Foundations of networks towards AI

Hicham Lesfari

UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

THÈSE DE DOCTORAT

Fondements Réseaux et l'IA

Hicham LESFARI

Université Côte d'Azur, Inria, CNRS, I3S, France

Présentée en vue de l'obtention du grade de docteur en INFORMATIQUE **d'**Université Côte d'Azur

Dirigée par : Frédéric GIROIRE, Directeur de recherche, CNRS

Soutenue le : 7 Octobre 2022

Devant le jury, composé de :

Andrea CLEMENTI, Full Professor, University of Rome Tor Vergata

Romuald ELIE, Research Scientist, DeepMind, Google

Frédéric GIROIRE, Directeur de recherche, CNRS

Marc LELARGE, Directeur de recherche, INRIA Paris

Pietro MICHIARDI, Full Professor, EURECOM

Giovanni NEGLIA, Directeur de recherche, INRIA Sophia Antipolis

FONDEMENTS RÉSEAUX ET L'IA

Foundations of Networks towards AI

Hicham LESFARI

Jury :

Président du jury
Giovanni NEGLIA, Directeur de recherche, INRIA Sophia Antipolis

Rapporteurs
Marc LELARGE, Directeur de recherche, INRIA Paris
Pietro MICHIARDI, Full Professor, EURECOM

Examinateurs
Andrea CLEMENTI, Full Professor, University of Rome Tor Vergata
Romuald ELIE, Research Scientist, DeepMind, Google
Frédéric GIROIRE, Directeur de recherche, CNRS
Giovanni NEGLIA, Directeur de recherche, INRIA Sophia Antipolis

Directeur de thèse
Frédéric GIROIRE, Directeur de recherche, CNRS

Soutenue
7 Octobre 2022

Université Côte d'Azur

To the people who believed in me and taught me to believe.

Résumé

Le domaine de l'Intelligence Artificielle (IA) a un large impact sur la société d'aujourd'hui, ayant conduit notamment à une interaction passionnante entre plusieurs disciplines scientifiques. À cet égard, un double intérêt émerge dans la littérature.

D'une part, une tendance croissante dans les réseaux de télécommunication consiste à revisiter les problèmes d'optimisation classiques en utilisant des techniques d'apprentissage automatique afin d'exploiter leurs avantages potentiels. Nous nous focaliserons sur certains défis posés par la détection d'anomalies dans les réseaux ainsi que l'allocation des ressources dans le cadre des réseaux logiciels (SDN) et de la virtualisation des fonctions réseau (NFV).

D'autre part, un effort substantiel a été consacré dans le but d'apporter une compréhension théorique au comportement collectif des réseaux. Nous nous focaliserons sur certains défis posés par l'étude de la dynamique majoritaire au sein des systèmes multi-agents ainsi qu'à la compression des réseaux de neurones artificiels dans le but d'augmenter leur efficacité.

Dans cette étude, nous contextualisons les points focaux ci-dessus dans le cadre de l'étude de certains fondements de réseaux ; vus sous l'angle des réseaux de télécommunications et des réseaux neuronaux. Nous nous concentrons d'abord sur le développement de mesures de similarité de graphes pour la détection d'anomalies dans les réseaux. Ensuite, nous étudions la dynamique majoritaire déterministe et stochastique dans les systèmes multi-agents. Ensuite, nous discutons du problème de la somme de sous-ensembles aléatoires dans le contexte de la compression des réseaux neuronaux. Enfin, nous passons en revue quelques problèmes généraux divers.

Mots-clés : Apprentissage automatique, Optimisation des réseaux, Algorithmique, Théorie des graphes, Optimisation combinatoire.

Abstract

The field of Artificial Intelligence (AI) has brought a broad impact on today's society, leading to a gripping interaction between several scientific disciplines. In this respect, there has been a strong twofold interest across the literature.

On the one hand, a growing trend in telecommunication networks consists in revisiting classic optimization problems using machine learning techniques in order to exploit their potential benefits. We focus on some challenges brought by the detection of anomalies in networks, and the allocation of resources within software-defined networking (SDN) and network function virtualization (NFV).

On the other hand, a substantial effort has been devoted towards the theoretical understanding of the collective behavior of networks. We focus on some challenges brought by the study of majority dynamics within multi-agent systems, and the compression of artificial neural networks with the aim at increasing their efficiency.

In this study, we contextualize the above focal points in the framework of investigating some foundations of networks ; viewed through the lens of telecommunications networks and neural networks. We first focus our attention on developing graph similarity measures for network anomaly detection. Next, we study deterministic and stochastic majority dynamics in multi-agent systems. Then, we discuss the random subset sum problem in the context of neural network compression. Finally, we walk through some other miscellaneous problems.

Keywords : Machine learning, Network optimization, Algorithmics, Graph theory, Combinatorial optimization.

Contents

Other Works

CHAPTER 1

Introduction

In this chapter, we first introduce the background context in which this thesis takes place and the research problems that inspired our work. We then highlight our contributions and conclude this chapter with an outline of the remainder of this thesis.

1.1 Graph Learning

Learning on graph-structured data has gained a notable momentum with recent advances [Wu et al., 2020, Zhang et al., 2020] in Artificial Intelligence (AI). While graphs provide an ubiquitous data structure to represent complex structures and interactions, readily blending them with standard machine learning algorithms brings several challenges [Cai et al., 2018]. Of particular relevance is the progress in designing flexible and expressive methods [Hamilton et al., 2017b] for quantifying similarities between graphs. These methods fall broadly into two approaches.

Structure-based. We distinguish between four categories.

— *Exact matching* methods [McKay, 1990] aim at assessing how similar two graphs are by determining if they are isomorphic. This is known as the Graph Isomorphism (GI) problem [Babai and Luks, 1983, Babai, 2016], under which two graphs are said to be isomorphic if there exists a rearrangement of the nodes, keeping the edges untouched, such that the two graphs are identical. Formally, two graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are isomorphic if there exists a bijection $f : V_1 \to V_2$ such that for all $u, v \in V_1$:

$$(u, v) \in E_1 \Leftrightarrow (f(u), f(v)) \in E_2.$$

However, while GI is proven to be NP, determining if it is P or NP-complete remains a major open problem [Read and Corneil, 1977, Hartmanis, 1982] in theoretical computer science. Furthermore, if an edge is missing or has been mistakenly deleted, isomorphism search would attribute to almost similar graphs a similarity score of zero.

— *Inexact matching* methods, on the other hand, are more flexible and allow for noise in the nodes, edges, and their labels. For instance, graph edit distance (GED) [Neuhaus and Bunke, 2007b, Sanfeliu and Fu, 1983] aims to define a distance between graphs based on a measure of distortion [Wagner and Fischer, 1974] required to transform one graph into another. During this transformation, a cost is assigned to different types of operations including node or edge insertions, removals, and substitutions. Afterwards, the distance is encoded by the minimal cost associated to an edit path in the optimal alignments, which transforms one graph to the other. The GED is often implemented using an A* search algorithm [Hart et al.,

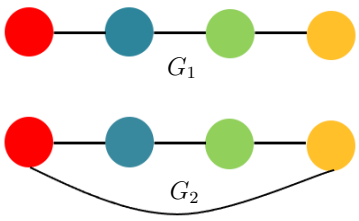


Figure 1.1 – Input graphs.

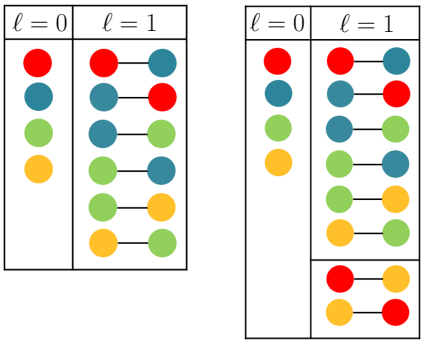


Figure 1.2 – Features induced by the 1-step random walk kernel for G_1 and G_2, respectively.

Figure 1.3 – The 1-step random walk kernel attributes a similarity score of $k_{\text{1-RWK}}(G_1, G_2) = 64$ to the input graphs, based on their feature embeddings $\phi(G_1) = [4, 6]^T$ and $\phi(G_2) = [4, 8]^T$.

1968]. Unfortunately, the computational complexity induced by this technique is prohibitive and computing the GED, even for uniform edit costs [Neuhaus and Bunke, 2007a], is NP-complete [Lewis, 1983] and APX-hard to approximate.

Moreover, the GED does not correspond to a distance in an Euclidean space [Dattorro, 2010]. Such limitation implies that the metric space defined by the edit distance on the set of graphs cannot be isometrically embedded into an Hilbert space. Thereby, the set of machine learning algorithms which may be used in conjunction with the GED [Neuhaus and Bunke, 2007b] are drastically restricted.

— *Descriptor* methods build similarity vectors by extracting several topological indicators [Todeschini and Consonni, 2008] from graphs. However, the transformation often induces a major loss of topological information and the most discriminant set of features rely heavily on the substantial domain knowledge. We note that such descriptors [Randic, 1975, Randic, 1975] have been mostly used in bio or chemoinformatics for comparing between molecular graphs, in order to correlate and predict biological activities from molecular structures [Gutman and Trinajstić, 1972, Wiener, 1947, Chung and Graham, 1997].

— *Frequent subgraph* methods [Maimon and Rokach, 2005, Agrawal et al., 1994, Yan and Han, 2002, Saigo et al., 2009] aim at detecting the most discriminant subgraphs that are frequent among a family of graphs. The core operation behind these techniques is based on computing the Maximum Common Subgraph [Bunke and Shearer, 1998, Neuhaus and Bunke, 2007b] which contains the Subgraph Isomorphism (SI) problem as an intermediate step. SI is analogue of GI checking in a setting in which two graphs have different sizes and, has been proven to be NP-complete [Garey and Johnson, 1979a].

Kernel-based. Graphs kernels emerged as an attractive tool for graph similarity. They retain concepts from previous traditional branches of graph comparison by capturing characteristics in terms of isomorphic structures, allowing for inexact matchings and, vectorizing graphs in a space of graph features. A graph kernel is a kernel function $k \colon \mathcal{G} \times \mathcal{G} \to \mathbb{R}$ [Schölkopf, 2000] defined over a set of graphs $\mathcal{G}$, and enables to compare between them. It is equivalent to an inner product

of the embeddings of a pair of graphs into a possibly high-dimensional Hilbert space [Berlinet and Thomas-Agnan, 2011]. Specifically, given a kernel k, there exists a map $\phi\colon \mathcal{G} \to \mathcal{H}$ into a Hilbert space $\mathcal{H}$ such that $k(G_1, G_2) = \langle \phi(G_1), \phi(G_2) \rangle$ for all $G_1, G_2 \in \mathcal{G}$.

Most of graph kernels have been defined as part of the R-convolutional framework [Vishwanathan et al., 2010] where graphs are decomposed into small substructures, then compared by counting the frequency of their matching substructures such as graphlets, shortest paths, trees, among others. The family of random walk kernels [Kashima et al., 2003, Gärtner et al., 2003] is perhaps one of the first successful efforts to design kernels between graphs that can be computed in polynomial time. For example, the ℓ-step Random Walk Kernel (ℓ-RWK) quantifies the similarity between a pair of input graphs based on the number of common walks up to length ℓ in the two graphs. An example of the 1-RWK over two toy graphs (see Figure 1.1) is illustrated in Figure 1.3. First, each input graph G_i is associated with a feature vector (also known as embedding) $\phi(G_i) \in \mathbb{R}^2$ whose j-th coordinate corresponds to the number of walks of length j in G_i (see Figure 1.2). Then, the final similarity score is computed by taking the dot product of the embeddings.

An important benefit of the embeddings generated by graph kernels is to allow efficient machine learning methods to be directly applied on graphs by extending the applicability of the whole arsenal of kernel methods [Hofmann et al., 2008] to graphs (e.g., for feature selection, classification, clustering, two-sample tests).

Such benefit blends well with the ability of kernels to operate on a feature space of arbitrary dimensionality without major computational difficulties, as long as the machine learning algorithms implementing the linear model of choice are rewritten exclusively in terms of inner products. To this end, one can evaluate the inner products in the feature space without explicitly having to describe each representation in that space. Therefore, the wealth of existing statistical approaches based on linear models for vectorial data is lifted to the nonlinear case. This observation is frequently referred to as the kernel trick by the machine learning community [Schölkopf, 2000]. The concept is illustrated in Figure 1.4: given a kernel function K defined on a vector space U, there is a vector space V and a function ϕ mapping U onto V such that: $K(x, y) = \phi(x) \cdot \phi(y)$ for each x, y in U. The value of the kernel calculated over two points x, y in the original space U can be therefore used as a substitute of the inner product of the corresponding points $\phi(x), \phi(y)$ in the transformed space V.

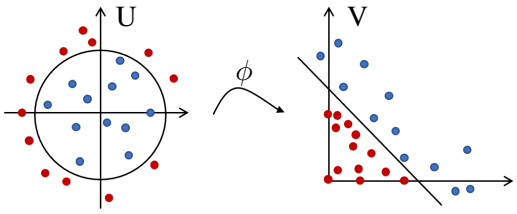


Figure 1.4 – U contains data not linearly separable; the mapping function ϕ remaps it into the space V, making the data linearly separable.

In addition, graph kernels offer provable theoretical guarantees and are easy to train thanks to the convexity of their optimization problem [Du et al., 2019] and thereby, provide a nice mathematical framework with effective insights over the underlying formalism. Another major advantage of measures which are defined directly on structured data is the merit of relieving the practitioner from the tedious operation of defining a vectorial representation of the data. From this point of view, graph kernels provide a natural and rich connection between graph space and machine learning. They have therefore known a wide success in several fields, especially in bioinformatics, computer vision and, natural language processing [Vishwanathan et al., 2008, Borgwardt et al., 2020, Kriege et al., 2020, Nikolentzos et al., 2021].

1.2 Graph Dynamics

Simple interaction rules have been a fundamental object of study in many sciences. In physics, the investigation of interacting particle systems contributed substantially in establishing statistical mechanics as a fundamental theory connecting the atomistic viewpoint to macroscopic phenomena [Liggett and Liggett, 1985]. In mathematics and theoretical computer science, the study of basic algorithmic processes on very simple distributed systems can be traced back to the study of cellular automata [Wolfram, 1984] and Conway's Game of Life [Gardner, 1970] formulated in the 70's. Since then, there has been a surge of interest in analyzing simple interaction rules from a computational point of view [Karp, 2011] as demonstrated by the application of the theory of distributed computing to the investigation of the algorithmic principles underlying the collective behavior of biological systems [Feinerman and Korman, 2013], such as insect colonies [Musco et al., 2017], flocks of birds [Chazelle, 2009], school of fish [Sumpter et al., 2008] and networks of neurons [Afek et al., 2011].

In the recent past, Multi-Agent Systems (for short, MAS) have gained a widespread recognition as a subfield of distributed AI. These are systems made up of multiple computing elements, known as *agents*, that interact with each other within an environment. Several real-life phenomena can be modeled by a MAS as discrete dynamical systems in the following manner: there is a set of agents with a neighborhood relation, and at each time step, the state of every agent is updated according to the states of its neighboring agents. Agents (i.e., nodes) interact by exchanging messages over an underlying communication graph. In this setting, dynamics are mathematical models which aim at investigating the long-term behavior of the MAS, and in particular, its ability to reach some form of *consensus*. The term dynamics refers to *simple* and *lightweight* protocols on graphs. We report below a definition that tries to formalize such concept that appeared in [Natale, 2017, Becchetti et al., 2020b, Becchetti et al., 2020a].

Definition 1.2.1. (Dynamics.) A dynamics is *a distributed algorithm characterized by a very simple structure in which the state of a node at round t depends only on its state and on a symmetric function of the multi-set of states of its neighbors at round $t-1$, while the update rule is the same for every graph and for every node and it does not change over time.*

A direct consequence of this definition is that nodes are identical between each other (i.e., *anonymous*) and they may not have a complete knowledge of the network, such as its topology and the total number of involved nodes.

Some instances of problems for which dynamics have been successfully employed range from opinion formation, epidemic spreading, innovation diffusion, to rumor spreading in social networks, to name a few. Famous examples of dynamics are Voter dynamics [Liggett and Liggett, 1985], 2-Median [Doerr et al., 2011], 2-Choice [Cooper et al., 2014], 3-Majority [Becchetti et al., 2017, Becchetti et al., 2016], Undecided-State [Becchetti et al., 2014] and Averaging [Becchetti et al., 2020b]. An important and natural class of non-linear dynamics are those based on *majority* rule. In the following, we focus on these dynamics which are object of analysis in this thesis.

Majority dynamics.

Consider a set of agents interacting by exchanging messages over a connected graph G, according to some communication model $\mathcal{M}$ which defines rules and constraints that agents follow in the communication, such as the synchronicity, the presence of possibly faulty links, the size of the

Algorithm 1 Synchronous Majority dynamics

Input: A connected graph $G = (V, E)$, a set of states $\Sigma = \{0, 1\}$ and a reversibility mode $rev \in \{\texttt{True}, \texttt{False}\}$.

Initialization: At round $t = 0$, every node $v \in V$ has a value $\mathbf{x}^{(0)} \in \Sigma$. Denote the set of nodes in state 0 at time t by $\mathcal{Z}^{(t)} = \{v \in V, \mathbf{x}^{(t)}(v) = 0\}$.

Update rule: **If** rev **then** $\mathcal{A}^{(t)} = V$, **else** $\mathcal{A}^{(t)} = \mathcal{Z}^{(t)}$. At each subsequent round $t \geq 1$, every node $v \in \mathcal{A}^{(t)}$ sets

$$\begin{cases} \mathbf{x}^{(t)}(v) = 0 \text{ if } \sum_{u \in \Gamma(v)} \mathbf{x}^{(t-1)}(u) < \left|\Gamma(v)\right|/2, \\ \mathbf{x}^{(t)}(v) = 1 \text{ if } \sum_{u \in \Gamma(v)} \mathbf{x}^{(t-1)}(u) > \left|\Gamma(v)\right|/2, \end{cases}$$

where $\Gamma(v)$ is the neighborhood of v, and ties are broken according to some fixed rule.

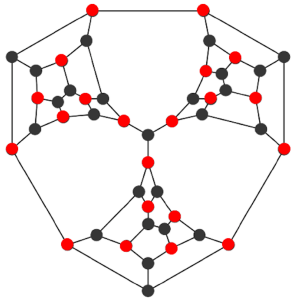


Figure 1.5 – Initial configuration at $t = 0$.

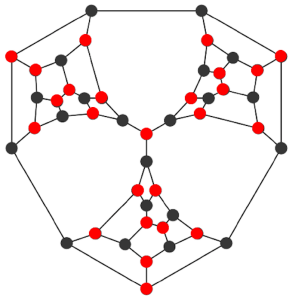


Figure 1.6 – Configuration at $t = 1$ for the reversible process.

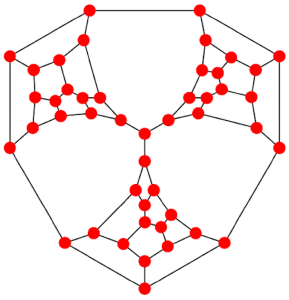


Figure 1.7 – Configuration at $t = 1$ for the irreversible process.

Figure 1.8 – One time step execution of the Synchronous Majority dynamics on the Tutte graph starting from an initial configuration of agents for both reversible and irreversible processes.

messages, among others. We summarize below some of the features described by the communication model $\mathcal{M}$ which are relevant in this study:

— **Activation.** If agents have a global clock, the communication is *synchronous* and, at each time step, every agent is active, that is, every agent is allowed to communicate.

On the other hand, when no global clock is available, the communication is *asynchronous* and, agents remain idle until one of them is activated by a random scheduler, in discrete time steps. After the communication has taken place, the active agent goes idle again. For example, an agent might become active at the arrival of an independent Poisson clock with rate 1 [Boyd et al., 2006].

— **Execution.** In this model, the exchanged messages are the states of the agents.

- **Update rule.** Initially, every node has a *binary* state: either 0 or 1. At each round, every active node pulls the states of its neighbors then updates its state to the most frequent one in its neighborhood. Ties, if any, are broken according to some fixed rule (e.g., the node keeps its current state).

- **Reversibility.** If one of the states (e.g., state 1, without loss of generality) is required to be *permanent*, i.e., every agent which is in state 1 (initially or at some subsequent round) keeps its state forever, then the dynamics is said *irreversible*. Otherwise, if agents with state 1 are allowed to update back to 0, then the dynamics is said *reversible*.

A majority dynamics under a synchronous activation is given in Algorithm 1.

Within such framework, the task of *Consensus* has attracted a lot of attention within different application domains. Consensus (also known as agreement) is a fundamental algorithmic problem where the system is required to converge to a stable configuration where all agents support the same state and this state must be valid, i.e., it must be supported by at least one agent in the initial configuration [Dijkstra, 1982]. In its most general formulation, we say that a protocol solves the consensus problem if it meets the following properties [Lynch, 1996]:

— *Agreement*: all nodes have the same state.

— *Validity*: the consensus state must be a valid one, i.e., a state which was initially supported by at least one node.

— *Termination*: every node eventually determine a final state.

If we allow the presence of adversaries (i.e., corrupted agents), the corresponding consensus problem goes under the name of Byzantine agreement [Rabin, 1983]. A protocol that solves the Byzantine agreement problem meets the previous property for every non-corrupted node.

In order to understand the long-term behavior of a MAS, some fundamental questions in the literature fall under the lenses of consensus. They typically concern determining the minimum size of an initial configuration of states that leads to consensus, as well as analyzing the amount of time steps required to reach it. This is motivated by the desire to understand for example under which conditions a video becomes viral, a new product or technology gets adopted by customers, an infection spreads to the population or a specific opinion becomes dominant, to name a few.

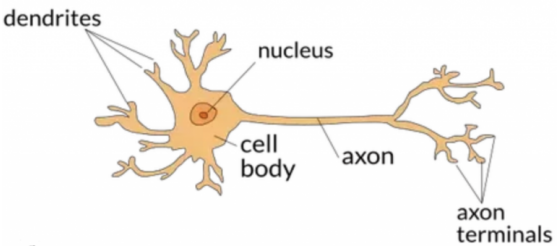


Figure 1.9 – A biological neuron.

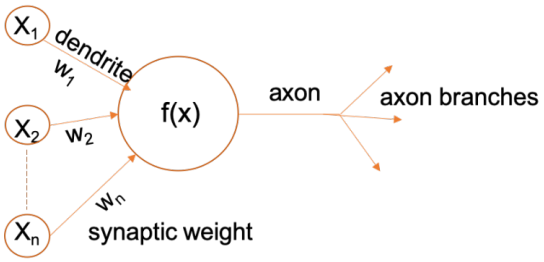


Figure 1.10 – An artificial neuron.

Figure 1.11 – A human neuron vs an artificial neuron.

1.3 Neural Network Compression

Artificial Neural Networks (ANNs, for short) are learning algorithms which were originally inspired by the nervous system of the human brain and modeled after it to perform specific computational tasks [Mitchell, 1997]. As illustrated schematically in Figure 1.11, similarly to a *biological* neuron which has dendrites to receive signals, a cell body to process them, and an axon to transmit them to other neurons through axon branches, an *artifical* neuron has a number of input channels, a processing unit, and an output that can fan out to multiple other neurons.

ANNs are made up of stacks of layers, with each layer comprising a collection of artificial neurons which are connected by edges (or connections) to other neurons, and where each connection is associated with a numerical value (or weight). A typical neural network contains an *input layer* representing the input data for the network, one or more *hidden layers* performing the computations necessary to produce the output, and an *output layer* representing the results of the neural network (see Figure 1.12). ANNs have led to a flurry of breakthroughs across a wide range of applications, and are typically classified into different architectures based on how the neurons are assembled and connected, in addition to their functionality (see Figure 1.13).

Many notable successes in machine learning have been achieved using neural network architectures with a massive number of trainable parameters. For example, the largest deep neural network ever created (GPT-3 [Brown et al., 2020]) shows impressive SOTA performance in various Natural Language Processing (NLP) tasks (e.g, translation, question-answering, cloze tests) and has 175 billion parameters. Therefore, such models come up with serious costs, requiring millions of dollars, as well as high amounts of energy and carbon emissions. As a result, much research in the field is aimed towards lowering the size of such networks while retaining good accuracy.

A substantial and extensive body of research [Deng et al., 2020, Han et al., 2015, Blalock et al., 2020] shows that one can *compress* a large network to a tiny fraction of its size by performing *pruning* (i.e., deleting some edges) while maintaining, sometimes even improving, its original accuracy. Pruning methods trace back to the 80's [LeCun et al., 1989, Mozer and Smolensky, 1988] and appear to be a mature and effective way of accomplishing large compression of neural networks as demonstrated by the several recent results that introduce sophisticated pruning, sparsification, and quantization techniques that result in significantly compressed model representations achieving SOTA accuracy [Deng et al., 2020, Cheng et al., 2020, Blalock et al., 2020]. Many of these

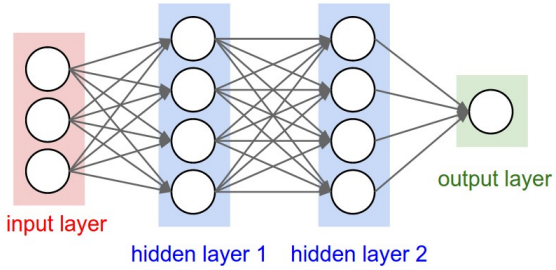


Figure 1.12 – An ANN with two hidden layers.

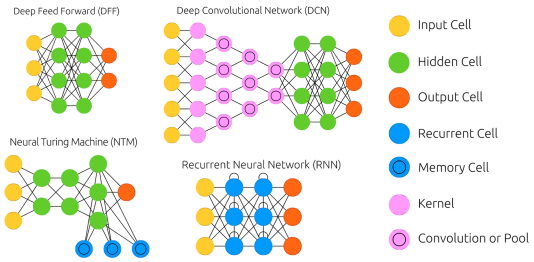


Figure 1.13 – Some architectures of an ANN [Leijnen and Veen, 2020].

Figure 1.14 – Artificial Neural Networks (ANNs) come in a varierty of architectures.

pruning techniques require several rounds of pruning and retraining, leading to a time-consuming and cumbersome to tune algorithms.

The *Lottery Ticket Hypothesis* (LTH, for short) [Frankle and Carbin, 2018] states that any randomly initialized neural network contains *lottery tickets*; that is, sparse subnetworks that can be trained just once and achieve the performance of the fully-trained original network. The existence of lottery tickets was supported experimentally based on a series of works using pruning methods [Frankle and Carbin, 2018, Lee et al., 2018]. If these lottery tickets can be found efficiently, then the computational burden of the pruning and retraining cycle can be avoided.

A striking finding was later on reported by Ramanujan et al. [Ramanujan et al., 2020]. It turns out that one does not even need to train the lottery tickets to obtain high accuracy: sparse models simply reside within larger random networks, and appropriate mere pruning can reveal them. This phenomenon goes by the name of the *Strong Lottery Ticket Hypothesis* (SLTH, for short). It states that any sufficiently large randomly initialized network contains, with high probability, subnetworks that can approximate any given sufficiently smaller neural network. In other terms, a sufficiently large randomly initialized network that can be successfully trained for a task, could instead be suitably pruned to obtain a network that achieves good accuracy, without any training. In particular, such hypothesis poses the deletion of connections (pruning) as a theoretically solid alternative to careful calibration of their weights (training).

Over the last few years, the SLTH was first proved for fully connected networks with ReLU activations by [Malach et al., 2020]. Specifically, they proved that one can approximate any target neural network of width d and depth ℓ, by pruning a sufficiently over-parameterized random network of width $\mathcal{O}(d^5\ell^2/\epsilon^2)$ and depth 2ℓ such that the gap between the pruned and target networks is bounded by ϵ. [Pensia et al., 2020a] and [Orseau et al., 2020] concurrently and independently improved this result by offering an exponential improvement on the over-parameterization (i.e., how much larger the random network has to be) required for the strong LTH to be true. Specifically, they reduced the width of the random initialized network to $\mathcal{O}(\text{poly}(d)\log(d\ell/\epsilon))$ and in addition, showed that this logarithmic over-parameterization is essentially optimal for constant depth networks.

Subset Sum Problem.

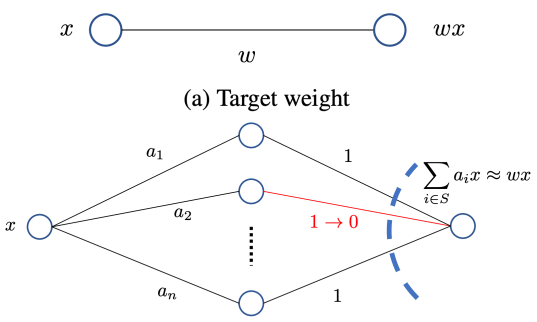
The theoretical analyses in the recent years [Pensia et al., 2020a, da Cunha et al., 2022b, Ferbach et al., 2022] behind the proofs of the SLTH, relied heavily on connecting pruning random neural networks to random instances of the Subset Sum Problem (SSP, for short). In the following, we explain the idea behind this connection by studying the case of approximating a single edge weight via pruning a random linear neural network.

Consider a simple target neural network (see (a) in Figure 1.15) whose output is given by $f(x) = w \cdot x$. We aim at approximating it by pruning an overparameterized network with random weights. We consider an overparameterized two-layered linear network (see (b) in Figure 1.15); for ease of illustration, we will further assume that the second layer is deterministic with all weights equal to 1. Hence, the network that we will prune, has the following linear architecture:

$$g(x) = \sum_{i=1}^{n} a_i x_i,$$

where the weights a_i are drawn i.i.d. from $\mathcal{U}([-1, 1])$.

Then, we ask: how large does the width n of the random network need to be in order to approximate wx (up to error ϵ) by pruning weights $\{a_i\}_{1 \leq i \leq n}$?



(a) Target weight

(b) Over-parameterized linear network

Figure 1.15 – Approximation of the weight w by pruning an over-parameterized two-layered network [Pensia et al., 2020a].

If $n = \mathcal{O}(1/\epsilon)$ then there exists a a_i that is ϵ-close to w [Malach et al., 2020]. In fact, we can also achieve the same approximation but with an *exponentially smaller* number of samples by not relying on just a single a_i but instead, a *subset* of $\{a_1, \ldots, a_n\}$ whose sum approximates the target weight. This is precisely the setting of the random SSP where [Lueker, 1998a] proved the existence of a subset $S \subseteq \{1, \ldots, n\}$ such that $|w - \sum_{i \in S} a_i| \leq \epsilon$ for $n = \mathcal{O}(\log(1/\epsilon))$. Therefore, we can prune the network by simply setting a_i to 0 for $i \notin S$ or equivalently, by pruning the output layer at indices that are not in S, as shown in red in Figure 1.15 (b).

1.4 Research Challenges and Contributions

In this section, we summarize our contributions and put them into context. We first present the contributions made in the context of graph learning for networks, for then summarizing the ones made in the context of dynamics within multi-agent systems. Next, we discuss the random subset sum problem in the context of neural network compression. Finally, we close by describing other miscellaneous contributions.

1.4.1 Learning on Networks

1.4.1.1 Network Kernels

With the continuous growing level of dynamicity, heterogeneity, and complexity of traffic data, anomaly detection remains one of the most critical tasks to ensure an efficient and flexible management of a network. Recently, driven by their empirical success in many domains, especially bioinformatics and computer vision, graph kernels [Kriege et al., 2020, Nikolentzos et al., 2021, Borgwardt et al., 2020] have attracted increasing attention. Our work aims at investigating their discrimination power for detecting vulnerabilities and distilling traffic in the field of networking.

In [Lesfari and Giroire, 2022], we propose `Nadege`, a new graph-based learning framework which aims at preventing anomalies from disrupting the network while providing assistance for traffic monitoring. Specifically, we design a graph kernel tailored for network profiling by leveraging propagation schemes which regularly adapt to contextual patterns. Moreover, we provide provably efficient algorithms and consider both offline and online detection policies. Finally, we demonstrate the potential of kernel-based models by conducting experiments on a wide variety of network environments.

1.4.2 Majority Dynamics

1.4.2.1 (Reversible) Biased Opinion Dynamics

We study opinion dynamics in multi-agent networks where agents hold binary opinions and are influenced by their neighbors while being biased towards one of the two opinions, called the superior opinion. The dynamics is modeled by the following process: at each round, a randomly selected agent chooses the superior opinion with some probability α, and with probability $1 - \alpha$ it conforms to the opinion manifested by the majority of its neighbors.

In [Lesfari et al., 2022], we exhibit classes of network topologies for which we prove that the expected time for consensus on the superior opinion can be exponential. This answers an open conjecture in the literature. In contrast, we show that in all cubic graphs, convergence occurs after a polynomial number of rounds for every α.

We rely on new structural graph properties by characterizing the opinion formation in terms of multiple domination, stable and decreasing structures in graphs, providing an interplay between bias, consensus and network structure. Finally, we provide both theoretical and experimental evidence for the existence of decreasing structures and relate it to the rich behavior observed on the expected convergence time of the opinion diffusion model.

1.4.2.2 (Irreversible) Deterministic Boostrap Percolation

Let $r \geq 1$ be any non negative integer and let $G = (V, E)$ be any undirected graph in which a subset $D \subseteq V$ of vertices are initially *infected*. We consider the process in which, at every step, each non-infected vertex with at least r infected neighbours becomes infected and an infected vertex never becomes non-infected. The problem consists in determining the minimum size $s_r(G)$ of an initially infected vertices set D that eventually infects the whole graph G. This problem is closely related to cellular automata, to percolation problems and to the Game of Life studied by John Conway. Note that $s_1(G) = 1$ for any connected graph G. The case when G is the $n \times n$ grid, $G_{n\times n}$, and $r = 2$ is well known and appears in many puzzle books, in particular due to the elegant proof that shows that $s_2(G_{n\times n}) = n$ for all $n \in \mathbb{N}$.

In [Benevides et al., 2021], we study the cases of square grids, $G_{n\times n}$, and tori, $T_{n\times n}$, when $r \in \{3, 4\}$. We show that $s_3(G_{n\times n}) = \lceil \frac{n^2+2n+4}{3} \rceil$ for every n even and that $\lceil \frac{n^2+2n}{3} \rceil \leq s_3(G_{n\times n}) \leq \lceil \frac{n^2+2n}{3} \rceil + 1$ for any n odd. When n is odd, we show that both bounds are reached, namely $s_3(G_{n\times n}) = \lceil \frac{n^2+2n}{3} \rceil$ if $n \equiv 5 \pmod 6$ or $n = 2^p - 1$ for any $p \in \mathbb{N}^*$, and $s_3(G_{n\times n}) = \lceil \frac{n^2+2n}{3} \rceil + 1$ if $n \in \{9, 13\}$. Finally, for all $n \in \mathbb{N}$, we give the exact expression of $s_3(T_{n\times n})$.

1.4.3 Neural Network Compression

1.4.3.1 Random Subset Sum Problem

The average properties of the well-known *Subset Sum Problem* can be studied by the means of its randomised version, where we are given a target value z, random variables $X_1, \ldots, X_n$, and an error parameter $\varepsilon > 0$, and we seek a subset of the X_i's whose sum approximates z up to error ε. In this setup, it has been shown that, under mild assumptions on the distribution of the random variables, a sample of size $\mathcal{O}\left(\log(1/\varepsilon)\right)$ suffices to obtain, with high probability, approximations for all values in $[-1/2, 1/2]$. Recently, this result has been rediscovered outside the algorithms community, enabling meaningful progress in other fields. In [da Cunha et al., 2022a], we present an alternative proof for this theorem, with a more direct approach and resourcing to more elementary tools, in the hope of disseminating it even further.

1.4.3.2 Multidimensional Random Subset Sum Problem

In the Random Subset Sum Problem, given n i.i.d. random variables $X_1, ..., X_n$, we wish to approximate any point $z \in [-1, 1]$ as the sum of a suitable subset $X_{i_1(z)}, ..., X_{i_s(z)}$ of them, up to error ε. Despite its simple statement, this problem is of fundamental interest to both theoretical computer science and statistical mechanics. More recently, it gained renewed attention for its implications in the theory of Artificial Neural Networks. An obvious multidimensional generalisation of the problem is to consider n i.i.d. d-dimensional random vectors, with the objective of approximating every point $\mathbf{z} \in [-1, 1]^d$. Rather surprisingly, after Lueker's 1998 proof that, in the one-dimensional setting, $n = O(\log \frac{1}{\varepsilon})$ samples guarantee the approximation property with high probability, little progress has been made on achieving the above generalisation.

In [Becchetti et al., 2022], we prove that, in d dimensions, $n = O(d^3 \log \frac{1}{\varepsilon} \cdot (\log \frac{1}{\varepsilon} + \log d))$ samples suffice for the approximation property to hold with high probability. As an application highlighting the potential interest of this result, we prove that a recently proposed neural network

model exhibits *universality*: with high probability, the model can approximate any neural network within a polynomial overhead in the number of parameters.

1.4.4 Other Works

1.4.4.1 NFV & SDN Protection in Network Slicing

Network Function Virtualization (NFV) enables the virtualization of core-business network functions on top of a NFV infrastructure. NFV has gained an increasing attention in the telecommunication field these last few years. Virtual network functions (VNFs) can be represented by a set of virtual network function components (VNFCs). These VNFCs are typically designed with a redundancy scheme and need to be deployed against failures of, e.g., compute servers. However, such deployment must respect a particular resiliency mechanism for protection purposes. Therefore, choosing an efficient mapping of VNFCs to the compute servers is a challenging problem in the optimization of the software-defined, virtualization-based next generation of networks.

In [Lesfari et al., 2021], we model the problem of reliable VNFCs placement under anti-affinity constraints using several optimization techniques. A novel approach based on an extension of bin packing is proposed. We perform a comprehensive evaluation in terms of performance under real-world ISP networks along with synthetic traces. We show that our methods can calculate rapidly efficient solutions for large instances.

1.4.4.2 Resource Allocation with Deep RL

The emerging 5G induces a great diversity of use cases, a multiplication of the number of connections, an increase in throughput as well as stronger constraints in terms of quality of service such as low latency and isolation of requests. To support these new constraints, Network Function Virtualization (NFV) and Software Defined Network (SDN) technologies have been coupled to introduce the network slicing paradigm. Due to the high dynamicity of the demands, it is crucial to regularly reconfigure the network slices in order to maintain an efficient provisioning of the network. A major concern is to find the best frequency to carry out these reconfigurations, as there is a trade-off between a reduced network congestion and the additional costs induced by the reconfiguration.

In [Gausseran et al., 2022], we tackle the problem of deciding the best moment to reconfigure by taking into account this trade-off. By coupling Deep Reinforcement Learning for decision and a Column Generation algorithm to compute the reconfiguration, we propose `Deep-REC` and show that choosing the best time during the day to reconfigure allows to maximize the profit of the network operator while minimizing the use of network resources and the congestion of the network. Moreover, by selecting the best moment to reconfigure, our approach allows to decrease the number of needed reconfigurations compared to an algorithm doing periodic reconfigurations during the day.

1.5 Plan of the Thesis

Table 1.1 illustrates the organization and structure of the contributions in the Thesis.

Topic	Chapter
Graph Kernels for Network Anomaly Detection	2
Opinion Dynamics using Graph k-domination	3
Deterministic Boostrap Percolation	4
Random Subset Sum Problem	5
Multidimensional Random Subset Sum Problem	6
NFV & SDN Protection in Network Slicing	7
Resource Allocation with Deep Reinforcement Learning	8

TABLE 1.1 – Topic organization within the Thesis.

1.6 List of Publications

International Conferences

[1] **Hicham Lesfari**, Frédéric Giroire. *Nadege: When Graph Kernels meet Network Anomaly Detection*, Proceedings of the IEEE International Conference on Computer Communications (`IEEE INFOCOM 2022`), London, United Kingdom, May 2022.

[2] **Hicham Lesfari**, Frédéric Giroire, Stéphane Pérennes. *Biased Majority Opinion Dynamics: Exploiting graph k-domination*, Proceedings of the International Joint Conference on Artificial Intelligence (`IJCAI 2022`), Vienna, Austria, July 2022.

[3] Adrien Gausseran, Redha Alliche, **Hicham Lesfari**, Ramon Aparicio-Pardo, Frédéric Giroire, Joanna Moulierac. *When to Reconfigure my Network Slices ? A Deep Reinforcement Learning Approach*, Proceedings of the IEEE International Conference on Cloud Networking (`CloudNet 2022`), Paris, France, November 2022.

