent qu'une vision partielle d'un système, dans le but de ne conserver que ce qui est important. En fonction de l'application, la question de ce qui est important peut avoir différentes réponses, ce qui entraîne à nouveau des problèmes et des techniques différents selon le paradigme choisi pour la description du modèle et des propriétés. Dans cette thèse, nous allons considérer des modèles avec des probabilités. Pourquoi des probabilités? Ils permettent de représenter un système avec des comportements aléatoires (non contrôlés par l'utilisateur et/ou l'environnement) ou des systèmes avec des informations incomplètes présentant des motifs statistiques. Par exemple, si le non-déterminisme permet de modéliser différentes possibilités, par exemple un adversaire ayant plusieurs choix, les probabilités permettent de représenter de nombreuses personnes qui feront toutes leur choix et lorsque ce nombre est grand, une distribution de probabilité de ces choix peut être déduite. Ces deux contextes différents (ici, un adversaire et une population) apportent deux formalismes différents (non déterminisme et systèmes stochastiques) De plus, on peut utiliser des modèles stochastiques pour effectuer une évaluation quantitative de certaines propriétés: l'utilisation de quantités ouvre la possibilité de répondre à plus de questions que de simples questions logiques (*i.e.*, celles auxquelles on répond "vrai"/"faux") . Ainsi, les probabilités sont utilisées pour représenter divers systèmes [BS13], tels que la robotique probabiliste pour des essaims de drones [Bra+12], les télécommunications [Alu+97],

pour le traitement du signal et des images [CNB98; CK97], les systèmes dynamiques en général [Smy94] ... Ils apparaissent dans la conception du traitement et de la reconnaissance des langues [Moh97; Rab89]. Ils interviennent également dans la modélisation des processus climatiques et biologiques [Edd04] pour la météo [ATT09], des séquences de protéines et d'acides aminés [Dur+98; Gou+01; Kro+01]...

Afin de représenter les probabilités, de nombreux modèles formels sont utilisés tels que les chaînes de Markov (à temps discret ou continu et espace d'états discret ou continu), les processus de décision Markoviens, les chaînes de Markov étiquetées, les réseaux de Petri stochastiques (...), dont certains seront détaillés et utilisés ultérieurement dans ce travail.

Information partielle: On peut également souhaiter représenter le fait que l'état exact d'un système peut ne pas être connu à chaque moment: en règle générale, les systèmes ne sont pas entièrement observables. En effet, nous n'avons pas un accès complet à ce qui se passe à l'intérieur pour de nombreuses raisons: sécurité, coûts financiers, manque de fiabilité des capteurs, sa taille, certains événements dépendent de l'environnement... Par conséquent, dans le monde réel, bien que les utilisateurs puissent connaître parfaitement un modèle d'un système, ils n'ont qu'une connaissance partielle de son état actuel lors d'une exécution. Nous devrons donc en tenir compte lors de la modélisation et du raisonnement sur nos modèles. Cela se reflétera par le fait que, pour une (séquence) d'informations disponible pour l'utilisateur, plusieurs états internes du système peuvent être simultanément possibles par rapport à ces informations. Notre tâche consistera souvent à récupérer (avec une probabilité élevée) des informations cachées sur l'exécution du système. Pour cela, le principal formalisme que nous allons utiliser est les chaînes de Markov étiquetées, où les probabilités modélisent l'incertitude dans le système et un alphabet modélise les informations qu'un observateur peut obtenir.

Apprentissage: S'il est intéressant de raisonner sur un système stochastique donné, un autre problème est de savoir comment l'obtenir. Une solution consiste à apprendre (un modèle du) le système à partir d'échantillons de ses exécutions. En général, l'apprentissage est la capacité d'acquérir de nouvelles connaissances ou de modifier des connaissances, des compétences, des valeurs (...) existantes en analysant des données. Ce processus peut être supervisé (avec un enseignant) ou par un processus d'essais et erreurs... Une difficulté est que le processus d'apprentissage n'est pas encore totalement compris, même (et surtout) pour les humains. L'apprentissage automatique a été introduit à la fin des années 50 dans le but de faire en sorte que les systèmes "apprennent" à répondre efficacement et avec

précision à des problèmes pour lequels aucun autre algorithme efficace n'existait. Pas assez efficace pendant plusieurs décennies, l'apprentissage automatique a commencé à gagner du terrain dans les années 1990 avec le passage à un paradigme basé sur des méthodes empruntées aux statistiques et à la théorie des probabilités [Lan11].

Selon les sources d'information disponibles, il existe différentes sous-catégories de techniques d'apprentissage: l'apprentissage supervisé [RN16] "réplique" l'idée de disposer d'un enseignant qui donne la solution correcte aux cas que l'algorithme apprend, l'apprentissage par renforcement [KLM96] est basé sur l'ajustement des comportements afin d'obtenir une récompense maximale, et l'apprentissage non supervisé [HSP99], au contraire, ne dépend pas des informations fournies par une autorité supérieure: le processus d'apprentissage doit effectuer l'évaluation lui-même dans un processus d'essais et d'erreurs.

L'apprentissage peut être effectué sur différents modèles pour de nombreuses applications: vision par ordinateur [Lee+09], reconnaissance automatique de la parole [Wai+89], diagnostic médical [Kon01] ...

Dans ce qui suit, nous nous intéresserons à l'apprentissage de modèles probabilistes. Fondamentalement, cela peut être séparé en deux parties. Tout d'abord, à l'instar d'un modèle non probabiliste, il faut obtenir la structure du modèle, c'est-à-dire les états possibles et les transitions d'un état à un autre. Ensuite, on voudrait estimer avec précision les valeurs de ces probabilités. Cette partie est au moins aussi difficile que la première: les probabilités dépendent de nombreux facteurs généralement insolubles. En outre, une petite erreur dans l'évaluation peut entraîner une différence énorme quand on considère des propriétés globales, comme nous le verrons plus tard. Ceci est mis en évidence dans la figure 4: dans cet exemple, à chaque étape nous restons dans s_0 avec probabilité $1-2\varepsilon$, allons dans le bon état ☺ avec probabilité ε et dans le mauvais état ☹ avec la probabilité ε. La probabilité d'atteindre au bout d'un moment ☺ est de $0,5$. Supposons que ε soit petit et que nous ayons estimé les probabilités du modèle et obtenu $1-2\varepsilon$, $3\varepsilon/2$ et $\varepsilon/2$. Les probabilités sont très proches puisque ε est très faible, mais maintenant, la probabilité d'atteindre éventuellement ☺ est de $0,75$, ce qui est très différent de $0,5$. Dans ce document, nous allons nous concentrer sur l'apprentissage des chaînes de Markov, l'un des modèles les plus simples qui contiennent des probabilités.

Plan: En résumé, dans ce document, nous allons étudier comment **récupérer des informations** et **quantifier** sur des **systèmes stochastiques** avec des **informations partielles**. Pour cela, nous donnerons d'abord quelques définitions, notations et résultats généraux précédents, puis nous étudierons trois problèmes: la diagnosticabilité, la

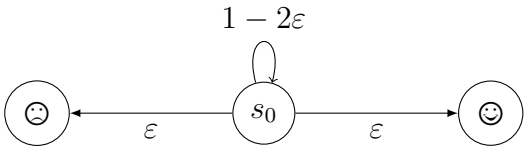

Figure 4: Exemple jouet où les probabilités de cahque transition sont proches mais où le comportement global est très différent.

classification et l'apprentissage. Cette thèse est organisée comme suit:

- Dans le chapitre 2, nous introduisons des notations et des définitions utiles que nous utiliserons tout au long de ce document. Nous présentons également quelques résultats généraux qui seront utiles pour différentes parties de la thèse.

- Au chapitre 3, nous nous concentrons sur le diagnostic. Similairement à la médecine, la possibilité de diagnostic est la capacité de récupérer une information (telle qu'un événement correspondant à une erreur) d'une exécution du système. Nous irons plus loin que la question binaire «pouvons-nous déduire cette information ou non», en la transformant en "avec quelle probabilité pouvons-nous déduire cette information"? Nous examinerons également la question "à quelle vitesse pouvons-nous déduire cette information?" en poussant les questions quantitatives.

- Dans le chapitre 4, nous traitons de la classifiabilité qui est, étant donné deux systèmes et une observation, la capacité de décider lequel de ces systèmes a produit l'observation. La classifiabilité peut être considérée dans un certain sens comme une généralisation de certains problèmes tels que la diagnostiabilité, c'est-à-dire décider si l'exécution a été produite par la partie du système où l'événement d'erreur a eu lieu ou non. Notez que cela n'est pas techniquement vrai, car le diagnostic est intrinsèquement asymétrique et la classification est symétrique, mais le concept et les preuves utilisées peuvent être similaires.

- Dans le chapitre 5, nous passons à un problème orthogonal, qui consiste à apprendre un système stochastique. Ce problème soulève certaines des questions précédemment discutées sur ce qu'est un bon modèle. Étant donné un système stochastique, on souhaiterait connaître les probabilités de ses transitions pour obtenir un modèle fidèle de son fonctionnement.

Table of Contents

Chapter 1

Introduction

Our reliance on automatized processes is growing in every aspect on our life: financial, transportation, robotic, communication, safety, medical systems... Further, this growth is accelerating: every object has now a "connected" version. In classes, homework is given and done on online platforms. Specific AI are trained to perform medical diagnosis from medical imagery. Cash is disappearing from financial transactions. The examples of the invasion of technology are (almost) infinite.

What happens when something does not work as intended? In the worst cases, the cost is counted in human lives and millions/billions of euros. Some of the most infamous occurrences are the Mars Climate Orbiter Crash, where a subcontractor designed a navigation system using imperial units instead of the metric system, the Ariane V failed launch, where a 64 bits number was stored in a 16 bits space, or in 1983 a Soviet early warning satellite picked up sunlight reflections off cloud-tops and mistakenly interpreted them as missile launches in the United States, almost causing the start of world war III. Besides, there is a huge variety of problems that may not be that extreme. [WHK18] presents how the change of one pixel in an image changes the output in a recognition software: a red traffic light was classified as green, another traffic light was now an oven... It is easy to see the importance of these problems for automatized driving. While the most famous examples concern critical systems, there are some seemingly benign suboptimal issues that, while not a danger, may cost a few euros at each time, multiplied by thousand or millions of usages. An example is the Orion system developed by UPS for their truck drivers [Hol+17]. It optimizes paths for their travels by limiting left turns: crossing the road leads to more idle time going through ongoing traffic, leading to a loss of time and gas. Another field is telecommunications. Due to physical reasons, numerous protocols (codes, repetitions...) try to ensure that a message is not lost. A question is "what guarantees can be given on a network under different scenarios". These guarantees can be on the coverage, the resources used... Many recent works aim at verifying networks, especially sensor networks (such as [Law+03; Tob+07]...) for properties such as coverage and robustness to failure of one component.

When a problem has been identified, how does one correct it? In some cases, the change to make is easily identifiable, such as for the Mars Climate Orbiter Crash ("just use the metric system!"). In other cases, it is a much more difficult question. Taking again the example of recognition software, it is a complex task to understand why two images that are more than 99.99% identical are so differently classified. In general, we want to find ways to certify that the observed behavior of a system is conform to its intended behavior.

That is where formal verification comes in. Formal verification is defined as checking the correctness of a design/product using mathematical techniques. It can be done by proving mathematically that the correctness holds. It can also answer by the negative by providing a counter-example. A verification question can also ask for a more detailed answer than "yes/no". Recently, several works expanded formal verification to quantitative questions [HK97], considering quantities such as time [Che+09] or probabilities [Bri+13]. These quantities allow one to get finer results, however techniques around them are still young and can be improved. This is a direction this thesis considers.

When performing formal verification, different techniques can be used. In the following, we discuss about the main techniques, as well as the mathematical objects verification can be performed on.

We can distinguish two main frameworks in verification: either we work directly on a real system, or on an abstraction of this system, called a model. A first question is "what is a good model?". This is a tricky question: in order to model the system, one has to decide what is important and what is not, and formalize the different interactions, reactions and basically what can occur in an execution of the system. Thus, some information will be lost at the creation of a model. However, this loss is necessary in order to obtain a model of a tractable size. In some cases, different models with different accuracies can be designed. These models can be compared with one another. For example, if A is a refinement of B, one may want that what happens in B happens also in A, possibly with additional details. The need to obtain a suitable model is well expressed in [BK08] (chapter 1), "any verification using model-based techniques is only as good as the model of the system". This highlights the fact that the finality of model-based techniques is not to certify the "perfection" of the system but only to gain confidence. Indeed, some problems can be masked by the modeling. Thus, obtaining formal guarantees on these models is needed to boost the confidence. This is a direction we will explore along this thesis.

On modeling: Models can be generated from an existing system in order to verify the correctness of the system. For that, a precise and unambiguous representation of the system and of the properties to check have to be created. As an example, a subset of the properties of a communication protocol, the Integrated Services Digital Network (ISDN) protocol, has been formalized. This formalization showed that a huge part (55%) of the requirement were inconsistent [Hol92]. Thus, formal modelization allows one to find bugs on existing systems or on their specifications. Another example is the Deep Space-1 spacecraft. Model-based verification showed some design flaws [Hav+00] in the form of concurrency errors. A deadlock that did not occurred in the hundreds of hours of system-testings was created by an unlikely scheduling condition during the 24 first hours of operation.

On the other hand, one can at first work on a model until something "satisfying" has been obtained and then develop a system with respect to this model. An example is the performance evaluation of an urban train system [Ade+17]. Instead of performing a measurement on an execution of the system (*i.e.*, watching trains and measuring some indicators such as delay, punctuality...), an efficient simulation model that can represent a network is designed and measurements are performed on this simulation model. While the former has to be done in real time, a simulation of hours of train movements can be done in seconds. This highlights another point of model based techniques, that is higher performances for several problems.

What can one ensure? Not all properties can be verified on every system. For example, let us consider the question "does the program terminate". On C programs, it leads to an undecidable problem. However, restricting this question to specific classes of models can ensure decidability. A challenge is then to find a framework expressive enough to encode interesting properties while leading to decidability in reasonable time. Subsets of possible "questions" have been studied and techniques for these specific subsets have been developed *e.g.*, LTL and CTL logics.

Deductive verification: A first way to ensure that a system does behaves as expected is by encoding the correctness of this system as a mathematical statement and then prove this statement. Among other possibilities, these proofs can be done with the help of a proof assistant (such as Coq, Isabelle, Why3...) or satisfiability modulo theories solvers (such as CVC4, OpenSMT...). This approach has the advantage to be very powerful: one "just" needs to express the correctness as a mathematical statement. However it has a

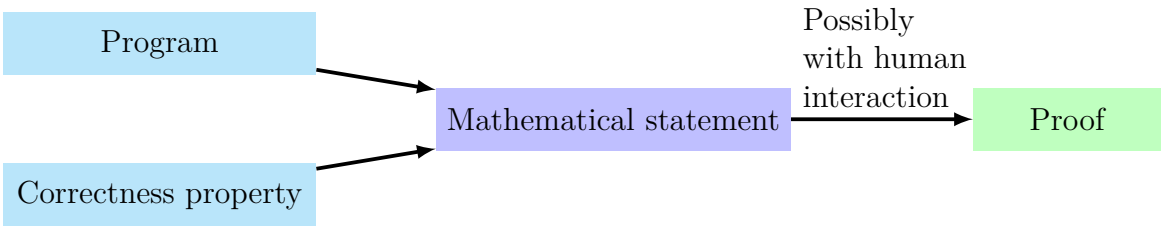

Figure 1.1: Principle of deductive verification.

related disadvantage: it needs an expert, not only to define the mathematical statement, but also to find a strategy to prove this statement, such as in the form of a sequence of theorems.

Testing: Given a system and a specification, testing consists in executing the system with different input values and observing whether the intended behavior appears (*i.e.*, the specification holds). It aims at showing that the intended and actual behaviors of a system differ, or at gaining confidence that they do not. Testing can be made on a system (*e.g.*, "let us try this car on a circuit and perform some measurements") or on some model of this system [UPL12; GS18] (*e.g.*, "here is a modelization of this car and a flow simulation software, let us study the aerodynamics on some different settings"). Though the idea of testing is quite ancient, recent work aimed at formalizing testing and making it more efficient [Pel13; Ber07]. An advantage of testing is that it is straightforward and can be done in reasonable time. However, efficiency of testing depends on the pertinence of the tests: there is a lack of completeness in this method. A key question is test coverage *i.e.*, determining if the cases verified are broad enough to obtain robust guarantees.

Model-checking: As the name suggests, model-checking consists in checking a model of a system, that is verifying if some given specification holds on a model of the system with an exploration of the states and transitions of the model of the system. In order to be able to perform this verification, both the model and the specification have to be expressed in mathematical languages. Then, the model-checker verifies if the model satisfies the formula. If it does not, it provides a counterexample, *i.e.*, a proof that the specification is violated. This counterexample can then be analyzed to change the specification if it reveals a design flaw or to refine the model if this counterexample is a "false positive", thanks to techniques such as the Counterexample-Guided Abstraction Refinement (CEGAR) [Cla+00]. Model-checking has been introduced in the early 80s [CE80; CE81;

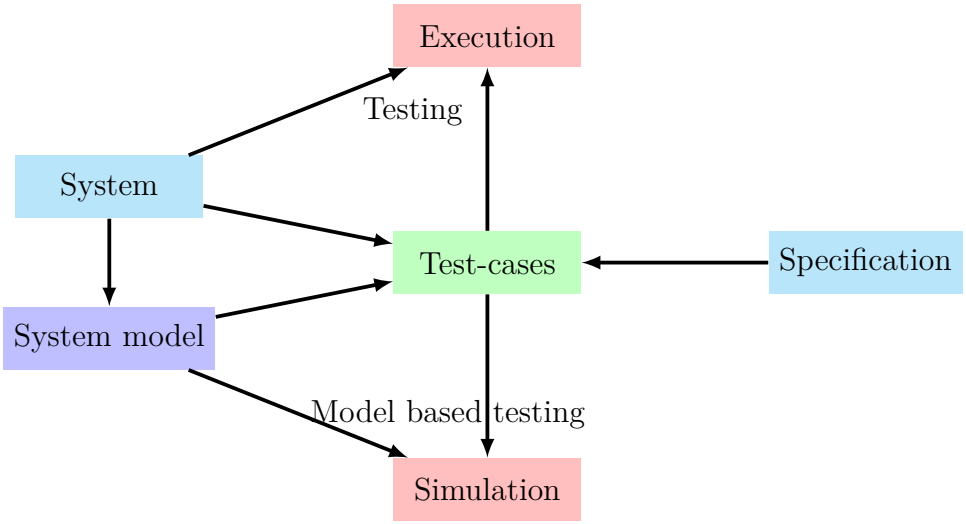

Figure 1.2: Principle of testing.

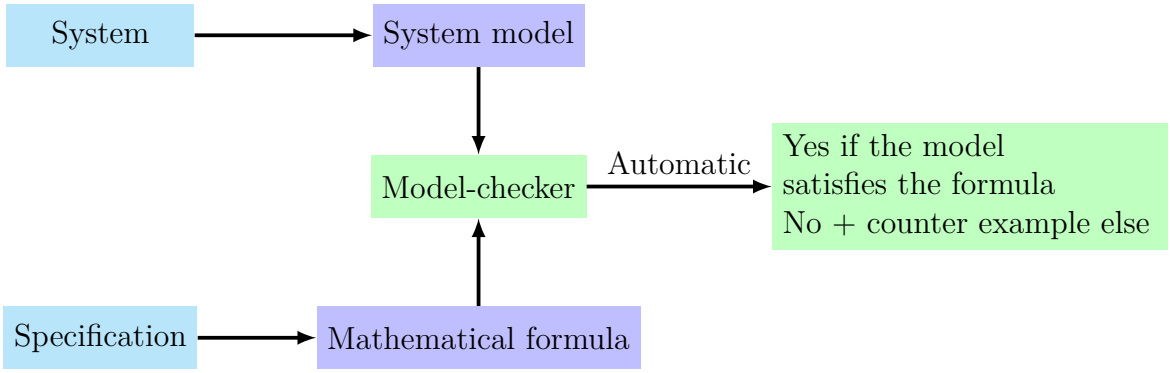

Figure 1.3: Principle of model-checking.

QS82] and has been continuously developed and studied since then, with the introduction of more expressive languages that allow one to describe more complex specifications (such as extensions of logics [AHK02]), and to associate them to precise evaluations of the complexities (*e.g.*, in [SC85]). For practical applications, testing and model-checking are in competition, each with advantages and drawbacks [BL17]. A first drawback of model-checking is that due to the exhaustive exploration, model-checking techniques may not scale up well with the size of the model. A second, of course, is the need for a model. On the other hand, a tremendous advantage of model-based verification is that the verification can be performed in a systematic and autonomous way, and that it is exhaustive.

In this thesis, we will focus on model-checking techniques. As said before, we need some

model to be able to apply these techniques. Thus, we detail some important characteristic of the models we will consider.

Stochastic systems: As mentioned before, models represent only a partial vision on a system, with the aim at keeping only what is important. Depending on the application, the question of what is important may have different answers, again leading to different problems and techniques depending on the chosen paradigm of the description of model and properties. In this thesis we will consider models with probabilities. Why probabilities? They allow one to represent a system with either random behaviors (non-controlled by the user and/or the environment) or systems with incomplete information that exhibit statistical patterns. For example, while non-determinism allow one to model different possibilities, such as an adversary that may have several possible choices, probabilities enable to represent many people that all will do their own choice and when this number of people is high, a probability distribution of these choices can be inferred. These two different settings (here, an adversary and a population) bring two different formalisms (non-determinism and stochastic systems) Further, one can use stochastic models to perform a quantitative evaluation of some properties: the usage of quantities opens up the possibility to answer more questions than only logical ones (*i.e.*, those answered by "true"/"false"). Thus, probabilities are used for representing various real systems [BS13], such as probabilistic robotics for swarms of drones [Bra+12], telecommunications [Alu+97], for signal and image processing [CNB98; CK97], dynamic systems in a large sense [Smy94]... They appear in the design of language processing and recognition [Moh97; Rab89]. They also intervene in the modelization of climatic and biological processes [Edd04] for weather [ATT09], sequences of proteins and amino-acids [Dur+98; Gou+01; Kro+01]... In order to represent probabilities, numerous formal models are used such as (discrete/continuous time/states) Markov Chains, Markov Decision Processes, Labeled Markov Chains, Stochastic Petri nets (...), some of which we will detail and use later in this work.

Incomplete information: One may also wish to represent the fact that the exact state of a system may not be known at each time: usually, real life systems are not fully observable. Indeed, we have no full access to what happens inside for many reasons: security, financial costs, unreliability of sensors, its size, some events depend on the environment... Hence, in real world, users may know perfectly a model of a system but when running it, they only have a partial knowledge about its current state. Thus, we will have to take this

into account when modeling and reasoning on our models. This will be reflected by the fact that for one (sequence of) information available to the user, several internal states of the system may be simultaneously possible with respect to this information. Our task will often be to recover (with high probability) some hidden information about the execution of the system. For that, the main formalism we will use is Labeled Markov Chains, where probabilities model the uncertainty in the system and an alphabet models the information an observer can obtain.

Learning: While it is interesting to reason on a given stochastic system, another problem is how to obtain it. One way is by learning (a model of) the system from samples of its executions. In general, learning is the ability to acquire new or modify existing knowledge, skills, values (...) by analyzing data. This process can be supervised (*i.e.*, with a teacher), or by trial and error/evaluation... A difficulty is that the process of learning is still not totally understood, even (and especially) for humans. Machine learning was introduced at the end of the 1950s and aimed at making systems "learn" how to answer efficiently and accurately to some problems that no other efficient algorithms could solve. Not effective enough for several decades, machine learning started to gain traction in the 1990s with the shift to a paradigm based on methods borrowed from statistics and probability theory [Lan11]. Different sub-categories of learning techniques exist according to the available sources of information: supervised learning [RN16] "replicates" the idea of having a teacher that gives the correct solution to the cases the algorithm learns, reinforcement learning [KLM96] is based on learning how to adjust behaviors in order to obtain a maximal reward, and unsupervised learning [HSP99], on the contrary, does not depend on information given by a higher authority: the learning process grades itself in a process of trial and error. Learning can be performed on various models for many applications: computer vision [Lee+09], speech recognition [Wai+89], medical diagnosis [Kon01]...

In the following, we will be interested in learning probabilistic models. Basically, this can be separated in two parts. First, similarly to a non-probabilistic model, one has to obtain the structure of the model, that is the possible states and the transitions from one state to another. Then, one would like to estimate accurately the values of these probabilities. This part is at least as difficult as the first one: probabilities depend on many factors that are usually intractable. Further, a small error in the evaluation may lead to a huge difference when focusing on some properties, as we will pinpoint later. This is highlighted in figure 1.4: in this example, at each step we stay in s_0 with probability $1-2\varepsilon$, go in the good state ☺ with probability ε and go in the bad state ☹ with probability

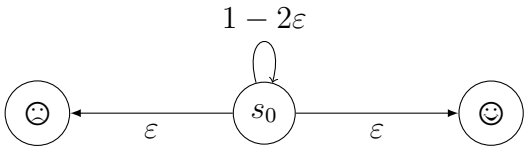

Figure 1.4: Toy example where local transitions are close enough but general properties are very different.

ε. Then, the probability to eventually reach ☺ is 0.5. Let us suppose that ε is small and that we estimated the model and obtained the probabilities $1-2\varepsilon$, $3\varepsilon/2$ and $\varepsilon/2$. Probabilities are very close since ε is very small, but now the probability to eventually reach ☺ is 0.75, which is very different. In this document, we will focus on the learning of Markov Chains, one of the simplest model that contain probabilities.

Outline: To summarize, in this document we will study how to **retrieve information** and **quantify** it on **stochastic systems** with **partial information**. For that, we will first give some definitions, notations and previous general results and then study three problems: diagnosability, classifiability and learning. This thesis is organized as follows:

- In Chapter 2, we introduce useful notations and definitions that we will use all along this document. We also present some general results that will be useful for different parts of the thesis.

- In Chapter 3, we focus on diagnosability. Similarly to medicine, diagnosability is the ability to retrieve an information (such as an event corresponding to an error) from an execution of the system. We will go further than the binary question "are we able to deduce this information or not", transforming it in "how often can we deduce this information". We will also investigate the question "how fast can we deduce this information", pushing the quantitative questions.

  After a state of the art addressing qualitative diagnosability and some extensions to probabilistic models, we introduce some definitions of quantitative diagnosability along with algorithms to compute these quantities. Then, we present how to use these algorithms in order to approximately closely reconstruct the distribution of fault delay and to derive bounds on these delays with some confidence intervals. We also show the optimality of these bounds. This chapter is based on the contributions presented in [BFG17; BFG18b; BFG18a].

- In Chapter 4, we address classifiability which is, given two systems and one observation, the ability to decide which system produced the observation. Classifiability can be seen in some sense as a generalization of some problems such as diagnosability, *i.e.*, decide if the execution has been produced by the part of the system where the error event occurred or not. Notice that it is not technically true, as diagnosis is intrinsically asymmetric and classification is symmetric, but the concept and the proofs used can be similar.

  First we present a state of the art, including some recent works that tackle notions that are equivalent to classifiability. We then present a new proof of the complexity of classifiability. We then investigate the notion of classifiability in a security context: what happens if we give an attacker some ability to act on the systems? This chapter is based on the contributions presented in [Aks+19].

- In Chapter 5, we shift to an orthogonal problem, that is learning a stochastic system. This problem raises some of the questions discussed before about what is a good model. Given a stochastic systems, one would like to learn its transition probabilities to obtain a faithful model of its operation.

  After presenting different techniques to estimate the transition probabilities from the execution of a system and the framework of Probably Approximately Correct (PAC) learning, we focus on obtaining (PAC) guarantees on the learned models with respect to global properties. We first consider simple formulas, such as time to failures and then we show than on some set of logical properties computing a confidence level for all properties in this set is possible, whereas for some other set, this is not possible. For the paradigms where this is possible, we provide an algorithm and an evaluation of the results.

  This chapter is based on the contributions we will present in [Baz+].

The details of my publications this thesis is based on are provided before the references.

Chapter 2

Preliminaries

In this chapter, we present the different notations and definitions that will be useful along this document. We also present some fundamental results. In section 2.1 we present numerous models of interest that we will use in this thesis: we go from general models (automata, Markov Chains) to specific ones (labeled Markov Chain, Probabilistic Finite Automata...). In section 2.2 we recall specific vocabulary and properties of Markov Chains. In section 2.3 we remind vocabulary and notions of convergences on random variables. In section 2.4 we state some problems that we will be interested in and give the notations of logic that will help us define these problems. In section 2.5 we give some general algorithmic results on which we will base some of our results.

Given an alphabet Σ, we denote by Σ^* (resp. Σ^ω) the set of finite (resp. infinite) words over Σ. $\mathbb{N}$ is the set of natural numbers, $\mathbb{Q}$ the rational numbers and $\mathbb{R}$ the real numbers. For a set X, we write 2^X the powerset of X.

2.1 Classes of models

In this thesis, systems are represented by formal models. A tremendous number of different models exist. Each one has its specificity and allows one to express different kinds of properties. First, we are going to present different models of interest that we will use alongside this document.

2.1.1 Transition systems and automata

Dynamical systems are characterized by their current state and their trajectories. Thus, a natural representation of a dynamical system can be made by using a set of states, each representing the current state of the system, and an alphabet representing either the different possible actions or observations of the execution. In the following, we will distinguish systems with stopping time and those without. As a consequence, in the first case, the corresponding executions will have a finite length and in the second one we will

consider (set of) infinite executions. Both settings will be relevant later. We also remind some usual vocabulary on these systems and set notations.

The first basic model is the labeled transition system (LTS), defined as follow:

Definition 2.1 (Labeled Transition System)**.**
A labeled transition system A is a quadruple (S, Σ, I, T) such that:

- *S is a finite set of states,*
- *Σ is a finite alphabet,*
- *$I \subseteq S$ is the set of initial states,*
- *$T \subseteq S \times \Sigma \times S$ is the set of transitions.*

For a LTS A, we denote $(s \xrightarrow{a} s')$ a transition such that $(s, a, s') \in T$. Given $t = (s \xrightarrow{a} s')$ a transition, its observation denoted $o(t)$ is the letter a, its initial state $s^-(t) = s$ and its final state $s^+(t) = s'$. In the literature, the observation is often called "label". In this thesis, we consider generative systems where the letters will generally be signals given by an execution, hence the name "observation". A finite path of A is a sequence of transitions $\pi = t_1 \dots t_n$ such that for all $1 \leq i < n$, $s^+(t_i) = s^-(t_{i+1})$ and its observation is $o(\pi) = o(t_1) \dots o(t_n)$. For two states s and s', the set of paths from s to s' is denoted $\mathcal{P}(s, s')$. For S_f a set of states, we denote by $\mathcal{P}_{S_f}(s, s')$ the set of paths from s to s' that do not have a state of S_f as an intermediary state. The set of finite paths of A is denoted $\mathcal{P}(A)$. A finite run ρ is a path such that its first state is an initial state and the set of runs of A is denoted $\mathcal{R}(A)$. Similarly, we define infinite paths and runs as an infinite set of transitions $(t_i)_{i \in \mathbb{N}}$ such that for all i, $s^+(t_i) = s^+(t_{i+1})$ and their set, $\mathcal{P}^\omega(A)$ and $\mathcal{R}^\omega(A)$.

The language $\mathcal{L}(A)$ is defined as the set of observations w such that there exists $\rho \in \mathcal{R}(A)$ with $w = o(\rho)$. Again, we define the infinite language $\mathcal{L}^\omega(A)$. Let $\pi = t_1 \dots t_n$ be a finite path. We denote its length $|\pi| = n$. The definition and notation of the observation, the initial state and the final of a path is naturally extended from those of a transition: we have $s^-(t_1 \dots t_n) = s^-(t_1)$, $s^+(t_1 \dots t_n) = s^+(t_n)$ and $o(\pi(s \xrightarrow{a} s')) = o(\pi)a$. Given $\pi' = t'_1 \dots t'_m$ such that $s^+(t_n) = s^-(t'_1)$, the concatenation of π and π' denoted $\pi\pi'$ is the path $t_1 \dots t_n t'_1 \dots t'_m$. A path π is a prefix (resp. suffix) of π'' if there exists π' such that $\pi'' = \pi\pi'$ (resp. $\pi'' = \pi'\pi$). For a finite run ρ, the cylinder of ρ, denoted $Cyl(\rho)$ is the set of infinite runs ρ' such that ρ is a prefix of ρ'. By extension, for a observation w, the cylinder of w is the set of infinite size observations having w as a prefix.

A state is s is reachable from s' if there exists a path with initial state s' and final state s. s is reachable if there exists a path starting from an initial state and ending in s. A strongly connected component (SCC) is a set $Q \subseteq S$ such that for all $s, s' \in Q$, s is

reachable from s'. Moreover, it is a bottom strongly connected component (BSCC) if for all $s \in Q$, s' is reachable from s implies $s' \in Q$.

We notice that runs in a LTS correspond to process that have no defined end. The "finite run" counterpart of the LTS is the finite state automaton, defined as follow:

Definition 2.2 (Finite State Automaton)**.**

An automaton A is a quintuple (S, Σ, I, T, F) such that:

- *S is a finite set of states,*
- *Σ is a finite alphabet,*
- *$I \subseteq S$ is the set of initial states,*
- *$T \subseteq S \times \Sigma \times S$ is the set of transitions,*
- *F is the set of final states.*

Previous definitions on finite paths still apply. The language $L(A) \subseteq \Sigma^*$ of an automaton A is the set of observations w such that there exists a *final path* starting in a state of I, ending in a state of F and with observation w, *i.e.*, $L(A) = \{o(\rho),\ s^-(\rho) \in I,\ s^+(\rho) \in F\}$.

Example 2.1. *LTS A represented in figure 2.1 recognizes the set of infinite observations $L^\omega(A) = \{a^\omega\} \cup \{a^n b^\omega, n \in \mathbb{N}^*\}$. Its BSCC are $\{s_1\}$ and $\{s_2\}$.*

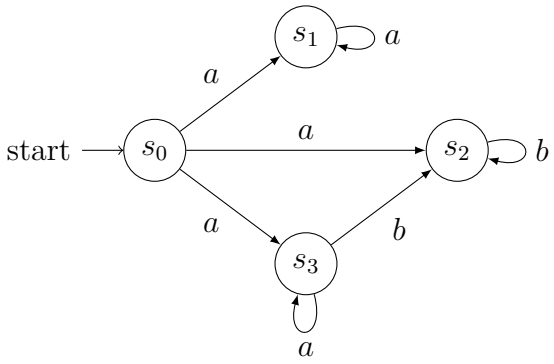

Figure 2.1: Example of an LTS A.

2.1.2 Partial observation

In a perfect world, an observer would know the exact state of a system, of all its parameters... Unfortunately, this is not usually the case. There are many reasons for this: a

system may be designed to be opaque in some way for security reasons. Moreover, gathering information is costly: many sensors, captors... would be needed. Furthermore, some data can be unreliable, such that those caused by environment perturbation. The amount of data needed to have a complete grasp of the system could also simply be too huge. Hence, in real world, observers may know perfectly a system but when running it, they only have a partial view of its current state. In order to formalize this, we have to state what information can be gathered. In this thesis, we will consider a natural paradigm: the current state of the transition system is unknown, and two actions that give the same information to an observer may lead to two different outcomes. Thus, two transitions from a same state labeled by the same letter can lead to different states: we consider non-deterministic systems.

Another way to model partial observation would be to have silent transitions: the alphabet is partitioned in two, the observable and the unobservable ones: $\Sigma = \Sigma_o \cup \Sigma_u$. In this setting, the observation is the projection over the observable alphabet: given a path π and t labeled by a, its observation $\tilde{o}(\pi t)$ is $\tilde{o}(\pi)a$ if $a \in \Sigma_o$, and $\tilde{o}(\pi)$ else. Having silent transitions does not extend the expressivity of the model. These silent transitions can be removed by a process of ε transition removal [Moh02a], creating a non-deterministic system where states are not observable, *i.e.*, the models we consider in this thesis. This process is conducted as follows: for a sequence of transitions $u_1 \dots u_n t$ from s to s' such that for all i, u_i is silent and t is observable and labeled by a, we create the transition (s, a, s') if it does not exist already. Notice that in this process we consider an equivalence at the moment an observation is raised, and not during a sequence of silent transitions. This is not a problem: in general, a judgment on the system will be raised at the moment an observation is gathered.

Hence, we will consider in the following transition systems where the alphabet is fully observable and the states are unobservable.

Example 2.2. *In figure 2.2, the transitions labeled by u and f are unobservable. However, the sequence fa makes the system go in state s_1 with the observation a, thus we add a transition labeled by a from s_0 to s_1. Similarly, after the sequence ub, the system is in state s_3. Thus, we add a transition from s_0 to s_3 labeled by b. This transition "forgets" that the sequence went through state s_2, highlighting that the equivalence is at the exact moment an observation is raised.*

Sometimes, silent transitions are considered slightly differently: the label is hidden at the observation, however the user knows that a transition occurred. In this case, it is easy

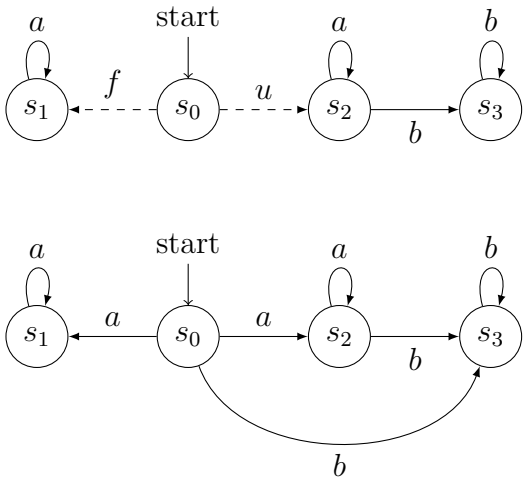

Figure 2.2: An LTS A with silent transitions (above) and its equivalent A' after ε-removal (below).

to build an equivalent system by replacing each silent transitions by one with a special letter whose meaning will be "a transition occurred".

2.1.3 Quantitative systems

Weighted automata: a general model with quantities

In order to add information to executions of a system, we want to enrich the model. A general way to do this is to add quantities to the transitions. These quantities may be very general: weights, costs, rewards... These quantities can be modeled not only by (real) numbers, but also by elements of more complex structures. In this section, we present weighted automata, where the weights of the transitions are elements of semirings. We will see what properties some semirings have and how they help us to calculate interesting information. Overviews of this formalism can be found in [Sch61; KS85].

Definition 2.3 (Semiring)**.**

Let $\mathbb{K}$ *be a set,* $\oplus$ *such that* $(\mathbb{K}, \oplus)$ *is a commutative monoid with identity element* $\overline{0}$ *and* $\otimes$ *such that* $(\mathbb{K}, \otimes)$ *is a commutative monoid with identity element* $\overline{1}$*.*

$(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ *is a semiring iff*

- $\otimes$ *distributes over* $\oplus$
- $\otimes$ *annihilates over* $\overline{0}$*: for all* x *in* $\mathbb{K}$*,* $x \otimes \overline{0} = \overline{0} \times x = \overline{0}$

Furthermore, the semiring is said to be **closed** if

- for all $x \in \mathbb{K}$, $\bigoplus_n x^{\otimes n}$ is well defined and in $\mathbb{K}$ (this operator is denoted x^*),
- associativity, commutativity and distributivity hold for countable sums of elements of $\mathbb{K}$.

Example 2.3. *The probability semiring* $(\mathbb{R}^+, +, \times, 0, 1)$ *is a first natural example of a semiring that will be useful later. This semiring is not closed: we do not have the associativity, commutativity and distributivity of countable sums. Furthermore, the geometric sum does not converge for all positive number:* $\sum_n 1^n = \infty$.

Another classical semiring is the tropical semiring $(\mathbb{R}^+, min, +, \infty, 0)$*, associated with shortest distance problems. This one is closed: the* * *operator is* $\min(0, x, 2x, \dots) = 0$.

We can now define automata with weights over a semiring:

Definition 2.4 (Weighted automaton)**.**

Let $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ *be a semiring. A weighted automaton* $\mathcal{A}$ *over* $\mathbb{K}$ *is a quintuple* $(S, \Sigma, \lambda, \gamma, \mathit{F})$ *such that:*

- S *is a finite set of states,*
- Σ *is a finite alphabet,*
- $\lambda : S \to K$ *is the set of initial weights,*
- $\gamma : S \times \Sigma \times S \to K$ *is the function assigning weights to the transitions,*
- $\mathit{F} : S \to K$ *is the set of final weights.*

The weight of a finite path $\pi = t_1 \dots t_n$ is equal to $\bigotimes_{i=1}^n \gamma(t_i)$. Given an execution $\rho = t_1 \dots t_n$ with $s^-(\rho) = s$ and $s^+(\rho) = s'$, the weight of ρ is equal to

$$\lambda(s) \otimes \gamma(\pi) \otimes \mathit{F}(s')$$

Thanks to example 2.3, we saw that the closure of a semiring is very restrictive. However, when considering weighted automata, an infinite sum will occur when we consider the possibility to go through a cycle arbitrary many times. Then, we only need the closure with respect to these elements.

Definition 2.5 (Closure w.r.t a weighted automaton)**.**

Let $\mathcal{A}$ *be a weighted automaton over* $\mathbb{K}$*.* $\mathbb{K}$ *is said to be closed over* $\mathcal{A}$ *if for all cycles* c *of* $\mathcal{A}$*,*

- $\gamma(c)^*$ *is well defined and in* $\mathbb{K}$*,*

- *associativity, commutativity and distributivity hold for these specific countable sums: for all* $x, y, .$

Example 2.4. *For a weighted automaton* $\mathcal{A}$ *over the probabilistic semiring, if all cycles have a weight lower than* 1*, then this semiring is closed with respect to* $\mathcal{A}$*.*

2.1.4 Stochastic systems

In the following, we consider specific quantitative models, where the quantities are probabilities. First, we consider systems where there may be some non-determinism but the information about the current state is available. The simplest stochastic system we consider is Markov Chains:

Definition 2.6 (Finite state discrete time Markov chain)**.**
Let S *be a set of states.*
A Markov chain $\mathcal{M}$ *can be modeled as a triple* (S, M, μ_0) *where:*

- S *is a set of states*
- $M \in [0,1]^{|S|\times|S|}$ *is the stochastic transition matrix.*
- $\mu_0 : S \to [0,1]$ *with* $\sum_s \mu_0(s) = 1$ *is the initial probability mapping.*

Notions of paths and executions are naturally extended from those on transition systems.

A related definition (for statisticians) is that a discrete time Markov chain is a sequence of random variables $(X_i)_{i\in\mathbb{N}}$ that can take a finite number of values with the Markov property, *i.e.*, such that the probability of the next state given all the past of the run depends only on the last state of the run, that is

$$\Pr(X_n = s | X_{n-1}, \ldots, X_1) = \Pr(X_n = s | X_{n-1})$$

Then, the probability distribution after m steps is equal to $(\mu_0(s_0), \ldots, \mu_0(s_n))M^m$.

Example 2.5. *Figure 2.3 depicts a Markov chain with initial distribution* $\mu_0(s_0) = 1$*, and for all* $i > 0$ $\mu_0(s_i) = 0$*. After one step, the probability distribution is* $(0, \frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ *and after two steps, it is* $(0, \frac{1}{2}, \frac{3}{8}, \frac{1}{8})$*.*

Another class of fully observable probabilistic systems are Markov Decision Processes (MDPs). They provide a framework to model decision making when possibilities are partly random. In MDPs, the alphabet is the set of choices. At each step, the user chooses

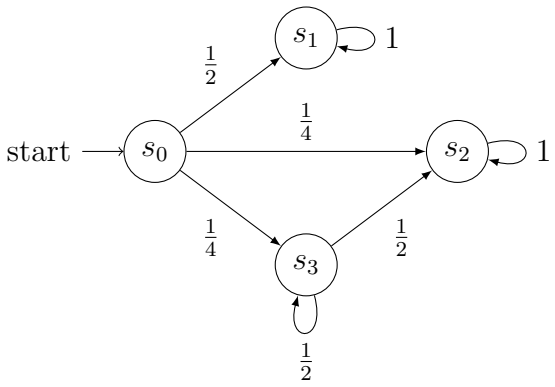

Figure 2.3: Example of a Markov Chain $\mathcal{M}$.

an action available, and the resulting state is chosen at random among the different possibilities.

Definition 2.7 (Markov Decision Processes)**.**

A Markov Decision Process $\mathcal{A}$ is a quadruple $(S, \Sigma, \mu_0, (M_a)_{a\in\Sigma})$ with:

- *S is a set of states,*
- *Σ an alphabet,*
- *$\mu_0 : S \to [0, 1]$ with $\sum_s \mu_0(s) = 1$ is the initial probability mapping,*
- *for every $a \in \Sigma$, M_a is a matrix such that each line is either stochastic (the state allows a) or zero (a is not allowed).*

2.1.5 Partially observable stochastic systems

In this subsection we consider partially observed stochastic systems. We explained in section 2.1.2 that the information about the current state is hidden. In general, there are two alphabets: the control alphabet Σ_c and the signal one Σ_s. Intuitively, the control alphabet Σ_c is the set of actions a player can choose and Σ_s represents the set of observations that can be raised. These two alphabets have then orthogonal meanings. When considering stochastic systems, the meanings of the probabilities associated to the control alphabet and the signal alphabet are again orthogonal. When a control action has been chosen, we want the sum of the probabilities of all possible outcomes of **this action** in one state to be 1. However, when considering the signal alphabet, we want the sum of the probabilities of **all possible outcoming signals** in one state to be 1.

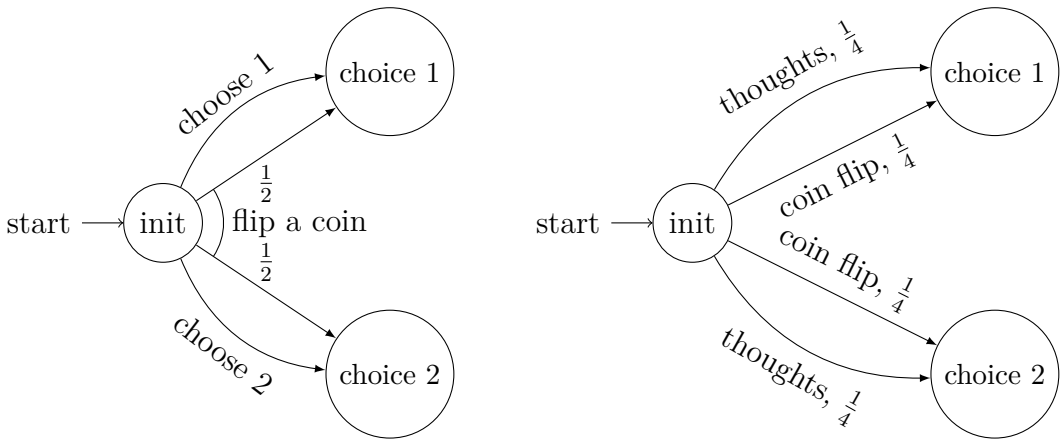

Figure 2.4: A model where the user has a choice represented by an MDP (left) and a model of an observer witnessing a possible strategy represented by an LMC (right).

Partially Observed MDPs (POMDPs) [Ast65] are a class of systems that can be represented with these two alphabets. The user chooses an available action (in Σ_c) and then a signal (in Σ_s) is raised with respect to the transition that was effectively performed. In POMDPs, states are usually partitioned into equivalence classes and the user receives as observation the class of the resulting state.

In this thesis, we focus on two kinds of models: *Labeled Markov Chains*, where the user has no control and only receives signals. The second is *Rabin's Probabilistic Finite Automata*, which are POMDPs with only one class of equivalence: the current state is totally hidden.Notice than in both case, we will need only one alphabet. Thus, to simplify, we will write Σ instead of Σ_c and Σ_s, but remember that they have different meanings.

Example 2.6. *In figure 2.4, we model the same situation where someone has to make a choice between two possibilities. One model (left) is from the choice maker. He can either decide to choose* 1 *or* 2*, or he is undecided and will flip a coin. There are then two deterministic choices and one non-deterministic. On the right side, an observer looks at the choice maker. The observer will only see if a coin has been flipped or not.*

Words describing observation sequences

First, we discuss about models where the words associated to executions represent a signal given to an observer. This can be represented by a special case of weighted automata. A weighted automata is said to be stochastic if it is over the probability semiring $(\mathbb{R}^+, +, \times, 0, 1)$ and for all state s, $\sum_{a,s'} \gamma((s, a, s')) = 1$. In [Moh02b], this system is called

"probabilistic automaton". However, we will avoid to use this name, since it may be confused with Rabin's "probabilistic finite automaton" that will also be of interest in this document. In [Mug96], it is referenced as a stochastic automaton. When necessary, we will use this name.

The corresponding event system (*i.e.*, with no notion of final state) is the Labeled Markov Chain [CK14; DHR08]. In the literature, it appears with different names. In [BP66], it is called Hidden Markov Model. It also appears as probabilistic-LTS in [Lef18] and Hidden Markov Chain in [KS16].

Definition 2.8 (Labeled Markov Chain)**.**

A Labeled Markov Chain $\mathcal{M}$ is a quadruple (S, Σ, μ_0, p) such that:

- *S is a finite set of states,*
- *Σ is a finite alphabet,*
- *$\mu_0 : S \to [0, 1]$ with $\sum_s \mu_0(s) = 1$ is the initial probability mapping,*
- *$p : S \times \Sigma \times S \to K$ with $\sum_{a,s'} p((s, a, s')) = 1$ gives the probabilities of the transitions.*

We may notice that a Markov Chain is a labeled Markov Chain with exactly one letter. The (possibly infinite) language of a LMC is the set of words w such there exists an execution in this LMC labeled by this word.

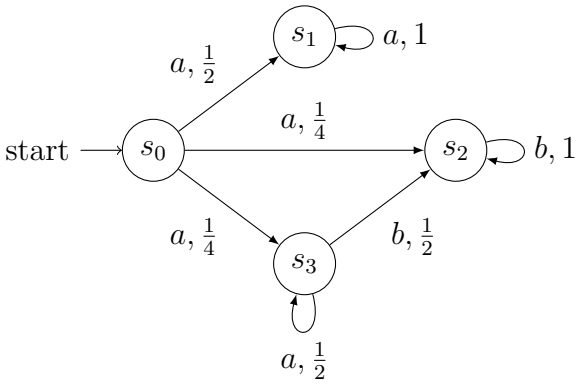

Figure 2.5: Example of an LMC $\mathcal{M}$.

Example 2.7. *In figure 2.5, the word ab is the label of two distinct runs, both ending in s_2. The probability of ab is then equal to $\frac{1}{4} \times 1 + \frac{1}{4} \times \frac{1}{2}$.*

Words modeling control sequences

In this document, we will also consider Rabin's Probabilistic Finite Automaton (PFA), introduced in [Rab63]. A PFA can be seen as a MDP where states are totally unobservable. We also add stopping time modeled by final states.

Definition 2.9 (Probabilistic Finite Automata)**.**

A complete Probabilistic Finite Automaton $\mathcal{A}$ is a quintuple $(S, \Sigma, \mu_0, (M_a)_{a\in\Sigma}, F)$ with:

- *S is a set of states,*
- *Σ an alphabet,*
- *$\mu_0 : S \to [0, 1]$ with $\sum_s \mu_0(s) = 1$ is the initial probability mapping,*
- *for every $a \in \Sigma$, M_a is a matrix such that each line is either stochastic (the state allows a) or zero (a is not allowed),*
- *$F \subseteq S$ is the set of final states.*

Furthermore, if for every letter a, M_a is a stochastic matrix then $\mathcal{A}$ is said to be complete. Every PFA can be completed by adding a dummy state with self-loops and adding transitions to this state for every missing letter. Given a distribution δ and a letter a, the distribution $\delta' = \delta \cdot a$ is defined as $\delta'(t) = \sum_s \delta(s)M_a(s, t)$. This can be naturally extended to words with for all distribution δ, for all word w and letter a, $\delta\cdot(wa) = (\delta\cdot w)a$. The probability of acceptation of a word w is $\sum_{t\in F} \mu_0 \cdot w(t)$.

Example 2.8.

In figure 2.6, an execution for the word aba is accepted in $\mathcal{A}$ if it ends in state s_1. This has a probability of $0.7 \times 1 \times 0.5 + 0.3 \times 0.6 \times 0.5 + 0.3 \times 0.4 \times 0.7 = 0.524$.

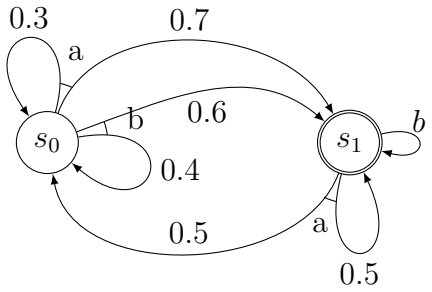

Figure 2.6: Example of a PFA $\mathcal{A}$.

2.1.6 Construction of a probability measure on infinite words

On the different systems we defined, we gave the definition of the probability of a finite execution and its cylinder. However, in some applications, we would like to discuss about set of infinite executions. Intuitively, the probability of an infinite run π should be the limit of the probabilities of its prefixes. However, in general, this limit is 0, we then need do state that the set of infinite runs is measurable and give its measure. Caratheodory's theorem[1] [AD00] allows us to correctly define this measure on these infinite runs. For that, we recall some definitions and then state the theorem. First, we need building blocks. We remind the definition of a cylinder, given in section 2.1.1: for a finite run ρ, the cylinder of ρ, denoted $Cyl(\rho)$ is the set of infinite runs ρ' such that ρ is a prefix of ρ', and for a word w, the cylinder of w is the set of infinite words that have w as a prefix. We state that the set of the union of cylinders is a topology on the set of infinite runs:

Definition 2.10 (Topology)**.**
Given a set X, Y is said to be a topology on X if

- $\emptyset \in Y$,
- $X \in Y$,
- *Y is stable by (any) union, i.e., if for all $i \in I$, $O_i \in Y$ then $\bigcup_i O_i \in Y$,*
- *Y is stable by finite intersection, i.e., if for all $i \in [\![1, n]\!]$, $O_i \in Y$ then $\bigcap_i O_i \in Y$.*

An element O in Y is called an open set.

Example 2.9. *Let us consider the alphabet $\Sigma = \{a, b\}$. The set of infinite words having at least one a can be expressed as the union of the cylinders of words ending by an a: $\{w \in \Sigma^\omega,\ a \in w\} = \bigcup_{w \in \Sigma^*} Cyl(wa)$. However, the set of words having no a is not an open set: we need infinite intersection to define it: $\{w \in \Sigma^\omega,\ a \notin w\} = \bigcap_{w \in \Sigma^*} Cyl(wb)$.*

From this set of open sets, we can inductively define the Borel hierarchy generated by the open sets Y and the complement operator:

Definition 2.11 (Borel hierarchy)**.**
The Borel hierarchy is given by the classes $\mathbf{\Sigma}^0_\alpha$, $\mathbf{\Pi}^0_\alpha$, $\mathbf{\Delta}^0_\alpha$ for every countable ordinal α, such as:

- *$\mathbf{\Sigma}^0_1$ is the set of open sets,*
- *$\forall \alpha \geq 1$, $B \in \mathbf{\Pi}^0_\alpha$ iff $B^C \in \mathbf{\Sigma}^0_\alpha$,*

1. also called Hahn-Kolmogorov, or Caratheodory-Hahn

- $\forall \alpha \geq 2$, $B \in \mathbf{\Sigma}^0_\alpha$ *iff there exists a family (possibly infinite)* (B_i) *in* $B \in \mathbf{\Pi}^0_{\alpha-1}$ *and* $B = \bigcup B_i$,
- $\forall \alpha \geq 1$, $\mathbf{\Delta}^0_\alpha = \mathbf{\Sigma}^0_\alpha \cup \mathbf{\Pi}^0_\alpha$.