[4] **Hicham Lesfari**, Frédéric Giroire, Giuseppe Di Lena, Chidung Lac. *A multidimensional coloured packing approach for network slicing with dedicated protection*, Proceedings of IEEE Global Communications Conference (`GLOBECOM 2021`), Madrid, Spain, December 2021.

International Journals

[5] Fabricio Siqueira Benevides, Jean-Claude Bermond, **Hicham Lesfari**, Nicolas Nisse. *Minimum lethal sets in grids and tori under 3-neighbour bootstrap percolation*, `European Journal of Combinatorics`, 2022.

Posters at International Conferences

[6] **Hicham Lesfari**, Frédéric Giroire, Stéphane Pérennes. *Biased Majority Opinion Dynamics: Exploiting Graph k-domination*, Highlights of Algorithms (`HALG 2022`), London, United Kingdom, June 2022.

National Conferences

[7] **Hicham Lesfari**, Frédéric Giroire. *Une rencontre entre les noyaux de graphes et la détection d'anomalies dans les réseaux*, 23ème Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (ALGOTEL 2021), La Rochelle, France, September 2021.

Research Reports

[8] Luca Becchetti, Andrea Clementi, Arthur C. W. da Cunha, Francesco D'Amore, **Hicham Lesfari**, Emanuele Natale, Luca Trevisan. *On the Multidimensional Random Subset Sum Problem*.

[9] Arthur C. W. da Cunha, Francesco D'Amore, Frédéric Giroire, **Hicham Lesfari**, Emanuele Natale, Laurent Viennot. *Revisiting the Random Subset Sum Problem*.

Learning on Networks

CHAPTER 2

Network Kernels

Contents

2.1 Introduction

Learning on graph-structured data has gained a notable momentum with recent advances [Wu et al., 2020] in Artificial Intelligence (AI). While graphs provide an ubiquitous data structure to represent complex structures and interactions, readily blending them with standard machine learning algorithms brings several challenges [Cai et al., 2018]. Of particular relevance is the progress in designing flexible and expressive methods [Hamilton et al., 2017b] for quantifying similarities between graphs.

A graph kernel is a kernel function [Schölkopf, 2000] defined directly on pairs of graphs, and enables to compare between them. It corresponds to an inner product that is equivalent to an embedding of the set of graphs into a possibly high-dimensional Hilbert space [Berlinet and Thomas-Agnan, 2011]. An important benefit of such embedding is to allow efficient machine learning methods to be directly applied on graphs by extending the applicability of the whole arsenal of kernel methods [Hofmann et al., 2008] to graphs (e.g., for feature selection, classification, clustering, two-sample tests).

Furthermore, graph kernels offer provable theoretical guarantees and are easy to train thanks to the convexity of their optimization problem [Du et al., 2019]. Thereby, they provide a rich connection between graph space and machine learning. However, despite their wide success in other domains, especially in bioinformatics, computer vision and, natural language processing (NLP) [Kriege et al., 2020], they have received little attention in the networking field.

Among the prominent networking problems, anomaly detection is a crucial task in network security. According to recent studies, cybercrime damage costs are predicted to reach 10.5 trillion dollars annually by 2025 [magazine, 2021]. This alarming situation is further demonstrated by the adoption of recent cybersecurity regulations [Srinivas et al., 2019]. However, efficiently identifying the increasingly complex malicious activities from network traffic brings some major challenges.

Most traditional methods consider the statistical variations of traffic volume in order to detect anomalous traffic. Such approach is convenient for high-rate attacks where, for instance, a host is deemed as performing a port scan if a sudden increase in the volume of destination port numbers is noticed. However, low-rate attacks such as link-flooding [Liaskos et al., 2016] are not visible in a large network and thus, are very difficult to discover. Moreover, volume-based techniques are inadequate to capture the correlations in communications between hosts and the impact of cross-host anomalous activities since they focus on the statistics of traffic sent by individual hosts.

In this chapter, we present `Nadege`, a graph-based learning framework which aims at detecting anomalous hosts in networks and classifying traffic. First, we model the communication patterns of each host by a graph structure of multiple network flows that provides a compact representation of its network activity. Then, we develop a new graph kernel to measure the similarities between these activity graphs by leveraging correlations between flows of all the hosts in a network, such that it is possible to detect stealthy anomalies.

While considering the local structural information of each host is important, neglecting the context is insufficient to distinguish whether a network flow is triggered with or without other anomalous flows. Therefore, we increment our graph kernel with the ability to learn representative latent features from activity graphs with the aid of both local and contextual views. The key observation behind `Nadege` is the collective integration of the graph structure and node attributes information in our graph kernel at different network layers. Such integration enables us to charac-

terize each change of anomalies by specific patterns of a particular feature set.
Our contributions can be summarized as follows.

— We advocate a general end-to-end learning framework to detect traffic anomalies and classify network traffic using a graph-based kernel approach.

— We prototype our framework by providing efficient algorithms, with theoretical approximation guarantees, to ensure a fast and accurate feature extraction. To the best of our knowledge, this is the first work to formulate a graph kernel crafted from a networking perspective and that is tailored to the analysis of network flows.

— Finally, we perform extensive experiments using real-word traces from different providers (e.g., ISP, University). In our evaluation based on a variety of network environments, both our graph kernel and `Nadege` achieve significant improvements over the popular baselines.

The rest of this chapter is organized as follows. In Section 2.2, we review the related works. In Section 2.3, we present details and analysis of each component of our framework. In Section 2.4.2, we describe the learning design choices. Then, we evaluate `Nadege` on several traces and discuss the obtained results. Finally, we draw our conclusions in Section 2.5.

2.2 Related work

Graph kernels. Most of graph kernels have been defined as part of the R-convolutional framework [Vishwanathan et al., 2010] where graphs are decomposed into small substructures, then compared by counting the frequency of their matching substructures such as graphlets, shortest paths, trees, among others. However, it is worth noting that an increase in the size of substructures leads to a decrease in the probability of two graphs sharing common substructures, which results in the so called diagonal dominance issue [Kriege et al., 2020], leading to models prone to overfitting. Our fingerprint kernel differs significantly from the R-convolution based graph kernels. We capture similarity between graphs by an implicit extraction of structural nodes and pairs of nodes, avoiding consequently a direct decomposition into substructures.
Another family of works derives kernels based on spectral, propagation or geometric approaches. For example, optimal assignment kernels [Kriege et al., 2016] represent a departure from the R-convolution framework by computing a matching between substructures of objects such that the overall object pairwise similarity is maximized. Unlike our proposed kernel, they are unfortunately not guaranteed to be positive semi-definite for all choices of base kernels [Vert, 2008]. For a further review of graph kernels, we refer the reader to the surveys [Borgwardt et al., 2020, Kriege et al., 2020, Nikolentzos et al., 2021].

Graph-based anomaly detection. Much interest has been generated for anomaly detection methods using graphs. The first family of techniques explores probabilistic or graph similarity measures [Le Bars and Kalogeratos, 2019]. The second family is based on machine learning and comes in supervised or unsupervised branches [Khatuya et al., 2018]. Our hybrid framework combines techniques from both families by devising a learning-flavored similarity kernel from a networking perspective. The closest works to ours are [Yao et al., 2019] and [Harshaw et al., 2016].
In [Yao et al., 2019], the authors combine the standard shortest-path graph kernel [Borgwardt and Kriegel, 2005] with a deep convolutional neural network to analyze network attacks. However, since shortest-paths do not consider neighborhood structures, the graph similarity is only captured at fine granularities [Ye et al., 2019]. Moreover, the shortest-path kernel requires a quartic time

Symbol	Description
$\mathcal{A}$	Activity graph
S	Source set of size s
T	Destination set of size t
l_i	The i-th layer of $\mathcal{A}$
J	Weighted adjacency matrix
$\widetilde{J}$	Normalized adjacency
D	Weighted degree matrix
P	Transition probability matrix
$\widetilde{\mathcal{L}}$	Normalized Laplacian
$\mathcal{S}$	Set of activity graphs
$\mathcal{M}_{\mathcal{S}}$	Merge graph

TABLE 2.1 – Glossary of notations.

complexity in the size of the graphs and thus, is expensive to compute for very large graphs. In addition, deep neural networks require a large amount of training data and are computationally expensive, especially when considering the entire network flows.
[Harshaw et al., 2016] introduces a graph-mining technique to detect network anomalies. In their model, they consider a single host-based reduced graph representation built from self-harvested network flow data. To detect anomalies at the host level, they first count the automorphism orbits of graphlets—small induced subgraphs that describe local topology. Then, a robust algorithm fits the gaussian distribution using the Minimum Covariance Determinant method of [Rousseeuw and Driessen, 1999] on the sequence of graphlets for outlier detection. While the traffic flows collected have similar attributes, our per-host graph model uses a clear-cut representation of all network flow information. We compare the performance of `Nadege` to the ones of existing methods in Section 2.4.

2.3 Framework description

In this section, we define the notation used in the rest of the chapter. Then, we introduce each component of `Nadege` (which stands for **N**etwork **A**nomaly **DE**tection with **G**raph k**E**rnels).

2.3.1 Preliminaries and Notation

Consider an undirected labeled graph $G = (V, E, l, w)$ where V is a set of graph vertices of size $n = |V|$ and E is a set of graph edges of size $m = |E|$, while $l : V \to \mathbb{N}$ and $w : E \to \mathbb{N}$ are functions that assign labels from a set of positive integers to nodes and edges, respectively. The weighted adjacency matrix J of G is an $n \times n$ symmetric matrix such that $J[i, j] = w_{ij}$. The weighted degree matrix D of G is a diagonal matrix such that $D[i, i] = \sum_{(i,j)\in E} w_{ij}$. The first and last i rows of D are denoted by $D_{:i}$ and $D_{i:}$, respectively. Moreover, we denote a set by $\{a, b, ..., c\}$, an ordered set (tuple) by $(a, b, ..., c)$ and a multiset by $\{\!\{a, b, ..., c\}\!\}$. $\mathcal{N}_v^i$ is the i-th neighborhood of vertex v. Finally, $\odot, \oplus, \wedge, \dot{\vee}$ denote the Hadamard product, the concatenation, logical conjunction and exclusive disjunction operators, respectively.
A walk of arbitrary size k initiated from node v_0 is defined as a sequence of vertices $\{\!\{v_0, \ldots, v_k\}\!\}$

such that $\{v_i, v_{i+1}\} \in E$ for $0 \leq i \leq k-1$. A path is a walk that consists of all distinct vertices and edges. A random walk on G is a discrete-time Markov chain $(X_0, X_1, X_2...)$ endowed with the Markov property [Levin and Peres, 2017], which induces the transition matrix $P = JD^{-1}$. We summarize the notation in Table 2.1.

2.3.2 Activity Graphs

Due to privacy, legal, and technological constraints [Yi et al., 2015], it is important to consider a profiling mechanism that does not expose private or personally identifiable information (PII) and, a fortiori, does not access the packet payload. In addition, computing behavioral statistics over databases of network streams is incomprehensive due to the existence of redundant features and the difficulty in interpretation without further processing. Thus, our aim is to use the flow records gathered by each host in order to build a compact graph structure to track its profile activity, while limiting the redundancy. We model a network flow by a path graph (see Fig. 2.1(a)) which acts as a summarized indicator.

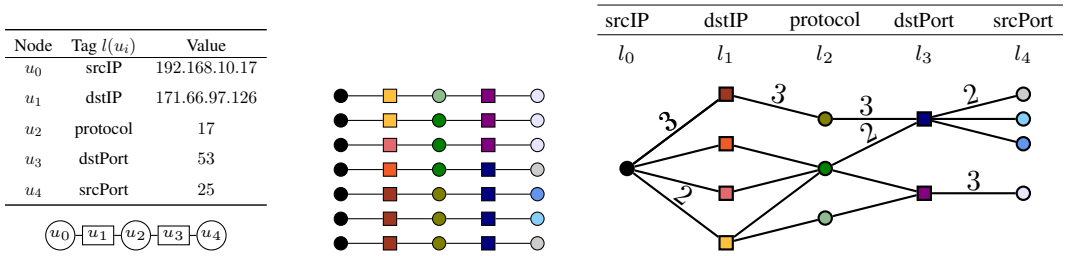
Definition 2.3.1. A bag of network flows associated with a host is a set of flows, i.e., of labelled chains such that each chain $\mathcal{P} = (u_0, ..., u_f)$ is made up of a sequence of identifiers which i-th node is tagged by $l(u_i)$ and, where u_0, u_f correspond to the end-points of the flow.

Observing all the incoming and outgoing flows of a host during a given time window, gives rise to a set $\mathcal{B}$ of network flows. Clearly, in a bag of flows, the source IP is always unique (as depicted by the black circles in Fig. 2.1(b)). This collection is then used to build the profile activity of a host according to a two-step procedure: we start by arranging a graph in layers $l_0, ..., l_f$ corresponding to the tagged identifiers in the flows. To make the layers intuitively tractable, we will refer by source (resp. destination) layers those located at an index position which is even (resp. odd). Next, we merge the network flows while counting the frequency of appearance of each edge across the bag $\mathcal{B}$. Thus, each flow creates a path starting from the host IP address on the first layer and traversing entities in each subsequent layer in the graph (see Fig. 2.1(c)).

Definition 2.3.2. (Activity graph): Let $\mathcal{B}$ be a bag of network flows associated with a host. An activity graph of a host is a weighted connected graph defined by $\mathcal{A} = (\mathcal{B}, E, w)$ where $w : E \to \mathbb{N}$ is the count of flows contributing to each edge.

Such representation served useful when it comes to identify application patterns [Karagiannis et al., 2007] in a rather straightforward and intuitive manner, without accessing the packet payload. We further incorporate weights in order to track more latent temporal features, such as the number of packets or bytes for all flows transiting a given path. The main key advantages of handling activity graphs are: (i) they offer a succint representation of all flow information while operating in a single structure, (ii) they evolve in a way that reflects evolutionary changes over time, and (iii) they allow intuitive interpretable information through the pairwise interaction between layers. We note that activity graphs can be used in conjunction with several protocols running over TCP/IP in a wide variety of environments such as the MQTT protocol [Lin et al., 2017] in Internet of Things (IoT), augmenting thereby their diameter while remaining expressive.
We notice that any given activity graph has a bipartite structure, represented by a partition (S, T) such that $S = \bigcup_{i \equiv 0[2]} l_{i+1}$ collects nodes* from source layers, while $T = \bigcup_{i \equiv 1[2]} l_{i+1}$ collects

*. The term node indicates the component of an activity graph, while the term host indicates a communicating device.



(a) A network flow.

(b) Bag of network flows.

(c) An activity graph reflecting communication patterns.

Figure 2.1 – Generation instance of an activity graph ($f = 4$).

nodes from destination layers. The next proposition shows that the biadjacency matrix, denoted by L_{st}, suffices to uniquely represent an activity graph from its partition (S, T) with source (resp. destination) set S (resp. T) of size denoted by s (resp. t). The proof follows directly from the bipartitness of an activity graph.

Proposition 1. *Let $k \in \mathbb{N}$. The weighted adjacency matrix J for an activity graph and its powers can be expressed as*

$$J = \begin{array}{c} \\ S \\ T \end{array} \begin{array}{c} \begin{array}{cc} S & T \end{array} \\ \begin{bmatrix} \mathbf{0} & \mathbf{L}_{st} \\ \mathbf{L}_{st}^T & \mathbf{0} \end{bmatrix} \end{array}, \quad J^{2k} = \begin{bmatrix} (\mathbf{L}_{st}\mathbf{L}_{st}^T)^k & \mathbf{0} \\ \mathbf{0} & (\mathbf{L}_{st}^T\mathbf{L}_{st})^k \end{bmatrix}$$

and

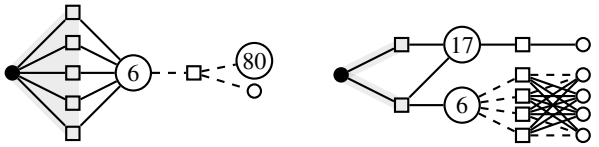
$$J^{2k+1} = \begin{bmatrix} \mathbf{0} & (\mathbf{L}_{st}\mathbf{L}_{st}^T)^k\mathbf{L}_{st} \\ (\mathbf{L}_{st}^T\mathbf{L}_{st})^k\mathbf{L}_{st}^T & \mathbf{0} \end{bmatrix}$$

2.3.3 Intrinsic Fingerprint

In order to capture the local-global characterization of an activity graph, we need a processing mechanism with the ability to zoom in and out the topological environment of each node. To this end, we capitalize on random-walk based methods for their relationship to multiple graph properties. For instance, [Zhang et al., 2018] derived features based on the isomorphism-invariance property of the return probabilities of random walks. The efficiency of these features lies on their convergence to a certain value (denoted by π) known as the stationary probability in Markov chain theory [Levin and Peres, 2017]. However, such a stable distribution does not exist for activity graphs due to their bipartiteness. To circumvent this limitation, we explore the graph using an ergodic random walk variant $\widehat{W}$ where at each time step, with probability $1/2$, the walk remains at its current node, otherwise it jumps to one of its neighbors.

Proposition 2. *Let $\mathcal{A}$ be an activity graph. Let $R_{\mathcal{A}}$ denote the transition probability matrix induced by $\widehat{W}$ on $\mathcal{A}$. Then $R_{\mathcal{A}} = \frac{1}{2}(I_{\mathcal{A}} + J_{\mathcal{A}}D_{\mathcal{A}}^{-1})$.*

Proof.



(a) Benign host. (b) Attack host.

Figure 2.2 – Structural roles for some intrinsic components.

The proof follows by noting that $R_{\mathcal{A}} = \frac{1}{2}(I_{\mathcal{A}} + P_{\mathcal{A}})$ where

$$[P_{\mathcal{A}}]_{ij} = \begin{cases} \frac{1}{D[j,j]} & \text{if } \{i,j\} \in E \\ 0 & \text{otherwise} \end{cases}$$

is the probability of going from node j to i.

□

Definition 1. *(Intrinsic Fingerprint)*
Let $\tau > 0$. The intrinsic fingerprint of an activity graph $\mathcal{A}$ is the set of vectors

$$i_{\mathcal{A}} = \{i_v, v \in V_{\mathcal{A}}\}, \tag{2.1}$$

where $i_v = \left[R^1_{\mathcal{A}}[v,v], ..., R^{\tau+1}_{\mathcal{A}}[v,v]\right]^T$ and τ are referred to by the intrinsic component and range, respectively.

The intrinsic fingerprint provides a rich information on the graph structure and can be seen as a subgraph centrality measure. By focusing on closed walks, it captures the interaction of a node with the subgraphs involving it. Moreover, such structural role characterization blends well with the interpretation in terms of the host profile activity. Fig. 2.2(a) displays an instance of a benign host in our traces corresponding to an HTTP web server. Fig. 2.2(b) displays an attack where the host is attempting to connect through TCP to potential vulnerable ports at the same destination host.
At the first layer, the intrinsic fingerprint captures the popularity of a host in terms of the number of other hosts it communicates with. At the third level, as illustrated by the dashed lines, the star and cycle subgraphs involving the TCP node reflect functional role patterns where the last layer benefits from the knowledge acquired in the previous layers. In the former subgraph, portrayed by a knot at the destination port layer, the host is providing a service which translates into the use of a single port for the majority of its interactions. In the latter subgraph, closed walks of order four unveil in this case a port scan where the host identifies vulnerabilities at specific destination ports.

A straightforward computation of $i_{\mathcal{A}}$ takes $\mathcal{O}(\tau(s+t)^3)$ by successively computing the powers of R. Since only the diagonals of the transition matrices are required, we use Proposition 3, based on SVD, which leads to Algo. 2 and shows that we can do better. We note that SVD can be approximated in time linear to the number of non-zero entries [Halko et al., 2011], which makes the practical computation fast due to the sparsity of the biadjacency matrix.

Proposition 3. *Let $\mathcal{A}$ be an activity graph. $i_{\mathcal{A}}$ can be computed in $\mathcal{O}(st\min(s,t) + (\tau+1)(s^2 + t^2) + \tau(\tau+1)(s+t))$.*

Proof.

For clarity of notation, we omit the graph subscript $\mathcal{A}$. Let us fix $k > 0$. We write

$$R^k = \frac{1}{2^k} \sum_{i=0}^{k} \binom{k}{i} (JD^{-1})^i = \frac{1}{2^k} \sum_{i=0}^{k} \binom{k}{i} D^{\frac{1}{2}} \widetilde{J}^i D^{-\frac{1}{2}} \tag{2.2}$$

where $\widetilde{J} = D^{-\frac{1}{2}} J D^{-\frac{1}{2}}$ is symmetric. We have

$$\widetilde{J} = \begin{bmatrix} 0 & D_s^{-\frac{1}{2}} L_{st} D_t^{-\frac{1}{2}} \\ D_t^{-\frac{1}{2}} L_{st}^T D_s^{-\frac{1}{2}} & 0 \end{bmatrix} \triangleq \begin{bmatrix} 0 & N \\ N^T & 0 \end{bmatrix}$$

Using Proposition 1, we note that

$$\widetilde{J}^{2i} = \begin{bmatrix} U(\Sigma\Sigma^T)^i U^T & 0 \\ 0 & V(\Sigma^T\Sigma)^i V^T \end{bmatrix}$$

and $\widetilde{J}^{2i+1}[v, v] = 0$, where we use a singular value decomposition on $N = U\Sigma V^T$ with U, V real orthogonal matrices and Σ a diagonal with non-negative real values $\sigma_1 \geq ... \geq \sigma_{f:=\min(s,t)}$. Hence, Eq. (2.2) yields

$$i^{(k)} \triangleq \left[R^k[v_1, v_1], ..., R^k[v_n, v_n]\right]^T = \frac{1}{2^k} \sum_{j \equiv 0[2]}^{k} \binom{k}{j} Y^j \tag{2.3}$$

where $Y^j = \left((U \odot U)W^{2j}\right) \oplus \left((V \odot V)W^{2j}\right)$ with $W^j = (\sigma_1^j, ..., \sigma_f^j)^T$.

Decomposing N via SVD takes $\mathcal{O}(st \min(s,t))$ while computing Y^j for a given j requires $\mathcal{O}(s^2 + t^2)$. By performing $\mathcal{O}(\tau(\tau+1)(s+t))$ additions, using Eq. (2.3), the result follows.

□

Clearly, the intrinsic fingerprints are sensitive to the choice of the intrinsic range which intuitively, corresponds to the depth of the graph explorations initiated from the nodes. Therefore, we need a provable guarantee which bridges the exploration intensity with the informativeness of the extracted features. To this end, Theorem 2.3.1 tells us how far we need to explore, while ensuring a controlled view on the scope towards the stationary distribution.

Theorem 2.3.1. *Let $\mathcal{A}$ be an activity graph and δ in $(0, 1]$. If*

$$\tau_\delta \geq \frac{2}{1 + \frac{1}{\sqrt{st}}} \log(\frac{n}{\delta}), \tag{2.4}$$

then $\nu(p_t, \pi) \leq \delta$, where $\nu(p, q) = \|p - q\|_1$ denotes the total variational distance and p_t the probability distribution of the position of the random walk at time t.

Proof.

Let p_t denotes the probability distribution of the position of the random walk at time t. First, we express a lower bound on $t(\delta)$ where

$$t(\delta) = \max_{p_0} \arg\min_t \{\nu(p_t, \pi) \leq \delta\}.$$

Algorithm 2 Intrinsic Fingerprint

Input: An activity graph $\mathcal{A}$, depth parameter δ in (0,1]
Output: $i_{\mathcal{A}}$

1: Compute $N = D_{:s}^{-\frac{1}{2}} L_{st} D_{t:}^{-\frac{1}{2}}$
2: $N = U\Sigma V^T$ ▷ SVD Reduction
3: Compute τ_δ according to Eq. (2.4)
4: **parallel for** $k = 1, ..., \tau_\delta + 1$ **do**
5: Compute $i_{\mathcal{A}}^{(k)}$ according to Eq. (2.3)
6: **parallel for** v in $V_{\mathcal{A}}$ **do**
7: Add $i_{\mathcal{A}}^{(k)}[v]$ to i_v ▷ Intrinsic Component
8: Collect $i_{\mathcal{A}}$ according to Eq. (2.1)
9: **return** $i_{\mathcal{A}}$

Observe that $p_t = R^t p_0$ and $\pi(v) = \frac{d(v)}{2|E|}$ where

$$R = \frac{1}{2}(I + JD^{-1}) = I - \frac{1}{2} D^{\frac{1}{2}} \widetilde{\mathcal{L}} D^{-\frac{1}{2}},$$

with $\widetilde{\mathcal{L}} = I - D^{-\frac{1}{2}} J D^{-\frac{1}{2}}$ the normalized Laplacian. Let $\lambda_1 = 0 \leq ... \leq \lambda_n = 2$ denote the spectrum of $\widetilde{\mathcal{L}}$ associated with the eigenvectors $\{r_i\}_{1 \leq i \leq n}$. Since

$$\forall i \in \{1, ...n\} : RD^{\frac{1}{2}} r_i = \alpha_i D^{\frac{1}{2}} r_i,$$

where $\alpha_i = 1 - \frac{\lambda_i}{2}$, then $\{(\alpha_i, D^{\frac{1}{2}} r_i)\}_{1 \leq i \leq n}$ are the eigenpairs of R and $\{D^{\frac{1}{2}} r_i\}_i$ forms a basis of $\mathbb{R}^n$ as $D^{\frac{1}{2}}$ has rank n. Hence, for a set of $\{\mu_i \triangleq r_i^T D^{-\frac{1}{2}} p_0\}_{1 \leq i \leq n}$ and an arbitrary starting distribution p_0, as $\alpha_1 = 1$ and $\mu_1 D^{\frac{1}{2}} r_1 = \pi$, it holds

$$\nu(p_t, \pi) = \left\| R^t p_0 - \pi \right\|_1 \leq \sqrt{n} \left\| \sum_{i=2}^{n} \mu_i \alpha_i^t D^{\frac{1}{2}} r_i \right\|_2,$$

using Cauchy-Schwarz inequality. Also

$$\left\| \sum_{i=2}^{n} \mu_i \alpha_i^t D^{\frac{1}{2}} r_i \right\|_2^2 \leq \alpha_2^{2t} \left\| p_0 \right\|_2^2 \leq \alpha_2^{2t}.$$

Consequently, $\nu(p_t, \pi) \leq \delta$ for $t(\delta) \geq \frac{2}{\lambda_2} \log(\frac{n}{\delta})$ using the inequality $(1 - x) \leq e^{-x}$ for $x \geq 0$. The characterization of λ_2 has been studied in [Sun and Das, 2019]. In particular, it was proved that if G is connected bipartite such that $G \neq K_{s,t}$ then $\lambda_2 \geq 1 + \frac{1}{\sqrt{st}}$. Since $\mathcal{A}$ is a valid candidate, we set $\tau_\delta \triangleq \frac{2}{1 + \frac{1}{\sqrt{st}}} \log(\frac{n}{\delta}) \geq t(\delta)$ which only depends on the size of the graph for δ fixed.

□

When $\delta = \frac{1}{4}$, the depth is equivalent to the mixing time [Levin and Peres, 2017].

2.3.4 Contextual Fingerprint

Besides identifying the notable structural attributes of activity graphs, it is crucial to consider the context in which they emerge. Different hosts exhibiting different behaviors may induce activity graphs with similar shapes, resulting in limited identification as collective information about the current state of the network is not exploited. In such cases, relying solely on the intrinsic feature is optimistic since two isomorphic graphs have the same intrinsic fingerprints. Thus, analyzing the activity of a host among group of hosts provides more insight about the prospective state of the network.

As an example, let us consider three hosts a, b and c. Fig. 2.3 presents a *merge* graph constructed by merging their corresponding activity graphs $\mathcal{A}_a$, $\mathcal{A}_b$ and $\mathcal{A}_c$ (depicted by the double circles) into a unified representation. While $\mathcal{A}_b$ and $\mathcal{A}_c$ are isomorphic, they correspond to different malicious hosts, a UDP-based port scanner and an IP address space scanner, respectively. On the other hand, $\mathcal{A}_a$ corresponds to a bot used to perform a Distributed Denial-of-Service (DDoS) attack while acting as a DNS and mail server (indicated by the SMTP source port identifier). Since the layers of activity graphs encode relationships between the hosts at several levels, considering pairs of nodes over a unified graph enables us to take into consideration the temporal nature of host group behaviors and understand complex attacks. For instance, a pair from layers l_0, l_1 enables us to track the host a (as depicted in red in Fig. 2.3) across the activity graphs of hosts which fall into the same temporal traffic window. In this scenario, both hosts b, c act as C&C channels for the botmaster a.
We thus devise in this section a *method* (see Algo. 3) *based on a set of merged activity graphs* where pairs of nodes provide latent collective representations according to a *variant of the Weisfeiler-Lehman isomorphism test*. We further equip it with an *efficient approximation calculation technique* (see Algo. 4).

Message passing models are currently the core element of the recent attention in the use of deep learning on graphs. [Morris et al., 2019, Xu et al., 2018] proved somewhat remarkably that graph neural networks (GNNs) [Wu et al., 2020], which are based on these models, are less expressive than the k-Weisfeiler-Lehman (WL) isomorphism tests hierarchy ($k \in \mathbb{N}^*$) [Grohe, 2017]. The popular 1-WL kernel [Shervashidze et al., 2011] can only support a measure of similarity based on the occurrences of subtree patterns [Shervashidze and Borgwardt, 2009, Bach, 2008] over each independent node in the graph, as a byproduct. Meanwhile, going for higher dimensions in the WL hierarchy is computationally expensive as k-WL runs in $\mathcal{O}(n^k)$ where n is the graph size [Arvind et al., 2020]. Thus, we leverage the 2-WL which offers an effective middle ground framework to reason collectively over pairs of nodes. For a review of 2-WL with its different neighborhood rules, we refer the reader to [Maron et al., 2019, Morris et al., 2017].

Our method differs from the standard 2-WL procedure in two important ways. First, 2-WL is typically executed over each graph g independently, then g is associated with a feature vector $\psi(g)$ where, for each σ in Σ, its coordinate $\psi_\sigma(g)$ counts the frequency of the color σ of the procedure in g. Instead, we operate over a single *merge graph* (see Def. 2 and Fig. 2.3) to capture the contextual interactions between activity graphs so that each frequency $\psi_\sigma(g)$ depends on all the graphs in $\mathcal{S}$ and not only g. Second, we consider a novel *adaptive neighborhood* policy which adapts the exploration based on a *node centrality* measure that we compute efficiently.

Definition 2. *(Merge graph) Let $\mathcal{S}$ be a family of activity graphs. Let s_u and t_u be two universal nodes acting as source and sink, respectively. The merge graph $\mathcal{M}_\mathcal{S}$ is unweighted undirected, cha-*

racterized by $V_{\mathcal{M}} = \bigcup_{g\in\mathcal{S}} V_g \cup \{s_u, t_u\}$ *and* $E_{\mathcal{M}} = \bigcup_{g\in\mathcal{S}} E_g \cup \{(s_u, v), (w, t_u), v \in l_0(g), w \in l_4(g)\}$.

Node centrality. The centrality measure that we consider comes from Lemma 2.3.2 which tells us that the eigenpairs, namely the eigenvectors, can be easily retrieved while executing Algo. 2 when computing the intrinsic fingerprints; therefore avoiding a separate centrality computation. Specifically, we associate with each node v its component (denoted by $f(v)$) in the eigenvector corresponding to the largest eigenvalue (which always equals 1) of $\widetilde{J}$. Note that $f(v) > 0$ from the Perron-Frobenius theorem [Levin and Peres, 2017]. The use of $f(v)$ as a centrality measure is motivated by the following observation:

$$f(v) = [\widetilde{J}f][v] = \sum_{u\in V_{\mathcal{A}}} \widetilde{j}_{vu} f(u) = \sum_{u\in\mathcal{N}(v)} \frac{1}{\sqrt{d_v d_u}} j_{vu} f(u). \tag{2.5}$$

Eq. (2.5) tells us that assigning $f(v)$ to v is equivalent to implicitly averaging over its neighbors according to their interaction intensity (dictated by the degree function d). Thus, $f(v)$ corresponds to an eigenvector centrality measure that considers that a node v with a neighbor u solely connected to it, provides more information that a same node v for which u is connected to many others at the same time.

Lemma 2.3.2. *From the SVD of* $N = U\Sigma V^T$*, the eigendecomposition of* $\widetilde{J}$ *can be retrieved by writing*

$$\widetilde{J} = \begin{bmatrix} 0 & N \\ N^T & 0 \end{bmatrix} = B \begin{bmatrix} \Sigma & 0 \\ 0 & -\Sigma \end{bmatrix} B^T \text{where } B = \begin{bmatrix} \widetilde{U} & \widetilde{U} \\ \widetilde{V} & -\widetilde{V} \end{bmatrix}$$

and $\widetilde{U} = \frac{U}{\sqrt{2}}, \widetilde{V} = \frac{V}{\sqrt{2}}$.

Adaptive neighborhood. As seen in Section 2.3.2, end-nodes in a pair of the activity graph play different roles depending on the layer they fall into. Thus, instead of treating both end-nodes in the same way, we define a new neighborhood rule where the neighborhood of a pair (u, v) is dynamic and based on the previous node centrality measure:

$$\mathcal{N}_{uv} = \{uw, w \in \mathcal{N}_v^*\} \cup \{wv, w \in \mathcal{N}_v^*\}, \tag{2.6}$$

where $\widetilde{f}(v) = \frac{f(v)}{\sum_{v\in V_{\mathcal{A}}} f(v)} \in (0, 1]$ and $\mathcal{N}_v^* = \mathcal{N}_v^1 \cup \mathcal{N}_v^2$ if $\widetilde{f}(v) < 0.5$, otherwise $\mathcal{N}_v^* = \mathcal{N}_v^1$. Intuitively, a low-central node needs to cover a larger portion of its neighborhood compared to a high-central node which is already connected to high-central nodes according to Eq. (2.5).

Approximation speedup. Since the propagation procedure of 2-WL runs in $\mathcal{O}(n^2)$, to make sure that `Nadege` is able to operate with high velocity data-streams, we propose an algorithm (see Algo. 4) which provably (see Theorem 2.3.5) approximates the exact features in Algo. 3.

In order to formulate concentration bounds and prove Theorem 2.3.5, we first derive an extension of the Hoeffding's inequality [Hoeffding, 1994] to the multivariate case by applying it over several dimensions.

Lemma 2.3.3. *(Hoeffding's inequality [Hoeffding, 1994])*
Let $X_1, ..., X_n$ *be independent random variables such that* $|X_i| \leq C$*, for each* $i = 1, ..., n$.
Set $\mu_X = \frac{1}{n}\sum_{i=1}^n X_i$*. Then, for any* $\varepsilon > 0$*, we have*

$$P\left(|\mu_X - \mathbb{E}[\mu_X]| \geq \varepsilon\right) \leq 2\exp\left(-\frac{n\varepsilon^2}{2C^2}\right).$$

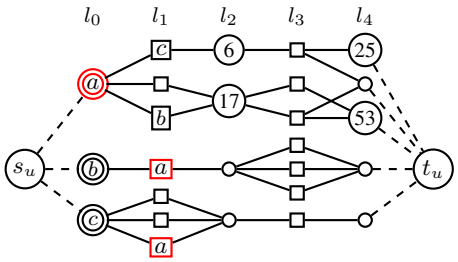


Figure 2.3 – Instance of a merge graph with hosts a, b and c.

Algorithm 3 Contextual Fingerprint

Input: A set $\mathcal{S}$ of activity graphs, θ an iteration parameter
Output: A set of features $c_{\mathcal{S}}$

Construct the merge graph $\mathcal{M}_{\mathcal{S}}$
Color each pair $p = (u, v)$ of $\mathcal{M}_{\mathcal{S}}$ per isomorphism type
parallel for $g \in \mathcal{S}$ **do**
  for all $r = 1, ..., \theta$ **do**
    Compute standard 2-WL over $\mathcal{M}_{\mathcal{S}}$ using Eq. (2.6)
    for all $\sigma \in \Sigma_r$ **do**
      $p^{in}_{\sigma} \leftarrow |\{(u, v), u \wedge v \in V_g^2 \text{ with color } \sigma\}|$
      $p^{out}_{\sigma} \leftarrow |\{(u, v), u \dot{\vee} v \in V_g^2 \text{ with color } \sigma\}|$
      $\psi_{r,\sigma}(g) \leftarrow (p^{in}, p^{out})^T$
    $\psi_r(g) \leftarrow (\psi_{r,\sigma_1}(g) \dots \psi_{r,\sigma_{|\Sigma_r|}}(g))$
  $c_{\mathcal{S}}[g] \leftarrow \oplus_r \text{vec}(\psi_r(g))$
return $c_{\mathcal{S}}$

Lemma 2.3.4. *(Multivariate Hoeffding's inequality)*
Let $\mathcal{Z}_1, ..., \mathcal{Z}_n$ be independent r-dimensional random variables such that $\|\mathcal{Z}_i\|_2 \leq C$, for each $i = 1, ..., n$. Set $\mu_{\mathcal{Z}} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{Z}_i$. Then, for any $\varepsilon > 0$, we have

$$P\left(\left\|\mu_{\mathcal{Z}} - \mathbb{E}[\mu_{\mathcal{Z}}]\right\|_2 \geq \varepsilon\right) \leq 2r \exp\left(-\frac{n\varepsilon^2}{2C^2 r}\right).$$

Proof.

For each $i = 1, ..., n$, let $\mathcal{Z}_{i,l}$ denote the l-th component of $\mathcal{Z}_i$ where $l = 1, ..., r$.
Let us fix i in $\{1, ..., n\}$. Since $\|\mathcal{Z}_i\|_2 \leq C$, then $|\mathcal{Z}_{i,l}| < C$. Hence, we can apply Hoeffding's inequality over each dimension l of $\mathcal{Z}_i$, which yields

$$P\left(|\mu_{\mathcal{Z}_l} - \mathbb{E}[\mu_{\mathcal{Z}}]_l| \geq \frac{\varepsilon}{\sqrt{r}}\right) \leq 2\exp\left(-\frac{n\varepsilon^2}{2C^2 r}\right), \tag{2.7}$$

where $\mu_{\mathcal{Z}l} = \frac{1}{n}\sum_{i=1}^{n} \mathcal{Z}_{i,l}$, for each $l = 1, ..., r$.
Suppose for each $l = 1, ..., r$ that $|\mu_{\mathcal{Z}_l} - \mathbb{E}[\mu_{\mathcal{Z}}]_l| < \frac{\varepsilon}{\sqrt{r}}$. Then it holds that

$$\left\|\mu_{\mathcal{Z}} - \mathbb{E}[\mu_{\mathcal{Z}}]\right\|_2 = \sqrt{\sum_{l=1}^{r} (\mu_{\mathcal{Z}_l} - \mathbb{E}[\mu_{\mathcal{Z}}]_l)^2} < \varepsilon.$$

Algorithm 4 Speedy Contextual Fingerprint

Input: A set $\mathcal{S}$ of activity graphs, θ a refinement parameter, ν confidence probability, ε error tolerance

Output: $\widetilde{c}_{\mathcal{S}}$

Construct the merge graph $\mathcal{M}_{\mathcal{S}}$
Populate $\Gamma^{\mathcal{S}}_{\nu,\varepsilon,\theta}$ according to (2.9) ▷ Uniform sampling
parallel for $p = (u, v) \in \Gamma^{\mathcal{S}}_{\nu,\varepsilon,\theta}$ **do**
    $\mathcal{H}_p \leftarrow \{g \in \mathcal{S} | u \in V_g \vee v \in V_g\}$ ▷ Involved hosts
    Build $\mathcal{G}_p$ the induced subgraph over $\mathcal{N}(p)$
    Color each node of $\mathcal{G}_p$ per isomorphism type
    for all $r = 1, ..., \theta$ **do**
        Determine $\sigma(p)$ over $\mathcal{G}_p$ ▷ Run diffusion process
        parallel for $g \in \mathcal{H}_p$ **do**
            $i \leftarrow \mathbb{1}_{|\mathcal{H}_p|=1}$
            $\widetilde{\psi}_{r,\sigma}(g)[i] \leftarrow \widetilde{\psi}_{r,\sigma}(g)[i] + \frac{1}{|\Gamma^{\mathcal{S}}_{\nu,\varepsilon,\theta}|}$
$\widetilde{c}_{\mathcal{S}}[g] \leftarrow \oplus_r \operatorname{vec}(\widetilde{\psi}_r(g))$
return $\widetilde{c}_{\mathcal{S}}$

Consequently, using Eq. (2.7)

$$
\begin{aligned}
&P\left(\left\|\mu_{\mathcal{Z}} - \mathbb{E}[\mu_{\mathcal{Z}}]\right\|_2 \geq \varepsilon\right) \\
&\leq P\left(\exists l \in \{1, ..., r\}, |\mu_{\mathcal{Z}_l} - \mathbb{E}[\mu_{\mathcal{Z}}]_l| \geq \frac{\varepsilon}{\sqrt{r}}\right) \\
&\leq 2r \exp\left(-\frac{n\varepsilon^2}{2C^2 r}\right).
\end{aligned}
$$

□

Theorem 2.3.5. *Let $\mathcal{S}$ be a set of activity graphs and $\mathcal{M}$ its associated merge graph. Let us fix r in $\{1, ..., \theta\}$, ε in $(0, 1]$ and $\nu > 0$. Then, there exists $\Gamma^{\mathcal{S}}_{\nu,\varepsilon,\theta}$ in $V^2_{\mathcal{M}}$ such that with probability ν, it holds that*

$$\forall g \in \mathcal{S} : \left\|\psi_r(g) - \widetilde{\psi}_r(g)\right\|_F \leq \varepsilon,$$

where $\|.\|_F$ denotes the Frobenius norm.

Proof.

Let g in $\mathcal{S}$. First, note that

$$\|\psi_r(g) - \widetilde{\psi}_r(g)\|^2_F = \sum_{\sigma \in \Sigma_r} \|\psi_{r,\sigma}(g) - \widetilde{\psi}_{r,\sigma}(g)\|^2_2.$$

Using Boole's inequality

$$
\begin{aligned}
&P(\bigcup_{g\in\mathcal{S}} \|\psi_r(g) - \widetilde{\psi}_r(g)\|_F \geq \varepsilon) \\
&\leq \sum_{g\in\mathcal{S}} P(\sum_{\sigma\in\Sigma_r} \|\psi_{r,\sigma}(g) - \widetilde{\psi}_{r,\sigma}(g)\|_2^2 \geq \varepsilon^2) \\
&\leq \sum_{g\in\mathcal{S}} \sum_{\sigma\in\Sigma_r} P(\|\psi_{r,\sigma}(g) - \widetilde{\psi}_{r,\sigma}(g)\|_2 \geq \frac{\varepsilon}{\sqrt{|\Sigma_r|}}).
\end{aligned}
$$

For clarity of notation, we denote $p_\sigma^{in}, p_\sigma^{out}$ by p_σ^1, p_σ^2 and let P_σ^i be the set of σ-colored pairs of type i with $p_\sigma^i = |P_\sigma^i|$. We set $\gamma_g^i = \sum_\sigma p_\sigma^i$ and $\gamma_{\mathcal{M}} = \binom{V_{\mathcal{M}}}{2}$. Let $\mathcal{Z}_{r,\sigma}(g) = (z_1, z_2)^T$ be a 2-dimensional random variable such that

$$
z_i = \alpha_i \mathbb{1}_{P_\sigma^i}, \tag{2.8}
$$

where $\alpha_i = \gamma_g^i/\gamma_{\mathcal{M}}$ and $\mathbb{1}_{\mathcal{E}}$ is 1 if at round r of Algo. 3, we sample p such that p is in $\mathcal{E}$, otherwise 0.

Let $\mu_{\mathcal{Z}_{r,\sigma}}(g) := \frac{1}{|\Gamma|}\sum_{r=1}^{|\Gamma|} \mathcal{Z}_{r,\sigma}(g) = \widetilde{\psi}_{r,\sigma}(g)$. Since for each i, $\mathbb{E}[z_i] = \frac{\alpha_i p_\sigma^i}{\gamma_g^i}$, then $\mathbb{E}[\mu_{\mathcal{Z}_{r,\sigma}}(g)] = \psi_{r,\sigma}(g)$. Note that $\left\|\mathcal{Z}_{r,\sigma}(g)\right\|_2 \leq \sqrt{2}$. By Lemma 2.3.4, we have

$$
P(\|\psi_{r,\sigma}(g) - \widetilde{\psi}_{r,\sigma}(g)\|_2 \geq \frac{\varepsilon}{\sqrt{|\Sigma_r|}}) \leq 4\exp\left(-\frac{\Gamma\varepsilon^2}{8|\Sigma_r|}\right).
$$

Therefore, by setting

$$
\Gamma := \Gamma_{\nu,\varepsilon,\theta}^{\mathcal{S}} = \frac{8|\Sigma_\theta|}{\varepsilon^2}\log\left(\frac{4|\mathcal{S}||\Sigma_\theta|}{1-\nu}\right), \tag{2.9}
$$

where Σ_θ is an upper bound on the maximum number of different colors, we obtain

$$
P(\bigcup_{g\in\mathcal{S}} \|\psi_r(g) - \widetilde{\psi}_r(g)\|_F \geq \varepsilon) \leq 1-\nu.
$$

□

2.3.5 Fingerprint Graph Kernel

Let $\mathcal{S}$ be a set of activity graphs. Each g in $\mathcal{S}$ is represented by its fingerprint set $r_g = \{(i_v, c_g)\}_{v\in V_g}$ in a real feature space denoted Ω. Let k be a kernel on Ω such that for (g,h) in $\mathcal{S}^2$:

$$
k[(i_v, c_g), (i_u, c_h)] = \left(\frac{\|i_v - i_u\|^2}{2\sigma^2}\right)\exp\left(\frac{-\|c_g - c_h\|^2}{2\sigma^2}\right), \tag{2.10}
$$

with bandwidth $\sigma > 0$. Let ϕ and $\mathcal{H}$ be the feature map and the reproducing kernel Hilbert space (RKHS) associated with k; whose existence is derived from Moore–Aronszajn theorem. We view r_g as a sample following a distribution denoted by $\mathcal{D}_g$ that we embed into a mean map $\mu_{\mathcal{D}_g}$ in $\mathcal{H}$ by kernel mean embedding (KME), as it suffers less from the curse of dimensionality compared to density estimation methods [Muandet et al., 2016]:

$$
\mu_{\mathcal{D}_g} := \mathbb{E}_{X\sim\mathcal{D}_g}[k(X,.)] = \mathbb{E}_{X\sim\mathcal{D}_g}[\phi(X)] = \int_\Omega \phi(x)d\mathcal{D}_g(x).
$$

We choose Gaussian kernels in Eq. (2.10) as they belong to the kernel class of radial basis functions (RBFs) which enjoy nice properties [Szabó and Sriperumbudur, 2017]. In particular, their universality property implies that KME retains all information, namely moments, about the distribution $\mathcal{D}_g$. Further details on the properties of the RKHS and different classes of kernel functions can be found in [Szabó and Sriperumbudur, 2017, Muandet et al., 2016].
Since access to the true distribution $\mathcal{D}_g$ is lacking, we use the empirical mean estimator $\widehat{\mu}_{D_g} = \frac{1}{n_g}\sum_{v\in V_g}\phi(i_v, c_g)$, by treating the data as a probability mass distribution associated with r_g, i.e., $\mathcal{D}_g = \frac{1}{n_g}\sum_v \delta_{(r_v,c_g)}$ with δ_x the Dirac measure defined for x in Ω. We note that $\widehat{\mu}_{\mathcal{D}_g}$ is a good proxy for $\mu_{\mathcal{D}_g}$ as it is is unbiased and consistent [Smola et al., 2007], with convergence rate $\mathcal{O}(\mathcal{R}_{n_g}(\mathcal{H}) + \frac{1}{\sqrt{n_g}})$ where $\mathcal{R}(\mathcal{H})$ denotes the Rademacher complexity of $\mathcal{H}$.
Thereby, we define the *valid* (see Proposition 4) fingerprint graph kernel K (denoted by FGK) as follows

$$K(g,h) = \langle \widetilde{\mu}_{\mathcal{D}_g}, \widetilde{\mu}_{\mathcal{D}_h}\rangle_{\mathcal{H}} = \frac{1}{n_g n_h}\sum_{v,u} k[(i_v, c_g), (i_u, c_h)].$$

Proposition 4. *The fingerprint graph kernel K is symmetric positive definite.*

Proof.

Let g and h be two activity graphs. Then

$$K(g,h) = \frac{1}{2n_g n_h \sigma^2}\sum_{v,u} f_1(v,u) f_2(g,h),,$$

where $f_1(v,u) = \|i_v - i_u\|^2$ and $f_2(g,h) = \exp\left(\frac{-\|c_g - c_h\|^2}{2\sigma^2}\right)$.

Observe that f_1 is clearly positive definite, while the positive definiteness of f_2 is obtained from Corollary 3 in [Schoenberg, 1938]. Since the sum and multiplication of positive definite kernels are still positive definite, we conclude that K is positive definite. Moreover, it is clearly symmetric.

□

Computing K takes $\mathcal{O}(|\mathcal{S}|^2 n^2)$ (we assume for simplicity that all graphs have the same size n). Since $\mathcal{S}$ could be potentially large, we use an approximate kernel transformation known as Fast-Food [Le et al., 2014]. For each fingerprint x in $\mathbb{R}^{\mathrm{d}}$, the latter provides a randomized explicit feature representation $\widetilde{\phi}(x)$ in $\mathbb{R}^p$ (with $p < d$), running in $\mathcal{O}(d \log p)$ while ensuring that $\langle \widetilde{\phi}(x_i), \widetilde{\phi}(x_j)\rangle$ converges to $\langle \phi(x_i), \phi(x_j)\rangle$ with a rate of $\mathcal{O}(\frac{\log(\frac{2}{\delta})}{\sqrt{d}})$ where δ is a failure probability.

2.4 Experimental evaluation

In this section, we provide a broader evaluation assessment of `Nadege` in terms of its detection and runtime performance. We open by describing the evaluation setup, followed by the learning models, and finally, close by presenting our results.

2.4.1 Evaluation setup

Datasets. We conducted experiments on six widely-used and recent datasets covering different types of network environments. Detailed descriptions of these traces can be found in [Hindy et al.,

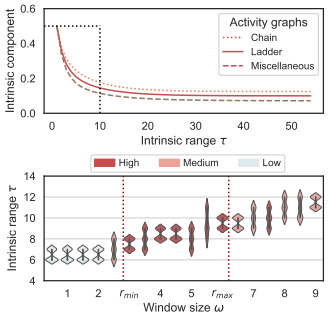


Figure 2.4 – Parameter sensitivity analysis.

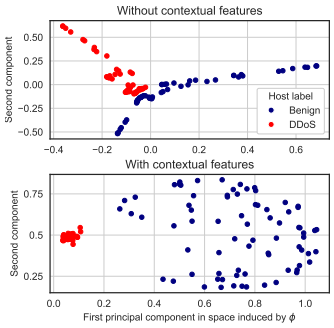


Figure 2.5 – Kernel-PCA projection of FGK.

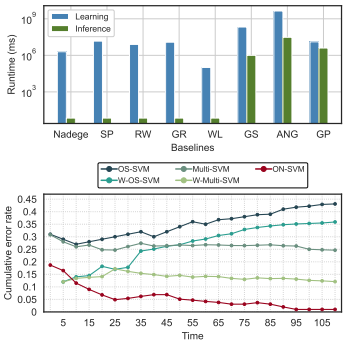


Figure 2.6 – Runtime and cumulative error rates.

2020]. We provide some relevant summary statistics in Table 2.3. All traces labeled the network anomaly hosts, which means we have ground truth of the traffic. Moreover, they vary in volumes and forms of anomalies mixing both high and low-intensity attacks. Thus, overall these traces offer a good diversity for evaluating `Nadege` robustness across different real-word situations.

Baselines. We compared *Nadge* with two categories of state-of-the-art approaches. *Graph kernel-based* methods include the Shortest-Path kernel (SP) [Borgwardt and Kriegel, 2005], the Random-Walk kernel (RW) [Vishwanathan et al., 2010], the Graphlet kernel (GR) [Shervashidze et al., 2009], and the Weisfeiler-Lehman subtree kernel (WL) [Shervashidze et al., 2011]. We also picked GraphSAGE (GS) [Hamilton et al., 2017a], a representative inductive *learning-based* method which fit into the graph neural network framework proposed by [Gilmer et al., 2017]. In particular, these methods serve also at assessing the performance of the Fingerprint kernel. The second category comprises *anomaly-based* methods, namely AnoNG (ANG) [Yao et al., 2019] and Graphprints (GP) [Harshaw et al., 2016] (described in Section 2.2).

2.4.2 Learning

Metrics. The detection performance of an algorithm, on a particular trace, can be measured in terms of its true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN). In our evaluations, we consider for each approach three metrics: precision ($\mathcal{P}=\frac{TP}{TP+FP}$), recall ($\mathcal{R}=\frac{TP}{TP+FN}$), and F1-score ($\mathcal{F}=\frac{2\times\mathcal{P}\times\mathcal{R}}{\mathcal{P}+\mathcal{R}}$). In network intrusion detection, it is important to obtain a minimal number of false alarms since it is time consuming and expensive for an analyst to investigate each alarm.

Sliding window principle. We monitor the flow traffic of each trace using overlapping sequential observation windows to mimic a sliding time window model. Each window gives rise to a set $\mathcal{S}$ of labeled (benign or anomalous) activity graphs which are characterized by a similarity matrix $K_{\mathcal{S}}$ resulting from the fingerprint graph kernel. We then feed $K_{\mathcal{S}}$ into a kernelized Support Vector Machine (SVM) learning model, while considering both offline and online execution modes.

Models. As a state-of-the-art kernel method, SVM classifiers [Boser et al., 1992, Cortes and Vapnik, 1995] find an optimal separating hyperplane specified by a subset of the training data points, known as support vectors (SVs). Their success comes from a statistical learning theory viewpoint

where they operate under the principle of risk minimization, providing theoretical guarantees on the generalization error. For further details on kernel methods, we refer the reader to [Hofmann et al., 2008]. We consider several learning models which all support class-weighting which is designed to deal with class imbalance:
– *Offline models.* We consider four models: OS-SVM (C_1), Multi-SVM (C_2), W-OS-SVM (C_3), W-Multi-SVM (C_4) where OS stands for One Shot and W for Wide. For OS-SVM, the model is trained only once on the first batch of windows. For Multi-SVM, the model is retrained every day by using, however, only samples from the previous day. W-OS-SVM mimics OS-SVM and W-Multi-SVM mimics Multi-SVM, however, the batch size for training and retraining covers several batches instead of one.
– *Online models.* We consider a popular incremental version of SVM, denoted by ON-SVM (C_5), which processes data on the fly and modifies its hyperplane, if necessary, as new training samples arrive [Bordes et al., 2005]. Moreover, it introduces a SV removal step and supports active selection.

Settings. In all our experiments, we followed common practices of performance evaluation by repeating each 10-fold internal cross validation. Within each fold, we selected the trade-off C of SVM from $\{10^{-3}, ..., 10^{3}\}$ and the depth θ from $\{0, \ldots, 7\}$. We set the parameters δ and ε to 0.1 and 0.05 respectively, ν to 0.95, and σ in the FGK to $1/d_{att}$ with d_{att} the attribute dimension [Feragen et al., 2013]. Instead of using random folds for the validation of the approach, we preserved the temporal order of the network data. Indeed, it is essential for measuring the performance of intrusion detection systems that the training and test data are temporal disjunct and future attacks are not available during learning to avoid experimental bias [Pendlebury et al., 2019].
Moreover, we made use of the GraKel library [Siglidis et al., 2020] for graph kernel implementations and normalized all kernel matrices. We note that GraKel was entirely developed in Python, is compatible with scikit-learn, and exploits the Cython extension to benefit from a fast implementation in C.

2.4.3 Results and Discussions

We first discuss the strengths of our fingerprints, then, we evaluate the detection performance of `Nadege`.

Intrinsic fingerprint. We start by a sensitivity analysis over the intrinsic range τ which captures the depth of the closed walks explorations. Fig. 2.4 shows the maximum range τ_δ across all datasets per window size ω. We note the existence of an optimal region $[r_{min}, r_{max}]$ where the detection for `Nadege` is always high (i.e., F-score > 0.9 on average, across all datasets). When $\omega < r_{min}$, features are under-representative since graphs are mostly chains and very sparse trees and, when $\omega > r_{max}$, the SVM decision problem becomes coarser. As the time step goes to infinity, the intrinsic components will not change much, converging to the stationary probability. Hence, we gain little new information from the intrinsic fingerprint by increasing τ (see Fig. 2.5).

Contextual fingerprint. In Table. 2.2, we report the average F-score across all datasets along with the node utilization (defined as the *normalized* average value of $\sum_{p\in\Gamma} |\mathcal{N}_p|$ across merge graphs, with Γ the sample size of Algo. 4), which captures the intensity of exploration. We compare our centrality-based neighborhood rule with three others, namely 2-WL, 2-FWL (Folklore) and 2-LWL (Local) [Maron et al., 2019, Morris et al., 2017]. We observe that centrality-based explorations offer a nice trade-off between contextual informativeness and the size of the exploration

Method	Policy	Score	Utilization
2-WL	Global	0.8941	49.7%
2-FWL	Folklore	0.9157	49.7%
2-LWL	Local	0.8840	23.8%
`Nadege`	Centrality	0.9716	25.0%

TABLE 2.2 – Exploration strategies.

space as they adapt their strategy according to the current topology of the merge graph. In Fig 2.3, we performed a Kernel PCA [Schölkopf et al., 1998] projection to the hosts in D-1 based on the FGK. We observe that while the intrinsic features provide an already strong discrimination against most attacks, adding the contextual features improves the classification performance of the FGK in discriminating some collaborative attacks such as Botnets and DDos.

Learning models. In Fig. 2.6, we report over time, the ratio of misclassifications over the number of classified graphs so far of the SVM models for D-1. C_3 and C_4 surpassing C_1 and C_2 respectively, reveals the necessity of frequently updating models over time to mitigate concept drift. C_3 and C_4 surpassing both C_1 and C_2 indicates the importance of training on more data. We obtained similar error curves on the other datasets where only C_4, C_5 yielded low generalization errors, demonstrating the effectiveness of `Nadege` in both offline and online execution modes.

Detection performance. In Table 2.3, we observe that `Nadege` significantly outperforms all baseline methods in six datasets, while producing the lowest false alarm rates. In addition, `Nadege` yielded the most stable variances (i.e., gaps between maximum and minimum) among all methods. We report the good performance of AnoNG for low frequency attacks at the expense of, however, high memory and time costs (see Fig 2.6). Even though Graphprints and the Graphlet kernel produce similar features, the former performs better mainly due to the robust MCD [Rousseeuw and Driessen, 1999] removal procedure of outliers. The Fingerprint kernel is adapted for network discrimination against anomalies as demonstrated by its superiority over the kernel-based methods. It is worth noting that WL and GraphSAGE exhibit close average scores as the latter can be seen as a continuous approximation to the WL test [Hamilton et al., 2017a]. Overall, these observations enable us to advocate `Nadege` as an accurate, fast, and versatile learning framework.

Trace	Approach	Precision	Recall	F1-Score
D-1	**Nadege**	**0.9991**	**0.9987**	**0.9989**
Name: CSE-CIC-IDS-18	*Shortest-Path*	0.6740	0.7290	0.7290
#Total: 16,232,943	*Random-Walk*	0.6706	0.7174	0.6932
Benign: 82%	*Graphlet*	0.9724	0.9423	0.9571
Attacks: 17%	*WL-Subtree*	0.9905	0.9372	0.9631
> Brute force, Dos, DDoS	*GraphSAGE*	0.9962	0.9827	0.9894
PortScan, Web, Infiltration	*AnoNG*	0.9799	0.9960	0.9879
	Graphprints	0.9679	0.9631	0.9679
D-2	**Nadege**	**0.8184**	**0.9842**	**0.8937**
Name: UGR'16	*Shortest-Path*	0.6793	0.6187	0.6476
#Total: 12,517,654	*Random-Walk*	0.5275	0.6392	0.5780
Benign: 64%	*Graphlet*	0.8265	0.7392	0.7804
Attacks: 36%	*WL-Subtree*	0.8739	0.8283	0.8505
> Botnet, Blacklist, Dos,	*GraphSAGE*	0.7776	0.7329	0.7546
Scan, Spam	*AnoNG*	0.8591	0.9021	0.8801
	Graphprints	0.7973	0.8590	0.7973
D-3	**Nadege**	**0.9987**	**0.9985**	**0.9986**
Name: Bot-IoT	*Shortest-Path*	0.8134	0.7593	0.7854
#Total: 3,668,522	*Random-Walk*	0.5168	0.5789	0.5461
Benign: 42%	*Graphlet*	0.6372	0.6719	0.7789
Attacks: 58%	*WL-Subtree*	0.8095	0.8480	0.8283
> Scan, Dos, DDos, Theft	*GraphSAGE*	0.8612	0.7180	0.7831
	AnoNG	0.8612	0.7180	0.9870
	Graphprints	0.6635	0.7370	0.7402
D-4	**Nadege**	**0.9523**	**0.9892**	**0.9704**
Name: N-BaIoT	*Shortest-Path*	0.6537	0.9892	0.7872
#Total: 5,256,390	*Random-Walk*	0.4567	0.8820	0.6018
Benign: 67%	*Graphlet*	0.7137	0.8196	0.7630
Attacks: 33%	*WL-Subtree*	0.8272	0.8741	0.8500
> Botnets (Mirai & Bashlite)	*GraphSAGE*	0.8984	0.8120	0.8530
	AnoNG	0.9736	0.9548	0.9641
	Graphprints	0.6394	0.8548	0.7316
D-5	**Nadege**	**0.9780**	**0.9941**	**0.9860**
Name: Tor-nonTor	*Shortest-Path*	0.7347	0.6492	0.6893
#Total: 67,834	*Random-Walk*	0.7283	0.7239	0.7261
Benign: 78%	*Graphlet*	0.8074	0.7032	0.7517
Anomalies: 22%	*WL-Subtree*	0.8552	0.9290	0.8906
	GraphSAGE	0.9791	0.8410	0.9048
	AnoNG	0.8903	0.9300	0.8903
	Graphprints	0.7526	0.8320	0.7903
D-6	**Nadege**	**0.9757**	**0.9890**	**0.9823**
Name: VPN-nonVPN	*Shortest-Path*	0.5530	0.7930	0.6516
#Total: 125,634	*Random-Walk*	0.7012	0.7153	0.7082
Benign: 87%	*Graphlet*	0.6887	0.6136	0.6490
Anomalies: 13%	*WL-Subtree*	0.8109	0.8693	0.8391
	GraphSAGE	0.9853	0.9382	0.9382
	AnoNG	0.9085	0.8039	0.8530
	Graphprints	0.9332	0.8790	0.9053

TABLE 2.3 – Statistics and evaluation results on 6 traces.

2.5 Conclusion and Future Work

In this chapter, we proposed `Nadege`, a new framework for anomaly detection and traffic classification in networks. In particular, we developed a graph kernel tailored for networking, based on random walks and the Weisfeiler-Lehman hierarchy of isomorphism tests. The advantages of the kernel lie on its effectiveness in capturing rich local and global information on the host profile activity, while incorporating context awareness. The framework was made fast and scalable by using special purpose graphs along with an approximation algorithm for which we derived some theoretical guarantees. In our evaluation using a variety of traces from heterogeneous environments, `Nadege` achieved high F1-scores, significantly outperforming existing state-of-the-art approaches. An interesting future work is assessing the performance of `Nadege` in the context of adversarial attacks and within next generation networks (SDN and NFV).

Majority Dynamics

CHAPTER 3

Biased Opinion Dynamics

Contents

3.1 Introduction

In everyday life, when sharing or forming an opinion about a set of issues of interest, individuals often consult with their friends, relatives, acquaintances, or others, in their close social group. Furthermore, with the widespread use of online social networks, social influence comes to play a prominent role in several phenomena such as the diffusion of technological innovations, the rise of political movements, and the intensification of fears during outbreaks. Consequently, there has been a growing interest in understanding the opinion-forming processes that drive the formation of consensus and opinion clustering in social systems.

Opinion dynamics are mathematical models that enable to investigate how a group of agents change their beliefs under the influence of other agents. While various models considered in the literature confer the same *intrinsic* value to all opinions [Coates et al., 2018], an agent may be biased towards a "preferred" opinion ; for instance, reflecting intrinsic superiority of one alternative (e.g., a technological innovation) over the status quo. We represent a multi-agent network by a graph made up of n agents that are modeled as nodes, and an edge between two nodes corresponds to a relation between the respective agents such as friendship, common interests, or advice. We focus on the scenario where each agent must choose between two alternatives by exhibiting a bias toward one of the opinions. In the remainder, we use labels 0 and 1 for the two opinions and we assume 1 is the *superior opinion*.

Starting from an initial state in which all agents share opinion 0, the system evolves in rounds. In each round, one agent is selected uniformly at random. With some probability α (called *bias*), the agent adopts 1, while with probability $1-\alpha$, the agent adopts to the majority opinion on the basis of those held by its neighbors in the underlying network. When $\alpha > 0$ the process always converge to global adoption of the opinion 1. Since dynamics are aimed at modeling the spread of opinions, an important issue is to determine how fast the superior opinion takes over the network [Mossel and Tamuz, 2017]. In [Anagnostopoulos et al., 2020], the authors show that under the *linear* voter rule, where agents copy the opinion of a randomly selected neighbor, consensus is reached quickly within $\mathcal{O}(\frac{1}{\alpha} n \log n)$ rounds regardless of the underlying topology. In contrast, under the *non-linear* majority rule where agents update their opinion to the majority opinion in their neighborhood, it turns out that the convergence time is super-polynomial in expectation whenever the network is dense (i.e., when the minimum degree is $\omega(\log n)$ [*]).

One might wonder if the converse occurs, namely, whether the biased majority dynamics always affords (expected) polynomial convergence to the absorbing state when the network is not dense. While this is indeed the case for cycles, trees, and disconnected cliques of size $\mathcal{O}(\log n)$, understanding the behavior of the dynamics remains open for bounded degree topologies, inducing challenging open problems formulated in [Anagnostopoulos et al., 2020, Cruciani et al., 2021].

In this work, we aim at contributing to the general understanding of the evolution of biased opinion dynamics under the non-linear majority rule by studying their behavior theoretically and empirically. We make the following contributions:

— We show a polynomial time convergence for new classes of topologies (namely, cubic graphs) and characterize them in terms of *stable* structures.

— We provide a threshold value on the bias α, above which the expected time for consensus on the superior opinion is polynomial for random regular graphs.

∗. The notation ω is defined as $f(n) = \omega(g(n))$ iff $\forall k > 0, \exists n_0, \forall n > n_0 \colon |f(n)| > k \cdot g(n)$. Hence, $f(n) = \omega(g(n))$ is equivalent to $g(n) = o(f(n))$.

— We answer negatively to the open problem in [Anagnostopoulos et al., 2020] by exhibiting classes of network topologies (namely, random Δ-regular graphs with $\Delta \geq 5$) for which we prove that the expected time for consensus on the superior opinion is exponential for small values of α.

— We provide insights into the dynamical properties of network structures that are implicitly responsible for the dichotomy between the slow and fast consensus behavior, in light of a generalized notion of domination in graphs. To the best of our knowledge, this is the first work on biased opinion dynamics that characterizes consensus in terms of multiple domination.

— Finally, we support our theoretical findings by consistent experiments, relating the speed of consensus with properties of the network structures.

The rest of this chapter is organized as follows. In Section 3.2, we review the related works. In Section 3.3, we formally describe the biased opinion dynamics under the non-linear majority rule. In Section 3.4, we present an extension of standard domination in graphs and leverage it to analyze the expected time to reach consensus for random regular graphs in Section 3.6 and Section 3.7. Then we validate our theoretical results through experiments and discuss the obtained results in Section 3.8. Finally, we draw our conclusions in Section 3.10.

3.2 Related Work

The problem we consider lies at the intersection of several areas for which there is a vast amount of existing literature. In what follows, we discuss contributions that most closely relate to the topics of this work.

Opinion diffusion and consensus. A substantial line of research has been devoted to the study of opinion dynamics, mostly motivated by phenomena that arise from social sciences, to physics and biology. Some recent contributions analyzed the spread of opinion formation in social influence [Out and N. Zehmakan, 2021, Zehmakan, 2021]. For a more detailed survey on opinion dynamics in multi-agent systems, we refer the reader to [Coates et al., 2018, Becchetti et al., 2020a].

In this chapter, we study the non-linear majority rule which originates from the study of agreement phenomena in spin systems [Krapivsky and Redner, 2003]. It has lately received renewed attention, mostly around the investigation of the time and conditions that cause agents to reach consensus.

Consensus and biased majority. Some forms of bias have been considered in the literature. In [Mukhopadhyay et al., 2020], each agent updates each of its opinions at points of different independent Poisson point processes, which introduces a bias towards the opinion with the lowest firing rate frequency. The works closest to ours are [Anagnostopoulos et al., 2020, Cruciani et al., 2021].

In [Anagnostopoulos et al., 2020], the speed of convergence under the majority rule is shown to be affected by the underlying topology, namely, is superpolynomial for dense networks. The synchronous setting has been considered in [Cruciani et al., 2021] with qualitatively consistent findings, albeit under a different model where agents sample k neighbors uniformly at random with replacement and update their state to the most frequent state among those in the sample. Yet, these results only apply to very dense networks with minimum degree $\omega(n)$.

We show that the expected convergence time can be exponential even in sparse networks, suggesting a more complicated dependence of the convergence time on the degree distribution. Our overall approach is different since it characterizes the majority opinion formation in terms of multiple domination in graphs, providing an interplay between bias, consensus and network structure.

Majority and graph domination. Network structure plays a crucial rule in opinion diffusion under several models [Donnelly and Welsh, 1983,Hassin and Peleg, 1999,Cooper et al., 2013,Morris, 2000]. [Auletta et al., 2015] showed that initial majority can be subverted for all but some topologies, including cliques and quasi-cliques. Moreover, while there exist always an initial opinion distribution, such that the final majority will reflect the initial one, regardless of the topology, computing an initial opinion configuration that will subvert an initial majority is topology-dependent and $\mathcal{NP}$-hard in general [Auletta et al., 2018].

Dominating sets can be useful to reach all the nodes efficiently in the network. The books [Haynes et al., 2013, Haynes, 2017] supply a comprehensive introduction to theoretical and applied facets of domination in graphs.

3.3 Majority Dynamics

In this section, we define the terminology used throughout the chapter. Then, we describe the Majority Dynamics.

Notation and Preliminaries. We model the multi-agent network by an undirected graph $G = (V, E)$ with $|V| = n$ nodes, each node $v \in V$ representing an agent. The system evolves in discrete time steps and, at any given time $t \in \mathbb{N}$, each node v holds an *opinion* $x_v^{(t)} \in \{0, 1\}$ (see Definition 3.3.1). We denote by $X_t = (x_1^{(t)}, \ldots, x_n^{(t)})$ the corresponding *state* of the system at time t. For each $v \in V$, we denote the open neighborhood of v with $\Gamma(v) := \{u \in V : \{u, v\} \in E\}$ and the degree of v with $d(v) := |\Gamma(v)|$. For any $X \subseteq V$, let $G[X]$ denote the subgraph of G induced by X. Finally, for a family of events $\{\mathcal{E}_n\}_{n\in\mathbb{N}}$ we say that $\mathcal{E}_n$ occurs *with high probability* (*w.h.p*, for short) if a constant $k > 0$ exists such that $\mathbf{P}(\mathcal{E}_n) = 1 - \mathcal{O}(n^{-k})$, for every sufficiently large n.

Definition 3.3.1. For every $t > 0$, a node v is said to be **active** if $x_v^{(t)}$ is 1. Otherwise, it is **idle**. Moreover, we say that a subset S of V is active if every node in S is active.

M-Dynamics. We study the random process $\{X_t\}_{t\in\mathbb{N}}$ defined on G as follows: starting from the initial state $X_0 = (0, \ldots, 0)$, in each round t, every node $v \in A_t$ updates its value according to the non-linear rule:

$$x_v^{(t)} = \begin{cases} 1 & \text{with probability } \alpha, \\ M_G(v, X_{t-1}) & \text{with probability } 1 - \alpha, \end{cases}$$

where $A_t \subseteq V$ is the set of nodes that update their opinion, $\alpha \in (0, 1]$ (modeling the *bias*) is the probability to transition to the superior opinion, and $M_G(v, X_{t-1})$ is the value held in configuration X_{t-1} by the majority of the neighbors of node v in G:

$$M_G(v, X_{t-1}) = \begin{cases} 1 & \text{if } \sum_{w\in\Gamma(v)} x_w^{(t-1)} > \frac{\Gamma(v)}{2}, \\ 0 & \text{if } \sum_{w\in\Gamma(v)} x_w^{(t-1)} < \frac{\Gamma(v)}{2}, \end{cases}$$

and ties are broken uniformly at random, that is, if $\sum_{w\in\Gamma(v)} x_w^{(t-1)} = \frac{\Gamma(v)}{2}$ then $M_G(v, X_{t-1}) = 0$ or 1 with probability $1/2$.

Stabilization. The M-Dynamics is a discrete-time Markov chain with a very large state space of size 2^n and has $\mathbf{1} = (1, \ldots, 1)^\top$ as the only absorbing state. This implies that, since the graphs are finite, such an absorbing state will be reached in finite time with probability 1. We use $\tau_\alpha(G)$ to denote the *stabilization time*, which is the number of rounds for the process to reach the absorbing state $\mathbf{1}$:

$$\tau_\alpha(G) = \inf\left\{t \in \mathbb{N}, \forall v \in V : x_v^{(t)} = 1\right\}.$$

Models. We distinguish between two main models of the M-dynamics, which differ according to the choice of A_t:

- *Asynchronous (Async).* In each round t, some agent v_t, chosen randomly, updates its opinion, $A_t = \{v_t\}$.
- *Synchronous (Sync).* In each round t, all agents update their opinion concurrently, that is, $\mathcal{A}_t = V$.

Our results and proof techniques are similar between the Async and Sync models. Therefore, for the sake of presentation, we will focus in this chapter on the Async setting and defer discussions of the Sync model to Section 3.9.

3.4 Decreasing Structures and k-domination

In this section, we relate the notion of domination to the M-Dynamics and introduce the concept of *decreasing* sets.

3.4.1 Multiple Domination

We present a generalized notion of domination in graphs.

Definition 3.4.1. (k-domination) Let $S \subseteq V$ and $k \in \{1, \ldots, n\}$.

- We say that a vertex v is k-dominated by S (equivalently, S k-dominates v) if $|\Gamma(v) \cap S| \geq k$. We denote by $D_k(S)$ the set of all nodes k-dominated by S.
- Let $U \subseteq V$. U is said to be k-dominated by S if S k-dominates all vertices $u \in U$, that is, if $U \subseteq D_k(S)$.

When $k = 1$, a 1-dominating set S of smallest size such that $D_1(S) = n$ is called a minimum dominating set and its size is known as the domination number, denoted by $\gamma(n)$. The problem of determining $\gamma(n)$ is one of the core $\mathcal{NP}$-complete optimization problems in graph theory and remains $\mathcal{NP}$-complete even for planar graphs of maximum degree 3 [Garey and Johnson, 1979a]. In the following, we focus on the case of majority domination.

Definition 3.4.2. (M-domination) Let $S \subseteq V$. We say that a vertex v is M-dominated by S (equivalently, S M-dominates v) if v is k-dominated by S with $k = \left\lfloor \frac{\mathcal{N}(v)}{2} \right\rfloor + 1$. We denote by $D_M(S)$ the set of all nodes M-dominated by S.

3.4.2 Complexity analysis of M-domination

The decision problem of finding a set S of size s such that $|D_M(S)| = l$ for arbitrary non-negative integers s, l is $\mathcal{NP}$-hard. In order to prove this result, we first introduce the following auxiliary function.

Definition 3.4.3. Let $k \in \{1, ..., n\}$. For every s in $\{1, ..., n\}$, we define the function $\phi_k : \{1, ..., n\} \to \{1, ..., n\}$ as

$$\begin{aligned}\phi_k(s) &= \max_{U \subseteq V}\{|U|, \exists S \subseteq V, |S| = s, S \text{ k-dominates } U\},\\ &= \max_{S \subseteq V, |S|=s} |D_k(S)|.\end{aligned}$$

Moreover, we denote ϕ_k by ϕ_M when $k = \left\lfloor \frac{\mathcal{N}(v)}{2} \right\rfloor + 1$.

It corresponds to the size of the largest set k-dominated by a set of size s among the $\binom{n}{s}$ candidates. Note that $\phi_k(s) = 0$ for every $s < k$ and $\phi_k(n) = n$.

We characterize the complexity of the *M-dominating set problem* which decision problem is as follows.

Instance : A graph G, non-negative integers s, l.

Question : Is there a set S of size s such that $|D_M(S)| = l$?

Theorem 3.4.1. *The M-dominating set problem is $\mathcal{NP}$-hard.*

Proof.

We consider the following decision problem corresponding to the standard *dominating set problem*:

Instance : A graph G and a non-negative integer s.

Question : Is there a dominating set of size at most s?

Our aim is to exhibit a polynomial-time reduction from the dominating set problem (denoted by DS), which is known be $\mathcal{NP}$-complete (see [Garey and Johnson, 1979a]), to the M-dominating set problem (denoted by $\mathcal{MD}$). We note in particular, that DS is $\mathcal{NP}$-complete in cubic graphs (see [Alimonti and Kann, 2000]). Given a cubic graph $G = (V, E)$, we build an auxiliary graph $H = (V_H, E_H)$ such that $V_H = V \cup U$ and $E_H = E \cup E_U$ where

$$\begin{cases} U = \{s_i, i \in [2]\}, \\ E_U = \{u.v, \forall u \in U, \forall v \in V\}. \end{cases}$$

Let $\gamma(G)$ be the size of the minimum dominating set of G. Observe that

$$\begin{cases} \phi(s) = n & \text{if } \gamma(G) \leq s - 2, \\ \phi(s) < n & \text{otherwise.} \end{cases}$$

Thus, checking the existence of a set of size s such that $\phi_M(s) = n$ in H is equivalent to checking the existence of a dominating set of size at most s in G. We conclude that $MD \leq_P DS$, and thus the M-dominating set problem is $\mathcal{NP}$-hard.

□

Note that for Δ-regular graphs where $\Delta = 2k - 1$, the notions of M-domination and k-domination become identical.

3.4.3 Stable and Decreasing Structures

In order to analyze the stabilization time of the M-dynamics, we first introduce the notions of *stability* and *decrease*.

Definition 3.4.4. Let $S \subseteq V$. S is **stable** if $S \subseteq D_M(S)$.

Definition 3.4.5. Let $S \subseteq V$. S is **decreasing** if $|D_M(S)| < |S|$.

Once stable structures become active under the M-dynamics at some round t, they remain active forever (i.e., for all rounds $t' \geq t$). Hence, we introduce the following definition.

Definition 3.4.6. A subset S of V is **stabilized** if S is active and stable. Its induced subgraph $G[S]$ is also said to be **stabilized**.

Note that, informally, a simple condition for polynomial time stabilization is the existence of a covering of G by *small stable structures* as we illustrate in Section 3.6 for cubic graphs.

Large bias. For large values of α, [Anagnostopoulos et al., 2020] shows that the stabilization time is polynomial for dense networks, but leaves as open the case for sparse networks. In the following theorem, whose proof is omitted for the sake of space, we show that the stabilization time is also fast for Δ-regular graphs whenever the bias α is greater than some threshold value that depends on the degree.

Theorem 3.4.2. *Let G be a Δ-random regular graph on n nodes. Whenever $\alpha \geq \frac{k-1}{\Delta}$ where $k = \lceil \frac{\Delta+1}{2} \rceil$, then the expected stabilization time for the Async M-Dynamics is polynomial.*

3.5 Fast stabilization for large α in random regular graphs

For large values of α, [Anagnostopoulos et al., 2020] shows that the stabilization time is polynomial for dense networks but leaves as open the case for sparse networks. In this section, we show that the stabilization time is also fast for Δ-regular (with $\Delta = 2k-1$ and $\Delta = 2k$) graphs whenever $\alpha \geq \frac{k-1}{\Delta}$.

3.5.1 Odd degree case

Theorem 3.5.1. *Let G be a random regular graph with odd degree $\Delta = 2k-1$ of size n. Whenever $\alpha \geq \frac{k-1}{\Delta}$, the expected stabilization time for the Async M-Dynamics is polynomial.*

Proof.