A set in some class of the Borel hierarchy is called a Borel set.

Example 2.10. *Continuing example 2.9, the set of words having no a is in* $\mathbf{\Pi}^0_1$*, as it is the complement of the set of words having at least one a. Then, the set of words having only a finite number of a (i.e.,* $\{wb^\omega, w \in \Sigma^*\}$ *is in* $\mathbf{\Sigma}^0_2$*, as the uncountable union over n of words of length n followed by* b^ω*. Hence, as its complement, the set of words having infinitely many a is in* $\mathbf{\Pi}^0_2$.

We can notice that the Borel hierarchy defines a hierarchy of complexity on sets of infinite words.

In the following, we recall that Borel sets are measurable and that for all properties we want to measure, the set of runs that satisfy this property can be expressed as a Borel set. For now, we introduce some additional vocabulary that will allow us to state the Caratheodory theorem.

Definition 2.12 (Ring of sets)**.**

Given a set X*, a ring of sets* $\mathcal{R}$ *of* X *is a subset of* 2^X *containing the empty set, closed under pairwise union and relative complement, that is:*

- $\emptyset \in \mathcal{R}$,
- $\forall A, B \in \mathcal{R}, A \cup B \in \mathcal{R}$,
- $\forall A, B \in \mathcal{R}, A \setminus B \in \mathcal{R}$.

A σ-algebra is a ring of sets with additional requirements:

Definition 2.13 (σ-algebra)**.**

Given a set X*, a* σ*-algebra* $\mathcal{S}$ *of* X *is a subset of* 2^X *containing the empty set, closed under countable union and complement, that is:*

- $\emptyset \in \mathcal{S}$
- $\forall (A_i)_{i \in \mathbb{N}} \in \mathcal{S}, \bigcup A_i \in \mathcal{S}$
- $\forall A \in \mathcal{S}, X \setminus A \in \mathcal{S}$

Notice that the set of Borel sets is the σ-algebra generated by the open sets.

Definition 2.14 (Pre-measure and measure)**.** *A pre-measure* μ *on a ring of sets* $\mathcal{R}$ *is a function* $\mu : \mathcal{R} \to \mathbb{R}^+$ *such that:*

- $\mu(\emptyset) = 0$
- *for all countable family of sets of* $\mathcal{R}$ *pairwise disjoint* $(A_i)_{i\in\mathbb{N}}$, $\mu(\bigcup A_i) = \sum_i \mu(A_i)$.

If $\mathcal{R}$ *is a* σ*-algebra, then* μ *is called a measure.* μ *is called* σ*-finite if there exists a countable collection* $(A_i)_{i\in\mathbb{N}} \in \mathcal{R}$ *such that* $\mathcal{R} = \bigcup A_i$.

The probability we defined on finite executions is a pre-measure. Caratheodory's theorem allows to define the corresponding measure:

Theorem 2.1 (Caratheodory's extension theorem [AD00])**.**

Let X *be a space and* $\mathcal{R}$ *a set of rings on* X, μ *a pre-measure on* $\mathcal{R}$ *that is* σ*-finite. Then there exists a unique measure* μ' *on the sigma algebra generated by* $\mathcal{R}$ *such that for all* $A \in \mathcal{R}$, $\mu(A) = \mu'(A)$.

In our case, we have that the probability of the empty set of words is 0, and that for disjoint set of words (A_i), $\mathbb{P}(\bigcup A_i) = \sum_i \mathbb{P}(A_i)$. By using this theorem and extending the pre-measure defined on the set of rings generated by the open sets, we proved that all Borel sets are measurable.

Example 2.11. *In figure 2.11, every finite word of length* n *has probability* $\frac{1}{2^n}$*, and we denote by* μ *the pre-measure associated. The set* W *of words having an infinite number of* a *is not in the ring of sets generated by the open sets, however, by extending* μ *into* p*, we find that* $p(W) = 1$.

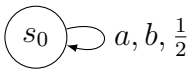

Figure 2.7: An LMC such that for all $w \in \Sigma^n$, $P(w) = \frac{1}{2^n}$.

2.2 Vocabulary and properties of Markov Chains

In this section, we remind some definitions about Markov Chains and some of their properties. First, we define irreducibility, that is the underlying graph is strongly connected and then the periodicity.

Definition 2.15 (Irreducibility)**.**

Let $\mathcal{M}$ *be a Markov Chain.* $\mathcal{M}$ *is irreducible if for all pair of states* s, t *of* $\mathcal{M}$*, there is a path from* s *to* t.

Definition 2.16 (Period)**.** *Let $\mathcal{M}$ be a Markov Chain and s one of its states. Its period d is the GCD of the lengths of all cycles on s.*

Definition 2.17 (Aperiodicity)**.**

Let $\mathcal{M}$ be a Markov Chain. $\mathcal{M}$ is aperiodic if for all state s_i of $\mathcal{M}$, its period d_i is 1*.*

If a Markov chain is both irreducible and aperiodic then it is called **ergodic**. Ergodic Markov Chains have interesting properties, especially about stationary distributions:

Definition 2.18 (Stationary distribution)**.**

Let $\mathcal{M}$ be a Markov Chain, M its transition matrix and δ a probability distribution on the states of $\mathcal{M}$. δ is said to be stationary if $\delta M = \delta$.

Theorem 2.2 (Fundamental Theorem of Markov Chains)**.** *Let $\mathcal{M}$ be an ergodic Markov Chain and M its transition matrix. Then $\mathcal{M}$ admits a unique stationary distribution σ. Further, $M^t_{x,y} \to_{t\to\infty} \sigma_y$ for all states x, y.*

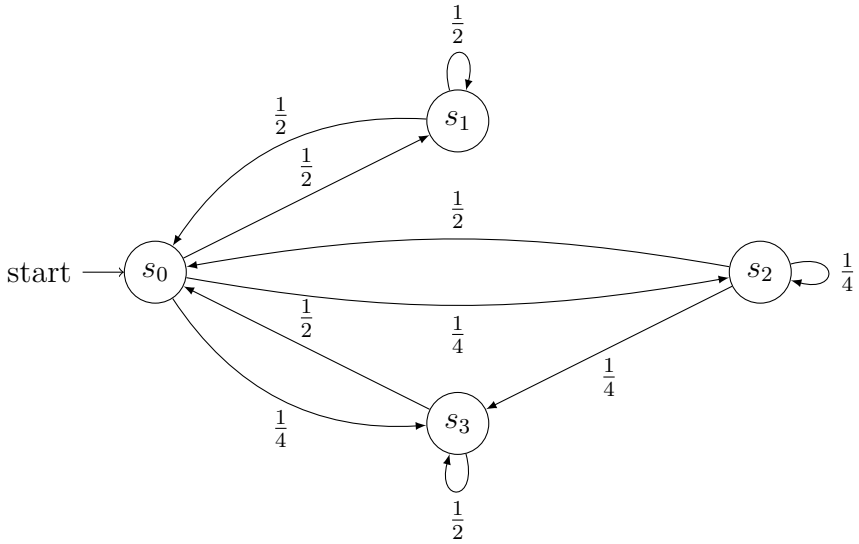

Figure 2.8: Example of a Markov Chain $\mathcal{M}$.

Example 2.12. *The Markov Chain $\mathcal{M}$ presented in 2.8 is ergodic: it is strongly connected and there are self-loops ensuring that GCD of lengths of cycles on states is* 1*. Thus, $\mathcal{M}$*

admits a stationary distribution δ. This distribution satisfies the following system:

$$(\delta_0, \delta_1, \delta_2, \delta_3) \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix} = (\delta_0, \delta_1, \delta_2, \delta_3) \text{ and } \delta_0 + \delta_1 + \delta_2 + \delta_3 = 1$$

The solution of this system is $(\frac{1}{3}, \frac{1}{3}, \frac{1}{9}, \frac{2}{9})$.

2.3 Vocabulary and properties of probability distributions

In this section, we recall some definitions and set notations around the notion of random variables and probability distributions. In the following, the random variables will be implicitly defined on probabilistic space $(\Omega, \mathcal{F}, \mathbb{P})$ with Ω the outcome possibilities, $\mathcal{F}$ a σ-algebra on the powerset of Ω and $\mathbb{P}$ the probability measure on $\mathcal{F}$. We will immediately place ourselves in the case where the values associated to outcomes are real numbers. Furthermore, most of the definitions will be given for the case of discrete random variables.

Definition 2.19 (Expected value)**.**
Let X be a real-valued random variable. The expected value of X is $\mathbb{E}[X] = \sum x \times \mathbb{P}(x)$.

Notice that for some random variables, this sum may diverge and the expected value is not properly defined.

Definition 2.20 (Moments)**.**
Let X be a real-valued random variable. The $n-$th moment of X is given by $\mathbb{E}[X^n]$.

Similarly, moments are properly defined when the sum does not diverge.

Example 2.13. *Let us consider the random variable X following a geometric law of parameter $\frac{1}{2}$, i.e., for all $n \geq 1$, $\mathbb{P}(X = n) = \frac{1}{2^n}$. The expected value of X is equal to $\sum_n \frac{n}{2^n} = 2$. Its variance is equal to $\mathbb{E}[X^2] - \mathbb{E}[X]^2 = 2$.*

An important property we will often try to ensure is that the probability of some bad case is low enough. Generally speaking, this bad case can be such as a long waiting time. To ensure that this has a low probability, we use concentration inequalities. We recall two of the most used.

Proposition 2.3 (Markov's inequality)**.**

Let X be a real-valued random variable. Then, for all $\alpha > 0$, we have $\mathbb{P}(|X| \geq \alpha) \leq \frac{\mathbb{E}[|X|^n]}{\alpha^n}$.

One particular case of Markov's inequality is by using $X = Y - \mathbb{E}[Y]$ for Y a real valued random variable. This is the Chebychev's inequality that guarantees that no more than a certain fraction of the distribution can be at more than a certain distance of the mean value.

Proposition 2.4 (Chernoff's inequality)**.**

Let X be a real-valued random variable. Then, for all $\alpha, t > 0$, we have $\mathbb{P}(|X| \geq \alpha) \leq \frac{\mathbb{E}[e^{t \cdot X}]}{e^{t \cdot a}}$.

We also remind the different notions of convergences from the weakest to the strongest:

Definition 2.21 (Convergence in law)**.**

A sequence of random variables $(X_n)_{n\in\mathbb{N}}$ converges in law to X (denoted $X_n \xrightarrow{\mathcal{L}} X$) iff for all continuous bounded function φ,

$$lim_n \mathbb{E}[\varphi(X_n)] = \mathbb{E}[\varphi(X)]$$

Definition 2.22 (Convergence in probability)**.**

A sequence of random variables $(X_n)_{n\in\mathbb{N}}$ converges in probability to X (denoted $X_n \xrightarrow{p} X$) iff

$$\forall \varepsilon,\ \lim_n \mathbb{P}(|X_n - X| \geq \varepsilon) = 0$$

Definition 2.23 (L_p convergence)**.** *A sequence of random variables $(X_n)_{n\in\mathbb{N}}$ converges in L_p to X (denoted $X_n \xrightarrow{L_p} X$) iff*

$$\forall \varepsilon,\ \lim_n \mathbb{E}[|X_n - X|^p] = 0$$

Definition 2.24 (Almost sure convergence)**.**

A sequence of random variables $(X_n)_{n\in\mathbb{N}}$ almost surely converges to X (denoted $X_n \xrightarrow{p.s} X$) iff $\mathbb{P}(\lim_n X_n = X) = 1$.

Proposition 2.5. *Let $(X_n)_{n\in\mathbb{N}}$ be a sequence of random variables, and X a random variable. We have*

$$X_n \xrightarrow{p.s} X \Rightarrow X_n \xrightarrow{p} X \Rightarrow X_n \xrightarrow{\mathcal{L}} X$$

In this thesis, we want to evaluate limit behaviors of series of random variables. To do that, the central limit theorem and the law of large numbers give answers to this question.

Theorem 2.6 ((Strong) Law of large numbers)**.**

Let $X_1, \ldots, X_n$ be a collection of independent and identically distributed (i.i.d.) random variables drawn from a distribution of expected value given by μ and $S_n = \frac{\sum_{i=1}^{n} X_i}{n}$.

S_n converges almost surely to μ.

Theorem 2.7 (Central limit theorem)**.**

Let $X_1, \ldots, X_n$ be a collection of independent and identically distributed (i.i.d.) random variables drawn from a distribution of expected value given by μ, finite variance given by σ^2 and $S_n = \frac{\sum_{i=1}^{n} X_i}{n}$.

$\sqrt{n}(S_n - \mu)$ converges almost surely to the normal law of parameters $(0, \sigma)$.

2.4 Questions of interest for the verification of stochastic systems

Along this document, we will talk about different kinds of properties. We will go from specific properties to more general ones. In this section, we present basic definitions and vocabulary that will allow the reader to have a better catch on the progression we will make. Techniques and specific state of the art will be included in corresponding sections.

2.4.1 Reachability

A first simple but necessary property that we want to tackle is the reachability, that can be considered in different ways. First, it can be reckoned in a qualitative way: given a target (*i.e.*, a set of states), is there an execution that reach this set. Then, we can investigate the quantitative version of this question: how much of the executions reach this target? Finally, we can wonder how fast executions reach the target.

Let $\mathcal{A}$ be a labeled Markov chain, and s a target. The set of infinite executions that eventually reach s is given by $\bigcup_{s^+(\rho)=s} Cyl(\rho)$. This formulation as a countable union of cylinders show us that unsurprisingly this set is measurable. This can be rewritten as the sum of the probabilities of all paths that go from s_0 to s without reaching s before:

$$\sum_{\pi \in \mathcal{P}_s(s_0, s)} \mathbb{P}(\pi)$$

Notice that for all pair of paths π, π' in $P_s(s_0, s)$, we have that π is not a prefix of π'. Since the set of runs that reach s can be expressed as the union of the cylinders of all finite runs that reach s, the set of runs that satisfy the reachability problem is measurable. Several algorithms allow to compute this quantity. We present one in section 2.5.1.

2.4.2 Expressing general properties as temporal logics

After reachability, we focus on general properties that have broader expressivity. In this work, we are interested in learning discrete time Markov Chains such that the learnt one is close enough to the model with high probability. To describe this notion of proximity, we use temporal logic [BPM83; Pnu77]. In this section, we introduce formalisms that allow one to express such properties. State of the art on the subject of learning Markov Chains will be addressed in Chapter 5.

Temporal logics

Temporal logics allow one to reason about properties related to a succession of events in an execution. Formulas in temporal logics can express complex properties, such as liveness (always φ), safety (never φ), fairness (if φ then eventually ψ)... Temporal logics are broadly used in formal verification. There is a huge variety of temporal logics, however in this document we will focus on two of them: Linear Temporal Logic (LTL) [Pnu77] and Computation Tree Logic (CTL) [CE81]. First, we recall what these logics are, especially on our models. In the following, we assume $\mathcal{M}$ to be a discrete time Markov Chain.

The first temporal logic we consider, LTL, gives a way to specify properties on a single execution.

Definition 2.25 (Linear Temporal Logic)**.**

Let S be the set of state names of $\mathcal{M}$. LTL is built upon the following grammar:

$$\varphi ::= s \in S \mid \varphi \wedge \varphi \mid \neg\varphi \mid X\varphi \mid \varphi U \varphi$$

We define the semantic as follows: an execution ρ satisfies a formula φ, denoted $\rho \models \varphi$ according to the following rules:

- $\rho \models \boldsymbol{true}$,
- $\rho \models s$ iff $s^-(\rho) = s$,
- $\rho \models \varphi \wedge \psi$ iff $\rho \models \varphi$ and $\rho \models \psi$,
- $\rho \models \neg\varphi$ iff $\rho \not\models \varphi$,

- $t\rho \models X\varphi$ iff $\rho \models \varphi$,
- $\rho \models \varphi U \psi$ iff there exists n such that $\rho = t_1 \dots t_n \rho'$ and for all $i \leq n$, $t_i \dots t_n \rho' \models \varphi$ and $\rho' \models \psi$.

A Markov chain $\mathcal{M}$ satisfies an LTL property φ denoted $\mathcal{M} \models \varphi$ iff all traces of executions in $\mathcal{M}$ satisfy φ.

We have defined a minimal set of operators, however, for commodity, we may use the following operators:

- or: $\varphi \vee \psi \equiv \neg(\neg\varphi \wedge \neg\psi)$,
- false: $\boldsymbol{false} \equiv \neg\boldsymbol{true}$,
- eventually: $F\varphi \equiv \boldsymbol{true}\, U\varphi$,
- always: $G\varphi \equiv \neg F \neg\varphi$.

By contrast with LTL, CTL does not handle single runs, but rather on execution trees, *i.e.*, the possible futures of a run. There are two kinds of quantifiers: those on paths (Exists a future and All possible futures), and the path specific quantifiers.

Definition 2.26 (Computation Tree Logic)**.**
Let S be the set of state names of $\mathcal{M}$. CTL is built upon the following grammar:

$$\varphi :== S \mid \varphi \wedge \varphi \mid \neg\varphi \mid EG\varphi \mid EU\varphi \mid EX\varphi$$

Unlike in LTL, the semantics of CTL formulas is not defined on executions but rather on states, we denote it $(\mathcal{M}, s) \models \varphi$. We will allow ourselves to write it $s \models \varphi$ when there is no ambiguity on the system considered:

- $s \models \boldsymbol{true}$,
- $s \models s$,
- $s \models \varphi \wedge \psi$ iff $s \models \varphi$ and $s \models \psi$,
- $s \models \neg\varphi$ iff $s \not\models \varphi$,
- $s \models EG\varphi$ iff there exists $(s_i)_{i\in\mathbb{N}}$, for all i, $p(s_i, s_{i+1}) > 0$ and $s_i \models \varphi$,
- $s \models E[\varphi U \psi]$ iff there exists $s_1 \dots s_n$ such that $s = s_1$, for all $i < n$, $p(s_i, s_{i+1}) > 0$ and $s_i \models \varphi$, and $s_n \models \psi$,
- $s \models EX\varphi$ iff there exists s', $p(s, s') > 0$ and $s' \models \varphi$.

A Markov chain $\mathcal{M}$ satisfies a CTL property φ denoted $\mathcal{M} \models \varphi$ iff every initial state satisfies φ.

From this minimal set of operators, we may derive the following operators:

- or: $\varphi \vee \psi \equiv \neg(\neg\varphi \wedge \neg\psi)$,
- false: $\boldsymbol{false} \equiv \neg\boldsymbol{true}$,
- exists eventually: $EF\varphi \equiv E[\boldsymbol{true}\ U\varphi]$,
- always eventually: $AF\varphi \equiv \neg EG\neg\varphi$,
- always forever: $AG\varphi \equiv \neg EF\neg\varphi$,
- always next: $AX\varphi \equiv \neg EX\neg\varphi$,
- always until: $A[\varphi U\psi] \equiv \neg(E[(\neg\psi)U\neg(\varphi \vee \psi)] \vee EG\neg\psi)$.

An LTL formula φ and a CTL formula ψ are said to be equivalent if for all $\mathcal{M}$, $\mathcal{M} \models \varphi$ iff $\mathcal{M} \models \psi$.

CTL and LTL are incomparable. Each one allows one to express properties that are inexpressible in the other. As an example, the property $AGEFp$ in CTL has no equivalent in LTL: this property states that every execution at every point has the possibility to reach a state where p is true. LTL does not allow to reason about this possibility: as a logic about traces, it can only state if it does or does not reach a state where p is true. Conversely, the LTL formula $F(p \wedge Xp)$ has no equivalent in CTL.

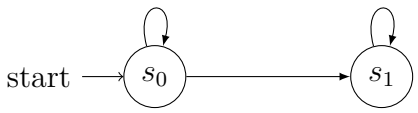

Figure 2.9: A model satisfying $AGEFs_1$ but not GFs_1.

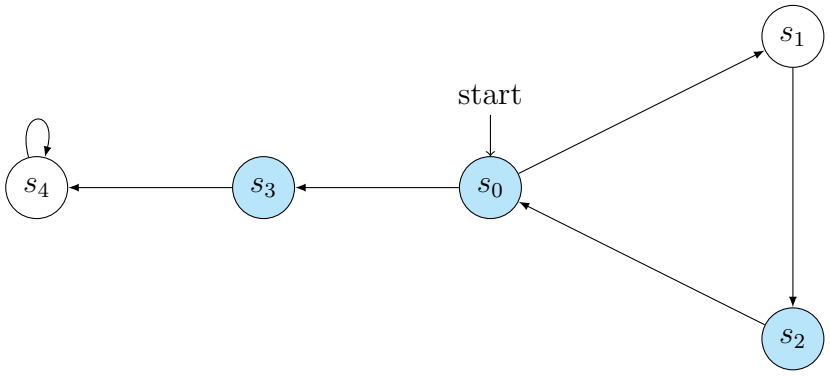

Figure 2.10: A model satisfying $F((s_0 \vee s_2 \vee s_3) \wedge X(s_0 \vee s_2 \vee s_3))$ but not $AF((s_0 \vee s_2 \vee s_3) \wedge AX(s_0 \vee s_2 \vee s_3))$.

In figure 2.9, at every point of every execution, there is a possibility to reach s_1. Hence, this model satisfies $AGEFs_1$. However, it does not satisfy GFs_1: the execution s_0^ω never

reaches s_1. In figure 2.10, every execution will eventually in two consecutive blue states, hence the formula $F((s_0 \vee s_2 \vee s_3) \wedge X(s_0 \vee s_2 \vee s_3))$ is satisfied. However, the execution $s_0 s_3 s_4^\omega$ does not satisfy $AF((s_0 \vee s_2 \vee s_3) \wedge AX(s_0 \vee s_2 \vee s_3))$: there is a possible successor to s_0 that is not blue. These two examples illustrate the difference between both logics. In order to state some positive results for both CTL and LTL, we will sometimes state them for the logic CTL* [Pnu77]. This temporal logic is a generalization of both CTL and LTL. Since we will not use this logic for other purposes in this thesis, we do not recall its formal definition.

We want a way to quantify the proportion of executions that satisfy the formula in order to know more than if a system does or does not satisfy a formula. We have the following result on CTL*:

Theorem 2.8 ([BK08])**.** *Given a system $\mathcal{M}$ and a CTL* formula φ, the volume of executions of $\mathcal{M}$ satisfying the formula φ is measurable.*

Therefore, we will be able to define probability measures accordingly to the systems and the formulas.

2.5 General algorithmic results

In this section, we present some general results and algorithms that will be useful alongside this document. Results treated in this section are not specifically related to a single chapter. We first describe polynomial time algorithms for reachability in fully observable systems. Perfect information is key for these algorithms to run in PTIME. In the presence of partial information, many problems become undecidable. We recall some of these undecidability results on PFA at the end of the section.

2.5.1 PTIME algorithms for quantifying reachability in fully observable systems

Floyd-Warshall algorithm on weighted automata

The probability to reach a set of states can be computed by using a well known algorithm, the Floyd-Warshall algorithm [Flo62]. Although this algorithm was first designed to answer the shortest path problem, its uses are wider. We present the extension of this algorithm to closed semirings. For instance, this extension has been used in [Moh02b; Cor+08] to compute quantities in some specific semirings.

Let $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ be a closed semiring, $\mathcal{A}$ a weighted automaton over $\mathbb{K}$ with states numbered from 0 to $n-1$, and let S_f be a set of states that cannot be intermediate states in a path and S_k the set of states $\{s_k, s_{k+1}, \ldots, s_{n-1}$. For convenience, let us assume that $S_f = \{s_{n-1} \ldots s_{n-m}\}$, *i.e.*, $S_f = S_{n-m}$. We recall that for a subset of states Q, $\mathcal{P}_Q(s_i, s_j)$ is the set of paths from s_i to s_j that do not have an intermediate state in Q. For two states s_i and s_j, the sets of paths $(\mathcal{P}_{S_k}(s_i, s_j))_{k<n}$ satisfy the following recursion:

- $\mathcal{P}_{init}(s_i, s_j) = \{(s_i, a, s_j), a \in \Sigma\}$,
- $\forall k \in [\![0; n-m-1]\!], \mathcal{P}_{S_k}(s_i, s_j) = \mathcal{P}_{S_{k-1}}(s_i, s_j) \uplus \mathcal{P}_{S_{k-1}}(s_i, s_k)\mathcal{P}_{S_{k-1}}(s_k, s_k)^*\mathcal{P}_{S_{k-1}}(s_k, s_j)$.

Then, $\mathcal{P}_{S_{n-m-1}}(s_0, s_{n-1})$ is exactly the set of paths going from s_0 to s_{n-1} that do not have an intermediate state in S_f. We define $W_k(i, j)$ the total weight of the paths in $\mathcal{P}_{S_k}(s_i, s_j)$. Since for all k the sets $\mathcal{P}_{S_{k-1}}(s_i, s_j)$ and $\mathcal{P}_{S_{k-1}}(s_i, s_k)\mathcal{P}_{S_{k-1}}(s_k, s_k)^*\mathcal{P}_{S_{k-1}}(s_k, s_j)$ are disjoint, we have by construction that $W_k(i, j) = W_{k-1} \oplus W_{k-1}(i, k) \otimes W_{k-1}(k, k)^* \otimes W_{k-1}(k, j)$. This gives us the skeleton of the algorithm, with pseudo-code presented in algorithm 1.

Algorithm 1 Floyd-Warshall algorithm for semirings

```
W_init is the matrix of weights of the transition
W_k are the matrix of weights at round k
for i, j ∈ [[1, n]] do
    W_0(i, j) ← W_init(i, j) ⊕ W_init(i, k) ⊗ W_init(k, k)* ⊗ W_init(k, j)
end for
for k ∈ [[1, n − m − 1]] do
    for i, j ∈ [[1, n]] do
        W_k(i, j) ← W_{k−1}(i, j) ⊕ W_{k−1}(i, k) ⊗ W_{k−1}(k, k)* ⊗ W_{k−1}(k, j)
    end for
end for
return W(1, n)
```

Notice that for readability we kept one matrix per k, but we can make this with only two matrices. In Chapter 3, we will use this algorithm with different semirings in order to obtain a large variety of information.

Theorem 2.9.

Let $\mathcal{A}$ be a weighted automaton with S as state set over $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ and $W = \bigoplus_{\pi \in \mathcal{P}_{S_f}(s_0, s_{n-1})} \gamma(\pi)$. Then there is an algorithm that computes W with complexity $O(|S|^3)$.

For instance, let us consider the problem "what is the probability to eventually reach s_{n-1} from s_0?". By taking the probability semiring, and $S_f = \{s_{n-1}\}$, we obtain exactly the probability to go from s_0 to s_{n-1}. We may also notice that by taking the tropical semiring, we obtain exactly a shortest path computation algorithm.

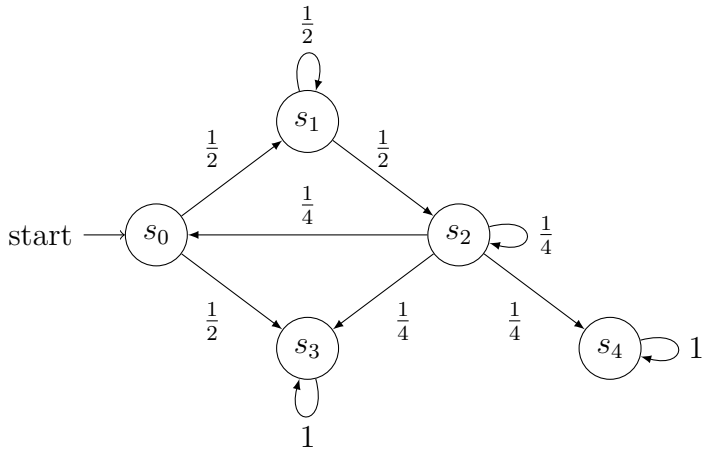

Figure 2.11: Markov Chain for the example of the Floyd-Warshall algorithm.

Example 2.14. *We apply the algorithm to the Markov chain described in figure 2.11: we want to calculate the probabilities to eventually reach s_3 and s_4 from the initial state. Thus, we need to use only s_0, s_1, s_2 as intermediate states. The semiring used here is $(\mathbb{R}^+, +, \times, 0, 1)$. We initialize the weights with the probabilities of the transitions in W. Then, we authorize s_0 as an intermediate state. For example, $W_0(s_2, s_1) = W(s_2, s_1) + W(s_2, s_0) \cdot W(s_0, s_0)^* \cdot W(s_0, s_1)$. Since $W(s_0, s_0) = 0$, we have $W(s_0, s_0)^* = 1$ and then $W_0(s_2, s_1) = 0 + \frac{1}{4} \cdot 1 \cdot \frac{1}{2} = \frac{1}{8}$.*

$$W = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad W_0 = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{8} & \frac{1}{4} & \frac{3}{8} & \frac{1}{4} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Then, s_1 is added as possible intermediate state. There is a self-loop on s_1, then $W_0(s_1, s_1)^* = \frac{1}{1-\frac{1}{2}} = 2$. Thus, $W_1(s_1, s_2) = W_0(s_1, s_2) + W_0(s_1, s_1) \cdot W_0(s_1, s_1)^* \cdot W_0(s_1, s_2) = \frac{1}{2} + \frac{1}{2} \cdot 2 \cdot \frac{1}{2} = 1$. Finally, we add s_2.

$$W_1 = \begin{bmatrix} 0 & 1 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 1 & 1 & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{3}{8} & \frac{3}{8} & \frac{1}{4} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad W_2 = \begin{bmatrix} \frac{1}{5} & \frac{6}{5} & \frac{4}{5} & \frac{4}{5} & \frac{1}{5} \\ \frac{2}{5} & \frac{7}{5} & \frac{8}{5} & \frac{3}{5} & \frac{2}{5} \\ \frac{2}{5} & \frac{2}{5} & \frac{3}{5} & \frac{3}{5} & \frac{2}{5} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

*Then, the probability to eventually reach s_3 from s_0 is $\frac{4}{5}$ and the probability to reach s_4 is $\frac{1}{5}$. Notice that the weights are **not** probabilities in general: it comes from the fact that in intermediate calculations we add weights of paths that can be prefix from each other. Hence, $W_2(s_0, s_0)$ is not the probability to eventually reach s_0 from s_0! (To compute this quantity, we would need not to use s_0 as an intermediate state.)*

Fix point algorithm for reachability probabilities

Reachability probabilities can also be computed by a fix point problem. Given a Markov Chain $\mathcal{M} = (S, M, \mu_0)$, the set of states S is partitioned in three sets. First is $S_{=0}$, the states that cannot reach the goal, *i.e.*, there is no path from those states to the target. The second is $S_{=1}$ the set of states that will eventually reach the target with probability one. Notice that these two sets $S_{=0}$ and $S_{=1}$ can be computed in linear time. The third one is $S_? = S \setminus (S_{=0} \cup S_{=1})$ the set of states that have a probability to reach the goal strictly between 0 and 1.

Then, $M_?$ is the restriction of M to states in $S_?$: $M_? = (M(s,t))_{s,t \in S_?}$. We also define b the vector of size $|S_?|$ such that $\forall s \in S_?$, $b_s = \sum_{t \in S_{=1}} M(s,t)$. Then, we have the following result:

Theorem 2.10 ([BK08])**.** *The vector $x = (\text{probability to reach the goal from } s)_{s \in S_?}$ is the least fixed point of the operator $\Psi : [0,1]^{S_?} \to [0,1]^{S_?}$ given by*

$$\Psi(y) = M_? y + b$$

Besides, if $x^{(0)}$ is the zero vector, and $x^{(n+1)} = \Psi(x^{(n)})$, then:

- *$x_s^{(n)}$ is the probability to reach the goal from state s in n steps or less,*
- *$x = \lim_{n \to \infty} x^{(n)}$.*

To go deeper, notice that all eigenvalues λ of $M_?$ satisfy $|\lambda| < 1$, thus by taking I the identity matrix of the appropriate dimension $I - M_?$ is invertible. Thus, the least fix point x is given by $(I - M_?)^{-1} b$. For more complex semirings, we will use the Floyd-Warshall algorithm as elements of semirings may not have an inverse.

Example 2.15. *We continue example 2.14. The goal is state s_3. Then, $S_{=0} = \{s_4\}$, $S_{=1} = \{s_3\}$, and $M_?$ and b are given by:*

$$M_? = \begin{bmatrix} 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & 0 & \frac{1}{4} \end{bmatrix} \qquad b = \begin{bmatrix} \frac{1}{2} \\ 0 \\ \frac{1}{4} \end{bmatrix}$$

Then we invert matrix $I - M_?$ and obtain:

$$(I - M_?)^{-1} = \begin{bmatrix} \frac{6}{5} & \frac{6}{5} & \frac{4}{5} \\ \frac{2}{5} & \frac{12}{5} & \frac{8}{5} \\ \frac{2}{5} & \frac{2}{5} & \frac{8}{5} \end{bmatrix} \qquad (I - M_?)^{-1}(b) = \begin{bmatrix} \frac{4}{5} \\ \frac{3}{5} \\ \frac{3}{5} \end{bmatrix}$$

Notice that as expected, we obtain the same results as with the Floyd-Warshall algorithm.

Reachability in MDPs

In MDPs, reachability is more subtle. Indeed, the possibility to reach a state and further its probability depend on the control strategy adopted, *i.e.*, depends on wich letters are played. As the probability depends on a controller $\mathcal{C}$, we denote it $\mathbb{P}_{\mathcal{C}}$. A natural question is to find the controller that gives the maximal probability $\mathbb{P}^{max}$ to reach the goal G. Here, we denote by $\mathbb{P}^{max}(\text{reach } G \text{ from } s)$ the maximal probability to reach G from s and $\mathbb{P}_{\mathcal{C}}(\text{reach } G \text{ from } s)$ the probability to reach G using controller $\mathcal{C}$. Finding the best controller seems difficult, as there are infinitely many. However, the following result reduces the number of controllers to investigate:

Theorem 2.11 ([BK08])**.** *Let $\mathcal{A}$ be an MDP, S its state set and G the goal. There exists a memoryless and pure controller such that for any state s,*

$$\mathbb{P}_{\mathcal{C}}(s) = \mathbb{P}^{max}(s)$$

The probability $\mathbb{P}^{max}(s)$ can be computed in polynomial time.

Memoryless means that the policy only depends on the current state but not on the previous ones. Pure means that for every state, the controller will only choose one action and forbid the others. In order to find the maximal probability, several methods are possible. A first is linear programming. The vector $(x_s)_{s \in S}$ with $x_s = \mathbb{P}^{max}(s)$ is the unique solution to the following linear program:

- $s \in G \Rightarrow x_s = 1$,
- G is not reachable from $s \Rightarrow x_s = 0$,
- for all $a \in \Sigma$, $x_s \geq \sum_{t \in S} M_a(s,t) \cdot x_t$.

Then, this linear program can be solved in polynomial time. Other methods, such as value iteration can be used to find the maximal probability. Similarly, minimal probability for reachability can be calculated in polynomial time: in the linear program, $\geq$ becomes $\leq$.

2.5.2 Undecidable problems on partially observable systems

As one can expect, going from perfect to imperfect information may make some problems much more difficult and often undecidable. In order to prove the undecidability of some of the problems we will consider on these partially observable systems, we will need to perform a reduction from other undecidable problems. In this section, we consider a few undecidable problems for PFA that we will use later.

We first define the language associated to a cut-point.

Definition 2.27 (Language wrt a cut-point)**.** *Let $\mathcal{A}$ be a PFA. Let $0 \leq \eta \leq 1$. The language of $\mathcal{A}$ with respect to the cut-point η is $L_{\mathcal{A}}(\eta) = \{w, P_{\mathcal{A}}(w) \geq \eta\}$.*

A canonical problem is to determine if this set is empty:

Definition 2.28 (Emptiness problem for a PFA)**.**

Given a PFA $\mathcal{A}$ and $0 \leq \eta \leq 1$, the emptiness problem consists in determining if $L_{\mathcal{A}}(\eta) = \emptyset$.

It is called strict emptiness if we use a strict equality instead of a large one. We notice that for $\eta = 0$, the strict inequality problem trivially reduces to the emptiness problem for a non-deterministic automaton, which is in NLOGSPACE. For $\eta = 1$, the emptiness problem reduces to the universality problem for non-deterministic automata which is PSPACE-complete [Koz77]. This gives us lower bounds for the complexity of the problem. However, the general case of this problem is much more difficult:

Theorem 2.12. *For $0 < \eta < 1$, the emptiness problem for a PFA is undecidable.*

The undecidability has first been proved in [Paz71]. New proofs have been presented in [MHC03] and in [GO10]. These more recent proofs have been made more legible and have precised a few points, such that the emptiness problem is undecidable for PFA with

only two probabilistic transitions, *i.e.*, there are two couples $s \in S, a \in \Sigma$ such that there exists t and $0 < M_a(s,t) < 1$ [GO10]. Thus, the undecidability threshold is crossed very quickly.

A second notion we will need is about isolated cut-points:

Definition 2.29 (Isolated cut-point)**.**

Let $\mathcal{A}$ be a PFA and $0 \leq \eta \leq 1$. η is said to be isolated with respect to $\mathcal{A}$ iff there exists $\varepsilon > 0$ such that for all $w \in \Sigma^*$, $|P_{\mathcal{A}}(w) - \eta| \geq \varepsilon$.

This definition leads us to the following decision problem:

Definition 2.30 (Isolation problem)**.**

Given a PFA $\mathcal{A}$ and $0 \leq \eta \leq 1$, the isolation problem consists in determining if η is isolated.

Again, this problem has been proved undecidable.

Theorem 2.13. *For $0 \leq \eta \leq 1$, the isolation problem is undecidable.*

The case $0 < \eta < 1$ has been proved in [Ber75]. The special case 0 and 1 have long stayed open, but have been showed undecidable in [GO10].

Chapter 3

Diagnosability analysis of Labeled Markov Chains

In everyday language, diagnosability is the ability to decide the nature and cause of something. This term is generally used in medicine: a doctor examines the symptoms in order to diagnose the illness, or in any field related to find causes of a problem (diagnose a car...). In some sense, this is what we want to do here: given a model of a system and one of its execution, we want to know if some "error" occurred. What we call an error can be any binary property, that is a question such that its answer is "yes" or "no". Our challenges are similar to those in other fields. First, we must define what we want to diagnose exactly. A second challenge is that the answer should be accurate: false positive and false negative may occur. Finally, an answer should be given "quickly". Diagnosability has no interest if it cannot be done in a reasonable time. These three notions can be summarized by *verdict, correctness* and *reactivity*.

These notions have been extensively studied for qualitative diagnosability for labeled transition systems and probabilistic transition systems. In this chapter, we want to extend these notions to quantified diagnosis.

The notion of verdict stays unchanged: given a system and an execution, several investigation are possible. We may want to know if it has been erroneous for sure, or if it has been correct for sure, or even both at the same time. In this chapter, we focus on the detection of errors in the context of permanent faults: the system cannot recover of a problem.

Correctness of diagnosis has been studied as determining if there is 0 error (or with probability 0) on the diagnosis. In this chapter, we tackle the issue of determining with precision the "amount of correctness", that is giving a measure of the diagnosability. This gives a hint on how close to diagnosable a system is.

Finally, reactivity has in the literature been studied in two ways. First, for diagnosability of LTSs, reactivity is the existence of a bounded delay after which an answer is given. This assessment is too strong for probabilistic systems, where no bounds can be

given but the probability of non diagnosis decreases to 0 with the time. Thus, the reactivity requirement must be adapted. This can be done by requiring that with probability 1 the fault will be detected in finite time. In this chapter, we investigate a quantitative version of reactivity, based on concentration inequalities. For example, we can ask "after how many steps am I sure to detect an error with probability at least 0.9?

This chapter is organized as follows: section 3.1 presents a state of the art on diagnosability. We start this state of the art in subsection 3.1.1 by presenting a definitions and results on diagnosability of finite Labeled Transition Systems. Then, this state of the art investigates what has been studied on diagnosability of probabilistic systems through A-diagnosability in subsection 3.1.2, AA-diagnosability in subsection 3.1.3 and finally on quantified diagnosis in 3.1.4. Section 3.2 presents a first contribution on quantified diagnosability. Definitions and semantics are given in subsection 3.2.1, related algorithms in subsection 3.2.2 and possible optimizations in subsection 3.2.3. In section 3.3, we present how to evaluate the distribution of diagnosis speed and its applications. Subsection 3.3.1 helps us to define the mathematical tools we need, subsection 3.3.2 states how to approximate the distribution of detection delay and subsections 3.3.3 and 3.3.4 discuss how to derive concentration bounds from this evaluation. Finally, we conclude and give some possibility of future work. Some related work that was not close enough to be put in the state of the art is also mentioned.

This chapter is based on the results presented in [BFG17; BFG18b; BFG18a].

3.1 State of the art

3.1.1 Diagnosis and diagnosability of finite LTS

As stated in the introduction, diagnosability is the ability to detect a binary property on a run of a system from the observation produced by that run. This property is usually called the presence of a "fault" event and detecting this occurrence is called the diagnosis. Therefore, our model must include a way to represent errors. Faults can equivalently be represented as a subset of the alphabet or a subset of states of the system. In the literature, the former is generally adopted. In this thesis, as we consider stochastic systems, it is simpler to use the latter, that is state based errors: changes of states are registered as soon as an observation is collected. Recall that we consider systems where the states are unobservable. Thus, we can remove the unobservable alphabet by a procedure of ε-removal about which we spoke about in Chapter 2.1.2. We will thus present some previous results

in our formalism.

Let $A = (S, \Sigma, I, T)$ be an LTS. The set of states S is partitioned in two: the set of correct states S_C and the set of faulty states S_F. The set of faulty states is said to be absorbing iff for all $s \in S_F$, for all $a \in \Sigma$, $(s, a, s') \in T$ implies that $s' \in S_F$. In this case, faults are said to be permanent.

Definition 3.1 (*Faulty run*)**.**
Let $A = (S, \Sigma, I, T)$ be an LTS, $S = S_C \uplus S_F$. Wlog, assume that $I \subseteq S_C$. A run ρ is said to be faulty if $s^+(\rho) \in S_F$.

A run ρ is said to be *minimal faulty* if all its strict prefixes are non-faulty, *i.e.*, correct. We denote by $F(A)$ (resp. $C(A)$) the set of faulty (resp. correct) runs of A. The set of faulty runs $\rho = \rho'\rho''$ such that ρ' is minimal faulty and $|\rho''| = n$ is denoted $F_n(A)$ and correct runs of length n is $C_n(A)$. For infinite runs, we denote it $F_\infty(A)$ (resp. $C_\infty(A)$). When there is no possible ambiguity, we will drop the name of the automaton (*e.g.* F_n instead of $F_n(A)$). Given a run ρ and its observation $o(\rho)$, our goal is to determine if ρ is faulty ($\rho \in F$) or correct ($\rho \in C$). Given an observation $o(\rho)$, three judgments on ρ are possible: either it can only be produced by correct runs and then it is correct, or it can only be produced by faulty runs and then it is faulty, or both and we cannot decide. This last judgment is called ambiguity.

Definition 3.2 (Ambiguity)**.**
Let $A = (S, \Sigma, I, T)$ be an LTS, $S = S_C \uplus S_F$. A run ρ of length $n \in \mathbb{N} \cup \{\infty\}$ is said to be faulty ambiguous if $\rho \in F_n$ and there exists $\rho' \in C_n$ such that $o(\rho) = o(\rho')$. Similarly, we define correct ambiguous runs.

We set some notation for all these runs: the set of faulty (resp correct) ambiguous runs is denoted F_{amb} (resp C_{amb}). We denote the set of faulty (resp. correct) non ambiguous runs F_{namb} (resp C_{namb}). As before, for all these sets, we will denote those of a specific length n by adding n as a subscript (ex: $F_{namb,n}$). Given an observation, we want to determine in which category it falls. This evaluation is performed by a diagnoser.

Definition 3.3 (Diagnoser)**.**
Let $A = (S, \Sigma, I, T)$ be an LTS, $S = S_C \uplus S_F$. A diagnoser is a function $D : \Sigma^* \to \{C, F, Amb\}$ such that for an observation $obs \in \Sigma^*$,:

- $D(obs) = C$ *if* $o^{-1}(obs) \subseteq C(A)$,
- $D(obs) = F$ *if* $o^{-1}(obs) \subseteq F(A)$,

- $D(obs) = Amb$ *else.*

As stated in the introduction, different verdicts are possible. Some works consider the diagnosis of faulty runs only, *i.e.*, intuitively a system should be diagnosable if any faulty run loses its ambiguity after a bounded extension. Others consider the diagnosis of all runs, that is a system should be diagnosable if any run loses its ambiguity after a bounded extension. In this chapter, we will consider the former and we present we will use as a starting point the definition of [Sam+96], that is we restrict ourselves to permanent faults and the study of diagnosability of faulty runs.

Definition 3.4 (k-Diagnosability of a run)**.**
Let $A = (S, \Sigma, I, T)$ be an LTS, $S = S_C \uplus S_F$. A faulty run $\rho \in F$ is said to be k-diagnosable if for all π such that $\rho\pi \in F$ and $|\pi| \geq k$ then $D(o(\rho\pi)) = F$.

Then, a notion of diagnosability for a system can be deduced from the notion of diagnosability for its runs. An LTS A is said to be k-diagnosable if all its runs are k-diagnosable. Originally, diagnosability has to be *uniform*: there is a bound such that all executions are diagnosed before this bound. More formally:

Definition 3.5 ((Uniform) Diagnosability of an LTS)**.**
An LTS A is said to be diagnosable if there exists k such that all its faulty runs are k-diagnosable.

Later we will see that the requirement of uniformity is not restrictive for LTSs, that is if for every faulty run ρ there is a k such that ρ is k-diagnosable, then there is a k such that all faulty runs ρ are k-diagnosable. A natural decision problem arises:

Definition 3.6 (Diagnosability)**.**
Given an LTS A, the diagnosability problem consists in determining whether A is diagnosable.

This problem has been studied in [Sam+96] and the exact complexity has been stated later [Jia+01; YL02].

An LTS A is diagnosable if there is no arbitrarily long ambiguous faulty sequence, *i.e.*, $\cup_{n\in\mathbb{N}} \cap_{m\geq n} F_{amb,m} = \emptyset$. We define the twin plant $\tilde{A}$ that allows one to tackle the complexity of the problem.

Definition 3.7 (Twin plant)**.**
Let $A = (S, \Sigma, I, T)$ be an LTS, $S = S_C \uplus S_F$ and $I \subseteq S_C$. The twin plant of A, denoted $\tilde{A}$ is the LTS $(\tilde{S}, \Sigma, \tilde{I}, \tilde{T})$ with:

- $\tilde{S} = S \times S_C$,
- $\tilde{I} = I \times I$,
- $\tilde{T} \subseteq \tilde{S} \times \Sigma \times \tilde{S}$, *with* $((s, s'), a, (s_1, s'_1)) \in \tilde{T}$ *iff* $(s, a, s_1) \in T$ *and* $(s', a, s'_1) \in T$.

Proposition 3.1. *Diagnosability of LTS is decidable and is NLOGSPACE-complete.*

A state $(s, s') \in \tilde{S}$ is said to be ambiguous if $s' \in S_C$ and $s \in S_F$. A path in $\tilde{A}$ is ambiguous if the last state of the path is ambiguous. The diagnosability problem is equivalent to the existence of an ambiguous cycle in the twin plant $\tilde{A}$. Then, a cycle detection algorithm is enough to decide if an LTS is diagnosable, hence the NLOGSPACE complexity. Notice that we only use $S \times S_C$ because we do not need to keep in memory states in $S_F \times S_F$: as faults are permanent, $S_F \times S_F$ is absorbing and cannot have an ambiguous future. NLOGSPACE-hardness was shown in [Bér+17].

This algorithm shows one more thing: on LTS, the notion of uniform diagnosability is not restrictive. Given an LTS A, if a run is diagnosable then it is $|S|^2$-diagnosable. Hence a uniform bound for all executions.

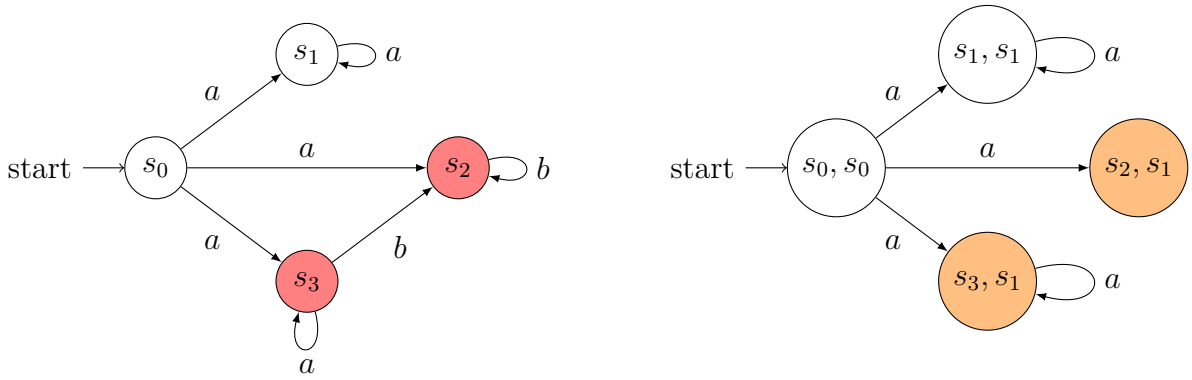

Figure 3.1: LTS A (left) with faulty states in red and its twin plant $\tilde{A}$ (right) with ambiguous states $\{s_1, s_3\}$ and $\{s_1, s_2\}$ in orange.

Example 3.1. *We extend example 2.1. In this automaton, we define the state of faulty states* $S_F = \{s_2, s_3\}$*, in red in figure 3.1 left. From this, we build the twin plant, represented alongside with ambiguous states in orange. The state* (s_3, s_1) *is ambiguous and then, the transition* $((s_3, s_1), a, (s_3, s_1))$ *constitutes an ambiguous cycle. Therefore, our LTS is not diagnosable.*

3.1.2 A-Diagnosability of LMCs

The first notion of diagnosability can literally be applied to stochastic systems such as LMCs. However, probabilistic systems enable more refined definitions. We saw that for LTS, a notion of uniformity for diagnosability was not restrictive. This is not the case for probabilistic systems: if an event that enables a verdict appears with probability p at each step, for any k we have no assurance it happens in less than k steps. However, as the number of steps goes to infinity, the probability that this event occurs goes to 1. It means that a distinguishing (serie of) event(s) may allow one to have a verdict with probability 1 if we let ourselves have as much information as we desire.

In order to deal with this subtlety, [TT05] introduced A-diagnosability where A stands for asymptotic:

Definition 3.8 (A-diagnosable system)**.**
An LMC $\mathcal{A}$ is said to be A-diagnosable if $\limsup_n \mathbb{P}(F_{Amb,n}) = 0$.

This formally states the intuition above: the probability to never detect a faulty run is 0 as the length of the observation following the fault goes to infinity. Notice that for finite systems this condition is equivalent to $\mathbb{P}(F_{Amb,\infty} = 0)$: indeed, for finite systems, $F_{Amb,n+1} \subseteq F_{Amb,n}$, thus the lim sup is only a limit. This notion leads to another decision problem:

Definition 3.9 (A-diagnosability)**.**
Given an LMC $\mathcal{A}$, the A-diagnosability problem consists in determining whether $\mathcal{A}$ is A-diagnosable.

[CK13] claimed that this problem could be solved in PTIME, which was later disproved in [BHL14], clarifying the complexity of A-diagnosability.

Proposition 3.2 ([BHL14])**.** *A-diagnosability of LMC is PSPACE-complete.*

In order to prove the hardness, [BHL14] reduces a variant of language universality (*i.e.*, $L(A) = \Sigma^*$?) to A-diagnosability. They create an LMC where the faulty language is Σ^* and the safe language is $L(A)$. Then, the LMC is A-diagnosable iff there is no word u such that $u^{-1}L(A) = \Sigma^*$. Notice that in [BHL14], they prove that A-diagnosability coincides with their IF-diagnosability, *i.e.*, the diagnosability of infinite faulty runs. This rephrases the point we stated before: for finite LTS, $\limsup_n \mathbb{P}(F_{Amb,n}) = 0 \Leftrightarrow \mathbb{P}(F_{Amb,\infty}) = 0$.

Now, in order to prove that A-diagnosability is in PSPACE (see Proposition 3.4), we detail the algorithm that builds an A-diagnoser of an LMC $\mathcal{A}$. For that, we need to define

the observer $\dot{\mathcal{A}}$. The observer results from classical powerset construction on the support of $\mathcal{A}$:

Definition 3.10 (Observer)**.** *Given an LMC $\mathcal{A} = (S, \Sigma, \mu_0, p)$, we define its observer $\dot{\mathcal{A}} = (Q, \Sigma, I, T)$ as a deterministic finite state machine such that:*

- $Q \subseteq 2^{S_C}$,
- $I \subseteq Q$, *defined by* $I = \{\{s \in S_C, \mu_0(s) > 0\}\} = supp(\mu_0)$,
- $T \subseteq 2^{S_C} \times \Sigma \times 2^{S_C}$ *such that for all* $(q, a, q') \in T$, $s \in q \Rightarrow \forall s',\ p(s, a, s') > 0 \Rightarrow s' \in q'$ *and* $s' \in q' \Rightarrow \exists s \in Q,\ p(s, a, s') > 0$,
- *The automaton is trimmed: only states reachable from I are kept.*

Notice that the definition implies that the observer is deterministic, since the successor of q by a has to be maximal with respect to the condition in the definition.

Definition 3.11. *Given an LMC $\mathcal{A} = (S, \Sigma, \mu_0, p)$, its A-diagnoser $\overline{\mathcal{A}}$ is the synchronized product between $\mathcal{A}$ and its observer $\dot{\mathcal{A}}$: $\overline{\mathcal{A}} = \mathcal{A}||\dot{\mathcal{A}}$.*

Proposition 3.3. *$\overline{\mathcal{A}}$ is a well defined LMC with the same language as $\mathcal{A}$. Denoting $\tilde{p}$ the probability mapping in $\overline{\mathcal{A}}$, we have that the natural projection that associates transition $\tilde{t} = ((s, q), a, (s', q'))$ of $\overline{\mathcal{A}}$ to a transition $t = (s, a, s')$ of $\mathcal{A}$ satisfies $\tilde{p}(t) = p(t)$. This projection establishes a one-to-one correspondence between runs $\tilde{\rho}$ of $\overline{\mathcal{A}}$ and runs ρ of $\mathcal{A}$, and this correspondence preserves likelihoods: $\tilde{p}(\tilde{\rho}) = p(\rho)$. Moreover, $\tilde{\rho}$ is faulty (resp. safe) in $\overline{\mathcal{A}}$ iff ρ is faulty (resp. safe) in $\mathcal{A}$.*

Thus, the A-diagnoser is itself an LMC that accepts the same stochastic language as $\mathcal{A}$. A state (s, q) of the diagnoser is called correct (resp. faulty, ambiguous) if $q \subseteq S_C$ (resp. $q \subseteq S_F$, none of the above).