Let S_t be a random variable indicating the set of active nodes at round t and let $s_t = |S_t|$. Observe that the number of active nodes at time t increases by one with probability α if a node outside S_t is selected and with probability $1-\alpha$ if a node in $D_M(S_t) \setminus S_t$ is selected. Therefore, we get

$$E[s_{t+1} \mid S_t] = s_t + \frac{1}{n}\left(\alpha(|V| - s_t) + (1-\alpha)(|D_M(S_t)| - s_t)\right), \tag{3.1}$$

Next, we prove a lower bound for $D_M(S_t)$ that simply relies on the fact that S is incident to $\Delta|S|$ edges. The argument is that we have to pack those $\Delta|S|$ edges on the vertices: vertices

in $D_M(S)$ can receive up to Δ edges while in contrast vertices in $V \setminus D_M(S)$ receive at most $k-1$ edges since they are not M-dominated. It follows that

$$\Delta|D_M(S_t)| + (k-1)(n - |D_M(S_t)|) \geq \Delta s_t.$$

Since $\Delta = 2k-1$, we get

$$|D_M(S_t)| \geq \frac{\Delta s_t - (k-1)n}{k}.$$

Using the bound in (3.1), we obtain

$$E[s_{t+1} \mid S_t] \geq s_t + \frac{1}{n}\left(\alpha n - \alpha s_t + (1-\alpha)\left(\frac{\Delta s_t - (k-1)n}{k} - s_t\right)\right),$$

When regrouping terms, we get

$$\begin{aligned}
E[s_{t+1} \mid S_t] &\geq s_t + \frac{1}{n}\left(\frac{n}{k}(k\alpha - (1-\alpha)(k-1)) - \frac{s_t}{k}(k\alpha - (1-\alpha)\Delta + (1-\alpha)k)\right), \\
E[s_{t+1} \mid S_t] &\geq s_t + \frac{1}{n}\left(\frac{n}{k}(2k\alpha - k - \alpha + 1) - \frac{s_t}{k}(2k\alpha - k - \alpha + 1)\right) \\
E[s_{t+1} \mid S_t] &\geq s_t + \frac{1}{n}\left(\frac{n - s_t}{k}\right)((2k-1)\alpha - (k-1)) \\
E[s_{t+1} \mid S_t] &\geq s_t + \frac{1}{n}\left(\frac{n - s_t}{k}\right)(\Delta\alpha - (k-1)).
\end{aligned}$$

Now, since $n - s_t$ is always positive, we find out that, whenever $\Delta\alpha - (k-1) > 0$, the cardinal of S_t grows in expectation. Moreover, the drift $(E[s_{t+1} \mid S_t] - s_t)$ decreases with s_t, but always stays larger than $\frac{C}{n}$, with $C = \frac{\Delta\alpha - (k-1)}{k}$.
Let Z_t be the random variable defined as $Z_t = s_{t+1} - s_t \in \{-1, 0, 1\}$. Let τ be the stabilization time and $T > 0$. We have that

$$P[\tau \geq T] = P[\sum_{t=1}^{T} Z_t < n].$$

Since $s_t < n$ when $t < T$, we know that

$$E[\sum_{t=1}^{T} Z_t] \geq \frac{TC}{n}.$$

As $-1 \leq Z_t \leq 1$, we can use Hoeffding concentration inequality (see Lemma 2.3.3) to bound the probability to observe a deviation from the mean. We get

$$\begin{aligned}
P[\sum_{t=1}^{T} Z_t < n] &= P[E[\sum_{t=1}^{T} Z_t] - \sum_{t=1}^{T} Z_t > \frac{TC}{n} - n], \\
&\leq \exp\left(-\frac{2(\frac{TC-n^2}{n})^2}{4T}\right).
\end{aligned}$$

We now set T such that $\frac{2(\frac{TC-n^2}{n})^2}{4T} = \frac{1}{n}$, that is $T = \left(\frac{n\sqrt{2C+\frac{1}{n}}+\sqrt{n}}{\sqrt{2}C}\right)^2$, giving

$$P[\tau \geq T] \leq \exp\left(-\frac{1}{n}\right).$$

Now, let us prove that the expected stabilization time is polynomial. We have that

$$E[\tau] \leq \sum_{t=1}^{T} iP[\tau = i] + \sum_{t=T}^{\infty} iP[\tau \geq i].$$

A classic result is that $\sum_{i=0}^{\infty} i\exp(-i) = \frac{1}{e(1-1/e)}$. Thus,

$$E[\tau] \leq T + \frac{1}{e(1-1/e)} = \mathcal{O}(n^2),$$

giving the desired result.

□

3.5.2 Even degree case

We show here that the stabilization time is also fast for Δ-regular graphs of even degree $\Delta = 2k$ when $\alpha \geq \frac{k-1}{\Delta}$.

We note $T(S_t)$ the set of nodes with have k neighbors in S_t at time t.

Theorem 3.5.2. *Let G be a random regular graph with even degree $\Delta = 2k$ of size n. Whenever $\alpha \geq \frac{k-1}{\Delta}$, the expected stabilization time for the Async M-Dynamics is polynomial.*

Proof.

Let S_t be a random variable indicating the set of active nodes at round t and let $s_t = |S_t|$. Observe that the number of active nodes at time t increases by one with probability α if a node outside S_t is selected and with probability $1-\alpha$ if a node in $D_M(S_t) \setminus S_t$ is selected.

- +1 if the selected node is in $D_M(S_t) \setminus S_t$
- - 1 if the selected node is in $S_t \setminus (D_M(S_t) \cup T(S_t))$
- 0 if $D_M(S_t) \cap S_t$
- -1 or 0 with proba $1/2$ or 0 if $T(S_t) \cap S_t$
- +1 or 0 with proba $1/2$ if in $T(S_t) \setminus S_t$

Therefore, we get

$$\begin{aligned} E[s_{t+1} \mid S_t] = s_t + \frac{1}{n}\Big(&\alpha(|V| - s_t) \\ &+ (1-\alpha)(|D_M(S_t)| - s_t + \frac{1}{2}T(s_t))\Big), \end{aligned} \tag{3.2}$$

Next, we prove a lower bound for $D_M(S_t)$ that simply relies on the fact that S is incident to $\Delta|S|$ edges. The argument is that we have to pack those $\Delta|S|$ edges on the vertices: vertices

in $D_M(S)$ can receive up to Δ edges while in contrast vertices in $V \setminus D_M(S)$ receive at most $k-1$ edges since they are not M-dominated. It follows that

$$\Delta|D_M(S_t)| + kT(S_t) + (k-1)(n - |D_M(S_t)| - |T(S_t)|) \geq \Delta s_t.$$

Since $\Delta = 2k$, we get

$$|D_M(S_t)| \geq \frac{2ks_t - (k-1)n - |T(S_t)|}{k+1}.$$

Using the bound in (3.2), we obtain

$$E[s_{t+1} \mid S_t] = s_t + \frac{1}{n}\Big(\alpha(|V| - s_t) + (1-\alpha)(\frac{2ks_t - (k-1)n - |T(S_t)|}{k+1} - s_t + \frac{1}{2}T(s_t))\Big),$$

When regrouping terms, we get

$$E[s_{t+1} \mid S_t] \geq s_t + \frac{1}{n(k+1)}\Big((n-s_t)(\alpha\Delta - k + 1) + \frac{1-\alpha}{2}(k-1)T(S_t)\Big).$$

We observe that the worst case is when we do not have vertices with ties ($T(S_t) = 0$). We thus obtain the bound

$$E[s_{t+1} \mid S_t] - s_t \geq \frac{(n-s_t)(\alpha\Delta - k + 1)}{n(k+1)}.$$

Now, since $n - s_t$ is always positive, we find out that, whenever $\Delta\alpha - (k-1) > 0$, the cardinal of S_t grows in expectation. Moreover, the drift ($E[s_{t+1} \mid S_t] - s_t$) decreases with s_t, but always stays larger than $\frac{C}{n}$, with $C = \frac{\Delta\alpha-(k-1)}{k+1}$. Note that the condition for α is the same as in the odd case and that the constant is almost the same. The end of the proof thus is similar to the one for the odd case.

□

3.6 Fast Stabilization for Cubic Graphs

In this section, we show that cubic graphs stabilize in expected polynomial time.

Theorem 3.6.1. *Let G be a cubic graph of size n. The expected absorption time for the Async M-Dynamics in G is*

$$\mathbb{E}\left[\tau_\alpha(G)\right] = \mathcal{O}(n^{3-\mathcal{O}(\log\alpha)}\log^2 n).$$

The result is derived by observing that cubic graphs have a small girth (see Lemma 3.6.2) and by making use of properties that cycles are stable structures and that a path linking two stable substructures is itself stable (see Lemma 3.6.3). Indeed, cubic graphs are covered by logarithmic stable structures which ensures expected polynomial time stabilization. Note that Theorem 3.6.1

implies that $\mathbb{E}\left[\tau_\alpha(G)\right] = \mathcal{O}(n^{f(\alpha)})$ where f is an increasing function of the bias α. There exist families of graphs with finite stable structures such as planar cubic graphs for which the expected stabilization time is at most $g(\alpha)n^c$ (for some function g sensitive to the bias) where c does not depend on α.

Lemma 3.6.2. *[Bollobás, 2004]*
Let G be a Δ-regular graph on n nodes with $\Delta \geq 3$. Let $g(G)$ denote the girth of G. Then

$$g(G) \leq 2\log_{\Delta-1} n + 1.$$

Lemma 3.6.3. *Let G be a cubic graph on n nodes. Suppose that G has a subgraph S which is either a cycle of size l or a path of length l between two stabilized structures. Let $\bar{\tau}_S$ denote the number of rounds for S to become active. Then, we have*

$$\mathbb{E}\left[\bar{\tau}_S\right] = \mathcal{O}\left(\frac{1}{\alpha^l} n \log n\right).$$

We note that Lemma 3.6.3 implies that $\mathbb{E}\left[\bar{\tau}_S\right]$ is still $\mathcal{O}(n^{1+r}\log n)$ when $\alpha = \Theta(\frac{1}{n^r})$ for any $r > 0$, hence polynomial as long as r is constant.

We prove the first lemma about the substructure of a cycle.

Lemma 3.6.4. *Let G be a cubic graph of order n. Suppose that G has a cycle C of size c and let $\bar{\tau}_C$ denote the number of rounds for C to become active. Then, we have*

$$E[\bar{\tau}_C] = \mathcal{O}\left(\frac{1}{\alpha^c} n \log n\right).$$

Proof.

Let C be a cycle in G and let T_0 denote the number of time steps such that all nodes in G update their opinion at least once. We note that $T_0 = \mathcal{O}(n \log n)$, w.h.p., by coupon collector argument. Furthermore, we look at the state of the system within the time intervals $\{\mathcal{I}_i\}_{i\geq 0}$ where $\mathcal{I}_0 = [0, T_0]$ and $\mathcal{I}_i =]iT_0, (i+1)T_0]$ for $i \geq 1$. We refer to each time interval $\mathcal{I}_i$ by the i-th *master round*.

Let t_C denote the number of master rounds needed for C to become active and let F be a number indicating whether C succeeded in becoming active during the first master round, that is $F = 1$ if $\bar{\tau}_C \in \mathcal{I}_0$ and 0 otherwise. We note that: $\bar{\tau}_C \leq T_0 t_C$.

We denote by $p_\alpha = P(\bar{\tau}_C \in \mathcal{I}_0)$ the probability for C to become active after at most T_0 rounds. Observe that since $E[t_C|F = 1] = 1$ and $E[t_C|F = 0] = 1 + E[t_C]$, we have:

$$E[t_C|F] = \begin{cases} 1 & \text{with probability } p_\alpha, \\ 1 + E[t_C] & \text{with probability } 1 - p_\alpha. \end{cases}$$

Therefore, by the law of iterated expectation, we have

$$E[t_C] = E[E[t_C|F]] = p_\alpha + (1 - p_\alpha)(1 + E[t_C]).$$

Solving this equation, we get

$$E[\bar{\tau}_C] \leq E[T_0]E[t_C] \leq \frac{E[T_0]}{p_\alpha}. \tag{3.3}$$

Moreover, every time a node updates its opinion the node chooses opinion 1 with probability at least α. Therefore: $p_\alpha \geq \alpha^{|C|}$. and by Equation (3.3), the cycle C becomes active in $\mathcal{O}(\frac{1}{\alpha^{|c|}} n \log n)$ rounds, in expectation.

□

We have a similar lemma for another substructure.

Lemma 3.6.5. *Let G be a cubic graph of order n. Suppose that G has a path P of length l between two stabilized structures and let $\bar{\tau}_P$ denote the number of rounds for P to become active. Then, we have*

$$E[\bar{\tau}_P] = \mathcal{O}\left(\frac{1}{\alpha^l} n \log n\right).$$

Proof.

The proof is similar to the one of Lemma 3.6.4. The time for the path to become active is $\mathcal{O}(\frac{1}{\alpha^l} n \log n)$. Then, the path stays active as every node is connected to 2 active nodes.

□

We now prove the main theorem.

Theorem 3.6.1. *Let G be a cubic graph of size n. The expected absorption time for the Async M-Dynamics in G is*

$$\mathbb{E}\left[\tau_\alpha(G)\right] = \mathcal{O}(n^{3-\mathcal{O}(\log \alpha)} \log^2 n).$$

Proof.

At the initial step of the process, all nodes of the graph are idle. Since G is cubic, by Lemma 3.6.2, there exists a cycle C of size c at most $2\log_2 n + 1$ in G. Therefore, by Lemma 3.6.4, C becomes active after $E[\bar{\tau}_C]$ time steps where

$$E[\bar{\tau}_C] = \mathcal{O}\left(\frac{1}{\alpha^{\mathcal{O}(\log n)}} n \log n\right) = \mathcal{O}\left(\frac{1}{\alpha} n^{1-\mathcal{O}(\log \alpha)} \log n\right).$$

Furthermore, since $\delta(G[C]) = 2 \geq \lceil \frac{3}{2} \rceil$, C is stable. Hence, after $E[\bar{\tau}_C]$ rounds, nodes in G are either stabilized (namely, those in C) or idle. Note that some nodes may be active, but the worst case is to consider that they are none of them.

Let us consider an idle node v in at a time step $t \geq E[\bar{\tau}_C]$. Let E_{idle} denote the expected number of rounds for v to become active in G given that C is stabilized. Conducting a breadth first search (BFS) of G starting from v produces a BFS tree formed by three subtrees S_1, S_2, S_3 rooted at the three children of v. Moreover, let r_1 and r_2 be two stabilized nodes in C.

- If r_1 and r_2 both belong to different subtrees, then, there exists a path between r_1 and r_2 with at least an idle node (the node v) of length $\mathcal{O}(\log_3 n)$.

- Otherwise, r_1 and r_2 are all located in a single subtree, say S_1 without loss of generality. Consider another subtree, say S_2. When nodes at distance $\log_3 n$ from its root have been explored, n nodes have been visited, thus, including some of another subtree. This shows the existence of a cycle of length $\mathcal{O}(\log_3 n)$ with idle vertices (at least the ones in S_2).

Therefore, every idle node v belongs to either a path of length $\mathcal{O}(\log_3 n)$ between two stabilized nodes or to a cycle of length $\mathcal{O}(\log_3 n)$. By Lemma 3.6.4 and Lemma 3.6.5, we thus obtain

$$E_{\text{idle}} = \mathcal{O}\left(\frac{1}{\alpha^{\mathcal{O}(\log n)}} n \log n\right).$$

Note that there are at most $n - c$ idles nodes. Hence, we get

$$E[\tau] \leq E[\bar{\tau}_C] + (n-c)E_{\text{idle}}\, T_0,$$

where T_0 denote the number of time steps such that all nodes in G update their opinion at least once. We note that $T_0 = \mathcal{O}(n \log n)$, w.h.p., by coupon collector argument. We therefore get

$$E[\tau] = \mathcal{O}\left(\frac{1}{\alpha^{\mathcal{O}(\log n)}} n^3 \log^2 n\right) = \mathcal{O}(n^{3-\mathcal{O}(\log \alpha)} \log^2 n).$$

□

3.7 Slow Stabilization for Random Regular Bipartite Graphs With Odd Degree

In this section, we show that there exist graphs (namely, random Δ-regular bipartite graphs with odd degree [†] $\Delta \geq 5$) for which every linear substructure (of size smaller than Cn, with C a non-negative constant) is decreasing for small values of α. This leads to an exponential expected stabilization time.

3.7.1 Model Discussion

We consider random Δ-regular balanced bipartite graphs G of the form $G = (A \cup B, E)$ with $|A| = |B|$ and $E \subset (A, B)$. We first study the case of an odd degree $\Delta = 2k - 1 (k \geq 3)$ for which there cannot be ties under the majority update rule. Let $|A| = n$ and note that $|E| = \Delta n$. The random regular graph model [‡] that we use is analogous to the configuration model proposed by [Bollobás, 1980].

We shall prove that there exists an $\alpha_0 > 0$ such that for every $\Delta \geq 5$ the expected stabilization time is exponential whenever $\alpha \leq \alpha_0$. This phenomena is mainly due to the random structure which fosters the existence of small decreasing structures.

Construction. The random Δ-regular [§] bipartite graph is built as follows: each edge will be made of two *ends* (or *half edges*), one attached to a node in A and the other to a node in B. Then,

†. Similar results hold when the degree is even and G is non bipartite. See brief discussion in Section 3.8.

‡. The bipartite version was introduced by [Margulis, 1973], [Pippenger, 1977], and [Valiant, 1975] to prove that expanders exist.

§. Note that since G is regular and Δ is odd, it implies that the size n of the network is even.

edges are created by associating to each of the Δn half-edges adjacent to A one of the Δn half-edges adjacent to B in a one-to-one manner. We note that the injective mapping that we use can be identified with a random permutation of $S_{\Delta n}$. To this end, we first identify both A, B with $\{1, \dots, n\}$. Then, since each node is adjacent to Δ edges, by Hall's theorem, we can select randomly and independently Δ random elements $\pi_1, \dots \pi_\Delta$ from S_n and build G as the union of the Δ random matchings formed by the edges $\{(a, \pi_i(a)), a \in A, \pi_i(a) \in B, i \in \{1, \dots, n\}\}$.

3.7.2 Existence of Decreasing Sets

In the following, we prove (see Proposition 3.7.3) that there exist Δ-regular graphs for which all sufficiently small linear sets S M-dominate a strictly smaller set (with an actual linear gap).

To obtain this result, we employ Lemma 3.7.1 corresponding to the case for which sets S are contained in only one side of the bipartition of G (without loss of generality, we assume that $S \subseteq A$). To make the proof more comprehensive, we introduce first some auxiliary functions.

Definition 3.7.1. (Decrease functions For $(\sigma, \tau) \in [0, 1]^2$, we let $N(\sigma, \tau)$ be the expected number of pairs of subsets (S, T) with $S \subseteq A, T \subseteq B$ of respective sizes $\sigma n, \tau n$ such that S k-dominates T.

We define $\widehat{F} : (0, 1]^2 \to \mathbb{R}$ as

$$\widehat{F}(\sigma, \tau) = \frac{\log_2(N(\sigma n, \tau n))}{n}.$$

Moreover, for every β in $(0, 1)$, we define $\widehat{G}_\beta : (0, 1) \to \mathbb{R}$ as

$$\widehat{G}_\beta(\sigma) = \widehat{F}(\sigma, \beta\sigma).$$

and we refer to β by the decrease intensity.

Observe that if $\widehat{G}_\beta(\sigma) < 0$ then $N(\sigma, \beta\sigma) < 1$, which implies that all subsets S of size σn are decreasing (with a gap β). Therefore, we refer to $\widehat{F}$ and $\widehat{G}$ as decrease functions as they define regions of existence of decrease, namely when $\widehat{G}_\beta$ is negative.

Lemma 3.7.1. *Let $G = (A \cup B, E)$ be a random Δ-regular bipartite graph with $2n$ nodes. Then, there exist $\frac{1}{k-1} < \beta < 1$ and $\gamma_\Delta(\beta) > 0$ such that for any $0 < \lambda < \gamma_\Delta(\beta)$ we have*

$$\forall S \subset A, \lambda n \leq |S| \leq \gamma_\Delta(\beta) n : |D_M(S)| \leq \beta |S|.$$

Proof.

Let $N(s, t)$ be the expected number of pairs of subsets $S \subset A, T \subseteq B$ of respective sizes s, t such that S k-dominates T. Since we are studying sets of linear sizes we let $s = \sigma n$ and $t = \tau n$ with $\sigma, \tau \in (0, 1]$. We first count the configurations that fulfill the k-domination constraints in order to get an upperbound on $N(s, t)$:

Claim 3.7.2. *It holds that*

$$N(\sigma n, \tau n) \leq \binom{n}{\sigma n} \binom{n}{\tau n} \frac{\binom{\Delta \sigma n}{k \tau n}}{\binom{\Delta n}{k \tau n}} \binom{\Delta}{k}^{\tau n}.$$

Proof.

Observe that

$$N(s,t) = \binom{n}{s}\binom{n}{t}P_{st}, \tag{3.4}$$

where P_{st} is the probability that S k-dominates T.

We start by counting such configurations. For any vertex $t_0 \in T$, we select which of its adjacent half-edges are used to dominate it. There are $\binom{\Delta}{k}$ ways of selecting these k half-edges among the Δ half-edges adjacent to t_0. Since we do it for each node in T, this introduces a factor of $\binom{\Delta}{k}^t$. Remark too, that if t_0 is dominated by more than k vertices there are several choices for the k edges, so such configurations are counted several times and we do overestimate P_{st}. Then these selected kt half edges used to dominated T are constrained to be matched with some of the Δs half-edges incident to S. The number of possibilities to select their other is henceforth

$$\Pi_{i\in[0,kt-1]}(\Delta s - i) = \frac{(\Delta s)!}{(\Delta s - kt)!}.$$

To complete the process we must match the remaining $\Delta n - kt$ half-edges on both sides. To this end, we use a permutation of $\{1,\dots,\Delta n - kt\}$. Again note that this step may increase the domination since new extra edges may be created between S and T. Since there are $(\Delta n - kt)!$ such permutations, the final number of configurations is lesser than

$$\binom{\Delta}{k}^t \frac{(\Delta s)!}{(\Delta s - kt)!}(\Delta n - kt)!$$

And as there are $(\Delta n)!$ matchings between the two sets it follows that

$$P_{st} \leq \frac{1}{(\Delta n)!}\binom{\Delta}{k}^t \frac{(\Delta s)!}{(\Delta s - kt)!}(\Delta n - kt)!$$

Reorganizing, we get

$$P_{st} \leq \frac{(\Delta n - kt)!(kt)!}{(\Delta n)!}\frac{(\Delta s)!}{(\Delta s - kt)!(kt)!}\binom{\Delta}{k}^t = \frac{\binom{\Delta s}{kt}}{\binom{\Delta n}{kt}} \times \binom{\Delta}{k}^t.$$

Thus, the thesis follows by Equation (3.4).

□

Using Stirling's approximation, we get $\binom{n}{pn} \leq 2^{nH(p)}$ for every $\frac{1}{n} \leq p \leq \frac{1}{2}$. Hence, by Claim 3.7.2, we obtain the following property which provides upperbounds on the decrease functions (see Definition 3.7.1).

Property 1. *Let σ, τ be positive reals in $[\frac{1}{n}, \frac{1}{2}]$. We have*

$$\widehat{F}(\sigma,\tau) \leq F(\sigma,\tau),$$

where

$$F(\sigma,\tau) = H(\sigma) + H(\tau) + H\left(\frac{k\tau}{\Delta\sigma}\right)\Delta\sigma - H\left(\frac{k\tau}{\Delta}\right)\Delta + \tau\log_2\binom{\Delta}{k},$$

and H is the binary entropy function ¶. *Moreover, let*

$$G_\beta(\sigma) = F(\sigma, \beta\sigma).$$

We want to show that there exist σ and τ such that $\widehat{F}(\sigma,\tau) < 0$. This implies that $N(\sigma n, \tau n) < 1$ and therefore, that there exist random regular graphs with no subsets S of size σn that k-dominate a subset T of size τn.
Note that if there exists τ' such that $\tau' \leq \tau$ and $N(\sigma n, \tau' n) < 1$ then $N(\sigma n, \tau n) < 1$. Therefore, it is sufficient to let $\tau = \beta\sigma$ for a positive real $\beta \in (0,1)$ and prove that $\widehat{G}_\beta(\sigma) < 0$, or alternatively, by Definition 3.7.1 and Property 1 that $G_\beta(\sigma) < 0$. We have

$$G_\beta(\sigma) = H(\sigma) + H(\beta\sigma) - H\left(\frac{k\beta\sigma}{\Delta}\right)\Delta + \left(H\left(\frac{k\beta}{\Delta}\right)\Delta + \beta\log_2\binom{\Delta}{k}\right)\sigma.$$

Observe that for every $0 \leq x \leq \frac{1}{2}$, we have

$$-x\log_2(x) \leq H(x) \leq -x\log_2(x) - \frac{x}{2}.$$

Therefore

$$G_\beta(\sigma) \leq -\sigma\log_2(\sigma) - \beta\sigma\log_2(\beta\sigma) + k\beta\sigma\log_2\left(\frac{k\beta\sigma}{\Delta}\right) + \frac{\sigma}{2}(1+\beta) + \left(H\left(\frac{k\beta}{\Delta}\right)\Delta + \beta\log_2\binom{\Delta}{k}\right)\sigma.$$

That is

$$\frac{G_\beta(\sigma)}{\sigma} \leq (k\beta - (1+\beta))\log_2(\sigma) + C_\beta(k,\Delta), \tag{3.5}$$

where $C_\beta(k,\Delta) = H\left(\frac{k\beta}{\Delta}\right)\Delta + \beta\log_2\left(\frac{\Delta}{k}\right) - \beta\log_2(\beta) + k\beta\log_2(\frac{k\beta}{\Delta}) + \frac{(1+\beta)}{2}$ is a constant.

Since $\log_2(\sigma)$ diverges to $-\infty$ when $\sigma \to 0$, then $G_\beta(\sigma)/\sigma < 0$ implies that $k\beta - (1+\beta) > 0$, that is $\beta > \frac{1}{k-1}$. Furthermore, by setting

$$\sigma_\Delta(\beta) = \frac{\log 2}{\beta(k-1)-1}e^{-C_\beta(k,\Delta)} > 0,$$

then for every $0 < \sigma < \sigma_\Delta(\beta)$, we have $G_\beta(\sigma)/\sigma < 0$. Hence, there exists $\gamma_\Delta(\beta) \geq \sigma_\Delta(\beta)$ such that all linear sets $|S|$ of size σn with $0 < \sigma < \gamma_\Delta(\beta)$ satisfy $|D_M(S)| \leq \beta|S|$ with $\beta \in (\frac{1}{k-1}, 1)$ and the thesis follows.

□

Existence of decrease. Let us consider *regions* defined by $\mathcal{R}^\beta_\Delta = (\lambda, \gamma_\Delta(\beta)]$ where $\lambda > 0$ and

$$\gamma_\Delta(\beta) = \max\{\sigma \in (0,1], G_\beta(\sigma) < 0\}.$$

By Lemma 3.7.1, all linear sets S of size σn with $\sigma \in \mathcal{R}^\beta_\Delta$ are decreasing (with a gap of β) and will define a regime where the process is slow, leading to consensus on the superior opinion taking place after an exponential number of rounds. By plotting the variation of $G_\beta(\sigma)$ with σ (see

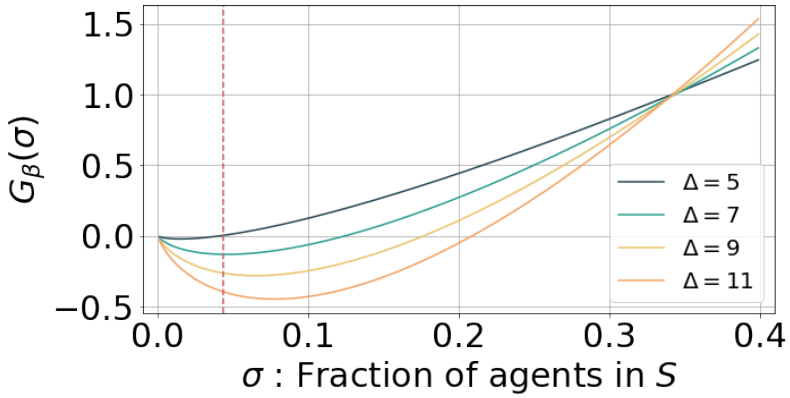


Figure 3.1 – Existence of regions $\mathcal{R}_\Delta^\beta$ for $\beta = 0.99$ in which all linear sets S of size σn satisfy $|D_M(S)| \leq \beta|S|$ (see Property 1).

Property 3.7.3), in Figure 3.1, we illustrate the regions $\mathcal{R}_\Delta^{0.99}$ for different values of Δ. For $\Delta = 5$, a red vertical line indicates the value of $\gamma_5(\beta) \sim 0.043$. Furthermore, we note that the highest is the degree Δ, the largest is the number $|\mathcal{R}_\Delta^\beta|$ of linear sets which are all decreasing in the network.

We now state the general case[‖] for which sets S can be in $A \cup B$ and not only in A.

Proposition 3.7.3. *Let $G = (A \cup B, E)$ be a random Δ-regular bipartite graph with n nodes. Then, there exist $\frac{1}{k-1} < \beta^+ < 1$ and $\gamma_\Delta(\beta^+) > 0$ such that for any $0 < \lambda < \gamma_\Delta(\beta^+)$ we have*

$$\forall S \subset A \cup B, \lambda n \leq |S| \leq \gamma_\Delta(\beta^+)n : |D_M(S)| \leq \beta^+|S|.$$

Proof.

Let $S_A = S \cap A, S_B = S \cap B$, then $S = S_A \cup S_B$ and $|S| = |S_A| + |S_B|$. Similarly, let $|D_M(S)| = |D_M(S_A)| \cup |D_M(S_B)|$ and $|D_M(S)| = |D_M(S_A)| + |D_M(S_B)|$.

Assume that $|S| = \sigma n$ with $0 < \sigma \leq 1$ and let r be a positive integer.

— If both $|S_A|, |S_B| \geq \sigma n/r$, we can apply Lemma 3.7.1 twice, giving $|D_M(S_A)| \leq \beta|S_A|$ and $|D_M(S_B)| \leq \beta|S_B|$. We get the result $|D_M(S)| \leq \beta|S|$.

— If one side is small, assume without loss of generality that $|S_A| \leq \sigma n/r$. Lemma 3.7.1 applies to S_B and we have $|D_M(S_B)| \leq \beta|S_B|$. For S_A, as the graph is Δ-regular, a set of size s cannot k-dominate a set of size larger than $\Delta s/k$. Thus, $|D_M(S_A)| \leq \frac{\Delta}{k}|S_A| \leq \frac{\Delta}{k}\frac{\sigma n}{r}$ and we get $D_M(S) \leq \beta|S_B| + \frac{\Delta}{k}\frac{\sigma n}{r} \leq (\beta + \frac{\Delta}{kr})|S|$. We set $\beta^+ = \beta + \frac{\Delta}{kr}$. By picking r such that $r > \frac{\Delta}{k(1-\beta)}$, we have $\beta^+ < 1$ and the thesis follows.

□

3.7.3 Exponential Stabilization

Theorem 3.7.4. *Let G be a random regular graph with odd degree Δ of size n. There exists $\alpha_\Delta > 0$ such that for every $\alpha < \alpha_\Delta, \mathbb{E}\left[\tau_\alpha(G)\right]$ is exponential for the Async M-Dynamics.*

¶. $H(x) = -(x\log_2(x) + (1-x)\log_2(1-x))$.

‖. In the case of Δ-regular graphs ($\Delta = 2k-1$), a k-domination corresponds to an M-domination. However, Proposition 3.7.3 can be made general without any relationship presumed with the degree Δ.

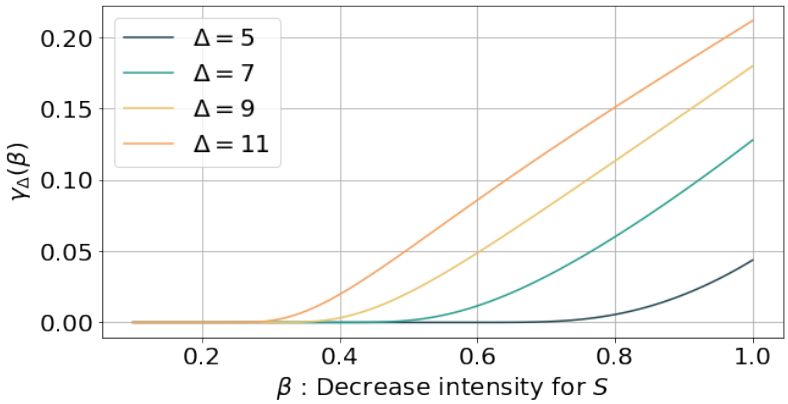


Figure 3.2 – γ_Δ tends to zero when β tends to $\frac{1}{k-1}$. Moreover, γ_Δ is an increasing function of Δ.

Proof.

Let S_t be a random variable indicating the set of active nodes at round t and let $s_t = |S_t|$. We first show that s_t has a *negative drift* inside a *linear time* interval. We then use this fact to prove that the expected stabilization time is exponential.

Negative drift. Observe that the number of active nodes at time t increases by one with probability α if a node outside S_t is selected and with probability $1-\alpha$ if a node in $D_M(S_t) \setminus S_t$ is selected. Therefore, we get

$$\mathbb{E}\left[s_{t+1} \mid S_t\right] = s_t + \frac{1}{n}\left(\alpha(|V| - s_t) + (1-\alpha)(|D_M(S_t)| - s_t)\right)$$

$$\mathbb{E}\left[s_{t+1} \mid S_t\right] - s_t \leq \alpha + \frac{1}{n}(1-\alpha)(|D_M(S_t)| - s_t) \tag{3.6}$$

By Proposition 3.7.3, there exist $\frac{1}{k-1} < \beta^+ < 1$ and $\gamma_\Delta(\beta^+) > 0$ such that for any $0 < \lambda < \gamma_\Delta(\beta^+)$, each subset S of V with $\lambda n \leq |S| \leq \gamma_\Delta(\beta^+)n$, satisfies $|D_M(S)| \leq \beta^+|S|$. We set $\lambda = \frac{\gamma_\Delta(\beta^+)}{r}n$ where r is a positive integer.

It follows that if $s_t \in [\frac{\gamma_\Delta(\beta^+)}{r}n, \gamma_\Delta(\beta^+)n]$, then since $\beta^+ - 1 < 0$ we get from Inequality (3.6) that

$$\mathbb{E}\left[s_{t+1}|S_t\right] - s_t \leq \alpha + \frac{(\beta^+ - 1)\gamma_\Delta(\beta^+)}{r}$$

Therefore, by picking any $\alpha < (1-\beta^+)\frac{\gamma_\Delta(\beta^+)}{r}$, we get that the sequence $\{s_t\}_{t\geq 0}$ has a fixed negative drift δ.

$$\mathbb{E}\left[s_{t+1} \mid S_t\right] - s_t \leq \delta < 0, \tag{3.7}$$

Exponential stabilization. We study the process when it is in the *critical* region for which the drift is negative. The following property is derived by noting that the time to exit a linear interval by a variant of a biased random walk on $\{s_t\}_{t\geq 0}$ with a fixed negative bias is exponential in expectation.

Property 2. *Let $\beta^+ \in [\frac{1}{k-1}, 1[$. Then $\mathbb{E}\left[\tau_\alpha(G)\right]$ is exponential when $\alpha < (1-\beta^+)\gamma_\Delta(\beta^+)$.*

Proof.

Let $T > 0$. We introduce two random variables $T_1 = \min\{t < T, s_t = \lceil \gamma_\Delta(\beta^+)n \rceil\}$ and $T_0 = \max\{t \leq T_1, s_t = \lceil \frac{\gamma_\Delta(\beta^+)}{2} n \rceil\}$. As the number of active nodes can increase by at most one per time step, i.e., $|S_t|$ can vary by the values in $\{1, -1\}$, then $\bar{t} \geq \frac{\gamma_\Delta(\beta^+)}{2} n$. Therefore, $[T_0, T_1]$ has a size linear in n. In the following, the main idea is to complete the argument by using concentration inequalities to show that due to the negative drift, traversing the interval $[\frac{\gamma_\Delta(\beta^+)}{2} n, \gamma_\Delta(\beta^+)n]$ takes $\sim e^{\Theta(n)}$ time.

We want to bound the probability $P[\tau_\alpha(G) < T]$ for any time T. Note first that if the process stabilizes before a time T, it means that there exist times $t_1 < T$ and $t_0 < t_1 < T$ such that $T_0 = t_0$ and $T_1 = t_1$. The reverse clearly is not true. So, summing on all possible values, we get

$$P[\tau_\alpha(G) < T] \leq \sum_{t_1=0}^{T} \sum_{t_0=0}^{t_1} P[T_1 = t_1 \cap T_0 = t_0].$$

As $P[T_1 = t_1 \cap T_0 = t_0] \leq P[T_1 = t_1 | T_0 = t_0]$, we are interested by bounding the right-hand side term.

Let X_t be the random variable defined as $Z_t = s_{t+1} - s_t \in \{-1, 0, 1\}$. Note that

$$\begin{aligned} P[T_1 = t_1 | T_0 = t_0] &= P[\sum_{t=t_0}^{t_1} Z_t = \frac{\gamma_\Delta(\beta^+)}{2} n] \\ &\leq P[\sum_{t=t_0}^{t_1} Z_t \geq \frac{\gamma_\Delta(\beta^+)}{2} n]. \end{aligned}$$

Since between T_0 and T_1, $s_t \in [\frac{\gamma_\Delta(\beta^+)}{2} n, \gamma_\Delta \beta^+ n]$ then by Inequality 3.7, we have for every $\alpha < (1-\beta^+)\frac{\gamma_\Delta(\beta^+)}{r}$ that $E[Z_t] \leq \delta$. Thus

$$E[\sum_{t=t_0}^{t_1} Z_t] \leq \delta \bar{t}.$$

Remark that the sum deviates from its mean by $\frac{\gamma_\Delta(\beta^+)}{2} n - \delta \bar{t}$. Since $-1 \leq Z_t \leq 1$, we can use Hoeffding concentration inequality (see Lemma 2.3.3) to bound the probability to observe such a deviation. We get

$$P[\sum_{t=t_0}^{t_1} Z_t \geq \frac{\gamma_\Delta(\beta^+)}{2} n] \leq \exp\left(-\frac{2(\frac{\gamma_\Delta(\beta^+)}{2} n - \delta \bar{t})^2}{4\bar{t}}\right).$$

The above function decreases with $\bar{t}$ and since $\bar{t} \geq \frac{\gamma_\Delta(\beta^+)}{2} n$, we get

$$P[\sum_{t=t_0}^{t_1} Z_t \geq \frac{\gamma_\Delta(\beta^+)}{2} n] \leq \exp\left(-\frac{(1-\delta)^2 \gamma_\Delta(\beta^+)}{4} n\right).$$

It thus gives

$$P[\tau_\alpha(G) < T] \leq T^2 \exp\left(-\frac{(1-\delta)^2 \gamma_\Delta(\beta^+)}{4} n\right),$$

with $\lambda = \frac{(1-\delta)^2 \gamma_\Delta(\beta^+)}{4} > 0$. Hence, for every exponential growth $T(n) \sim e^{\theta n}$ with $\theta < \frac{\lambda}{2}$, we have that $P[\tau_\alpha(G) \geq T(n)] = 1$ with high probability, and the thesis follows.

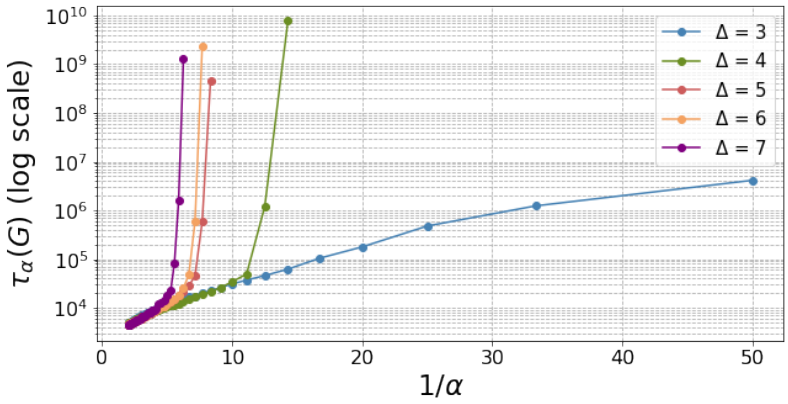


Figure 3.3 – Effect of the bias α on the stabilization time $\tau_\alpha(G)$ in a random Δ-regular network made up of 1000 agents.

□

Let $\alpha_{\beta^+} = (1-\beta^+)\gamma_\Delta(\beta^+)$. Note that Property 2 provides various bounds for the bias α_{β^+} depending on the decrease intensity β^+. The limiting cases yielding $\alpha_{\beta^+} \to 0$ occur when $\beta^+ \to 1$ or $\beta^+ \to \frac{1}{k-1}$ (in such case $\gamma_\Delta(\beta^+) \to 0$). In Figure 3.2, we observe that γ_Δ is an increasing function of Δ and confirm that $\frac{1}{k-1}$ is a limiting value for β as per Proposition 3.7.3. We get the best bound for α by maximizing over β^+. We obtain

Property 3. *Let* $\alpha_\Delta = \max_{\frac{1}{k-1}<\beta^+<1} (1-\beta^+)\gamma_\Delta(\beta^+)$. *Then,* $E[\tau_\alpha(G)]$ *is exponential for every* $\alpha < \alpha_\Delta$.

□

3.8 Experiments and Outlook

In this section, we present and discuss experiments on the stabilization time and the existence of decreasing structures.

Bias and stabilization. We first study the effect of the bias α on the stabilization time $\tau_\alpha(G)$. In Figure 3.3, we plot $\tau_\alpha(G)$ in a random Δ-regular network of size $n = 1000$ as a function of $\frac{1}{\alpha}$. Each experiment over G is averaged over 10 iterations and terminated if $\tau_\alpha(G)$ bypasses 10^{10} iterations (which requires a prohibitive computation time of 71 hours). When the bias is large (e.g. $\alpha \geq 0.15$), stabilization occurs very quickly and bears no significant dependence of the degree. The picture changes when the bias gets smaller where we observe a fast stabilization for $\Delta = 3$ (even for arbitrary small values of α) and a very clear explosion for $\Delta \geq 4$. Moreover, the higher the degree, the sooner the explosion occurs. We also see that on a network with only 1000 agents, the convergence for $\alpha = 0.08$ and $\Delta = 4$ takes at least 10^{10} iterations.

In Figure 3.4, we visualize the impact of the size n on stabilization in a random 5-regular network. We note the presence of two *regimes* (values of α) $\mathcal{R}_F$ and $\mathcal{R}_S$ depicted respectively in

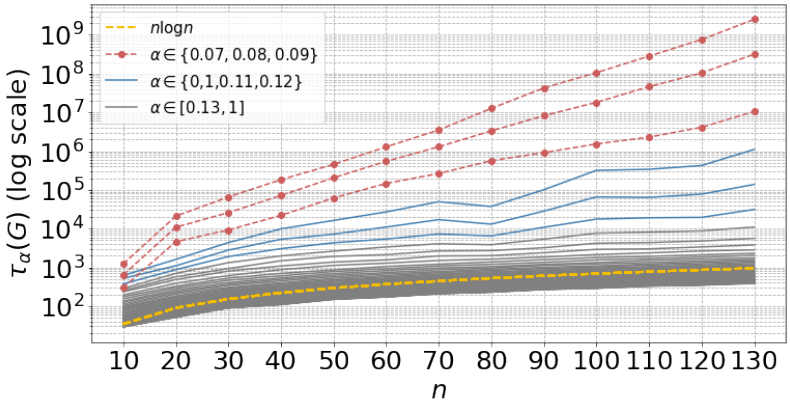


Figure 3.4 – Slow-fast dichotomy behavior of the stabilization time $\tau_\alpha(G)$ induced by α in random 5-regular networks of size n.

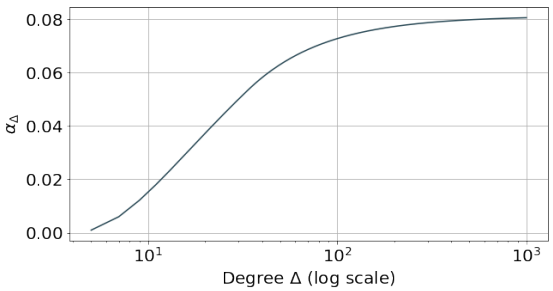


Figure 3.5 – Variation of α_Δ with Δ in a random Δ-regular bipartite graph.

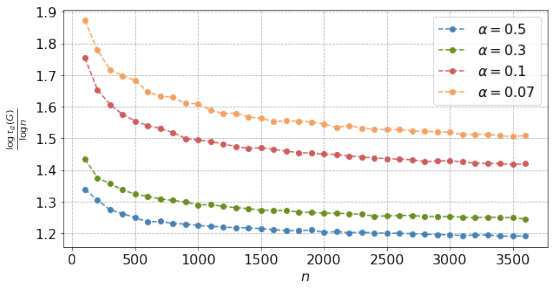


Figure 3.6 – $\log \tau / \log n$ is almost constant. Stabilization time $\tau_\alpha(G)$ induced by α in random cubic networks of size n.

gray and red. Each experiment over G is averaged over 500 (resp. 5) iterations for every α in $\mathcal{R}_F$ (resp. $\mathcal{R}_S$).

The first regime $\mathcal{R}_F$ corresponds to a large bias ($\alpha \geq 0.13$) for which we observe that the network stabilizes in polynomial time ($\log \tau / \log n$ is almost constant). We also conducted experiments on large cubic networks with α small, confirming what we proved in Theorem 3.6.1, that is $\mathbb{E}\left[\tau_\alpha(G)\right] \leq n^{f(\alpha)}$ (see Figure 3.6).

In the second regime $\mathcal{R}_S$, stabilization takes an exponential (note that τ is observed on a logarithmic scale) number of rounds ($\log \tau / \log n$ increases with n) and this occurs as soon as we start from $\alpha \sim 0.09$.

Outlook. This chapter leaves a number of open questions. A first one concerns closing the gap $g_\Delta = \alpha_e - \alpha_\Delta$ between the empirical and the theoretical values for the bias below which (expected) exponential stabilization occurs due to the existence of decreasing structures. When $\Delta = 5$, the best bound α_Δ is attained with a decrease intensity of $\beta \sim 0.9$ for which we get $g_5 \sim 0.08903$. Note that for an arbitrary large degree Δ, we get $\alpha_\Delta \sim 0.3$ (see Figure 3.5). Furthermore, when $\Delta \geq 5$, the evolution of the stabilization time during the intermediate regime (depicted in blue in Figure 3.4) is not completely clear and might suggest an intermediate stabilization growth (i.e., superpolynomial and subexponential) or a sharp transition between $\mathcal{R}_F$ and $\mathcal{R}_S$.

We also point a rather surprising result: in the course of the proof of Proposition 3.7.3, we showed that it is impossible for any sufficiently small linear set S to k-dominate more that $\frac{|S|}{k-1}$ nodes (note that this bound is tight), which implies that for some $a_0 > 0$ depending only on α and Δ, we get that $\min_{|S|\leq a_0 n} D_k(S) \sim \frac{|S|}{k-1}$. Thus, maximizing almost exactly $D_k(S)$ is trivial for small linear sets, but for larger sets, the question remains open. Unlike in the Erdős–Rényi model, domination problems have not yet been solved in random regular graphs even if some works [Duckworth and Wormald, 2006, Hoppen and Mansan, 2021]) have been done. We remark that the lack of accurate results in this field somewhat preludes the gaps in our bounds since the dynamics relate to some already complex domination based questions. Finally, we believe that the techniques we have exposed can be used to prove many other results. For instance, we proved similar results in the general case of random regular graphs and planar cubic graphs. While the arguments are similar, the analysis is slightly more complicated to expose due to ties in the even degree case and the fact that $S \cap D_k(S) \neq \emptyset$ in the non bipartite case.

3.9 Synchronous model

We presented our results for the Asynchronous model, but they all also hold in the Synchronous case. Moreover, the proofs are almost exactly identical, with only cosmetic changes. This is a consequence of the following key observation. Let S_t be a random variable indicating the set of active nodes at round t and let $s_t = |S_t|$. We have

$$E[s_{t+1} \mid S_t] \overset{\text{Sync}}{=} \alpha|V| + (1-\alpha)|D_M(S_t)|$$
$$E[s_{t+1} \mid S_t] \overset{\text{Async}}{=} s_t + \frac{1}{n}\left(\alpha(|V| - s_t) + (1-\alpha)(|D_M(S_t)| - s_t)\right),$$

We observe that, since all the n nodes play simultaneously, the variation of s_t for the Sync model is exactly n times the one for the Async model.

As an example, we prove that the result of Theorem 3.5.1 for Async, translates to an equivalent result for Sync.

Theorem 3.9.1. *Let G be a random regular graph with odd degree $\Delta = 2k-1$ of size n. Whenever $\alpha \geq \frac{k-1}{\Delta}$ . . . , the expected stabilization time for the Sync M-Dynamics is polynomial.*

Proof.

Let S_t be a random variable indicating the set of active nodes at round t and let $s_t = |S_t|$. Observe that:

$$\begin{aligned} E[s_{t+1} \mid S_t] &= s_t + \alpha(|V| - s_t) + (1-\alpha)(|D_M(S_t)| - s_t), \\ &= \alpha|V| + (1-\alpha)|D_M(S_t)|. \end{aligned} \tag{3.8}$$

Recall the following lower bound for $D_M(S_t)$ which holds independently of the model

$$|D_M(S_t)| \geq \frac{\Delta s_t - (k-1)n}{k}.$$

Using the bound in (3.8), we obtain

$$E[s_{t+1} \mid S_t] \geq \frac{(\alpha\Delta - k + 1)n + (1-\alpha)\Delta s_t}{k}.$$

We observe that , whenever $\Delta\alpha-(k-1) > 0$, the cardinal of S_t grows in expectation. Moreover, we have

$$\begin{aligned} E[s_{t+1} \mid S_t] - s_t &\geq \frac{(\alpha\Delta - k + 1)n + ((1-\alpha)\Delta - k)s_t}{k}, \\ &\geq C(n - s_t), \end{aligned}$$

where $C = \frac{\alpha\Delta-k+1}{k}$.
Thus, the drift $(E[s_{t+1} \mid S_t] - s_t)$ decreases with s_t and always stays larger than $C(n - s_t)$ when $t < \tau$.
Let Z_t be the random variable defined as $Z_t = s_{t+1} - s_t \in \{-1, 0, 1\}$. Let τ be the stabilization time and $T > 0$. We have that

$$P[\tau \geq T] = P[\sum_{t=1}^{T} Z_t < n].$$

Since $s_t < n$ when $t < T$, we know that

$$E[\sum_{t=1}^{T} Z_t] \geq TC$$

As $-1 \leq Z_t \leq 1$, we can use Hoeffding concentration inequality (see Lemma 2.3.3) to bound the probability to observe a deviation from the mean. We get

$$\begin{aligned} P[\sum_{t=1}^{T} Z_t < n] &= P[E[\sum_{t=1}^{T} Z_t] - \sum_{t=1}^{T} Z_t > TC - n], \\ &\leq \exp\left(-\frac{2(TC-n)^2}{4T}\right). \end{aligned}$$

We now set T such that $\frac{2(TC-n)^2}{4T} = \frac{1}{n}$, that is $T = \frac{n}{C} + \frac{\sqrt{2cn^2+1}+1}{c^2 n}$, giving

$$P[\tau \geq T] \leq \exp\left(-\frac{1}{n}\right).$$

Now, let us prove that the expected stabilization time is polynomial. We have that

$$E[\tau] \leq \sum_{t=1}^{T} iP[\tau = i] + \sum_{t=T}^{\infty} iP[\tau \geq i].$$

A classic result is that $\sum_{i=0}^{\infty} i\exp(-i) = \frac{1}{e(1-1/e)}$. Thus,

$$E[\tau] \leq T + \frac{1}{e(1-1/e)} = \mathcal{O}(n),$$

giving the desired result.

□

3.10 Conclusion and Future Work

In this chapter, we studied a biased opinion dynamics where agents are influenced by the majority of their neighbors. We have shown that consensus on the preferred opinion exhibits a dichotomy by proving that convergence time is always polynomial for cubic graphs, whereas it becomes exponential (for a small enough bias) in random Δ-regular graphs ($\Delta \geq 4$), answering an open conjecture in the literature. Moreover, we analyzed this dichotomy by exploiting structural properties of graphs in light of majority domination. An interesting avenue for further research is to to extend our results to the case of multiple opinions.

CHAPTER 4

Deterministic Bootstrap Percolation

Contents

4.1 Introduction

Originally developed by von Neumann after a suggestion of Ulam, *cellular automata* are a dynamical system defined over graphs endowed with some homogeneous and local update rules. The notion of *bootstrap percolation* is an example of a cellular automaton which has been extensively investigated by mathematicians, physicists, computer scientists and sociologists, among others. For further applications to several other areas, we refer the reader to the survey article by Adler and Lev [Adler and Lev, 2003], and the references therein.

Several variants of classical bootstrap percolation on graphs have been considered. The particular model we are studying belongs to the class of r-*neighbour bootstrap processes* ($r \geq 1$). It was first introduced by Chalupa, Leath, and Reich [Chalupa et al., 1979] as a monotone version of the Glauber dynamics in ferromagnetism.

Let $r \geq 1$ be a given integer and G a finite graph whose vertices (usually called *sites* in the context of percolation) can be in one of two states: *healthy* or *infected*. An initial set of infected nodes is given. At any step, any healthy vertex with at least r infected neighbours becomes infected, while infected vertices remain infected forever. The parameter r reflects at which extent sites *resist* to the infection spread. Note that the term *bootstrap percolation* is sometimes used with the implicit assumption that the initial set of vertices infected is random. In this chapter we consider the same model but use a deterministic initial set.

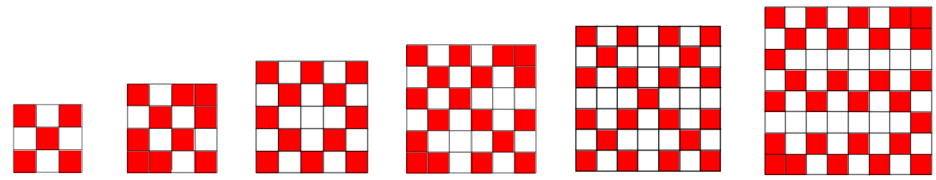
An initial set of infected vertices is called *lethal* if at the end of the process all the vertices are infected. A problem widely considered in the literature consists in determining the minimum size of a lethal set denoted by $s_r(G)$. The determination of this value provides a starting role in the study of the probability for a random initial set to be lethal.

Related work. Lethal sets have been considered in the literature under various names according to the domain of applications. In [Centeno et al., 2011, Dreyer Jr and Roberts, 2009], they are called irreversible r-conversion sets and infected vertices are called vertices in state 1. In the area of viral marketing [Chiang et al., 2013b, Chiang et al., 2013a], a lethal set is called a perfect target set and infected vertices are called active. In [Flocchini et al., 2004], lethal sets are called irreversible dynamic monopolies (dynamos, for short) and infected vertices are called black or faulty, stemming from the distributed computing field. Note that reversible dynamos [Peleg, 2002] have been widely investigated.

Computing exactly $s_r(G)$ is NP-hard [Centeno et al., 2011, Chen, 2009, Dreyer Jr and Roberts, 2009] and the hardness result holds even when $r = 2$ and G has maximal degree Δ, where Δ is a constant not depending on the size of G [Chen, 2009].

The parameter $s_r(G)$ has been studied for several families of graphs including trees [Centeno et al., 2011, Dreyer Jr and Roberts, 2009, Riedl, 2012], d-dimensional grids [Balogh and Bollobás, 2006, Balogh and Pete, 1998, Przykucki and Shelton, 2020], chordal graphs [Bessy et al., 2019, Centeno et al., 2011, Chiang et al., 2013a], hexagonal grids [Adams et al., 2011], honeycombs graphs [Chiang et al., 2013b], hypercubes [Balogh and Bollobás, 2006, Morrison and Noel, 2018], expanders [Coja-Oghlan et al., 2014], and random regular graphs [Guggiola and Semerjian, 2015].

Cases of grids and tori. In particular, the most known case is the square grids $G_{n\times n}$ with $r = 2$ as it is referenced in puzzles books [Winkler, 2003, Bollobás, 2006, Levitin and Levitin, 2011] and appears as a popular puzzle regularly in newspapers. According to [Winkler, 2003], the problem appeared for the first time in the Soviet magazine Kvant around 1986. The value $s_2(G_{n\times n}) = n$ can be proved with the elegant perimeter argument. Our interest to the problem



(a) $n = 3$ (b) $n = 4$ (c) $n = 5$ (d) $n = 6$ (e) $n = 7$ (f) $n = 8$

Figure 4.1 – Some instances of minimum lethal sets in $G_{n\times n}$ when $r = 3$. The vertices are depicted as small squares. Initially infected (resp., not infected) vertices are red (resp., white).

was revived, in fact, due to a problem proposed in the french newspaper "Le Monde" (Problem 1141, entitled "Le jeu de la viralité", April 2020) in honor to John Conway. This led us to study the open problem of determining the value $s_3(G_{n\times n})$. Some examples of minimul lethal sets for small grids when $r = 3$ are illustrated in Figure 4.1.To the best of our knowledge, this problem was first considered in [Pete, 1997, Balogh and Pete, 1998]. In [Pete, 1997] it is proven that $\lceil (n^2+2n)/3 \rceil \leq s_3(G_{n\times n}) \leq (n^2+4n)/3 + O(1)$. The lower bound uses also the perimeter argument. In [Bollobás, 2006], Bollobás gave the same lower bound for n odd, but a slightly better one for n even. He also proves that for $n \equiv 2 \pmod 6$ the lower bound is attained and in that case $s_3(G_{n\times n}) = (n^2+2n+4)/3$ (see Figure 4.1(f) for $n = 8$). The case of the grid was also considered in [Dreyer Jr and Roberts, 2009] where they claim that $s_3(G_{n\times n}) \leq n^2/3 + (7n+55)/12$, but the known lower bound of $(n^2-2n+2)/3$ [Bollobás, 2006, Pete, 1997] is larger than than their upper bound for n large enough.

The problem for tori $T_{m\times n}$ was considered and solved in [Bollobás, 2006, Flocchini et al., 2004, Pete, 1997, Winkler, 2003] for $r = 2$, where $s_2(T_{m\times n}) = \lceil \frac{n+m}{2} \rceil - 1$. For the case $r = 3$, it has been proved that $\lceil (mn+1)+1)/3 \rceil \leq s_3(T_{m\times n}) \leq \max\{\lceil m/3 \rceil (n+1), \lceil n/3 \rceil (m+1)\}$ [Flocchini et al., 2004]. Finally, in [Dreyer Jr and Roberts, 2009], exact expressions of $s_4(G_{m\times n})$ and $s_4(T_{m\times n})$ are given.

Other related problems have been considered in the literature such as finding the size of a largest inclusion-minimal lethal set [Morris, 2009, Riedl, 2010] and determining the speed of convergence in the square grid [Benevides and Przykucki, 2013, Benevides and Przykucki, 2015].

Our contributions. In this chapter, we study the contamination process in square grids $G_{n\times n}$ and square tori $T_{n\times n}$. The tools developed here can be used to get results for non square grids and tori. In Section 4.2, we formally define the problem and recall the results obtained for $r = 2, 4$ and some known results for $r = 3$ that we use throughout the chapter. Our main result is the following theorem that deals with an almost tight bound of $s_3(G_{n\times n})$.

Theorem 4.1.1. *Let $n \in \mathbb{N}^*$. Let $LB_n = \lceil \frac{n^2+2n}{3} \rceil$ if n is odd and $LB_n = \lceil \frac{n^2+2n+4}{3} \rceil$ if n is even.*

- *$s_3(G_{n\times n}) = LB_n$ if n is even or if $n \equiv 5$ mod 6 or if $n = 2^p - 1$;*
- *$LB_n \leq s_3(G_{n\times n}) \leq LB_n + 1$ if n is odd;*
- *$s_3(G_{n\times n}) = LB_n + 1$ if $n \in \{9, 13\}$.*

Section 4.3 is devoted to the proof of the first two items of Theorem 4.1.1. In Section 4.4, we prove the last item of Theorem 4.1.1. For the values $n = 9$ and 13, we prove that the lower bound is not

tight using a Linear Program. For $n = 9$, we also give a combinatorial proof. Then, Section 4.5 is devoted to settle the problem when $r = 3$ in tori:

Theorem 4.1.2. *For every $n \geq 3$, $s_3(T_{n\times n}) = \lceil \frac{n^2+1}{3} \rceil$.*

We conclude in Section 4.6 by describing several open questions.

4.2 Preliminaries

4.2.1 Notations and Problem statement

Let $G = (V, E)$ be a graph. For a vertex $v \in V$, $N(v) = \{u \in V \mid \{u, v\} \in E\}$ is the neighbourhood of v and $d(v) = |N(v)|$ is the *degree* of v.

The 2-dimensional (square) grid graph with n rows and n columns is denoted by $G_{n\times n}$. Its vertex set is defined by $V(G_{n\times n}) = \{(i, j) \mid 1 \leq i \leq n, 1 \leq j \leq n\}$. Two vertices (i, j) and (i', j') are adjacent if and only if $|i - i'| + |j - j'| = 1$. If $1 < i, j < n$, vertex (i, j) has degree 4. The *corners* are the vertices with degree at most 2, that is, the vertices $(1, 1), (1, n), (n, 1)$ and (n, n). The *border* $\mathcal{B}$ is the set of vertices with degree at most 3.

For $n \geq 3$, let also $T_{n\times n}$ be the torus with vertex-set $\{(i, j) \mid i \in \mathbb{Z}_n \text{ and } j \in \mathbb{Z}_n\}$, where $\mathbb{Z}_n$ denotes the set of integers modulo n denoted $0, 1, \ldots, n - 1$. Vertex (i, j) is adjacent to the 4 vertices $(i - 1, j)$, $(i + 1, j)$ $(i, j - 1)$ and $(i, j + 1)$.

Given a graph G and a natural number $r \in \mathbb{N}^*$, we consider the following deterministic process (known as r-neighbour bootstrap percolation) that models the spread of an infection in G. Let $D_0 = D$ be a subset of initially *infected* vertices and for each step $t \geq 1$, let

$$D_t = D_{t-1} \cup \{v \in V : |N(v) \cap D_{t-1}| \geq r\}.$$

For a set $D \subset V(G)$, let $\langle D \rangle = \bigcup_{t=0}^{\infty} D_t$ be the closure of D, i.e., the set of eventually infected vertices in the process that was started from D. A *configuration* is a pair (G, D) that consists of a graph G together with the subset D of its vertices that are initially infected. Abusing the notation, we call the *size* of the configuration as the size $|D|$ of the set D.

Any set $D \subseteq V$ is said *lethal* if $\langle D \rangle = V$. Similarly, a configuration (G, D) is lethal if D is a lethal set. Let $s_r(G)$ be the minimum size of a lethal set in G. Since $\langle V \rangle = V$, $s_r(G)$ is always well defined. A lethal set (resp. a lethal configuration) is *optimal* if it is of size $s_r(G)$. Note that $s_1(G) = 1$ for every connected graph G.

The main goal of this chapter is to investigate $s_3(G_{n\times n})$ and $s_3(T_{n\times n})$, i.e., focusing on square grids and tori for $r = 3$. Let us first deal with the cases $r \in \{2, 4\}$.

4.2.2 Cases $r \in \{2, 4\}$

For completeness, we first precisely recall the known results when $r = 2$ and give exact results in the case when $r = 4$.

The following result for the grid $G_{n\times n}$ and $r = 2$ can be considered as folklore and appeared in many books with puzzles [Bollobás, 2006, Levitin and Levitin, 2011, Winkler, 2003]. The lower-bound is proved using an elegant perimeter argument and an optimal lethal set is obtained by taking the n vertices of the diagonal.

Theorem 4.2.1. *[Bollobás, 2006, Levitin and Levitin, 2011, Winkler, 2003] Let $n \geq 1$, then $s_2(G_{n\times n}) = n$.*

The result for the torus can be found in [Bollobás, 2006, Flocchini et al., 2004, Winkler, 2003].

Theorem 4.2.2. *[Bollobás, 2006, Flocchini et al., 2004, Winkler, 2003]*
Let $n \geq 1$, then $s_2(T_{n\times n}) = n - 1$.

The case $r = 4$ was considered in [Balogh and Pete, 1998] where it is only stated that $s_4(G_{n\times n}) \sim n^2/2$, and has been completely solved in [Dreyer Jr and Roberts, 2009].

Proposition 4.2.3. *[Dreyer Jr and Roberts, 2009]*
For every $n \geq 3$, we have
— $s_4(G_{n\times n}) = \lfloor \frac{n^2+4n-4}{2} \rfloor$.
— $s_4(T_{n\times n}) = n\lfloor \frac{n+1}{2} \rfloor$.

4.2.3 Preliminaries on $s_3(G_{n\times n})$

We first present two properties that we will use afterwards.

Property 4. *When $r = 3$, every lethal set in $G_{n\times n}$ contains every corner.*

Proof.

Let u be a corner of $G_{n\times n}$ and let D be a lethal set. Since u has degree at most 2 and requires at least 3 neighbours to be infected, then $u \in D$.

□

The next property states that, for every two adjacent vertices on the border, at least one must be initially infected in order to eventually infect the whole grid.

Property 5. *Let u, v be two adjacent vertices in $\mathcal{B}$ of $G_{n\times n}$ and let D be a lethal set for $r = 3$, then $D \cap \{u, v\} \neq \emptyset$.*

Proof.

Let D be a lethal set of $G_{n\times n}$. For purpose of contradiction, let us assume that there are two adjacent vertices u and v of the border (so they have degree at most 3) that are not in D. Then, the first vertex to be contaminated in $\{u, v\}$ would have at most two contaminated neighbours before being infected, a contradiction.

□

The following lower bound has been proved in [Bollobás, 2006] using the perimeter argument and noting that, for n even, there are four pairs of adjacent infected vertices in $\mathcal{B}$. In particular, the notation LB_n defined below will be used throughout the chapter. We provide for completeness a sketch of the proof.

Theorem 4.2.4. *Problem 65, pages 171-172 in [Bollobás, 2006] Let $n \in \mathbb{N}^*$. It holds that:*

$$s_3(G_{n\times n}) \geq LB_n = \begin{cases} \left\lceil \frac{n^2+2n}{3} \right\rceil & \text{if } n \text{ is odd,} \\ \left\lceil \frac{n^2+2n+4}{3} \right\rceil & \text{if } n \text{ is even.} \end{cases}$$

Proof.

We rephrase the elegant perimeter's argument used in [Bollobás, 2006]. Each vertex has a perimeter 4 and so the initial perimeter of a lethal set is at most $4|D|$. In the process of infection, each time a new vertex is infected, the perimeter decreases by at least 2 and the final perimeter is $4n$. Hence, we get $4|D| - 2(n^2 - |D|) \geq 4n$ and so $|D| \geq \frac{n^2+2n}{3}$. If n is even, by Properties 4 and 5, there are on each side of the border $\mathcal{B}$ a pair of adjacent infected vertices. Therefore, the initial perimeter is at most $4|D| - 8$ and so we get $|D| \geq \frac{n^2+2n+4}{3}$.

□

This allows to prove the following corollary that we extensively use later.

Corollary 4.2.5. *Let $n \in \mathbb{N}$.*

$$LB_{n+3} - LB_n = \begin{cases} 2n+3 \text{ when } n \equiv 0,4 \pmod 6 \\ 2n+4 \text{ when } n \equiv 2 \pmod 6 \\ 2n+6 \text{ when } n \equiv 5 \pmod 6 \\ 2n+7 \text{ when } n \equiv 1,3 \pmod 6 \end{cases}$$

$$LB_{n+6} - LB_n = 4n + 16.$$

In [Bollobás, 2006], Bollobás proved that the lower bound LB_n is indeed reached in the case $n \equiv 2 \pmod 6$ (see Figure 4.1(f) for $n = 8$). We rephrase his result as follows by specifying an extra of some optimal lethal sets.

Theorem 4.2.6. *[Bollobás, 2006] Let $n \equiv 2 \pmod 6$, then $s_3(G_{n\times n}) = LB_n = \frac{n^2+2n+4}{3}$. Furthermore, there exists an optimal lethal set of $G_{n\times n}$ containing the vertices $(1, n-1)$, $(2, n)$, $(n-1, 1)$, and $(n, 2)$.*

4.3 Various constructions and optimal results

In this section, we present several ways to use (lethal or not) configurations for smaller grids in order to obtain new lethal configurations for larger grids. Each of the different tools that we propose allows us to prove Theorem 4.1.1.

4.3.1 From n even to $n+3$ odd and optimality for $n \equiv 5 \pmod 6$

Proposition 4.3.1. *Let n be even. If D is a lethal set in $G_{n\times n}$ containing the vertices $(1, n-1)$, $(2, n)$, $(n-1, 1)$, and $(n, 2)$, then there exists a lethal configuration $(G_{(n+3)\times(n+3)}, D')$ with size $|D'| = |D| + 2n + 4$.*

Proof.

From the configuration $(G_{n\times n}, D)$, let $(G_{(n+3)\times(n+3)}, D')$ be the configuration defined as follows. The restriction of D' to the subgrid $G_{n\times n}$ consisting of the first (topmost) n rows and first (leftmost) n columns of $G_{(n+3)\times(n+3)}$ will be the set D minus the vertices $(1, n)$ and $(n, 1)$ (which belong to D since D is a lethal set in $G_{n\times n}$ and they are corners). Moreover, D' contains the following infected vertices: $(1, n+1)$ and $(n+1, 1)$, plus the vertices $(n+2, 2j+2)$ for every $0 \leq j \leq n/2$ and $(n+3, 2j+1)$ for every $0 \leq j \leq n/2+1$. Symmetrically, D' contains the vertices $(2j+2, n+2)$ for every $0 \leq j \leq n/2-1$ and $(2j+1, n+3)$ for every

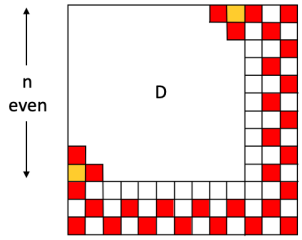


Figure 4.2 – Illustration of Proposition 4.3.1. The vertices are depicted as small squares (with most vertices inside $G_{n\times n}$ as well as their status omitted). Initially infected (resp., not infected) vertices of D' are red (resp., white). Orange vertices are the ones that are initially infected in D but not in D'.

$0 \leq j \leq n/2$. See Figure 4.2 for an illustration. Hence, we have $2n+6$ infected vertices in the last (bottommost) 3 rows and last (rightmost) 3 columns. Overall, $|D'| = |D| + 2n + 4$.

Now let us prove that D' is a lethal set. The vertices $(1, n)$ and $(n, 1)$ will become infected at the first step because they have 3 infected neighbours initially. Therefore, every vertex of the subgrid $G_{n\times n}$ will become infected (as D is a lethal set). At the first step, all the vertices of rows and columns $n+2$ and $n+3$ are also infected plus the vertices $(n+1, 2)$ and $(2, n+1)$. From these last two vertices the infection propagates along row $n+1$ and column $n+1$, until it reaches the vertex $(n+1, n+1)$.

□

By Corollary 4.2.5, we get:

Corollary 4.3.2. *Let D be a lethal set for $G_{n\times n}$ such that $\{(1, n-1), (2, n), (n-1, 1), (n, 2)\} \subseteq D$. If $n \equiv 0$ or $4 \pmod 6$, then $s_3(G_{(n+3)\times(n+3)}) - LB_{n+3} \leq |D| - LB_n + 1$. If $n \equiv 2 \pmod 6$, then $s_3(G_{(n+3)\times(n+3)}) - LB_{n+3} \leq |D| - LB_n$.*

Theorem 4.3.3. *Let $n \equiv 5 \pmod 6$. Then, $s_3(G_{n\times n}) = LB_n = \frac{n^2+2n+1}{3}$.*

Proof.

By Theorem 4.2.6, since $n - 3 \equiv 2 \pmod 6$, $s_3(G_{(n-3)\times(n-3)}) = LB_{n-3}$. Moreover, there exists an optimal lethal configuration of $G_{(n-3)\times(n-3)}$ containing the vertices $(n-1, 1), (n, 2), (1, n-1), (2, n)$. The result follows from the Corollary 4.3.2.

□

4.3.2 From n even to $n+6$ (even) and optimality for every n even

Proposition 4.3.4. *Let n be even. If D is a lethal set in $G_{n\times n}$ such that $\{(1, n-1), (2, n), (n-1, 1), (n, 2)\} \subseteq D$, then there exists a lethal configuration $(G_{(n+6)\times(n+6)}, D'')$ with size $|D''| = |D| + 4n + 16$ and such that $\{(1, n+5), (2, n+6), (n+5, 1), (n+6, 2)\} \subseteq D''$.*

Proof.

From the configuration $(G_{n\times n}, D)$, we first construct the configuration $(G_{(n+3)\times(n+3)}, D')$ as in Proposition 4.3.1 and then the configuration $(G_{(n+6)\times(n+6)}, D'')$ as follows. The restriction of D'' to the subgrid $G_{n+3\times n+3}$ consisting of the first (topmost) $n+3$ rows and first (leftmost)

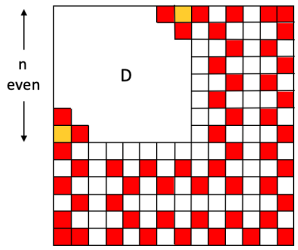


Figure 4.3 – Illustration of Proposition 4.3.4. The vertices are depicted as small squares (with most vertices inside $G_{n\times n}$ as well as their status omitted). Initially infected (resp., not infected) vertices of D'' are red (resp., white). Orange vertices are the ones that are initially infected in D but not in D''.

$n+3$ columns of $G_{(n+6)\times(n+6)}$ will be the set D' minus the vertex $(n+3,3)$. Then, D'' contains the following vertices: $(n+4,2)$ and $(n+4,4)$, $(n+5,1)$ plus the vertices $(n+5,2j+1)$ for every $2 \leq j \leq n/2+2$ (note that $(n+5,3)$ is not infected) and the vertices $(n+6,2j+2)$ for every $0 \leq j \leq n/2+2$. In addition, D'' contains the vertices $(2j+1,n+5)$ and $(2j+2,n+6)$ for every $0 \leq j \leq n/2+1$. Finally, D'' contains the two corners $(n+6,1)$ and $(1,n+6)$. See Figures 4.3 for an illustration. Overall, $|D''| = |D'| + 2n + 12 = D + 4n + 16$.

Now let us prove that D'' is a lethal set. The vertices $(1,n)$ and $(n,1)$ will become infected at the first step because they have 3 infected neighbours initially. Then, every vertex of the subgrid $G_{n\times n}$ will become infected (as D is a lethal set). At the first step, all the vertices of columns $n+2, n+3, n+5$ and $n+6$ become infected except $(n+4,n+2)$, $(n+4,n+3)$ and all the vertices of rows $n+2, n+3, n+5$ and $n+6$ except $(n+2,3)$, $(n+3,3)$, $(n+5,3)$, $(n+6,3)$. Vertices $(n+1,2)$, $(2,n+1)$,$(n+1,2)$ and $(n+4,5)$ are also infected at step 1. From the vertex $(2,n+1)$ the infection propagates along column $n+1$ until it reaches the vertex $(n+1,n+1)$ and then along row $n+1$ until it reaches vertex $(n+1,4)$. Then vertices $(n+1,3)$, $(n+2,3)$, $(n+3,3)$, $(n+4,3)$, $(n+5,3)$, $(n+6,3)$ become infected. From the vertex $(n+4,5)$ the infection propagates along row $n+4$ until it reaches the vertex $(n+4,n+4)$ and then along column $n+4$ until it reaches vertex $(1,n+4)$. See an example in Figure 4.4.

□

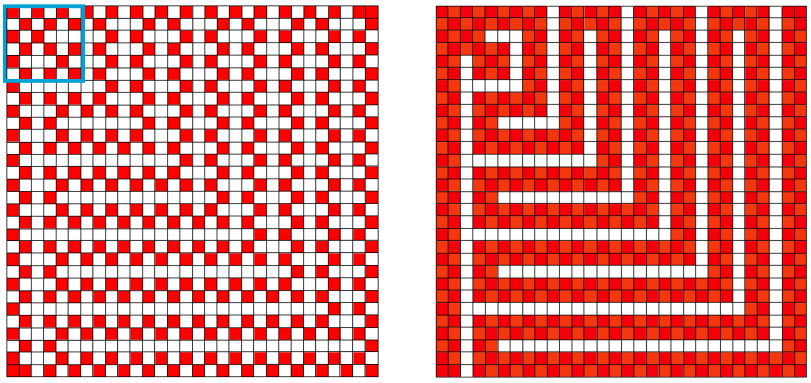
Remark: We can do the proof by deleting from D' instead of vertex $(n+3,3)$ any vertex $(n+3,2j_0+1)$ with $1 \leq j_0 \leq n/2$. Then D'' should contain in row $n+4$ the infected vertices: $(n+4,2j_0)$ and $(n+4,2j_0+2)$ and in row $n+5$ all the vertices $(n+5,2j+1)$ for every $0 \leq j \leq n/2+2$ except vertex $(n+5,2j_0+1)$. That might be useful to diminish the propagation time.

Theorem 4.3.5. *Let n be even. Then, $s_3(G_{n\times n}) = LB_n$. Moreover, there exists an optimal lethal set which contains the vertex $(2,2)$.*

Proof.

The theorem is true for $n = 2, 4, 6$, in which cases, optimal grids can be constructed directly (for $n = 4$ and 6, see Figure 4.1). Then, the theorem follows by induction on n (even) using Proposition 4.3.4 and Corollary 4.2.5. The fact that all the grids contain the vertex $(2,2)$ will be useful for the proof of the existence of optimal tori ($s_3(T_{n\times n}) = \lceil \frac{n^2+1}{3} \rceil$).

□



(a) Infection process at $t = 0$. (b) Infection process at $t = 1$.

Figure 4.4 – Construction for $n = 30$. Here we start from the optimal grid $G_{6\times 6}$ (depicted by a blue frame) obtained from the grid $G_{3\times 3}$ (Theorem 21) and apply the construction recursively. In the left part, initially infected (resp., not infected) vertices are red (resp., white). In the right part, vertices infected at step 1 are in orange and those not infected at the end of step 1 are in white. To get optimal lethal sets for the grids with n even we should also have the corners infected and to get almost optimal for the grids with n odd we should also have the vertex $(n, 3)$ infected.

Theorem 4.3.6. *Let $n \equiv 1$ or $3 \pmod 6$. Then, $s_3(G_{n\times n}) \leq LB_n + 1$. Moreover, there exists a lethal set of size $LB_n + 1$ which contains the vertex $(2, 2)$.*

Proof.

The theorem follows from Theorem 4.3.5 and Corollary 4.3.2.

□

4.3.3 From n to $2n + 1$ and optimality for $n = 2^p - 1$

We give now another tool which, while rather simple, allows us to identify a new class of grids (namely, $n = 2^p - 1$) with odd side where the lower bound is tight.

For any configuration $(G_{n\times n}, D)$, let $(G_{(2n+1)\times(2n+1)}, D') = \mathcal{H}(G_{n\times n}, D)$ be the configuration defined as follows. The restriction of D' to each of the four connected components obtained from $G_{(2n+1)\times(2n+1)}$ by removing its central row and central column (i.e., the $(n+1)^{th}$ row and $(n+1)^{th}$ column) is equal to D. Finally, D' also contains the vertex $(n+1, n+1)$. See Fig. 4.5 for an illustration.

Lemma 4.3.7. *If $(G_{n\times n}, D)$ is a lethal configuration, then $\mathcal{H}(G_{n\times n}, D)$ is a lethal configuration.*

Proof.

Since D is a lethal set for $G_{n\times n}$, each of the component of $G_{(2n+1)\times(2n+1)}$ obtained by removing the central row and the central column will be infected independently. Finally, the central row and central column will be infected using the fact that every other vertex and the central one $(n+1, n+1)$ are infected.

□

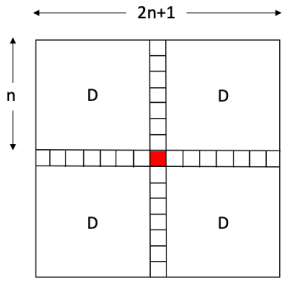


Figure 4.5 – Illustration of the configuration $\mathcal{H}(G_{n\times n}, D)$.

Definition 4.3.1. Let H_1 be the configuration that consists of the grid $G_{1\times 1}$ whose single vertex is infected. For every $p > 1$, let $H_p = \mathcal{H}(H_{p-1})$.

Proposition 4.3.8. *For every $p \geq 1$, H_p is an optimal lethal configuration for the grid $G_{2^p-1\times 2^p-1}$ with $\frac{4^p-1}{3}$ vertices initially infected.*

Proof.