Example 3.2. *We extend example 2.5 by adding probabilities to the model and obtain the LMC pictured in figure 3.2, augmented with faulty states in red. The observer states that as long as one observes only a^k, $\mathcal{A}$ may be in s_1, but after $a^k b$ is observed, s_1 is no more possible. There is no faulty ambiguous BSCC in the A-diagnoser. We will see that this LMC is then A-diagnosable.*

Proposition 3.4 ([BHL14])**.** *An LMC $\mathcal{A}$ is A-diagnosable iff its A-diagnoser has no ambiguous (state in a) Bottom Strongly Connected Component (BSCC).*

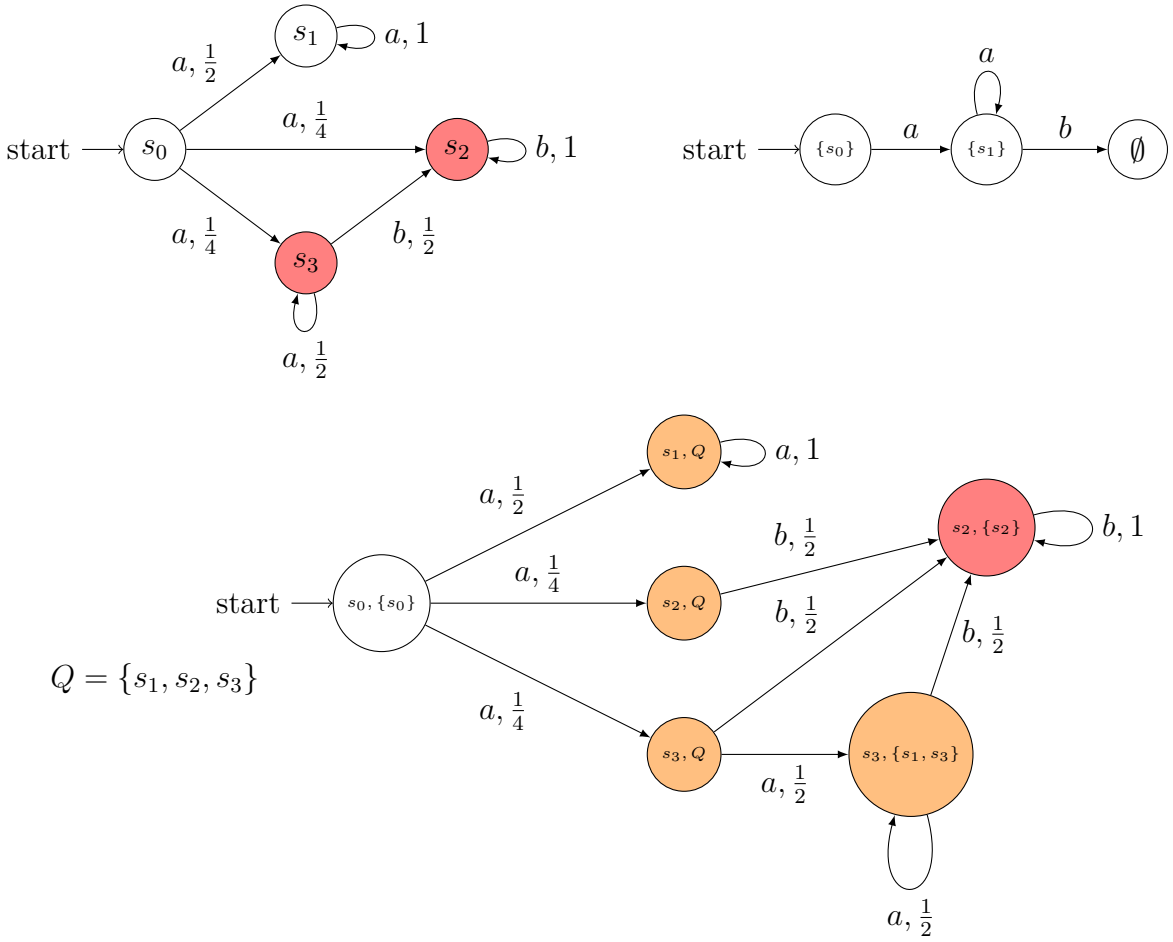

Figure 3.2: LMC $\mathcal{A}$ (above left), its observer $\dot{\mathcal{A}}$ (above right), and the A-diagnoser $\overline{\mathcal{A}}$ (below).

First, notice that all states in a BSCC will have the same status. Indeed, for all s, s' in a same BSCC there is a path from s to s' and the notions of faultiness and non-ambiguity are absorbing. Since there is probability 1 to eventually reach a BSCC, a diagnosis will be given with probability 1. This also states that the complexity of A-diagnosability is PSPACE: indeed, the search for an ambiguous state in a BSCC of the diagnoser can be made in NLOGSPACE of the size of the extended LMC $\overline{\mathcal{A}}$, which is exponential in the size of $\mathcal{A}$ (powerset based construction). Hence, a NPSPACE complexity and thanks to Savitch's theorem, we obtain the PSPACE complexity. The hardness proved in [BHL14] is proved thanks to a reduction from a variant of language universality.

3.1.3 AA-diagnosability of LMCs

In addition to A-diagnosability, [TT05] introduced the concept of AA-diagnosability. Intuitively, the former states that "with probability 1, a faulty run will eventually have an observation that reveals the occurrence of a fault" while the latter states that "with probability 1, the observation of a faulty run will have an arbitrarily small likelihood to be derive from a correct run".

Example 3.3. *This intuition is illustrated by figure 3.3. The LMC $\mathcal{A}$ is not A-diagnosable: both correct and faulty infinite language are $a(a+b)^\omega$. However, the faulty part has a far bigger chance to produce an "a" at each step, while the correct part will produce a "b" with larger probability. Therefore, a trace with many more "a" than "b" should be labeled as "faulty with high probability". Hypothesis testing based on likelihood ratio would have an arbitrary low rate of undetection as the length of the observation goes to infinity. In this example, a word having k_1+1 "a" and k_2 "b" would have a likelihood ratio to be in s_2 rather than s_1 of $(\frac{1}{2}\cdot\frac{4^{k_1}}{5^{k_1+k_2}})/(\frac{1}{2}\cdot\frac{4^{k_2}}{5^{k_1+k_2}}) = 4^{\frac{k_1}{k_2}}$.*

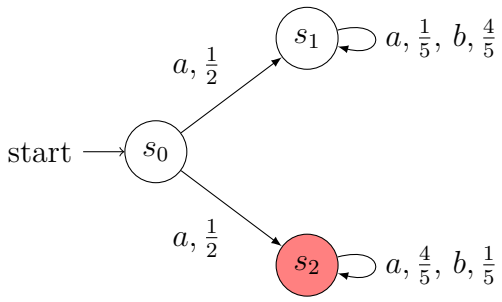

Figure 3.3: An LMC $\mathcal{A}$, faulty states in red.

More formally, we denote by F^{ε}_{amb} (resp. $F^{\varepsilon}_{amb,n}$) the set of faulty runs ρ (resp. of length n) that can be labeled as faulty with a probability of error lower than ε, that is:

$$\frac{\mathbb{P}(o^{-1}(\rho)\cap C)}{\mathbb{P}(o^{-1})(\rho)} \leq \varepsilon$$

This leads to the definition of ε-diagnosability, that we take from [BHL16a] and allows us to define precisely AA-diagnosability:

Definition 3.12 (ε-diagnosability)**.**

An LMC $\mathcal{A}$ is ε-diagnosable if for all faulty run ρ and $\alpha > 0$, there exists $n_{\rho,\alpha}$ such that for all $n \geq n_{\rho,\alpha}$:

$$\mathbb{P}(Cyl(\rho) \cap F^{\varepsilon}_{amb,n+|\rho|}) \leq \alpha$$

Intuitively, an LMC is ε-diagnosable if all faulty runs will have a low proportion ($\leq \alpha$) of futures (*i.e.*, in $Cyl(\rho)$) that have a "correctness level" bigger than ε (*i.e.*, in $F^{\varepsilon}_{amb,n+|\rho|}$).

Definition 3.13 (AA-diagnosability)**.**

An LMC $\mathcal{A}$ is AA-diagnosable if it is ε-diagnosable for all $\varepsilon > 0$.

We put in relation the different notions of diagnosis we just gave. We saw that AA-diagnosability implies ε-diagnosability for all ε. Besides, A-diagnosability can be defined as 0-diagnosability. Indeed, it is a diagnosis with 0 error with probability 1.

Proposition 3.5. *Let $\mathcal{A}$ be an LMC and $\varepsilon > 0$. We have the following:*

$$\mathcal{A} \text{ is A-diagnosable} \Rightarrow \mathcal{A} \text{ is AA-diagnosable} \Rightarrow \mathcal{A} \text{ is } \varepsilon\text{-diagnosable for some } \varepsilon$$

Similarly to A-diagnosability, the decision problem associated with AA-diagnosability was incorrectly analyzed in [CK13] and its complexity was established in [BHL16a].

Proposition 3.6. *AA-diagnosability of an LMC can be decided in PTIME.*

This was proved by reducing AA-diagnosability to the distance-1 problem for two LMCs, a problem that has been shown to be solvable in polynomial time in [CK14]. However, interestingly, while this means that AA-diagnosability of an LMC is computationally easy, deciding if it is ε-diagnosable for a given $\varepsilon > 0$ is intractable.

Proposition 3.7. *For all $\varepsilon > 0$, ε-diagnosability is undecidable.*

This result was also proved in [BHL16a].

3.1.4 Towards quantitative diagnosability analysis

The previous two sections gave a first attempt at dealing with stochastic systems. However, it was still in a qualitative way: both A-diagnosability and AA-diagnosability are logical properties, where the answer can only be yes or no. A more precise result is to try to quantify the part of the system that is non-diagnosable. Are almost all faulty runs non-diagnosable? Only a few? Few authors addressed this question, that we further develop in section 3.2. In particular, let us present the contribution in [ND08] that defines a quantification of the non-diagnosable part of the system.

Definition 3.14 (MC-diagnoser)**.**

Given an LMC $\mathcal{A}$ and its A-diagnoser $\overline{\mathcal{A}} = (S \times Q, \Sigma, \mu_0, p)$, the MC-diagnoser $\mathcal{M}_{\overline{\mathcal{A}}}$ is a Markov Chain $(S \times Q, M, \mu_0)$ with for all $s_i, s_j \in S$, $M_{i,j} = \sum_{a \in \Sigma} p(s_i, a, s_j)$.

Then, the degree of diagnosability of $\mathcal{A}$ is defined as follows: $d(\mathcal{A})$ is the ratio between the probability to reach a non-ambiguous faulty BSCC in $\mathcal{M}_{\overline{\mathcal{A}}}$ and the probability to reach a faulty BSCC in this same Markov Chain. In [ND08], this computation is made thanks to a fix point algorithm.

Example 3.4. *Let us consider the LMC $\mathcal{A}$ pictured in figure 3.4. This LMC is not A-diagnosable: the infinite word a^ω is ambiguous and has positive probability. Those are the runs that lead to an orange BSCC in the MC-diagnoser $\mathcal{M}_{\overline{\mathcal{A}}}$. However, some faulty runs can be diagnosed: if a b is observed then we know the run is faulty: those are the one that lead to a red state in $\mathcal{M}_{\overline{\mathcal{A}}}$. Runs with a b have a probability $\frac{3}{8}$ and faulty runs have a probability $\frac{1}{2}$. Thus, the proportion of diagnosable faulty runs is $\frac{3/8}{1/2} = \frac{3}{4}$.*

The second quantified property one can expect is the time to detect a fault. We can define a probability distribution over the infinite faulty runs with value in $\mathbb{R}^+$ that associates each infinite faulty run to the time needed to detect its fault. However, this distribution may be not fully computable. Thus, [ND08] investigates the mean time to reach a faulty non-ambiguous BSCC in $\mathcal{M}_{\overline{\mathcal{A}}}$ (conditionally to all runs that reach one).

Example 3.5. *Going back to figure 3.4, we look at the mean time to reach a faulty non-ambiguous BSCC. Runs going through s_2 will be diagnosed after the second observation and have probability $\frac{1}{4}$. Those going through state s_3 may reach a non-ambiguous faulty BSCC in $3+i$ steps with probability $\frac{1}{16} \cdot \frac{1}{2^i}$. Thus, the mean time to reach this BSCC is*

$$\frac{\frac{1}{4} \cdot 2 + \frac{1}{16} \cdot \left(\sum_{i \geq 0} \frac{3+i}{2^i}\right)}{\frac{1}{4} + \frac{1}{16} \cdot \sum_{i \geq 0} \frac{1}{2^i}} = \frac{8}{3}$$

3.2 Quantifying diagnosis

In this section, we investigate the problem of comparing non-diagnosable systems. In a way, we generalize the work of Nouioua and Dague in [ND08] and expand it. We define several diagnosability degrees with one of them corresponding to the degree d in [ND08]. These definitions will allow us to have more precise results on the time of diagnosis in section 3.3. We also give algorithms to optimize the computation of the degrees in subsection 3.2.3. In the following, we use the notions of observers, diagnosers, MC-diagnosers as defined in the state of the art.

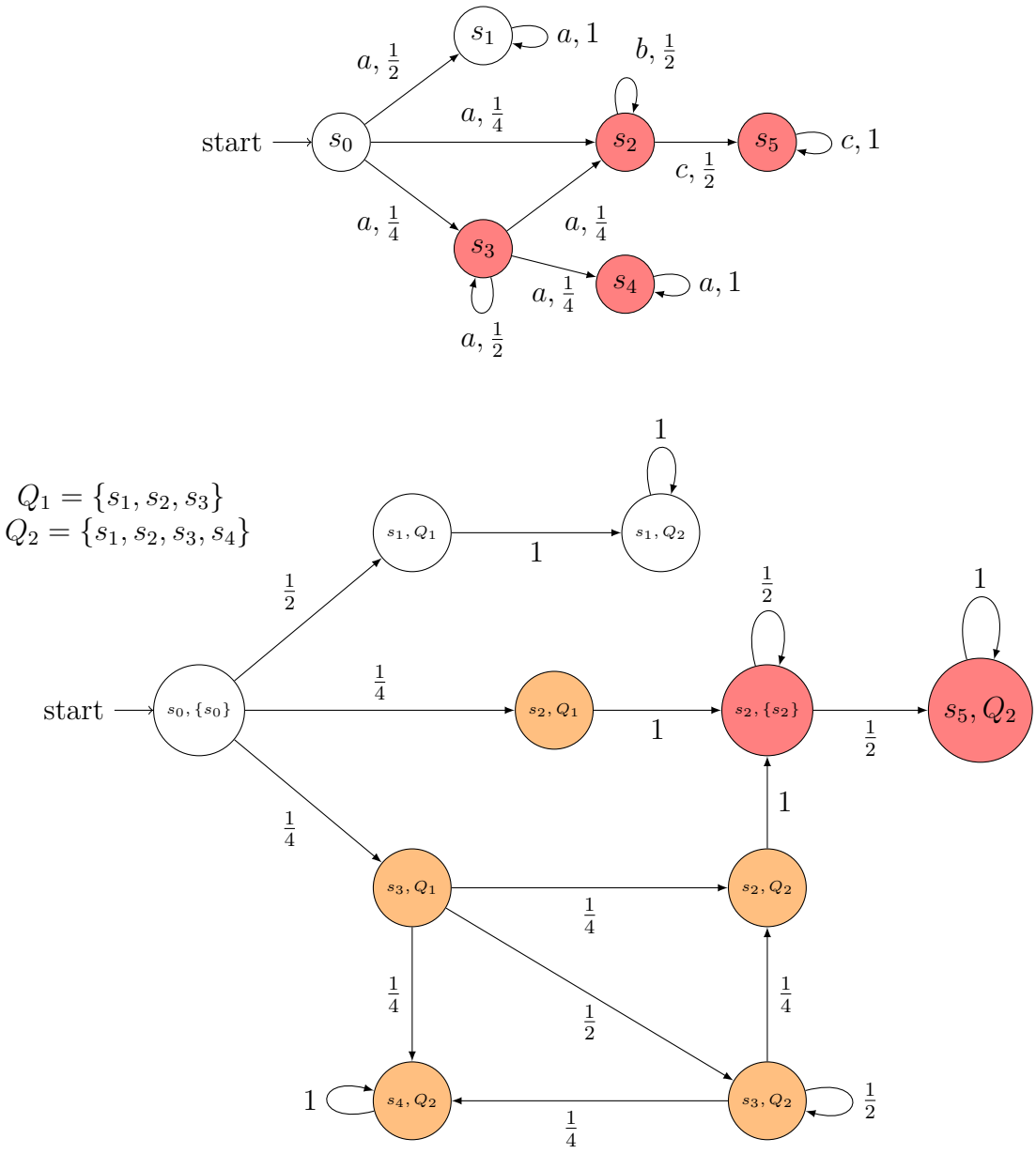

Figure 3.4: An LMC $\mathcal{A}$, faulty states in red (above) and the Markov Chain $\mathcal{M}_{\overline{\mathcal{A}}}$ associated with its diagnoser with faulty ambiguous states in orange (below).

3.2.1 Diagnosability degrees

We now examine LMCs that may not be fully A-diagnosable. We recall that the set of states is partitioned between correct states S_C and faulty states S_F, and that S_F is absorbing in our setting. Diagnosability is defined for (faulty) runs in the first place, and

then extended to systems, so it is natural to measure the proportion of problematic faulty runs, *i.e.*, those that may not lead to fault detection. Along this line, one may imagine countless notions of diagnosability degrees. For example, among the most natural ones

(a) the probability to make a fault (ie to enter into S_F) that is (k-)diagnosable, conditionally to the occurrence of a fault,
(b) or the probability that k steps after the occurrence of a fault, diagnosability holds, again conditionally to the occurrence of a fault *i.e.*, dectection will take place in the future for sure,
(c) or the probability to detect a fault k (or less) steps after it appears, still conditionally to the occurrence of a fault,
(d) or the probability to eventually detect a fault after it appears, conditionally to the occurrence of a fault.

Example 3.6. *We continue example 3.4 and use it to illustrate the different examples we just gave. Faulty runs entering at state s_2 are diagnosed in one step, but those entering at state s_3 are not diagnosable because of the loop labeled by a. So a criterion of type (a) would result in a (1-)diagnosability of degree $1/2$. However, from state s_3 one could go to state s_2 and produce the correct diagnosis in 1 step, while only paths through s_4 lead to non diagnosability. So for a criterion of type (b), 1 step after the fault diagnosability holds with degree $5/8$. Similarly, for a criterion of type (c), the detection degree after 2 steps is $5/8$, and after 3 steps it reaches $11/16$. For criterion (d), where the detection delay is not bounded, one gets a diagnosability degree of $3/4$.*

All these notions are meaningful and lead to similar developments, so for simplicity we focus on (c) and (d). We use the notation $F_{namb,\leq k} = \cup_{n\leq k} F_{namb,n}$ to denote all the faulty runs that are diagnosed in k steps or less. Then, we define the degree of k-diagnosability as follows:

Definition 3.15 (k-diagnosability degree)**.**

The k-diagnosability degree of an LMC $\mathcal{A}$ is defined as the probability to detect a fault in at most k steps after it occurs, conditionally to the occurrence of a fault :

$$\Delta_k(\mathcal{A}) = \mathbb{P}(\, F_{namb,\leq k}(\mathcal{A}) \,|\, F_\infty(\mathcal{A})) = \frac{\mathbb{P}(\, F_{namb,\leq k}(\mathcal{A}))}{\mathbb{P}(F_\infty(\mathcal{A}))}$$

And similarly, we define the diagnosability degree of a $\mathcal{A}$:

Definition 3.16 (Diagnosability degree)**.**

The diagnosability degree of an LMC $\mathcal{A}$ is defined as the probability to detect a fault after it occurs, conditionally to the occurrence of a fault :

$$\Delta(\mathcal{A}) = \mathbb{P}(\, F_{namb,\infty}(\mathcal{A}) \,|\, F_{\infty}(\mathcal{A})) = \frac{\mathbb{P}(\, F_{namb,\infty}(\mathcal{A}))}{\mathbb{P}(F_{\infty}(\mathcal{A}))}$$

Notice that the diagnosability degree is the limit of k-diagnosability when there is an arbitrary high time limit to decide, that is $\Delta(\mathcal{A}) = lim_k\, \Delta_k(\mathcal{A})$. We can also make a link with classical diagnosability:

Proposition 3.8. *There exists k, $\Delta_k(\mathcal{A}) = 1$ iff $\mathcal{A}$ is k-diagnosable.*

$\Delta(\mathcal{A}) = 1$ iff $\mathcal{A}$ is A-diagnosable.

3.2.2 Computation of diagnosability degrees

In this section, we present an evaluation algorithm for diagnosability degrees of an LMC by reducing the problem to reachability probabilities on extensions of this LMC.

The first probability we consider is the probability to produce a fault: $\mathbb{P}(\, F_{\infty}(\mathcal{A})\,)$. The set $F_{\infty}(\mathcal{A})$ corresponds to the property of reaching S_F, so: $\mathbb{P}(\, F_{\infty}(\mathcal{A})\,) = \mathbb{P}(\{\rho = t_1...t_n : s^-(t_n) \in S_C,\, s^+(t_n) \in S_F\})$

Section 2.5.1 has detailed a polynomial time algorithm to evaluate such quantities. For the other term $\mathbb{P}[\, F_{namb,\leq k}(\mathcal{A})\,]$, we show below that the probability of this set can also be characterized as a reachability probability.

Observe that, after a fault, a faulty run ρ is first ambiguous for some time and then may become "diagnosed" when fault detection takes place. We thus need to characterize the ambiguous segment following a fault, which length can range from 0 to infinity. In other words, we must characterize the time at which fault detection occurs after a fault. To this end, the first step consists in attaching a counter to faulty states. This can be performed by a simple state augmentation on $\mathcal{A}$. Equivalently, and without loss of generality, one can directly assume that faulty states of $\mathcal{A}$ are partitioned as $S_F = S_{F,0} \uplus S_{F,1} \uplus ... \uplus S_{F,k} \uplus S_{F,>k}$, and that transitions from S_C to S_F point to $S_{F,0}$, while transitions within S_F go from $S_{F,l}$ to $S_{F,l+1}$ for some $0 \leq l \leq k$ or stay within $S_{F,>k}$. If $\rho \in F(\mathcal{A})$ satisfies $s^+(\rho) \in S_{F,l}$, then ρ performed l steps after the fault. The second step consists in characterizing the moment at which a faulty run becomes diagnosed (if it does). This is most conveniently performed on the A-diagnoser $\overline{\mathcal{A}}$ presented in section 3.1.2:

Proposition 3.9. *A (finite) faulty run $\overline{\rho} \in \mathcal{R}(\overline{\mathcal{A}})$ is diagnosed in at most k steps iff it terminates in a state $(s, q) \in S_{F,k} \times Q$ with $D(q) = F$, or equivalently iff $(s, q) \in S_{F,k} \times 2^{S_F}$.*

This is a direct consequence of the structure of $\mathcal{A}$ and of the definition of an A-diagnoser $\overline{\mathcal{A}}$ of $\mathcal{A}$. Since the A-diagnoser has the same stochastic language as the original automaton, we obtain:

$$\mathbb{P}(\, F_{namb,\leq k}(\mathcal{A})\,) = \mathbb{P}(\{\overline{\rho} \in \mathcal{R}(\overline{\mathcal{A}}) :\, s^+(\overline{\rho}) \in S_{F,k} \times 2^{S_F}\})$$

Thus, this term is turned into another reaching probability, in $\overline{\mathcal{A}}$ this time. The polynomial techniques of Section 2.5.1 still apply, with the limitation that $\overline{\mathcal{A}}$ can be exponentially larger than $\mathcal{A}$, because of the observer $\dot{\mathcal{A}}$ that is present in the synchronous product.

Example 3.7. *Figure 3.5 pictures a classical example of an LMC that has an exponential diagnoser. Indeed, its observer has an exponential size. After seeing an a, the current state cannot be s_2. That lack of transition allows to introduce a "shift" that is transmitted by every b: safe runs can produce a c only $n-1$ steps after producing a b.*

To evaluate the diagnosability degree of $\mathcal{A}$, we need to compute $\mathbb{P}(\, F_{namb,\infty}(\mathcal{A}))$, *i.e.*, the probability that a fault is eventually detected. Here, the layering of S_F is not necessary, as time since the initial fault needs not to be counted. Thus, we do not need to perform a state augmentation on $\mathcal{A}$. We recall that a faulty run is diagnosed at the moment it reaches a non-ambiguous faulty state in the A-diagnoser $\overline{\mathcal{A}}$. Hence, another reachability property to compute this quantity:

$$\mathbb{P}(\, F_{namb,\infty}(\mathcal{A})\,) = \mathbb{P}(\{\overline{\rho} \in \mathcal{R}(\overline{\mathcal{A}}) :\, s^+(\overline{\rho}) \in S_F \times 2^{S_F}\})$$

Unlike [ND08], we do not use the BSCC in this definition. However, both notions of diagnosability degree are equivalent:

Proposition 3.10. *For all $\mathcal{A}$, $\Delta(\mathcal{A}) = d(\mathcal{A})$.*

Proof. $\Delta(\mathcal{A}) = \frac{\mathbb{P}(\, F_{namb,\infty}(\mathcal{A}))}{\mathbb{P}(F_\infty(\mathcal{A}))}$ and $d(\mathcal{A}) = \frac{\mathbb{P}(\text{reach an unambiguous faulty BSCC in } \mathcal{M}_{\overline{\mathcal{A}}})}{\mathbb{P}(\text{reach an faulty BSCC in } \mathcal{M}_{\overline{\mathcal{A}}})}$

A run ρ that reaches a faulty (resp. unambiguous faulty) BSCC is in $F_\infty(\mathcal{A})$ (resp. $F_{namb,\infty}(\mathcal{A})$)). Conversely, the runs in $F_\infty(\mathcal{A})$ (resp. $F_{namb,\infty}(\mathcal{A})$)) that do not reach a BSCC have a probability 0. Further, the nature of the BSCC can only be faulty (resp. unambiguous faulty). □

Example 3.8. *Continuing example 3.4, we compute the diagnosability degree. The probability to have a faulty run is the probability to reach s_2 or s_3, that is $1/2$. The probability to be faulty and diagnosed is equal to the probability to reach state $(s_2, \{s_2\})$ in the diagnoser, that is $3/8$. Then, the probability to having a fault detected conditionally to the occurrence of a fault is $\frac{3}{8}/\frac{1}{2} = 3/4$.*

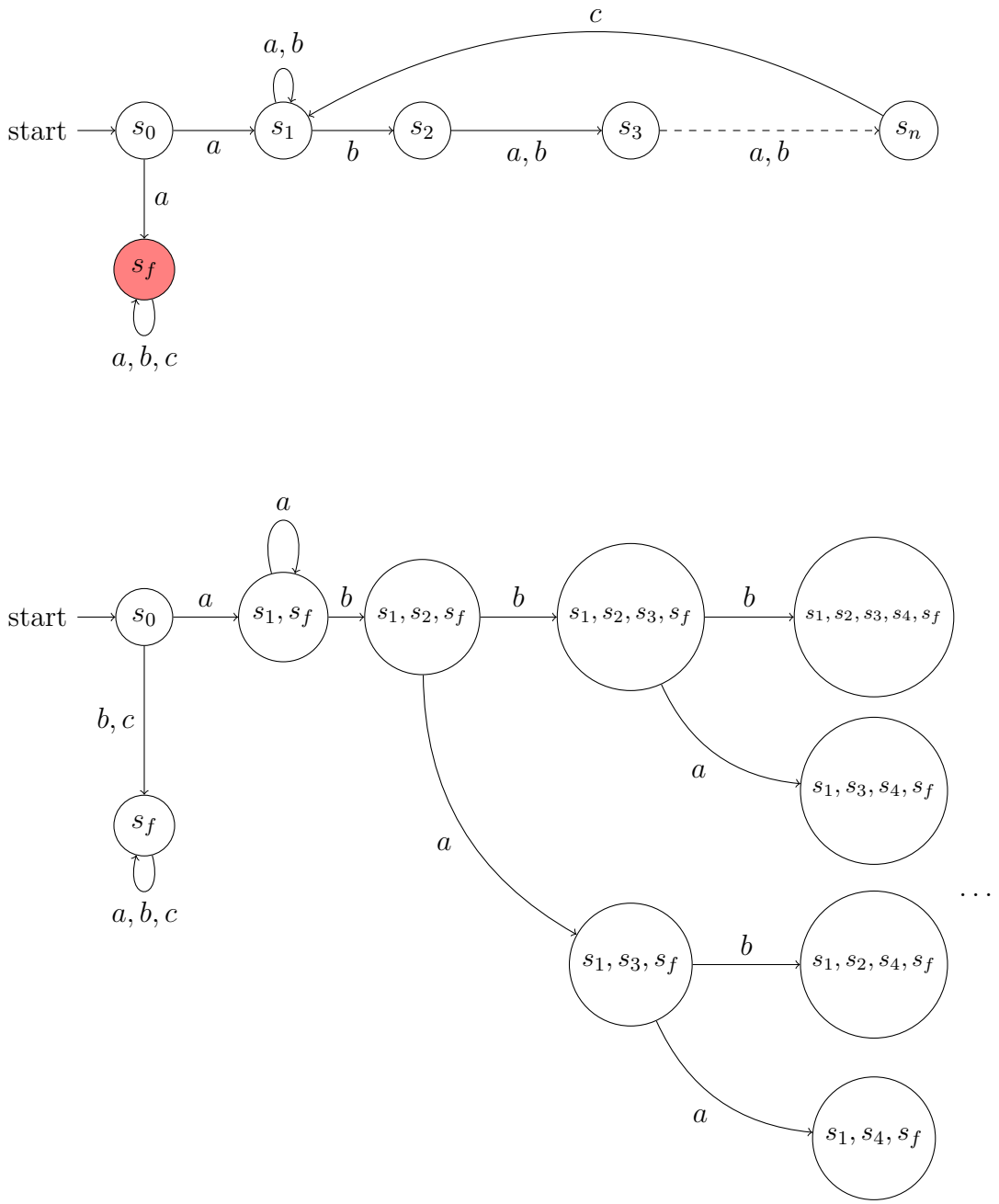

Figure 3.5: An LMC $\mathcal{A}$, faulty states in red (top) and its power set construction that has an exponential size (bot).

3.2.3 Reducing the number of states in the diagnoser

We have seen that a diagnoser $\overline{\mathcal{A}}$ may have an exponential size in the size of the LMC, with the example 3.7. However, in some cases, we may not need the whole diagnoser to compute the diagnosability degree.

General idea

Let $\mathcal{A}$ be an LMC. Our idea is the following: the purpose of building a quantified diagnoser is to attach to each state s of $\mathcal{A}$ the signal indicating whether the current state estimate q given past observations is non-faulty, faulty or ambiguous. We are mainly interested in pairs $(s, q) \in S \times Q$ where $s \in S_F$ and q is ambiguous, and further in checking whether this ambiguity will last forever with a positive probability. As $q \subseteq S$, the ambiguity of q comes from the existence of a non-faulty state $t \in q$. Using the twin plant, one can easily check whether the ambiguity due to pair $(s, t) \in S_F \times S_C$ can persist forever (and with positive probability), or will vanish in the future and not prevent fault detection. If the ambiguity due to t will for sure vanish, one needs not take it into account to compute the diagnosability degree, and may replace (s, q) by (s, q') where $q' = q \setminus \{t\}$. In doing so, one anticipates on the disappearing of an irrelevant ambiguity source due to t. In other words, one may anticipate a fault detection that will take place for sure. So the diagnosis probability does not change, but the detection delay may be shortened.

Let us now focus on the characterization of pairs of states $(s, t) \in S \times S_C$ that can be safely discarded without changing the diagnosability degree. Let $\mathcal{P}^\omega(\mathcal{A}, s)$ denote infinite paths of $\mathcal{A}$ starting from state s, and similarly $\mathcal{P}^\omega_F(\mathcal{A}, s)$, $\mathcal{P}^\omega_C(\mathcal{A}, s)$ for faulty and non-faulty paths.

Definition 3.17 (Negligible pair of type 1)**.**

Given an LMC $\mathcal{A}$, the pair $(s, t) \in S \times S_C$ is a negligible pair of type 1 *iff there is no pair of infinite runs $\rho \in \mathcal{P}^\omega_F(\mathcal{A}, s)$, $\rho' \in \mathcal{P}^\omega_C(\mathcal{A}, t)$ with $o(\rho) = o(\rho')$. We denote by NE_1 the set of negligible pairs of type 1.*

From such pairs, the ambiguity that may hold at state (s, t) or that may appear after state (s, t) will vanish for sure in the future. Notice that we do not require s to be faulty.

One can also ignore pairs of states $(s, t) \in S \times S_C$ for which any ambiguity that may hold or appear in the future will later vanish with probability 1 in $\mathcal{A}$.

Definition 3.18 (Negligible pair of type 2)**.**

Given an LMC $\mathcal{A}$, the state pair $(s,t) \in S \times S_C$ is a negligible pair of type 2 *iff* $\mathbb{P}(\rho \in \mathcal{P}_F^\omega(\mathcal{A},s) \ : \ \exists \rho' \in \mathcal{P}_C^\omega(\mathcal{A},t),\, o(\rho) = o(\rho')) = 0$. *We denote by NE_2 the set of negligible pairs of type 2.*

The above probability is computed over trajectories of LMC $\mathcal{A}$, and the involved set of runs can be shown to be measurable. We have trivially $NE_1 \subseteq NE_2$. Characterizing pairs of states in NE_2 algorithmically is clearly more difficult than for NE_1 (which only requires to consider the twin plant), as this is where the PSPACE complexity of checking A-diagnosability comes into the picture. However, easily checkable sufficient conditions can be derived that capture most of such pairs, as we will show next.

Let us define $NE(s) = \{t \mid (s,t) \in NE_2\}$. Consider now the classical quantified diagnoser $\bar{\mathcal{A}}$, and assume this machine is in state $(s,q) \in S \times 2^S$ after some observed sequence $w \in \Sigma^*$, with $t \in q \cap S_C$. Assume pair $(s,t) \in NE_2$. Then t could be safely removed from q without changing the diagnosability degree: the part of ambiguity due to pair (s,t) in (s,q) will almost surely vanish in the future (ie with probability 1). Thus, it cannot lead to an ambiguous cycle of positive likelihood.

"Removing" such negligible pairs (s,t) from $\bar{\mathcal{A}}$ can be done in several ways. Either abruptly, by replacing each state (s,q) of $\bar{\mathcal{A}}$ by pairs $(s, q \backslash NE(s))$. Or better, by recursively synchronizing $\mathcal{A}$ with a constrained state estimator, which gives a smaller stochastic automaton: let (s,q) be a state of $\bar{\mathcal{A}}'$, such that $q \cap NE(s) = \emptyset$, if (s,a,s') exists in $\mathcal{A}$, then add transition $((s,q),a,(s',q'))$ to $\bar{\mathcal{A}}'$ where $q' = \{t : \exists (s,a,t) \text{ in } \mathcal{A}\} \setminus NE(s')$. This recursive construction starts with initial state $(s_0, \{s_0\})$. The machine $\bar{\mathcal{A}}'$ obtained in that way is called the *pseudo quantified diagnoser* of $\mathcal{A}$. Notice that $\bar{\mathcal{A}}'$ is a well defined LMC, just like $\bar{\mathcal{A}}$, and that there is still a one to one correspondence between runs of $\mathcal{A}$ and runs of $\bar{\mathcal{A}}'$, which preserves likelihood.

We want to compute $\mathbb{P}(\rho \in \mathcal{P}_F^\omega(\mathcal{A}) \ : \ \nexists \rho' \in \mathcal{P}_C^\omega(\mathcal{A}),\, o(\rho) = o(\rho'))$ to get the diagnosability degree (by dividing by $\mathbb{P}(\rho \in \mathcal{P}_F^\omega(\mathcal{A}))$ which is easy to compute), and also check whether $\mathcal{A}$ is A-diagnosable (iff the degree is 1). Using the usual quantified diagnoser $\bar{\mathcal{A}}$, we have that $\mathbb{P}(\rho \in \mathcal{P}_F^\omega(\mathcal{A}) \ : \ \nexists \rho' \in \mathcal{P}_C^\omega(\mathcal{A}),\, o(\rho) = o(\rho'))$ is the probability to reach states of $\bar{\mathcal{A}}$ labeled F(aulty). We denote by B the set of Faulty states (s,q) for $\bar{\mathcal{A}}$, that is the set of states (s,q) s.t. $q \subseteq S_F$. We now show that the probability to reach B in $\bar{\mathcal{A}}$ can also be computed as the probability to reach a set of state B' in $\bar{\mathcal{A}}'$. This gives us a faster algorithm to check A-diagnosability or compute the degree of diagnosability as $\bar{\mathcal{A}}'$ is generally smaller than $\bar{\mathcal{A}}$ (sometimes up to an exponential factor as shown in the example 3.9). We set B' to be the set of states $(s,q) \in S_F \times 2^{S_F}$.

Lemma 3.11. *The probability to reach states B' in $\bar{\mathcal{A}}'$ is $\mathbb{P}(\rho \in \mathcal{P}_F^\omega(\mathcal{A}) : \nexists \rho' \in \mathcal{P}_C^\omega(\mathcal{A}), o(\rho) = o(\rho'))$.*

Proof. Given a finite path ρ of $\mathcal{A}$, it has a unique image in $\bar{\mathcal{A}}$ and a unique image in $\bar{\mathcal{A}}'$. In particular, the paths in $\bar{\mathcal{A}}$ and in $\bar{\mathcal{A}}'$ have cylinders with identical probabilities, as the probability only depends on the path of $\mathcal{A}$ and not on the labeling attached by $\bar{\mathcal{A}}$ or $\bar{\mathcal{A}}'$. Further, if ρ reaches s in $\mathcal{A}$, then it reaches some (s, q) in $\bar{\mathcal{A}}$ and some (s, q') in $\bar{\mathcal{A}}'$ with $q' \subseteq q$, by construction of $\bar{\mathcal{A}}'$.

We show that the probabilities to reach B' in $\bar{\mathcal{A}}'$ and to reach B in $\bar{\mathcal{A}}$ are actually the same.

For a run ρ of $\mathcal{A}$, denoting by (s, q) and (s, q') the states reached in $\bar{\mathcal{A}}$ and $\bar{\mathcal{A}}'$ following ρ, if $(s, q) \in B$, then $(s, q') \in B'$: As $s \in q$, we also have $s \in S_F$. Also $q' \subseteq q$ by construction, and hence for all $t \in q'$, $t \in S_F$. That is, the probability to reach B' in $\bar{\mathcal{A}}'$ is at least the probability to reach B in $\bar{\mathcal{A}}$. We show the converse now.

By definition, the probability to reach B in $\bar{\mathcal{A}}$ is equal to:

$$\sum_{\rho \in ((S \times 2^S) \setminus B)^* B} \mathbb{P}(cyl(\rho))).$$

The same holds for reaching B' in $\bar{\mathcal{A}}'$.

Let $\pi = (s_1, a_1, s_2) \cdots (s_{k-1}, a_{k-1}, s_k)$ be a (faulty) path of $\mathcal{A}$ corresponding to some path in $(S \times 2^S \setminus B')^* B'$ for $\bar{\mathcal{A}}'$. Let R be the set of paths of $\mathcal{A}$ extending π and corresponding to some path in $(S \times 2^S \setminus B)^* B$ for $\bar{\mathcal{A}}$. Hence paths in R are pairwise not prefix of one another and the probability of the union is the sum of probabilities. We show now that $\mathbb{P}(\bigcup_{\rho \in R} cyl(\rho)) = \mathbb{P}(cyl(\pi)))$, which will show that the probability to reach B in $\bar{\mathcal{A}}$ is at least as much as to reach B' in $\bar{\mathcal{A}}'$.

By contradiction, if it was not the case, there would exist an extension π' of π in $\mathcal{A}$ (hence $\mathbb{P}_{\mathcal{A}}(\pi') > 0$) such that all paths in $cyl(\pi')$ reach in $\bar{\mathcal{A}}$ states not in B.

Let (s, q) be the state of $\bar{\mathcal{A}}$ reached on π'. Now, $cyl(\pi') = \pi' \cdot \bigcup_{t \in q} \{\rho \in \mathcal{P}_F^\omega(\mathcal{A}, s) : \exists \rho' \in \mathcal{P}_C^\omega(\mathcal{A}, t), o(\rho) = o(\rho')\}$ because every run of $cyl(\pi')$ is faulty but also ambiguous (as not reaching a state of B). As $\mathbb{P}_{\mathcal{A}}(\pi') > 0$, there exists a $t \in q$ such that $\mathbb{P}[\rho \in \mathcal{P}_F^\omega(\mathcal{A}, s) : \exists \rho' \in \mathcal{P}_C^\omega(\mathcal{A}, t), o(\rho) = o(\rho')] > 0$. Let $\rho' = (t_1, a_1, t_2) \cdots (t_{n-1}, a_{n-1}, t_n)$ be a non-faulty path with $o(\rho') = o(\pi')$, $t_n = t$ and $n \geq k$. It is easy to show that for all $i \leq k$, (s_i, t_i) is not negligible of type 2. Hence the state of $\bar{\mathcal{A}}'$ reached on π is (s_k, q'_k) with q_k containing $t_k \notin S_F$. That is, (s_k, q'_k) is not in B', a contradiction.

□

Negligible pairs in the twin plant

We now explain how to compute a set $NE \subseteq NE_2$ of negligible pairs of states. We first compute the strongly connected components $C_1, \ldots, C_k$ of the twin plant $\tilde{\mathcal{A}}$ using Tarjan's algorithm, in linear time in the number of states of the twin plant. Remember that the number of states of the twin plant is quadratic at most in the number of states of $\mathcal{A}$.

We label a strongly connected component of $\tilde{\mathcal{A}}$ as ambiguous if it contains some state in $S_F \times S_C$. Notice that in this case, as faulty state remains faulty and the second component of $\tilde{\mathcal{A}}$ is in S_C, the states reachable from a state in $S_F \times S_C$ are also in $S_F \times S_C$. We recursively remove from ambiguous SCCs any loopless BSCC, because it does not characterize an ambiguous loop: they have no ambiguous infinite future.

We can then characterize the set NE_1 of negligible states of type 1 as the set of states of the twin plant which cannot reach any ambiguous SCCs. This can be done in time linear in the number of states of the twin plant, by considering first bottom strongly connected components and then inductively considering components C_i which can reach only components C_j already considered.

Lemma 3.12. *NE_1 is the set of states (s,t) of the twin plant which cannot reach a loop around some ambiguous state (x,y) with $x \in S_F, y \in S_C$.*

Proof. Let $(s,t) \notin NE_1$. Then there exists ρ an infinite faulty path from s and ρ' an infinite non-faulty path from t which are observationally equivalent. Considering the sequence of pairs of states (s_i,t_i) from $(s_0,t_0) = (s,t)$ along (ρ,ρ'). Let I be an index such that $s_I \in S_F$, which exists as ρ is faulty. As the number of pairs of states is finite and the path is infinite, there must exist two indices $j > i > I$ such that $(s_i,t_i) = (s_j,t_j)$. Denoting (ρ_1,ρ_1') and (ρ_2,ρ_2') the paths from (s_0,t_0) to (s_i,t_i) and from (s_i,t_i) to (s_j,t_j), we have a path $(s,t) \to^* (s_i,t_i) \to^* (s_j = s_i, t_j = t_i)$ with $s_i \in S_F$ as $i > I$ and ρ is faulty and $t_i \in S_C$ as ρ' is not-faulty.

The converse is trivial as if there is a path $(s,t) \to^* (x,y) \to^* (x,y)$ with $x \in S_F, y \in S_C$ in the twin plant, then there is also an infinite faulty path ρ from s and an infinite non-faulty path ρ' from t which are observationally equivalent. □

We are now ready to define a set NE with $NE_1 \subseteq NE \subseteq NE_2$. It will contain only pairs $(s,t) \in S \times S_C$ such that $\mathbb{P}(\rho \mid s^-(\rho) = (s,t) \wedge \rho = \rho_1\rho_2, \rho_2 \in (S_F \times S_C)^\omega) = 0$. $\mathbb{P}_{s,t}(\rho \mid s^-(\rho) \in S_F \times S_C) = 0$ for $s^-(\rho)$ the set of pairs of states seen infinitely often along ρ. To define NE, we define inductively a sequence $P_1 \subsetneq \ldots \subsetneq P_\ell$ of sets of states of the twin plant $\tilde{\mathcal{A}}$ that cannot be used to give a positive probability to stay ambiguous forever. Then, NE will be defined as the set of states that cannot reach an ambiguous

cycle avoiding P_ℓ. This can be computed in linear time in the size of $\tilde{\mathcal{A}}$ by using Tarjan's algorithm. It suffices to remove states of P_ℓ and to look for SCCs with self loops.

We now define P_i inductively as follows:

$$\begin{aligned} P_1 =& NE_1 \\ P_{i+1} =& P_i \cup \{(s,t) | \exists a\, a \text{ is fireable from } s \text{ and } (s,t) \rightarrow^a (s',t') \Rightarrow (s',t') \in P_i\} \end{aligned}$$

When $P_\ell = P_{\ell+1}$, which must happen after a number of steps bounded by the number of states in $\tilde{\mathcal{A}}$, we stop the process. That is, P_ℓ is a smallest fix point of $\phi(P_i) = P_{i+1}$ that can be obtained in polynomial time. We have:

Lemma 3.13. *From every state $(s,t) \in P_\ell$ with $s \in S_F$, there exists a path ρ, $s^-(\rho) = s$ such that for every ρ' such that $s^-(\rho') = t$ and $o(\rho) = o(\rho')$, one has that ρ' is faulty.*

We can now define formally NE as the set of states that cannot reach an ambiguous cycle avoiding P_ℓ, that is $NE = (S \times S_C) \setminus \{(s,t) \mid \exists \rho = \rho_1\rho_2, s^-(\rho) = (s,t) \wedge \rho_2 \text{ avoids } P_\ell \wedge s^-(\rho_2) = s^+(\rho_2)\}$. Using lemma 3.13, we obtain:

Lemma 3.14. $NE_1 \subseteq NE \subseteq NE_2$.

Proof. Let $(s,t) \in NE_1$ be a pair of type 1. It cannot reach an ambiguous loop, thus in particular it cannot reach an ambiguous loop avoiding P_ℓ.

Similarly, let (s,t) be a pair in NE, *i.e.*, such that (s,t) cannot reach an ambiguous cycle avoiding P_ℓ. Thanks to lemma 3.13, we know that the probability that for infinite paths ρ, ρ', $s^-(\rho) = s', s^-(\rho') = t'$ and ρ, ρ' are ambiguous is 0 since they will always have an occasion to have a future that disambiguates them (*i.e.*, the probability to avoid in P_ℓ is 0).

Thus, $\mathbb{P}[\rho \in \mathcal{P}_F^\omega(\mathcal{A}, s) \,:\, \exists \rho' \in \mathcal{P}_C^\omega(\mathcal{A}, t),\, o(\rho) = o(\rho')] = 0$ and $NE \subseteq NE_2$. □

We can use this lemma to reduce the size of a pseudo-quantitative-diagnoser:

Theorem 3.15. *From an LMC $\mathcal{A}$, one can build in quadratic time a pseudo-quantitative-diagnoser $\bar{\mathcal{A}}'$ such that the probability of an infinite faulty ambiguous run in $\mathcal{A}$ is equal to the probability to reach an ambiguous SCC in $\bar{\mathcal{A}}'$. Further, there exists an LMC $\mathcal{A}$ such that the size of $\bar{\mathcal{A}}'$ is exponentially smaller than that of the quantitative diagnoser built in Section 3.2.*

Proof. The set NE is computable in quadratic time w.r.t to the number of transitions of the original automaton $\mathcal{A}$. We then define the pseudo-diagnoser $\bar{\mathcal{A}}' = (S \times Q, \Sigma, (s_0, \{s_0\}), T')$ with $T' = \{((s,q), a, (s', q' \backslash \{t \mid (s,t) \in NE\}))\}$ such that $(s,a,s') \in T$ and $q' = \{t' \mid \exists t \in q(t,a,t') \in T\}$. Since $NE \subseteq NE_2$ (Lemma 3.13), we obtain that the probability of a faulty ambiguous run in $\mathcal{A}$ is equal to the probability to reach an ambiguous SCC in $\bar{\mathcal{A}}'$.

The twin plant has a number of states quadratic in the size of the original automaton. Besides, determining the sets P_ℓ and NE can be done in a time quadratic in the size of the twin plant, hence the biquadratic complexity.

□

An example with an exponential reduction

Example 3.9. *Continuing example 3.5, figure 3.6 presents an example where the pseudo diagnoser $\bar{\mathcal{A}}'$ is exponentially smaller than the natural diagnoser based on the determinized of $\bar{\mathcal{A}}$.*

Indeed, the number of states of the natural diagnoser $\bar{\mathcal{A}}_3$ is $O(2^n)$, as safe runs can produce a c only $n-1$ steps after producing a b. That is, the diagnoser needs to distinguish between 2^{n-1} cases, depending on the last $n-1$ letters in $\{a,b\}$.

Using the twin plant, the number of states of the pseudo-quantitative-diagnoser is dramatically smaller. First, NE_1 $(= P_1)$ is the set $\{(s_i, s_j) | i > 0, j > 0\}$. Then, $P_2 = P_1 \cup \{(s_f, s_i) | i \leq n\}$. Indeed, for all $i < n$, there is a transition (s_f, c, s_f) but there is no transition starting in s_i labeled by c and then no successor to (s_f, s_i) by c. Thus, for all transition $(s_f, s_i) \rightarrow^c (s_f, s'')$, we have $(s_f, s'') \in P_1$, because there is no such transition $(s_f, s_i) \rightarrow^c (s_f, s'')$. Thus $(s_f, s_i) \in P_2$ for all $i < n$. Similarly, we obtain that $(s_f, s_n) \in P_2$ since transition (s_f, a, s_f) can occur and there is no transition labeled by a from s_n.

Now, state (s_0, s_0) only has successors in P_2. Thus $(s_0, s_0) \in P_3$. That is, P_3 is made of all the states of the twin plant and since there is no ambiguous cycle outside P_3, NE contains all the states of the twin plant. Hence, the pseudo-diagnoser is very simple: for all s, $NE(s) = S$ and then every state in the pseudo-diagnoser is in the form $(s, \emptyset)$ with $s \in S$. That is, the pseudo-diagnoser is isomorphic to the original LMC.Therefore, this transformation avoids the exponential blow-up required by using an exact diagnoser.

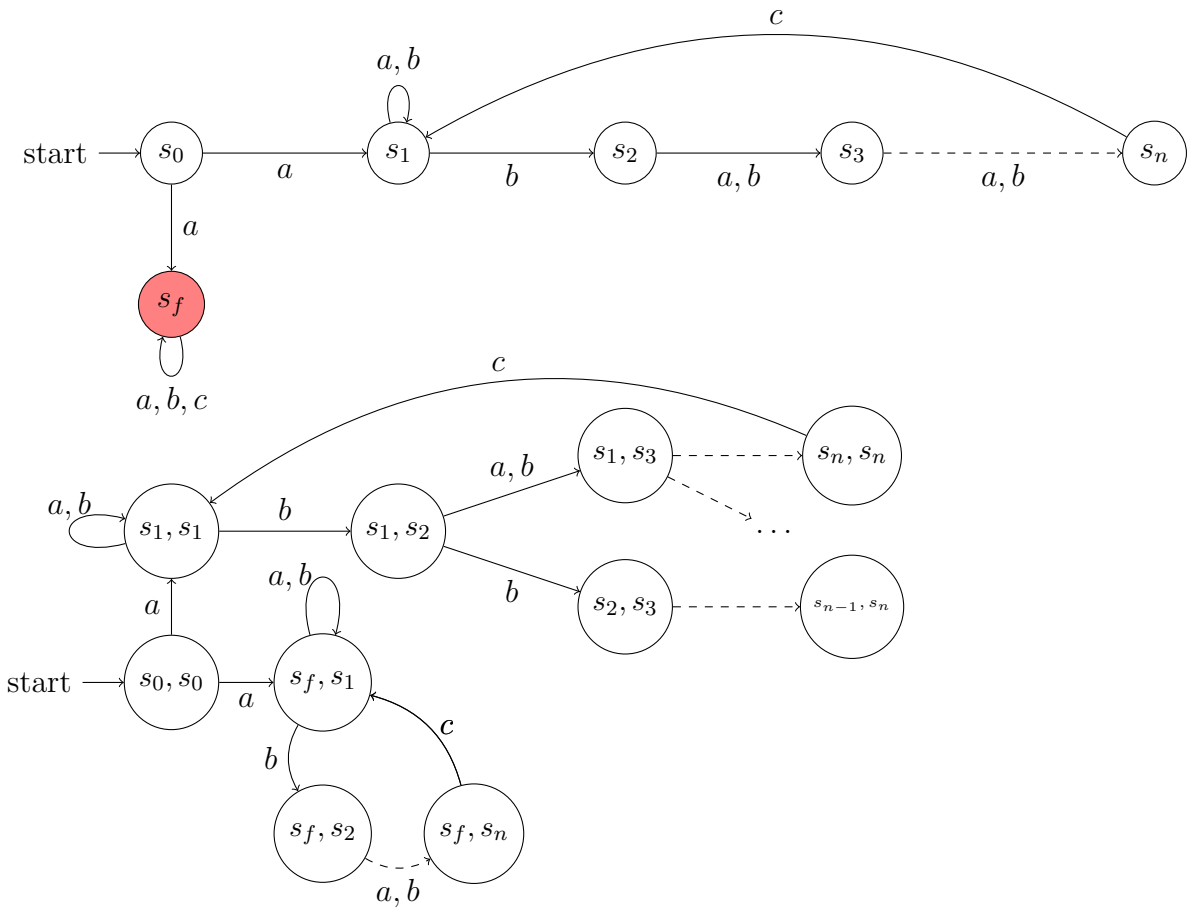

Figure 3.6: An example of an LMC $\mathcal{A}_3$ (top) that has an exponential sized diagnoser, and its twin plant (bottom)

3.3 Distributions of fault detection delay

We are now interested in the evaluation of the detection delay (conditionally to the occurrence of a detection). Generally, the number of observations before detection can be arbitrarily long. However, the probability that the diagnosis occurs only after k steps goes to 0 as k increases. Hence, we are interested in two things: approximating the probability distribution associated with the detection delay and giving bounds on the probability of detection after a certain number of steps. For the first objective, we will present a sequence of distributions that converge to the real one thanks to the computation of the moments of this distribution. For the second objective, we will also use these moments and concentration inequalities to obtain the desired bounds.

3.3.1 Semirings for moments

In this section, we present how semirings will allow us to formally compute the moments of the distribution thanks to the Floyd-Warshall algorithm. In order to compute moments of response time, we first fix a state s_f (symbolizing a fault has just occurred) and a set of states FD (symbolizing a fault has just been diagnosed). We introduce a set of semirings that will allow us to compute the n-th moment of detection delay to the fault from state s_f, for all $n \in \mathbb{N}$. We will compute the moment inductively on a disjoint subset Π of paths of $\mathcal{A}$ from s_f to FD. For an integer n, we denote $\mu_n(\Pi) = \sum_{\pi \in \Pi} \mathbb{P}(\pi)|\pi|^n$.

Notice that for all π_1, π_2 paths in $\mathcal{P}(s_f, FD)$ π_1 is not a prefix of π_2.Thus, we have that $\mathcal{P}(s_f, FD)$ is the moment of order n of the distribution of detection delay to the fault associated with state s_f.

We now give some properties of μ. Let Π_1 be a set of paths ending in some state s and let Π_2 be a set of paths starting from s. We denote by $\Pi_1 \cdot \Pi_2$ the set of paths $\rho_1\rho_2$ with $\rho_1 \in \Pi_1$ and $\rho_2 \in \Pi_2$.

Proposition 3.16. *For all n, we have $\mu_n(\Pi_1 \cdot \Pi_2) = \sum_{i=0}^{n} \binom{n}{i} \mu_i(\Pi_1) \cdot \mu_{n-i}(\Pi_2)$*

Proof.

$$
\begin{aligned}
\mu_n(\Pi_1.\Pi_2) &= \sum_{\pi_1 \in \Pi_1} \sum_{\pi_2 \in \Pi_2} \mathbb{P}(\pi_1\pi_2)|\pi_1\pi_2|^n \\
&= \sum_{\pi_1 \in \Pi_1} \sum_{\pi_2 \in \Pi_2} \mathbb{P}(\pi_1)\mathbb{P}(\pi_2) \sum_{i=0}^{n} \binom{n}{i} |\pi_1|^i |\pi_2|^{n-i} \\
&= \sum_{i=0}^{n} \binom{n}{i} \sum_{\pi_1 \in \Pi_1} \mathbb{P}(\pi_1)|\pi_1|^i \sum_{\pi_2 \in \Pi_2} \mathbb{P}(\pi_2)|\pi_2|^{n-i} \\
&= \sum_{i=0}^{n} \binom{n}{i} \mu_i(\Pi_1) \cdot \mu_{n-i}(\Pi_2)
\end{aligned}
$$

□

This property hints to a set of semirings $(\mathbb{R}^{n+1}, \oplus_n, \otimes_n, \overline{0}_n, \overline{1}_n)$ with good properties to compute moments. For $(n+1)$-uplets $(x_0, \ldots, x_n)$ and $(y_0, \ldots, y_n)$, we define operations $\oplus_n$ and $\otimes_n$:

- $(x_0, \ldots, x_n) \oplus_n (y_0, \ldots, y_n) = (x_0 + y_0, \ldots, x_n + y_n)$
- $(x_0, \ldots, x_n) \otimes_n (y_0, \ldots, y_n) = (z_0, \ldots, z_n)$ with $z_i = \sum_{j=0}^{i} \binom{i}{j} x_j y_{i-j}$

The neutral element for $\oplus_n$ is $\overline{0}_n = (0, \ldots, 0)$. $\overline{0}_n$ is annihilating for $\otimes_n$. The neutral element for $\otimes_n$ is $\overline{1} = (1, 0, \ldots, 0)$. In the following, we will denote the different laws and elements by $\oplus$, $\otimes$, $\overline{0}$ and $\overline{1}$.

Proposition 3.17. *For $n \geq 0$, $(R_+^{n+1}, \oplus, \otimes, \overline{0}, \overline{1})$ defines a commutative semiring.*

Proof. It is clear that $(R_+^{n+1}, \oplus, \overline{0})$ is a commutative monoid. Associativity and commutativity in $(R_+^{n+1}, \otimes, \overline{1})$ come from the symmetric role of the x_i and y_i in $\otimes$. Thus, we have to prove that $\otimes$ is distributive over $\oplus$. Since $\otimes$ is commutative, we only have to prove that for all $x, y, z \in R_+^{n+1}$, $x \otimes y \oplus x \otimes z = x \otimes (y \oplus z)$. For $i \geq 0$, we check the i-th component:

$$\begin{aligned}
((x \oplus y) \otimes z)_i &= \sum_{j=0}^{i} \binom{i}{j} (x \oplus y)_j \cdot z_{j-i} \\
&= \sum_{j=0}^{i} \binom{i}{j} (x_j + y_j) \cdot z_{j-i} \\
&= \sum_{j=0}^{i} \binom{i}{j} [x_j \cdot z_{j-i} + y_j \cdot z_{j-i}] \\
&= (x \otimes z) \oplus (y \otimes z)
\end{aligned}$$

□

Then, we obtain the property that will allow us to use the Floyd-Warshall algorithm in order to compute the moments of the distribution of detection delay:

Proposition 3.18. *If for all $i \leq n$, we have $x_i = \mu_i(\Pi_1)$ and $y_i = \mu_i(\Pi_2)$, denoting $(z_0, \ldots, z_n) = (x_0, \ldots, x_n) \otimes_n (y_0, \ldots, y_n)$, we get $\mu_i(\Pi_1 \cdot \Pi_2) = z_i$. Further, if both Π_1, Π_2 are disjoint, and if no path of Π_1 (resp. Π_2) is a prefix of a path of Π_2 (resp. Π_1), then $\mu_i(\Pi_1 \cup \Pi_2) = x_i + y_i$.*

The proof is straightforward, since we chose the operators to match exactly what we wanted to compute the moments. Thus, we will be able to use the Floyd-Warshall algorithm to compute the moments of the distribution. Before that, we need to define the initial weights of the transitions.

Definition 3.19. *[Weighted diagnoser]*

Given an LMC $\mathcal{A} = (S, \Sigma, \mu_0, p)$ and its A-diagnoser $\overline{\mathcal{A}} = (S', \Sigma, \mu_0', p')$, its weighted diagnoser $\overline{\mathcal{A}_w}$ is a quadruplet $(S', \Sigma, \gamma_0, \gamma)$ with:

- $\gamma_0 : S' \to \mathbb{R}^{n+1}$ *with* $\gamma_0(s') = (\mu_0(s'), 0, \ldots, 0)$,

- $\gamma : S' \times \Sigma \times S' \to \mathbb{R}^{n+1}$ *with*

$$\gamma(s', a, s'') = \begin{cases} (p'(s', a, s''), 0, \ldots, 0) \text{ if } s' \in S_C \\ (p'(s', a, s''), \ldots, p'(s', a, s'')) \text{ else} \end{cases}$$

Intuitively, if the initial state of the transition is in S_C, then no delay is added to the detection, hence the 0. On the contrary, if the initial state of a transition t is in S_F, then a delay is added. The transition t has a length of 1, so for any n, $\mu_n(\{t\}) = p'(t)$.

Let $\mathcal{A}$ be an LMC and $\overline{\mathcal{A}_w}$ its weighted diagnoser over a partition over faulty and safe states. For every faulty state s_f of the weighted diagnoser, we denote $\mathbb{P}(s_f)$ the probability that it is the first faulty state reached by an execution. We denote S_{FD} the set of faulty states where the diagnosis holds.

Theorem 3.19.

The n-th moment of the distribution of detection delay conditionally to this detection is given by:

$$\frac{\sum_{s_f} \mathbb{P}(s_f) W(\mathcal{P}_{S_{FD}}(s_f, S_{FD}))_n}{\sum_{s_f} \mathbb{P}(s_f) W(\mathcal{P}_{S_{FD}}(s_f, S_{FD}))_0}$$

Proof. The denominator comes from the fact we ask for the distribution conditionally to the fact that the diagnosis holds. The proof is straightforward, since the algorithm is proven to compute the quantity we are interested in. The probability of the set of paths is the first component of the weight, and the n-th moment is μ_n divided by the probability to be diagnosed. Notice that this quantity is equal to (assuming without loss of generality there is only one initial state s_0):

$$\frac{W(\mathcal{P}_{S_{FD}}(s_0, S_{FD}))_n}{W(\mathcal{P}_{S_{FD}}(s_0, S_{FD}))_0}$$

□

Theorem 3.20. *Let $\mathcal{A} = (S, \Sigma, \mu_0, p)$ be an LMC and $\overline{\mathcal{A}_w}$ its weighted diagnoser with S' as set of states. One can compute the n first moments of the diagnosability degree of $\mathcal{A}$ in time $O(n^2 \times |S'|^3)$.*

Proof. Since we use the Floyd-Warshall algorithm to perform this calculation, the complexity is cubic in the number of states. Notice that the number of states of the weighted diagnoser can be exponential in the size of the original LMC. Furthermore, its complexity

is quadratic in the number of moments we want to know: in the semiring computation, the calculation of the n-th moments is performed through a sum on all the previous moments, that gives us the well known complexity $T(n) = \sum_{i<n} T(i)$, hence quadratic. □

Algorithms for moments had been proposed in the performance evaluation community. Methods used there are mostly numerical [Bra+06; Tar05]. These methods are efficient but not robust to changes: every value is set and computations do not use parameters. On the contrary, ours may be slower, but they have the same computational complexity and allows one to have parameters and formal calculus.

In this work, we presented to the computation of moments applied to quantified diagnosis, but it has many more applications. In particular, we presented these techniques in [BFG18b] for the notion of response time, that is the delay between a query and the moment it is answered. In the framework of diagnosis, the query is the occurrence of the fault and the answer is obtained when the diagnosis holds. These techniques also have an interest for computational biology. In [BBW16; Bog+15; Gon+13], complex functions describing the evolution of molecular species are approximated using the first k moments, for some k.

Observe that we set the time of a transition to one unit of time, but this is not a restriction: indeed, transitions t could have arbitrary lengths, or even length as a random variable X_t. The algorithm would still work, we just need to give the accurate initial values: $\mu_i(\{X_t\}) = \mathbb{P}(t)\mathbb{E}[X_t^i]$. In particular, this allows us to encompass detection delays on systems where time is not given as units, such as labeled systems where the support is a Continuous Time Markov Chain (CTMC):

Definition 3.20. *A CTMC is a tuple $(S, M, \mu_0, (\tau_s)_{s\in S})$ with:*

- *(S, M, μ_0) is a Markov Chain,*
- *for all s, τ_s is the sojourn parameter associated with state s. That is, the PDF function of the sojourn time is $X_s(t) = \tau_s e^{-\tau_s \cdot t}$ and the probability to stay in s at least t units of time is $e^{-\tau_s \cdot t}$.*

In this continuous context, we need integrals instead of sums to define the i-th moment of a variable X: $\mu_i(X) = \int_0^\infty X(t)t^i dt = 1$. For every state $s \in S$, let $X_s(t) = \tau_s e^{-\tau_s \cdot t}$. For all i, for all s, $\mu_i(X_s)$ is well defined and $\mu_i(X_s) = \frac{i!}{\tau_s^i}$

We can easily extend the computation of moments for CTMCs. The inductive formulas for probabilities and moments of the reaching time distribution remain unchanged. We only need to change the definition of moments for every transition, which is input at the

initialization phase of the Floyd-Warshall algorithm: for all $s, t \in S$, we set $W_n(s,t)_0 = M(s,t)$ and $W(s,t)_i = M(s,t)\frac{i!}{\tau_s^i}$ for all $i \in [1,n]$.

Theorem 3.21. *Let $\mathcal{A} = (S, M, \mu_0, (\tau_s)_{s\in S})$ be a CTMC. One can compute $\mu_i(s,t)$ for all $i \leq n$ and $s,t \in S$ in time $O(n^2 \times |S|^3)$.*

3.3.2 Approximating the distribution from its moments

It is known [TH07] that phase-type distributions of order n are determined by their first $2n-1$ moments. First passage distribution time in Labeled Markov chains with n states are phase type distribution of order n. However, [TH07] does not help characterizing bounds as it does not ensure that a non-phase type distribution cannot have the exact same moments as a phase type distribution, unlike our result.

In this section, we discuss how the calculation of moments is sufficient to approximate the distribution of detection delay. For that, we first prove that given a sequence of moments $(\mu_n)_{n\in\mathbb{N}}$ the distribution is unique for the case of detection delay. Secondly, we discuss how to define a sequence of distributions converging to the real one.

The first point is an instance of the moment problem.

Definition 3.21 (Moment problem)**.** *Given a sequence of numbers μ_n, does there exist a random variable that has for n-th moment μ_n and is this random variable unique?*

The special case we are investigating, that is a random variable with values in $\mathbb{R}^+$ is called the Stieljes moment problem.

As a start, we pinpoint that in general, there may be several distributions that correspond to a given sequence of moments $(\mu_n)_{n\in\mathbb{N}}$. This would compromise approximating the distribution using moments, as there would not be a unique such distribution.

Example 3.10. *Let us consider a distribution δ on $\mathbb{R}^+$. If δ has the sequence of moments $\{\mu_n = n! \mid n \in \mathbb{N}\}$, then δ is the exponential distribution with parameter 1. Similarly, the sequence of moments $\{\mu_n = (2n)! \mid n \in \mathbb{N}\}$ for a distribution on $\mathbb{R}^+$ is characteristic of the square of the exponential distribution of parameter 1.*

Now, consider the cube of the exponential distribution of parameter 1. Its sequence of moments is $\{\mu_n = (3n!) \mid n \in \mathbb{N}\}$. However, there exist an infinite number of distributions with this sequence of moments [Sto06].

We now prove answer positively to the Stieljes moment problem for the case of the distribution of detection delay, that is its sequence of moments respects the Carleman's

condition from year 1922, that guarantees the uniqueness of the distribution. The condition is that $\sum_{n\in\mathbb{N}} \mu_n(\delta)^{-\frac{1}{2n}} = \infty$.

Theorem 3.22. *Let $\mathcal{A}$ be an LMC. For all $n \in \mathbb{N}$, let μ_n be the moment of order n of the detection delay of $\mathcal{A}$. Then there exists a unique distribution δ such that $\mu_n(\delta) = \mu_n$ for all $n \in \mathbb{N}$.*

Proof. The existence is given by the construction of our numbers μ_n. The difficult part is the unicity.

Let m be the number of states of $\mathcal{A}$, p be the minimal probability to detect a fault without taking any loop, and ℓ the probability of the highest-probability loop that can be part of a path to this detection. We denote by δ the distribution of detection delay.

For $i \leq m$, $P(\delta = i) < 1$. For $i > m$, a successful path has to take at least $\frac{i}{m}$ loops, then $p(i) \leq \ell^{\frac{i}{m}} p$. Thus, we have $\mu_n(\delta) \leq \sum_{i=0}^{m} \mathbb{P}(\delta = i) i^n + \Sigma_{i=m+1}^{\infty} \ell^{\frac{i}{m}} p i^n$.

Thus, $\mu_n(\delta) \leq \sum_{i=0}^{m} i^n + \Sigma_{i=m+1}^{\infty} \ell^{\frac{i}{m}} i^n$.

The first part is lesser than $(m+1)m^n$. We now need to bound the second part.

$$\begin{aligned}
\sum_{i=m+1}^{\infty} \ell^{\frac{i}{m}} i^n &\leq \Sigma_{i=1}^{\infty} \ell^{\frac{i}{m}} i^n \\
&\leq \frac{1}{(1-\ell^{\frac{1}{m}})^{n+1}} \sum_{i=0}^{n} E(n,i) (\ell^{\frac{1}{m}})^{n-i} && (1)\\
&\leq \frac{1}{(1-\ell^{\frac{1}{m}})^{n+1}} \sum_{i=0}^{n} E(n,i) && (2)\\
&\leq \frac{n!}{(1-\ell^{\frac{1}{m}})^{n+1}} && (3)
\end{aligned}$$

(1): $E(n,i)$ is the eulerian number of parameter n, i. We obtain this line because $\Sigma_{i=1}^{\infty} \ell^{\frac{i}{m}} i^n$ is the polylogarithm function $Li_{-n}(\ell^{\frac{1}{m}})$.

(2): $\forall i$, $(\ell^{\frac{1}{m}})^{n-i} \leq 1$

(3): $\sum_{i=0}^{n} E(n,i) = n!$

We want to find a lower bound to the $2n$-th root of $\mu_n(\delta)$, in order to prove the moments verify the Carleman's condition.

$$\begin{aligned}\mu_n(\delta)^{\frac{1}{2n}} &\leq ((m+1)m^n + \frac{n!}{(1-\ell^{\frac{1}{m}})^{n+1}})^{\frac{1}{2n}} \\ &\leq ((m+1)m^n)^{\frac{1}{2n}} + (\frac{n!}{(1-\ell^{\frac{1}{m}})^{n+1}})^{\frac{1}{2n}} \qquad (4) \\ &\leq (m+1)^{\frac{1}{2n}} \cdot \sqrt{m} + \frac{n!^{\frac{1}{2n}}}{(1-\ell^{\frac{1}{m}})^{\frac{n+1}{2n}}}\end{aligned}$$

(4): $\sqrt[2n]{x+y} \leq \sqrt[2n]{x} + \sqrt[2n]{y}$.

Then, we have that $\sum_{n\in\mathbb{N}} \mu_n(\delta)^{-\frac{1}{2n}} \geq \sum_{n\in\mathbb{N}} \frac{1}{(m+1)^{\frac{1}{2n}}\cdot\sqrt{m} + \frac{n!^{\frac{1}{2n}}}{(1-\ell^{\frac{1}{m}})^{\frac{n+1}{2n}}}}$

Thanks to the Stirling equivalent for the factorial $n! \approx \sqrt{2\pi \cdot n}(\frac{n}{e})^n$, we find that an equivalent of the denominator for large n is $\alpha \cdot n^{\frac{n+1}{2n}}$ with α some real number. Since the sum of the inverses of $n^{\frac{n+1}{2n}}$ diverges, we deduce that

$$\sum_{n\in\mathbb{N}} \mu_n(\delta)^{-\frac{1}{2n}} = \infty$$

Due to Carleman's condition, the distribution δ corresponding to the moments $(\mu_n(\delta))_{n\in\mathbb{N}}$ is thus unique. □

Here, we presented the proof for distribution over detection delays for an LMC. This proof also holds for response times on different models, such as CTMC. However, in this setting it would get very technical and hard to read: in the sums, i^n is replaced by $\mathbb{E}[X_i^n]$ where X_i is the random variable associated with i successive sojourn time. Then, an upper bound is found for $\mathbb{E}[X_i^n]$ and the rest of the proof is similar.

Since we have unicity of the distribution corresponding to the sequence of moments of the distribution of detection delay of a probabilistic automaton, we obtain the following convergence in law:

Proposition 3.23. *[PR69] Let δ be the distribution of detection delays of an LMC. Let $(\delta_i)_{i\in\mathbb{N}}$ be a sequence of distributions on $\mathbb{R}^+$ such that for all n, $\lim_{i\to\infty} \mu_n(\delta_i) = \mu_n(\delta)$. Then, if C_i is the cumulative distribution function of δ_i and C the cumulative distribution function of δ, then for all x $\lim_{i\to\infty} C_i(x) = C(x)$.*

Thus, C can be approximated by taking a sequence $(\delta_n)_{n\in\mathbb{N}}$ of distribution such that for all $i \leq n$, $\mu_i(\delta_n) = \mu_i(\delta)$. A reasonable choice for δ_n is to consider the distribution of maximal entropy corresponding to the moments $\mu_1, \ldots, \mu_n$, as presented in [CT12].

The distribution of maximal entropy can be understood as the distribution that assume the least information. It can be approximated as close as desired, for instance $\frac{1}{n}$ close to the distribution of maximal entropy having moments $(\mu_1(\delta), \ldots, \mu_n(\delta))$. Applying Proposition 3.23, we thus obtain that the cumulative distribution function associated with δ_i converges towards the cumulative distribution function associated with δ.

3.3.3 Bounds on the detection delay

We now explain how to use moments in order to obtain optimal bounds on the detection delay. First, notice that as soon as there exists a loop between a fault and its detection, then there will be runs with arbitrarily long detection delay, although there might be probability 1 to eventually answer every query. We thus turn to a more quantitative evaluation of the detection delay.

Let $0 < p < 1$. We are interested in a bound T on the delay between a fault and the detection such that a proportion greater than $1 - p$ of the faults is diagnosed before this bound. For a distribution $\delta : \mathbb{R}^+ \to \mathbb{R}^+$ of detection delays, we denote by $B(\delta, p)$ the lowest T such that the probability to have a detection delay above T is lower than p. Equivalently, we look for the highest T such that the probability of a detection delay above T is at least p.

Markov bounds associated with one moment

Let $i \in \mathbb{N}$ and $\mu_i > 0$. We let Δ_{i,μ_i} be the set of distributions of detection delay which have μ_i as moment of order i. We are interested in bounding $B(\delta, p)$ for all $\delta \in \Delta_{i,\mu_i}$, that is for all distributions with μ_i as moment of order i. Such a bound is provided by *Markov inequality* presented in section 2.3, and it is optimal:

Proposition 3.24. *Let $i \in \mathbb{N}$ and μ_i. Let $\alpha_i(\mu_i, p) = \sqrt[i]{\frac{\mu_i}{p}}$. Then for all $\delta \in \Delta_{i,\mu_i}$, we have $B(\delta, p) \leq \alpha_i(\mu_i, p)$. Further, $\exists \delta \in \Delta_{i,\mu_i}$ such that $B(\delta, p) = \alpha_i(\mu_i, p)$.*

Proof. It suffices to remark that $\mu_i > pb^i$ for b the bound we want to reach. Further, this bound is trivially optimal: it suffices to consider a distribution with a Dirac of mass $(1-p)$ at 0 and a Dirac of mass p at $\alpha_i(\mu_i, p)$. □

Given an LMC, let δ be its associated distribution of detection delay. We can compute its associated moments μ_i as presented in section 3.3.1. We thus know that $\delta \in \Delta_{i,\mu_i}$. Given different values of i, one can compute the different moments and apply for each of them the Markov's bound and use the minimal bound obtained.

Understanding the relationship between the α_i is thus important. For $i < j$, one can use Jensen's inequality for the convex function $f : x \to x^{\frac{j}{i}}$ over $\mathbb{R}^+$, and obtain: $(\mu_i)^j \leq (\mu_j)^i$. For instance, $\mu_1^2 < \mu_2$.

For $p = 1$, this gives $\alpha_i(p = 1) < \alpha_j(p = 1)$. On the other hand, for p sufficiently close to 0, we have $\alpha_j(p) < \alpha_i(p)$. That is, when p is very small, moments of high orders will give better bounds than moments of lower order. On the other hand, if p is not that small, moments of small order will suffice.

3.3.4 Optimal bounds for a pair of moments

We now explain how to extend Markov's bounds to pairs of moments: we consider the set of distributions where two moments are fixed. Let $i < j$ be two orders of moments and $\mu_i, \mu_j > 0$. We denote by $\Delta_{i,\mu_i}^{j,\mu_j}$ the set of distributions with μ_i, μ_j as moments of order i, j respectively. As $\Delta_{i,\mu_i}^{j,\mu_j}$ is strictly included into Δ_{i,μ_i} and in Δ_{j,μ_j}, $\min(\alpha_i(p), \alpha_j(p))$ is a bound for any $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$. However, it may be the case that $\min(\alpha_i(p), \alpha_j(p))$ is not optimal. We now provide *optimal* bounds $\alpha_i^j(p)$ for any pair $i < j$ of order of moments and probability p:

Theorem 3.25. *Let $i < j$ be natural integers, $p \in (0, 1)$, and let $\mu_i, \mu_j > 0$. Let $\alpha_i = (\frac{\mu_i}{p})^{\frac{1}{i}}$ and $\alpha_j = (\frac{\mu_j}{p})^{\frac{1}{j}}$. We define $\alpha_i^j(p)$ to be:*

- *α_i if $\alpha_i \leq \alpha_j$,*
- *$(\frac{\mu_j - M}{p})^{\frac{1}{j}}$ otherwise, where $0 \leq M \leq \mu_j$ is the smallest positive real root of:*

$$\mu_i = (1-p)^{\frac{j-i}{j}} M^{\frac{i}{j}} + p^{\frac{j-i}{j}} (\mu_j - M)^{\frac{i}{j}}.$$

For all $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$, we have $B(\delta, p) \leq \alpha_i^j$, and $\exists \delta \in \Delta_{i,\mu_i}^{j,\mu_j}$ with $B(\delta, p) = \alpha_i^j$

Let p such that $0 < p < 1$ and μ_i, μ_j be positive real numbers.

case $\alpha_i < \alpha_j$ We prove that in the case where $\alpha_i < \alpha_j$, α_i is actually optimal in $\Delta_{i,\mu_i}^{j,\mu_j}$. This is the first item in Theorem 3.25.

As it is a bound for all $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$, we just need to show that it is optimal.

Proof. Let $0 < \eta < 1$, $0 < p < 1$, and $z > \alpha_i$ a positive real that will be set later.

Let δ be the distribution with mass $(1 - p)$ in 0, mass p_1 in $\eta\alpha_i$, mass p_2 in α_j and mass p_3 in z, with $p_1 + p_2 + p_3 = p$.

We want to choose p_1, p_2, p_3 such that μ_i is the moment of order i and μ_j is the moment of order j, that is such that $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$. We thus have the following equations:

$$p_1 + p_2 + p_3 = p \tag{1}$$
$$p_1(\eta\alpha_i)^i + p_2\alpha_j^i + p_3 z^i = \mu_i \tag{2}$$
$$p_1(\eta\alpha_i)^j + p_2\alpha_j^j + p_3 z^j = \mu_j \tag{3}$$

We denote $A = \alpha_i^i$, $B = \alpha_j^i$, $C = z^i$, $D = (\eta\alpha_i)^j$ and $F = z^j$.

Using (1) and (3), we obtain:

$$p_3 = (p - p_2)\frac{(\mu_j - p(\eta\alpha_i)^j)}{p(F - (\eta\alpha_i)^j)} \tag{4}$$

Granted $p_2 < p$, for $F > (\eta\alpha_i)^j$ (that is $z > \alpha_i$ which we assumed), we get $p_3 > 0$.

Now, using (2), we obtain: $p_3(C - \eta^i A) + p_2(B - \eta^i A) = \mu_i - p\eta^i A$. As $\mu_i = pA$, we get $p_2(B - \eta^i A) + p_3(C - \eta^i A) = pA(1 - \eta^i)$.

Using equivalents for z going to ∞, we get $p_3(C - \eta^i A)$ equivalent to $(1 - p_2/p)C/F$. Notice that C/F tends to 0. We obtain $p_2 = \frac{(1-\eta^i)pA - O(C/F)}{(B-\eta^i A) - O(C/F)}$. For z big enough (η being fixed), we get $p_2 > 0$.

Dividing terms by A, we get $p2 < p\frac{(1-\eta^i)}{(B/A-\eta^i)-O(C/AF)}$. We have $B/A > 1$. For z big enough, $O(C/AF) < B/A - 1$, and we get $p_2 < p$. That is $p_3 > 0$ as well.

Also, remark that in (4), we have $\frac{(\mu_j - p(\eta\alpha_i)^j)}{p(F-(\eta\alpha_i)^j)}$ tends to 0 when z tends to infinity. Hence for z big enough, $p_3 < (p - p_2)$. That is, $p_1 = p - p_2 - p_1 > 0$.

That is, for z big enough, we can chose p_1, p_2, p_3 positive and satisfying the equations we wanted to obtain. That is, $0 < p_1, p_2, p_3 < p$ as $p = p_1 + p_2 + p_3$, and $\mu_1(\delta) = \mu_1$ and $\mu_2(\delta) = \mu_2$. Thus, $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$. Last, we have $B(\delta, p) = \eta\alpha_i$.

□

Case $\alpha_j < \alpha_i$ We now consider the case where $\alpha_j < \alpha_i$, that is the second item of Theorem 3.25. We first prove that the α_i^j defined is a bound for all $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$. We take δ any distribution with μ_i, μ_j for i-th and j-th moments. We let $b = B(\delta, p)$. We partition δ in 2 parts: δ_1 between 0 and b (and 0 elsewhere), and δ_2 after b (and 0 before). We denote $\mu_k(\delta_\ell) = \int_0^\infty \delta_\ell(t)t^k dt$, for $\ell \in \{1, 2\}$.

As δ_2 represents a proportion p of the distribution, and as all the mass is after b, we have the following:

$$\mu_j(\delta_2) = \mu_j - \mu_j(\delta_1) \geq pb^j$$

Lemma 3.26.

$$\mu_j(\delta_1) \geq \frac{(\mu_i - [\mu_j - \mu_j(\delta_1)]^{\frac{i}{j}} p^{\frac{j-i}{j}})^{\frac{j}{i}}}{(1-p)^{\frac{j-i}{i}}}$$

Proof. We apply Jensen inequality to both δ_1 and δ_2.

We obtain $\mu_j(\delta_1) \geq \frac{\mu_i(\delta_1)^{\frac{j}{i}}}{(1-p)^{\frac{j-i}{i}}}$ and $\mu_i(\delta_2) \leq \mu_j(\delta_2)^{\frac{i}{j}} p^{\frac{j-i}{j}}$.

As $\mu_i(\delta_1) = \mu_i - \mu_i(\delta_2)$, we obtain $\mu_i(\delta_1) \geq \mu_i - \mu_j(\delta_2)^{\frac{i}{j}} p^{\frac{j-i}{j}} = \mu_i - [\mu_j - \mu_j(\delta_1)]^{\frac{i}{j}} p^{\frac{j-i}{j}}$, which yields the statement. □

We define the operator f with:

$$f(x) = \frac{(\mu_i - [\mu_j - x]^{\frac{i}{j}} p^{\frac{j-i}{j}})^{\frac{j}{i}}}{(1-p)^{\frac{j-i}{i}}}$$

This operator will allow us to find the bound by a fixpoint computation.

Lemma 3.27. $(f^n(0))_{n\in\mathbb{N}}$ *is strictly increasing and converges towards some* M*.*

Proof. We show by induction on n that $f^n(0)$ is an increasing sequence. First, since $\alpha_j < \alpha_i$, we have that $f(0) \geq 0$.

Then, let $n \in \mathbb{N}$ such that $f^n(0) \geq f^{n-1}(0)$. We have that

$$\begin{aligned}
\mu_j - f^n(0) &\leq \mu_j - f^{n-1}(0) \\
(\mu_i - [\mu_j - f^n(0)]^{\frac{i}{j}} p^{\frac{j-i}{j}})^{\frac{j}{i}} &\geq (\mu_i - [\mu_j - f^{n-1}(0)]^{\frac{i}{j}} p^{\frac{j-i}{j}})^{\frac{j}{i}} \\
\frac{(\mu_i - [\mu_j - f^n(0)]^{\frac{i}{j}} p^{\frac{j-i}{j}})^{\frac{j}{i}}}{(1-p)^{\frac{j-i}{i}}} &\geq \frac{(\mu_i - [\mu_j - f^{n-1}(0)]^{\frac{i}{j}} (p)^{\frac{j-i}{j}})^{\frac{j}{i}}}{(1-p)^{\frac{j-i}{i}}}
\end{aligned}$$

And so, $f^{n+1}(0) \geq f^n(0)$.

Then, let us show that the sequence $f^n(0)$ is bounded. By applying lemma 3.26 with $\mu_j(\delta_1) \geq 0$ on the left hand side, we obtain $\mu_j(\delta_1) \geq f(0)$. Hence we can apply lemma 3.26 with $\mu_j(\delta_1) \geq f(0)$ on the left hand side, yielding $\mu_j(\delta_1) \geq f(f(0))$. By a trivial induction, we obtain $\mu_j(\delta_1) \geq f^n(0)$ for all n.

As this sequence is increasing and bounded, it converges to some quantity M. □

The M of Lemma 3.27 and Theorem 3.25 will be $\mu_j(\delta_1)$ for δ a distribution realizing $B(\delta, p) = \alpha_i^j(p)$. We now prove the second part of Theorem 3.25 and Lemma 3.27.

Lemma 3.28. *Let μ_i, μ_j and p such that $\alpha_j(p, \mu_j) < \alpha_i(p, \mu_i)$. Then for all $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$, we have:*

$$B(\delta, p) \leq (\frac{\mu_j - M}{p})^{\frac{1}{j}}$$

for $M \leq \mu_j$ the smallest positive real root of:

$$\mu_i = (1-p)^{\frac{j-i}{j}} M^{\frac{i}{j}} + p^{\frac{j-i}{j}} (\mu_j - M)^{\frac{i}{j}}.$$

For $i = 1, j = 2$, we can compute explicitly M and obtain:

$$B(\delta, p) \leq \mu_1 + \sqrt{\frac{(1-p)}{p}(\mu_2 - \mu_1^2)}$$

Proof. Let $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$. We denote $b = B(\delta, p)$. We decompose $\delta = \delta_1 + \delta_2$ with δ_1 on $[0, b)$ and δ_2 from $[b, \infty)$.

We showed that the sequence $(f^n(0))$ converges to its convergence point M. We also have $f(M) = M$. Thus, $M \leq \mu_j$ and it is the smallest positive real root of:

$$(1-p)^{\frac{j-i}{j}} M^{\frac{i}{j}} = \mu_i - p^{\frac{j-i}{j}} (\mu_j - M)^{\frac{i}{j}}.$$

Now, we know that $\mu_j(\delta_1) \geq M$. This gives $pB(\delta, p)^j \leq \mu_j - M$.

□

We now tackle the last item of the statement, that is for $i = 1$, $j = 2$, $B(\delta, p) \leq \mu_1 + \sqrt{\frac{(1-p)}{p}(\mu_2 - \mu_1^2)}$.

Proof. Let $i = 1, j = 2$. We have $B(\delta, p) \leq \sqrt{\frac{\mu_2 - M}{p}} = \frac{\mu_1 - \sqrt{1-p}\sqrt{M}}{p}$.

We let $x = \sqrt{M}$. This x satisfies the equation

$$\sqrt{1-p}x = \mu_1 - \sqrt{p}\sqrt{(\mu_2 - x^2)}.$$

That is

$$\sqrt{p}\sqrt{(\mu_2 - x^2)} = \mu_1 - \sqrt{(1-p)}x$$

and hence:

$$p\mu_2 - px^2 = \mu_1^2 + (1-p)x^2 - 2\mu_1\sqrt{(1-p)}x$$

We have the second degree equation:

$$x^2 - 2\mu_1\sqrt{(1-p)}x + \mu_1^2 - p\mu_2 = 0$$

The smallest solution is $x = \mu_1\sqrt{1-p} - \sqrt{(1-p)\mu_1^2 + p\mu_2 - \mu_1^2} = \mu_1\sqrt{1-p} - \sqrt{p}\sqrt{\mu_2 - \mu_1^2}$. This gives:

$$B(\delta, p) \leq \frac{\mu_1 - \sqrt{1-p}(\mu_1\sqrt{1-p} - \sqrt{p}\sqrt{\mu_2 - \mu_1^2})}{p} = \mu_1 + \sqrt{\frac{1-p}{p}(\mu_2 - \mu_1^2)}.$$

□

We end the proof of Theorem 3.25 by showing optimality of the bound for $\Delta_{i,\mu_i}^{j,\mu_j}$:

Lemma 3.29. *Let μ_i, μ_j and p such that $\alpha_j(\mu_j, p) < \alpha_i(\mu_i, p)$. Then there exists a distribution $\delta \in \Delta_{i,\mu_i}^{j,\mu_j}$ with $B(\delta, p) = \sqrt[j]{\frac{1}{p}(\mu_j - M_1)}$ for $M_1 \leq \mu_j$ the smallest positive real root of:*

$$\mu_i = (1-p)^{\frac{j-i}{j}}(M_1)^{\frac{i}{j}} + p^{\frac{j-i}{j}}(\mu_j - M_1)^{\frac{i}{j}}.$$

Proof. Let us consider the distribution δ with:

- $(1-p)$ of the mass at $\sqrt[j]{\frac{M_1}{1-p}}$ and
- p of the mass at $\sqrt[j]{\frac{(\mu_j - M_1)}{p}}$

It trivially satisfies $B(\delta, p) = \sqrt[j]{\frac{(\mu_j - M_1)}{p}}$. Also, we have easily $\mu_j(\delta) = (1-p)\frac{M_1}{1-p} + p\frac{\mu_j - M_1}{p} = \mu_j$.

Now, consider $\mu_i(\delta) = (1-p)^{\frac{j-i}{j}}(M_1)^{\frac{i}{j}} + p^{\frac{j-i}{j}}(\mu_j - M_1)^{\frac{i}{j}}$. By definition of M_1 as a root of the equation $\mu_i = (1-p)^{\frac{j-i}{j}}(M_1)^{\frac{i}{j}} + p(\mu_j - M_1)^{\frac{i}{j}}$, we obtain $\mu_i(\delta) = \mu_i$.

□

To obtain a value for M, one can use for instance Newton's method. For $i = 1, j = 2$, we can compute explicitly M and obtain:

$$\alpha_1^2 = \mu_1 + \sqrt{\frac{(1-p)}{p}(\mu_2 - \mu_1^2)}.$$

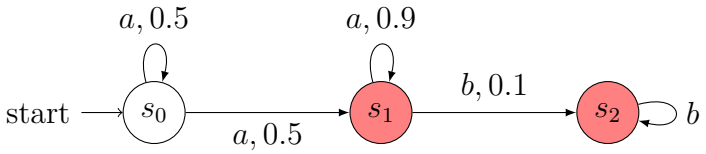

Figure 3.7: A toy example LMC $\mathcal{A}$, faulty states in red.

Example 3.11. *Consider the distribution associated with the system of fig 3.7. Faults occur in state s_1 and are diagnosed in s_2. The distribution follows an exponential law of parameter 0.9. The first moment is $\mu_1 = 10$ and the second is $\mu_2 = 190$.*

With $p = 0.1$, the Markov inequality gives us the following bounds: $\alpha_1(0.1) = \frac{10}{0.1} = 100$, and $\alpha_2(0.1) = \sqrt{\frac{190}{0.1}} \approx 43.6$.

Using the bound we proved, we obtain $\alpha_1^2(0.1) = 10 + \sqrt{\frac{0.9}{0.1}(190 - 10^2)} \approx 38.5$.

We can also use the following result which allows one to underapproximate the value of M, and thus overapproximate the optimal bound, by iterating the following operator f from $x = 0$:

$$f : x \mapsto \frac{(\mu_i - [\mu_j - x]^{\frac{i}{j}} p^{\frac{j-i}{j}})^{\frac{j}{i}}}{(1-p)^{\frac{j-i}{i}}}$$

Example 3.12. *We continue example 3.11 and use the previous method in order to compute bounds using higher moments. We obtain the following bounds $\alpha_i(p), \alpha_i^{i-1}(p)$ considering different values of p and i:*

i	μ_i	$\alpha_i(0.1)$	$\alpha_i^{i-1}(0.1)$	$\alpha_i(0.01)$	$\alpha_i^{i-1}(0.01)$
1	*10*	*100*	*100*	*1000*	*1000*
2	*190*	*43.6*	*38.5*	*137.8*	*104.9*
3	*5410*	*37.8*	***36.8***	*81.5*	*73.9*
4	*205390*	*37.9*	*37.8*	*67.4*	*63.8*
5	*9747010*	*39.6*	*37.9*	*64.2*	***61.43***
6	*555066190*	*42.1*	*39.6*	*62.8*	*61.47*

For $p = 0.1$, it is not useful to consider moments of order higher than 3. On the other hand, for $p = 0.01$, the moment of order 5 provides better bounds than moment of lower orders.

3.4 Related work on diagnosis and diagnosability

In the previous sections, we tackled existing diagnosis notions over our main domain of interest: finite stochastic systems. Of course, these are not the only works talking about diagnosis and diagnosability. In the following, we present different problems around the notion of diagnosis.

3.4.1 Diagnosis of infinite LTS

Although we consider finite models in this thesis, the set of possible systems may be infinite. To represent an LTS with infinite many states, higher order models have to be considered. Different models correspond to this definition, such as Petri Nets and Pushdown Automata.

The semantic of Petri Nets (introduced in [Pet66]) is a reachability graph that is infinite iff the net is not bounded. Cabasino et al. studied diagnosability for both bounded [CGS09] and unbounded [Cab+09] Petri Nets both refined in [Cab+12]. They construct a *verifier net* that is analog to the twin-plant we describe for LTS and the *coverability graph* of this *verifier net*, that is a finite abstraction of the reachability graph of the *verifier net*. However, the *coverability graph* may have an Ackermannian size in the *verifier net*, hence a bad complexity in practice. [Bér+17] tackled this problem and proved that diagnosability of Petri Nets is in EXPTIME by reducing it to the model-checking of an LTL formula on the *verifier net*. An overview of diagnosis on Petri Nets is available in [Bas14]. Diagnosis with Petri Nets unfoldings is presented in [HF13].

Pushdown automata are another class of infinite systems with finite representation. [MP09] investigates their diagnosability and proves the undecidability of this problem by a reduction from the emptiness problem for an intersection of context-free languages. However, it is decidable when restricted to visibly pushdown automata (VPA, introduced in [AM04]). For that, another analog of the twin-plant is used and a Büchi condition is defined on this twin-plant, leading to a PTIME algorithm. More recently, [BHL16b] extended this work to Partially Observable VPA, developing the notions of diagnosability they gave in [BHL14]. Interestingly, although their different settings of diagnosability had all PSPACE-complete complexity on finite LMC, their characterizations are now very different when considering POVPA, leading to different class of complexities, which depend in particular if the system is finitely branching. Surprisingly, when it is infinitely branching, some notions of diagnosability lead to non Borelian set of non-diagnosable runs.

3.4.2 Active diagnosis

Although we present diagnosability in passive way, that is by simply observing the behavior of the systems, some have studied what is called active diagnosis: at each step, a subset of the alphabet $\Sigma_c \subseteq \Sigma$ is selected by a controller and the next action is chosen at random in this subset. The controller may be all powerful (ie it can choose any subset of the alphabet) or it can have some restrictions, such as some actions will always be enabled. Thus, different strategies lead to different controllers. Some controllers will make the system diagnosable while others will not. The goal of active diagnosis is then to find a controller that makes the system diagnosable. This problem was introduced in [SLT98]. In this works, the authors present a procedure to synthesize a sublanguage that is diagnosable wrt the original language thanks to an iterative procedure. However, the complexity of this procedure is not evaluated by the authors but is presented as double exponential by [Haa+13]. The latter proves that active diagnosis is EXPTIME-complete by using two players games on Büchi automata and define optimal controllers, that is with minimal memory. This notion is extended to stochastic systems in [Ber+14] where the authors prove that the complexity for this enriched problem is still EXPTIME-complete. However, they also show that enforcing diagnosability while preserving a positive probability to non faulty runs is an undecidable problem. To prove these claims, the authors make a strong link between active diagnosis on probabilistic systems and Partially Observed Markov Decision Processes.

3.4.3 Diagnosis of distributed systems

Notions of diagnosis presented before were based on the observation of only one system. However, many real life systems are distributed, such as a sensor network: every sensor has a partial view of what happens, and the whole network gives the full information. [FBJ02] and [Su+02] introduce some problems on diagnosis of asynchronous systems. In those works, there is no global time nor global state, thus the challenge is be able to efficiently communicate. Different techniques are investigated in order to tackle this issue, such as Petri Net unfolding [Ben+03; Fab+05]. Standard issues in distributed computations are raised, such as the robustness of distributed diagnosers wrt to the failure of some parts, or improve the scalability [SW04].

Numerous models of distributed diagnosis have been defined, depending on the settings of the model: synchronization, communication delays and/or losses, order preservation of information... Without detailing them, we can mention a few: joint diagnosability [ST02],

codiagnosability [QK06], D-codiagnosability [WYL07]...

Finally, we also refer to [ZL13] for a broad overview of diagnosis on Discrete Event Systems.

3.5 Conclusion

3.5.1 Summary

In section 3.1, a state of the art on diagnosability of stochastic systems has been provided, giving us the tools and foundations we based ourselves on.

Section 3.2 presents results on quantified diagnosability: subsection 3.2.1 gives precise definitions and makes links with previous results by relating our degrees to notions previously defined, especially [ND08] degree and A-diagnosability and subsection 3.2.2 provides algorithms to compute these degrees. Subsection 3.2.3 presented methods to optimize the calculation of a degree. Even if the worst case is unchanged due to the PSPACE-hardness bound, we saw that in some cases the gain could up to an exponential factor.

Section 3.3 explored the time to detect a fault and gave finer results than what we presented in the state of the art. Subsection 3.3.1 presented a mathematical analysis that allowed us to derive appropriate mathematical objects that enabled to compute easily the moments of the distribution of detection delay. These moments were useful in several ways: first, we showed how to use them in order to approximate the distribution as precisely as wanted in subsection 3.3.2. This approximation was possible because the distribution associated to this set of moments is unique in our case. We also proved that these moments allow one to derive better concentration bounds that the ones commonly used in subsection 3.3.3. Moreover, when considering a subset of the moments, the bound derived from this subset is optimal, as shown in subsection 3.3.4.

3.5.2 Future work

Section 3.2.3 provided an algorithm to accelerate the computation of the diagnosability degree, and we saw that in some cases it is very efficient. However, no performance evaluation has been made. This may be a difficult work for some reasons. First, we know that the worst case complexity is unchanged. Maybe one could find some subclasses of LMCs such that this algorithm is efficient on these subclasses. Another option is to evaluate this algorithm on real systems: this is the most interesting benchmark.

Another perspective is quantified opacity. Opacity is another binary property stating if a secret has been leaked or not and has been widely studied on transition systems [Bry+05; Lin11]. Some attempts have been made in order to introduce a quantitative version of opacity [BMS15]. We believe that similar techniques to the one we presented may be useful to calculate some opacity degrees.

Chapter 4

Classification among Labeled Markov Chains

4.1 Introduction

Given several stochastic systems, the problem of classification is to associate a trace of an execution to which system produced it. This can be seen as a symmetric generalization of more specific problems such as diagnosis or opacity. The former, presented in Chapter 3 can be seen as a classification between a faulty language and a correct one. The latter can be seen in some sense as the problem to be able to classify between high and low privileges part of the system [KH18], or between "secret" and "non-secret" part of the system.

In this chapter, we study classification on Labeled Markov Chains, which has been explored by different communities before, such as formal methods [CK14; BHL16b] and control [KH18]. Several variants of this notion can be defined: either one wants to classify for sure, with probability 1 or with arbitrarily small error... Here, after establishing a link between the first two notions (for sure and with probability 1) and well-known problems, we will focus on the last notion (arbitrarily small error), that we call limit-sure classifiability.

More formally, let $(A_i)_{i\leq k}$ be a set of LMCs representing different behaviors of a system under observation. We want to *classify*, *i.e.*, discover which LMC/behavior the system is following, by only looking at an observation sequence $w \in \Sigma^\omega$ it produces. The observer has access to an arbitrarily long prefix of this sequence. Naturally, the longer we observe the system, the larger the size of the observation and the better the information we have to discover the LMC. As it suffices to consider LMCs pairwise, we will consider in the following that there is only a choice between $k = 2$ LMCs. We will denote them by $\mathcal{A}_1$, with n states, and $\mathcal{A}_2$, with m states. In this chapter, we consider a setting where the system will pick $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) with probability $1/2$ and then runs an execution of $\mathcal{A}_1$ (resp. $\mathcal{A}_2$).

A classifier is a function $f : \Sigma^* \to \{\bot, 1, 2\}$ that outputs the index of the LMC from an observation, or possibly $\bot$ if it cannot conclude (yet). Consider for example $\mathcal{A}_1, \mathcal{A}_2$, both following the LMC in figure 4.1, the difference being that $\mathcal{A}_1$ starts in x while $\mathcal{A}_2$ starts in z. If the observation w starts with b, then we know the systems follows $\mathcal{A}_2$, because b is not possible from x. We can thus for all w' let $f(bw') = 2$. However, if the observation is ab^2a, then it could come from any $\mathcal{A}_1$ or $\mathcal{A}_2$. There are several notions of classifiability:

- *sure classifiability*: there exists a classifier f that eventually identifies the accurate LMC that generated w. That is, for all $w \in \Sigma^\omega$, there exists a finite prefix v of w and a classifier f for v such that $f(v) = 1$ (resp. $f(v) = 2$) iff there exists no path ρ of $\mathcal{A}_2$ (resp. of $\mathcal{A}_1$) with $obs(\rho) = w$.
- *almost-sure classifiability*: there exists a classifier f that eventually identifies the accurate LMC that generated w with probability 1. This classifier cannot make errors when it outputs 1 or 2, but there may exist infinite observations that cannot be classified, though the total probability is 0 (such as tossing tail forever on a fair coin).
- *limit-sure classifiability*: there exists a classifier f that, for any $\epsilon > 0$, eventually discovers the correct LMC with probability greater than $1 - \epsilon$.

This leads to the two main questions that we are interested in, for each of the above notions:

- (i) how easily can one decide the existence of a classifier?
- (ii) if there exists a classifier, how easily can one build it explicitly?

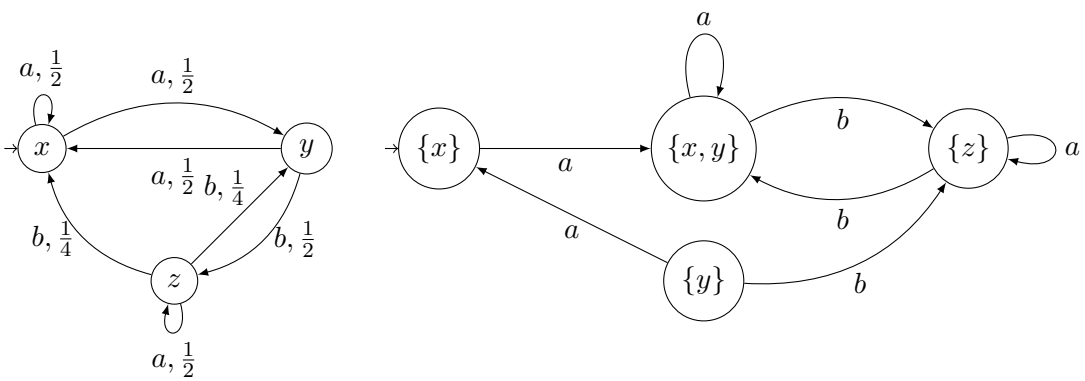

Figure 4.1: Example of an LMC $\mathcal{A}$ on alphabet $\Sigma = \{a, b\}$ and of an NFA $\mathcal{B}_\mathcal{A}$ on alphabet Σ.

For the first two notions, namely sure and almost-sure classification, we recall some well-known results in section 4.2. For limit-sure classifiability, some pioneering works are also presented in the state of the art [CK14; KS16; KH18]. In this chapter, we reinvestigate the problem: in order to answer limit-sure classification, we define a notion of stationary distribution for LMCs to study the long run statistics of the observation w, extending the standard notion of stationary distribution for Markov Chains. To do so, we focus on beliefs, that is the set of states that can be reached with the same observation w_n. We show that a notion of stationary distribution can be defined for beliefs in Bottom Strongly Connected Components (BSCCs), and that it also corresponds to a notion of asymptotic distribution, describing the asymptotic statistics of beliefs.

Stationary distributions allow us to characterize limit-sure classifiability as detailed in [KH18] for a subclass of LMCs. We show that one cannot classify between two LMCs iff they have beliefs which can be reached by the same observation and for which the stationary distributions can be separated by *one finite word* (for which the probability is different). This gives us a polynomial algorithm to decide if two LMCs are limit-sure classifiable. Notice that polynomial time result had been shown in [KS16] with a different approach.

Finally, we consider the classification problem in a security context, called attack-classification: instead of deciding if every observation can be classified, we check for the existence of such an observation that can be classified and the existence of a strategy to obtain it. We then show that deciding if there exists a limit-sure attack-classifier is PSPACE-complete.

This chapter is organized as follows: section 4.2 describes the state of the art on classification. In section 4.3 we define stationary distributions for LMCs and show some of their properties. Section 4.4 contains the main results about limit-sure classification relying on the characterization developed in 4.4.2 and the corresponding algorithm (in 4.4.3). This section also contains a comparison with previous contributions we presented in the state of the art. Attack-classification is then presented in section 4.5. This chapter closes with a discussion about related work.

4.2 State of the art

This section details some results on sure and almost-sure classification and then several ways to compare stochastic systems, going from stronger notions (equivalence) to weakest ones (distances). We start by presenting well-known results on sure and almost-sure

classification in subsection 4.2.1. Then, we state results on the equivalence of stochastic languages in subsection 4.2.2, then we describe notions of distance for stochastic automata in subsection 4.2.3. As we are interested in LMC and not in stochastic automata, we depict an adaptation of these distances to LMCs in subsection 4.2.4. Finally, we outline some very closely related works on distinguishability that we will use as a reference in subsection 4.2.5.

4.2.1 Sure and almost-sure classification

Regarding the first question given in the introduction, that is deciding if there exists a classifier, one can answer easily for the sure and the almost-sure classification, which have been studied in different contexts, such as fault diagnosis (see Chapter 3.1.1 and 3.1.2).

Proposition 4.1. *[Sam+96; BHL14] One can surely classify among 2 LMCs iff* $L^\omega(\mathcal{A}_1) \cap L^\omega(\mathcal{A}_2) = \emptyset$*, and this can be checked in PTIME. One can almost-surely classify among 2 LMCs iff the set* $L^\omega(\mathcal{A}_1) \cap L^\omega(\mathcal{A}_2)$ *has probability 0, and this is a PSPACE-complete problem.*

Proof. The first result is a classical result, in the context of fault-diagnosis [Sam+96], which can be adapted trivially to the case of classification. Clearly, an observation $w \in L^\omega(\mathcal{A}_1) \cap L^\omega(\mathcal{A}_2)$ cannot be classified. Conversely, if $L^\omega(\mathcal{A}_1) \cap L^\omega(\mathcal{A}_2) = \emptyset$, then the product of both LMCs has no loop. It means that with n and m the number of states of $\mathcal{A}_1$ and $\mathcal{A}_2$, after at most $n \cdot m$ observation, we can classify. Checking the existence of a loop in the twin machine is doable in polynomial time (it is an NLOGSPACE-complete problem, see Proposition 3.1).

For the second result we use [TT05; BHL14]: if $L^\omega(\mathcal{A}_1) \cap L^\omega(\mathcal{A}_2)$ has a positive probability, then clearly no almost-sure classifier exists for these observations. Conversely, assume that $L^\omega(\mathcal{A}_1) \cap L^\omega(\mathcal{A}_2)$ has probability 0. Consider the belief automata $\mathcal{B}_1$, $\mathcal{B}_2$ associated with $\mathcal{A}_1, \mathcal{A}_2$ and compute their synchronized product $\mathcal{B}_1 \times \mathcal{B}_2$. The hypothesis implies that all states in BSCCs of this product are either of the form $(D_1, \emptyset)$ or $(\emptyset, B_2)$: one can thus classify when BSCCs are reached, which eventually happens with probability 1. To get the PSPACE algorithm, it suffices to check whether a BSCC of the belief product, with both components non-empty, can be reached. The PSPACE-lower bound follows the one in [BHL14]. □

Finally, for sure and almost-sure classification, building the classifier is also easy: it suffices to compute the set of states reached with the observation w (called *belief* in the

next section) for both LMCs, and wait for a time when one of these beliefs becomes empty and then return the name of the LMC with a non-empty belief state. This event must eventually happen (almost surely with the second notion).

4.2.2 Equivalence of stochastic languages

Given two systems, a strong assessment one can verify is whether they have the same language. In terms of qualitative languages, it means that $L_1 = L_2$ *i.e.*, for any word $w \in \Sigma^*$, $w \in L_1 \Leftrightarrow w \in L_2$. For stochastic systems, such as LMCs, the notion of equivalence has to be extended to take into account the probabilities. Intuitively, two stochastic languages are equivalent if all words have exactly the same probability to be executed by both LMCs. In this section, we present the equivalence problem and LMCs, its complexity and a sketch of proof.