Since H_1 is clearly lethal, by induction on p and Lemma 4.3.7, then H_p is lethal for all p. Let D_p be the lethal set of H_p, then $|D_p| = 4|D_{p-1}| + 1$ and as $|D_1|$= 1, we get $D_p = \frac{4^p-1}{3}$. Finally, the optimality comes from Theorem 4.2.4 that states that for $n = 2^p - 1$, $LB_n = \frac{n(n+2)}{3} = \frac{4^p-1}{3}$.

□

Theorem 4.3.9. *For every $p \in \mathbb{N}^*$, let $n = 2^p - 1$. Then, $s_3(G_{n\times n}) = LB_n = \frac{n(n+2)}{3}$.*

4.4 The lower bound is not always tight

In the case when n is odd such that $n \equiv 1, 3 \pmod 6$ and $n \neq 2^p - 1$, it remains to decide whether $s_3(G_{n\times n})$ equals LB_n or $LB_n + 1$. The first two values which are undetermined yet are $n = 9$ and $n = 13$. For these values, we prove that the lower bound is not tight using a Linear Program. Finally, for $n = 9$, we also provide a combinatorial proof.

4.4.1 Useful properties of lethal sets

First, let us present some useful properties that must be satisfied by any lethal set (or more precisely by the complement of any lethal set) of a grid.

Property 6. *Let D be a lethal set of the grid $G_{n\times n}$. Let $H = V \setminus D$. Then, H induces an acyclic subgraph whose every connected component has at most one vertex on the border.*

Proof.

For purpose of contradiction, let us first assume that H induces a subgraph containing a cycle C and let u be the first vertex of C contaminated by D. Then, when u becomes infected, it has at most two infected neighbours, a contradiction.

Similarly, if H induces a subgraph that contains a path P (with at least two vertices) whose ends are in the border, then the first vertex to be infected in P would have at most 2 already infected neighbours, a contradiction.

□

4.4.2 Linear Program

We present a Mixed-Integer Linear Program (MILP) for computing $s_r(G)$ in any graph G. As initially described in section 4.2, we consider a *step* of the contamination process. That is, at every step, all non-infected vertices with at least r infected neighbours become infected. The question addressed in this section is as follows.

Instance : A graph $G = (V, E)$, non-negative integers r and T.

Question : What is the minimum number $s_r^T(G)$ of initially infected vertices such that the whole graph is contaminated in at most T steps ?

Note that, for every $k \in \mathbb{N}$: $s_r(G) \leq k$ if and only if $s_r^{|V|-k}(G) \leq k$.

We formulate the problem as the MILP below, which computes an optimal solution for $s_r^T(G)$. The main variables are the binary variables $c_{v,t}$ for $v \in V$ and $t \in \{1, ..., T\}$, defined as follows: $c_{v,t} = 1$ if the vertex v is infected after the t^{th} step of the contamination process and 0 otherwise.

We describe hereafter the different constraints (1-6) of the MILP, starting by the objective function:

(1) Minimize the number of initially infected vertices.

(2) After T steps, all the vertices are infected.

(3) An infected vertex remains infected.

(4) A non infected vertex with less than r infected neighbors cannot become infected.

(5) Every vertex with a degree strictly smaller than r must be initially infected (case for instance of the corners of a grid $G_{n\times n}$ when $r = 3$ (Property 4)).

(6) If two adjacent vertices have degree r, then at least one must be initially infected (case for instance of two adjacent vertices of the border of a grid $G_{n\times n}$ when $r = 3$, (Property 5)).

(7) If D is a lethal set and $v \in D$ such that $|N(v) \cap D| \geq r$, then $D \setminus \{v\}$ is a lethal set.

$$\text{Minimize} \sum_{v \in V} c_{v,0} \tag{4.1}$$

$$\text{Subject to} \sum_{v \in V} c_{v,T} = |V| \tag{4.2}$$

$$c_{v,t-1} \leq c_{v,t} \qquad \forall v \in V, t \in \{1, ..., T\} \tag{4.3}$$

$$c_{v,t} \leq c_{v,t-1} + \frac{1}{r} \sum_{w \in N(v)} c_{w,t-1} \qquad \forall v \in V, t \in \{1, ..., T\} \tag{4.4}$$

$$c_{v,0} = 1 \qquad \forall v \in V, d(v) < r \tag{4.5}$$

$$c_{u,0} + c_{v,0} \geq 1 \qquad \forall \{u, v\} \in E, d(u) = d(v) = r \tag{4.6}$$

$$c_{v,0} + \sum_{w \in S} c_{w,0} \leq r \qquad \forall v \in V, d(v) \geq r, \forall S \subseteq N(v), |S| = r \tag{4.7}$$

$$c_{v,t} \in \{0, 1\} \qquad \forall v \in V, t \in \{1, ..., T\} \tag{4.8}$$

We note that the above cuts were not sufficient to make the MILP terminate (in *sufficiently reasonable time*) for $G = G_{n \times n}$ except for some small values of n. Therefore, to speedup the search, we propose further optimization cuts which hold under certain conditions on G.

Further cuts if $r \geq 3$ and and $G = G_{n \times n}$.
Let $r \geq 3$. Consider the graph G' obtained from G by adding a new vertex x adjacent to every vertex of the border of G. Let H' be the subgraph induced by H and x. By Property 6, if D is a lethal set, then $H = V \setminus D$ is acyclic, and moreover, every connected component of H has at most one vertex on the border. Note that H satisfies both these properties if and only if H' is acyclic.

We introduce the next constraints (9-14) to express this property.

First, let $c_{x,0} = 0$ be a new variable expressing that we consider the new vertex x as non infected. Then, we consider the binary variables y_e for $e = \{u, v\} \in E(G')$, defined as follows: $y_e = 1$ if and only if $c_{u,0} = 0$ and $c_{v,0} = 0$ (i.e., $y_e = 1$ if and only if e is an edge of H').

$$y_e + c_{u,0} + c_{v,0} \geq 1 \quad \forall e = \{u, v\} \in E(G') \tag{4.9}$$

$$2y_e + c_{u,0} + c_{v,0} \leq 2 \quad \forall e = \{u, v\} \in E(G') \tag{4.10}$$

$$y_e \in \{0, 1\} \quad \forall e = \{u, v\} \in E(G') \tag{4.11}$$

Then, to ensure that H' is acyclic, we leverage upon an efficient formulation usually used to design a Linear Program for computing the Maximum Average Degree of a graph in polynomial time. To this end, we consider the real non-negative variables $z_{e,u}$ and $z_{e,v}$ for every $e = \{u, v\} \in E(G')$. Intuitively, each edge induced by H' distributes a potential of one from each of its end-vertices, and H' is acyclic if and only if no vertex receives a potential of at least one.

$$z_{e,u} + z_{e,v} = y_e \quad \forall e = \{u, v\} \in E(G') \tag{4.12}$$

$$\sum_{\substack{e' = \{w,v\} \\ w \in N(v)}} z_{e',v} < 1 \quad \forall v \in V(G') \tag{4.13}$$

$$z_{e,u}, z_{e,v} \geq 0 \quad \forall e = \{u, v\} \in E(G') \tag{4.14}$$

Further optimizations. We also provided the MILP with lower and upper bounds on the expected solution. That is, we add constraints of the following form $LB \leq \sum_{v \in V} c_{v,0} \leq UB$ for some

given integers LB and UB. We note that the main purpose of the previous MILP is to determine whether the lower bound LB_n is tight for the cases not covered by Theorem 4.1.1. Therefore, we added some specific constraints which reflect the properties of lethal sets of size LB_n. For instance, when $n \equiv 1$ or 3 (mod 6), by Property 7, every such lethal set D does not contain two adjacent vertices and a vertex in $V \setminus D$ cannot be surrounded by four vertices in D. In some cases, we also imposed some further restrictions (such as, in the case when n is odd, that the vertices in the border of the grids are precisely characterized if we expect a lethal set of size LB_n) or explicitly implemented branch and bound algorithms where we specified the values of some vertices.

Overall, these LPs allowed us to obtain best known solutions for every n up to 20 (but 19). So far, the case $n = 19$ is the smallest size that still resists to the MILP as it provided a solution with $LB_{19} + 1$ vertices initially contaminated, but was not able to terminate when requiring a solution with LB_{19} vertices. This is in contrast with the case when $n = 13$, where the MILP terminates certifying that there are no solutions using LB_{13} vertices initially contaminated. In summary, our experiments based on the MILP allowed us as such to show that $s_3(G_{n\times n}) = LB_n + 1$ for $n \in \{9, 13\}$ (Proposition 4.4.1); but the MILP does not terminate for $n = 19$.

Proposition 4.4.1. $s_3(G_{n\times n}) = LB_n + 1$ *if* $n \in \{9, 13\}$.

Proof.

The upper bounds comes from Theorem 4.3.6. The fact that this is optimal results from the execution of the LP described in this section and that certified that no solutions with LB_n vertices exist.

□

4.5 Tori

In this section, we focus on the square tori. We note that Property 6 proved for the grids holds for any graph. This property was used for tori in [Flocchini et al., 2004] to prove the following lower bound.

Proposition 4.5.1. *[Flocchini et al., 2004]* $s_3(T_{n\times n}) \geq LBT_n = \lceil \frac{n^2+1}{3} \rceil$.

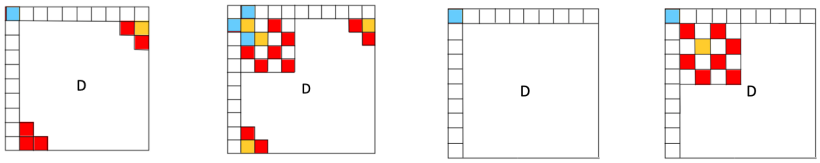
An upper-bound of $\lceil n/3 \rceil (n+1)$ for $s_3(T_{n\times n})$ is also given in [Flocchini et al., 2004]. Note that it differs from the lower bound by a factor of order n (more precisely, $n/3 - 1$ if $n \equiv 0 \pmod 3$, $(2n-1)/3$ if $n \equiv 2 \pmod 3$ and n if $n \equiv 1 \pmod 3$). Here, we determine the exact value of $s_3(T_{n\times n})$ for every $n \geq 3$, as given in Theorem 4.1.2.

Theorem 4.5.2. *If n is odd, then* $s_3(T_{n\times n}) = LBT_n$.

Proof.

As n is odd, $n - 1$ is even. By Theorem 4.3.5, there exists a lethal configuration $(G_{(n-1)\times(n-1)}, D)$ of size LB_{n-1} and such that $(2, 2), \subseteq D$. We build a lethal configuration $(T_{n\times n}, D')$ as follows.

— Let $n \equiv 3$ (mod 6). The restriction of D' to the subgrid $G_{(n-1)\times(n-1)}$ consisting of the last (bottommost) $n - 1$ rows and last (rightmost) $n - 1$ columns of $G_{(n-1)\times(n-1)}$ will be the set D minus the corners $(1, n)$. Moreover, D' contains the vertex $(0, 0)$ (see Figure 4.6(a)). So, $|D'| = |D|$. We have $|D| = LB_{n-1} = \frac{(n-1)^2+2(n-1)+4}{3} = \frac{n^2+3}{3} = LBT_n$.



(a) $n \equiv 3$ [6] (b) $n \equiv 1, 5$ [6] (c) $n \equiv 0$ [6] (d) $n \equiv 2, 4$ [6]

Figure 4.6 – Illustration of Theorem 4.5.2 and Theorem 4.5.3 for the torus $T_{n\times n}$. The vertices are depicted as small squares. Red squares are those of a lethal set D of $G_{(n-1)\times(n-1)}$. Orange squares are deleted from D and not infected in the torus. Blue vertices are extra infected vertices in the torus. In Figure 4.6(b) (resp. 4.6(d)), we start from the optimal grid $G_{4\times4}$ (resp. $G_{6\times6}$) and apply recursively the construction of proposition 4.3.4 (resp. plus proposition 4.3.1).

— Let $n \equiv 1, 5 \pmod 6$. The restriction of D' to the subgrid $G_{(n-1)\times(n-1)}$ consisting of the last (bottommost) $n-1$ rows and last (rightmost) $n-1$ columns of $G_{(n-1)\times(n-1)}$ will be the set D minus the three corners $(1,n),(n,1),(1,1)$ and vertex $(2,2)$. Moreover, D' contains the three vertices $(0,1),(1,0)$ and $(2,1)$ (see Figure 4.6(b)). So, $|D'| = |D| - 1$. We have $|D| = LB_{n-1} = \frac{(n-1)^2+2(n-1)+6}{3} = \frac{n^2+5}{3}$. So, $|D'| = \frac{n^2+2}{3} = LBT_n$.

We claim that D' is lethal in both cases. For the first case, the infection reaches all the vertices of the subgrid $G_{(n-1)\times(n-1)}$ except $(1,n)$ as D is a lethal set. Then, it propagates along line 0 till $(0,n)$ and so $(1,n)$ becomes infected and then the infection propagates along column 0. For the second case, the infection reaches all the vertices of the subgrid $G_{(n-1)\times(n-1)}$; indeed vertices $(1,1),(1,n),(2,2),(2,3),(n,1)$ are infected thanks to $(0,1),(1,0),(2,1)$ and then vertex $(1,2)$ becomes infected. Then the infection propagates along column 0 till $(n,0)$ and so $(0,0)$ becomes infected. Finally the infection propagates along line 0 till $(0,n)$.

□

Theorem 4.5.3. *If n is even, then $s_3(T_{n\times n}) = LBT_n$.*

Proof.

— Let $n \equiv 0 \pmod 6$. So $n-1 \equiv 5 \pmod 6$. By Theorem 4.3.6, there exists a lethal set D in the grid $G_{(n-1)\times(n-1)}$ of size LB_{n-1}. We construct a lethal configuration $(T_{n\times n}, D')$ as follows. The restriction of D' to the subgrid $G_{(n-1)\times(n-1)}$ consisting of the last (bottommost) $n-1$ rows and last (rightmost) $n-1$ columns of $G_{(n-1)\times(n-1)}$ will be the set D. Moreover, D' contains the vertex $(0,0)$. (see Figure 4.6(c)). So, $|D'| = |D| + 1$. We have $|D| = LB_{n-1} = \frac{(n-1)^2+2(n-1)+1}{3} = \frac{n^2}{3}$. So, $|D'| = \frac{n^2+3}{3} = LBT_n$.

— Let $n \equiv 2, 4 \pmod 6$. So $n-1 \equiv 1, 3 \pmod 6$. By Theorem 4.3.6, there exists a lethal set D in the grid $G_{(n-1)\times(n-1)}$ of size $LB_{n-1}+1$ containing vertex $(2,2)$. We construct a lethal configuration $(T_{n\times n}, D')$ as follows. The restriction of D' to the subgrid $G_{(n-1)\times(n-1)}$ consisting of the last (bottommost) $n-1$ rows and last (rightmost) $n-1$ columns of $G_{(n-1)\times(n-1)}$ will be the set D minus the vertex $(2,2)$. Moreover, D' contains the vertex $(0,0)$ (see Figure 4.6(d)). So, $|D'| = |D|$. We have $|D| = LB_{n-1} + 1 = \frac{(n-1)^2+2(n-1)}{3} + 1 = \frac{n^2+2}{3}$. So, $|D'| = |D| = \frac{n^2+2}{3} = LBT_n$.

Similarly to the case where n is odd, we can check that D' is lethal in both cases.

□

4.6 Further Work

The first open problem would be to completely settle the question of determining $s_3(G_{n\times n})$ (either LB_n or LB_{n+1}) for any undetermined n, i.e., for every odd $n > 13$ such that $n \not\equiv 5 \pmod 6$ and $n \neq 2^p - 1$. Note that the first undetermined value is $n = 19$ for which our Linear Program did not terminate. The next graph class to be considered would be d-dimensional grids for $d > 2$. Last but not least, it would be interesting to investigate further the question of the speed of the infection, i.e., determining optimal (in terms of size) lethal sets that infect the whole graph as fast (resp., as slow) as possible.

Neural Network Compression

CHAPTER 5

Random Subset Sum Problem

Contents

5.1 Introduction

In the *Subset Sum Problem (SSP)*, one is given as input a set of n integers $X = \{x_1, x_2, \ldots, x_n\}$ and a target value z, and wishes to decide if there exists a subset of X that sums to z. That is, one is to reason about a subset $S \subseteq [n]$ such that

$$\sum_{i \in S} x_i = z.$$

The special case where z is half of the sum of X is known as the *Number Partition Problem (NPP)*. The converse reduction is also rather immediate. *

Be it in either of these forms, the SSP finds applications in a variety of fields, ranging from combinatorial number theory [Sun, 2003] to cryptography [Gemmell and Johnston, 2001, Kate and Goldberg, 2011]. In complexity theory, the SSP is a well-known NP-complete problem, being a common base for NP-completeness proofs. In fact, the NPP version figures among Garey and Johnson's six basic NP-hard problems [Garey and Johnson, 1979b]. Under certain circumstances, the SSP can be challenging even for heuristics that perform well for many other NP-hard problems [Johnson et al., 1991, Ruml et al., 1996], and a variety of dedicated algorithms have been proposed to solve it [Helm and May, 2018, Bringmann and Wellnitz, 2021, Jin and Wu, 2018, Jin et al., 2021, Esser and May, 2019], including some that employ quantum computing [Bernstein et al., 2013, Helm and May, 2018, Li and Li, 2019]. Nonetheless, it is not hard to solve it in polynomial time if we restrict the input integers to a fixed range [Bellman, 1966]. It suffices to recursively list all achievable sums using the first i integers: we start with $A_0 = \{0\}$ and compute A_{i+1} as $A_i \cup \{a + x_{i+1} \mid a \in A_i\}$. For integers in the range $[0, R]$, the search space has size $\mathcal{O}(nR)$.

Studying how the problem becomes hard as we consider larger ranges of integers (relative to n) requires a randomised version of the problem, the *Random Subset Sum Problem (RSSP)*, where the input values are taken as independently and identically distributed random variables. In this setup, the work [Borgs et al., 2001] proved that the problem experiences a phase transition in its average complexity as the range of integers increases.

The result we approach in this work comes from related studies on the typical properties of the problem. In [Lueker, 1998b] the author proves that, under fairly general conditions, the expected minimal distance between a subset sum and the target value is exponentially small. More specifically, they show the following result.

Theorem 5.1.1 Lueker, 1998. *Let $X_1, \ldots, X_n$ be independent uniform random variables over $[-1, 1]$, and let $\varepsilon \in (0, 1/3)$. There exists a universal constant $C > 0$ such that, if $n \geq C \log(1/\varepsilon)$, then, with probability at least $1 - \varepsilon$, for all $z \in [-1, 1]$ there exists $S_z \subseteq [n]$ for which*

$$\left| z - \sum_{i \in S_z} X_i \right| \leq \varepsilon.$$

That is, a rather small number (of the order of $\log \frac{1}{\varepsilon}$) of random variables suffices to have a high probability of approximating not only a single target z, but all values in an interval.

Even though theorem 5.1.1 is stated and proved for uniform random variables over $[-1, 1]$, it is not hard to extend the result to a wide class of distributions. † With this added generality, the

∗. To find a subset of X summing to z, one only needs to solve the NPP for the set $X \cup \{2z, \sum_{i \in [n]} x_i\}$. By doing so, one of the parts must consist of the element $\sum_{i \in [n]} x_i$ alongside the desired subset.

†. Distributions whose probability density function f satisfies $f(x) \geq b$ for all $x \in [-a, a]$, for some constants $a, b > 0$ (see Corollary 3.3 from [Lueker, 1998b]).

theorem becomes a powerful tool for the analysis of random structures, and has recently proven to be particularly useful in the field of Machine Learning, taking part in a proof of the Strong Lottery Ticket Hypothesis [Pensia et al., 2020b] and in subsequent related works [da Cunha et al., 2022b, Fischer and Burkholz, 2021, Burkholz et al., 2022], and in Federated Learning [Wang et al., 2021].

The simplicity and ubiquity of the SSP has granted the related results a special didactic place. Be it as a first example of NP-complete problem [Garey and Johnson, 1979b], a path to science communication [Hayes, 2002], or simply as a frame for the demonstration of advanced techniques [Mertens, 2001], it has been a tool to make important, but sometimes complicated, ideas easier to communicate. We try to recover some of this essence in the proof we present, which not only attains itself to more accessible tools, but also preserves much of the intuitions behind the theorem.

5.1.1 Comparison to former work

The work [Lueker, 1998b] approaches theorem 5.1.1 by considering the random variable associated to the proportion of the values in the interval $[-1, 1]$ that can be approximated up to error ε by the sum of some subset of the first t variables, $X_1, \dots, X_t$.

After restricting to some specific types of subsets, they proceed to evaluate the expected per-round growth of this proportion, conditioned on the outcomes of $X_1, \dots, X_t$. Their strategy is to analyse this expected increase by martingale theory, which only becomes possible after a non-linear transformation of the variables of interest. Those operations hinder any intuition for the obtained martingale. Nonetheless, a subsequent application of the Azuma-Hoeffding bound [Azuma, 1967] followed by a case analysis leads to the result.

The argument presented here starts in the same direction as the original one, tracking the mass of values with suitable approximations as we reveal the values of the random variables $X_1, \dots, X_n$ one by one. However, we quickly diverge from [Lueker, 1998b], managing to obtain an estimation of the expected growth of this mass without discarding any subset-sum. We eventually restrict the argument to some types of subsets, but we do so at a point where the need for such restriction is clear.

We proceed to directly analyse the estimation obtained, without any transformations. This estimation reveals two expected behaviours: (*i*) as we consider the first variables, the proportion of approximated values is expected to grow very fast; (*ii*) then, after a certain point, this expected growth quickly slows down.

We find that the fast growth portion of the process, (*i*), can be analysed with elementary techniques from the field of randomised algorithms. Then, to tackle (*ii*), we redirect our focus from the proportion of values that posses suitable approximations to that of values that do not. This allows us to leverage the results obtained for (*i*), vastly simplifying the rest of the analysis.

Altogether, we believe our work offers a substantially simpler alternative to the original proof. Moreover, while tools from martingale theory such as Azuma-Hoeffding's inequality are not part of standard Computer Science curricula, our argument makes use of much more elementary results[‡] which should make it accessible enough for an undergraduate course on randomised algorithms.

Finally, a very recent paper [Chen et al., 2022a] on a generalisation of the random subset sum problem contains a proof whose structure bears some resemblance to ours. They define a similar stochastic process and also show that it enjoys exponential growth in expectation up to a certain

‡. Namely, the Intermediate Value Theorem, Markov's inequality, and standard Hoeffding bounds.

threshold. However, they use a different approach to translate this expected behaviour into a probability of fast convergence and, while the second part of their process is analogous to (*ii*), their treatment of it does not resemble our strategy. Still, we remark that while conceiving our proof we were unaware of the existence of [Chen et al., 2022a].

5.2 Our argument

In this section, we provide an alternative argument for proving theorem 5.1.1. It takes shape much like the pseudo-polynomial algorithm we described in the introduction. Leveraging the recursive nature of the problem, we construct a process which, at time t, describes the proportion of the interval $[-1, 1]$ that can be approximated by some subset of the first t variables.

We will show that with a suitable number of uniform variables (proportional to $\log(1/\varepsilon)$) a factor of $1 - \varepsilon/2$ of the values in $[-1, 1]$ can be approximated up to error ε. This implies that any $z \in [-1, 1]$ which cannot be approximated within error ε is at most ε away from a value that can. Therefore it is possible to approximate z up to error 2ε.

5.2.1 Preliminaries

Let $X_1, \ldots, X_n$ be realisations of random variables as in theorem 5.1.1, and, without loss of generality, fix $\varepsilon > 0$. We say a value $z \in \mathbb{R}$ is *ε-approximated at time t* if and only if there exists $S \subseteq [t]$ such that

$$\left| z - \sum_{i \in S} X_i \right| < \varepsilon.$$

For $0 \leq t \leq n$, let $f_t \colon \mathbb{R} \to \{0, 1\}$ be the indicator function for the event "z is ε-approximated at time t". Therefore, we have $f_0 = \mathbb{1}_{(-\varepsilon, \varepsilon)}$, since only the interval $(-\varepsilon, \varepsilon)$ can be approximated by an empty set of values. From there, we can exploit the recurrent nature of the problem: a value z can be ε-approximated at time $t + 1$ if and only if either z or $z - X_{t+1}$ could already be approximated at time t. This implies that for all $z \in \mathbb{R}$ we have

$$f_{t+1}(z) = f_t(z) + (1 - f_t(z))\, f_t(z - X_{t+1}). \tag{5.1}$$

To keep track of the proportion of values in $[-1, 1]$ that can be ε-approximated at each step, we define, for each $0 \leq t \leq n$, the random variable

$$v_t = \frac{1}{2} \int_{-1}^{1} f_t(z) \mathrm{d}z.$$

For better readability, throughout the text we will refer to v_t simply as "the volume."

As we mentioned, it suffices to show that, with high probability, at time n, enough of the interval is ε-approximated (more precisely, that $v_n \geq 1 - \varepsilon/2$) to conclude that the entire interval is 2ε-approximated.

5.2.1.1 Expected behaviour

Our first lemma provides a lower bound on the expected value of v_t.

Lemma 5.2.1. *For all $0 \leq t < n$, it holds that*

$$\mathbb{E}\left[v_{t+1} \,\middle|\, X_1, \ldots, X_t\right] \geq v_t \left[1 + \frac{1}{4}\left(1 - v_t\right)\right].$$

Proof.

The definition of v_t and the recurrence in eq. (5.1) give us that

$$\begin{aligned}
\mathbb{E}\left[v_{t+1} \,\middle|\, X_1, \ldots, X_t\right] &= \mathbb{E}\left[\frac{1}{2}\int_{-1}^{1} f_{t+1}(z)\mathrm{d}z \,\middle|\, X_1, \ldots, X_t\right] \\
&= \int_{-1}^{1} \frac{1}{2}\left(\frac{1}{2}\int_{-1}^{1} f_t(z) + \left(1 - f_t(z)\right) f_t(z - x)\mathrm{d}z\right)\mathrm{d}x \\
&= \frac{1}{2}\int_{-1}^{1} f_t(z)\mathrm{d}z \int_{-1}^{1} \frac{1}{2}\mathrm{d}x + \frac{1}{2}\int_{-1}^{1} \frac{1}{2}\int_{-1}^{1} \left(1 - f_t(z)\right) f_t(z - x)\mathrm{d}z\mathrm{d}x \\
&= v_t + \frac{1}{4}\int_{-1}^{1} \left(1 - f_t(z)\right) \int_{-1}^{1} f_t(z - x)\mathrm{d}x\mathrm{d}z \\
&= v_t + \frac{1}{4}\int_{-1}^{1} \left(1 - f_t(z)\right) \int_{z-1}^{z+1} f_t(y)\mathrm{d}y\mathrm{d}z,
\end{aligned}$$

where the last equality holds by substituting $y = z - x$. For the previous ones we apply basic properties of integrals and Fubini's theorem to change the order of integration.

We now look for a lower bound for the last integral in terms of v_t. To this end, we exploit that, since all integrands are non-negative, for all $u \in [-1/2, 1/2]$ we have

$$\begin{aligned}
\int_{-1}^{1} \left(1 - f_t(z)\right) \int_{z-1}^{z+1} f_t(y)\mathrm{d}y\mathrm{d}z &\geq \int_{u-\frac{1}{2}}^{u+\frac{1}{2}} \left(1 - f_t(z)\right) \int_{z-1}^{z+1} f_t(y)\mathrm{d}y\mathrm{d}z \\
&\geq \int_{u-\frac{1}{2}}^{u+\frac{1}{2}} \left(1 - f_t(z)\right) \int_{u-\frac{1}{2}}^{u+\frac{1}{2}} f_t(y)\mathrm{d}y\mathrm{d}z.
\end{aligned}$$

Both inequalities come from range restrictions: in the first we use that $u \in [-1/2, 1/2]$ implies $[u - 1/2, u + 1/2] \subseteq [-1, 1]$; for the second, we have that $[u - 1/2, u + 1/2] \subseteq [z - 1, z + 1]$ for all $z \in [u - 1/2, u + 1/2]$.

To relate the expression to v_t explicitly, we choose u in a way that the window $[u-1/2, u+1/2]$ entails exactly half of v_t. The existence of such u may become clear by recalling the definition of v_t. To make it formal, consider the function given by

$$h(u) = \frac{1}{2}\int_{u-\frac{1}{2}}^{u+\frac{1}{2}} f_t(y)\mathrm{d}y,$$

and observe that

$$\min\left\{h(-1/2),\, h(1/2)\right\} \leq \frac{v_t}{2}, \qquad \text{and} \qquad \max\left\{h(-1/2),\, h(1/2)\right\} \geq \frac{v_t}{2}.$$

Thus, by the intermediate value theorem (theorem 5.3.1), there exists $u^* \in [-1/2, 1/2]$ for which $h(u^*) = v_t/2$, that is, for which

$$\frac{1}{2}\int_{u^*-\frac{1}{2}}^{u^*+\frac{1}{2}} f_t(y)\mathrm{d}y = \frac{v_t}{2}.$$

Altogether, we can conclude that

$$\begin{aligned}
\mathbb{E}\left[v_{t+1} \,|\, X_1, \ldots, X_t\right] &= v_t + \frac{1}{4}\int_{-1}^{1} (1 - f_t(z)) \int_{z-1}^{z+1} f_t(y)\mathrm{d}y\mathrm{d}z \\
&\geq v_t + \frac{1}{2}\int_{u^*-\frac{1}{2}}^{u^*+\frac{1}{2}} (1 - f_t(z)) \left(\frac{1}{2}\int_{u^*-\frac{1}{2}}^{u^*+\frac{1}{2}} f_t(y)\mathrm{d}y\right)\mathrm{d}z \\
&= v_t + \left(\frac{1}{2} - \frac{v_t}{2}\right)\frac{v_t}{2} \\
&= v_t\left[1 + \frac{1}{4}(1 - v_t)\right].
\end{aligned}$$

□

lemma 5.2.1 tells us that, if v_t were to behave as expected, it should grow exponentially up to $1/2$, at which point $1 - v_t$ starts to decrease exponentially. The rest of the proof follows accordingly, with section 5.2.2 analysing the progress of v_t up to one half, and section 5.2.3 analogously following the complementary value, $1 - v_t$, starting from one half. By building on the results from section 5.2.2, we obtain fairly straightforward proofs in section 5.2.3. Thus, the following subsection comprises the core of our argument.

5.2.2 Growth of the volume up to 1/2

Arguably, the main challenge in analysing the RSSP is the existence of over-time dependencies and deciding how to overcome it sets much of the course the proof will take. Our strategy consists in constructing another process which dominates the original one while being free of dependencies.

Let τ_1 be the first time at which the volume exceeds $1/2$, that is, let

$$\tau_1 = \min\{t \geq 0 : v_t > 1/2\}.$$

We just proved that up to time τ_1 the process v_t enjoys exponential growth in expectation. In the following lemma we apply a basic concentration inequality to translate this property into a constant probability of exponential growth for v_t itself.

Lemma 5.2.2. *Given $\beta \in (0, 1/8)$, let $p_\beta = 1 - \frac{7}{8(1-\beta)}$. For all integers $0 \leq t < \tau_1$ it holds that*

$$\Pr\left[v_{t+1} \geq v_t(1+\beta) \,|\, X_1, \ldots, X_t,\, t < \tau_1\right] \geq p_\beta.$$

Proof.

The result shall follow easily from reverse Markov's inequality (lemma 5.3.3) and the bound from lemma 5.2.1. However, doing so requires a suitable upper bound on v_{t+1} and, while $2v_t$ would serve the purpose, such bound does not hold in general.

We overcome this limitation by fixing t and considering how much v_t would grow in the next step if we were to consider only values ε-approximated at time t that happen to lie in $[-1, 1]$ after being translated by X_{t+1}. Making it precise by the means of the recurrence in eq. (5.1), we define

$$\tilde{v} = \frac{1}{2}\int_{-1}^{1} \left[f_t(z) + (1 - f_t(z))\, f_t(z - X_{t+1}) \cdot \mathbb{1}_{[-1,1]}(z - X_{t+1})\right]\mathrm{d}z.$$

This expression differs from the one for v_{t+1} only by the inclusion of the characteristic function of $[-1,1]$. This not only implies that $\tilde{v} \leq v_{t+1}$, but also that $\tilde{v}$ can replace v_{t+1} in the bound from lemma 5.2.1, since the argument provided there eventually restricts itself to integrals within $[-1,1]$, trivialising $\mathbb{1}_{[-1,1]}$. Moreover, as we obtain $\tilde{v}$ without the influence of values from outside $[-1,1]$, we must have $\tilde{v} \leq 2v_t$. Finally, using that $t < \tau_1$ implies $v_t < 1/2$ and chaining the previous conclusions in respective order, we conclude that

$$\begin{aligned}
\Pr\left[v_{t+1} \geq v_t(1+\beta) \,\middle|\, X_1, \ldots, X_t, t < \tau_1\right] &\geq \Pr\left[\tilde{v} \geq v_t(1+\beta) \,\middle|\, X_1, \ldots, X_t, t < \tau_1\right] \\
&\geq \frac{\mathbb{E}\left[\tilde{v} \,\middle|\, X_1, \ldots, X_t, t < \tau_1\right] - v_t(1+\beta)}{2v_t - v_t(1+\beta)} \\
&\geq \frac{\frac{9}{8}v_t - v_t(1+\beta)}{2v_t - v_t(1+\beta)} \\
&= 1 - \frac{7}{8(1-\beta)},
\end{aligned}$$

where we applied the reverse Markov's inequality in the second step.

□

The previous lemma naturally leads us to look for bounds on τ_1, that is, to estimate the time needed for the process to reach volume $1/2$. As expected, the exponential nature of the process yields a logarithmic bound.

Lemma 5.2.3. *Let t be an integer and given $\beta \in (0, 1/8)$, let $p_\beta = 1 - \frac{7}{8(1-\beta)}$ and*

$$i^* = \left\lceil \frac{\log \frac{1}{2\varepsilon}}{\log(1+\beta)} \right\rceil.$$

If $t \geq \frac{i^*}{p_\beta}$, then

$$\Pr\left[\tau_1 \leq t\right] \geq 1 - \exp\left[-\frac{2p_\beta^2}{t}\left(t - \frac{i^*}{p_\beta}\right)^2\right].$$

Proof.

The main idea behind the proof is to define a new random variable which stochastically dominates τ_1 while being simpler to analyse. We begin by discretising the domain $(0, 1/2]$ of the volume into sub-intervals $\{I_i\}_{1 \leq i \leq i^*}$ as follows:

$$\begin{cases}
I_1 = (0, \varepsilon], \\
I_i = \left(\varepsilon\,(1+\beta)^{i-1}, \varepsilon\,(1+\beta)^{i}\right] \text{ for } 2 \leq i < i^*, \\
I_{i^*} = \left(\varepsilon\,(1+\beta)^{i^*-1}, \frac{1}{2}\right],
\end{cases}$$

where i^* is the smallest integer for which $\varepsilon\,(1+\beta)^{i^*} \geq \frac{1}{2}$, that is,

$$i^* = \left\lceil \frac{\log \frac{1}{2\varepsilon}}{\log(1+\beta)} \right\rceil.$$

Now, for each $i \geq 0$, we direct our interest to the number of steps required for v_t to exit the subinterval I_i after first entering it. By lemma 5.2.2, this number is majorised by a geometric random variable $Y_i \sim Geom(p_\beta)$. Therefore, we can conclude that τ_1 is stochastically dominated by the sum of such variables, that is, for $t \in \mathbb{N}$, we have

$$\Pr[\tau_1 \geq t] \leq \Pr\left[\sum_{i=1}^{i^*} Y_i \geq t\right]. \tag{5.2}$$

Let $B_t \sim \text{Bin}(t, p_\beta)$ be a binomial random variable. For the sum of geometric random variables, it holds that

$$\Pr\left[\sum_{i=1}^{i^*} Y_i \leq t\right] = \Pr\left[B_t \geq i^*\right].$$

Since $\mathbb{E}[B_t] = tp_\beta$, the Hoeffding bound for binomial random variables (lemma 5.3.4) implies that, for all $\lambda \geq 0$, we have

$$\Pr\left[B_t \leq tp_\beta - \lambda\right] \leq \exp\left(-\frac{2\lambda^2}{t}\right).$$

Setting t such that $tp_\beta - \lambda = i^*$, we get

$$\begin{aligned}
\Pr\left[\sum_{i=1}^{i^*} Y_i \geq t\right] &\leq \Pr\left[B_t \leq i^*\right] \\
&\leq \exp\left[-\frac{2\left(tp_\beta - i^*\right)^2}{t}\right] \\
&= \exp\left[-\frac{2p_\beta^2\left(t - \frac{i^*}{p_\beta}\right)^2}{t}\right],
\end{aligned}$$

which holds as long as $\lambda = tp_\beta - i^* \geq 0$, that is, for all

$$t \geq \frac{1}{p_\beta}\left\lceil \frac{\log\frac{1}{2\varepsilon}}{\log(1+\beta)} \right\rceil.$$

Finally, applying this to eq. (5.2) and passing to complementary events, we obtain that

$$\Pr[\tau_1 \leq t] \geq 1 - \exp\left(-\frac{2p_\beta^2\left(t - \frac{i^*}{p_\beta}\right)^2}{t}\right).$$

□

5.2.3 Growth of the volume from 1/2

Here we study the second half of the process: from the moment the volume reaches 1/2 up to the time it gets to $1 - \varepsilon/2$. We do so by analysing the complementary stochastic process, i.e., by tracking, from time τ_1 onwards, the proportion of the interval $[-1, 1]$ that does not admit an ε-approximation. More precisely, we consider the process $\{w_t\}_{t\geq 0}$, defined by $w_t = 1 - v_{\tau_1+t}$.

We shall obtain results for w_t similar to those we have proved for v_t. Fortunately, building on the previous results makes those proofs quite straightforward. We start with an analogous of lemma 5.2.1.

Lemma 5.2.4. *For all $t \geq 0$, it holds that*

$$\mathbb{E}\left[w_{t+1} \,\middle|\, X_1, \ldots, X_{\tau_1+t}\right] \leq w_t \left[1 - \frac{1}{4}\left(1 - w_t\right)\right].$$

Proof.

From the definition of w_{t+1} and lemma 5.2.1, it follows that

$$\begin{aligned}
\mathbb{E}\left[w_{t+1} \,\middle|\, X_1, \ldots, X_{\tau_1+t}\right] &= 1 - \mathbb{E}\left[v_{\tau_1+t+1} \,\middle|\, X_1, \ldots, X_{\tau_1+t}\right] \\
&\leq 1 - v_{\tau_1+t}\left[1 + \frac{1}{4}\left(1 - v_{\tau_1+t}\right)\right] \\
&= w_t - \frac{1}{4} w_t (1 - w_t) \\
&= w_t \left[1 - \frac{1}{4}\left(1 - w_t\right)\right].
\end{aligned}$$

□

Let τ_2 the first time that w_t gets smaller than or equal to ε, that is, let

$$\tau_2 = \min\left\{t \geq 0 : w_t \leq \frac{\varepsilon}{2}\right\}.$$

The following lemma bounds this quantity, in analogy to lemma 5.2.3.

Lemma 5.2.5. *For every $t > 0$, it holds that*

$$\Pr\left[\tau_2 \leq t\right] \geq 1 - \frac{1}{\varepsilon}\left(\frac{7}{8}\right)^t.$$

Proof.

Applying that $1 - w_t = v_{\tau_1+t} > \frac{1}{2}$ to lemma 5.2.4 gives the bound

$$\mathbb{E}\left[w_{t+1} \,\middle|\, X_1, \ldots, X_{\tau_1+t}\right] \leq \frac{7}{8} w_t. \tag{5.3}$$

Moreover, from the conditional expectation theory, for any two random variables X and Y, we have $\mathbb{E}\left[\mathbb{E}\left[X \mid Y\right]\right] = \mathbb{E}\left[X\right]$. From this and eq. (5.3), we can conclude that

$$\begin{aligned}
\mathbb{E}\left[w_t\right] &= \mathbb{E}\left[\mathbb{E}\left[w_t \,\middle|\, X_1, \ldots, X_{\tau_1+t-1}\right]\right] \\
&\leq \frac{7}{8}\mathbb{E}\left[w_{t-1}\right],
\end{aligned}$$

which, by recursion, yields

$$\begin{aligned}\mathbb{E}\left[w_t\right] &\leq \left(\frac{7}{8}\right)^t \mathbb{E}\left[w_0\right] \\ &\leq \frac{1}{2}\left(\frac{7}{8}\right)^t.\end{aligned}$$

Finally, by Markov's inequality (lemma 5.3.2),

$$\begin{aligned}\Pr\left[\tau_2 \geq t\right] &\leq \Pr\left[w_t \geq \frac{\varepsilon}{2}\right] \\ &\leq \frac{2\mathbb{E}\left[w_t\right]}{\varepsilon} \\ &\leq \frac{1}{\varepsilon}\left(\frac{7}{8}\right)^t,\end{aligned}$$

and the thesis follows from considering the complementary event.