More formally, the equivalence problem is given as follows.

Definition 4.1 (Equivalence of languages for LMCs)**.**
Let $\mathcal{A}_1$, $\mathcal{A}_2$ be two LMCs. $\mathcal{A}_1$ and $\mathcal{A}_2$ are equivalent iff for all w, $P_{\mathcal{A}_1}(w) = P_{\mathcal{A}_2}(w)$.

The equivalence problem for stochastic automata (*i.e.*, LMCs with stopping probabilities) is defined in the same way.

Example 4.1. *Three LMCs are pictured in figure 4.2. $\mathcal{A}_1$ and $\mathcal{A}_2$ are equivalent: all words in $(a+b)^n$ have probability $1/2^n$. However, $\mathcal{A}_3$ is not equivalent with them. As a counterexample, $P_{\mathcal{A}_3}(ab) = 1/2$ instead of $P_{\mathcal{A}_1}(ab) = 1/4$.*

Proposition 4.2 ([Bal93])**.** *The problem of equivalence of two LMCs is decidable in PTIME.*

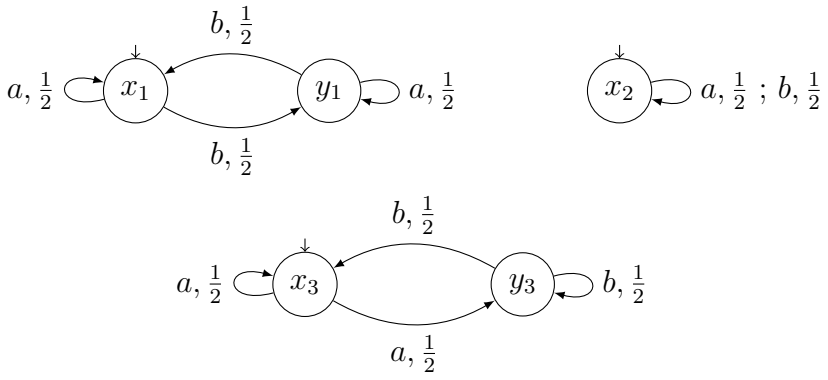

Figure 4.2: Three LMCs $\mathcal{A}_1$ (top left), $\mathcal{A}_2$ (top right) and $\mathcal{A}_3$ (bottom).

The problem to decide if two LMC are equivalent has been proved to be decidable in PTIME in [Bal93]. An extension of the equivalence problem to weighted automata has been presented in [DHR08]. We present a sketch of proof for LMCs following [CK14].

(Sketch of proof for Proposition 4.2). Given two LMCs $\mathcal{A}_1, \mathcal{A}_2$ with initial distributions σ_1, σ_2, the equivalence problem amounts to verifying if for all $w \in \Sigma^*$, $P_{\mathcal{A}_1}(w) = P_{\mathcal{A}_2}(w)$. In matrix form, for an observation $w = a_1 \dots a_k$, this can be written as $\sigma_1 M_1(w)\mathbf{1} = \sigma_2 M_2(w)\mathbf{1}$ with $M_i(w) = \Pi_{j=1}^{k} M_1(a_j)$, $M_1(a_j)$ the transition matrix of $\mathcal{A}_1$ associated to letter a_j and $\mathbf{1}$ the column vector containing only 1s (and similarly for $M_2(w)$). This yields

$$\forall w \in \Sigma^*, \quad (\sigma_1 \quad \sigma_2) \cdot \begin{bmatrix} M_1(w) & \emptyset \\ \emptyset & M_2(w) \end{bmatrix} \cdot (1, \cdots, 1, -1, \cdots, -1)^T = 0$$

We define $Eq(\mathcal{A}_1, \mathcal{A}_2) = span\{ \begin{bmatrix} M_1(w) & \emptyset \\ \emptyset & M_2(w) \end{bmatrix} \cdot (1, \cdots, 1, -1, \cdots, -1)^T \mid w \in \Sigma^* \}$.

$Eq(\mathcal{A}_1, \mathcal{A}_2)$ is a vector space and its dimension is at most $n + m$, thus we can build a basis $v_1, \dots v_\ell$ for $Eq(\mathcal{A}_1, \mathcal{A}_2)$ of size $\ell \leq n + m$. It suffices then to check whether $(\sigma_1 \quad \sigma_2)$ falls in the left kernel of $Eq(\mathcal{A}_1, \mathcal{A}_2)$ which amounts to $(\sigma_1 \quad \sigma_2) \cdot v_i = 0$ for all $i \leq \ell$.

□

Notice that this problem is very close to equivalence for languages of PFAs which has first been shown to be in coNP in [Paz71] and then in PTIME in [Tze92]. A corollary of [DHR08] is that equivalence for both settings (LMCs and PFAs) is inter-reducible. Especially, the proof sketched from [CK14] uses very similar ideas to the one in [Tze92].

4.2.3 Distance between stochastic automata

When two systems define exactly the same (non-stochastic) language, one may want to quantify the difference between them. Different notions of distance have been used to perform this quantification. In this subsection, we illustrate some of them. The standards distances to study are the L_p ones. However, it is unclear how to define them on LMCs with infinite words. Thus, we start by giving definitions and results for these distances on stochastic automata. We will later refer to work on the total variation distance which is derived from the L_1 distance and is also suited for LMCs. We recall that a stochastic automaton is a weighted automaton over the probabilistic semiring and can be seen as an LMC with *stopping probability*. In the related work section, other distances will be mentioned.

L_p distance

In mathematics, L_p spaces involve functions which p-power is measurable and summable in the sense of Lebesgue. As probabilistic automata can be seen as functions that associate a real number to a word in Σ^* a L_p norm can be defined on them in a similar manner.

Definition 4.2 (L_p distance between two automata)**.**
Let $p \geq 1$ and let $\mathcal{A}_1, \mathcal{A}_2$ be two stochastic automata with respective probability distribution p_1 and p_2. The L_p distance between $\mathcal{A}_1$ and $\mathcal{A}_2$ is given by:

$$L_p(\mathcal{A}_1, \mathcal{A}_2) = (\sum_{w \in \Sigma^*} |p_1(w) - p_2(w)|^p)^{\frac{1}{p}}$$

Notice that for all p, two stochastic automata are equivalent (notion developed in section 4.2.2) iff their L_p distance is 0 . In particular, an L_p distance being 0 for some p is equivalent to being 0 for all p. The usually considered decision problem is "given $\mathcal{A}_1, \mathcal{A}_2, \theta \in \mathbb{R}$, $L_p(\mathcal{A}_1, \mathcal{A}_2) = \theta$?".

This problem has been tackled in various works [LP02; CMR06; CMR07; CK14; Kie18]. We recall the most important results.

Proposition 4.3 (L_p distance for even p [CMR06; CMR07])**.** *Given two stochastic automata $\mathcal{A}_1, \mathcal{A}_2$ and given an even value of p, the decision problem associated to the L_p distance is decidable with time complexity $O((|\mathcal{A}_1| + |\mathcal{A}_2|)^{6p})$, which is polynomial for a fixed p.*

Further, if the stochastic automaton is unambiguous, that is for all $w \in \Sigma^*$ there exists only one path that accepts w, then the complexity becomes polynomial even when p is part of the input with time complexity $O(2p|\mathcal{A}_1|^3|\mathcal{A}_2|^3)$. However, when p is odd, the problem becomes much more complex. [LP02] showed that the complexity was at least NP-hard for L_1 and [CMR06] extended this NP-hardness to every odd value of p. Finally, [Kie18] refined this result:

Proposition 4.4 (L_1 distance [Kie18])**.** *Given two stochastic automata $\mathcal{A}_1, \mathcal{A}_2$, the decision problem associated to the L_1 distance is undecidable.*

[Kie18] also proved that approximating the L_1 distance was in PSPACE and #P-hard.

Definition 4.3 (L_∞ distance)**.** *Let $\mathcal{A}_1, \mathcal{A}_2$ be two stochastic automata with respective probability distribution p_1 and p_2. The L_∞ distance between $\mathcal{A}_1$ and $\mathcal{A}_2$ is given by:*

$$L_\infty(\mathcal{A}_1, \mathcal{A}_2) = \max_{w \in \Sigma^*} |p_1(w) - p_2(w)|$$

The L_∞ distance is also sometimes considered, but its complexity remains high: at least NP-hard [LP02].

To sum up, deciding if two stochastic automata are equivalent, that is if they are at distance 0 for some L_p distance is computationally easy. However computing the exact distance in a general setting remains difficult.

4.2.4 Total variation distance and the distance 1 problem

While the L_p distances are not well suited for LMCs, one can define the total variation distance [GS02] for LMCs. For stochastic automata, we can show that the total variation distance and the L_1 distances are equal up to a factor 2, hence the total variation distance is a good replacement for the L_1 distance for LMCs. In the following, for a probability measure p on *finite* words in a stochastic automaton, we denote $p(W) = \sum_{w \in W} p(w)$. Analogously, for a probability measure p on *infinite* words in an LMC, we denote $p(W) = \sum_{w \in W} p(w)$. Notice that for LMCs we need the words to be infinite.

Definition 4.4 (Total variation distance on stochastic automata)**.**

Given two stochastic automata $\mathcal{A}_1, \mathcal{A}_2$, with respective probability distribution p_1 and p_2, the total variation distance is given by

$$d(\mathcal{A}_1, \mathcal{A}_2) = \max_{W \subseteq \Sigma^*} |p_1(W) - p_2(W)|$$

This distance is the biggest possible difference of event probabilities between $\mathcal{A}_1$ and $\mathcal{A}_2$. We can make the following link between the total variation distance and L_1:

Proposition 4.5 ([MU17])**.** $d(\mathcal{A}_1, \mathcal{A}_2) = \frac{1}{2} L_1(\mathcal{A}_1, \mathcal{A}_2)$

Because of Proposition 4.4, this proposition implies the impossibility to compute exactly the total variation distance, and the difficulty to approximate it [Kie18]. The total variation distance can be extended to LMCs as follows:

Definition 4.5 (Total variation distance on LMCs)**.**

Given two LMCs $\mathcal{A}_1, \mathcal{A}_2$, with respective probability distribution p_1 and p_2, the total variation distance is given by

$$d(\mathcal{A}_1, \mathcal{A}_2) = \sup_{\substack{W \subseteq \Sigma^\omega \\ \text{measurable}}} |p_1(W) - p_2(W)|$$

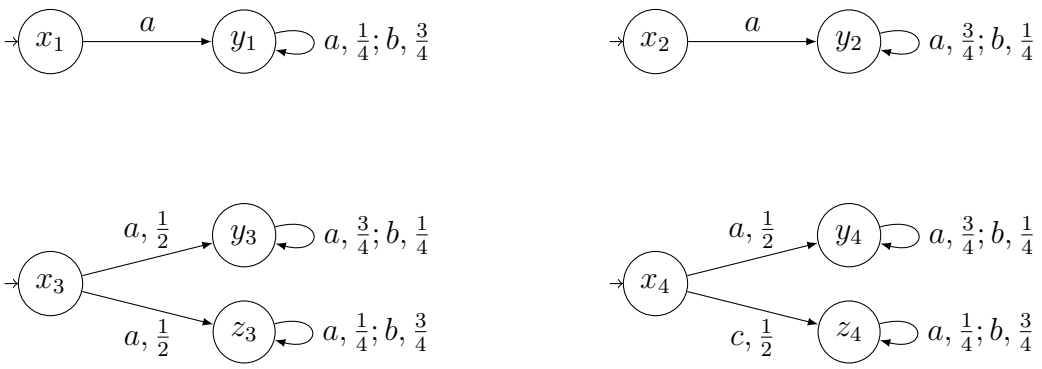

Figure 4.3: Four LMCs $\mathcal{A}_1$ (top left), $\mathcal{A}_2$ (top right), $\mathcal{A}_3$ (bottom left) and $\mathcal{A}_4$ (bottom right).

Notice that we now have a supremum due to the uncountable number of possible measurable sets W. However, one can show the existence of a measurable set of infinite runs that maximizes this quantity which turns this supremum into a maximum. We denote w_i the prefix of length i of w and $q(w) = \lim \frac{p_1(w_i)}{p_2(w_i)}$ if this limit exists.

Theorem 4.6 ([CK14])**.** *The set* $W_> = \{w \in \Sigma^\omega | q(w) > 1\}$ *maximizes* $|p_1(W) - p_2(W)|$.

Even if this distance is not computable in general, some verification can nevertheless be performed:

Definition 4.6 (Distance 1 problem)**.**
Given two LMCs $\mathcal{A}_1, \mathcal{A}_2$*, does* $d(\mathcal{A}_1, \mathcal{A}_2) = 1$ *hold.*

Intuitively, if two LMCs are at distance 1, then there exists a set of infinite runs that has probability 1 in one LMC and probability 0 in the other one. Coming back to our classification problem, it would be possible to know which LMC produced an observation $w \in \Sigma^\omega$.

Example 4.2. *We illustrate the notion of distance* 1 *with the four LMCs in figure 4.3.* $\mathcal{A}_1, \mathcal{A}_2$ *are at distance* 1*: by denoting* $|w_n|_a$ *the number of* a *in the prefix of length* n *of* w*, the set* $W_1 = \{w \in \Sigma^\omega \mid \lim_{n\to\infty} \frac{|w_n|_a}{n} > \frac{1}{2}\}$ *has probability* 1 *in* $\mathcal{A}_2$ *and probability* 0 *in* $\mathcal{A}_1$*. On the contrary,* $\mathcal{A}_1$ *and* $\mathcal{A}_3$ *are not at distance* 1*: intuitively, the upper part of* $\mathcal{A}_3$ *is totally different, however the lower part has exactly the same behavior as* $\mathcal{A}_1$*. Finally,* $\mathcal{A}_1$ *and* $\mathcal{A}_4$ *are at distance* 1*. Again, the upper part behaves in the same way as* $\mathcal{A}_2$ *and then is "very different" from the behavior of* $\mathcal{A}_1$*. Even if the state* z_4 *has the same behavior as* y_1*, as they are not reachable by the same prefix.*

The distance 1 problem is computationally easy:

Theorem 4.7 ([CK14])**.**
The distance 1 *problem can be decided in PTIME.*

We sketch the proof of this theorem since we will give an alternative one in this chapter. Let us define some notation: for a word $w \in \Sigma^*$, $\mathcal{A}_1(w)$ is the set of states s such that there exists an execution labeled by w leading to s. For two LMCs $\mathcal{A}_1 = (S_1, \Sigma, \mu_{01}, p_1), \mathcal{A}_2 = (S_2, \Sigma, \mu_{02}, p_2)$, two distributions μ_1, μ_2 are said to be equivalent if $\mathcal{A}'_1 = (S_1, \Sigma, \mu_1, p_1), \mathcal{A}'_2 = (S_2, \Sigma, \mu_2, p_2)$ are equivalent in the sense of languages of LMCs.

Proposition 4.8 ([CK14])**.** *Given two LMCs $\mathcal{A}_1, \mathcal{A}_2$, the following are equivalent:*

1. *$d(\mathcal{A}_1, \mathcal{A}_2) < 1$,*
2. *there exists $w \in \Sigma^*$ and probability subdistributions μ_1, μ_2 such that $supp(\mu_1) \subseteq \mathcal{A}_1(w)$, $supp(\mu_2) \subseteq \mathcal{A}_2(w)$ and μ_1 and μ_2 are equivalent,*
3. *there exists $r_1 \in S_1$ and equivalent subdistributions μ_1, μ_2 such that $r_1 \in supp(\mu_1)$ and $supp(\mu_2) \subseteq \{r_2 \mid (r_1, r_2) \in \mathcal{A}_1 \times \mathcal{A}_2\}$.*

Furthermore, by arguments similar to those developed in [Tze92] and [DHR08], if there exists such a w then there is one that has a length lower than $2 \cdot (|S_1| + |S_2|)$. The existence of these subdistributions can be checked in polynomial time thanks to linear programming, hence the polynomial time algorithm 2.

Algorithm 2 PTIME algorithm for the distance 1 problem

```
A is the twin automaton A_1 × A_2.
for r_1 ∈ S_1 do
    Let R_2 = {r_2 | (r_1, r_2) ∈ A}.
    if there exist two distributions μ_1, μ_2 with r_1 ∈ supp(μ_1) and supp(μ_2) ⊆ R_2
        with (A_1, μ_1) ≡ (A_2, μ_2) then
        return d(A_1, A_2) < 1
    end if
end for
return d(A_1, A_2) = 1
```

4.2.5 Distinguishability

Let $\mathcal{A}_1, \mathcal{A}_2$ be two LMCs. Distinguishability is the problem of determining the existence of a monitor that can check with arbitrary precision from which LMC an observation comes

from. It is thus similar to classification. This notion has been studied in [KS16] and is strongly related to the distance 1 problem.

Definition 4.7 (Monitor)**.**

A monitor is a function $M : \Sigma^* \to \{\bot, 1\}$ *such that if* $M(u) = 1$ *then for all* v*,* $M(uv) = 1$*.*

Given a monitor M, we denote by $L(M)$ the set of infinite executions w such that there exists a prefix u of w with $M(u) = 1$. The set $L(M)$ is measurable as a countable union of cylinders.

Definition 4.8 (Distinguishability)**.**

Let $\mathcal{A}_1, \mathcal{A}_2$ *be two LMCs inducing respective probability measure* π_1, π_2*.* $\mathcal{A}_1, \mathcal{A}_2$ *are said to be distinguishable if for all* $\varepsilon > 0$ *there exists a monitor* M_ε *such that*

$$\pi_1(L(M_\varepsilon)) \geq 1 - \varepsilon \text{ and } \pi_2(L(M_\varepsilon)) \leq \varepsilon$$

Note that even if this definition seems not symmetric, there exists a monitor M_ε such that $\pi_1(L(M_\varepsilon)) \geq 1 - \varepsilon$ and $\pi_2(L(M_\varepsilon)) \leq \varepsilon$ iff there exists a monitor M'_ε such that $\pi_1(L(M_\varepsilon)) \leq \varepsilon$ and $\pi_2(L(M_\varepsilon)) \geq 1 - \varepsilon$ [KS16].

As we said before, the existence of such a monitor is strongly related to the distance 1 problem:

Proposition 4.9 ([KS16])**.** *Two LMCs* $\mathcal{A}_1, \mathcal{A}_2$ *are distinguishable iff* $d(\mathcal{A}_1, \mathcal{A}_2) = 1$*.*

Proof. If $d(\mathcal{A}_1, \mathcal{A}_2) = 1$, then from [CK14] for every $\varepsilon > 0$ there exists W such that $p_1(W\Sigma^\omega) \geq 1 - \varepsilon$ and $p_2(W\Sigma^\omega) \leq \varepsilon$. Then, let M be a monitor outputting 1 after reading a string in W. We trivially have $p_1(L(M)) \geq 1 - \varepsilon$ and $p_1(L(M)) \leq \varepsilon$. Hence, $\mathcal{A}_1$ and $\mathcal{A}_2$ distinguishable.

If $\mathcal{A}_1$ and $\mathcal{A}_2$ are distinguishable, then for every $\varepsilon >$ there exists a monitor M_ε such that $p_1(L(M_\varepsilon)) \geq 1 - \varepsilon$ and $p_2(L(M_\varepsilon)) < \varepsilon$.Then,

$$\begin{aligned} d(\mathcal{A}_1, \mathcal{A}_2) &\geq \sup_\varepsilon |p_1(L(M_\varepsilon)) - p_2(L(M_\varepsilon))| \\ &\geq 1 - 2\varepsilon \\ &\geq 1 \end{aligned}$$

□

As a corollary, deciding if two LMCs are distinguishable can be done in PTIME. We will see in section 4.4.4 that distinguishability and limit-sure classifiability coincide on LMCs.

4.2.6 Misclassification

In this last part of the state of the art, we present a recent work on the probability of misclassification in the context of probabilistic opacity of LMCs [KH18]. Let $\mathcal{A}_1, \mathcal{A}_2$ be two LMCs with respective probability measures p_1, p_2. As for classification, the system chooses at random between two LMCs $\mathcal{A}_1$ and $\mathcal{A}_2$ (to simplify, here we consider probabilities half half) and an observation w from this LMC is produced. One would want to decide if w has been produced by $\mathcal{A}_1$ or $\mathcal{A}_2$.

Consider the maximum a posteriori probability (MAP) rule where the answer for an observation w is 1 (resp. 2) if $p_1(w) > p_2(w)$ (resp. $p_1(w) > p_2(w)$). Given an observation w, the probability to misclassify w is $P_{err}(w) = \min(p_1(w), p_2(w))$. Then, given $n \in \mathbb{N}$, the probability of misclassification, *i.e.*, the probability to make an error by watching an observation of size n is

$$P_{err}(n) = \sum_{w \in \Sigma^n} P_{err}(w)$$

Definition 4.9 (Misclassification error)**.**

We say that the probability of misclassification error among two LMCs tends to 0 *iff* $\forall \varepsilon > 0, \exists n_0 \in \mathbb{N}, \forall n > n_0, P_{err}(n) < \varepsilon$

Actually, distinguishability (and thus limit-sure classification) are equivalent with the probability of misclassification error tending to 0, as shown in Proposition 4.12. [KH18] obtain the following sufficient condition:

Proposition 4.10. *Let $\mathcal{A}_1, \mathcal{A}_2$ be two LMCs with corresponding MC $\mathcal{M}_1$ and $\mathcal{M}_2$ such that $\mathcal{M}_1$ and $\mathcal{M}_2$ are irreducible and aperiodic. Let σ_i be the stationary distribution of $\mathcal{M}_i$. Then, if $\mathcal{A}_1$ with initial distribution σ_1 and $\mathcal{A}_2$ with initial distribution σ_2 are equivalent (see 4.2.2) then the probability of misclassification error tends to* 0*, i.e.,*

$$\forall \varepsilon > 0, \exists n_0 \in \mathbb{N}, \forall n > n_0, P_{err}(n) < \varepsilon$$

[KH18] solves this problem on a subclass of LMCs: those for which the associated Markov Chain is strongly connected with period 1 and crucially, for all state the initial probability is positive.

In the following, we denote $\alpha(w) = \min\{\frac{p_1(w)}{p_1(w)+p_2(w)}, \frac{p_2(w)}{p_1(w)+p_2(w)}\}$.

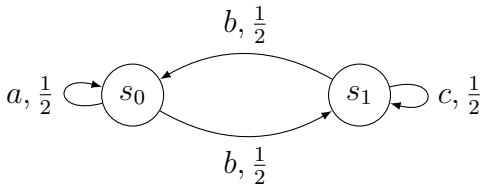

Figure 4.4: States and transitions for four LMCs $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$, $\mathcal{A}_4$ with different initial probabilities

Theorem 4.11. *Let $\mathcal{A}_1, \mathcal{A}_2$ be two LMCs with corresponding MC $\mathcal{M}_1$ and $\mathcal{M}_2$ such that $\mathcal{M}_1$ and $\mathcal{M}_2$ are irreductible and aperiodic and such that exry state is an initial state. Let σ_i be the stationary distribution of $\mathcal{M}_i$. Then $\sum_{w\in\Sigma^n,\alpha(w)>\alpha_0} p_1(w) + p_2(w) \to_0$ iff $\mathcal{A}_1$ with initial distribution σ_1 and $\mathcal{A}_2$ with initial distribution σ_2 are equivalent.*

Example 4.3. *We now show that the condition on the initial distribution is crucial.*

Let us consider some LMCs based on the structure given in figure 4.4. We consider different systems with different initial distributions. $\mathcal{A}_1$ (resp. $\mathcal{A}_2$, $\mathcal{A}_3$, $\mathcal{A}_4$) has initial distribution $(1, 0)$ (resp. $(0, 1)$, $(0.25, 75)$, $(0.5, 0.5)$). Of course, all MC corresponding to these LMCs have the same stationary distribution $(0.5, 0.5)$.

By looking at an observation long enough $\mathcal{A}_1$ and $\mathcal{A}_2$ can be differentiated with probability 1: in the former an "a" will only occur after an even number of "b" whereas in the latter it will only be after an odd number of "b". The only word of length n that is ambiguous is b^n whose probability is $1/2^n$. Thus, the probability of error tends toward 0.

However, when considering $\mathcal{A}_3$ and $\mathcal{A}_4$, this reasoning is not enough. Seeing an "a" after an even (resp. odd) number of "b" means that the initial state was s_0 (resp. s_1), and a similar reasoning can be applied for "c". This is possible for both $\mathcal{A}_3$ and $\mathcal{A}_4$. Since the sets of states and transitions are the same for $\mathcal{A}_3$ and $\mathcal{A}_4$, once we know the initial state we cannot gain more information. Thus, the probability of misclassification does not tend toward 0.

Finally, we make a link between distinguishability and misclassification.

Proposition 4.12. *Let $\mathcal{A}_1, \mathcal{A}_2$ be two LMCs.*

$\mathcal{A}_1$ and $\mathcal{A}_2$ are distinguishable $\Leftrightarrow$ the probability of misclassification between $\mathcal{A}_1$ and $\mathcal{A}_2$ tends to 0.

Proof. Let $\mathcal{A}_1, \mathcal{A}_2$ be two LMCs with respective probability distributions p_1 and p_2.

If $\mathcal{A}_1$ and $\mathcal{A}_2$ are distinguishable then for all ε, there exists k_ε and $W_{k_\varepsilon} \subseteq \Sigma^{k_\varepsilon}$ such that $p_1(W_{k_\varepsilon}\Sigma^\omega) \geq 1-\varepsilon$ and $p_2(W_{k_\varepsilon}\Sigma^\omega) \leq \varepsilon$ [CK14]. Then

$$P_{err}(k_\varepsilon) = \sum_{w \in W_{k_\varepsilon}} \min(p_1(w), p_2(w)) + \sum_{w \in \Sigma^k \setminus W_{k_\varepsilon}} \min(p_1(w), p_2(w))$$

$$P_{err}(k_\varepsilon) \leq \sum_{w \in W_{k_\varepsilon}} p_2(w) + \sum_{w \in \Sigma^k \setminus W_{k_\varepsilon}} p_1(w)$$

$$P_{err}(k_\varepsilon) \leq \varepsilon + \varepsilon$$

Thus, the probability of error tends to 0.

Conversely, if the probability of error between $\mathcal{A}_1$ and $\mathcal{A}_2$ tends to 0. For all ε, there exists k_ε such that $P_{err}(k_\varepsilon) \leq \varepsilon$.

Let $W_{1,k_\varepsilon} = \{w \in \Sigma^{k_\varepsilon} \mid p_1(w) \leq p_2(w)\}$ and $W_{2,k_\varepsilon} = \Sigma^{k_\varepsilon} \setminus W_{1,k_\varepsilon}$. We obtain that $\sum_{w \in W_{1,k_\varepsilon}} p_1(w) \leq \varepsilon$ and $\sum_{w \in W_{2,k_\varepsilon}} p_2(w) \leq \varepsilon$. Then, $p_1(W_{1,k_\varepsilon}\Sigma^\omega) \leq \varepsilon$ and $p_2(W_{2,k_\varepsilon}\Sigma^\omega) \geq 1-\varepsilon$. By [CK14], $\mathcal{A}_1$ and $\mathcal{A}_2$ are distinguishable. □

4.3 Beliefs and stationary distributions for LMCs

In order to solve the classification problem, we would like to use statistics on an observation $w \in \Sigma^\omega$. For this, it is important to know the proportion of time an execution spends in each state on average as done in [KH18]. With this information, we can deduce an "average behavior": since we observe an infinite execution, we know that with high probability its behavior will be close to the average. Stationary distributions, a concept used for *Markov chains* (see Chapter 2.1.4 and 2.2), give information on this average behavior. However, since we consider a more complex model, this is not enough. We will thus generalize this concept to LMCs in the following. While it is crucial in the realm of classifiability, we believe it is also of independent interest.

For a Markov chain $\mathcal{M}$, a stationary distribution σ is a distribution over states of $\mathcal{M}$ such that $\sigma \cdot M = \sigma$. In LMCs, the observation w plays an important role and changes our knowledge of states in which the run could be at each time. Thus, we consider the set of states that could be reached by an LMC $\mathcal{A}$ with a given observation, and call this the *belief-state* or simply the *belief*. Formally, let $w \in \Sigma^*$ be a finite observation. The *belief* $B_\mathcal{A}(w)$ associated with w is the set of states $\{s^+(\rho) \mid obs(\rho) = w\}$ that is states that can be reached by a path labeled by w. For instance, with the LMC $\mathcal{A}$ from figure 4.5, we have $B_\mathcal{A}(aa) = \{x, y\}$. We let $\mathcal{B}_\mathcal{A} = (2^S, \Delta, s_0)$ be the (deterministic) *belief automaton*

associated with $\mathcal{A}$:

- (i) its states are the subsets of states of $\mathcal{A}$,
- (ii) $(B, a, B') \in \Delta$ iff $B' = \{b' | \exists b \in B, M(a)_{b,b'} > 0\}$,
- (iii) $s_0 = \{s \mid \sigma_0(s) > 0\}$.

This is the usual subset construction used for determinizing an automaton, as shown on figure 4.5. Notice that $\mathcal{B}_\mathcal{A}$ is deterministic.

Consider a BSCC D of LMC $\mathcal{A}$ (as for Markov chains, this is to ensure irreducibility). For $x \in D$, we denote by $\mathcal{B}_D^x$ the subgraph of $\mathcal{B}_\mathcal{A}$ reachable from $\{x\}$. (Notice that $\{x\}$ may not be reachable in $\mathcal{B}_\mathcal{A}$ from its initial state.) On figure 4.5, we have $\mathcal{B}_D^y = \mathcal{B}_\mathcal{A}$. It has a unique BSCC (of beliefs), with 2 beliefs $\{x, y\}$ and $\{z\}$. Remember that we always exclude the trivial $\{\emptyset\}$ BSCC. We now show that this is the general form of the belief automaton:

Lemma 4.13. *There is a unique BSCC in $\mathcal{B}_D^x$, and it does not depend upon $x \in D$.*

Proof. Assume by contradiction that $X_1 \subseteq S$ and $X_2 \subseteq S$ belong to two distinct BSCCs of $\mathcal{B}_D^x$ (wlog, we can choose $x \in X_1, x \in X_2$ as x is reachable from any state in D, and thus x must belong to at least one member of each BSCC). Let w_1, w_2 be observations reaching X_1 and X_2 respectively from $\{x\}$. As $x \in X_1$, there is a path in $\mathcal{B}_D^x$ labeled w_2 from X_1 to some X_2' with $X_2 \subsetneq X_2'$ (they cannot be equal because they are in 2 different BSCCs).

As $x \in X_2$, there is a path in $\mathcal{B}_D^x$ labeled w_1 from X_2 to some X_1' with $X_1 \subsetneq X_1'$. We can then play w_2 to obtain some X_2'' from X_1' with $X_2' \subsetneq X_2''$. We can iterate this process infinitely, which gives a contradiction with the bounded number of states.

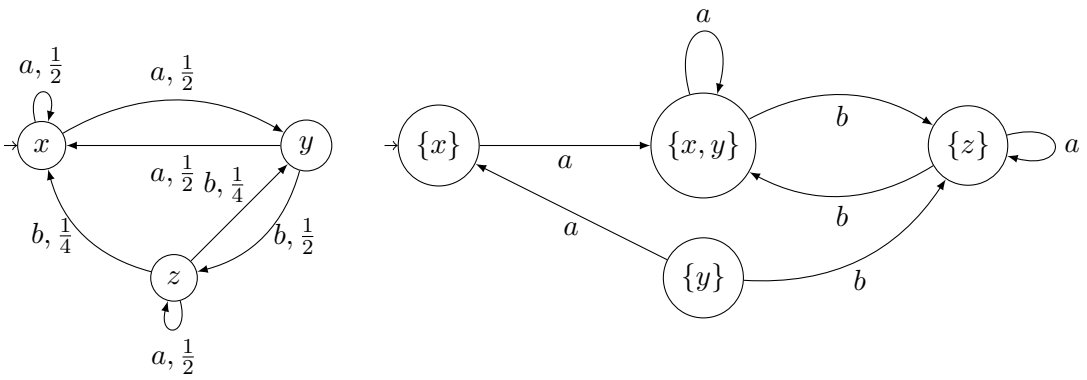

Figure 4.5: Example of an LMC $\mathcal{A}$ on alphabet $\Sigma = \{a, b\}$ and of an NFA $\mathcal{B}_\mathcal{A}$ on alphabet Σ.

In the same way, consider $\mathcal{B}_D^x$ and $\mathcal{B}_D^y$, and assume by contradiction that they have different BSCCs. Let Y (resp. X) be a configuration in the unique BSCC of $\mathcal{B}_D^x$ (resp. $\mathcal{B}_D^y$), reachable by playing w_1 (resp. w_2), with $x \in X$ and $y \in Y$. One can play w_2 (resp. $w_1 w_2$) from Y (resp. X) and reach some X'', with $X \subsetneq X' \subsetneq X''$. Again, one can iterate and reach a contradiction with the boundedness of the number of states. □

Definition 4.10. *Let $\mathcal{A}$ be an LMC. For D a BSCC of $\mathcal{A}$ and $x \in D$, let B_D^x the belief automaton with starting state $\{x\}$. We denote E_D the set of beliefs X in the unique BSCC of $\mathcal{B}_D^x$ (remember it does not depend on $x \in D$). Last, we denote $E_\mathcal{A} = \bigcup_D E_D$, that is the union of E_D over all BSCCs D of $\mathcal{A}$.*

Notice that this definition can be also applied to non-probabilistic systems, namely LTS. Notice also that $E_\mathcal{A}$ may not contain all beliefs in the BSCCs of $\mathcal{B}_\mathcal{A}$, because E_D is restricted to beliefs X reachable from $\{x\}$ with a single state x of a BSCC of $\mathcal{A}$. This is crucial for lemma 4.13 to hold. We will see that considering singletons is not a restriction: assume that the belief reached in a BSCC of beliefs comes from a belief $\{x, y\}$ with $x \neq y$. Either the stochastic languages from x and y are the same, in which case we change nothing by considering only x as a starting point. Otherwise, they induce different statistics on upcoming observations, and looking at the observed statistics will give away with arbitrarily small error the state x or y which they originate from.

For Markov chains (*i.e.*, LMCs on a one letter alphabet), the BSCC E_D is exactly $X_1 \to X_2 \cdots \to X_k \to X_1$, with k the *period* of this BSCC. Hence, the construction above can be seen as a generalization to LMCs of the notion of Markov chain's period. We use it to generalize the Fundamental theorem of Markov chains (see Chapter 2.2) to LMCs.

Let $X \in E_\mathcal{A}$. We are interested in the *asymptotic distribution* associated to belief X, that is the asymptotic distribution over states of X given that the belief state is X. From that, we will be able to deduce the statistics over observations. Let W_X be the (possibly countable infinite) set of words which bring from belief X to belief X without seeing belief X in-between and $W_X^i = \{w_1 \dots w_i \,|\, \forall k \leq i, w_k \in W_X\}$, that is a concatenation of i words in W_X. For two states y, x and a finite observation w, we define $M(y, w, x) = 1_y \cdot M(w) 1_x^T$ with 1_y (resp. 1_x) the vector equal to 1 in position y (resp. x) and 0 elsewhere.

Let $y \in X$ and $i \in \mathbb{N}$. Consider $\sigma_{y,i}$ the distribution over X such that $\sigma_{y,i}(x) = \sum_{w \in W_X^i} M(y, w, x)$, the probability of reaching x from y after seeing i words of W_X. We want to compute the limit of $\sigma_{y,i}$. First, let us remark that this limit exists, as W_X^i is increasing with i and $\sum_{w \in W_X^i} M(y, w, x) \leq 1$ since for all $w, w' \in W_X^i$, w is not a prefix of w'. Then, we define the stationary distribution $\sigma_X : X \to [0, 1]$ of the LMC given a belief X. For that, we enrich the states of $\mathcal{A}$ with its beliefs, considering the product

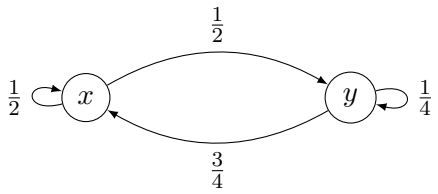

Figure 4.6: Markov chain $\mathcal{M}_{x,y}$ associated with the belief $\{x, y\}$

$\mathcal{A} \times \mathcal{B}_\mathcal{A}$ (same runs with same probabilities as in $\mathcal{A}$). For all $y, x \in X$, let $M_X(y, x)$ be the probability in the LMC $\mathcal{A} \times \mathcal{B}_\mathcal{A}$ to reach (x, X) from (y, X) before reaching any other (z, X) (which can be computed by the algorithm for ε-removal presented in Chapter 2.1.2). We have that for all $x \in X$, $\sum_{y \in X} M_X(x, y) = 1$, that is M_X is the transition matrix of a Markov chain.

Example 4.4. *Let us consider the LMC in figure 4.5 and let $X = \{x, y\} \in E_\mathcal{A}$. The Markov chain $\mathcal{M}_X$ build from this belief state is depicted in figure 4.6 has a unique stationary distribution $\sigma(x) = \frac{3}{5}$ and $\sigma(y) = \frac{2}{5}$.*

We obtain:

Theorem 4.14. *Given an LMC $\mathcal{A}$, let X be a belief in $E_\mathcal{A}$. Then, M_X has a unique stationary distribution denoted $\sigma_X : X \to [0, 1]$, i.e., $\sigma_X \cdot M_X = \sigma_X$. Further, for all $y \in X$, $\sigma_{y,i} \underset{i \to +\infty}{\longrightarrow} \sigma_X$.*

Proof. We first prove that there exists ℓ such that for all $x, y \in X$, we have $M_X^\ell(x, y) > 0$. Then using the Fundamental theorem of Markov chains (see Theorem 2.2), we will be able to conclude that there is a unique stationary distribution σ_X of M_X [KS60]. So, to see the former statement, for all $x \in X$, by lemma 4.13, there is an observation v_x leading from $\{x\}$ to X, *i.e.*, $\Delta(\{x\}, v_x) = X \in \mathcal{B}_\mathcal{A} = X_1$. Now, let $X_2 = \Delta(X_1, v_x)$. We know that $X_1 \subseteq X_2$ as $x \in X_1$ and $\Delta(\{x\}, v_x \subseteq \Delta(X_1, v_x))$ by construction of $\mathcal{B}_\mathcal{A}$. If $X_1 \subsetneq X_2$, then we apply v_x again. As $\Delta(\{x\}, v_x^i) = X_i$ is increasing with i and $|\Delta(\{x\}, v_x^i)| \leq n$ for all i, we will reach a fix point X_n, such that $X_n = \Delta(X_n, v_x)$. In particular, $\Delta(\{x\}, v_x^{n+1}) = \Delta(X, v_x^n) = X_{n+1} = X_n$. As X is in the BSCC of $\mathcal{B}_\mathcal{A}$, there is an observation v with $\Delta(X_n, v) = X$. Let $w_x = v_x^{n+1} v$. Thus, $\Delta(\{x\}, w_x) = \Delta(X, w_x) = X$. Let $w_x = v_x^{n+1} v$. Thus,

$$\Delta(\{x\}, w_x) = \Delta(X, w_x) = X \, (*)$$

Now, by induction on the size of X, we build a uniform word w such that $\Delta(\{x\}, w) = X$ for all $x \in X$. Let $x_1, \ldots, x_k$ be the elements of X. The word w starts with w_{x_1}. We have

that for all $i \leq k$, $\Delta(\{x_i\}, w_{x_1}) \subseteq X$. Let $y_2 \in \Delta(\{x_2\}, w_{x_1})$. Hence $y_2 \in X$, and we will append to w_{x_1} the observation w_{y_2}, obtaining $\Delta(\{x_1\}, w_{x_1}w_{y_2}) = \Delta(\{x_2\}, w_{x_1}w_{y_2}) = X$, and for all $i \leq k$, $\Delta(\{x_i\}, w_{x_1}w_{y_2}) \subseteq X$ (by $(*)$). By induction, we will obtain the desired word w. Then, for ℓ the size of w, we will have $M_X^\ell(x, y) > 0$ for all $x, y \in X$. That is, M_X is irreducible and aperiodic.

We denote by σ_X the stationary distribution of M_X. Let W_X the (possibly countable infinite) set of words which brings from belief X to belief X without seeing belief X in-between. Consider $\sigma_{y,i}$ the distribution over X such that $\sigma_{y,i}(x) = \sum_{w \in W_X^i} P(w)M(y, w, x)$, the probability of reaching x from y after seeing i words of W_X. We now apply the Fundamental theorem of Markov chains (see Theorem 2.2) to the irreducible and aperiodic Markov chain M_X: for $\sigma_{y,i}^X$ the distribution with $\sigma_{y,i}^X(x) = M_X^i(y, x)$, we have that $lim_{i\to\infty}\sigma_{y,i}^X$ exists and is unique, it does not depend upon $y \in X$, and it is equal to σ_X. Now, it suffices to notice that by definition of M_X, we have $\sigma_{y,i}^X = \sigma_{y,i}$. □

4.4 Limit-sure Classifiability

We start by formally defining the problem of *limit-sure classification*:

Definition 4.11 (Limit-sure classifiability)**.** *Two LMCs $\mathcal{A}_1, \mathcal{A}_2$ are limit-sure classifiable iff there exists a computable function $f : \Sigma^* \to \{\bot, 1, 2\}$ such that $P(\rho$ run of $\mathcal{A}_1$ of size $k \mid f(obs(\rho)) \neq 1) \to_{k\to\infty} 0$, and similarly for ρ run of $\mathcal{A}_2$*

Unlike sure and almost-sure classifiability, *limit-sure classifiability* cannot be as easily expressed in terms of languages. Indeed, it is possible to limit-surely classify among $\mathcal{A}_1, \mathcal{A}_2$, and yet $L(\mathcal{A}_1) = L(\mathcal{A}_2)$ (*i.e.*, in the sense the non-stochastic languages). Also, a limit-sure classifier can use statistics over letters in $w \in \Sigma^\omega$ in order to make its decision, which opens a lot of possibilities.

Example 4.5. *Let us illustrate this: consider again $\mathcal{A}_1, \mathcal{A}_2$, where both are the LMC $\mathcal{A}$ from figure 4.5, where $\mathcal{A}_1$ starts from x and $\mathcal{A}_2$ starts from z. Again, if the observation starts with b, then it is easy to conclude that the LMC is $\mathcal{A}_2$. If it starts with a, then the set of states which can be reached after observation a is $\{x, y\}$ in $\mathcal{A}_1$ and $\{z\}$ in $\mathcal{A}_2$, which are both in the BSCCs. Actually, after an even number of b's (and any number of a's), we still have $\{x, y\}$ the set of states possible in $\mathcal{A}_1$ and $\{z\}$ in $\mathcal{A}_2$. In the following section using stationary distributions on LMCs, we will show how to compute that if the LMC is $\mathcal{A}_1$, after an even number of b's, the long term average is $\frac{3}{5}$ to be in x and $\frac{2}{5}$ to be in y. From this, we deduce that the long term average is $\frac{4}{5} = \frac{3}{5}1 + \frac{2}{5}\frac{1}{2}$ to perform an*

a after an even number of b's. On the other hand, if the LMC is $\mathcal{A}_2$*, then the state is* z *and we obtain the average frequency over the observation will tend towards the long term average by law of large numbers. Thus the classifier* $f(w) = 1$*, if the average frequency of a's after an even number of b's observed in* w *is closer to* $\frac{4}{5}$ *than to* $\frac{1}{2}$*, is limit-sure. Notice that using the standard stationary distributions on* Markov chains *as in [KH18] only tells us that both* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *stay in long term average frequency* $\frac{3}{7}$ *in* x*,* $\frac{2}{7}$ *in* y*, and* $\frac{2}{7}$ *in* z *, and thus do* $\frac{5}{7} = \frac{3}{7} + \frac{2}{7}\frac{1}{2} + \frac{2}{7}\frac{1}{2}$ *of a's in average, which cannot limit-surely classify between* $\mathcal{A}_1, \mathcal{A}_2$*.*

Consider the *Maximum A Posteriori Probability (MAP)* classifier [Ram07; KH18]. Remember it answers 1 if $P^{\mathcal{A}_1}(w) > P^{\mathcal{A}_2}(w)$, and 2 otherwise. To compute these two probabilities, it just needs to record for every state s_1 of $\mathcal{A}_1$ (resp. every state s_2 of $\mathcal{A}_2$) the probability to observe w and finish in state s_1 (resp. s_2). It can also give its confidence level about its decision: there is probability confidence(i, w)= $\frac{P^{\mathcal{A}_i}(w)}{P^{\mathcal{A}_1}(w)+P^{\mathcal{A}_2}(w)}$ that decision i is correct after observing w. Notice that confidence is not necessarily non-decreasing as $|w|$ increases, and that the answer of a classifier can also switch from one answer to the other answer. We will show in section 4.4.2 that if $(\mathcal{A}_1, \mathcal{A}_2)$ is limit-sure classifiable, then the MAP classifier will be a limit-sure classifier. The main problem is to decide when limit sure classification holds. This problem can actually be solved in PTIME. The rest of this section is dedicated to proving this property.

4.4.1 The Twin Automaton and the Twin Belief Automaton

Given LMCs $\mathcal{A}_1, \mathcal{A}_2$, we define their *twin automaton* $\mathcal{A} = (S = S_1 \times S_2, \Delta, s_0)$ as the product of the automata associated with $\mathcal{A}_1 \times \mathcal{A}_2$ by forgetting the probabilities. Notice

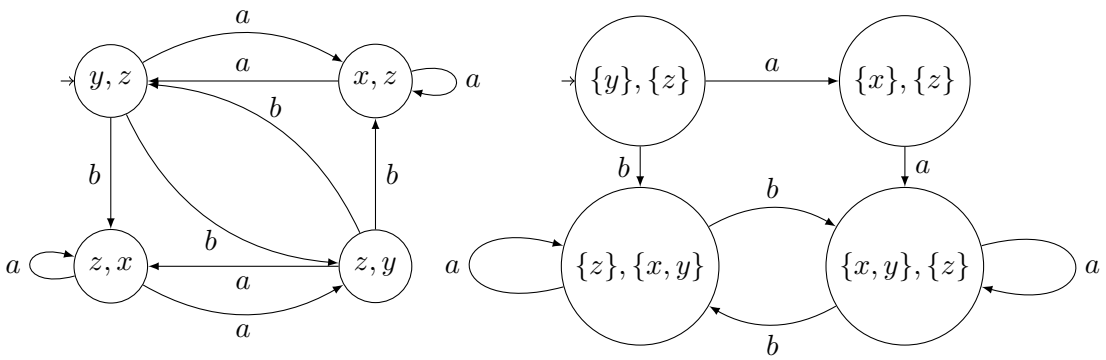

Figure 4.7: Twin automaton (on the left) and twin-belief automaton (on the right), for $\mathcal{A}_1, \mathcal{A}_2$ starting in states y and z

that this notion is close to the *twin plant* (definition 3.7) defined for diagnosis where we considered the product of the unprobabilized LMC and its correct states. The transition relation is $\Delta = \{((s_1, s_2), a, (t_1, t_2)) \mid \delta_1(s_1, a, t_1) > 0, \delta_2(s_2, a, t_2) > 0\}$, with initial state $s_0 = (s_0^1, s_0^2)$. We call states of $\mathcal{A}$ *twin states* and we have $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$ (*i.e.*, non-stochastic languages). In the following, we will often consider the belief automata $\mathcal{B}_\mathcal{A}, \mathcal{B}_{\mathcal{A}_1}, \mathcal{B}_{\mathcal{A}_2}$ associated with $\mathcal{A}, \mathcal{A}_1, \mathcal{A}_2$, obtained by the subset construction (see section 4.3). States of $\mathcal{B}_\mathcal{A}$ will be called *twin beliefs*. Notice that although twin beliefs are formally sets of pairs of states in $2^{S_1 \times S_2}$, we can also present them as pairs of sets of states $2^{S_1} \times 2^{S_2}$ because if (s_1, s_2) and (s_1', s_2') are in the same twin belief, then we also have (s_1, s_2') and (s_1', s_2) in this twin belief. We will thus write the twin belief $X(u)$ associated with observation u as $X(u) = (X_1(u), X_2(u))$, with $X_1(u), X_2(u)$ the beliefs states of $\mathcal{B}_{\mathcal{A}_1}, \mathcal{B}_{\mathcal{A}_2}$ associated with u. Figure 4.7 presents an example with a twin automaton and the twin belief automaton for two copies of the LMC given in figure 4.5, one starting in state y and the other starting in state z.

Lemma 4.15 (Sufficient condition for non-classifiability: Proposition 18 in [CK14])**.** *Let (X_1', X_2') be a reachable twin belief of $\mathcal{B}_\mathcal{A}$. Let $X_1 \subseteq X_1', X_2 \subseteq X_2'$. Let σ_1, σ_2 two distributions over X_1, X_2 such that $(\mathcal{A}_1, \sigma_1) \equiv (\mathcal{A}_2, \sigma_2)$. Then one cannot classify with probability 1 observations from $\mathcal{A}_1$ and $\mathcal{A}_2$.*

Proof. Let u be a word with $B_{\mathcal{A}_1}(u) = X_1'$ and $B_{\mathcal{A}_2}(u) = X_2'$. Hence $P^{\mathcal{A}_1}(u) > 0$ and $P^{\mathcal{A}_2}(u) > 0$. Let $p = \min(P^{\mathcal{A}_1}(u), P^{\mathcal{A}_2}(u)) > 0$. For all $x_1 \in X_1$, let $p_1(x_1) > 0$ be the probability to reach x_1 conditionally to read u. In the same way, we define $p_2(x_2)$ for all $x_2 \in X_2$. We also denote $P(w) = P_{\sigma_1}^{\mathcal{A}_1}(w) = P_{\sigma_2}^{\mathcal{A}_2}(w)$.

Let $\alpha_1 = \min_{x_1 \in X_1} \frac{p_1(x_1)}{\sigma_1(x_1)}$ and similarly for α_2. Let $\alpha = \min(\alpha_1, \alpha_2)$. Now, for any observation w, we have $P^{\mathcal{A}_1}(uw) \geq P^{\mathcal{A}_1}(u) \cdot \alpha P_{\sigma_1}^{\mathcal{A}_1}(w)$, and $P^{\mathcal{A}_2}(uw) \geq P^{\mathcal{A}_2}(u) \cdot \alpha P_{\sigma_2}^{\mathcal{A}_2}(w)$.

Assume by contradiction that there exists a limit-sure classifier f. Let k be a length of observation such that $P(w|f(w) = \bot) < \varepsilon$. Let $R_1 = \{w \in \Sigma^k \mid f(uw) = 1\}$ and $R_2 = \{w \in \Sigma^k \mid f(uw) = 2\}$. We have $\sum_{w \in R_1} P(w) + \sum_{w \in R_2} P(w) \geq 1 - \varepsilon$. Assume for instance that $\sum_{w \in R_1} P(w) \geq \frac{1-\varepsilon}{2}$ (the other case is symmetric). The probability of misclassification for size $|u|+k$ is thus at least $\sum_{w \in R_1} P^{\mathcal{A}_2}(uw) \geq \alpha p \sum_{w \in R_1} P(w) \geq \frac{1-\varepsilon}{2} \alpha p$. This lower bound does not depend upon k, and then does not decrease to 0 when k goes to ∞. □

4.4.2 Characterization of classifiability

Our goal is to use Theorem 4.14 to obtain stationary distributions over beliefs of $\mathcal{A}_1, \mathcal{A}_2$, and classify between them by comparing the stochastic language wrt these stationary distributions using probabilistic equivalence (decidable in polynomial time, see Proposition 4.2). In order to do this, we first need to compare the same information in both LMCs. The idea is to consider twin beliefs from each LMC: we will enrich $\mathcal{A}_1$ with the beliefs of $\mathcal{A}_2$, and vice versa. Let $\mathcal{A}'_1$ be the LMC where the state space is $S_1 \times 2^{S_2}$, and the transition matrix is $M_{\mathcal{A}'_1}((x, Y), a, (x', Y')) = M_{\mathcal{A}_1}(x, a, x')$ if $Y' = \{y' \mid (y, a, y'), y \in Y\}$, and 0 otherwise, for all x, Y, a, x', Y'. We define similarly $\mathcal{A}'_2$ with set of states $S_2 \times 2^{S_1}$. It is easy to see that for any observation w, the belief state $\mathcal{B}_{\mathcal{A}'_1}(w) = \{(x_1, \mathcal{B}_{\mathcal{A}_2}(w)) \mid x_1 \in \mathcal{B}_{\mathcal{A}_1}(w)\}$ is isomorphic to the twin belief $(\mathcal{B}_{\mathcal{A}_1}(w), \mathcal{B}_{\mathcal{A}_2}(w))$, isomorphic to $\mathcal{B}_{\mathcal{A}'_2}(w)$, and we will abuse notation and represent beliefs of $\mathcal{A}'_1$ and $\mathcal{A}'_2$ as twin belief (X_1, X_2), where X_1 or X_2 can be empty.

We are interested in what happens after a BSCC of $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$ is reached. We thus consider twin beliefs reachable from some (x_1, x_2) in the BSCCs of $\mathcal{A}$. The set of twin beliefs reachable in $\mathcal{A}'_1$ and in $\mathcal{A}'_2$ from $(\{x_1\}, \{x_2\})$ are almost the same, except for twin beliefs of the form $(X_1, \emptyset)$ which cannot be reached in $\mathcal{A}'_2$, and of the form $(\emptyset, X_2)$ which cannot be reached in $\mathcal{A}'_1$.

Definition 4.12 (Oblivious twin belief)**.**

We say that a twin belief (X_1, X_2) *is* oblivious *if* $L(\mathcal{B}^{X_1}_{\mathcal{A}_1}) = L(\mathcal{B}^{X_1}_{\mathcal{A}_2})$*, i.e., the languages of* $\mathcal{B}_{\mathcal{A}_1}$ *from* X_1 *and of* $\mathcal{B}_{\mathcal{A}_2}$ *from* X_2 *are the same.*

Let $X = (X_1, X_2)$ be oblivious such that (X_1, X_2) is reachable from some $(\{x_1\}, \{x_2\})$ with $(x_1, x_2) \in \mathcal{B}_{\mathcal{A}}$ in some BSCC of $\mathcal{A}$. The twin beliefs reachable from (X_1, X_2) are the same in $\mathcal{A}'_1$ and $\mathcal{A}'_2$. By definition, if (X_1, X_2) is not oblivious, there exist words differentiating X_1 and X_2, *i.e.*, that belongs only to one of the two languages. We focus next on oblivious pairs.