□

5.2.4 Putting everything together

In this section we conclude our argument, finally proving theorem 5.1.1. We first prove a more general statement and then detail how it implies the theorem.

Let $\tau = \tau_1 + \tau_2$, the first time at which the process $\{v_t\}_{t\geq 0}$ reaches at least $1 - \varepsilon/2$.

Lemma 5.2.6. *Let $\varepsilon \in (0, 1/3)$. There exist constants $C' > 0$ and $\kappa > 0$ such that for every $t \geq C' \log \frac{1}{\varepsilon}$, it holds that*

$$\Pr\left[\tau \leq t\right] \geq 1 - 2\exp\left[-\frac{1}{\kappa t}\left(t - C'\log\frac{1}{\varepsilon}\right)^2\right].$$

Proof.

Let $\beta = \frac{1}{16}$ and $p_\beta = 1 - \frac{7}{8(1-\beta)} = \frac{1}{15}$. The definition of τ allows us to apply lemmas 5.2.3 and 5.2.5 quite directly. Indeed if, for the sake of lemma 5.2.3, we assume $t \geq \frac{2}{p_\beta}\left\lceil \frac{\log \frac{1}{2\varepsilon}}{\log(1+\beta)}\right\rceil$, we have that

$$\begin{aligned}\Pr\left[\tau \leq t\right] &= \Pr\left[\tau_1 + \tau_2 \leq t\right] \\ &\geq \Pr\left[\tau_1 \leq t/2, \tau_2 \leq t/2\right] \\ &\geq \Pr\left[\tau_1 \leq t/2\right] + \Pr\left[\tau_2 \leq t/2\right] - 1 \\ &\geq 1 - \exp\left[-\frac{p_\beta^2}{t}\left(t - \frac{2}{p_\beta}\left\lceil\frac{\log\frac{1}{2\varepsilon}}{\log(1+\beta)}\right\rceil\right)^2\right] - \frac{1}{\varepsilon}\left(\frac{7}{8}\right)^{t/2} \\ &= 1 - \exp\left[-\frac{1}{15^2 t}\left(t - 30\left\lceil\frac{\log\frac{1}{2\varepsilon}}{\log\frac{17}{16}}\right\rceil\right)^2\right] - \frac{1}{\varepsilon}\left(\frac{7}{8}\right)^{t/2}, \qquad (5.4)\end{aligned}$$

where the second inequality holds by the union bound. The remaining of the proof consists in computations to connect this expression to the one in the statement.
Consider the first exponential term in eq. (5.4). Taking $t \geq \frac{60}{\log \frac{17}{16}} \cdot \log \frac{1}{\varepsilon}$, since $\varepsilon < 1/3$, it follows that

$$\exp\left[-\frac{1}{15^2 t}\left(t - 30\left\lceil\frac{\log\frac{1}{2\varepsilon}}{\log\frac{17}{16}}\right\rceil\right)^2\right] \leq \exp\left[-\frac{1}{15^2 t}\left(t - \frac{60}{\log\frac{17}{16}} \cdot \log\frac{1}{\varepsilon}\right)^2\right].$$

Now, consider the second exponential term in eq. (5.4). It holds that

$$\begin{aligned}\frac{1}{\varepsilon}\left(\frac{7}{8}\right)^{\frac{t}{2}} &= \exp\left[\log\frac{1}{\varepsilon} - \frac{t}{2}\log\frac{8}{7}\right] \\ &\leq \exp\left[\log\frac{1}{\varepsilon} - \frac{t}{15}\right] \\ &= \exp\left[-\frac{1}{15}\cdot\frac{\left(t - 15\cdot\log\frac{1}{\varepsilon}\right)^2}{t - 15\cdot\log\frac{1}{\varepsilon}}\right].\end{aligned}$$

Moreover, for $t \geq 15 \cdot \log\frac{1}{\varepsilon}$,

$$\begin{aligned}\exp\left[-\frac{1}{15}\cdot\frac{\left(t - 15\cdot\log\frac{1}{\varepsilon}\right)^2}{t - 15\cdot\log\frac{1}{\varepsilon}}\right] &\leq \exp\left[-\frac{1}{15t}\left(t - 15\cdot\log\frac{1}{\varepsilon}\right)^2\right] \\ &\leq \exp\left[-\frac{1}{15^2 t}\left(t - \frac{60}{\log\frac{17}{16}}\cdot\log\frac{1}{\varepsilon}\right)^2\right].\end{aligned}$$

Altogether, we have that

$$\exp\left[-\frac{p_\beta^2}{t}\left(t - \frac{2}{p_\beta}\left\lceil\frac{\log\frac{1}{2\varepsilon}}{\log(1+\beta)}\right\rceil\right)^2\right] + \frac{1}{\varepsilon}\cdot\left(\frac{7}{8}\right)^{t/2} \leq 2\exp\left[-\frac{1}{15^2 t}\left(t - \frac{60}{\log\frac{17}{16}}\cdot\log\frac{1}{\varepsilon}\right)^2\right],$$

and the thesis follows by setting $\kappa = 15^2$ and $C' = \frac{60}{\log\frac{17}{16}}$.

□

The expression in the claim of lemma 5.2.6 can be reformulated as

$$\Pr\left[v_t \geq 1 - \frac{\varepsilon}{2}\right] \geq 1 - 2\exp\left[-\frac{1}{\kappa t}\left(t - C'\log\frac{1}{\varepsilon}\right)^2\right];$$

hence, theorem 5.1.1 follows by taking $C \geq 3C'$ and observing that once we can approximate all but an $\varepsilon/2$ proportion of the interval $[-1, 1]$, any $z \in [-1, 1]$ either is ε-approximated itself, or is at most ε away from a value that is, which implies that z is 2ε-approximated.

5.3 Tools

Theorem 5.3.1 Intermediate Value Theorem**.** *Let $g\colon [a,b] \to \mathbb{R}$ be a continuous real-valued function such that $\lambda = \min\{g(a), g(b)\} < \max\{g(a), g(b)\} = \Lambda$. Then, for any value $\delta \in [\lambda, \Lambda]$, there exists a point $a < c_\delta < b$ such that $g(c_\delta) = \delta$.*

Lemma 5.3.2 Markov's inequality**.** *Let X be a non-negative random variable. Then for all $c > 0$, we have*

$$\Pr[X \geq c] \leq \frac{\mathbb{E}[X]}{c}.$$

Lemma 5.3.3 Reverse Markov's inequality**.** *Let X be a random variable such that $X \leq u$ for some constant $u \in \mathbb{R}$. Then for all $c < u$, we have*

$$\Pr[X > c] \geq \frac{\mathbb{E}[X] - c}{u - c}.$$

Proof.

We apply Markov's inequality (Lemma 5.3.2) to the random variable $Y = u - X$. Note that Y is non-negative, since $X \leq u$. We get

$$\Pr[X \leq c] = \Pr[Y \geq u - c] \leq \frac{\mathbb{E}[Y]}{u - c} = \frac{u - \mathbb{E}[X]}{u - c},$$

and the thesis follows.

□

Lemma 5.3.4. *Hoeffding bounds [Dubhashi and Panconesi, 2009] Let $X_1, \ldots, X_n$ be independent random variables such that $\Pr[0 \leq X_i \leq 1] = 1$ for all $i \in [n]$. Let $X = \sum_{i=1}^{n} X_i$ and $\mathbb{E}[X] = \mu$. Then*

(i) for any $\lambda \geq 0$ and $\mu \leq \mu_+$, it holds that

$$\Pr[X \geq \mu_+ + \lambda] \leq \exp\left(-\frac{2\lambda^2}{n}\right);$$

(ii) for any $\lambda \geq 0$ and $0 \leq \mu_- \leq \mu$, it holds that

$$\Pr[X \leq \mu_- - \lambda] \leq \exp\left(-\frac{2\lambda^2}{n}\right).$$

CHAPTER 6

Multidimensional Random Subset Sum Problem

Contents

6.1 Introduction

In the *Random Subset Sum Problem (RSSP)*, given a target value z, an error parameter $\varepsilon \in \mathbb{R}_{>0}$ and n independent random variables $\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_n$, one is interested in estimating the probability that there exists a subset $S \subseteq [n]$ for which

$$\left| z - \sum_{i \in S} \mathrm{X}_i \right| \leq \varepsilon.$$

Historically, the analysis of this problem was mainly motivated by research on the average case of its deterministic counterpart, the classic Subset Sum Problem, and the equivalent Number Partition Problem. These investigations lead to a number of insightful results, mostly in the 80s and 90s [Lueker, 1982, Karmarkar et al., 1986, Lueker, 1998a]. In addition, research on the phase transition of the problem extended to the early 2000s, with interesting applications in statistical physics [Mezard and Montanari, 2009, Borgs et al., 2001, Borgs et al., 2004].

More recently, one of the results on the RSSP has attracted quite some attention. A simplified statement for it would be

Theorem 6.1.1. *Lueker, [Lueker, 1998a] Let $\mathrm{X}_1, \ldots, \mathrm{X}_n$ be i.i.d. uniform random variables over $[-1, 1]$, and let $\varepsilon \in (0, 1)$. There exists a universal constant $C > 0$ such that, if $n \geq C \log_2 \frac{1}{\varepsilon}$, then, with high probability, for all $z \in [-1, 1]$ there exists a subset $S_z \subseteq [n]$ for which*

$$\left| z - \sum_{i \in S_z} \mathrm{X}_i \right| \leq 2\varepsilon.$$

That is, a rather small number (of the order of $\log \frac{1}{\varepsilon}$) of random variables suffices to have a high probability of approximating not only a single target z, but all values in an interval. In fact, this result is asymptotically optimal, since each of the 2^n subsets can cover at most one of two values more than 2ε apart and, hence, we must have $n = \Omega(\log \frac{1}{\varepsilon})$. Also, the original work generalises the result to a wide class of distributions.

Those features allowed Theorem 6.1.1 to be quite successful in applications. In the field of Machine Learning, particularly, many recent works, such as [Pensia et al., 2020a, da Cunha et al., 2021, Fischer and Burkholz, 2021, Burkholz et al., 2022, Ferbach et al., 2022, Wang et al., 2021], leverage this result. We discuss those contributions in more detail in Section 6.2.

In this chapter, we investigate a natural multidimensional generalisation of Theorem 6.1.1. Mainly, we prove

Theorem 6.1.2 Main Theorem**.** *Given $\varepsilon \in (0, 1)$ and $d, n \in \mathbb{N}$, consider n independent d-dimensional standard normal random vectors $\mathbf{X}_1, \ldots, \mathbf{X}_n$. There exists a universal constant $C > 0$ for which, if*

$$n \geq C d^3 \log_2 \frac{1}{\varepsilon} \cdot \left(\log_2 \frac{1}{\varepsilon} + \log_2 d \right),$$

then, with high probability, for all $\boldsymbol{z} \in [-1, 1]^d$ there exists a subset $S_{\boldsymbol{z}} \subseteq [n]$ for which

$$\left\| \boldsymbol{z} - \sum_{i \in S_{\boldsymbol{z}}} \mathbf{X}_i \right\|_\infty \leq 2\varepsilon.$$

Moreover, the approximations can be achieved with subsets of size $\frac{n}{6\sqrt{d}}$.

We believe many promising applications of the RSSP can become feasible with this extension of Theorem 6.1.1 to multiple dimensions. To illustrate this, we consider the *Neural Network Evolution (NNE)* model recently introduced by [Gorantla et al., 2019]. It is natural to wonder whether their model is *universal*, in the sense that, with high probability, it can approximate any dense feed-forward neural network. While applying Theorem 6.1.1 to this end would yield exponential bounds on the required overparameterization, in Section 6.6 we prove the universality of the model within polynomial bounds. To broaden the scope of our result, we additionally provide some useful generalisations in Appendix B. In particular, we extend it to a wide class of distributions, proving an analogous extension to the one [Lueker, 1998a] given for Theorem 6.1.1. Finally, in Sections 6.8 and 6.9 we discuss a discretization of our result and potential applications in the context of nondeterministic random walks.

Organisation of the chapter. After discussing related works in Section 6.2, we present a high level overview of the difficulties posed by the problem and of our proof of Theorem 6.1.2 (Section 6.3). We then introduce our notation in Section 6.4 in preparation for the presentation of our analysis in Section 6.5. We follow up with an application of our result to the NNE model ([Gorantla et al., 2019]) and conclude with some notes on the tightness of our analysis in Section 6.7. Finally, generalisations of our results, further extensions, as well as all omitted proofs can be found in the Appendix.

6.2 Related work

As remarked in the Introduction, the first studies of the RSSP were mainly motivated by average-case analyses of the classic Subset Sum and Number Partition problems [Karmarkar et al., 1986, Lueker, 1982, Lueker, 1998a]. Both can be efficiently solved if the precision of the values considered is sufficiently low relative to the size of the input set. In particular, [Mertens, 1998] applies methods from statistical physics to indicate that this is a fundamental property of the problem: the amount of exact solutions for the randomised version exhibits a phase transition when the precision increases relative to the sample size. The work [Borgs et al., 2001] later confirmed formally the existence of a phase transition. [Lueker, 1982] shows that the median of the minimum error in the RSSP is exponentially small when the target is near the expected sum of the random variables. This work was followed by [Lueker, 1998a], which proves Theorem 6.1.1. Recently, [da Cunha et al., 2022a] provided a simpler alternative to the original proof. The discrete setting of a variant of RSSP has also been recently studied in [Chen et al., 2022b] which proves that an integral linear combination (with coefficients in $\{-1, 0, 1\}$) of the sample variables can approximate a range of target values.

In the last few years, Theorem 6.1.1 has been very useful in studying the *Strong Lottery Ticket Hypothesis*, which states that Artificial Neural Networks (ANN) with random weights are likely to contain an approximation of any sufficiently smaller ANN as a subnetwork. In particular, such claim poses the deletion of connections (pruning) as a theoretically solid alternative to careful calibration of their weights (training). [Pensia et al., 2020a] uses Theorem 6.1.1 to prove the hypothesis under optimal overparameterization for dense ReLU neural networks. [da Cunha et al., 2021] extends this result to convolutional networks and [Ferbach et al., 2022] further extends the latter to the class of equivariant networks. Also, [Burkholz et al., 2022] applies Theorem 6.1.1 to construct neural networks that can be adapted to a variety of tasks with minimal retraining.

6.3 Overview of our analysis

6.3.1 Insights on the difficulty of the problem

In d dimensions, since we need $2^{\Theta(d \log \frac{1}{\varepsilon})}$ hypercubes of radius ε to cover the set $[-1,1]^d$, we need a sample of $\Omega(d \log \frac{1}{\varepsilon})$ vectors to be able to approximate (up to error ε) every vector in $[-1,1]^d$.

On the other hand, having $n = \mathcal{O}(d \log \frac{1}{\varepsilon})$ vectors is enough in expectation. To see it, it is sufficient to consider subsets of the sample with $\frac{n}{2}$ vectors. There are $\binom{n}{n/2} \approx 2^{n-o(n)}$ such subsets, each summing to a random vector distributed as $\mathcal{N}(\mathbf{0}, \frac{n}{2} \cdot \boldsymbol{I}_d)$. Thus, given any $\boldsymbol{z} \in [-1,1]^d$, each of those sums has probability approximately $\varepsilon^d (\frac{n}{2})^{-\frac{d}{2}} = 2^{-d \log \frac{1}{\varepsilon} - \frac{d}{2} \log \frac{n}{2}}$ of being at most ε far from $\boldsymbol{z}$. We can then conclude that the expected number of approximations is $2^{n-o(n)} \cdot 2^{-d \log \frac{1}{\varepsilon} - \frac{d}{2} \log \frac{n}{2}}$, which is still of order $2^{n-o(n)}$ provided that $n \geq Cd \log \frac{1}{\varepsilon}$ for a sufficiently large constant C.

It would thus suffice to prove concentration bounds on the expectation. The technical challenge is handling the stochastic dependency between subsets of the sample, as pairs of those typically intersect, with many random variables thus appearing for both resulting sums. The original proof of Theorem 6.1.1 [Lueker, 1998a] and the simplified one [da Cunha et al., 2022a] address dependencies in similar ways. Both keep track of the fraction of values in $[-1,1]$ that can be approximated by a sum of a subset of the first i random variables, $\mathrm{X}_1, \ldots, \mathrm{X}_i$. Their core goal is to bound the proportional increase in this fraction when an additional random variable X_{i+1} is considered. As it turns out, the *conditional expectation* of this increment can be bounded by a constant factor, regardless of the values of $\mathrm{X}_1, \ldots, \mathrm{X}_i$. Unfortunately, naively extending those ideas to d dimensions leads to an estimation of this increment that is exponentially small in d. It is not clear to the authors how to make the estimation depend polynomially on d without leveraging some knowledge of the actual values of $\mathrm{X}_1, \ldots, \mathrm{X}_i$. In fact, even which kind of assumption on the previous samples could work in this sense is not totally clear.

As for other classical concentration techniques that might appear suitable at first, we remark our failed attempts to leverage an average bounded differences argument [Warnke, 2016]. Specifically, we could not identify any natural function related to the fraction of values that can be approximated, which was also Lipschitz relative to the sample vectors. Moreover, both Janson's variant of Chernoff bound [Janson, 2004] and a recent refinement of it [Wang et al., 2017] seem to capture the stochastic dependence of the subset sums too loosely for our needs.

6.3.2 Our approach

Our strategy to overcome the difficulties highlighted in the previous subsection consists in a second-moment approach.

Unlike the proofs for the single dimensional case, our argument, at first, analyses the probability of approximating a single target value $\boldsymbol{z} \in [-1,1]^d$. To this end, consider a sample of n independent random vectors $\mathbf{X}_1, \ldots, \mathbf{X}_n$ and a family $\mathcal{C}$ of subsets of the sample. Let Y be the number of subsets in $\mathcal{C}$ whose sum approximates $\boldsymbol{z}$ up to error ε.

For a single subset, it is not hard to estimate the probability with which a subset-sum $\sum_{i \in S} \mathbf{X}_i$ lies close to $\boldsymbol{z}$. This allows us to easily obtain good bounds on $\mathbb{E}[\mathrm{Y}]$.

We, then, proceed to estimate the variance of Y, circumventing the obstacles mentioned in the previous section by restricting the analysis to families of subsets with sufficiently small pairwise intersections. While this restriction limits the maximum amount of subsets that are available, a

standard probabilistic argument allows us to prove the existence of large families of subsets with the desired property, ensuring that $\mathbb{E}[\mathrm{Y}]$ can be large enough for our purposes.

For each pair of subsets, S and T, we leverage the hypothesis on the size of intersections to consider partitions $S = S_{\mathrm{A}} \cup S_{\mathrm{B}}$ and $T = T_{\mathrm{C}} \cup T_{\mathrm{B}}$, with S_{A} and T_{C} being large, stochastically independent parts, and the smaller parts S_{B} and T_{B} containing $S \cap T$. The bulk of our analysis then consists in deriving careful bounds on their reciprocal dependencies and consequent contributions to the second moment of Y.

The resulting estimate allows us to apply Chebyshev's inequality to Y, obtaining a constant lower bound on $\Pr[\mathrm{Y} \geq 1]$. That is, with conclude that with at least some constant probability at least one of the subsets yields a suitable approximation of $\boldsymbol{z}$. Finally, we employ a probability-amplification argument in order to apply a union bound over all possible target values in $[-1, 1]^d$.

6.4 Preliminaries

Notation. Throughout the text we identify the different types of objects by writing their symbols in different styles. This applies to scalars (e.g. x), real random variables (e.g. X), vectors (e.g. $\boldsymbol{x}$), random vectors (e.g. $\mathbf{X}$), matrices (e.g. $\boldsymbol{X}$). and tensors (e.g. $\boldsymbol{\mathbf{X}}$). In particular, for $d \in \mathbb{N}$, the symbol $\boldsymbol{I}_d$ represents the d-dimensional identity matrix, where $\mathbb{N}$ refers to the set of positive integers. Let $n \in \mathbb{N}$. We denote the set $\{1, \ldots, n\}$ by $[n]$, and given a set S employ the notation $\binom{S}{n}$ to refer to the family of all subsets of S containing exactly n elements of S. Let $\boldsymbol{x} \in \mathbb{R}^d$. The notation $\|\boldsymbol{x}\|_2$ represents the euclidean norm of $\boldsymbol{x}$ while $\|\boldsymbol{x}\|_\infty$ denotes its maximum-norm. Moreover, given $r \in \mathbb{R}_{>0}$ we denote the set $\{\boldsymbol{y} \in \mathbb{R}^d : \|\boldsymbol{y} - \boldsymbol{x}\|_\infty \leq r\}$ by $\mathcal{B}^d_\infty(\boldsymbol{x}, r)$. We represent the variance of an arbitrary random variable X by σ^2_{X} and its density function by φ_{X}. Finally, the notation $\log(\cdot)$ refers to the binary logarithm. Let $d, n \in \mathbb{N}$ and $\varepsilon \in \mathbb{R}_{>0}$, and consider $\boldsymbol{z} \in [-1, 1]^d$ and n independent standard normal d-dimensional random vectors $\mathbf{X}_1, \ldots, \mathbf{X}_n$. Given $S \subseteq [n]$ we define the random variable

$$\mathrm{Y}_{S,\varepsilon,\boldsymbol{z},\mathbf{X}_1,\ldots,\mathbf{X}_n} = \begin{cases} 1 & \text{if } \|\boldsymbol{z} - \sum_{i \in S} \mathbf{X}_i\|_\infty \leq \varepsilon, \\ 0 & \text{otherwise,} \end{cases}$$

that we represent simply by Y_S when the other parameters are clear from context. Since we are interested in studying families of subsets, we also define, for $\mathcal{C}$ contained in the power set of $[n]$, the random variable

$$\mathrm{Y}_{\mathcal{C},\varepsilon,\boldsymbol{z},\mathbf{X}_1,\ldots,\mathbf{X}_n} = \sum_{S \in \mathcal{C}} \mathrm{Y}_S,$$

which we represent simply as Y.

We control the stochastic dependency among subsets by restricting to families of subsets with small pairwise intersection. While this reduces how many subsets we can be considered, we can use the probabilistic method to prove that large families are still available.

Lemma 6.4.1. *For all $n \in \mathbb{N}$ and $\alpha \in (0, \frac{1}{2})$, there exists $\mathcal{C} \subseteq \binom{[n]}{\alpha n}$ with $|\mathcal{C}| \geq 2^{\frac{\alpha^2 n}{6}}$ such that for all $S, T \in \mathcal{C}$, if $S \neq T$, then*

$$|S \cap T| \leq 2\alpha^2 n.$$

Proof.

If $\alpha \geq \frac{1}{2}$, then $2\alpha^2 n \geq \alpha n$, and the result holds trivially. So we assume $\alpha < \frac{1}{2}$.
Let k be any integer, and let $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, with each C_i drawn uniformly from the collection of subsets of $[n]$ with size αn. Given $i, j \in [k]$, if $i \neq j$, then

$$\begin{aligned}\mathbb{E}\left[|C_i \cap C_j|\right] &= \sum_{a \in [n]} \Pr\left[a \in C_i \cap C_j\right] \\ &= \sum_{a \in [n]} \Pr\left[a \in C_i\right] \cdot \Pr\left[a \in C_j\right] \\ &= \alpha^2 n.\end{aligned}$$

By the multiplicative form of Chernoff bounds (see Lemma 6.10.2), it holds that

$$\Pr\left[|C_i \cap C_j| > 2\alpha^2 n\right] \leq \exp\left(-\frac{\alpha^2 n}{3}\right).$$

Finally, for the event of interest, we have that

$$\begin{aligned}\Pr\left[\bigcap_{i \neq j \in [k]} \left\{|C_i \cap C_j| \leq 2\alpha^2 n\right\}\right] &= 1 - \Pr\left[\bigcup_{i \neq j \in [k]} \left\{|C_i \cap C_j| > 2\alpha^2 n\right\}\right] \\ &\geq 1 - \binom{k}{2} \exp\left(-\frac{\alpha^2 n}{3}\right) \\ &\geq 1 - 2^{\frac{\alpha^2 t}{3}} \cdot \exp\left(-\frac{\alpha^2 n}{3}\right) \qquad (6.1) \\ &> 0,\end{aligned}$$

where in Equation (6.1) we have chosen $k = 2^{\frac{\alpha^2 n}{6}}$.

□

Notice that, while this amount is still exponential, it already imposes $n = \mathcal{O}(\frac{d}{\alpha^2} \log \frac{1}{\varepsilon})$ if we are to approximate all points in $[-1, 1]^d$ up to error ε.

6.5 Proof of the main result

As we frequently consider values relatively close to the origin, approximation of the normal distribution by a uniform one is sufficient for many of our estimations.

Lemma 6.5.1. *Let $d \in \mathbb{N}$, $\varepsilon \in (0, 1)$, $\sigma \in \mathbb{R}_{>0}$, and $\boldsymbol{z} \in [-1, 1]^d$. If $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \cdot \boldsymbol{I}_d)$, then*

$$e^{-\frac{2d}{\sigma^2}} \cdot \frac{(2\varepsilon)^d}{\left(2\pi\sigma^2\right)^{\frac{d}{2}}} \leq \Pr\left[\mathbf{X} \in \mathcal{B}_\infty^d(\boldsymbol{z}, \varepsilon)\right] \leq \frac{(2\varepsilon)^d}{\left(2\pi\sigma^2\right)^{\frac{d}{2}}}.$$

As a corollary, we bound the first moment of the random variable Y.

Corollary 6.5.2. *Given $d, n \in \mathbb{N}$, $\varepsilon \in (0,1)$, and $\alpha \in (0, \frac{1}{2})$, let $\mathbf{X}_1, \dots, \mathbf{X}_n$ be independent standard normal d-dimensional random vectors. Then, for all $\boldsymbol{z} \in [-1,1]^d$ and $\mathcal{C} \subseteq \binom{[n]}{\alpha n}$, it holds that*

$$e^{-\frac{2d}{\alpha n}} \frac{(2\varepsilon)^d |\mathcal{C}|}{(2\pi\alpha n)^{\frac{d}{2}}} \leq \mathbb{E}[\mathrm{Y}] \leq \frac{(2\varepsilon)^d |\mathcal{C}|}{(2\pi\alpha n)^{\frac{d}{2}}}.$$

Proof.

Let $S \in \mathcal{C}$ and, hence, $|S| = \alpha n$. Since $\mathbf{X}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$ for all $i \in [n]$, we have that $\sum_{i \in S} \mathbf{X}_i \sim \mathcal{N}(\mathbf{0}, \alpha n \cdot \boldsymbol{I}_d)$. Therefore, as $\Pr[\mathrm{Y}_S = 1] = \Pr\left[\sum_{i \in S} \mathbf{X}_i \in \mathcal{B}_\infty^d(\boldsymbol{z}, \varepsilon)\right]$, by Lemma 6.5.1, we have that

$$e^{-\frac{2d}{\alpha n}} \frac{(2\varepsilon)^d}{(2\pi\alpha n)^{\frac{d}{2}}} \leq \Pr[\mathrm{Y}_S = 1] \leq \frac{(2\varepsilon)^d}{(2\pi\alpha n)^{\frac{d}{2}}},$$

and we can conclude the thesis by noting that $\mathbb{E}[\mathrm{Y}] = \sum_{S \in \mathcal{C}} \Pr[\mathrm{Y}_S = 1]$.

□

We proceed by estimating the second moment of Y.

Lemma 6.5.3. *Given $d, n \in \mathbb{N}$, $\varepsilon \in (0,1)$, and $\alpha \in (0, \frac{1}{6}]$, let $\mathbf{X}_1, \dots, \mathbf{X}_n$ be independent d-dimensional standard normal random vectors, $\boldsymbol{z} \in [-1,1]^d$, and $\mathcal{C} \subseteq \binom{[n]}{\alpha n}$. If $n \geq \frac{81}{\alpha(1-2\alpha)}$ and any two subsets in $\mathcal{C}$ intersect in at most $2\alpha^2 n$ elements, then*

$$\mathrm{Var}[\mathrm{Y}] \leq \frac{(2\varepsilon)^{2d} |\mathcal{C}|^2}{(2\pi\alpha n)^d} \cdot \left[(1 - 4\alpha^2)^{-\frac{d}{2}} - e^{-\frac{4d}{\alpha n}}\right] + \frac{(2\varepsilon)^d |\mathcal{C}|}{(2\pi\alpha n)^{\frac{d}{2}}}.$$

Proof.

We have

$$\begin{aligned}
\mathrm{Var}[\mathrm{Y}] &= \sum_{S,T \in \mathcal{C}} \mathrm{Cov}[\mathrm{Y}_S, \mathrm{Y}_T] \\
&= \sum_{S,T \in \mathcal{C}} \left(\mathbb{E}[\mathrm{Y}_S \cdot \mathrm{Y}_T] - \mathbb{E}[\mathrm{Y}_S]\,\mathbb{E}[\mathrm{Y}_T]\right) \\
&= \sum_{S,T \in \mathcal{C}} \left(\Pr[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1] - \Pr[\mathrm{Y}_S = 1]\Pr[\mathrm{Y}_T = 1]\right) \\
&= \sum_{S \neq T \in \mathcal{C}} \left(\Pr[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1] - \Pr[\mathrm{Y}_S = 1]^2\right) + \sum_{S \in \mathcal{C}} \Pr[\mathrm{Y}_S = 1]\left(1 - \Pr[\mathrm{Y}_S = 1]\right).
\end{aligned}$$

We shall use Lemma 6.5.1 to estimate $\Pr[\mathrm{Y}_S = 1]$, thus, the core of our argument is to bound the joint probability $\Pr[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1]$. To this end, since $\mathrm{Cov}[\mathrm{Y}_S, \mathrm{Y}_T]$ increases monotonically with $|S \cap T|$, we fix $S, T \in \mathcal{C}$ with $|S \cap T| = 2\alpha^2 n$. Moreover, since Y_S is defined in terms of the max-norm, we can analyse the associated event for each coordinate independently. So, we let $\mathrm{X}_1, \cdots, \mathrm{X}_n \sim \mathcal{N}(0,1)$ and $z \in [-1,1]$.

Consider the partitions $S = S_\mathrm{A} \cup S_\mathrm{B}$ and $T = T_\mathrm{C} \cup T_\mathrm{B}$, with $S_\mathrm{B} = T_\mathrm{B} = S \cap T$, and let

$$\mathrm{A} = \sum_{i \in S_\mathrm{A}} \mathrm{X}_i, \qquad \mathrm{C} = \sum_{i \in T_\mathrm{C}} \mathrm{X}_i, \qquad \mathrm{B} = \sum_{i \in S \cap T} \mathrm{X}_i.$$

In this way, we have $\sum_{i\in S} \mathrm{X}_i = \mathrm{A} + \mathrm{B}$ and $\sum_{i\in T} \mathrm{X}_i = \mathrm{C} + \mathrm{B}$, with A, C independent random variables distributed as $\mathcal{N}(0, \sigma_{\mathrm{A}}^2)$ and $\mathrm{B} \sim \mathcal{N}(0, \sigma_{\mathrm{B}}^2)$, where $\sigma_{\mathrm{A}}^2 = \alpha n(1-2\alpha)$ and $\sigma_b^2 = 2\alpha^2 n$. With this setup, we have,

$$\Pr\left[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1\right] = \Big(\Pr\left[\mathrm{A}+\mathrm{B} \in (z-\varepsilon, z+\varepsilon),\, \mathrm{C}+\mathrm{B} \in (z-\varepsilon, z+\varepsilon)\right]\Big)^d.$$

From the law of total probability, it holds that

$$\begin{aligned}
&\Pr\left[\mathrm{A}+\mathrm{B} \in (z-\varepsilon, z+\varepsilon),\, \mathrm{C}+\mathrm{B} \in (z-\varepsilon, z+\varepsilon)\right] \\
&\quad= \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \Pr\left[\mathrm{A}+x \in (z-\varepsilon, z+\varepsilon),\, \mathrm{C}+x \in (z-\varepsilon, z+\varepsilon)\right] \mathrm{d}x \\
&\quad= \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon),\, \mathrm{C} \in (z-x-\varepsilon, z-x+\varepsilon)\right] \mathrm{d}x \\
&\quad= \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \Big(\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\Big)^2 \mathrm{d}x, \qquad (6.2)
\end{aligned}$$

where the last equality follows from the independence of A and C.
Since A is a normal random variable with 0 average, by Claim B.3, we have that

$$\begin{aligned}
\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \Big(\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\Big)^2 \mathrm{d}x &\le \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \Big(\Pr\left[\mathrm{A} \in (x-\varepsilon, x+\varepsilon)\right]\Big)^2 \mathrm{d}x \\
&= \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \left(\int_{x-\varepsilon}^{x+\varepsilon} \varphi_{\mathrm{A}}(y)\,\mathrm{d}y\right)^2 \mathrm{d}x.
\end{aligned}$$

The hypothesis on n implies that $2\sigma_a^2 \ge 162$, so, by Claim B.4,

$$\begin{aligned}
\left(\int_{x-\varepsilon}^{x+\varepsilon} \varphi_{\mathrm{A}}(y)\,\mathrm{d}y\right)^2 &\le \left[\int_{x-\varepsilon}^{x+\varepsilon} \frac{\exp\left(-\frac{(x+\varepsilon)^2}{2\sigma_{\mathrm{A}}^2}\right) + \exp\left(-\frac{(x-\varepsilon)^2}{2\sigma_{\mathrm{A}}^2}\right)}{2\sqrt{2\pi\sigma_{\mathrm{A}}^2}} \cdot \exp\left(\frac{\varepsilon^2}{2\sigma_{\mathrm{A}}^2}\right) \mathrm{d}y\right]^2 \\
&= \frac{(2\varepsilon)^2}{2\pi\sigma_{\mathrm{A}}^2} \cdot \frac{\exp\left(-\frac{(x+\varepsilon)^2}{\sigma_{\mathrm{A}}^2}\right) + \exp\left(-\frac{(x-\varepsilon)^2}{\sigma_{\mathrm{A}}^2}\right) + 2\exp\left(-\frac{x^2+\varepsilon^2}{\sigma_{\mathrm{A}}^2}\right)}{4} \cdot \exp\left(\frac{\varepsilon^2}{\sigma_{\mathrm{A}}^2}\right) \\
&= e^{\varepsilon^2/\sigma_{\mathrm{A}}^2} \cdot \frac{1}{\sqrt{2}} \cdot \frac{(2\varepsilon)^2}{\sqrt{2\pi\sigma_{\mathrm{A}}^2}} \cdot \frac{\varphi_{\mathrm{A}/\sqrt{2}}(x+\varepsilon) + \varphi_{\mathrm{A}/\sqrt{2}}(x-\varepsilon) + 2e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2} \cdot \varphi_{\mathrm{A}/\sqrt{2}}(x)}{4}.
\end{aligned}$$

Moreover, it holds that

$$\begin{aligned}
&\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \left[\varphi_{\mathrm{A}/\sqrt{2}}(x+\varepsilon) + \varphi_{\mathrm{A}/\sqrt{2}}(x-\varepsilon) + 2e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2} \cdot \varphi_{\mathrm{A}/\sqrt{2}}(x)\right] \mathrm{d}x \\
&\quad= (\varphi_{\mathrm{B}} * \varphi_{\mathrm{A}/\sqrt{2}})(\varepsilon) + (\varphi_{\mathrm{B}} * \varphi_{\mathrm{A}/\sqrt{2}})(-\varepsilon) + 2e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2} \cdot (\varphi_{\mathrm{B}} * \varphi_{\mathrm{A}/\sqrt{2}})(0) \\
&\quad= \varphi_{\mathrm{B}+\mathrm{A}/\sqrt{2}}(\varepsilon) + \varphi_{\mathrm{B}+\mathrm{A}/\sqrt{2}}(-\varepsilon) + 2e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2} \cdot \varphi_{\mathrm{B}+\mathrm{A}/\sqrt{2}}(0) \\
&\quad= \frac{2e^{-\varepsilon^2/\sigma^2_{\mathrm{B}+\mathrm{A}/\sqrt{2}}} + 2e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2}}{\sqrt{2\pi\sigma^2_{\mathrm{B}+\mathrm{A}/\sqrt{2}}}} \\
&\quad\le 4 \cdot \frac{e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2}}{\sqrt{2\pi\sigma^2_{\mathrm{B}+\mathrm{A}/\sqrt{2}}}},
\end{aligned}$$

here $*$ denotes the convolution operation, and the last inequality comes from the hypothesis $\alpha \leq \frac{1}{6}$, which implies that $\sigma^2_{\mathrm{B}+\mathrm{A}/\sqrt{2}} \leq \sigma^2_{\mathrm{A}}$.

Altogether, we have

$$\begin{aligned}\Pr\left[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1\right] &\leq \left(e^{\varepsilon^2/\sigma^2_{\mathrm{A}}} \cdot \frac{1}{\sqrt{2}} \cdot \frac{(2\varepsilon)^2}{\sqrt{2\pi\sigma^2_{\mathrm{A}}}} \cdot \frac{e^{-\varepsilon^2/\sigma^2_{\mathrm{A}}}}{\sqrt{2\pi\sigma^2_{\mathrm{B}+\mathrm{A}/\sqrt{2}}}}\right)^d \\ &= \left(\frac{(2\varepsilon)^2}{2\pi} \cdot \frac{1}{\sqrt{2\sigma^2_{\mathrm{A}}\sigma^2_{\mathrm{B}+\mathrm{A}/\sqrt{2}}}}\right)^d \\ &= \frac{(2\varepsilon)^{2d}}{(2\pi\alpha n)^d} \cdot (1-4\alpha^2)^{-\frac{d}{2}},\end{aligned} \tag{6.3}$$

where the last equality follows from recalling that $\sigma^2_{\mathrm{B}} = 2\alpha^2 n$ and $\sigma^2_{\mathrm{A}} = \alpha n\,(1-2\alpha)$, and, thus, $\sigma^2_{\mathrm{B}+\mathrm{A}/\sqrt{2}} = 2\alpha^2 n + \frac{\alpha t}{2}\,(1-2\alpha)$.

Finally, from this bound and from Lemma 6.5.1 we can conclude that

$$\begin{aligned}\mathrm{Var}\left[\mathrm{Y}\right] &= \sum_{S \neq T \in \mathcal{C}} \left(\Pr\left[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1\right] - \Pr\left[\mathrm{Y}_S = 1\right]^2\right) + \sum_{S \in \mathcal{C}} \Pr\left[\mathrm{Y}_S = 1\right]\left(1 - \Pr\left[\mathrm{Y}_S = 1\right]\right) \\ &\leq \sum_{S \neq T \in \mathcal{C}} \left[\frac{(2\varepsilon)^{2d}}{(2\pi\alpha n)^d} \cdot (1-4\alpha^2)^{-\frac{d}{2}} - \frac{(2\varepsilon)^{2d}}{(2\pi\alpha n)^d} \cdot e^{-\frac{4d}{\alpha n}}\right] + \sum_{S \in \mathcal{C}} \frac{(2\varepsilon)^d}{(2\pi\alpha n)^{\frac{d}{2}}}\left[1 - e^{-\frac{2d}{\alpha n}} \cdot \frac{(2\varepsilon)^d}{(2\pi\alpha n)^{\frac{d}{2}}}\right] \\ &\leq \frac{(2\varepsilon)^{2d}\left|\mathcal{C}\right|^2}{(2\pi\alpha n)^d} \cdot \left[(1-4\alpha^2)^{-\frac{d}{2}} - e^{-\frac{4d}{\alpha n}}\right] + \frac{(2\varepsilon)^d\left|\mathcal{C}\right|}{(2\pi\alpha n)^{\frac{d}{2}}}.\end{aligned}$$

□

Remark 6.5.1 – In the proof of Lemma 6.5.3, after applying the law of total probability it is possible to employ Lemma 6.5.1 to estimate the joint probability. While this simplifies the argument, doing so would ultimately weaken the bound in Theorem 6.5.5 by a factor of d. In fact, in Section 6.7 we argue that the estimation we provide is essentially optimal.