Let $E_{\mathcal{A}}$ be the union of states BSCCs of twin beliefs accessible from twin states in some BSCC D of $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$, as in definition 4.10. Let $X \in E_{\mathcal{A}}$ and assume X is oblivious. In this case, we say that X is in the BSCCs of twin-beliefs. We define $\sigma^1_X : X_1 \to [0, 1]$ the *stationary distribution* in $\mathcal{A}'_1$ around the twin belief X (formally, σ^1_X is defined on (x, X_2) for all $x \in X_1$, and we omit the second component X_2 because it is constant). In the same way, we define $\sigma^2_X : X_2 \to [0, 1]$ for the second component X_2 around the twin belief X. We can then look for words distinguishing from $\mathcal{A}_1$ and $\mathcal{A}_2$, i.e. with different likelihoods from σ^1_X and from σ^2_X. We can now state the main characterization of classifiability:

Theorem 4.16. *The following are equivalent:*

1. *One cannot limit-surely classify between* $\mathcal{A}_1, \mathcal{A}_2$,
2. *There exists an oblivious* $X \in E_{\mathcal{A}}$ *in a BSCC of twin beliefs such that* $(\mathcal{A}_1, \sigma_X^1) \equiv (\mathcal{A}_2, \sigma_X^2)$,
3. *There exists a BSCC* D *of* $\mathcal{A}$ *and* $X_1 \subseteq S_1, X_2 \subseteq S_2$, *and* $y_1 \in X_1$, $y_2 \in X_2$, *such that* $(y_1, x_2) \in D$ *for all* $x_2 \in X_2$ *and* $(x_1, y_2) \in D$ *for all* $x_1 \in X_1$, *and two distributions* σ^1 *over* X_1 *and* σ^2 *over* X_2 *such that* $(\mathcal{A}_1, \sigma^1) \equiv (\mathcal{A}_2, \sigma^2)$.

Recall that when we say classify, in this section, we mean limit-sure classification. The second condition is useful to show that MAP is limit-sure. However, checking it explicitly is not algorithmically efficient, as building the belief automaton is exponential. To obtain a PTIME algorithm to check limit-sure classifiability, we will use the third condition.

For comparison, in [CK14], a variant of (1) $\Leftrightarrow$ (3) is shown, without using the stationary distributions σ_X^1, σ_X^2 of (2) (see Proposition 4.8).

The rest of this section is devoted to the proof of this theorem. We first observe that 2 implies 3 is easy. Indeed, consider any twin-belief $X = (X_1, X_2) \in E_{\mathcal{A}}$: we have that each pair $(x_1, x_2) \in X = (X_1, X_2)$ belongs to the same BSCC D of $\mathcal{A}$. Remember also that by construction all states (x_1, x_2) in a belief (X_1, X_2) of $E_{\mathcal{A}}$ is in a BSCC of $\mathcal{A}$ (see definition 4.10). Thus, we can take any $y_1 \in X_1, y_2 \in X_2$ and $\sigma^1 = \sigma_X^1$ and $\sigma^2 = \sigma_X^2$, which gives us the proof. In the remaining of the subsection, we prove the two remaining implications. We start by showing $1 \Rightarrow 2$. Then we show $3 \Rightarrow 1$, completing the proof.

($1 \implies 2$): MAP is a limit-sure classifier when condition 2 is false

To prove 1 implies 2, we prove that negation of 2 implies that the MAP classifier (introduced in section 4.4) is limit-sure, which of course implies that 1 cannot hold. Intuitively, (not 2) means that every pair of accessible beliefs X has a distinguishing word w. It then suffices to consider the frequency with which w occurs from X. If this belief occurs an arbitrarily large number of times, we can deduce with arbitrarily high probability the originating LMC by comparing the observed frequency with the theoretical frequencies.

Let $\varepsilon > 0$. Intuitively, when the observation u is long enough, the MAP classifier can claim that the observation comes from one LMC with probability at least $1 - \varepsilon$. Long enough means that we can decompose u into $u = u_1u_2u_3$, with some properties on subwords on u_1, u_2, u_3: intuitively, segments u_1, u_2, u_3 are such that there is a high probability to reach a BSCC of the twin automaton with u_1, then to reach a BSCC of

the twin *belief* automaton after u_2, and u_3 allows with high probability to eliminate one of the two possible LMCs.

We now formalize this decomposition into u_1, u_2, u_3. Let u be an observation from a run of $\mathcal{A}_1$. We denote by $p_1(s,u)$ (resp. $p_2(t,u)$) the probability in $\mathcal{A}_1$ to observe u and reach state s (resp. $\mathcal{A}_2$ and state t). Let $\varepsilon > 0$. Then $u = u_1u_2u_3$ is a *good decomposition* if the following conditions hold:

- u_1 is such that there exists R_1, R_2 sets of states of $\mathcal{A}_1, \mathcal{A}_2$ with:
  1. (s,t) is in a BSCC of $\mathcal{A}$ for all $(s,t) \in R_1 \times R_2$,
  2. $\sum_{s \notin R_1} p_1(s,u_1) < \epsilon$,
  3. $\sum_{t \notin R_2} p_2(t,u_1) < \epsilon^2 \min_{s \in R_1} p_1(s,u_1)$.
- u_2 is such that for all $(s,t) \in R_1 \times R_2$, the twin-belief $X_{s,t} = (X_s, X_t)$ reached by reading u_2 from (s,t) is in the BSCC of the twin-belief automaton. It is easy to see that eventually with probability 1, one will observe such a u_2.
- Last, we tackle the condition on u_3. If $X_{s,t}$ is oblivious, let $\sigma^1_{s,t}, \sigma^2_{s,t}$ be the stationary distributions built for $X_{s,t}$. By hypothesis (not 2), there exists $w_{s,t}$ such that $P^{\mathcal{A}_1}_{\sigma^1_{s,t}}(w_{s,t}) \neq P^{\mathcal{A}_2}_{\sigma^2_{s,t}}(w_{s,t})$. Let $\alpha(s,t) = |P^{\mathcal{A}_1}_{\sigma^1_{s,t}}(w_{s,t}) - P^{\mathcal{A}_2}_{\sigma^2_{s,t}}(w_{s,t})|$. From any state of X_s, denoting by $n_{s,t}(u_3)$ the number of times $X_{s,t}$ has been a twin-belief along u_3, and $n'_{s,t}(u_3)$ the number of times $w_{s,t}$ has been observed from $X_{s,t}$, by the central limit theorem, we have that $\frac{n'_{s,t}(u_3)}{n_{s,t}(u_3)}$ tends towards $P^{\mathcal{A}_1}_{\sigma^1_{s,t}}(w_{s,t}) \neq P^{\mathcal{A}_2}_{\sigma^2_{s,t}}(w_{s,t})$ with probability 1. We consider observations u_3 in $L(\mathcal{B}_{\mathcal{A}_1}, X_s) = L(\mathcal{B}_{\mathcal{A}_2}, X_t)$ such that:
  - $\frac{n'_{s,t}(u_3)}{n_{s,t}(u_3)}$ is in $[P^{\mathcal{A}_1}_{\sigma^1_{s,t}}(w_{s,t}) - \alpha(s,t)/4\,,\ P^{\mathcal{A}_1}_{\sigma^1_{s,t}}(w_{s,t}) + \alpha(s,t)/4]$.

Let $W_k(\varepsilon)$ be the set of observations $u_1u_2u_3$ of size k which are good decompositions. Then,

Lemma 4.17. *For all $\varepsilon' > 0$, for k large enough, we have $P^{\mathcal{A}_1}(\rho \mid obs(\rho) \in W_k(\varepsilon)) > 1 - \varepsilon'$.*

Proof. As runs converge towards BSCCs, eventually with probability 1, observation u_1 satisfies the first two conditions. For the last one, consider some u_1 satisfying the first two conditions. Then let $p_1(u_1) = \min_{s \in S_1} p_1(s,u_1)$. Considering extensions $u_1u'_1$ of u_1, one gets $p_1(u_1u'_1) > p_1(u_1)/n$ because states in BSCCs can only reach states in BSCCs. The worst case is when these runs are split into several ending states, and there are at most n states. Eventually with probability 1, one observes $u_1u'_1$ such that $\sum_{t \notin R_2} p_2(t, u_1u'_1) <$

$\epsilon^2 p_1(s,u_1)/n$, because $p(s,u_1)$ is constant when u_1' grows longer. Then u_1u_1' satisfies all the conditions.

Let W_k be the set of observations u_3 in $L(\mathcal{B}_{\mathcal{A}_1},X_s)=L(\mathcal{B}_{\mathcal{A}_2},X_t)$ of size k satisfying the condition of u_3. We have that $q_1(k)=\sum_{w\in W_k} p_1(s,u_1)P_s^{\mathcal{A}_1}(u_2u_3)\to p_1(s,u_1)P_s^{\mathcal{A}_1}(u_2)=q_1$, and that $q_2(k)=\sum_{w\in W_k} p_2(t,u_1)\cdot P_t^{\mathcal{A}_2}(u_2u_3)\to 0$ when k tends to ∞. Let $k_{s,t}$ such that $q_1(k_{s,t})>q_1-\varepsilon$ and $q_2(k_{s,t})<q_1\varepsilon^2$.

If (X_s,X_t) is not oblivious, then there is a word $w_{s,t}\in L_{X_s}^{\mathcal{B}_{\mathcal{A}_1}}\setminus L_{X_t}^{\mathcal{B}_{\mathcal{A}_2}}$, or a word $w_{s,t}\in L_{X_t}^{\mathcal{B}_{\mathcal{A}_2}}\setminus L_{X_s}^{\mathcal{B}_{\mathcal{A}_1}}$. In both case we have $P^{\mathcal{A}_1}_{\sigma^1_{s,t}}(w_{s,t})\neq P^{\mathcal{A}_2}_{\sigma^2_{s,t}}(w_{s,t})$, and we proceed as in the oblivious case. Trivially, eventually, $|u_3|>k_{s,t}$ for all $(s,t)\in R_1\times R_2$. □

Using Lemma 4.17, we can show that the MAP classifier is indeed limit-sure if 2 does not hold.

Proposition 4.18. *Assume point* 2 *of theorem 4.16 does not hold. Then for all $\varepsilon'>0$, there exists k' such that for all $k\geq k'$, $P^{\mathcal{A}_1}(u\in\Sigma^k\mid P^{\mathcal{A}_2}(u)>P^{\mathcal{A}_1}(u))\leq\varepsilon'$, and similarly $P^{\mathcal{A}_2}(u\in\Sigma^k\mid P^{\mathcal{A}_2}(u)<P^{\mathcal{A}_1}(u))\leq\varepsilon'$.*

Proof. With high probability, $obs(\rho)\in W_k(\varepsilon)$ for k large enough. Let us consider runs of $\mathcal{A}_1$ with observation in $W_k(\varepsilon)$ depending on the state s reached after observation u_1. With probability at most ε, s is not in R_1. Hence with high probability, s is in R_1. We want to show that for almost all observations of $\mathcal{A}_1$, $P^{\mathcal{A}_2}(u_1u_2u_3)<p_1(s,u_1)\cdot P_s^{\mathcal{A}_1}(u_2u_3)\leq P^{\mathcal{A}_1}(u_1u_2u_3)$, that is $MAP(u_1u_2u_3)=1$. We decompose $P^{\mathcal{A}_2}(u_1u_2u_3)=\sum_{t\in S_2}p_2(t,u_1)\cdot P_t^{\mathcal{A}_2}(u_2u_3)$.

Fix a u_1 such that there exists u_2,u_3 with $u_1u_2u_3\in W_k(\varepsilon)$. First, we show that with high probability, $\sum_{t\notin R_2}p_2(t,u_1)\cdot P_t^{\mathcal{A}_2}(u_2u_3)$ is negligible wrt $p_1(s,u_1)\cdot P_s^{\mathcal{A}_1}(u_2u_3)$. For that, consider the set of observation such that it is not the case: $W_{S_2\setminus R_2}=\{u_1u_2u_3\in W_k(\varepsilon)\mid\sum_{t\notin R_2}p_2(t,u_1)\cdot P_t^{\mathcal{A}_2}(u_2u_3)>\varepsilon p_1(s,u_1)\cdot P_s^{\mathcal{A}_1}(u_2u_3)\}$. We prove that this happens with arbitrarily small probability: $P^{\mathcal{A}_1}(W_{S_2\setminus R_2})\leq\varepsilon$. Else, by contradiction, we would have $P^{\mathcal{A}_1}(W_{S_2\setminus R_2})>\varepsilon$, which by definition of $W_{S_2\setminus R_2}$ implies that $P^{\mathcal{A}_2}(u_1u_2u_3\in W_{S_2\setminus R_2}\mid u_1\text{ reaches }t\notin R_2)>\varepsilon P^{\mathcal{A}_1}(u_1u_2u_3\in W_{S_2\setminus R_2}\mid u_1\text{ reaches }s)>\varepsilon^2p_1(s,u_1)$. Thus, $\sum_{t\notin R_2}p_2(t,u_1)\geq P^{\mathcal{A}_2}(u_1u_2u_3\in W_{S_2\setminus R_2}\mid u_1\text{ reaches }t\notin R_2)>\varepsilon^2p_1(s,u_1)$, a contradiction with the definition of $W_k(\varepsilon)$.

We can now focus on $t\in R_2$: fix a u_2 such that there is a u_3 with $u_1u_2u_3\in W_k$. For all $t\in R_2$, consider the word $w_{s,t}$. We now show that with high probability, $p_2(t,u_1)\cdot P_t^{\mathcal{A}_2}(u_2u_3)$ is negligible wrt $p_1(s,u_1)\cdot P_s^{\mathcal{A}_1}(u_2u_3)$. For that, we consider the set of observations such that it is not the case: $W_k'=\{u_1u_2u_3\in W_k\mid p_2(t,u_1)\cdot P_t^{\mathcal{A}_2}(u_2u_3)>\varepsilon\cdot p_1(s,u_1)\cdot P_s^{\mathcal{A}_1}(u_2u_3)\}$. Let $q_1'=\sum_{u_1u_2u_3\in W_k'}p_1(s,u_1)\cdot P_s^{\mathcal{A}_1}(u_2u_3)$ and

$q'_2 = \sum_{u_1u_2u_3 \in W'_k} p_2(t, u_1) \cdot P_t^{\mathcal{A}_2}(u_2u_3)$. We have $q'_1 \leq p_1(s, u_1) \cdot P_s^{\mathcal{A}_1}(u_2) \cdot \varepsilon$. Indeed, by contradiction, if $q'_1 > p_1(s, u_1) \cdot P_s^{\mathcal{A}_1}(u_2) \cdot \varepsilon$, then $q'_2 > p_1(s, u_1) \cdot P_s^{\mathcal{A}_1}(u_2) \cdot \varepsilon^2$, a contradiction with $q'_2 \leq q_2(k) \leq p_1(s, u_1) \cdot P_s^{\mathcal{A}_1}(u_2) \cdot \varepsilon^2$. Hence, with probability at least $p_1(s, u_1)P_s^{\mathcal{A}_1}(u_2) - 2\varepsilon$, observation $u_1u_2u_3$ is in $W_k \setminus W'_k$, and it satisfies $P_t^{\mathcal{A}_2}(u_2u_3) \leq \varepsilon \cdot P_s^{\mathcal{A}_1}(u_2u_3)$. With probability at least $p_1(s, u_1)P_s^{\mathcal{A}_1}(u_2)(1 - 2m\varepsilon)$, this is true for all t. It remains to sum over all u_1, u_2 and states s to obtain probability at least $1 - 2m\varepsilon$ to have $P^{\mathcal{A}_2}(u_1u_2u_3) \leq \varepsilon + \sum_{t \in R_2} p_2(t, u_1) \cdot P_t^{\mathcal{A}_2}(u_2u_3) \leq \varepsilon + m\varepsilon P^{\mathcal{A}_1}(u_1u_2u_3) \leq P^{\mathcal{A}_1}(u_1u_2u_3)$ for ε small enough. This implies that $\mathrm{MAP}(u_1u_2u_3) = 2$ with probability at most $2\varepsilon + 2\varepsilon \cdot m \leq \varepsilon'$ for ϵ small enough. □

$(3 \implies 1)$: Language equivalence implies non-classifiability

Let D a BSCC of $\mathcal{A}$, $X_1, X_2, \sigma^1, \sigma^2$ as in the hypothesis of 3. We write $X_1 = \{i_1, \dots i_n\}$ and $X_2 = \{j_1, \dots j_m\}$. We let $i_1 = y_1$ and $j_1 = y_2$. If there exists an observation w such that $X_1 \subseteq B_{\mathcal{A}_1}(w)$ and $X_2 \subseteq B_{\mathcal{A}_2}(w)$ then lemma 4.15 implies that one cannot classify between $\mathcal{A}_1, \mathcal{A}_2$. However, there are cases where such an observation w does not exist. Recall that lemma 4.15 is only a sufficient condition. Instead, we will show that one has probabilistic equivalence of languages from y_1, y_2 after reading some observation u. As (y_1, y_2) can be reached in $\mathcal{A}$, we can conclude on the non-classifiability using lemma 4.15. We first show that every twin belief in the BSCC E_D is oblivious.

Proposition 4.19. *Let (H_1, H_2) be a twin belief in the BSCC E_D. Then (H_1, H_2) is oblivious.*

Proof. Let u be an observation. Let $B_k(u)$ be the belief of $\mathcal{A}_1$ reached by u from $\{i_k\}$, and $C_k(u)$ be the belief of $\mathcal{A}_2$ reached by u from $\{j_k\}$. We define $Z_1(u)$ the set of beliefs $B_l(u), l \leq n$ and $Z_2(u)$ the set of beliefs $C_l(u), l \leq m$. Notice that the sizes $|Z_1(u)|$ and $|Z_2(u)|$ (the number of non empty beliefs) are non increasing with u.

First, assume that there is a word u possible from H_1 in $\mathcal{B}_{\mathcal{A}_1}$ but not possible from H_2 in $\mathcal{B}_{\mathcal{A}_2}$. Consider j_1. As $(y_1, j_1) \in D$, by lemma 4.13, there is some u_1 with $B_1(u_1) = H_1$ and $C_1(u_1) = H_2$. And hence, $B_1(u_1u) \neq \emptyset$ and $C_1(u_1u) = \emptyset$. Hence, $|Z_2(u_1)| \leq m-1$. Consider j_2 and $Z_2(u_1u)$. Assume that $C_2(u_1u) \neq \emptyset$. Thus, there exists u_2 with $B_1(u_1uu_2) = H_1$ and $C_2(u_1uu_2) = H_2$. Thus $B_1(u_1uu_2u) \neq \emptyset$ and $C_2(u_1uu_2u) = \emptyset$. Otherwise, we already have $B_1(u_1u) \neq \emptyset$, and $C_2(u_1u) = \emptyset$. Either way, $|Z_2| \leq m - 2$. By induction, we can find an observation w with $Z_2(w) = \emptyset$ and $B_1(w) \in Z_1(w) \neq \emptyset$, a contradiction, as $0 < P_{\sigma_1}(w) = P_{\sigma_2}(w) = 0$.

The case w possible from X_2 but not from X_1 is symmetric, using y_1 and C_1 as the non-empty set. □

It is not necessarily the case that we can reach the BSCC E_D of twin beliefs in a uniform way over all $(x_1, x_2) \in D$. Let $(H_1, H_2) \in E_D$. In the following, we will consider observations that reach the BSCC of E_D from u. Let u_1 such that $B_1(u_1) = H_1$ and $C_1(u_1) = H_2$. Such u_1 exists by lemma 4.13. Let V be the language from H_1, which is equal to the language from H_2. Now, consider what happens from i_2 reading observations in V. There are several cases. First, assume that there is an observation v_2 in V such that a belief state in the BSCC of beliefs is reached from $\{i_2\}$ reading u_1v_2. That is, $(B_2(u_1v_2), C_1(u_1v_2)) \in E_D$. Now, compare the language from $(B_2(uv)$ in $\mathcal{A}_1$ and from $C_1(u_1v_2))$ in $\mathcal{A}_2$. If it is the same language, we say that i_2 is of type 1. Otherwise, or if there is no observation $v_2 \in V$ such that the BSCC of beliefs can be reached reading u_1v_2, then we say that i_2 is of type 2. Intuitively, a state of type 2 will be negligible when following y_1, y_2, whereas a state of type 1 needs to be tracked because it is not negligible. We then consider the state i_3 and the belief $B_3(u_1v_2)$, and classify each state $i_3 \dots$ then $j_2 \dots$ inductively into type 1 and type 2. We have an observation w leading all the type 1 state to their BSCC, and all the type 1 states have the same language.

We reorder $X_1 = \{i_1, \dots i_n\}$ and $X_2 = \{j_1, \dots j_m\}$ such that $i_1, \dots i_k$ and $j_1, \dots, j_\ell$ are of type 1 and the rest is of type 2. We now follow every type 1 belief in parallel. Consider a $(k+\ell)$-belief $H = (H_1, \dots, H_k, K_1, \dots, K_\ell)$ in the BSCC of belief states of $\mathcal{A}_1^k \times \mathcal{A}_2^\ell$. Let u an observation such that $B_r(u) = H_r$ for all $r \leq k$ and $C_r(u) = K_r$ for all $r \leq \ell$. Because the language for the type 1 states are the same from their belief state, we can compute $\sigma_r : H_r \to [0,1]$ the stationary distribution for i_r to be around belief H for all $r \leq k$ and $\tau_r : K_r \to [0,1]$ be the stationary distribution over H for all $r \leq \ell$. Let W_H be the set of observations from the $(k+\ell)$-belief H to H without seeing H in-between.

For all w', we have by definition of the equivalence: $\sum_{w \in W_H^\kappa} \sum_{r \leq n} \sigma(i_r) P_{i_r}^{\mathcal{A}_1}(uww') = \sum_{w \in W_H^\kappa} \sum_{r \leq \ell} \tau(j_r) P_{j_r}^{\mathcal{A}_2}(uww')$. Considering the limit when κ tends to infinity, we have for all $r > k$, $\lim_{\kappa \to \infty} \sum_{w \in W_H^\kappa} \alpha_r P_{i_r}^{\mathcal{A}_1}(uw) = 0$. Indeed, consider $i_r, r > k$. For paths reaching a state such that the BSCC of beliefs cannot be reached, the probability to stay out of the BSCC tends to 0 with the size of the run. Otherwise, the path reaching the BSCC of beliefs, *e.g.*, in belief X_r. By definition of type 2 state, the language is not the same as the language of H_1, which is W_H^*. Hence either there is a word in W_H^* which cannot be done from X_r and can be done from H_1, in which case avoiding this word forever have probability 0, or there is a word which can be done from X_r but not from H_1: this word is not in W_H^*, and at each W_H iteration, there is some missing probability from X_r, *e.g.*,

$1-\epsilon$, and eventually the probability is 0. We thus obtain:

$$\forall w', \sum_{r\leq k} \sigma(i_r) P^{\mathcal{A}_1}_{\sigma_r}(w') = \sum_{r\leq \ell} \tau(j_r) P^{\mathcal{A}_2}_{\tau_r}(w')$$

Let $\alpha = \sigma(i_1)$, and $\alpha_r = \sigma(i_r)/(1-\alpha)$ for all $r \leq k$. Let $\tau = \sum_{r\leq \ell} \tau(j_r)\tau_r$, and $\sigma = \sum_{2\leq r\leq k} \alpha_r \sigma_r$. We have $(\mathcal{A}_1, \alpha\sigma_1 + (1-\alpha)\sigma) \equiv (\mathcal{A}_2, \tau)$. We show:

Proposition 4.20. $(\mathcal{A}_1, \sigma_1) \equiv (\mathcal{A}_1, \sigma) \equiv (\mathcal{A}_2, \tau)$.

Proof. Assume by contradiction that it is not the case: That is, there is a w such that $P^{\mathcal{A}_1}_{\sigma_1}(w) > P^{\mathcal{A}_1}_{\sigma}(w)$. Let us write $x = P^{\mathcal{A}_1}_{\sigma_1}(w) = \gamma P^{\mathcal{A}_1}_{\sigma}(w) = \gamma x$, with $\gamma < 1$. We have the following:

$$P^{\mathcal{A}_2}_{\tau}(w) = \alpha P^{\mathcal{A}_1}_{\sigma_1}(w) + (1-\alpha) P^{\mathcal{A}_1}_{\sigma}(w) = \alpha x + (1-\alpha)\gamma x$$

We let W' be the set of minimal observation u sending to X from $(B_1(w), \dots, B_k(w), C_1(w), \dots, C_\ell(w))$. We have that $\sum_{w'\in W' W_H^\kappa} P^{\mathcal{A}_1}_{\sigma}(ww')$ tends towards $P^{\mathcal{A}_1}_{\sigma}(w) \cdot P^{\mathcal{A}_1}_{\sigma}(w')$ as κ tends to infinity, and similarly for σ_1, τ. Hence, $\sum_{w'\in W' W_H^\kappa} P^{\mathcal{A}_2}_{\tau}(ww'w)$ converges towards $P^{\mathcal{A}_2}_{\tau_X}(w)^2$ as κ tends to infinity. Also, for all κ, this is equal with $\sum_{w'\in W' W_H^\kappa} \alpha P^{\mathcal{A}_1}_{\sigma_1}(ww'w) + (1-\alpha) P^{\mathcal{A}_1}_{\sigma}(ww'w)$. Again, this converges towards $\alpha x^2 + (1-\alpha)\gamma^2 x^2$. That is, we have after simplifying by x^2:

$$(\alpha + (1-\alpha)\gamma)^2 = \alpha + (1-\alpha)\gamma^2$$

Now, the function $x \mapsto x^2$ is *strictly* convex (its second derivative is strictly positive). Applying the definition to $(1, \gamma)$ (this is also Jensen's inequality), we obtain a contradiction:

$$(\alpha + (1-\alpha)\gamma)^2 < \alpha + (1-\alpha)\gamma^2$$

□

Once this result is established, we can apply it symmetrically to the second component and obtain $(\mathcal{A}_1, \sigma_1) \equiv (\mathcal{A}_2, \tau_1)$. As $(i_1, j_1) = (y_1, y_2)) \in D$, we can conclude about non-classifiability using lemma 4.13.

4.4.3 A PTIME Algorithm

Theorem 4.16 gives us a characterization for the existence of a limit-sure classifier. The third condition is particularly interesting, because it does not require computing beliefs. The third condition actually implies an efficient algorithm, similar to [CK14], to test in PTIME whether there exists a limit-sure classifier between $\mathcal{A}_1, \mathcal{A}_2$.

Our algorithm, presented in 3, uses linear programming. We let $v_1, \ldots, v_\ell$ be the basis of $Eq(\mathcal{A}_1, \mathcal{A}_2)$. There exist two distributions σ^1, σ^2 over X_1, X_2 with $(\mathcal{A}_1, \sigma_1) \equiv (\mathcal{A}_2, \sigma_2)$ iff the linear system of equations (for all $j \leq \ell$, $(\sigma_1 \quad \sigma_2) \cdot v_j = 0$) has a solution (with σ_1, σ_2 as variables), which can be solved in Polynomial time.

Algorithm 3 Limit-sure Classifiability

```
Compute D_1, ..., D_k the BSCCs of the twin automaton A.
for i=1..k do
    for (y_1, y_2) ∈ D_i do
        Let X_1 = {x_1 | (x_1, y_2) ∈ D_i}, X_2 = {x_2 | (y_1, x_2) ∈ D_i}.
        if there exist two distributions σ_1, σ_2 over X_1, X_2 with σ_1(y_1) > 0 and σ_2(y_2) > 0
            with (A_1, σ_1) ≡ (A_2, σ_2) then
            return not classifiable
        end if
    end for
end for
return classifiable
```

The correctness of the algorithm is immediate from Theorem 4.16, as it checks explicitly for the third condition to hold, in which case it returns not classifiable. If the third condition is false for every BSCC D, then it returns classifiable.

4.4.4 Comparison with Distinguishability between LMCs [KS16]

We complete this section, by comparing our results with a related result on LMCs. In [KS16], the problem of distinguishability between labeled Markov Chains has been considered. First, labeled Markov Chains are just another name for LMCs. The idea behind distinguishability is similar to the idea behind classifiability. Still, there are some technical differences: distinguishability asks that for all $\varepsilon > 0$, there exists a $(1-\varepsilon)$-*classifier*, that is a classifier $f : \Sigma^* \to \{\bot, 1, 2\}$, such that if the classifier answers $f(u) = 1$, then there is probability at least $(1-\varepsilon)$ that the observation comes from a run from $\mathcal{A}_1$, and similarly

for $f(u) = 2$. To compare, limit-sure classifiers need to be uniform over ε (see the next section).

It is not to hard to show that limit-sure classification coincide with the notions of distinguishability and distance 1 as well for LMCs:

Theorem 4.21. *The following are equivalent:*

1. *There exists a limit-sure classifier for* $\mathcal{A}_1, \mathcal{A}_2$,
2. *For all* $\varepsilon > 0$, *there exists a* $(1-\varepsilon)$-classifier *for* $\mathcal{A}_1, \mathcal{A}_2$,
3. $d(\mathcal{A}_1, \mathcal{A}_2) = 1$.

Proof. (1) implies (2) is obvious (the classifier we built provides an $(1-\varepsilon)$-classifier for all ε. (2) implies (3) is done in [KS16].

It remains to show that 3 implies 1: Assume that $d(\mathcal{A}_1, \mathcal{A}_2) = 1$. We will show that the MAP classifier is a limit-sure classifier. Let $\mathrm{mis}(\mathcal{A}_1, \mathcal{A}_2, w)$ be its probability of misclassification. Thus, for all $\varepsilon > 0$, there exists k and $W_k \subset \Sigma^k$ such that $P_1(W_k\Sigma^\omega) \geq 1-\varepsilon$ and $P_2(W_k\Sigma^\omega) \leq \varepsilon$ and we obtain:

$$\sum_{|w|=k} \mathrm{mis}(\mathcal{A}_1, \mathcal{A}_2, w)P(w) = \sum_{w \in W_k} \mathrm{mis}(\mathcal{A}_1, \mathcal{A}_2, w)P(w) + \sum_{w \in \Sigma^k \setminus W_k} \mathrm{mis}(\mathcal{A}_1, \mathcal{A}_2, w)P(w)$$

$$\leq P_2(W_k) + P_1(\Sigma^k \setminus W_k) \leq 2\varepsilon$$

That is, when $k \to \infty$, the probability of misclassification tends towards 0. □

The proofs to obtain the PTIME algorithms are quite different though: we use stationary distributions in LMCs while [CK14] focuses on separating events. Some intermediate results are however related: our Proposition 4.20 is to be compared with Proposition 19 b) of [CK14]. Our statement is stronger as the equivalence is true from *all* pairs of states with the same (non-stochastic) language - and in particular from $(i_1, j_1) = (y_1, y_2)$ (cf Proposition 4.19). Also, the proof of Proposition 4.20 is simple, using strict convexity focusing on *one* finite separating word, while in [CK14], the existence of a maximal separating events (sets of infinite words) is used crucially in the proof of Proposition 19 b).

Surprisingly, our resulting algorithm is very similar to the one in [CK14], whereas we use very different methods. Still, we can restrict the search to *distributions* in a BSCC of twin states, while [CK14] considers *subdistributions* on the whole state space of twin states. This allows us to optimize the number of variables in the Linear Program.

Figure 4.8: Two LMCs $\mathcal{A}_1$ and $\mathcal{A}_2$ with no limit-sure classifier

4.5 Attack-classification

4.5.1 Classification in a security context

While limit-sure classification allows for some misclassification, *i.e.*, , error in classification, it requires that every single pair of executions of the LMCs are classifiable. From a security perspective, if one wants to make sure that two systems cannot be distinguished from each other, then the question changes slightly: from the point of view of an attacker who could exploit the knowledge of which model the system is following, it need not classify every single execution. It only needs to find one execution for which it can decide. This gives rise to what we call *attack-classification*, which amounts to providing the attacker with a reset action she can play when she believes the execution cannot be classified. Then, a new (possibly the same) LMC is taken at random and an execution of this new LMC is observed by the attacker.

We start by considering *limit-sure attack-classifiers*, namely, we require that there exists a *reset-strategy*, which with probability 1, resets only finitely many times, and a limit-sure classifier for the observation after the last reset. We also consider what happens if instead of limit-sure classifier, we ask for the existence of a family of $(1-\varepsilon)$-classifiers after the last reset, one for each ε. The difference is that the reset action can take into account the ε in the latter, but not in the former. While both notions coincide for the classifiers defined in the previous section, we show now that they do not coincide for attack-classification.

Example 4.6. *Figure 4.8 illustrates the difference between these two notions. First, for all $\varepsilon > 0$, there exists an $(1-\varepsilon)$-attack-classifier: given an ε, the reset strategy resets if the first letter b happens within the first $k_\varepsilon = log(\frac{1}{9\varepsilon})$ steps. Otherwise, the observation is $a^{k_\varepsilon}w$, and the classifier claims that the LMC is $\mathcal{A}_1$, which is true with probability at least $(1-\varepsilon)$. However, this reset strategy is not compatible with limit-sure classifier (and, in fact, no reset strategy is), because it is not uniform wrt all ε: once a b has been produced, no more information can be gathered.*

Note that anything that can be limit-sure attack-classified can also be classified with $(1-\varepsilon)$-attack-classifiers for all ε. Thus the former notion of limit-sure attack-classifier is strictly contained in the latter. We now compare the complexities: decide the former is PSPACE-complete, while the latter turns out to be undecidable.

4.5.2 Limit-sure attack-classifiability is PSPACE-complete

Let us first formalize our definition of attack-classification.

Definition 4.13. *We say two LMCs $\mathcal{A}_1, \mathcal{A}_2$ are* limit-sure attack-classifiable *if there exists:*

1. reset strategy $\tau : \Sigma^* \to \{\bot, reset\}$ *telling when to reset, and which eventually stops resetting, with probability 1 on the reset runs, and*
2. *limit-sure classifier for u, where $u \in \Sigma^*$ denotes the suffix of observations since last reset.*

We say that an observation with reset w_1 reset $w_2 \cdots$ reset w_k follows τ, with $w_k \in \Sigma^\omega$, if for all v_i strict prefix of w_i, $\tau(v_i) = \bot$ for all i, and $\tau(w_i) = reset$ for all $i < k$.

In the following, we show an algorithmic characterization for this concept. Intuitively, there needs to exist one execution of one LMC (say $\mathcal{A}_1$), such that no matter the execution of the other LMC with the same observation, we can eventually classify between these two executions. We will thus consider $\mathcal{A}'_1$ and $\mathcal{A}'_2$, the LMCs $\mathcal{A}_1$ and $\mathcal{A}_2$ enriched with the beliefs of the other LMC.

First, we define classifiable twin states in the BSCC of twin states: $(x_1, x_2) \in \mathcal{A}$ is classifiable iff for (X_1, X_2) in the unique BSCC of twin beliefs, either (X_1, X_2) is non-oblivious or (X_1, X_2) is oblivious and $(\mathcal{A}_1, \sigma^1_{X_1,X_2}) \not\equiv (\mathcal{A}_2, \sigma^2_{X_1,X_2})$, for $(\sigma^1_{X_1,X_2}, \sigma^2_{X_1,X_2})$ the stationary distributions around (X_1, X_2). Notice that it does not depends upon the choice of (X_1, X_2). For a belief state X_2 of $\mathcal{A}_2$, we say that $(x_1, X_2) \in \mathcal{A}'_1$ is classifiable if (x_1, x_2) is classifiable for all $x_2 \in X_2$ (in particular, every (x_1, x_2) is in a BSCC of twin states). In particular, $(x_1, \emptyset)$ is classifiable. We define $(x_2, X_1) \in \mathcal{A}'_2$ similarly.

Proposition 4.22. *$(\mathcal{A}_1, \mathcal{A}_2)$ is limit-sure attack-classifiable iff there exists a classifiable $(x_1, X_2) \in \mathcal{A}'_1$, or a classifiable $(x_2, X_1) \in \mathcal{A}'_2$.*

Proof. First, if there exists a classifiable $(x_1, X_2) \in \mathcal{A}'_1$, then let ρ_1 be a path in $\mathcal{A}'_1$ ending in (x_1, X_2). Now, for all $x_2 \in X_2$, consider (x_1, x_2), and let (Y_1, Y_2) be a twin belief in the

BSCC of twin beliefs reachable from (x_1, x_2) by path ρ_2. As (x_1, x_2) is classifiable, there are several cases:

- either there is a word $w_{x_2} \in L_{Y_1}^{B_{\mathcal{A}_1}} \setminus L_{Y_2}^{B_{\mathcal{A}_2}}$, and we consider path ρ_3 labeled by w_{x_2} after $\rho_1\rho_2$ in $\mathcal{A}_1$. It proves that the state cannot be x_2.
- or there is a word $w_{x_2} \in L_{Y_2}^{B_{\mathcal{A}_2}} \setminus L_{Y_1}^{B_{\mathcal{A}_1}}$, and we set $\rho_3 = \varepsilon$,
- otherwise, (Y_1, Y_2) is oblivious, and we also le $\rho_3 = \varepsilon$.

From $\rho_1\rho_2\rho_3$, we define $\rho_4\rho_5$ associated with another x_2, until we took into account every $x_2 \in X_2$. The path $\rho = \rho_1\rho_2\rho_3\rho_4 \cdots \rho_\ell$ has strictly positive probability to happen in $\mathcal{A}'_1$, and thus strictly positive probability to happen in the union of LMCs (remember the run are picked with uniform probability among the LMCs).

Given this path ρ and the associated observation w, the reset strategy is to play $\tau(u) =$ reset if:

1. The observation u of the system since the last reset is of length $|u| < |w|$, and u is not a prefix of w, or
2. otherwise, if there is no extension ρ' of ρ in $\mathcal{A}_1$ such that $\rho\rho'$ is labeled by u,
3. otherwise, if the statistical counts for the proportion of times w_{x_2} is done from (Y_1, Y_2) is closer to the average value av_{Y_2,Y_1} given by $\sigma^2_{Y_1,Y_2}$ than to the average value av_{Y_1,Y_2} given by $\sigma^1_{Y_1,Y_2}$.

The set of infinite paths in the system such that τ resets infinitely often is of probability 0, because to not reset, it suffices to draw $\mathcal{A}_1$, then perform ρ, which happens with strictly positive probability, in which case the first 2 items. The third item can still kicks in, by drawing many biased runs from (Y_1, Y_2), such that the statistic for w_{x_2} goes close to av_{Y_2,Y_1}. Let ℓ the number of times (Y_1, Y_2) is seen. We suppose that $av_{Y_2,Y_1} > av_{Y_1,Y_2}$ (the other case is symmetric). We use a special case of the Cramer's theorem [Cra38]. At every time (Y_1, Y_2) is seen and we are in the automaton $\mathcal{A}_1$, the probability to see w_{x_2} at step i follows a Bernoulli law X_i of parameter av_{Y_1,Y_2}. By denoting $S_\ell = \frac{1}{\ell}\sum_i^n X_i$ and $I(z)$ the Fenchel-Legendre transform of $\log(\mathbb{E}[e^{tX_1}])$, we have by Chernoff's inequality that for $x > av_{Y_1,Y_2}$, $P(S_\ell > x) < e^{-\ell I(x)}$ [SW95]. In particular, this is true for the value $x = av_{Y_1,Y_2} + \frac{av_{Y_1,Y_2} - av_{Y_2,Y_1}}{2}$. We notice that for all ℓ, we have that $P(S_\ell < x | S_{\ell-1} < x) \geq P(S_\ell < x)$ (intuitively, the chance to be lower than the bound after the ℓ-th step is greater if we were already lower at the $\ell - 1$-th step. Then, for all L the probability that for all $\ell \geq L$, $S_\ell \leq x$ is greater than $\Pi_{\ell=L}^{\infty}(1 - e^{-\ell I(x)}$, that is a positive quantity. Hence, there is a positive probability to always stay closer from av_{Y_1,Y_2} and the set of runs that will not

trigger a reset have a strictly positive probability. Thus, one of these run will be classified as being in $\mathcal{A}_1$, e.g. by using the classifier from section 3.5.

The converse is simpler: if there does not exist a classifiable $(x_1, X_2) \in \mathcal{A}'_1$, it means that for every x_1, there exists a x_2 such that (x_1, x_2) is not classifiable. In particular, we can get a positive probability p_{x_2} to perform the exact same observation from (x_1, x_2), and taking the $min_{x_2} p_{x_2} = p > 0$, taking by contradiction a reset strategy and a w_k, then there is probability at least p to misclassify w_k, no matter its size, a contradiction. □

In case there are more than two LMCs, we follow the state s of one LMC and the belief of every other LMCs along the observation, and we need to check classifiability between (s, t) for every t in the belief of any of the other LMCs. Using this characterization, we obtain:

Theorem 4.23. *Let $\mathcal{A}_1$, $\mathcal{A}_2$ be two LMCs. It is PSPACE-complete to check whether $(\mathcal{A}_1, \mathcal{A}_2)$ are limit-sure attack-classifiable.*

Proof. First, it is easy to see that the problem is in PSPACE: For each $(x_1, x_2) \in \mathcal{A}$, we test in PTIME whether (x_1, x_2) is classifiable, by using lines $3 - 7$ algorithm 3. Then, $(\mathcal{A}_1, \mathcal{A}_2)$ are limit-sure attack-classifiable iff one can reach a (x_1, X_2) classifiable in $\mathcal{A}'_1$ or a (x_2, X_1) classifiable in $\mathcal{A}'_2$, which is PSPACE as $\mathcal{A}'_1, \mathcal{A}'_2$ have an exponential number of states compared with $\mathcal{A}_1, \mathcal{A}_2$ and reachability is in NLOGSPACE.

To prove hardness, we reduce from the language inclusion for finite automaton. Let $\mathcal{B}_1$, $\mathcal{B}_2$ be two finite automata over alphabet Σ, with $\mathcal{B}_i = (S_i, s_0^i, \Delta_i, F_i)$, where F_i is a set of accepting states. We assume wlog that every state of S_i is reachable and F_i is reachable from any state s of S_i. We associate with $\mathcal{B}_i, i \in \{1, 2\}$ the LMC $\mathcal{A}_i = (S_i \cup \{s_F^i\}, \sigma_0^i, M_i,)$ over alphabet $\Sigma \cup \{f\}$ with:

- $\sigma_0^i(s) = 1$ for $s = s_0^i$, and $\sigma_0^i(s) = 0$ otherwise,
- $M_i(s, a, s') > 0$ iff $(s, a, s') \in \Delta_i$, for all $s, s' \in S_i, a \in \Sigma$,
- $M_i(s, f, s_F) > 0$ iff $s \in F_i$, for all $s \in S_i$,
- $M_i(s_F, f, s_F) = 1$.

Notice that the exact positive probability values will have no impact in the following (for instance, we can take these probabilities uniform). Now, it is easy to see that for any word $w \in \Sigma^*$, $w \in L(\mathcal{B}_i)$ iff $P_{\mathcal{A}_i}(wf) > 0$. Now, we prove that $(\mathcal{A}_1, \mathcal{A}_2)$ are limit-sure attack-classifiable iff $L(\mathcal{B}_1) \not\subset L(\mathcal{B}_2)$:

Assume that $L(\mathcal{B}_1) \subset L(\mathcal{B}_2)$. Hence, for all $(x_1, X_2) \in \mathcal{A}'_1$, we have $X_2 \neq \emptyset$. Also, if $x_1 \in F_1$, then $X_2 \cap F_2 \neq \emptyset$. As from every state, F_1 can be reached in $\mathcal{B}_1$, we have that

there is a unique BSCC of twin states $\{(s_f^1, s_f^2)\}$. Clearly, (s_f^1, s_f^2) is not classifiable and thus $(\mathcal{A}_1, \mathcal{A}_2)$ is not limit-sure attack-classifiable.

Conversely, assume that $L(\mathcal{B}_1) \not\subset L(\mathcal{B}_2)$. Thus, there exists ρ with label $w \in L(\mathcal{B}_1) \setminus L(\mathcal{B}_2)$, and if we consider the associated path in $\mathcal{A}_1'$, it reaches (x_1, X_2), with $x_1 \in F_1$ and $X_2 \cap F_2 = \emptyset$. Doing action f from there, we reach state $(s_f^1, \emptyset)$, which is classifiable. □

4.5.3 Existence of $(1-\varepsilon)$ attack-classifiers for all ε is undecidable.

We now turn to the other notion. Let $\varepsilon > 0$. An $(1-\varepsilon)$ attack-classifier for two LMCs $\mathcal{A}_1, \mathcal{A}_2$ is given by:

1. A reset strategy $\tau : \Sigma^* \to \{\bot, reset\}$ telling when to reset and which eventually stops resetting, with probability 1 on the reset runs, and
2. a $(1-\varepsilon)$-classifier for u, where $u \in \Sigma^*$ denotes the suffix of the observations since the last reset.

We next show that this notion, which we showed to be weaker than limit-sure attack-classifiability on Fig 4.8, is also computationally much harder. In fact, it is undecidable.

Theorem 4.24. *It is undecidable to know whether for all ε, there exists an $(1-\varepsilon)$ attack-classifier between 2 LMCs.*

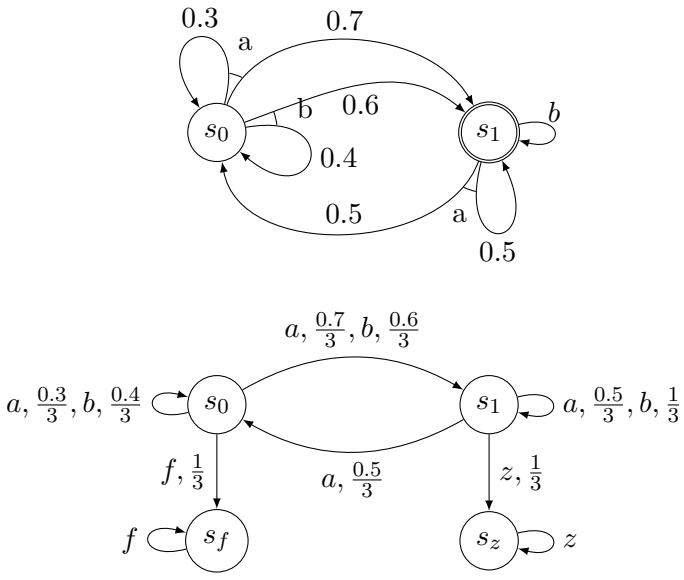

Figure 4.9: Example of the PFA (above) to LMC (below) reduction

Proof. It is undecidable [GO10] to know whether a PFA $\mathcal{B}$, that accepts all words with probability in $(0,1)$, is 0 and 1 isolated, that is, there is no sequence of words $(w_i)_{i\in\mathbb{N}}$ such that $lim_{n\to\infty} P^{\mathcal{B}}(w_i) = 0$ or $= 1$.

Let $\mathcal{B}_1$ be such a PFA. Wlog, we can assume that it is complete, that is from each state s and each letter $a \in \Sigma$, there is a transition from s labeled by a (it suffices to add a sink state if it is not the case). Further, let $\mathcal{B}_2$ be a PFA with a single state that accepts every word of Σ^* with probability 1. Let $\mathcal{B}_2$ be the complete PFA with 2 states (one accepting and one non accepting, with transition with probability $1/2$ to stay in the same state and $1/2$ to switch state) that accepts every word with probability $1/2$.

From $\mathcal{B}_1$ and $\mathcal{B}_2$, we define $\mathcal{A}_1$, $\mathcal{A}_2$ two LMCs in the following manner:

Let $\mathcal{B} = (S, s_0, (M_a)_{a\in\Sigma}, F)$ be a PFA over Σ. We denote $\mathcal{A}$ the LMC $(S\cup\{s_f, s_z\}, s_0, M)$ over $\Sigma \cup \{f, z\}$ with:

1. $M(s, a, s') = \frac{M_a[s,s']}{|\Sigma|+1}$ for all $s, s' \in S, a \in \Sigma$,
2. If $s \in F$, then $M(s, f, s_f) = \frac{1}{|\Sigma|+1}$.
3. If $s \notin F$, then $M(s, z, s_z) = \frac{1}{|\Sigma|+1}$.
4. $M(s_f, f, s_f) = 1$ and $M(s_z, z, s_z) = 1$.

An example of this construction is provided in figure 4.9. For all observation $w \in \Sigma^*$, we have:

- $P_{\mathcal{A}_1}(w) = P_{\mathcal{A}_2}(w) = \frac{1}{(|\Sigma|+1)^{|w|+1}}$,
- $P_{\mathcal{A}_1}(wf^k) = \frac{P_{\mathcal{B}_1}(w)}{(|\Sigma|+1)^{|w|+1}}$ and $P_{\mathcal{A}_1}(wz^k) = \frac{1-P_{\mathcal{B}_1}(w)}{(|\Sigma|+1)^{|w|+1}}$,
- $P_{\mathcal{A}_2}(wf^k) = P_{\mathcal{A}_2}(wz^k) = \frac{1}{2(|\Sigma|+1)^{|w|+1}}$.

If $\mathcal{B}_1$ is 0 and 1 isolated, then there exists a ε such that $\varepsilon < P_{\mathcal{B}_1}(w) < 1-\varepsilon$ for all $w \in \Sigma^*$. That is, for all words $w \in (\Sigma\cup\{z, f\})^*$, we have $2\varepsilon P_{\mathcal{A}_2}(w) \leq P_{\mathcal{A}_1}(w) \leq 2P_{\mathcal{A}_2}(w)$. Assume by contradiction that there exists a reset strategy and an $(1-\varepsilon)$ classifier f. The probability to see w is $P(w) = 1/2P_{\mathcal{A}_1}(w)+1/2P_{\mathcal{A}_2}(w)$. The probability of misclassification knowing that the observation is w is thus either $P_{\mathcal{A}_1}(w)/P(w)$ or $P_{\mathcal{A}_2}(w)/P(w)$. The first one is at least $2\varepsilon/3$ and the second one is at least $1/3$. That is, the limit when the size of the observation tends to infinity is also at least $2\varepsilon/3$, and there does not exists any $1-\varepsilon/2$ attack-classifier.

Conversely, if $\mathcal{B}_1$ is not 0 isolated, then for all ε, there exists w_ε such that $P_{\mathcal{B}_1}(w_\varepsilon) < \varepsilon$. The reset strategy waits to see $w_\varepsilon f$: that is, it resets if the observation u is not a prefix of $w_\varepsilon f$. When the observation $u = w_\varepsilon$, which happens eventually with probability 1, the classifier claims that the LMC is $\mathcal{A}_2$. This is true with probability $> 1-2\varepsilon$.

The last case is $\mathcal{B}_1$ is not 1 isolated, and for all ε, there exists w_ε such that $P_{\mathcal{B}_1}(w_\varepsilon) < \varepsilon$. The result is symmetrical: the reset strategy waits for $w_\varepsilon z$, in which case the classifier claims that the LMC is $\mathcal{A}_2$. This is true with probability $> 1 - 2\varepsilon$. □

4.6 Related work

4.6.1 Other distances

In this chapter, we mostly considered distances closely related to the L_p ones. In other domains such as machine learning, various distances have been used in order to measure the discrepancy between models. Some are not *distances* in the mathematical sense, since they may lack some property, such as symmetry or triangular inequality. However, they all express a notion of proximity, and for all these "distances" d we have $d(\mathcal{A}_1, \mathcal{A}_2) = 0$ iff they have the same stochastic language.

The Kullback-Leibler divergence, also called relative entropy is given by

$$D(\mathcal{A}_1, \mathcal{A}_2) = \sum_{w \in \Sigma^*} p_1(w) \cdot \log \frac{p_1(w)}{p_2(w)}$$

Notice that this divergence is not symmetrical: thus, it is not a distance. The divergence is infinite iff there exists a word w such that $p_1(w) > 0$ and $p_2(w) = 0$. It is used when the notion of inclusion of the language is strongly needed [Cor+08].

The Hellinger distance is given by

$$Hellinger(\mathcal{A}_1, \mathcal{A}_2) = (\sum_{w \in \Sigma^*} (\sqrt{p_1(x)} - \sqrt{p_2(x)})^2)^{\frac{1}{2}}$$

[TC13] uses it as a way to measure some information loss in the context of data protection.

The Jensen-Shannon divergence is given by

$$JS(\mathcal{A}_1, \mathcal{A}_2) = \sum_{w \in \Sigma^*} p_1(w) \cdot \log \frac{2 \cdot p_1(w)}{p_1(w) + p_2(w)} + p_2(w) \cdot \log \frac{2 \cdot p_2(w)}{p_1(w) + p_2(w)}$$

This divergence is similar to the Kullback-Leibler one but with the notable difference that it is symmetric and always finite. It has been used in machine learning alongside to Kullback-Leibler divergence in the context of adversarial machine learning [Goo+14].

4.6.2 Testing

Before the different work on distinguishability and classification, one way to differentiate systems was by performing hypothesis testing. Intuitively, two statistical data sets are compared: one given by the experience and one synthetic data set given by an idealized model. This comparison is used to determine if some underlying assumption is true. For example, an assumption could be "does the rate of growth of this process follow an exponential law"? Hypothesis testing has been studied in a long time for various problems[Wal45; Wet66; SS83].

Later, [ACY95] used another kind of testing to investigate the problem of determining the initial state of a finite state transition systems (possibly probabilistic) among several choices. The authors establish a link between the existence of a strategy to find the initial state and winning strategies in Markov Decision Processes with incomplete information.

4.7 Conclusion

4.7.1 Summary

This chapter presented a study on limit-sure classification with three main contributions. The first is the definition of stationary distributions for LMCs. We believe that this notion can find applications in contexts other than classification.

The second contribution was a characterization of limit-sure classifiability thanks to these stationary distributions. Two LMCs are not classifiable iff they have beliefs which can be reached by the same observation and for which the stationary distributions can be separated by one finite word. This characterization led to a PTIME algorithm surprisingly close to the one of [CK14], even if the methods to obtain the algorithms were very different.

The final contribution was the study of limit-sure classifiability in a security context: the attacker has a power to launch a new execution if he is not "satisfied" with the current one. We showed that deciding the existence of a limit-sure attack-classifier is PSPACE-complete. Further, the existence of a $(1-\varepsilon)$-attack-classifier is undecidable.

	limit-sure classifiability	limit-sure attack-classifiability	$\forall\varepsilon, (1-\varepsilon)$ attack-classifiability
Complexity	PTIME	PSPACE-complete	Undecidable

4.7.2 Perspectives

On this particular work, an interesting task would be to express an algorithm that does not use linear programming: although both algorithms ([CK14] and ours) are correct, the "how" is not intuitive. This was one of the goal we considered on this subject. However, our final solution still had to rely on linear programming.

On a broader view, the notion of stationary distribution for LMCs looks promising and we believe it could be applied in different contexts. It could be either on language problems (such as diagnosis or opacity) or even be extended to more powerful systems such as Markov Decision Processes, leading to opportunities in very different domains.

Chapter 5

Learning of Markov Chains

In the previous chapter, we saw how to differentiate two LMCs. However, one question is "how does one obtain these systems?". This question is crucial especially when the systems are estimated from real life machines. One way to perform this estimation is through automatized learning.

The last decades have seen the rise of automatized learning in order to tackle problems which are at first sight intractable. This process has shown huge success and is trending for many applications. Among them, many are critical and need extra safety: automatic cars, facial recognition, automatic translation... However, although automatic learning does work, with significant progress achieved every year, they suffer from some flaws. A first flaw is that because of the structure of the mathematical representation of the learning process, it is hard for a human to understand how and why it works. A second flaw is that until recently no real guarantee was possible on the learning process. A consequence of this flaw is that many systems were easily attacked: a small deviation in an entry may lead to significant change in the output. Further, there is often limited budget in observing and learning from the system, and the validity of the learned model is in question.

To counter that, a recent trend is to develop certifications: formal properties stating some notion of safety. These certifications are being developed for a wide diversity of learning processes, such as Deep Neural Networks [Hua+17; WHK18]. In this chapter, we focus on learning stochastic systems, especially Markov Chains. Comparing two Markov processes (in our case, the original model and the learned one) is a common problem that relies on a notion of divergence. Most existing approaches focus on deviation between the probabilities of local transitions (*e.g.*, [CT04; SGB95; CG08]). However, a single deviation in a transition probability between the original system and the learned model may lead to large differences in their global behaviors. For instance, the probability of reaching certain state may be magnified by paths which go through the same deviated transition many times. It is thus important to use a measure that quantifies the differences over global behaviors, rather than simply checking whether the differences between the individual transition probabilities are low enough.

A major change with respect to previous works on this topic is that we consider global behaviors instead of local deviations. We consider Temporal Logic (LTL and CTL) to model these global behaviors. Agreeing on all formulas of LTL means that the first order behaviors of the system and the model are the same, while agreeing on CTL means that the system and the model are bisimilar [BK08]. Our goal is to provide stopping rules in the learning process of Markov Chains that provides Probably Approximately Correct (PAC) bounds on the error in probabilities of every properties in the logic between the model and the system.

This chapter is organized as follows: section 5.1 presents a state of the art on several subjects of interest in this chapter. Subsection 5.1.1 describes different estimations techniques that are used in the literature. Subsection 5.1.2 presents what Probably Approximately Correct learning is, with standard algorithm and bounds introduced in subsection 5.1.3. Finally, we show a result from Daca et al. [Dac+16] that will be closely related to our results. In section 5.2 we present a special case of our problem: learning time to failures properties. A certification is provided in subsection 5.2.2 with an algorithm in subsection 5.2.3. Then, a more general framework is adopted in section 5.3. We start by providing a negative result for LTL in subsection 5.3.1. Then, in order to tackle CTL, we define mathematical tools in subsection 5.3.2 and prove their necessity in subsection 5.3.3. Then, PAC bounds are provided in subsection 5.3.4 with an algorithm in subsection 5.3.5. Some evaluation of the algorithm is provided in section 5.4.

5.1 State of the art

In the following, $\mathcal{M} = (S, \mu_0, M)$ will be a discrete time Markov Chain that has m states. A trace is a sequence of observations of states produced by the execution of a Markov Chain.

5.1.1 Estimators

In order to learn a Markov chain, we have to estimate the probabilities of each transition. For that, diverse methods called estimators have been studied. We present some here that we use in this thesis and others used in the literature.

Frequency estimation of a Markov Chain

Given a set W of n traces, we denote n_{ij}^W the number of times transition from state i to state j occurred and n_i^W the number of times a transition has been taken from state i.

The *frequency estimator* of $\mathcal{M}$ is the Markov Chain $\hat{\mathcal{M}}_W = (\hat{a}_{ij})_{1\leq i,j\leq m}$ given by $\hat{a}_{ij} = \frac{n_{ij}^W}{n_i^W}$ for all i, j, with $\sum_{i=1}^m n_i^W = \sum_{i=1}^m \sum_{j=1}^m n_{ij}^W = |W|$. In other words, to learn $\hat{\mathcal{M}}_W$, it suffices to count the number of times a transition from i to j occurred, and divide by the number of times state i has been observed. The matrix $\hat{\mathcal{M}}_W$ is trivially a Markov Chain, except for states i which have not been visited. In this case, one can set $\hat{a}_{ij} = \frac{1}{m}$ for all state j and obtain a Markov Chain.

Example 5.1. *Let us suppose we have a Markov chain with* 5 *states denoted* s_i *for* $1 \leq i \leq 5$ *and a sample* W *is constituted of the following observations:*

- 3 *times* $s_1\ s_2\ s_3\ s_5\ s_5$
- 2 *times* $s_1\ s_3\ s_1\ s_2\ s_4\ s_4$

Then, the Markov Chain estimated by the frequency estimator is depicted in figure 5.1. For example, the state s_1 *is followed five times by* s_2 *and two times by* s_3*, hence the probabilities* $\frac{5}{7}$ *and* $\frac{2}{7}$*.*

Learning Markov Chains with Laplace smoothing

A second estimator is based on the frequency estimator with an additional property: assuming one knows the structure of the system, one can add a bias to all possible transition.

Let $\alpha > 0$. For any state s, let k_s be the number of successors of s, that we know by hypothesis, and $T = \sum_{s\in S} k_s$ be the number of non-zero transitions. Let W be a set of traces, n_{ij}^W the number of transitions from state i to state j, and $n_i^W = \sum_j n_{ij}^W$. The

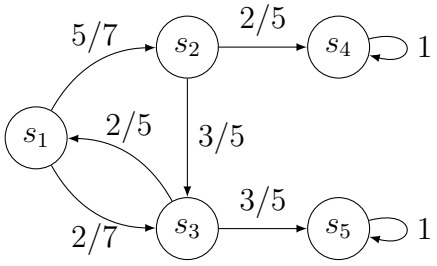

Figure 5.1: Example of a MC $\mathcal{M}$ learnt from a sample of executions by a frequency estimator

estimator for W with Laplace smoothing α is the Markov Chain $\hat{\mathcal{M}}_W^\alpha = (\hat{a}_{ij})_{1\leq i,j\leq m}$ given for all i, j by:

$$\hat{a}_{ij} = \frac{n_{ij}^W + \alpha}{n_i^W + k_i\alpha} \text{ if } a_{ij} \neq 0 \quad \text{and} \quad \hat{a}_{ij} = 0 \text{ otherwise}$$

In comparison with the frequency estimator, the Laplace smoothing adds for each state s a term α to the numerator and k_s times α to the denominator. This preserves the fact that $\hat{\mathcal{M}}_W^\alpha$ is a Markov chain, and it ensures that $\hat{a}_{ij} \neq 0$ iff $a_{ij} \neq 0$. In particular, compared with the frequency estimator, it avoids creating zeros in the probability tables. One difficulty of this estimator is that picking a good α may not be easy. In the following, we will give ground for defining defining a reasonable α.

Example 5.2. *We continue example 5.1. Let us suppose that there was an additional transition that was not seen from s_2 to s_1. By knowing the support, we add a bias with $\alpha = 1$ and we calculate the new probabilities. For example, the probability for the transition from s_1 to s_2 is now equal to $\frac{5+1}{7+2\times1} = \frac{2}{3}$. The transition from s_2 to s_1 has now the value $\frac{1}{8}$ and then the structure is preserved.*

Good-Turing frequency estimator

Other estimators have been used in the literature. One example is the Good-Turing one introduced in [Goo53], that has been used in learning of models associated with speech processing [GS95]. Let s_i be a state of the Markov chain and n_r the number of transitions leaving state s_i that have been seen r times. The total number of transitions seen is $N = \sum_r r \cdot N_r$. Then, the probability to see a transition that has been seen r times is

Figure 5.2: Example of a MC $\mathcal{M}$ whose support is known (left) learnt from a sample of executions with Laplace smoothing of parameter $\alpha = 1$ (right).

estimated to:

$$p_r = \frac{(r+1)N_{r+1}}{N}$$

Finally, the probability to be from a specific transition that has been seen r times is $\frac{p_r}{N_r}$. Some variants have been used, such as in [CG91; HYH13].

Other estimators

Different estimators exist and have been utilized in order to take into account more complex information. For example, the Kneser-Ney estimator [CG99] is an estimator that considers some kind of memory, that is the frequency of some transition will depend on the previous ones. Katz's back off model [Kat87] uses two different laws for the prediction, depending on if a transition has been seen more than some threshold.

5.1.2 Probably Approximately Correct learning

In learning theory, Probably Approximately Correct (PAC) learning introduced in [Val84] is a framework allowing one to reason about machine learning and especially about supervised learning. The general idea behind PAC is that the answer to exact qualitative guarantees about a learning process such as "is the system I learnt from observations exactly the same as the original one?" will generally be "no". Thus, the questions about conformity must contain a quantitative component. The general formulation of a PAC learning property is "does with high probability the distance between a model and the learned one is low?".

To analyze the behavior of a system, properties are specified in temporal logic (*e.g.*, LTL or CTL, defined in Chapter 2.4.2). Given a logic $\mathcal{L}$ and φ a property of $\mathcal{L}$, decidable in finite time, we denote $\omega \models \varphi$ if a path ω satisfies φ. Let $z : \Omega \times \mathcal{L} \to \{0, 1\}$ be the function that assigns 1 to a path ω if $\omega \models \varphi$ and 0 otherwise. In what follows, we assume that we have a procedure that draws path ω with respect to $\mathbb{P}^{\mathcal{M}}$ and outputs $z(\omega, \varphi)$. Further, we denote $\gamma(\mathcal{M}, \varphi)$ the probability that a path drawn with respect to $\mathbb{P}^{\mathcal{M}}$ satisfies φ. We omit the property or the Markov Chain in the notation when it is clear from the context. Finally, note that the behavior of $z(., \varphi)$ can be modeled as a Bernoulli random variable Z_φ parameterized by the mean value $\gamma(\mathcal{M}, \varphi)$.

Given $\varepsilon > 0$ and $0 < \delta < 1$, we say that a property φ of $\mathcal{L}$ is PAC-learnable if there is an algorithm $\mathcal{A}$ such that, given a sample of n paths drawn according to the procedure, with probability of at least $1 - \delta$, $\mathcal{A}$ outputs in polynomial time (in $1/\varepsilon$ and $1/\delta$) an approximation of the average value for Z_φ close to its exact value, up to an error less than

or equal to ε. Formally, φ is PAC-learnable if and only if $\mathcal{A}$ outputs an approximation $\hat{\gamma}$ such that:

$$\mathbb{P}\left(|\gamma - \hat{\gamma}| > \varepsilon\right) \leq \delta \tag{5.1}$$

Moreover, if the above statement for algorithm $\mathcal{A}$ is true for every property in $\mathcal{L}$, we say that $\mathcal{A}$ is a PAC learning algorithm for $\mathcal{L}$.

5.1.3 Monte-Carlo estimation and algorithm of Chen

Let φ be a formula such that with probability 1 φ is eventually satisfied or violated. Given a sample W of n paths drawn according to $\mathbb{P}^{\mathcal{M}}$ until φ is satisfied or violated, the crude Monte-Carlo estimator, denoted $\hat{\gamma}_W(\mathcal{M}, \varphi)$, of the mean value for the random variable Z_φ is given by the empirical frequency: $\hat{\gamma}_W(\mathcal{M}, \varphi) = \frac{1}{n}\sum_{i=1}^{n} z(\omega_i) \approx \gamma(A, \varphi)$.

The Okamoto inequality [Oka58] (also called the Chernoff bound in the literature) is often used to guarantee that the deviation between a Monte-Carlo estimator $\hat{\gamma}_W$ and the exact value γ by more than $\varepsilon > 0$ is bounded by a predefined confidence parameter δ.

Theorem 5.1 (Okamoto bound)**.** *Let $\varepsilon > 0$, δ such that $0 < \delta < 1$ and $\hat{\gamma}_W$ be the crude Monte-Carlo estimator of probability γ. If $n \geq \frac{1}{2\varepsilon^2}\log\left(\frac{2}{\delta}\right)$,*

$$\mathbb{P}(|\gamma - \hat{\gamma}_W| > \varepsilon) \leq \delta.$$

However, several sequential algorithms have been recently proposed to guarantee the same confidence and accuracy with fewer samples. In what follows, we use the algorithm of Chen [Che13].

Theorem 5.2 (Chen bound)**.** *Let $\varepsilon > 0$, δ such that $0 < \delta < 1$ and $\hat{\gamma}_W$ be the crude Monte-Carlo estimator, based on n samples, of probability γ.*

If $n \geq \frac{2}{\varepsilon^2}\log\left(\frac{2}{\delta}\right)\left[\frac{1}{4} - (|\frac{1}{2} - \hat{\gamma}_W| - \frac{2}{3}\varepsilon)^2\right]$,

$$\mathbb{P}(|\gamma - \hat{\gamma}_W| > \varepsilon) \leq \delta.$$

To ease the readability, we write $n_{\text{succ}} = \sum_{i=1}^{n} z(\omega_i)$ and

$$H(n, n_{\text{succ}}, \epsilon, \delta) = \frac{2}{\varepsilon^2}\log\left(\frac{2}{\delta}\right)\left[\frac{1}{4} - (|\frac{1}{2} - \hat{\gamma}_W| - \frac{2}{3}\varepsilon)^2\right]$$

When it is clear from the context, we only write $H(n)$. Then, the algorithm $\mathcal{A}$ that stops sampling as soon as $n \geq H(n)$ and outputs a crude Monte-Carlo estimator for $\gamma(\mathcal{M}, \varphi)$ is a PAC-learning algorithm for φ. The condition over n is called the stopping criteria of the algorithm. This algorithm requires fewer samples than other sequential algorithms (such as in [JSS17]). Note that the estimation of a probability close to 1/2 likely requires more samples since $H(n)$ is maximized in $\hat{\gamma}_W = 1/2$.

5.2 Learning for a time-to-failure property

In this section, we focus on property φ of reaching a failure state s_F from an initial state s_0 without re-passing by the initial state, which is often used for assessing the failure rate of a system and the mean time between failures (see *e.g.*, [Rid05]). Without loss of generality, we assume that there is a unique failure state s_F in A. We also assume that, with probability 1, the runs eventually re-pass by s_0 or reach s_F. We denote $\gamma(\mathcal{M}, \varphi)$ the probability, given Markov Chain $\mathcal{M}$, of satisfying property φ, *i.e.*, the probability of a failure between two visits of s_0.

5.2.1 Framework

Assume that the stochastic system $\mathcal{M}$ is observed from state s_0. Between two visits of s_0, property φ can be monitored. If s_F is observed between two instances of s_0, we say that the path $\omega = s_0 \cdot \rho \cdot s_F$ satisfies φ, with $s_0, s_F \notin \rho$. Otherwise, if s_0 is visited again from s_0, then we say that the path $\omega = s_0 \cdot \rho \cdot s_0$ violates φ, with $s_0, s_F \notin \rho$. We call *traces* paths of the form $\omega = s_0 \cdot \rho \cdot (s_0 \vee s_F)$ with $s_0, s_F \notin \rho$.

In the following, we show that it is sufficient to use a *frequency estimator* to learn a Markov Chain which provides a good approximate for such a property. Let $\hat{M}_W$ be the matrix learned using the frequency estimator from the set W of traces, and let M be the real probabilistic matrix of the original system $\mathcal{M}$.

We show that, in the case of time-to-failure properties, $\gamma(\hat{\mathcal{M}}_W, \varphi)$ is equal to the crude Monte Carlo estimator $\hat{\gamma}_W(A, \varphi)$ induced by W.

5.2.2 PAC bounds for a time-to-failure property

In this section, we present how we can obtain PAC bounds given a set of samples and a system. That is we bound the error between $\gamma(\mathcal{M}, \varphi)$ and $\gamma(\hat{\mathcal{M}}_W, \varphi)$:

Theorem 5.3. *Given a set W of n traces such that $n = \lceil H(n) \rceil$, we have:*

$$\mathbb{P}\left(|\gamma(\mathcal{M}, \varphi) - \gamma(\hat{\mathcal{M}}_W, \varphi)| > \varepsilon\right) \leq \delta \tag{5.2}$$

where $\hat{\mathcal{M}}_W$ is the frequency estimator of $\mathcal{M}$.

To prove Theorem (5.3), we first use the algorithm of Chen (Theorem 5.2) to establish:

$$\mathbb{P}\left(|\gamma(\mathcal{M}, \varphi) - \hat{\gamma}_W(\mathcal{M}, \varphi)| > \varepsilon\right) \leq \delta \tag{5.3}$$

It remains to show that $\hat{\gamma}_W(\mathcal{M}, \varphi) = \gamma(\hat{\mathcal{M}}_W, \varphi)$:

Proposition 5.4. *Given a set W of traces, $\gamma(\hat{\mathcal{M}}_W, \varphi) = \hat{\gamma}_W(\mathcal{M}, \varphi)$.*

It might be appealing to think that this result can be proved by induction on the size of the traces, mimicking the proof of computation of reachability probabilities by linear programming. This is actually not the case. The remaining of this section is devoted to proving Proposition (5.4).

We first define $q_W(u)$ the number of occurrences of sequence u in the traces of W. Note that u can be a state, an individual transition or even a path. We also use the following definitions in the proof.

Definition 5.1 (Equivalence)**.**
Two sets of traces W and W' are equivalent if for all $s, t \in S$, $\frac{q_W(s \cdot t)}{q_W(s)} = \frac{q_{W'}(s \cdot t)}{q_{W'}(s)}$.

Generally speaking, two set of traces are equivalent if they induce the same Markov Chain.

Example 5.3. *Let $W = \{s_0 s_1 s_2 s_2 s_3 s_3,\ s_0 s_2 s_1 s_3 s_3\}$, $W' = \{s_0 s_1 s_2 s_2 s_1 s_3 s_3,\ s_0 s_2 s_3 s_3\}$ and $W'' = \{s_0 s_1 s_2 s_2 s_2 s_1 s_3 s_3,\ s_0 s_2 s_3 s_3\}$. W and W' are equivalent and as a consequence they induce the same Markov Chain depicted in figure 5.3. W'' is not equivalent to them: the proportion of s_2 following s_2 is different. The Markov Chain corresponding to W'' is then different.*

Definition 5.2 (s-factor)**.**
Given a trace r and a state s, F is an s-factor of r if F is a factor of r and F starts by s. Moreover, F is elementary if it does not contain any other s.

We define a set of traces W' equivalent with W, implying that $\hat{\mathcal{M}}_W = \hat{\mathcal{M}}_{W'}$. This set W' of traces satisfies the following:

Lemma 5.5. *For any set of traces W, there exists a set of traces W' such that:*

1. *W and W' are equivalent,*
2. *for all $r, s, t \in S$, $q_{W'}(r \cdot s \cdot t) = \dfrac{q_{W'}(r \cdot s) \times q_{W'}(s \cdot t)}{q_{W'}(s)}$.*

A trace can then be seen as a set of factors $BF_1 \dots F_k E$ where B is the special factor beginning, F_i are some s-factors for all i and E is the special ending s-factor. We can notice that for all trace r and r' obtained by permutation of the F_i, $\{r\}$ and $\{r'\}$ are equivalent. Without loss of generality, we suppose that states s_j such that there exists a transition (s_j, s) are states $s_1, \dots, s_Q$. In W, we denote n_i the number of transitions (s, s_i), m_j the number of transitions (s_j, s) and $q_{i,j} = \frac{q_W(s_j \cdot s) \times q_W(s \cdot s_i)}{q_W(s)}$. $q_{i,j}$ represents then the number of times a transition (s_j, s) should be followed by a transition (s, s_i). By definition, we have that $\sum_i n_i = \sum_j m_j = q_W(s)$ and $\forall i, j, q_{i,j} > 0$. We denote $k = q_W(s)$. Finally, for a factor F, we denote by $q_{F,i,j}$ the number of occurrences of (s_j, s, s_i) in F.

Proof. (of Lemma 5.5) Let us suppose that W is made of only one trace $r = BF_1 \dots F_f E$.

We prove the lemma by induction on $s \in V$, then induction on the number of predecessors of s.

Let suppose that every factor in $\{B, F_1, \dots, F_f\}$ ends with s_1, that $f = m_1 - 1$ and that the sequence $s_1 s$ never appears neither in B nor in F_l for all l nor in E. We also suppose that $\forall i, \forall j > 1$, $q_{B,i,j} + \sum_{l=1}^{f} q_{F_l,i,j} + q_{E,i,j} = q_{i,j}$. It means that for all $j > 1$ for all i, we have $q_{W'}(s_j \cdot s \cdot s_i) = \frac{q_{W'}(s_j \cdot s) \times q_{W'}(s \cdot s_i)}{q_{W'}(s)}$ and we just have to consider s_1.

Let r' be $BF_1 \dots F_f E$. r' is equivalent to r. We also obtain that for all i, $q_{r,i,1} = q_{i,1}$ since there are exactly $q_{i,1}$ factors starting by s_i. Furthermore, $\forall i, \forall j > 1$, $q_{r,i,j} = q_{B,i,j} +$

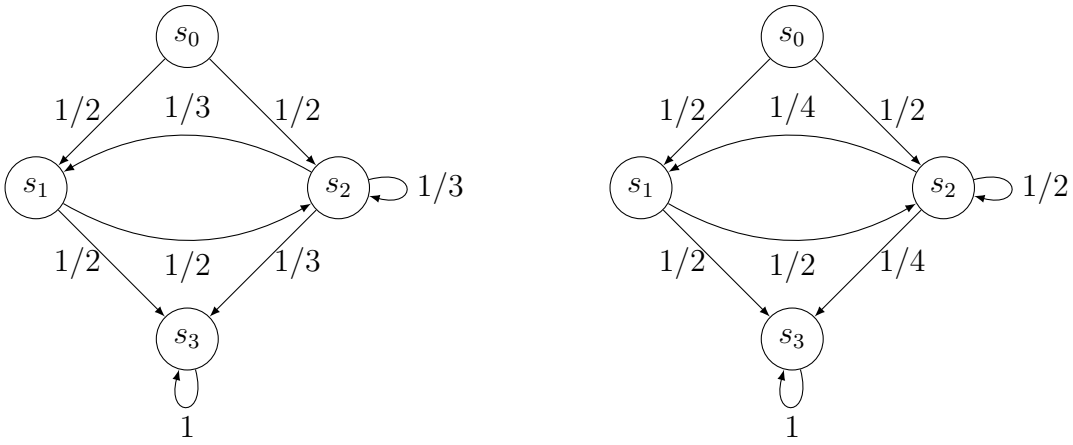

Figure 5.3: MC induced by W and W' (left) and by W'' (right)

$\sum_{l=1}^{f} q_{F_l,i,j} + q_{E,i,j} = q_{i,j}$. Indeed, none was added and we did not break those already existing. Then, $\forall i, \forall j, q_{W'}(s_j \cdot s \cdot s_i) = \frac{q_{W'}(s_j \cdot s) \times q_{W'}(s \cdot s_i)}{q_{W'}(s)}$.

Now, let us consider when there are J states to deal with, $J > 1$, and the factors $\{B, F_1 \cdots F_f, E\}$ such that $f = \sum_{j=1}^{J} m_j - 1$. Besides, for all $j \leq J$, exactly m_j factors in $\{B, F_1 \cdots F_f\}$ end with s_j and for all i, exactly $\sum_{j=1}^{J} q_{i,j}$ factors in $\{F_1 \cdots F_f, E\}$ start with ss_i. Furthermore, for all $j \leq J$, the sequence $s_j s$ never appears neither in B nor in F_l for all l and for all i, for all $j > J$, $q_{B,i,j} + \sum_{l=1}^{f} q_{F_l,i,j} + q_{E,i,j} = q_{i,j}$.

We create new factors in order to deal with s_J by merging the existing one. We apply the following algorithm:

Algorithm 4 Merge($Factors, J$)

for i from 1 to Q **do**
  for l from 1 to $q_{i,J}$ **do**
    Choose F_1 ending by s_J, a factor $F_2 \neq F_1$ beginning by s_i, we denote $F' = F_1 F_2$
    $Factors = Factors \setminus \{F_1, F_2\} \cup \{F'\}$
  end for
end for
return $Factors$

Since $\sum_i q_{i,j} = m_j$, there is always one factor ending by S_J that can be chosen. Let us suppose that there is no candidate for F_2. It means that no factor other than F_1 starts by ss_i, and then $\sum_{j=1}^{J} q_{i,j} \leq q_{i,J}$ (number available at start smaller than number used). We deduce that for all $j < J$, $q_{i,j} = 0$ and that is absurd.

We obtain the set of factors$\{B', F'_1, \cdots, F'_{f'}, E'\}$. We have merged m_J factors, then $f' = f - m_J = \sum_{j=1}^{j-1} m_j - 1$. For all $j < J$, the number of factors ending with s_j has not changed. For all i, there are $\sum_{j=1}^{J} q_{i,j} - q_{i,J} = \sum_{j=1}^{J-1} q_{i,j}$ factors in $\{F'_1 \cdots F'_{f'}, E'\}$ starting with ss_i. Furthermore, for all $j < J$, the sequence $s_j s$ still never appears neither in B nor in F_l for all l and for all i, for all $j \geq J$, $q_{B,i,j} + \sum_{l=1}^{f} q_{F_l,i,j} + q_{E,i,j} = q_{i,j}$.

At start, when considering all elementary factors $\{B, F_1, \cdots, F_f, E\}$, we have $f = k - 1 = \sum_{j=1}^{Q} m_j - 1$ and for all j, exactly m_j factors in $\{B, F_1 \cdots F_f\}$ ends with s_j and for all i, exactly $\sum_{j=1}^{Q} q_{i,j} = n_i$ factors in $\{F_1 \cdots F_f, E\}$ start with ss_i. Besides, since all factors are elementary, no sequence $s_j s$ appears in any of them and trivially, for all $j > Q$, $q_{B,i,j} + \sum_{l=1}^{f} q_{F_l,i,j} + q_{E,i,j} = 0$. Thus, the requirements are met. □

Example 5.4. *Let us consider again the set* $W = \{s_0 s_1 s_2 s_2 s_1 s_3 s_3,\ s_0 s_2 s_3 s_3\}$ *and* $s = s_1$. *The decomposition in* s_1 *factors of* W *gives the beginning blocks* s_0 *and* $s_0 s_2 s_3 s_3$, *the factor* $s_1 s_2 s_2$ *and the ending factor* $s_1 s_3 s_3$. *The predecessors of* s_1 *are* s_0 *and* s_2, *and its successors are* s_2 *and* s_3 *in an equal proportion. We need an equal number of* $s_0 s_1 s_2$ *and*

$s_0s_1s_3$ and similarly an equal number of $s_2s_1s_2$ and $s_2s_1s_3$. We thus need four occurrences of s_1.

W is equivalent to $W' = \{s_0s_1s_2s_2s_1s_3s_3,\ s_0s_2s_3s_3,\ s_0s_1s_2s_2s_1s_3s_3,\ s_0s_2s_3s_3\}$ (every run has been duplicated) and every block we gave is duplicated. Then applying the merging algorithm, we obtain $W'' = \{s_0s_1s_3s_3,\ s_0s_1s_2s_2s_1s_2s_2s_1s_3s_3,\ s_0s_2s_3s_3,\ s_0s_2s_3s_3\}$.

In Lemma 5.5, (1) ensures that $\hat{\mathcal{M}}_{W'} = \hat{\mathcal{M}}_W$ and (2) ensures the equality between the proportion of runs of W' passing by s and satisfying γ, denoted $\hat{\gamma}^s_{W'}$, and the probability of reaching s_F before s_0 starting from s with respect to $\hat{\mathcal{M}}_{W'}$.

Lemma 5.6. *For all $s \in S$, $\mathbb{P}_s^{\hat{\mathcal{M}}_{W'}}(\text{reach } s_f \text{ before } s_0) = \hat{\gamma}^s_{W'}$.*

Proof. Let S_0 be the set of states s with no path in $\hat{\mathcal{M}}_{W'}$ from s to s_f without passing through s_0. For all $s \in S_0$, let $p_s = 0$. Also, let $p_{s_f} = 1$. Let $S_1 = S \setminus (S_0 \cup \{s_f\})$. Consider the system of equations (5.4) with variables $(p_s)_{s\in S_1} \in [0,1]^{|S_1|}$:

$$\forall\, s \in S_1, \quad p_s = \sum_{t=1}^{m} \hat{\mathcal{M}}_{W'}(s,t)p_t \tag{5.4}$$

The system of equations (5.4) admits a unique solution [BK08] (theorem 10.19). Then, $(\mathbb{P}_s^{\hat{\mathcal{M}}_{W'}}(\text{reach } s_f \text{ before } s_0))_{s\in S_1}$ is trivially a solution of (5.4). But, since W' satisfies the conditions of Lemma 5.5, we also have that $(\hat{\gamma}^s_{W'})_{s\in S_1}$ is a solution of (5.4), and thus we have the desired equality. □

Notice that Lemma 5.6 does not hold in general with the set W. We have:

$$\begin{aligned}
\hat{\gamma}_W(A,\varphi) &= \hat{\gamma}^{s_0}_W \quad \text{(by definition)} \\
&= \hat{\gamma}^{s_0}_{W'} \quad \text{(by Lemma 5.5)} \\
&= \mathbb{P}_{s_0}^{\hat{\mathcal{M}}_{W'}}(\text{reach } s_f \text{ before } s_0) \quad \text{(by Lemma 5.6)} \\
&= \mathbb{P}_{s_0}^{\hat{\mathcal{M}}_{W}}(\text{reach } s_f \text{ before } s_0) \quad \text{(by Lemma 5.5)} \\
&= \gamma(\hat{\mathcal{M}}_W, \varphi) \quad \text{(by definition).}
\end{aligned}$$

That concludes the proof of Proposition 5.4.

It shows that learning can be as efficient as statistical model-checking on comparable properties.

Algorithm 5 Learning a matrix accurate for time-to-failure property

Learning($\mathcal{M}, s_0, s_F, \delta, \varepsilon$)

```
n_succ = 0
n = 1 (number of times s_0 has been visited)
s = s_0 (current state)
while n < H(n, n_succ,ϵ,δ) do
    ω_n = s_0 and W = ω_1 ··· ω_n
    while s ≠ s_0 or s ≠ s_F do
        Observe the next state s' (sampled with respect to A)
        Update ω_n and Â_W
        if s' = s_0 or s' = s_F then
            Output z(ω_n, φ), n_succ ← n_succ + z(ω_n, φ) and n ← n + 1
        end if
    end while
end while
return Â_W
```

5.2.3 Algorithm for the fixed time-to-failure property

A run ω is observed from s_0 and every time s_0 or s_F are observed, the reset operation is performed and a new path is being generated. W is the set of all those paths. Remember we assume that the probability of reaching s_0 or s_F is 1 in order to guarantee the termination of the algorithm.

5.3 Learning for the full CTL logic

In this section, we learn a Markov Chain $\hat{\mathcal{M}}_W$ such that $\hat{\mathcal{M}}_W$ and $\mathcal{M}$ have similar behaviors over all CTL formulas. This provides a much stronger result than on time-to-failure property, *e.g.*, , properties can involve liveness and fairness, and more importantly they are not known before the learning process. Notice that PCTL [HJ94] (that is an extension of CTL with probabilities) cannot be used, since an infinitesimal error on one > 0 probability can change the probability of a PCTL formula from 0 to 1. We recall that CTL has been defined in section 2.4.2.

As we want to compute the probability of paths satisfying a CTL formula, we consider formulas such that the highest operator is not quantified over paths (without **E** or **A**). That is, Ψ is the set of formulas φ of the form $\varphi = \mathbf{X}\varphi_1$, $\varphi = \varphi_1 \mathbf{U} \varphi_2$, $\varphi = \mathbf{F}\varphi_1$ or $\varphi = \mathbf{G}\varphi_1$, with φ_1, φ_2 CTL formulas. Notice that the property considered in the previous section is $(\neg s_0)\mathbf{U}s_F$.

In this section, for the sake of simplicity, the finite set W of traces is obtained by observing paths till a state is seen twice on the path. Then, the reset action is used and another trace is obtained from another path. That is, a trace ω from W is of the form $\omega = \rho \cdot s \cdot \rho' \cdot s$, with $\rho \cdot s \cdot \rho'$ a loop-free path.

We need an additional hypothesis. We assume that the support of transition probabilities is known, ie for any state i, we know the set of states j such that $a_{ij} \neq 0$. This assumption is needed both for Theorem 5.9 and to apply Laplace smoothing. We will show that this property is necessary in section 5.3.3.

5.3.1 No PAC bound for LTL

Inspired by the result given in [Dac+16] we prove that there is no learning algorithm that will give a Markov Chain that is accurate for all LTL formulas.

Theorem 5.7 ([Dac+16])**.** *Given $\varepsilon > 0$, $0 < \delta < 1$, and a finite set W of paths, there is no learning strategy such that, for all LTL formula φ,*

$$\mathbb{P}(|\gamma(\mathcal{M}, \varphi) - \gamma(\hat{\mathcal{M}}_W, \varphi)| > \varepsilon) \leq \delta \tag{5.5}$$

Proof. We prove it by defining a sequence of LTL properties that violates the specification above. As we show, it only relies on a single deviation in one transition. This is thus independent of the learning strategy.

Let $s_u \in S$ be a state that can be visited arbitrarily often from s_0 and let $s_v \in S$ be a non-unique successor of s_u. Assume that $\hat{\mathcal{M}}_W = (\hat{m}_{ij})_{1 \leq i,j \leq m}$ is an estimate of $\mathcal{M} = (m_{ij})_{1 \leq i,j \leq m}$ and note $\tau > 0$ the deviation between $\hat{m}_{uv}$ and m_{uv}. For simplicity, we assume $\hat{m}_{uv} = m_{uv} + \tau$ but a similar proof can be done with $\hat{m}_{uv} = m_{uv} - \tau$.

Let φ_n be the property "Transition $s_u s_v$ occurs at most $(m_{uv} + \tau/2)n$ times during the n first visits of s_i". This property is a LTL property since it can be written as a finite composition of $\mathbf{X}$, $\wedge$ and $\vee$. Let $(X_k)_{1 \leq k \leq n}$ be n independent Bernoulli random variables from the set of transitions possible in s_u to $\{0, 1\}$ assigning 1 when $s_u s_v$ is taken after the k-th visit of s_u and 0 if another transition is taken after the k-th visit of s_u. Then, we can rewrite:

$$\mathbb{P}(\varphi_n) = \mathbb{P}\left(\frac{1}{n}\sum_{k=1}^{n} X_k \leq m_{uv} + \tau/2\right) \tag{5.6}$$

By the law of large numbers, $\frac{1}{n}\sum_{k=1}^{n} X_k$ tends toward m_{uv} with respect to $\mathcal{M}$ when n

goes to infinity. Then,

$$\gamma(\mathcal{M}, \varphi_n) = \mathbb{P}_A\left(\frac{1}{n}\sum_{k=1}^{n} X_k \leq m_{uv} + \tau/2\right) \underset{n\to\infty}{\longrightarrow} 1.$$

But, with respect to $\hat{\mathcal{M}}_W$, $\frac{1}{n}\sum_{k=1}^{n} X_k$ tends toward $m_{uv} + \tau$. So,

$$\gamma(\hat{\mathcal{M}}_W, \varphi_n) = \mathbb{P}_{\hat{\mathcal{M}}_W}\left(\frac{1}{n}\sum_{k=1}^{n} X_k \leq m_{uv} + \tau/2\right) \underset{n\to\infty}{\longrightarrow} 0$$

Thus, $\gamma(\mathcal{M}, \varphi_n) - \gamma(\hat{\mathcal{M}}_W, \varphi_n) \underset{n\to\infty}{\longrightarrow} 1$ almost surely. More precisely, given $\varepsilon > 0$, δ, $0 < \delta < 1$ and a finite run W, there exists a rank N such that specification 5.5 can not be fulfilled for properties φ_n, $n \geq N$. □

5.3.2 Conditioning and Probability Bounds

Using Laplace smoothing slightly changes the probability of each transition by say an additive offset η. We now explain how this small error η impacts the error on the probability of a CTL property.

Let $\mathcal{M}$ be a Markov Chain, and $\mathcal{M}_\eta$ be a Markov Chain such that $M_\eta(i,j) \neq 0$ iff $M(i,j) \neq 0$ for all states s_i, s_j, and such that $\sum_j |M_\eta(i,j) - M(i,j)| \leq \eta$ for all state s_i. For all state $s \in S$, let $R(s)$ be the set of states s_i such that there exists a path from s_i to s. Let $R_*(s) = R(s) \setminus \{s\}$. Since both Markov Chains have the same support, R (and also R_*) is equal for A and A_η. Given m the number of states, we define the conditioning of $\mathcal{M}$ for $s \in S$ and $\ell \leq m$ as follows:

Definition 5.3 (Conditioning)**.**

$$Cond_s^\ell(\mathcal{M}) = \min_{i \in R_*(s)} \mathbb{P}_i^{\mathcal{M}}(\mathbf{F}_{\leq \ell} \neg R_*(s)) \tag{5.7}$$

i.e., the minimal probability from state $i \in R_*(s)$ to move away from $R_*(s)$ in at most ℓ steps. Let ℓ_s minimal such that $\mathrm{Cond}_s^{\ell_s}(\mathcal{M}) > 0$. This minimal ℓ_s exists as $\mathrm{Cond}_s^{m}(\mathcal{M}) > 0$ since, for all $s \in S$ and $i \in R_*(s)$, there is at least one path reaching s from i (this path leaves $R_*(s)$), and taking a cycle-free path, we obtain a path of length at most m. Thus, the probability $\mathbb{P}_i^{\mathcal{M}}(\mathbf{F}_{\leq m} \neg R_*(s))$ is at least the positive probability of the cylinder defined by this finite path.

Theorem 5.8. *Denoting φ the property of reaching state s in Markov Chain $\mathcal{M}$, we have:*

$$|\gamma(\mathcal{M},\varphi) - \gamma(\mathcal{M}_\eta,\varphi)| < \frac{\ell_s \cdot \eta}{Cond_s^{\ell_s}(\mathcal{M})}$$

Proof. Let v_s be the stochastic vector with $v_s(s) = 1$. We denote $v_0 = v_{s_0}$. Let $s \in S$. We assume that $s_0 \in R_*(s)$ (else $\gamma(\mathcal{M},\varphi) = \gamma(\mathcal{M}_\eta,\varphi)$ and the result is trivial). Without loss of generality, we can also assume that $M(s,s) = M_\eta(s,s) = 1$ (as we are interested in reaching s at any step). With this assumption:

$$|\gamma(A,\varphi) - \gamma(A_\eta,\varphi)| = \lim_{t\to\infty} v_0 \cdot (A^t - A_\eta^t) \cdot v_s$$

We bound this error, through bounding by induction on t:

$$E(t) = \max_{i\in R_*(s)} v_i \cdot (M^t - M_\eta^t) \cdot v_s$$

We then have trivially:

$$|\gamma(\mathcal{M},\varphi) - \gamma(\mathcal{M}_\eta,\varphi)| \leq \lim_{t\to\infty} E(t)$$

Note that for $i = s$, $\lim_{t\to\infty} v_i \cdot (M^t) \cdot v_s = 1 = \lim_{t\to\infty} v_i \cdot M_\eta^t \cdot v_s$, and thus their difference is null.

Let $t \in \mathbb{N}$. We let $j \in R_*(s)$ such that $E(t) = v_j \cdot (M^t - M_\eta^t) \cdot v_s$.

By the triangular inequality, introducing the term $v_j \cdot M^{\ell_s} M_\eta^{t-k} \cdot v_s - v_j \cdot M^{\ell_s} M_\eta^{t-k} \cdot v_s = 0$, we have:

$$E(t) \leq |v_j \cdot (M_\eta^t - M^{\ell_s} M_\eta^{t-\ell_s}) \cdot v_s| + |(v_j \cdot M^{\ell_s}) \cdot (M_\eta^{t-\ell_s} - M^{t-\ell_s}) \cdot v_s|$$

We separate vector $(v_j \cdot M^m) = w_1 + w_2 + w_3$ in three sub-stochastic vectors w_1, w_2, w_3: vector w_1 is over $\{s\}$, and thus we have $w_1 \cdot M_\eta^{t-m} = w_1 = w_1 \cdot M^{t-\ell_s}$, and the term cancels out. Vector w_2 is over states of $R_*(s)$, with $\sum_{i\in R_*} w_2[i] \leq (1 - \mathrm{Cond}_s^{\ell_s}(A))$, and we obtain an inductive term $\leq (1 - \mathrm{Cond}_s^{\ell_s}(\mathcal{M}))E(t-\ell_s)$. Last, vector w_3 is over states not in $R(s)$, and we have $w_3 \cdot M_\eta^{t-\ell_s} \cdot v_s = 0 = w_3 \cdot M^{t-\ell_s} \cdot v_s$, and the term cancels out.

We also obtain that $|v_j \cdot (M_\eta^t - M^{\ell_s} M_\eta^{t-\ell_s}) \cdot v_s| \leq \ell_s \cdot \eta$. Thus, we have the inductive formula $E(t) \leq (1 - \mathrm{Cond}_s^{\ell_s}(\mathcal{M}))E(t-\ell_s) + \ell_s \cdot \eta$. It yields for all $t \in \mathbb{N}$:

$$E(t) \leq (\ell_s \cdot \eta) \sum_{i=1}^{\infty} (1 - \mathrm{Cond}_s^{\ell_s}((M)))^i$$

$$E(t) \leq \frac{\ell_s \cdot \eta}{Cond_s^{\ell_s}(\mathcal{M})}$$

□

We can extend this result from reachability to formulas of the form $S_0 \mathbf{U} S_F$, where S_0, S_F are subsets of states. This formula means that we reach the set of states S_F through only states in S_0 on the way.

We define $R(S_0, S_F)$ to be the set of states which can reach S_F using only states of S_0, and $R_*(S_0, S_F) = R(S_0, S_F) \setminus S_F$. For $\ell \in \mathbb{N}$, we let:

$$\mathrm{Cond}^{\ell}_{S_0,S_F}(\mathcal{M}) = \min_{i \in R_*(S_0,S_F)} \mathbb{P}_i^{\mathcal{M}}(\mathbf{F}_{\leq \ell} \neg R_*(S_0, S_F) \vee \neg S_0).$$

Now, one can remark that $\mathrm{Cond}_{S_0,S_F}(\mathcal{M}) \geq \mathrm{Cond}_{S,S_F}(\mathcal{M}) > 0$. Let $\mathrm{Cond}^{\ell}_{S_F}(\mathcal{M}) = \mathrm{Cond}^{\ell}_{S,S_F}(\mathcal{M})$. We have $\mathrm{Cond}^{\ell}_{S_0,S_F}(\mathcal{M}) \geq \mathrm{Cond}^{\ell}_{S_F}(\mathcal{M})$. As before, we let $\ell_{S_F} \leq m$ be the minimal ℓ such that $\mathrm{Cond}^{\ell}_{S_F}(\mathcal{M}) > 0$, and obtain:

Theorem 5.9. *Denoting φ the property $S_0 \mathbf{U} S_F$, we have, given Markov Chain $\mathcal{M}$:*

$$|\gamma(\mathcal{M}, \varphi) - \gamma(\mathcal{M}_\eta, \varphi)| < \frac{\ell_{S_F} \cdot \eta}{Cond_{S_F}^{\ell_{S_F}}(\mathcal{M})}$$

We defined the conditioning as the probability to reach S_F or $S \setminus R(S, S_F)$. At the price of a more technical proof, we can obtain a better bound by replacing S_F by the set of states $R_1(S_F)$ that have probability 1 to reach S_F. We let $\overline{R_*}(S_F) = R(S, S_F) \setminus R_1(S_F)$ the set of states that can reach S_F with < 1 probability, and

$$\overline{\mathrm{Cond}}^{\ell}_{S_F}(\mathcal{M}) = \min_{i \in \overline{R_*}(S_F)} \mathbb{P}_i^{\mathcal{M}}(\mathbf{F}_{\leq \ell} \neg \overline{R_*}(S_F))$$

5.3.3 Optimality and necessity of knowing the transitions support

We show now that the bound we provide in Theorems 5.8 and 5.9 are close to optimal, and that the hypothesis on $M(i,j) \neq 0$ iff $M_\eta(i,j) \neq 0$ is necessary.

Let us consider Markov Chains $\mathcal{M}, \hat{\mathcal{M}}, \hat{\mathcal{M}}'$ in Fig. 5.4 and formula $\mathbf{F}\, s_2$ stating that s_2 is eventually reached. The probabilities to satisfy this formula in $\mathcal{M}, \hat{\mathcal{M}}, \hat{\mathcal{M}}$ are respectively $\mathbb{P}^{\mathcal{M}}(\mathbf{F}\, s_2) = \frac{1}{2}$, $\mathbb{P}^{\hat{\mathcal{M}}}(\mathbf{F}\, s_2) = \frac{2\tau+\eta}{4\tau}$ and $\mathbb{P}^{\hat{\mathcal{M}}'}(\mathbf{F}\, s_2) = 0$.

Assume that $\mathcal{M}$ is the real system and that $\hat{\mathcal{M}}$ and $\hat{\mathcal{M}}'$ are Markov Chains we learned from $\mathcal{M}$.

As we do not know precisely the transition probabilities in $\mathcal{M}$, we can only compute the conditioning on $\hat{\mathcal{M}}$ and not on $\mathcal{M}$. We have $R(s_2) = \{s_1, s_2\}$ and $R_*(s_2) = \overline{R_*}(s_2) = \{s_1\}$. The probability to stay in $R_*(s_2)$ after $\ell_{s_2} = 1$ step is $(1 - 2\tau)$, and thus $\mathrm{Cond}^1_{\{s_2\}}(\hat{\mathcal{M}}) = \overline{\mathrm{Cond}}^1_{\{s_2\}}(\hat{\mathcal{M}}) = 1-(1-2\tau) = 2\tau$. Taking $\mathcal{M}_\eta = \hat{\mathcal{M}}$, Theorem 5.9 tells us that $|\mathbb{P}^{\mathcal{M}}(\mathbf{F}\, s_2) - \mathbb{P}^{\hat{\mathcal{M}}}(\mathbf{F}\, s_2)| \leq \frac{\eta}{2\tau}$. Notice that on that example, even using $\ell_{s_2} = m = 3$, we obtain $\mathrm{Cond}^3_{\{s_2\}}(\hat{\mathcal{M}}) = 1 - (1 - 2\tau)^3 \approx 6\tau$, and we find a similar bound $\approx \frac{3\eta}{6\tau} = \frac{\eta}{2\tau}$.

Compare our bound with the exact difference $|\mathbb{P}^{\mathcal{M}}(\mathbf{F}\, s_2) - \mathbb{P}^{\hat{\mathcal{M}}}(\mathbf{F}\, s_2)| = \frac{2\tau+\eta}{4\tau} - \frac{1}{2} = \frac{\eta}{4\tau}$. Our upper bound only has an overhead factor of 2, considering this example is a case where the conditioning is particularly bad.

Without knowing that there are transitions from s_1 to s_2 and from s_1 to s_3, the chance to only witness the transition from s_1 to s_1 (and consequently to learn MC $\hat{\mathcal{M}}'$) is high if the inverse of τ is large enough compared to the number of observations. For $\hat{\mathcal{M}}'$, we have $\mathrm{Cond}^\ell_{S,s_2}(\hat{\mathcal{M}}') = 1$ for all $\ell > 0$. Let $\eta = 2\tau$ and $\mathcal{M}_\eta = \hat{\mathcal{M}}$. Now, there is no function of η and $\mathrm{Cond}^\ell_{S,s_2}(\hat{\mathcal{M}}')$ (without creating a moot bound of value at least $\frac{1}{2}$ for all $\eta, \mathrm{Cond}(\hat{\mathcal{M}}')$ which could bound the difference $|\mathbb{P}^{\mathcal{M}}(F\, s_2) - \mathbb{P}^{\hat{\mathcal{M}}'}(\mathbf{F}\, s_2)| = \frac{1}{2}$. Hence, the hypothesis on $M(i,j) \neq 0$ iff $M_\eta(i,j) \neq 0$ is necessary, which requires to know the support of the transitions of the real system $\mathcal{M}$.

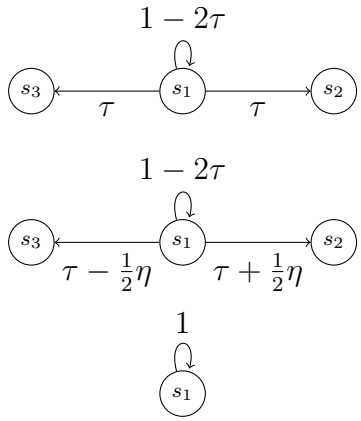

Figure 5.4: Three MCs $\mathcal{M}, \hat{\mathcal{M}}, \hat{\mathcal{M}}'$ (from top to bottom), with $0 < \eta < 2\tau < 1$

5.3.4 PAC bounds for $\sum_j |\hat{A}_W(i,j) - A(i,j)| \leq \eta$

As in Section 5.2, we use the algorithm of Chen in order to obtain PAC bounds. However, we do not use it to estimate a property, but rather the individual transition probabilities.

Let W be a set of traces drawn with respect to $\mathcal{M}$ such that every $\omega \in W$ is of the form $\omega = \rho \cdot s \cdot \rho' \cdot s$. For recall, for each state s_i, s_j of S, n_i^W is the number of transitions originating from s_i in W and n_{ij}^W is the number of transition $s_i s_j$ in W. Let $\delta' = \frac{\delta}{m_{\text{stoch}}}$, where m_{stoch} is the number of *stochastic* states, *i.e.*, with at least two outgoing transitions.

We want to sample traces until the empirical transition probabilities $\frac{n_{ij}^W}{n_i^W}$ are relatively close to the exact transition probabilities m_{ij}, for all $i, j \in S$. For that, we need to determine a stopping criteria over the number of state occurrences $(n_i)_{1 \leq i \leq m}$ such that:

$$\mathbb{P}\left(\exists i \in S, \sum_j \left| m_{ij} - \frac{n_{ij}^W}{n_i^W} \right| > \varepsilon\right) \leq \delta$$

First, note that for any observed state $s_i \in S$, if $m_{ij} = 0$ (or $a_{ij} = 1$), then with probability 1, $\frac{n_{ij}^W}{n_i^W} = 0$ (respectively $\frac{n_{ij}^W}{n_i^W} = 1$). Thus, for all $\varepsilon > 0$, $|m_{ij} - \frac{n_{ij}^W}{n_i^W}| < \varepsilon$ with probability 1. Second, for two distinct states s_i and $s_{i'}$, the transition probabilities $\frac{n_{ij}^W}{n_i^W}$ and $\frac{n_{i'j'}^W}{n_{i'}^W}$ are independent for all $j \neq j'$.

Let $s_i \in S$ be a stochastic state. If we observe n_i^W transitions from s_i such that $n_i^W \geq \frac{2}{\varepsilon^2} \log\left(\frac{2}{\delta'}\right) \left[\frac{1}{4} - \left(\max_j |\frac{1}{2} - \frac{n_{ij}^W}{n_i^W}| - \frac{2}{3}\varepsilon\right)^2\right]$, then, according to Theorem 5.2,

$$\mathbb{P}\left(\bigvee_{j=1}^{m} |m_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) \leq \delta'$$

In particular, $\mathbb{P}\left(\max_{j \in S} |m_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) \leq \delta'$. Moreover, we have:

$$\begin{aligned}
\mathbb{P}\left(\bigvee_{i=1}^{m} \max_{j \in S} |m_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) &\leq \sum_{i=1}^{m} \mathbb{P}\left(\max_{j \in S} |a_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) \\
&\leq m_{\text{stoch}} \delta' \\
&\leq \delta
\end{aligned}$$

In other words, the probability that "there exists a state $s_i \in S$ such that the deviation between the exact and empirical outgoing transitions from s_i exceeds ε" is bounded by δ

as soon as for each state $s_i \in S$, n_i^W satisfies the stopping rule of the algorithm of Chen using ε and the corresponding δ'. This gives the hypothesis $\sum_j |M_\eta(i,j) - M(i,j)| \leq \epsilon$ for all state s_i.

5.3.5 A Matrix $\hat{\mathcal{M}}_W$ accurate for all CTL properties

We now use Laplace smoothing in order to ensure the other hypothesis $M_\eta(i,j) \neq 0$ iff $M(i,j) \neq 0$ for all states s_i, s_j. For all $s_i \in S$, we define the Laplace offset depending on the state s_i as $\alpha_i = \frac{(n_i^W)^2 \varepsilon}{10 \cdot k_i^2 \max_j n_{ij}^W}$, where k_i is the number of transitions from state s_i. This ensures that the error from Laplace smoothing is at most one tenth of the statistical error. Let $\alpha = (\alpha_i)_{1 \leq i \leq m}$. From the sample set W, we output the matrix $\hat{\mathcal{M}}_W^\alpha = (\hat{m}_{ij})_{1 \leq i,j \leq m}$ with Laplace smoothing α_i for state s_i, *i.e.*, :

$$\hat{m}_{ij} = \frac{n_{ij}^W + \alpha_i}{n_i^W + k_i \alpha_i} \text{ if } m_{ij} \neq 0 \quad \text{and} \quad \hat{m}_{ij} = 0 \text{ otherwise}$$

It is easy to check that we have for all $s_i, s_j \in S$:

$$\left| \hat{m}_{ij} - \frac{n_{ij}^W}{n_i^W} \right| \leq \frac{\varepsilon}{10 \cdot k_i}$$

That is, for all state s_i, $\sum_j \left| \hat{m}_{ij} - \frac{n_{ij}^W}{n_i^W} \right| \leq \frac{\varepsilon}{10}$. Using the triangular inequality:

$$\mathbb{P}\left(\exists i \in S, \sum_j |a_{ij} - \hat{m}_{ij}| > \frac{11}{10} \varepsilon \right) \leq \delta$$

For all $s_i \in S$, let $H^*(n_i^W, \epsilon, \delta') = \max_{s_j \in S} H(n_i^W, n_{ij}^W, \epsilon, \delta')$ be the maximal Chen bound over all the transitions from state s_i. Let $B(\hat{\mathcal{M}}_W^\alpha) = \max_{S_F} \frac{\ell_{S_F}}{\text{Cond}_{S_F}^{\ell_{S_F}}(\hat{\mathcal{M}}_W^\alpha)}$. Applying Theorem 5.9, we obtain that:

Theorem 5.10. *Given a set W of traces, for $0 < \epsilon < 1$ and $0 < \delta < 1$, if for all $s_i \in S$, $n_i^W \geq \left(\frac{11}{10} B(\hat{\mathcal{M}}_W^\alpha)\right)^2 H^*(n_i^W, \epsilon, \delta')$, we have for any CTL property φ:*

$$\mathbb{P}(|\gamma(\mathcal{M}, \varphi) - \gamma(\hat{\mathcal{M}}_W^\alpha, \varphi)|) > \varepsilon) \leq \delta \tag{5.8}$$

Proof. First, $\hat{m}_{ij} \neq 0$ iff $m_{ij} \neq 0$, by definition of $\hat{\mathcal{M}}_W^\alpha$. Second, $\mathbb{P}(\exists i, \sum_j |m_{ij} - \hat{m}_{ij}| > \frac{11}{10}\varepsilon) \leq \delta$. We can thus apply Theorem 5.9 on $\hat{\mathcal{M}}_W^\alpha, \mathcal{M}$ and obtain (5.8) for φ any formula of the form $S_1 \mathbf{U} S_2$. For recall, we only need to prove the result for properties without $\mathbf{E}$

or **A**. It remains to show that for any formula $\varphi \in \Psi$, we can define $S_1, S_2 \subseteq S$ such that φ can be expressed as $S_1 \mathbf{U} S_2$.

Consider the different cases: If φ is of the form $\varphi = \varphi_1 \mathbf{U} \varphi_2$ (it subsumes the case $\varphi = \mathbf{F}\varphi_1 = \top \mathbf{U} \varphi_1$) with φ_1, φ_2 CTL formulas, we define S_1, S_2 as the sets of states satisfying φ_1 and φ_2, and we have the equivalence (see [BK08] for more details). If $\varphi = X\varphi_2$, define $S_1 = \emptyset$ and S_2 as the set of states satisfying φ_2.

The last case is $\varphi = \mathbf{G}\varphi_1$, with φ_1 a CTL formula. Again, we define S_1 the set of states satisfying φ_1, and S_2 the set of states satisfying the CTL formula $\mathbf{AG}\varphi_1$. The probability of the set of paths satisfying $\varphi = \mathbf{G}\varphi_1$ is exactly the same as the probability of the set of paths satisfying $S_1 \mathbf{U} S_2$. □ □

5.4 Evaluation and Discussion

In this section, we evaluate Algorithm 6 on 5 crafted systems and discuss its practical use. The objective of the evaluation is to provide some idea on how many samples would be sufficient for learning accurate MC estimations. We now describe the 5 systems:

Systems 1 and 2 are three-state models described in Fig. 5.5 and Fig. 5.6. Systems

Algorithm 6 Learning a matrix accurate for CTL

Data: $\mathcal{S}, s_0, \delta, \varepsilon$
$W := \emptyset$
$m = |S|$
for all $s \in S$ **do**
  $n_s^W := 0$
end for
Compute $\hat{\mathcal{M}} := \hat{\mathcal{M}}_W^\alpha$
Compute $B := B(\hat{\mathcal{M}})$
while $\exists s \in S, n_s^W < \left(\frac{11}{10} B(\hat{\mathcal{M}})\right)^2 H^*(n_s^W, \epsilon, \frac{\delta}{m})$ **do**
  Generate a new trace $\omega := s_0 \, \rho \, s_1 \, \rho' \, s_1$, and reset $\mathcal{S}$
  for all $s \in S$ **do**
    $n_s^W := n_s^W + n_s^{\{\omega\}}$
  end for
  add ω to W
  Compute $\hat{\mathcal{M}} := \hat{\mathcal{M}}_W^\alpha$
  Compute $B := B(\hat{\mathcal{M}})$
end while
return $\hat{\mathcal{M}}_W^\alpha$

3 (resp. 5) is a 30-state (resp. 200-states) clique in which every individual transition probability is 1/30 (resp. 1/200). System 4 is a 64-state system modeling failure and repair of 3 types of components (3 components each, 9 components in total). System 4 can be modeled with probabilistic model checker Prism[1] as a continuous time Markov chain (CTMC) that comprises three types (1, 2, 3) of three components each that may fail independently. Note however that we do not simulate the times between two changes of states but only the transitions between states, that lead to learn the induced MC instead. The components fail with rate $\lambda = 0.2$ and are repaired with rate $\mu = 1$. In addition, components are repaired with priority according to their type (type i has highest priority than type j if $i < j$). Components of type 1 and 2 are repaired simultaneously if at least two of their own type have failed. Type 3 components are repaired one by one as soon as one has failed. The probability transitions from state s_i to state s_j is given by the rate of the transition from the CTMC between state s_i and state s_j divided by the sum of all the rates of the enabled transitions from state s_i. The initial state is the state in which all the components are operational and the failure state is the state in which all the components are broken. We provide below the Prism code of the model for the readers

1. http://www.prismmodelchecker.org/

	System 1	System 2	System 3	System 4	System 5
# states	3	3	30	64	200
# transitions	4	7	900	204	40000
# events for time-to-failure	191 (16%)	991 (10%)	2753 (7.4%)	1386 (17.9%)	18335 (7.2%)
# events for full CTL	1463 (12.9%)	4159 (11.7%)	8404 (3.8%)	1872863	79823 (1.7%)

Table 5.1: Average number of observed events N (and relative standard deviation) given $\epsilon = 0.1$ and $\delta = 0.05$ for a time-to-failure property and for the full CTL logic using the refined conditioning $\overline{\text{Cond}}$.

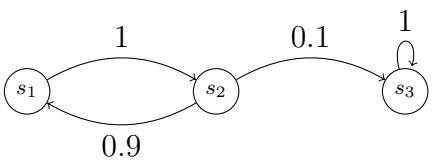

Figure 5.5: An example of MC $\mathcal{M}_1$

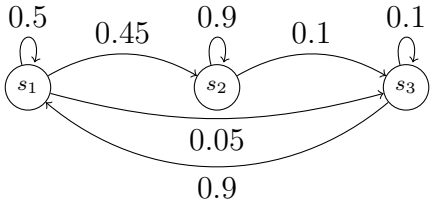

Figure 5.6: MC $\mathcal{M}_2$

who are interested to investigate this model in details:

```
ctmc

const int n=3;
const double lambda = 0.2;
const double mu = 1.0;

module type1
state1 : [0..n] init 0;
[] state1 < n -> (n-state1)*lambda : (state1'=state1+1);
[] state1 >=2 -> mu : (state1'=0);
endmodule

module type2
state2 : [0..n] init 0;
[] state2 < n -> (n-state2)*lambda : (state2'=state2+1);
[] state2 >=2 & state1 < 2 -> mu : (state2'=0);
endmodule

module type3
state3 : [0..n] init 0;
[] state3 < n -> (n-state3)*lambda : (state3'=state3+1);
[] state3 > 0 & state2 < 2 & state1 < 2 -> mu : (state3'=state3-1);
endmodule

label "failure" = state1 = n & state2 = n & state3 = n;
```

We tested time-to-failure properties by choosing as failure states s_3 for Systems 1, 2, 3, 5, and the state where all 9 components fail for System 4. We also tested Algorithm 1 (for full CTL logic) using the refined conditioning $\overline{\text{Cond}}$. We performed our algorithms 100 times for each model, except for full CTL on System 4, for which we only tested once since it is very time-consuming. We report our results in Table 5.1 for $\epsilon = 0.1$ and $\delta = 0.05$. In particular, we output for each model its number of states and transitions. For each (set of) property, we provide the average number of observations and the relative standard deviation.

The results show that for systems of average size, we can learn MCs which are accurate for all CTL formulas, although for some systems such as System 4, it can take a lot of events to be observed before Algorithm 6 terminates. The reason is that there are rare states, such as the state where all 9 components fail, which are observed with an extremely small probability. In order to evaluate the probabilities of CTL properties of the form: "if all 9 components fail, then CTL property φ is satisfied", this state needs to be explored many times, explaining the high number of events observed before the algorithm terminates. On the other hand, for properties that do not involve the 9 components failing as prior, such as time-to-failure, one does not need to observe this state even once to conclude that it has an extremely small probability to happen. This suggests that efficient algorithms could be developed for subsets of CTL formulas, *e.g.*, , in defining a subset of important events to consider. We believe that Theorem 4 and 5 could be extended to handle such cases.

Comparing results for time-to-failure and for the full CTL logic is interesting. Excluding System 4 which involves rare states, the number of events that needs to be observed varies between 4.3 to 7 times more, even for the model with 200 states. Surprisingly, the highest difference is with the smallest System 1. It is because every run of System 1 simulated for time-to-failure has size 3 ($s_1 s_2$ and either s_1 or s_3). However, in Systems 2, 3 and 5, samples for time-to-failure can be much longer (*i.e.*, s_1 or s_3 are seen). In comparison, every event observed in Algorithm 6 is used to estimate $\hat{A}^{\alpha}_{W}$.

Notice that for the system we tested, Cond was particularly large (more than 20) because for many states s, there was probability 0 to leave $R(s)$, and hence $\ell(s)$ was quite large. These are the cases where $\overline{\text{Cond}}$ is much more efficient, as then we can choose $\ell(s) = 1$ as the probability to reach s from states in $R(s)$ is 1 ($R_1(s) = R(s)$ and $\overline{R_*(s)} = \emptyset$). We used $\overline{\text{Cond}}$ in our algorithm.

5.5 Related work

This work lies at the crossroads of machine learning and Statistical Model Checking (SMC) [YS02]. However, the context and the outputs are different. SMC is a simulation-based approach that aims to infer conclusions about properties using probability estimation or hypothesis testing [Che+52; Wal45], within acceptable margins of error and confidence [Hér+04; JSS17; ZPC13]. A challenge in SMC is posed by unbounded properties (*e.g.*, , fairness) since the sampled executions are finite. Some algorithms have been proposed to handle unbounded properties but they require at least the knowledge of the

minimal probability transition of the system [Dac+17].

Another concern is the analysis of unknown or real-world systems. SMC algorithms have been proposed for black-box systems [SVA04] but providing statistical evidence remained questionable. The alternative to learn MC models from the system in order to reuse it for PMC has been posed in [Che+12; BS13; Brá+14; Wan+17; CPS18] but these approaches remain empirical and, contrary to this work, no analysis of the learning process is done there. In [Gho+17], the authors propose to analyze the learned model a posteriori to test whether it has some good properties. If not, then they tweak the model in order to enforce these properties.

Finally, in [Dac+16] that we already cited for the LTL result, the authors investigate several distances for the estimation of the difference between MCs, but they do not propose algorithms for learning. Also, several PAC-learning algorithms have been proposed for the estimation of stochastic systems [CG08; CT04] but these works focus on local transitions instead of global properties.

5.6 Conclusion

5.6.1 Summary

In this chapter, we have provided some foundations for certification of the learning of Markov Chains. Section 5.1 has provided a state of the art on the different kind of estimators for this learning and the mathematical notions around PAC learning.

In section 5.2 we investigated a first subproblem, that is the time to failure properties. We provided an algorithm with a certification on the likeliness to have a precise answer. Then, section 5.3 tackled a more general problem: providing bounds for all formula in some temporal logic. We saw that it was impossible to it for LTL, but we obtained a positive result for CTL. This is accompanied by an algorithm and a proof for the bounds. Moreover, we proved that our bounds are asymptotically tight: the use of conditioning is needed, up to a constant factor.

Finally, some proof of concept is shown through evaluations on different systems in section 5.4. We saw that the number of observations needed to have a good approximation stays reasonable in most cases. However, the problem of rare events can still require many observations in order to gather enough information on it (as one could have expected).

5.6.2 Future work

The field of certification of learning is quite recent and a lot of work remains to be done. In this chapter, we presented a special case, where we have strong assumptions. However, we believe that these studies should be extended to more complex models. In particular, one can think about Deep Neural Networks. Recent studies tackled the issue of their certification using various techniques such as abstract interpretation [Gop+18], numerical analysis [PT11; Dut+18], approximation with polyedras [Geh+18]... However, the size of DNN that can be certified is still some orders of magnitude smaller than the one industrially used (thousands of nodes vs millions of nodes). For these reasons, bringing more formal methods in the field of artificial intelligence may be beneficial.

Chapter 6

Conclusion

6.1 Contributions

This thesis aimed at better assessing stochastic systems. We developed algorithms focused on the quantification of various problems while reasoning on partial information in different contexts.

The first context was by defining diagnosability degrees of stochastic systems in Chapter 3. There were two quantities we evaluated: one is the probability that diagnosability holds after some time or eventually. The second is the detection delay distribution. By enriching the algorithm provided for computing the probability of diagnosis, we obtained a way to compute an arbitrary high number of moments of the distribution of detection delays. These moments allow us either to approximate or to provide accurate concentration bounds on this distribution.

A second context was by exploring how to distinguish between several stochastic systems based on a sequence of observations (Chapter 4). To take into account the partial information, we extended stationary distributions from Markov Chains to Labeled Markov Chains, by considering the Markov Chain induced by the restriction to a belief state. This extension was the first contribution, with proofs of its soundness. We believe it can have other applications. Then, a new proof based on this new notion of stationary distributions was presented for solving limit-sure classifiability in PTIME. Finally, we also established a link between contributions from different communities on equivalent/related notions: distinguishability, misclassification and limit-sure classifiability.

Third and finally, we focused on a problem that is "upstream": before reasoning on a stochastic model, how do we obtain these probabilities? We focused on the guarantees we could obtain given some estimation method for transition probabilities (Chapter 5). A huge difference with existing works on this subject is that we specifically aimed at global guarantees instead of only looking at local deviations on the transition probabilities. We first looked at a time to failure setting in a restrictive framework, and then we studied if and how we could guarantee that an estimated model behaves as the original system on

all formulas of a temporal logic. On that subject, we obtained a negative result with LTL and a positive one for CTL, and showed that it can be used in practice on average size stochastic systems.

6.2 Perspectives

Despite numerous works concerning the topics of this thesis that have been done in the last decade, many questions remain open. We start by presenting some short term perspectives, directly extending the thesis work before concluding with medium and long term perspectives.

Complexity and scalability: When considering different problems, especially diagnosability and learning, we gave worst-cases complexities/guarantees. However, we saw that the worst case is far from being the norm. A practical direction to investigate would be to determine how well these techniques scale up. Some classes of models/scenarios could be defined and different benchmarks and evaluations could be performed. For example, we gave in Chapter 3 some heuristics to help with the size of the diagnoser, but we did not give any practical evaluation. While an example showed that these heuristics could be very useful, we do not quantify "how much" or "how often".

Opacity: Opacity is a framework for stating properties about the potential leakage of some secret. It can be seen in two ways. The first one is asymmetrical opacity where one wants to be detect when a predicate holds, and the second one is symmetrical opacity where one wants to be certain whether a predicate holds or does not hold. For stochastic systems, the notion of asymmetric opacity is similar to diagnosability, as we want to decide if an event representing a leakage has been detected, and symmetric opacity is related to classifiability, where we want to decide if we are in the language where the predicate is true or in the language where the predicate is false. Similarly to the developments in Chapter 3, the quantification of "how often and how fast can one decide" can be considered. This is close to the *liberal* direction in [BMS15] and our techniques could be directly applied there. Another quantification one could imagine is related to security: while we may never be 100% sure that the secret is leaked, having a high level of confidence about a leakage can be critical in a security context. This direction is called *restrictive* in [BMS15]. However one has to be careful: many decision problems about almost certain diagnosability are undecidable (as described in [BHL16a]). For example, it could be hard to automatically

quantify the frequency we are 95% sure there was a leakage.

Extending the uncertainty: Regarding partial information systems, we have considered Labeled Markov Chains and Probabilistic Finite Automata, where the uncertainty was on the transitions: the underlying structure was non-deterministic. However, we considered that the probabilities were perfectly known, which is a strong assumption, as seen in Chapter 5. An interesting perspective could be to consider classes of models allowing imprecise probabilities. Several classes of models exist, especially the Interval Markov Chains [KU02] (IMCs) where probabilities may not be well known and lie in an interval. Similarly to LMCs, there exist labeled IMCs named Interval Labeled Markov Chains (ILMCs) [SVA06]. Several semantics exist for ILMCs, such as Interval Markov Decision Processes, where the actions are bound to choose the transition probabilities and Uncertain Markov Chains where the probabilities are unknown but lie in some interval and will not change during the execution. It could be interesting to extend the questions of quantification to these models and explore their decidability and complexity.

Security and confidentiality: In Chapter 4.5, we considered an attacker that had some (simple) power on the system. This notion of security with respect to an attacker has been used in several fields, such as opacity [DDM08]. We believe this could be extended for our work. We considered an attacker that was mostly passive with one action (the reset). What could be interesting is to explore different classes of attacks.

Now, we turn to longer term perspectives: what are the challenges that await us?

Trade-off between performances and guarantees: Different applications bring different needs. Critical applications leave no flexibility, while non-crucial problems may be handled differently. For this last category, it may be more important to have an answer quickly than 100% accurate. For example, one may imagine that a protocol supervising possible failures/faults answers "no" meaning that either there is no fault now or no incoming problem soon: the constraint has been relaxed, allowing the protocol to answer more quickly. This is an interesting field: while one may trade accuracy for efficiency, one may still wish for guarantees on this trade-off, either in terms of loss of precision or in terms of saved time/resources, *e.g.*, under the form of bounds on the amount of saved time.

Convergence between Formal Methods and AI: For many years, communities in Formal Methods and Artificial Intelligence have been disjoint and worked on related subjects with very different techniques. Drawbacks of both paradigms have been mentioned before, such as a lack of scalability for formal methods and the difficulties to obtain guarantees and confidence for AI. However, these last years have seen both communities work together. The contribution in Chapter 5 is an example of a work concerning both communities. The goals of these collaborations are diverse:

- A first goal is the evaluation of strengths and weaknesses of both communities. For each (category of) problem, which approach gives the best results with several criteria: efficiency, accuracy, robustness... For example, when considering image classification, AI-based techniques are the most efficient, even with the issues we raised (*e.g.*, in the introduction). However, they offer no guarantees whereas formal methods, especially verification are based on having formal guarantees.

- A second goal is to develop formal methods to address classical problems in AI. This is the point of our contribution in Chapter 5. The strength of Formal Methods is the guarantees it gives. Then, as we said before, one would like to apply these techniques to obtain guarantees on the tools that are used in AI. As an example, there are a lot of recent works whose goal is to certify Deep Neural Networks such as [Hua+17; WHK18].

- A third goal is how to use the tools in AI to help Formal Methods. For example, one can think about how to find a good heuristic in a theorem prover. This is a computationally difficult problem where the answer does not need to be optimal, as long as it is "good".

For these reasons, this convergence has a lot of applications, both academic and industrial. One can expect that this convergence will be beneficial to both fields.

List of my publications

Articles accepted by chronological order

[BFG17] **Diagnosability degree of stochastic discrete event systems**,
Hugo Bazille, Eric Fabre and Blaise Genest,
CDC, IEEE 56th Annual Conference on Decision and Control, 2017.

[BFG18b] **Symbolically quantifying response time in stochastic models using moments and semirings**,
Hugo Bazille, Eric Fabre and Blaise Genest,
FoSSaCS, International Conference on Foundations of Software Science and Computation Structures, 2018.

[BFG18a] **Complexity reduction techniques for quantified diagnosability of stochastic systems**,
Hugo Bazille, Eric Fabre and Blaise Genest,
WODES, 14th Workshop on Discrete Event Systems, 2018.

[BFG19] **Certification formelle des réseaux neuronaux profonds : un état de l'art en 2019**,
Hugo Bazille, Eric Fabre and Blaise Genest,
AI & Défense, 2019.

[Aks+] **Classification among Hidden Markov Models**,
Akshay, Hugo Bazille, Eric Fabre and Blaise Genest,
FSTTCS, 39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, 2019.

Articles submitted

[Baz+] **Global PAC Bounds for Learning Discrete Time Markov Chains**,
Hugo Bazille, Blaise Genest, Cyrille Jegourel and Jun Sun.
TACAS, 26th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, 2020.

[Baz+b] **Opacity Degree in Interval Labelled Markov Chains**,
Hugo Bazille, Eric Fabre, Kritin Garg, and Blaise Genest.
LATIN, 14th Latin American Theoretical Informatics Symposium, 2020.

References

[ACY95] Rajeev Alur, Costas Courcoubetis, and Mihalis Yannakakis, "Distinguishing tests for nondeterministic and probabilistic machines", *in*: *STOC*, vol. 95, Citeseer, 1995, pp. 363–372.

[AD00] Robert B. Ash and Catherine A. Doleans-Dade, *Probability and measure theory*, Academic Press, 2000.

[Ade+17] Bruno Adeline et al., "An efficient evaluation scheme for KPIs in regulated urban train systems", *in*: *International Conference on Reliability, Safety and Security of Railway Systems*, Springer, 2017, pp. 195–211.

[AH08] Eleftheria Athanasopoulou and Christoforos N Hadjicostis, "Probability of error bounds for failure diagnosis and classification in hidden Markov models", *in*: *2008 47th IEEE Conference on Decision and Control*, IEEE, 2008, pp. 1477–1482.

[AHK02] Rajeev Alur, Thomas A Henzinger, and Orna Kupferman, "Alternating-time temporal logic", *in*: *Journal of the ACM (JACM)* 49.*5* (2002), pp. 672–713.

[Aks+19] Akshay et al., "Classification among Hidden Markov Models", *in*: *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2019.

[Alu+97] Rajeev Alur et al., "Model-checking of real-time systems: a telecommunications application", *in*: *Proceedings of the International Conference on Software Engineering*, 1997.

[Alu03] Rajeev Alur, "Formal analysis of hierarchical state machines", *in*: *Verification: Theory and Practice*, Springer, 2003, pp. 42–66.

[AM04] Rajeev Alur and Parthasarathy Madhusudan, "Visibly pushdown languages", *in*: *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, ACM, 2004, pp. 202–211.

[Ast65] Karl J Astrom, "Optimal control of Markov processes with incomplete state information", *in*: *Journal of mathematical analysis and applications* 10.*1* (1965), pp. 174–205.

[ATT09] Pierre Ailliot, Craig Thompson, and Peter Thomson, "Space-time modelling of precipitation by using a hidden Markov model and censored Gaussian distributions", *in*: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 58.*3* (2009), pp. 405–426.

[Bal93] Vijay Balasubramanian, *Equivalence and reduction of hidden markov models*, tech. rep., Massachussets Institute of Technology, Cambridge Artificial Intelligence Lab, 1993.

[Bar+01] Boaz Barak et al., "On the (im) possibility of obfuscating programs", *in*: *Annual International Cryptology Conference*, Springer, 2001, pp. 1–18.

[Bas14] Francesco Basile, "Overview of fault diagnosis methods based on Petri net models", *in*: *2014 European Control Conference (ECC)*, IEEE, 2014, pp. 2636–2642.

[Baz+] Hugo Bazille et al., "Global PAC Bounds for Learning Discrete Time Markov Chains", *in*: *Upcoming submission in TACAS*.

[BBW16] Michael Backenköhler, Luca Bortolussi, and Verena Wolf, "Generalized method of moments for stochastic reaction networks in equilibrium", *in*: *International Conference on Computational Methods in Systems Biology*, Springer, 2016, pp. 15–29.

[BD95] Andrea Bianco and Luca De Alfaro, "Model checking of probabilistic and nondeterministic systems", *in*: *International Conference on Foundations of Software Technology and Theoretical Computer Science*, Springer, 1995, pp. 499–513.

[Bei03] Boris Beizer, *Software testing techniques*, Dreamtech Press, 2003.

[Bel57] Richard Bellman, "A Markovian decision process", *in*: *Journal of Mathematics and Mechanics* 6.*5* (1957), pp. 679–684.

[Ben+03] Albert Benveniste et al., "Diagnosis of asynchronous discrete-event systems: a net unfolding approach", *in*: *IEEE Transactions on Automatic Control* 48.*5* (2003), pp. 714–727.

[Ber+14] Nathalie Bertrand et al., "Active diagnosis for probabilistic systems", *in*: *International Conference on Foundations of Software Science and Computation Structures*, Springer, 2014, pp. 29–42.

[Bér+17] Béatrice Bérard et al., "The complexity of diagnosability and opacity verification for Petri nets", *in*: *International Conference on Application and Theory of Petri Nets and Concurrency*, Springer, 2017, pp. 200–220.

[Ber07] Antonia Bertolino, "Software testing research: Achievements, challenges, dreams", *in*: *2007 Future of Software Engineering*, IEEE Computer Society, 2007, pp. 85–103.

[Ber75] Alberto Bertoni, "The solution of problems relative to probabilistic automata in the frame of the formal languages theory", *in*: (1975), pp. 107–112.

[BFG17] Hugo Bazille, Eric Fabre, and Blaise Genest, "Diagnosability degree of stochastic discrete event systems", *in*: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, IEEE, 2017, pp. 5726–5731.

[BFG18a] Hugo Bazille, Eric Fabre, and Blaise Genest, "Complexity reduction techniques for quantified diagnosability of stochastic systems", *in*: *14th Workshop on Discrete Event Systems (WODES)* 51.7 (2018), pp. 82–87.

[BFG18b] Hugo Bazille, Eric Fabre, and Blaise Genest, "Symbolically quantifying response time in stochastic models using moments and semirings", *in*: *International Conference on Foundations of Software Science and Computation Structures*, Springer, 2018, pp. 403–419.

[BGG17] Nathalie Bertrand, Blaise Genest, and Hugo Gimbert, "Qualitative determinacy and decidability of stochastic games with signals", *in*: *Journal of the ACM (JACM)* 64.5 (2017), p. 33.

[BHL14] Nathalie Bertrand, Serge Haddad, and Engel Lefaucheux, "Foundation of diagnosis and predictability in probabilistic systems", *in*: *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'14)*, 2014.

[BHL16a] Nathalie Bertrand, Serge Haddad, and Engel Lefaucheux, "Accurate approximate diagnosability of stochastic systems", *in*: *Language and Automata Theory and Applications*, Springer, 2016, pp. 549–561.

[BHL16b] Nathalie Bertrand, Serge Haddad, and Engel Lefaucheux, "Diagnosis in infinite-state probabilistic systems", *in*: *27th International Conference on Concurrency Theory (CONCUR 2016)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

[BK08] Christel Baier and Joost-Pieter Katoen, *Principles of model checking*, MIT Press, 2008.

[BL17] Dirk Beyer and Thomas Lemberger, "Software verification: Testing vs. model checking", *in*: *Haifa Verification Conference*, Springer, 2017, pp. 99–114.

[BMS15] Béatrice Bérard, John Mullins, and Mathieu Sassolas, "Quantifying opacity", *in*: *Mathematical Structures in Computer Science* 25.*2* (2015), pp. 361–403.

[Bog+15] Sergiy Bogomolov et al., "Adaptive moment closure for parameter inference of biochemical reaction networks", *in*: *International Conference on Computational Methods in Systems Biology*, Springer, 2015, pp. 77–89.

[BP66] Leonard E Baum and Ted Petrie, "Statistical inference for probabilistic functions of finite state Markov chains", *in*: *The annals of mathematical statistics* 37.*6* (1966), pp. 1554–1563.

[BPM83] Mordechai Ben-Ari, Amir Pnueli, and Zohar Manna, "The temporal logic of branching time", *in*: *Acta informatica* 20.*3* (1983), pp. 207–226.

[Bra+06] J Bradley et al., "Response time densities and quantiles in large Markov and semi-Markov Models", *in*: (2006).

[Bra+12] Manuele Brambilla et al., "Property-driven design for swarm robotics", *in*: *International Conference on Autonomous Agents and Multiagent Systems, AAMAS, Valencia, Spain*, 2012, pp. 139–146.

[Brá+14] Tomás Brázdil et al., "Verification of Markov Decision Processes Using Learning Algorithms", *in*: *Automated Technology for Verification and Analysis - 12th International Symposium, ATVA, Sydney, NSW, Australia*, 2014, pp. 98–114.

[Bri+13] Luboš Brim et al., "Exploring parameter space of stochastic biochemical systems using quantitative model checking", *in*: *International Conference on Computer Aided Verification*, Springer, 2013, pp. 107–123.

[Bry+05] Jeremy W Bryans et al., "Opacity generalised to transition systems", *in*: *International Workshop on Formal Aspects in Security and Trust*, Springer, 2005, pp. 81–95.

[BS13] Luca Bortolussi and Guido Sanguinetti, "Learning and Designing Stochastic Processes from Logical Constraints", *in*: *Quantitative Evaluation of Systems - 10th International Conference, QEST, Buenos Aires, Argentina*, 2013, pp. 89–105.

[Cab+09] Maria Paola Cabasino et al., "Diagnosability analysis of unbounded Petri nets", *in*: *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, pp. 1267–1272.

[Cab+12] Maria Paola Cabasino et al., "A new approach for diagnosability analysis of Petri nets using verifier nets", *in*: *IEEE Transactions on Automatic Control* 57.*12* (2012), pp. 3104–3117.

[CE80] Edmund M Clarke and E Allen Emerson, "Characterizing correctness properties of parallel programs using fixpoints", *in*: *International Colloquium on Automata, Languages, and Programming*, Springer, 1980, pp. 169–181.

[CE81] Edmund M Clarke and E Allen Emerson, "Design and synthesis of synchronization skeletons using branching time temporal logic", *in*: *Workshop on Logic of Programs*, Springer, 1981, pp. 52–71.

[CG08] Jorge Castro and Ricard Gavaldà, "Towards Feasible PAC-Learning of Probabilistic Deterministic Finite Automata", *in*: *Grammatical Inference: Algorithms and Applications, 9th International Colloquium, ICGI, Saint-Malo, France*, 2008, pp. 163–174.

[CG91] Kenneth W Church and William A Gale, "A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams", *in*: *Computer Speech & Language* 5.*1* (1991), pp. 19–54.

[CG99] Stanley F. Chen and Joshua Goodman, "An Empirical Study of Smoothing Techniques for Language Modeling", *in*: *Computer Speech and Language* 13.*4* (1999), pp. 359–394.

[CGS09] Maria Paola Cabasino, Alessandro Giua, and Carla Seatzu, "Diagnosability of bounded Petri nets", *in*: *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, pp. 1254–1260.

[Che+09] Taolue Chen et al., "Quantitative model checking of continuous-time Markov chains against timed automata specifications", *in*: *2009 24th Annual IEEE Symposium on Logic In Computer Science*, IEEE, 2009, pp. 309–318.

[Che+12] Yingke Chen et al., "Learning Markov models for stationary system behaviors", *in*: *NASA formal methods symposium*, Springer, 2012, pp. 216–230.

[Che+52] Herman Chernoff et al., "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations", *in*: *The Annals of Mathematical Statistics* 23.*4* (1952), pp. 493–507.

[Che13] Jianhua Chen, "Properties of a New Adaptive Sampling Method with Applications to Scalable Learning", *in*: *WI, Atlanta*, 2013, pp. 9–15.

[CHH11] Krishnendu Chatterjee, Thomas A Henzinger, and Florian Horn, "The complexity of request-response games", *in*: *International Conference on Language and Automata Theory and Applications*, Springer, 2011, pp. 227–237.

[CK13] Jun Chen and Ratnesh Kumar, "Polynomial test for stochastic diagnosability of discrete-event systems", *in*: *IEEE Transactions on Automation Science and Engineering* 10.*4* (2013), pp. 969–979.

[CK14] Taolue Chen and Stefan Kiefer, "On the Total Variation Distance of Labelled Markov Chains", *in*: *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, 2014, 33:1–33:10.

[CK97] Karel Culik and Jarkko Kari, "Digital images and formal languages", *in*: (1997), pp. 599–616.

[CL09] Christos G Cassandras and Stephane Lafortune, *Introduction to discrete event systems*, Springer Science & Business Media, 2009.

[Cla+00] Edmund Clarke et al., "Counterexample-guided abstraction refinement", *in*: *International Conference on Computer Aided Verification*, Springer, 2000, pp. 154–169.

[CMR06] Corinna Cortes, Mehryar Mohri, and Ashish Rastogi, "On the computation of some standard distances between probabilistic automata", *in*: *International Conference on Implementation and Application of Automata*, Springer, 2006, pp. 137–149.

[CMR07] Corinna Cortes, Mehryar Mohri, and Ashish Rastogi, "Lp distance and equivalence of probabilistic automata", *in*: *International Journal of Foundations of Computer Science* 18.*04* (2007), pp. 761–779.

[CNB98] Matthew Crouse, Robert David Nowak, and Richard G Baraniuk, "Wavelet-based statistical signal processing using hidden Markov models", *in*: *IEEE Transactions on signal processing* 46.*4* (1998), pp. 886–902.

[Coc78] William G. Cochran, “Contributions to Survey Sampling and Applied Statistics”, *in*: ed. by H.A. David, Academic Press, New York, 1978, chap. Laplace’s ratio estimator, pp. 3–10.

[Cor+08] Corinna Cortes et al., “On the computation of the relative entropy of probabilistic automata”, *in*: *International Journal of Foundations of Computer Science* 19.*01* (2008), pp. 219–242.

[CP09] Elodie Chanthery and Yannick Pencolé, “Monitoring and active diagnosis for discrete-event systems”, *in*: *IFAC Proceedings Volumes* 42.*8* (2009), pp. 1545–1550.

[CPS18] Yuqi Chen, Christopher M. Poskitt, and Jun Sun, “Learning from Mutants: Using Code Mutation to Learn and Monitor Invariants of a Cyber-Physical System”, *in*: *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, 2018, pp. 648–660.

[Cra38] Harald Cramér, “Sur un nouveau théoreme-limite de la théorie des probabilités”, *in*: *Actual. Sci. Ind.* 736 (1938), pp. 5–23.

[CT04] Alexander Clark and Franck Thollard, “PAC-learnability of Probabilistic Deterministic Finite State Automata”, *in*: *Journal of Machine Learning Research* 5 (2004), pp. 473–497.

[CT12] Thomas M Cover and Joy A Thomas, *Elements of information theory*, John Wiley & Sons, 2012.

[DA05] Tugrul Dayar and Nail Akar, “Computing moments of first passage times to a subset of states in Markov chains”, *in*: *SIAM Journal on Matrix Analysis and Applications* 27.*2* (2005), pp. 396–412.

[Dac+16] Przemyslaw Daca et al., “Linear Distances between Markov Chains”, *in*: *27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada*, 2016, 20:1–20:15.

[Dac+17] Przemysław Daca et al., “Faster statistical model checking for unbounded temporal properties”, *in*: *ACM Transactions on Computational Logic (TOCL)* 18.*2* (2017), p. 12.

[DDM08] Jérémy Dubreil, Philippe Darondeau, and Hervé Marchand, “Opacity enforcing control synthesis”, *in*: *2008 9th International Workshop on Discrete Event Systems*, IEEE, 2008, pp. 28–35.

[Des+04] Josée Desharnais et al., "Metrics for labeled Markov processes", *in*: *Theoretical computer science* 318.*3* (2004), pp. 323–354.

[DHR08] Laurent Doyen, Thomas A Henzinger, and Jean-François Raskin, "Equivalence of labeled Markov chains", *in*: *International journal of foundations of computer science* 19.*03* (2008), pp. 549–563.

[Dur+98] Richard Durbin et al., *Biological sequence analysis: probabilistic models of proteins and nucleic acids*, Cambridge university press, 1998.

[Dut+18] Souradeep Dutta et al., "Output range analysis for deep feedforward neural networks", *in*: *NASA Formal Methods Symposium*, Springer, 2018, pp. 121–138.

[Edd04] Sean R Eddy, "What is a hidden Markov model?", *in*: *Nature biotechnology* 22.*10* (2004), p. 1315.

[Fab+05] Eric Fabre et al., "Distributed monitoring of concurrent and asynchronous systems", *in*: *Discrete Event Dynamic Systems* 15.*1* (2005), pp. 33–84.

[Fab13] Eric Fabre, *Control of Discrete-Event Systems - Automata and Petri Net Perspectives*, Springer, 2013, pp. 85–106.

[FBJ02] Eric Fabre, Albert Benveniste, and Claude Jard, "Distributed diagnosis for large discrete event dynamic systems", *in*: *IFAC Proceedings Volumes* 35.*1* (2002), pp. 1–6.

[FJ10] Eric Fabre and Loïg Jezequel, "On the construction of probabilistic diagnosers", *in*: *IFAC Proceedings Volumes* 43.*12* (2010), pp. 229–234.

[Flo62] Robert W Floyd, "Algorithm 97: shortest path", *in*: *Communications of the ACM* 5.*6* (1962), p. 345.

[Geh+18] Timon Gehr et al., "Ai2: Safety and robustness certification of neural networks with abstract interpretation", *in*: *2018 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2018, pp. 3–18.

[Gho+17] Shalini Ghosh et al., "Trusted Machine Learning: Model Repair and Data Repair for Probabilistic Models", *in*: *AAAI-17 Workshop on Symbolic Inference and Optimization*, 2017.

[GO10] Hugo Gimbert and Youssouf Oualhadj, "Probabilistic automata on finite words: Decidable and undecidable problems", *in*: *International Colloquium on Automata, Languages, and Programming*, Springer, 2010, pp. 527–538.

[Gon+13] Andres M Gonzalez et al., “Identification of biological models from single-cell data: a comparison between mixed-effects and moment-based inference”, *in*: *2013 European Control Conference (ECC)*, IEEE, 2013, pp. 3652–3657.

[Goo+14] Ian Goodfellow et al., “Generative adversarial nets”, *in*: *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[Goo53] Irving John Good, “The population frequencies of species and the estimation of population parameters”, *in*: *Biometrika* 40.*3-4* (Dec. 1953), pp. 237–264, ISSN: 0006-3444.

[Gop+18] Divya Gopinath et al., “Deepsafe: A data-driven approach for assessing robustness of neural networks”, *in*: *International Symposium on Automated Technology for Verification and Analysis*, Springer, 2018, pp. 3–19.

[Gou+01] Julian Gough et al., “Assignment of homology to genome sequences using a library of hidden Markov models that represent all proteins of known structure”, *in*: *Journal of molecular biology* 313.*4* (2001), pp. 903–919.

[GS02] Alison L Gibbs and Francis Edward Su, “On choosing and bounding probability metrics”, *in*: *International statistical review* 70.*3* (2002), pp. 419–435.

[GS18] Marcus Gerhold and Mariëlle Stoelinga, “Model-based testing of probabilistic systems”, *in*: *Formal aspects of computing* 30.*1* (2018), pp. 77–106.

[GS95] William A. Gale and Geoffrey Sampson, “Good-Turing Frequency Estimation Without Tears”, *in*: *Journal of Quantitative Linguistics* (1995), pp. 217–37.

[Haa+13] Stefan Haar et al., “Optimal constructions for active diagnosis”, *in*: *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2013)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

[Hav+00] Klaus Havelund et al., “Formal analysis of the remote agent before and after flight”, *in*: *Proceedings of the 5th NASA Langley Formal Methods Workshop*, vol. 134, 2000.

[Hér+04] Thomas Hérault et al., “Approximate Probabilistic Model Checking”, *in*: *VMCAI*, vol. 2937, LNCS, 2004, pp. 307–329.

[HF13] Stefan Haar and Éric Fabre, “Diagnosis with petri net unfoldings”, *in*: *Control of Discrete-Event Systems*, Springer, 2013, pp. 301–317.

[HJ94] Hans Hansson and Bengt Jonsson, “A logic for reasoning about time and reliability”, *in*: *Formal aspects of computing* 6.*5* (1994), pp. 512–535.

[HK97] Michael Huth and Marta Kwiatkowska, "Quantitative analysis and model checking", *in*: *Proceedings of Twelfth Annual IEEE Symposium on Logic in Computer Science*, IEEE, 1997, pp. 111–122.

[Hol+17] Chuck Holland et al., "UPS optimizes delivery routes", *in*: *Interfaces* 47.*1* (2017), pp. 8–23.

[Hol92] Gerard J Holzmann, "Practical methods for the formal validation of SDL specifications", *in*: *Computer Communications* 15.*2* (1992), pp. 129–134.

[Hor+15] Florian Horn et al., "Optimal strategy synthesis for request-response games", *in*: *RAIRO-Theoretical Informatics and Applications* 49.*3* (2015), pp. 179–203.

[HSP99] Geoffrey E Hinton, Terrence Joseph Sejnowski, and Tomaso A Poggio, *Unsupervised learning: foundations of neural computation*, MIT press, 1999.

[Hua+17] Xiaowei Huang et al., "Safety verification of deep neural networks", *in*: *International Conference on Computer Aided Verification*, Springer, 2017, pp. 3–29.

[HYH13] Feng-Long Huang, Ming-Shing Yu, and Chien-Yo Hwang, "An empirical study of good-turing smoothing for language models on different size corpora of chinese", *in*: *Journal of Computer and Communications* 1.*05* (2013), p. 14.

[Jia+01] Shengbing Jiang et al., "A polynomial algorithm for testing diagnosability of discrete-event systems", *in*: *IEEE Transactions on Automatic Control* 46.*8* (2001), pp. 1318–1321.

[JSS17] Cyrille Jégourel, Jun Sun, and Jin Song Dong, "Sequential Schemes for Frequentist Estimation of Properties in Statistical Model Checking", *in*: *Quantitative Evaluation of Systems - 14th International Conference, QEST, Berlin, Germany*, 2017, pp. 333–350.

[Kat87] Slava Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer", *in*: *IEEE transactions on acoustics, speech, and signal processing* 35.*3* (1987), pp. 400–401.

[Kel76] Robert M Keller, "Formal verification of parallel programs", *in*: *Communications of the ACM* 19.*7* (1976), pp. 371–384.

[KH18] Christoforos Keroglou and Christoforos N Hadjicostis, "Probabilistic system opacity in discrete event systems", *in*: *Discrete Event Dynamic Systems* 28.*2* (2018), pp. 289–314.

[Kie+11] Stefan Kiefer et al., “Language equivalence for probabilistic automata”, *in*: *International Conference on Computer Aided Verification*, Springer, 2011, pp. 526–540.

[Kie18] Stefan Kiefer, “On Computing the Total Variation Distance of Hidden Markov Models”, *in*: *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[KL51] Solomon Kullback and Richard A. Leibler, “On Information and Sufficiency”, *in*: *Annals of Mathematical Statistics* 22.*1* (1951), pp. 79–86.

[KLC98] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra, “Planning and acting in partially observable stochastic domains”, *in*: *Artificial intelligence* 101.*1-2* (1998), pp. 99–134.

[KLM96] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore, “Reinforcement learning: A survey”, *in*: *Journal of artificial intelligence research* 4 (1996), pp. 237–285.

[KNP11] Marta Kwiatkowska, Gethin Norman, and David PRISM Parker, “4.0: Verification of probabilistic real-time systems”, *in*: *International Conference on Computer Aided Verification*, 2011, p. 585591.

[Kon01] Igor Kononenko, “Machine learning for medical diagnosis: history, state of the art and perspective”, *in*: *Artificial Intelligence in medicine* 23.*1* (2001), pp. 89–109.

[Koz77] Dexter Kozen, “Lower bounds for natural proof systems”, *in*: *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)* (1977), pp. 254–266.

[Kro+01] Anders Krogh et al., “Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes”, *in*: *Journal of molecular biology* 305.*3* (2001), pp. 567–580.

[KS16] Stefan Kiefer and A Prasad Sistla, “Distinguishing hidden Markov chains”, *in*: *2016 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, IEEE, 2016, pp. 1–10.

[KS60] John G Kemeny and J Laurie Snell, “Finite Markov Chains. D Van Nostad Co”, *in*: *Inc., Princeton, NJ* (1960).

[KS85] Werner Kuich and Arto Salomaa, “Semirings, automata, languages”, *in*: (1985).

[KU02] Igor O Kozine and Lev V Utkin, “Interval-valued finite Markov chains”, *in*: *Reliable computing* 8.*2* (2002), pp. 97–113.

[Lan11] Pat Langley, “The changing science of machine learning”, *in*: *Machine Learning* 82.*3* (2011), pp. 275–279.

[Law+03] Yee Wei Law et al., “A formally verified decentralized key management architecture for wireless sensor networks”, *in*: *IFIP International Conference on Personal Wireless Communications*, Springer, 2003, pp. 27–39.

[Lee+09] Honglak Lee et al., “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations”, *in*: *Proceedings of the 26th annual international conference on machine learning*, ACM, 2009, pp. 609–616.

[Lef18] Engel Lefaucheux, “Controlling information in probalistic systems”, PhD thesis, Université Rennes 1, 2018.

[Leh77] Daniel J Lehmann, “Algebraic structures for transitive closure”, *in*: *Theoretical Computer Science* 4.*1* (1977), pp. 59–76.

[Lin11] Feng Lin, “Opacity of discrete event systems and its applications”, *in*: *Automatica* 47.*3* (2011), pp. 496–503.

[LP02] Rune B Lyngsø and Christian NS Pedersen, “The consensus string problem and the complexity of comparing hidden Markov models”, *in*: *Journal of Computer and System Sciences* 65.*3* (2002), pp. 545–569.

[LST16] Axel Legay, Sean Sedwards, and Louis-Marie Traonouez, “Rare Events for Statistical Model Checking an Overview”, *in*: *Reachability Problems - 10th International Workshop, RP, Aalborg, Denmark*, 2016, pp. 23–35.

[Mao+11] Hua Mao et al., “Learning probabilistic automata for model checking”, *in*: *2011 Eighth International Conference on Quantitative Evaluation of Systems*, IEEE, 2011, pp. 111–120.

[Mao+12] Hua Mao et al., “Learning Markov decision processes for model checking”, *in*: 103 (2012), pp. 49–63.

[MHC03] Omid Madani, Steve Hanks, and Anne Condon, “On the undecidability of probabilistic planning and related stochastic optimization problems”, *in*: *Artificial Intelligence* 147.*1-2* (2003), pp. 5–34.

[Moh02a] Mehryar Mohri, “Generic e-Removal and Input e-Normalization Algorithms for Weighted Transducers”, *in*: *Int. J. Found. Comput. Sci.* 13.*1* (2002), pp. 129–143.

[Moh02b] Mehryar Mohri, “Semiring frameworks and algorithms for shortest-distance problems”, *in*: *Journal of Automata, Languages and Combinatorics* 7.*3* (2002), pp. 321–350.

[Moh97] Mehryar Mohri, “Finite-state transducers in language and speech processing”, *in*: *Computational linguistics* 23.*2* (1997), pp. 269–311.

[Mos82] Yiannis Nicholas Moschovakis, *Descriptive Set Theory*, 1982.

[MP09] Christophe Morvan and Sophie Pinchinat, “Diagnosability of pushdown systems”, *in*: *Haifa Verification Conference*, Springer, 2009, pp. 21–33.

[MU17] Michael Mitzenmacher and Eli Upfal, *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*, Cambridge university press, 2017.

[Mug96] Stephen Muggleton, “Stochastic logic programs”, *in*: *Advances in inductive logic programming* 32 (1996), pp. 254–264.

[ND08] Farid Nouioua and Philippe Dague, “A probabilistic analysis of diagnosability in discrete event systems.”, *in*: *ECAI*, 2008, pp. 224–228.

[Oka58] Masashi Okamoto, “Some Inequalities Relating to the Partial Sum of Binomial Probabilities”, *in*: *Annals of the Institute of Statistical Mathematics* 10 (1958), pp. 29–35.

[Paz71] Azaria Paz, “Introduction to probabilistic automata (Computer science and applied mathematics)”, *in*: (1971).

[Pel13] Jan Peleska, “Industrial-strength model-based testing-state of the art and current challenges”, *in*: *arXiv preprint arXiv:1303.1006* (2013).

[Pet66] Carl Adam Petri, “Communication with automata”, *in*: (1966).

[Pnu77] Amir Pnueli, “The temporal logic of programs”, *in*: *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, IEEE, 1977, pp. 46–57.

[PR69] Yu. V. Prohorov and Yu. A. Rozanov, “Probability Theory. Basic Concepts. Limit Theorems. Random Processes.”, *in*: *Metrika* 17 (1969), pp. 261–262.

[PT11] Luca Pulina and Armando Tacchella, "NeVer: a tool for artificial neural networks verification", *in*: *Annals of Mathematics and Artificial Intelligence* 62.*3-4* (2011), pp. 403–425.

[QK06] Wenbin Qiu and Ratnesh Kumar, "Decentralized failure diagnosis of discrete event systems", *in*: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 36.*2* (2006), pp. 384–395.

[QS82] Jean-Pierre Queille and Joseph Sifakis, "Specification and verification of concurrent systems in CESAR", *in*: *International Symposium on programming*, Springer, 1982, pp. 337–351.

[Rab63] Michael O Rabin, "Probabilistic automata", *in*: *Information and control* 6.*3* (1963), pp. 230–245.

[Rab89] Lawrence R Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition", *in*: *Proceedings of the IEEE* 77.*2* (1989), pp. 257–286.

[Ram07] Daniel Ramage, "Hidden Markov models fundamentals", *in*: *CS229 Section Notes* 1 (2007).

[Rid05] Ad Ridder, "Importance Sampling Simulations of Markovian Reliability Systems Using Cross-Entropy", *in*: *Annals OR* 134.*1* (2005), pp. 119–136.

[RN16] Stuart J Russell and Peter Norvig, *Artificial intelligence: a modern approach*, Malaysia; Pearson Education Limited, 2016.

[Sad+14] Dorsa Sadigh et al., "Data-driven probabilistic modeling and verification of human driver behavior", *in*: *2014 AAAI Spring Symposium Series*, 2014.

[Sam+96] Meera Sampath et al., "Failure diagnosis using discrete-event models", *in*: *IEEE transactions on control systems technology* 4.*2* (1996), pp. 105–124.

[SC85] A Prasad Sistla and Edmund M Clarke, "The complexity of propositional linear temporal logics", *in*: *Journal of the ACM (JACM)* 32.*3* (1985), pp. 733–749.

[Sch61] Marcel Paul Schützenberger, "On the definition of a family of automata", *in*: *Information and control* 4.*2-3* (1961), pp. 245–270.

[SGB95] Chris Sherlaw-Johnson, Steve Gallivan, and Jim Burridge, "Estimating a Markov Transition Matrix from Observational Data", *in*: *The Journal of the Operational Research Society* 46.*3* (1995), pp. 405–410.

[Sin91] Kasim Sinnamohideen, "Discrete-event based diagnostic supervisory control system", *in*: *Proceedings of the AIChE Annual Meeting*, 1991.

[SLT98] Meera Sampath, Stéphane Lafortune, and Demosthenis Teneketzis, "Active diagnosis of discrete-event systems", *in*: *IEEE Transactions on Automatic Control* 43.*7* (1998), pp. 908–929.

[Smy94] Padhraic Smyth, "Hidden Markov models for fault detection in dynamic systems", *in*: *Pattern recognition* 27.*1* (1994), pp. 149–164.

[SS83] Norbert Schmitz and Benno Süselbeck, "Sequential probability ratio tests for homogeneous Markov chains", *in*: *Mathematical Learning Models—Theory and Algorithms*, Springer, 1983, pp. 191–202.

[ST02] Raja Sengupta and Stavros Tripakis, "Decentralized diagnosability of regular languages is undecidable", *in*: *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.* Vol. 1, IEEE, 2002, pp. 423–428.

[Sto06] Jordan Stoyanov, "Determinacy of distributions by their moments", *in*: *Proceedings for International Conference on Mathematics and Statistical Modeling*, 2006.

[Su+02] Rong Su et al., "Distributed diagnosis for qualitative systems", *in*: *Sixth International Workshop on Discrete Event Systems, 2002. Proceedings.* IEEE, 2002, pp. 169–174.

[SVA04] Koushik Sen, Mahesh Viswanathan, and Gul Agha, "Statistical model checking of black-box probabilistic systems", *in*: *International Conference on Computer Aided Verification*, Springer, 2004, pp. 202–215.

[SVA06] Koushik Sen, Mahesh Viswanathan, and Gul Agha, "Model-checking Markov chains in the presence of uncertainties", *in*: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, 2006, pp. 394–410.

[SW04] Rong Su and W. Murray Wonham, "A model of component consistency in distributed diagnosis", *in*: *IFAC Proceedings Volumes* 37.*18* (2004), pp. 417–422.

[SW95] Adam Shwartz and Alan Weiss, *Large deviations for performance analysis: queues, communication and computing*, vol. 5, CRC Press, 1995.

[Tar05] Árpád Tari, "Moments based bounds in stochastic models", *in*: (2005).

[TC13] Vicenç Torra and Michael Carlson, "On the Hellinger distance for measuring information loss in microdata", *in*: *Joint UNECE/Eurostat work session on statistical data confidentiality* (2013).

[TH07] Miklós Telek and Gábor Horváth, "A minimal representation of Markov arrival processes and a moments matching method", *in*: *Performance Evaluation* 64.*9-12* (2007), pp. 1153–1168.

[Tob+07] Llanos Tobarra et al., "Model checking wireless sensor network security protocols: Tinysec+ leap", *in*: *IFIP Conference on Wireless Sensor and Actor Networks*, Springer, 2007, pp. 95–106.

[TT05] David Thorsley and Demosthenis Teneketzis, "Diagnosability of stochastic discrete-event systems", *in*: *IEEE Transactions on Automatic Control* 50.*4* (2005), pp. 476–492.

[TYG08] David Thorsley, Tae-Sic Yoo, and Humberto E Garcia, "Diagnosability of stochastic discrete-event systems under unreliable observations", *in*: *2008 American Control Conference*, IEEE, 2008, pp. 1158–1165.

[Tze92] Wen-Guey Tzeng, "A polynomial-time algorithm for the equivalence of probabilistic automata", *in*: *SIAM Journal on Computing* 21.*2* (1992), pp. 216–227.

[UPL12] Mark Utting, Alexander Pretschner, and Bruno Legeard, "A taxonomy of model-based testing approaches", *in*: *Software Testing, Verification and Reliability* 22.*5* (2012), pp. 297–312.

[Val84] Leslie G. Valiant, "A Theory of the Learnable", *in*: *Commun. ACM* 27.*11* (1984), pp. 1134–1142.

[Wai+89] Alex Waibel et al., "Phoneme recognition using time-delay neural networks", *in*: *IEEE transactions on acoustics, speech, and signal processing* 37.*3* (1989), pp. 328–339.

[Wal45] Abraham Wald, "Sequential tests of statistical hypotheses", *in*: *The Annals of Mathematical Statistics* 16.*2* (1945), pp. 117–186.

[Wan+17] Jingyi Wang et al., "Should We Learn Probabilistic Models for Model Checking? A New Approach and An Empirical Study", *in*: *Fundamental Approaches to Software Engineering - 20th International Conference, FASE, Uppsala, Sweden*, 2017, pp. 3–21.

[Wet66] G Barrie Wetherill, "Sequential methods in statistics", *in*: (1966).

[WHK18] Matthew Wicker, Xiaowei Huang, and Marta Kwiatkowska, "Feature-guided black-box safety testing of deep neural networks", *in*: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, 2018, pp. 408–426.

[WYL07] Yin Wang, Tae-Sic Yoo, and Stéphane Lafortune, "Diagnosis of discrete event systems using decentralized architectures", *in*: *Discrete Event Dynamic Systems* 17.*2* (2007), pp. 233–263.

[YL02] Tae-Sic Yoo and Stéphane Lafortune, "Polynomial-time verification of diagnosability of partially observed discrete-event systems", *in*: *IEEE Transactions on automatic control* 47.*9* (2002), pp. 1491–1495.

[YS02] Håkan L. S. Younes and Reid G. Simmons, "Probabilistic Verification of Discrete Event Systems Using Acceptance Sampling", *in*: *Computer Aided Verification, 14th International Conference, CAV ,Copenhagen, Denmark*, 2002, pp. 223–235.

[ZL13] Janan Zaytoon and Stéphane Lafortune, "Overview of fault diagnosis methods for discrete event systems", *in*: *Annual Reviews in Control* 37.*2* (2013), pp. 308–320.

[ZPC13] Paolo Zuliani, André Platzer, and Edmund M Clarke, "Bayesian statistical model checking with application to Stateflow/Simulink verification", *in*: *Formal Methods in System Design* 43.*2* (2013), pp. 338–367.

List of Figures

Index

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

UNIVERSITÉ DE RENNES 1

Titre: Détection et Quantification d'Evenements dans les Systèmes Stochastiques

Mot clés : Systèmes stochastiques, Information partielle, Diagnosticabilité, Classification, Apprentissage, Vérification de modèles

Resumé : Les systèmes stochastiques à information partielle permettent de représenter de nombreux systèmes dont les paramètres sont inconnus et dont le fonctionnement dépend de facteurs en dehors de notre contrôle. Dans cette thèse, nous étudions plusieurs problèmes liés à ces systèmes. Le premier est la diagnosticabilité, c'est-à-dire la capacité de décider si un évènement particulier s'est produit. Le second est la classification qui est la capacité de décider à partir d'une trace d'une exécution quel système l'a produite. Enfin, nous nous intéressons aux garanties que l'on peut avoir quand on apprend les probabilités de transition d'un système stochastique.

Title: Detection and Quantification of Events in Stochastic Systems

Keywords : Stochastic systems, Partial information, Diagnosability, Classification, Learning, Model-checking

Abstract : Stochastic systems with partial information allow one to represent numerous systems whose parameters are unknown and whose operation may depend on out-of-control factors. In this thesis, we study several problems linked to these systems. First one is diagnosability, that is the capacity to decide if a particular event occurred. Second one is classification which is the capacity to decide from a trace of an execution which system produced it. Finally, we are interested in the guarantees one can obtain by learning the probability transitions of a stochastic system.