For our next result, recall that the existence of a suitable family of subsets is ensured by Lemma 6.4.1.

Lemma 6.5.4. *Given $d, n \in \mathbb{N}$, $\varepsilon \in (0,1)$, and $\alpha \in (0, \frac{1}{6}]$, let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be independent d-dimensional standard normal random vectors, $\boldsymbol{z} \in [-1,1]^d$, and $\mathcal{C} \subseteq \binom{[n]}{\alpha n}$ with $|\mathcal{C}| \geq 2^{\frac{\alpha^2 n}{6}}$. If any two subsets in $\mathcal{C}$ intersect in at most $2\alpha^2 n$ elements, $\alpha \leq \frac{1}{6\sqrt{d}}$, and*

$$n \geq \frac{144d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right),$$

then

$$\Pr\left[\mathrm{Y} \geq 1\right] \geq \frac{1}{3}.$$

Proof.

By Chebyshev's inequality, it holds that

$$\begin{aligned}\Pr\left[\mathrm{Y} \geq 1\right] &\geq \Pr\left[\left|\mathrm{Y} - \mathbb{E}\left[\mathrm{Y}\right]\right| < \frac{\mathbb{E}\left[\mathrm{Y}\right]}{2}\right] \\ &\geq 1 - \frac{4 \cdot \mathrm{Var}\left[\mathrm{Y}\right]}{\mathbb{E}\left[\mathrm{Y}\right]^2}.\end{aligned}$$

Applying Corollary 6.5.2 and Lemma 6.5.3, we get that

$$\begin{aligned}\frac{4 \cdot \mathrm{Var}\left[\mathrm{Y}\right]}{\mathbb{E}\left[\mathrm{Y}\right]^2} &\leq 4 \cdot \frac{e^{\frac{4d}{\alpha n}} \cdot (2\pi\alpha n)^d}{(2\varepsilon)^{2d}\left|\mathcal{C}\right|^2} \cdot \left[\frac{(2\varepsilon)^{2d}\left|\mathcal{C}\right|^2}{(2\pi\alpha n)^d} \cdot \left[(1-4\alpha^2)^{-\frac{d}{2}} - e^{-\frac{4d}{\alpha n}}\right] + \frac{(2\varepsilon)^d\left|\mathcal{C}\right|}{(2\pi\alpha n)^{\frac{d}{2}}}\right] \\ &= 4 \cdot \left(\frac{e^{\frac{4d}{\alpha n}}}{(1-4\alpha^2)^{\frac{d}{2}}} - 1\right) + \frac{4e^{\frac{4d}{\alpha n}} \cdot (2\pi\alpha n)^{\frac{d}{2}}}{(2\varepsilon)^d\left|\mathcal{C}\right|}.\end{aligned}$$

Since $n \geq \frac{68d}{\alpha}$ and $\alpha \leq \frac{1}{6\sqrt{d}}$, by Claim B.1

$$4 \cdot \left(\frac{e^{\frac{4d}{\alpha n}}}{(1-4\alpha^2)^{\frac{d}{2}}} - 1\right) \leq \frac{1}{2}.$$

Furthermore, as $n \geq \frac{144d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)$, $|\mathcal{C}| \geq 2^{\frac{\alpha^2 n}{6}}$, and $\alpha \leq \frac{1}{6}$, by Claim B.2,

$$\frac{4e^{\frac{4d}{\alpha n}} \cdot (2\pi\alpha n)^{\frac{d}{2}}}{(2\varepsilon)^d\left|\mathcal{C}\right|} \leq \varepsilon.$$

□

Applying an union bound, this we amplify the last lemma to get out main result.

Theorem 6.5.5. *Let $\varepsilon \in (0,1)$ and given $d, n \in \mathbb{N}$ let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be independent standard normal d-dimensional random vectors and let $\alpha \in \left(0, \frac{1}{6\sqrt{d}}\right]$. There exists a universal constant $C > 0$ such that, if*

$$n \geq C\frac{d^2}{\alpha^2}\log\frac{1}{\varepsilon} \cdot \left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right),$$

then, with probability

$$1 - \exp\left[-\ln 2 \cdot \left(\frac{n}{C\frac{d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)} - d\log\frac{1}{\varepsilon}\right)\right],$$

for all $\boldsymbol{z} \in [-1,1]^d$ there exists a subset $S_{\boldsymbol{z}} \subseteq [n]$ for which

$$\left\|\boldsymbol{z} - \sum_{i \in S_{\boldsymbol{z}}} \mathbf{X}_i\right\|_\infty \leq 2\varepsilon.$$

Moreover, this remains true even when restricted to subsets of size αn.

Theorem 6.1.2 follows from Theorem 6.5.5 by setting $\alpha = \frac{1}{6\sqrt{d}}$.

6.6 Application to Neural Net Evolution

In this section, we present an application of our main result on the multidimensional RSSP (see Theorem 6.1.2) to a neural network model recently introduced in [Gorantla et al., 2019].

We first provide a description of their model in a setting relevant to our application. Then, we prove that their model exhibits *universality*; namely, with high probability, it can approximate any neural network within a polynomial overhead in the number of parameters.

6.6.1 The NNE model

The *Neural Net Evolution* (NNE) model [Gorantla et al., 2019] has been recently introduced as an alternative approach to train neural networks based on evolutionary methods. The aim is to provide a biologically inspired alternative to the backpropagation process behind ANNs [Rumelhart et al., 1986, Goodfellow et al., 2016], which happens in evolutionary time, instead of lifetime.

The NNE model is inspired by a standard update rule in population genetics and, in [Gorantla et al., 2019], it is shown to succeed in creating neural networks that can learn linear classification problems reasonably well with no explicit backpropagation.

To define the NNE model, we first need to define random genotypes. Given a vector $\boldsymbol{p} \in [0,1]^n$, a random *genotype* $\boldsymbol{x} \in \{0,1\}^n$ is sampled by setting $x_i = 1$ with probability p_i, independently for each i. Each entry x_i indicates whether or not a *gene* is active.

Then, for each i, a random tensor $\boldsymbol{\Theta}^{(i)} \in \mathbb{R}^{\ell \times d \times d}$ is sampled. In the original version of the model [Gorantla et al., 2019], each entry of the tensor is chosen independently and uniformly at random from $[-1,1]$ with probability β, while is set to 0, otherwise. For the sake of our application, we here consider a natural variant of the model where the entries of the tensor are independently drawn from a standard normal distribution.

Now, given a genotype $\boldsymbol{x} \in \{0,1\}^n$, we define

$$\boldsymbol{\Theta}_{\boldsymbol{x}} = \sum_{i\,:\,x_i=1} \boldsymbol{\Theta}^{(i)}. \tag{6.4}$$

Each genotype is then associated with a *feed-forward neural network*, represented by a weighted complete multipartite directed graph. The graph is formed by layers $\{L_i\}_{i=0}^{\ell}$ of d nodes and two consecutive layers are fully connected via a biclique whose edge weights are determined by the tensor $\boldsymbol{\Theta}_{\boldsymbol{x}}$ in the following manner: for every $i \in [\ell]$, the edge between the j-th node of layer L_{i-1} and the k-th node of layer L_i has weight $(\boldsymbol{\Theta}_{\boldsymbol{x}})_{ijk}$.

Equation (6.4) tells us that if a gene is active then it will give a random contribution to each weight of the genotype network.

The learning process in the NNE model works by updating the genotype probabilities $\boldsymbol{p}$ according to some standard population genetics equations [Bürger, 2000, Chastain et al., 2014]. In [Gorantla et al., 2019], it is proved that the adopted update rule indirectly performs backpropagation and enables to decrease the loss function of the networks.

6.6.2 Universality and RSSP

Let $f\colon \mathbb{R}^d \to \mathbb{R}^d$ be a feed-forward neural network of the form

$$f(\boldsymbol{y}) = \mathbf{W}_\ell\, \sigma(\mathbf{W}_{\ell-1} \ldots \sigma(\mathbf{W}_1\, \boldsymbol{y})), \tag{6.5}$$

where $\mathbf{W}_i \in \mathbb{R}^{d\times d}$ is a weight matrix and $\sigma\colon \mathbb{R}^d \to \mathbb{R}^d$ is the ReLU (Rectified Linear Unit) activation function that converts each coordinate y_i of a given vector $\boldsymbol{y} \in \mathbb{R}^d$ to $y_i \cdot \mathbb{1}_{y_i\geq 0}$.

The restrictions on the weight matrix sizes $d \times d$ aim only to ease presentation and can be adapted to any arbitrary dimensions.

Let us construct a third-order tensor $\boldsymbol{\Theta}_f \in \mathbb{R}^{\ell\times d\times d}$ by stacking the weight matrices $\mathbf{W}_1, \ldots, \mathbf{W}_\ell$. We correspondingly denote f by $f_{\boldsymbol{\Theta}}$. Conversely, every tensor $\boldsymbol{\Theta} \in \mathbb{R}^{\ell\times d\times d}$ is associated with a neural network $f_{\boldsymbol{\Theta}}$ in the form of Equation (6.5) whose corresponding weight matrices are the tensor slices, that is, $\mathbf{W}_m = (\boldsymbol{\Theta})_{\substack{i=m\\ j,k\in[d]}}$ for every $m \in [\ell]$.

We can use Theorem 6.1.2 to prove a notion of universality for the NNE model.

Theorem 6.6.1. *Let $\varepsilon > 0$ and $n, d, \ell \in \mathbb{N}$. Let $\mathcal{F}$ be the class of neural networks $f\colon \mathbb{R}^d \to \mathbb{R}^d$ of the form given in Equation* (6.5) *such that their corresponding tensor satisfies $\max_{ijk}|(\boldsymbol{\Theta}_f)_{ijk}| < 1$. A constant $C > 0$ exists such that, if $n \geq C(\ell \cdot d \cdot d)^3 \log\frac{1}{\varepsilon} \cdot \left(\log\frac{1}{\varepsilon} + \log(\ell \cdot d \cdot d)\right)$, then w.h.p. the tensors $\boldsymbol{\Theta}^{(1)}, \ldots, \boldsymbol{\Theta}^{(n)}$ associated to each gene are such that, for any $f \in \mathcal{F}$, there is a genotype $\boldsymbol{x} \in \{0,1\}^n$ which satisfies*

$$\max_{\substack{i\in[\ell]\\ j,k\in[d]}} \left|(\boldsymbol{\Theta}_f)_{ijk} - (\boldsymbol{\Theta}_{\boldsymbol{x}})_{ijk}\right| < 2\varepsilon.$$

We note that standard techniques (e.g., [Pensia et al., 2020a, da Cunha et al., 2021]) can be used to provide bounds on the approximation of the output of neural networks, as well as translating Theorem 6.6.1 for general network architectures (e.g., convolutional neural networks).

6.7 Tightness of analysis

In Lemma 6.4.1 we prove the existence of a suitable family of subsets via a probabilistic argument, sampling their elements uniformly at random. The same argument also implies that the pairwise intersections of almost all subsets is at least $\frac{\alpha^2 n}{2}$. In the next result, we assume such lower bound and prove that our estimation of the joint probability $\Pr[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1]$ in Lemma 6.5.3 (specifically, in Eq. 6.3), is essentially tight. Namely, the next lemma implies that it is not possible to obtain a high-probability bound on Y in Lemma 6.5.4.

Lemma 6.7.1. *Given $d, n \in \mathbb{N}$, $\varepsilon \in (0,1)$, and $\alpha \in (0, \frac{1}{2})$, let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be independent standard normal d-dimensional random vectors and $\boldsymbol{z} \in [-1,1]^d$. If any two subsets in $\mathcal{C}$ intersect in at least $\frac{\alpha^2 n}{2}$ elements and $n \geq \frac{10}{\alpha(2-\alpha)}$, then*

$$\Pr[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1] \geq \frac{(2\varepsilon)^{2d}}{(2\pi\alpha n)^d} \cdot \left(1 - \frac{\alpha^2}{4}\right)^{-\frac{d}{2}} \cdot \exp\left(-\frac{3d}{\alpha n}\right).$$

We can extend the above result by letting $\boldsymbol{z}$ lie in a wider range. This will be useful for the generalisation section Appendix B.

Remark 6.7.1 – If $\lambda > 1$ and $\boldsymbol{z} \in [-\lambda\sqrt{n}, \lambda\sqrt{n}]^d$, then we have

$$\Pr[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1] \geq \frac{(2\varepsilon)^{2d}}{(2\pi\alpha n)^d} \cdot \left(1 - \frac{\alpha^2}{4}\right)^{-\frac{d}{2}} \cdot \exp\left(-\frac{3\lambda^2 d}{\alpha}\right).$$

6.8 Discrete setting

We believe that it should not be hard to adapt our proof to several discrete distributions, in order to obtain results similar to those discussed in the Related Work section. We also note that our Theorem 6.1.2 already implies an analogous discrete result. Suppose that we quantise our random vectors by truncating them to the $\lfloor \log \frac{1}{\delta} \rfloor$-th binary place, obtaining vectors $\hat{\mathbf{X}}_i$ such that $\|\hat{\mathbf{X}}_i - \mathbf{X}_i\|_\infty < \delta$. For any $\boldsymbol{z} \in [-1,1]^d$, Theorem 6.1.2 guarantees that w.h.p. there is a subset of indices $I \subseteq [n]$ such that $\|\boldsymbol{z} - \sum_{i\in I} \mathbf{X}_i\|_\infty < \varepsilon$ and, hence, by the triangular inequality, $\|\boldsymbol{z} - \sum_{i\in I} \hat{\mathbf{X}}_i\|_\infty < n\delta + 2\varepsilon$. As a special case ($\delta = 2\varepsilon$), we have the following:

Corollary 6.8.1 Discretization of Theorem 6.1.2**.** *Given $d \in \mathbb{N}$, $\varepsilon, \delta \in (0,1)$, let $\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_n$ be independent standard normal d-dimensional vectors truncated to the $\lfloor \log \frac{1}{\delta} \rfloor$-th binary place. There exists a universal constant $C > 0$ such that, if $n \geq Cd^3 \log^2 \frac{1}{\varepsilon}$, then, with high probability, for all vectors $\hat{\boldsymbol{z}}$ with entries in $\{k\delta\}_{\lceil -\frac{1}{\delta} \rceil \leq k \leq \lfloor \frac{1}{\delta} \rfloor}$ there exists a subset $S_{\boldsymbol{z}} \subseteq [n]$ for which*

$$\Big\| \hat{\boldsymbol{z}} - \sum_{i \in S_{\boldsymbol{z}}} \hat{\mathbf{X}}_i \Big\|_\infty \leq 2\varepsilon(n+1).$$

Moreover, the approximation can be achieved with subsets of size $\frac{n}{6\sqrt{d}}$.

6.9 Connection with non-deterministic random walks

Consider a discrete-time stochastic process whose state space is $\mathbb{R}^d$ which starts at the origin. At the first step, the process "branches" in two processes, one of which keeps still, while the other moves by the vector $\mathbf{X}_1$. Recursively, given any time i and any process, at the next time step the process branches in two other processes, one of which keeps still, while the other moves by the vector $\mathbf{X}_{i+1}$. In this setting, when $\mathbf{X}_{i+1}$ are sampled from a standard multivariate normal distribution, our results imply that the resulting process is space filling: the process eventually gets arbitrarily close to each point in $\mathbb{R}^d$. This should be contrasted with the fact that a Brownian motion is transient in dimension $d \geq 3$ [Mörters and Peres, 2010]. The above process can also be interpreted as a multi-dimensional version of nondeterministic walks as introduced in [Panafieu et al., 2019] in the context of the analysis of encapsulations and decapsulations of network protocols, where the i-th N-step is $\{\mathbf{X}_i, \vec{0}\}$.

6.10 Tools

Theorem 6.10.1 Chebyshev's inequality**.** *Let X be a random variable with finite expected value μ and finite non-zero variance σ^2. Then for any real number $k > 0$, it holds that*

$$\Pr\left[|\mathrm{X} - \mu| \geq k\right] \leq \frac{\sigma^2}{k^2}.$$

Lemma 6.10.2. *Chernoff-Hoeffding bounds [Dubhashi and Panconesi, 2009] Let $\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_n$ be independent random variables such that*

$$\Pr\left[0 \leq \mathrm{X}_i \leq 1\right] = 1$$

for all $i \in [n]$. Let $\mathrm{X} = \sum_{i=1}^n \mathrm{X}_i$ and $\mathbb{E}[\mathrm{X}] = \mu$. Then, for any $\delta \in (0,1)$ the following holds:

1. *if* $\mu \leq \mu_+$, *then* $\Pr\left[\mathrm{X} \geq (1+\delta)\mu_+\right] \leq \exp\left(-\frac{\delta^2\mu_+}{3}\right)$;
2. *if* $0 \leq \mu_- \leq \mu$, *then* $\Pr\left[\mathrm{X} \leq (1-\delta)\mu_-\right] \leq \exp\left(-\frac{\delta^2\mu_+}{2}\right)$.

Other Works

CHAPTER 7

NFV & SDN Protection in Network Slicing

Contents

7.1 Introduction

A technology which has gained an important momentum in the networking arena is the Network Function Virtualization (NFV) paradigm. The main idea is to dissociate the dependency on dedicated hardware, such as proprietary software appliances, by allowing network functions such as load balancing, firewall, content filtering and deep packet inspection, to be virtualized and executed on generic servers. As such, Virtual Network Functions (VNFs) can be instantiated and scaled on demand without the need to install new equipment, increasing flexibility to accompany user demands [NFV ISG, 2013]. Another fundamental and arising technology is Software-Defined Networking (SDN) that simplifies network monitoring and management. It enables to decouple the control plane from the data plane, and provides global vision and control of the network [Kim and Feamster, 2013]. The synergy of SDN and NFV gives rise to highly dynamic, programmable and flexible networks where both network infrastructures and resources are shared between the various network services.

Network slices are usually provided in the form of an end-to-end logical network provisioned with a set of isolated virtual resources on a shared physical infrastructure. Specifically, a network slice is composed of a set of VNFs, and each VNF is made up of several Virtual Network Function Components (VNFCs). Some of these components are anticipated to host services which usually operate at high velocity. Therefore, a short-lived network outage can disrupt significantly these slices. Reliable placement of these VNFCs is a vital part of the virtual resources allocation and remains one of the key challenges of network slicing. A viable solution for the placement of VNFCs with respect of network slicing consists in having a dedicated backup instance for each component such that some instances handle the service traffic, while others are used as a protection mechanism against failures of compute servers. Indeed, for resiliency purposes, some VNFCs have to be run in parallel such that different replicas of a same VNFC are concurrently deployed as a dedicated protection mechanism. Such resiliency constraints belong to the class of anti-affinity rules and are often related to the hosts that are anticipated to provision them. In particular, the network infrastructure across Internet Service Providers consists of compute servers made up of different capacity constraints zones as part of the horizontal scaling. An example of such zones are the NUMA zones [Gureya et al., 2020] which are used as a protection mechanism for the host and are practically being deployed as such on physical motherboards.

Our work consists in finding an efficient placement strategy to map the network slices with the infrastructure, while ensuring a dedicated protection. Such protection usually follows some design models according to the pre-provisioning, along with deployment models according to the type of traffic regulating the slice demands. Such slices are, for instance, handled in the OpenStack cloud computing infrastructure in an online fashion without any sorting mechanism. However, giving priority to resolving capacity constraints and ignoring the conflicting aspects of the problem addressed may lead to an overestimation of the necessary number of compute servers.

In this chapter, we propose different algorithms for the reliable allocation of resources for network slicing, and compare them with the placement algorithm used by the main computing engine behind the OpenStack infrastructure. Moreover, we propose a coloring based algorithm which extends over multiple dimensions, while satisfying the dedicated protection mechanism for the components. We also take into consideration the distribution of free resources across the compute servers after the provisioning while devising our algorithms. In particular, we consider the effect of the sorting criteria of the network slice over the total performance.

Symbol	Description
$\mathcal{S}$	Set of compute servers
Z	Set of zones for each compute
Γ	Total number of VNFC instances
$\mathcal{C}$	Set of all VNFCs
c	A component in $\mathcal{C}$
T_c	Number of available replicas of c
$\mathcal{C}_c^k$	The k-th replica of c
f_c^k	Network vector for $\mathcal{C}_c^k$
$R_c \| \| A_c$	Design resiliency scheme for c

TABLE 7.1 – Glossary of notations

The rest of this chapter is organized as follows. We first discuss the relevant research literature in Section 7.2. We then formulate in Section 7.3 the problem statement and describe different algorithms. In Section 7.4, we evaluate our approach over synthetic and real-world network slices and discuss the obtained results. Conclusions are drawn in Section 7.5, together with open questions for future work.

7.2 Related Work

A large number of works has been devoted to the deployment and management of network services. We refer the reader to the surveys in [Herrera and Botero, 2016] and [Mijumbi et al., 2015].

Failure protection. The problem of providing network protection against failures has been studied for different use cases such as bandwidth-optimal recovery placement [Tomassilli et al., 2021] and reliable service function chaining [Tomassilli et al., 2018b]. In [Kong et al., 2017], the problem of distributing VNF replicas between both primary and backup paths while maximizing the availability has been considered, based on a heuristic algorithm. The setting where slicing placement requires multiple network functions, which require a different amount of each resource to process a unit of flow has been studied in [Sallam et al., 2019]. Different from previous studies on failure recovery, we present a resilience mechanism for VNFCs placement to protect against compute server failures where the network slicing is deployed over an infrastructure of NUMA zones.

Vector Bin packing. The classical bin packing (BP) problem has been extensively studied. While for its one dimensional setting, [De La Vega and Lueker, 1981] showed an asymptotic polynomial time approximation scheme (APTAS) which was later improved by [Karmarkar and Karp, 1982], it turned out that there is no APTAS for square packing [Woeginger, 1997]. The approximability of the multidimensional setting of BP has been studied in [Chekuri and Khanna, 2004] and [Bansal et al., 2006], where improved approximation algorithms were designed. Some variants of BP [Jansen, 1999], [Epstein and Levin, 2008], [Epstein et al., 2008] have been studied ever since, where some pair of items are in conflict, i.e. not allowed to be packed together in the same bin. However, a setting under conflicts over an arbitrary number of dimensions has not been considered in previous works.

To the best of our knowledge, a general multidimensional setting under conflict constraints for reliable network slicing has not been explored. We combine both the multidimensional aspect with the capacity and resiliency constraints, and provide fast and efficient combinatorial algorithms.

7.3 Problem statement

7.3.1 Problem definition

In order to deploy a network service, Internet Service Providers (ISPs), Content Delivery Providers (CDPs) and other Telco operators usually buy or rent a technical solution from a vendor. Such solution is composed of a set of VNFs, each VNF being itself a set of VNFCs. Each VNFC requires for its operations a certain amount of virtual CPU and memory. To run the solution, the customer uses an infrastructure composed of compute servers, networking resources, and storage assets. Compute servers, used for illustration in this study, are made up of several zones, each of these zones being characterized by a maximum amount of CPU and memory.

The deployment of VNFs/VNFCs must respect a set of rules. First, since the capacity of a compute server is limited, only a subset of VNFCs can be hosted in any compute. Moreover, each single VNFC cannot be split across different zones. Last, for resiliency purposes, each VNFC is usually subject to an anti-affinity rule which is defined by a design redundancy model. Let us suppose that a certain VNFC has some design scheme denoted by $R||A$. Such a scheme implies that the vendor has decided that R replicas (replica and instance are used interchangeably in this chapter) of the VNFC are needed to handle the traffic, but A additional ones have to be deployed for protection purposes. Alternatively, every R instances must be protected by A backups such that if a single compute server fails, at least R instances of the VNFC always run. Such redundancy protection is therefore designed against single failures of the compute servers.

The main aim is to find the minimal number of compute servers needed to host the VNFs of the network service, by piling up the VNFC instances in these computes, while respecting the anti-affinity rule for the redundant instances. We refer to this problem by VNFC-PROTEC.

We use the notations in Table 7.1 in the rest of this chapter, to make it self-contained.

Network vector. We represent a network service by a set $\mathcal{C}$ of VNFCs, such that an instance $\mathcal{C}_c^k$ from $\mathcal{C}$ denotes the k-th replica of the VNFC c. Moreover, for each component c, let $\alpha(c) = \{\mathcal{C}_c^k, 1 \leq k \leq T_c\}$ denote the set of its available replicas. We associate with c a network vector f_c^k which acts as a summary indicator for the network characteristics required by the VNFC. Since all replicas of a same VNFC are similar, then for each component c fixed, it holds that

$$\forall k, k' \in \alpha(c) : f_c^k = f_c^{k'}.$$

Therefore, we can drop the subscript k for ease of notation and assimilate $f_c = f_c^k$ for each copy $k \in \alpha(c)$ of c.

For the VNFC-PROTEC problem, we consider a three-dimensional network vector $f_c = (f_c^{cpu}, f_c^{ram}, f_c^{res}) \in \mathbb{R}^3$ such that, for each VNFC c, $f_c^{cpu} \in (0, 1]$ is its normalized amount of needed CPU, $f_c^{ram} \in (0, 1]$ is its normalized amount of needed memory and $f_c^{res} \in \mathbb{N}$ is its deployment resiliency factor as described hereafter.

Deployment resiliency factor. Let us consider a VNFC c with a design scheme $R_c||A_c = 3||1$ decided by the vendor. Thus, every three instances of c needed to manage the traffic, must be protected by one additional instance. Moreover, let us assume that there are a total of $T_c = 7$

replicas of the VNFC. It implies that the compute servers in the infrastructure must host seven instances of c such that: 3 instances are protected by one backup and 2 instances are protected by one backup. Therefore, the deployment scheme induced by $(R_c||A_c, T_c)$ would be: $5||\mathbf{2}$. The deployment resiliency factor associated with the aforementioned VNFC c is defined as $f_c^{res} = 2$ from the deployment scheme.

More generally, the *deployment resiliency factor* for a component c under the design scheme $N_c||R_c$ is defined as

$$f_c^{res} = \left\lceil \frac{T_c}{R_c + A_c} \right\rceil .$$

The factor corresponds therefore to the maximum number of replicas for the VNFC c allocable to a single compute server. It captures the anti-affinity rule induced by the traffic for a given VNFC. Note that the *deployment scheme* for a VNFC c can be thus expressed as $T_c - f_c^{res}||f_c^{res}$.

7.3.2 Exact ILP

We describe an exact Integer Linear Program which computes the optimal solution by minimizing the number of hosts needed. The main variables are the boolean variables x_s and z_{kcsz} defined as follows

- $x_s = 1$ if there is a VNFC instance assigned to the compute server $s \in \mathcal{S}$ and 0 otherwise.
- $z_{kcsz} = 1$ if the copy k of the VNFC $c \in \mathcal{C}$ is assigned to the host $s \in \mathcal{S}$ in the zone $z \in Z$ and 0 otherwise.

The objective is to minimize the number of compute servers, i.e.,

$$\min \sum_{s \in \mathcal{S}} x_s \tag{7.1}$$

The maximum capacity for each zone is then

$$\sum_{c \in \mathcal{C}, k \in \alpha(c)} z_{kcsz} \cdot f_c^{cpu} \leq z_C \quad \forall s \in \mathcal{S}, z \in Z \tag{7.2}$$

$$\sum_{c \in \mathcal{C}, k \in \alpha(c)} z_{kcsz} \cdot f_c^{ram} \leq z_R \quad \forall s \in \mathcal{S}, z \in Z \tag{7.3}$$

All instances of each VNFC must be assigned

$$\sum_{s \in \mathcal{S}, z \in Z, k \in \alpha(c)} z_{kcsz} = T_c \quad \forall c \in \mathcal{C} \tag{7.4}$$

The anti-affinity rule for each VNFC is expressed as

$$\sum_{z \in Z, k \in \alpha(c)} z_{kcsz} \leq f_c^{res} \quad \forall c \in \mathcal{C}, s \in \mathcal{S} \tag{7.5}$$

A compute is used if at least one instance is assigned to

$$z_{kcsz} \leq x_s \quad \forall s \in \mathcal{S}, z \in Z, c \in \mathcal{C}, k \in \alpha(c) \tag{7.6}$$

The binary constraints are as follows

$$z_{kcsz} \in \{1,0\} \quad \forall s \in \mathcal{S}, z \in Z, c \in \mathcal{C}, k \in \alpha(c) \tag{7.7}$$

$$x_s \in \{1,0\} \quad \forall s \in \mathcal{S} \tag{7.8}$$

For small sized instances of the VNFC-PROTEC problem, the ILP provides good solutions (see Section 7.4). However, finding an optimal solution requires a prohibitive computation time as the problem is NP-complete. For large size instances, it is even impossible to find feasible solutions using the ILP. We thus propose, in the following, efficient heuristics and a coloring algorithm to solve the VNFC-PROTEC problem.

7.3.3 Lower bounds

In order to boost the subsequent algorithms, we provide a lower bound denoted by L_B on the minimum number of needed compute servers. Let z_C (respectively z_R) denote the maximum amount of virtual CPU cores (respectively memory units) available in a single zone. Since a feasible deployment must satisfy a set of constraints, then it holds that

$$L_B = \max\left(\frac{\sum_c T_c f_c^{cpu}}{|Z| \cdot z_C}, \frac{\sum_c T_c f_c^{ram}}{|Z| \cdot z_R}, \max_c \left\lceil \frac{T_c}{f_c^{res}} \right\rceil\right). \tag{7.9}$$

While the first two quantities are derived from the resource allocation of the VNFCs under the capacity constraints of the computes nodes, the third quantity stems from the anti-affinity deployment constraint.

7.3.4 Heuristics

Slice sorting. Since the network vector is multidimensional, we study the effect of sorting the set of VNFCs that occur within a slice over the placement allocation. A widely used approach in one-dimensional placement algorithms consists in sorting the instances in decreasing order by their size. We thus consider a descending lexicographical ordering rule denoted by $\mathcal{R}$ based on the network vector dimensions: f^{cpu}, f^{ram} and f^{res}. Each of the six possible rules alternates between giving priority to capacity constraints satisfaction and anti-affinity resolving. Let $F_{\mathcal{R}}$ denote the first dimension over which the instances are sorted within the slice according to $\mathcal{R}$.

Compute scheduling. In addition to minimizing the number of compute servers, it is also of relevant importance to allocate the VNFCs evenly across the infrastructure with respect to the resources. We propose different algorithms which determine differently how to dispatch the VNFCs across the computes.

— The ORDERED-VECTOR greedy algorithm extends the First Fit Decreasing approach which was shown to be an efficient fast heuristic for one-dimensional resource placement. It operates by first sorting the slice according to a rule $\mathcal{R}$. Then, at each step, a VNFC is always placed over the first compute server which satisfies both the capacity and resiliency constraints. Furthermore, we increment this algorithm with a swap procedure which first finds the set P_{min} of compute nodes with the minimum number of assigned set $C_{reallocate}$ of VNFCs. Then, it attempts to reallocate randomly each instance c in $C_{reallocate}$ across the computes in $\mathcal{S} \setminus P_{min}$ by swapping c with a random feasible instance among

the already placed instances in $\mathcal{C} \setminus C_{reallocate}$. We obtain a new algorithm denoted by ORDERED-VECTOR-SWAP.

— The LEAST-LOAD algorithm considers a sorted slice according to $\mathcal{R}$. Then, it sorts the compute servers in the decreasing order based on the free amount of available load along $F_{\mathcal{R}}$ where the load of a compute is defined as the total sum of the resource $F_{\mathcal{R}}$ across the instances that it currently hosts. For every VNFC, the instance is placed at the least loaded compute which doesn't break the resiliency constraint after sorting the computes. While the algorithm starts with the predetermined lowerbound in Eq. (7.9), the SLIDING-LEAST-LOAD algorithm (see Algorithm 5) increases the L_B every time L_B computes are not enough to host the network slice.

— We consider the default placement algorithm used by the Nova scheduler, which is the main computing engine behind the OpenStack cloud computing infrastructure, denoted by OS-NOVA. The algorithm is similar to LEAST-LOAD with the following differences. First, the network slice is not sorted. Second, the load of the computes is defined as the average sum of the CPU and RAM resources. We increment OS-NOVA with two variants. While ORDERED-OS-NOVA sorts according to the rule $\mathcal{R}$, HEURISTIC-RANDOM does the same but also chooses randomly a compute from a subset of the least loaded computes by uniformly shuffling a fraction γ of the first sorted computes.

7.3.5 Colored Bin Packing

We model the VNFC-PROTECT problem by a generalized version of bin packing that we denote by COLORED-BIN-PACKING. We consider an undirected graph $G = (V, E)$ where $V = \{1, ..., n\}$ represents the set of items of respective sizes $s_1, ..., s_n$ such that $s_i \in [0, 1]^d$ (for $d > 0$) and E the set of links between the items. The goal is to find a minimum-size partition of the items into independent sets (or *bins*) $B_1, ..., B_m$ of G such that $\|\sum_{i \in B_k} s_i\|_\infty \leq 1$, where $\|.\|_\infty$ denotes the standard l_∞ norm.

When $E = \emptyset$, COLORED-BIN-PACKING reduces to the standard bin packing problem ($d = 1$). Therefore, the problem is NP-complete, APX-hard and no algorithm can achieve a better approximation ratio than $\frac{3}{2}$ (unless P=NP). It comes from the approximation hardness of standard bin packing through a reduction from the partition problem [Garey and Johnson, 1979a]. Moreover, if $\sum_{i \in V} s_i \leq 1$, we obtain the special case of the graph coloring problem, i.e., determining the chromatic number $\chi(G)$, which cannot be approximated within factor $N^{1-\varepsilon}$ for an input of N items, for all $\varepsilon > 0$ (unless P=NP) [Zuckerman, 2006].

Equivalence. For each VNFC c with a given deployment scheme $T_c - f_c^{res} || f_c^{res}$, we dispatch its replicas in $\alpha(c)$ into f_c^{res} partitions where each partition is represented by a clique graph denoted by K_i^c, for $i = 1, ..., f_c^{res}$. The resiliency graph G is therefore a disjoint union of graphs $K_1, ..., K_\Gamma$ where $K_1 = \bigcup_{i=1}^{f_c^{res}} K_i^c$. By making the set of edges E depend on the deployment resiliency factor, we thus directly represent the anti-affinity rules of the VNFCs. In order to solve the COLORED-BIN-PACKING, we leverage upon efficient algorithms for the bin packing tailored for a certain class of graphs. Since the resiliency graph is clearly a perfect graph, it belongs to the class of graphs for which one can find in polynomial time a coloring that uses a minimum number of colors. Therefore, we propose COLORED-APPROX (see Algorithm 6) which operates over multi-dimensional network vectors by using different weighting systems. The main idea is to carefully remove small subgraphs of items which induce problematic instances due to the anti-affinity rule.

Algorithm 5 SLIDING-LEAST-LOAD

Input: VNFC set $\mathcal{C}$, Ordering rule $\mathcal{R}$
Output: A set of compute nodes $\mathcal{S}$ and a mapping of VNFC nodes to compute nodes $m_{\mathcal{R}} : \mathcal{C} \rightarrow \mathcal{S}$

Set L_B according to Eq. (7.9) ▷ Lower bound
$\mathcal{C}_{\mathcal{R}} \leftarrow$ sort $\mathcal{C}$ decreasingly using $\mathcal{R}$
unfeasible = True
while unfeasible **do**
    Add L_B compute nodes to $\mathcal{S}$
    for each VNFC c in $\mathcal{C}_{\mathcal{R}}$ **do**
        Mapped$[c] \leftarrow$ False
        for each compute node s in $\mathcal{S}$ **do**
            for each zone $z(s)$ **do**
                $\beta_z^{cpu} \leftarrow$ Free available CPU in z
                $\beta_z^{ram} \leftarrow$ Free available RAM in z
            $A(s) \leftarrow (\max_{z(s)} \beta_z^{cpu}, \max_{z(s)} \beta_z^{ram})$
        $\mathcal{S} \leftarrow$ sort $\mathcal{S}$ decreasingly per $(A, F_{\mathcal{R}})$
        for each compute node s in $\mathcal{S}$ **do**
            if s can host c **then** ▷ Assignment constraints
                Assign c to s
                Mapped$[c] \leftarrow$ True
                Break
        if not Mapped$[c]$ **then**
            $L_b \leftarrow L_b + 1$ ▷ Slide
            Break
    if Mapped$[c]$ **then**
        unfeasible = False
return $\mathcal{S}$

Weighting. The technique of weights has been used for one-dimensional bin packing problems and produces efficient algorithms [Jansen and Öhring, 1997, man Jr et al., 1996]. Each item i is assigned a weight w_i based on its size s_i and its packing in some fixed solution, such that the number of bins of the algorithm is close to the total sum of weights. We denote by $\mathcal{D}$ the set of network characteristics such as CPU, memory, disk usage.

The COLORED-APPROX algorithm operates in several steps. First, an item i is said to be *large* (resp. *small*) if the quantity $max\{f_i^d, d \in \mathcal{D}\}$ in $[0, 1]$ is larger (resp. smaller or equal) than $1/2$. The set of items V of the resiliency graph G is then partitioned into (H_1, H_2) such that H_1 (resp. H_2) collects the large (resp. small) items. Next, an edge (i, j) between two items i, j with i in H_1 and j in H_2 is added if the following conditions are satisfied:

$$\forall \in \mathcal{D}, f_i^d + f_j^d \leq 1 \text{ and } (i, j) \notin E \tag{7.10}$$

Eq. (7.10) captures both the capacity and resiliency constraints characterizing the situation where two items can be jointly placed in a bin. The edge (i, j) is associated with a weight w_{ij} defined as:

$$w_{i,j} = \mathcal{F}(f_i, f_j) + b_j \tag{7.11}$$

Algorithm 6 COLORED-APPROX

Input: Resiliency graph $G = (V, E)$, Ordering rule $\mathcal{R}$
Output: A set of compute nodes $\mathcal{S}$ and a mapping of VNFC nodes to compute nodes $m_{\mathcal{R}} : V \rightarrow \mathcal{S}$

1: Partition V into $P_H = (H_1, H_2)$ (See Sec. 7.3.5)
2: Set the pairs E_H with w_H according to Eq. (7.10) and (7.11)
3: Create a weighted bipartite graph $\mathcal{B} = (P_H, E_H, w_H)$
4: Find a maximum weight matching $\mathcal{M}$ in $\mathcal{B}$
5: **for each** (c_1, c_2) in $\mathcal{M}$ **do**
6: $\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{c_1, c_2\}$ ▷ Joint placement
7: Collect in M the nodes from $\mathcal{M}$
8: $\mathcal{N} = V \setminus M$ ▷ Components not yet placed
9: Compute a feasible coloring with $\chi_{\mathcal{R}}(G[\mathcal{N}])$ colors
10: $K_\chi \leftarrow$ classes from the obtained coloring
11: **for each** class κ in K **do**
12: $\quad$ Run ORDERED-VECTOR over the items in κ
13: $\quad$ Add the obtained computes to $\mathcal{S}$
14: **return** $\mathcal{S}$

where f_i, f_j are the network vectors, $\mathcal{F}$ is an aggregator function (e.g., maximum, mean, weighted sum) such that it outputs a value in $[0, 1]$, and b_j is a bias term associated with the small item. Several values of b_j were proposed in such a way to classify the items into intervals, then a weight is associated by either giving the same weight to all items in the same interval, or scaling them by a multiplicative factor [Seiden, 2002, Baker and Coffman, 1981]. We adopt the special weight function in [Epstein and Levin, 2008] which has the benefit of not rounding up the size (along each dimension) of an item to the next unit fraction. For each d in $\mathcal{D}$, let p_d be the integer such that $f_j^d \in (\frac{1}{p_d(p_d+1)}, \frac{1}{p_d}]$. The term b_j is set to $\frac{1}{\rho(\rho+1)}$ where $\rho = \max(p_d, d \in \mathcal{D})$. Note that as a special case, when $|\mathcal{D}| = 1$, the network vector is one-dimensional and COLORED-APPROX is a $\frac{5}{2}$-approximation algorithm [Epstein and Levin, 2008].

7.4 Performance Evaluation

In this section, we first present our evaluation setup. Then, we provide a broader assessment of the various solutions proposed under different scenarios.

7.4.1 Evaluation setup

Slide demands. We consider the design and deployment of several slices in the network. Each slice has to implement a set of VNFs where each VNF is composed of a given number VNFCs. Each VNFC requires a specific amount of CPU, memory, and has anti-affinity constraints. Our aim is to evaluate the performance of the different placement algorithms (see Section 7.3) over the resource allocation of a given slice across an infrastructure. The latter is made up of a given number of compute servers where each compute is composed of a set of zones with a given CPU and memory capacities.

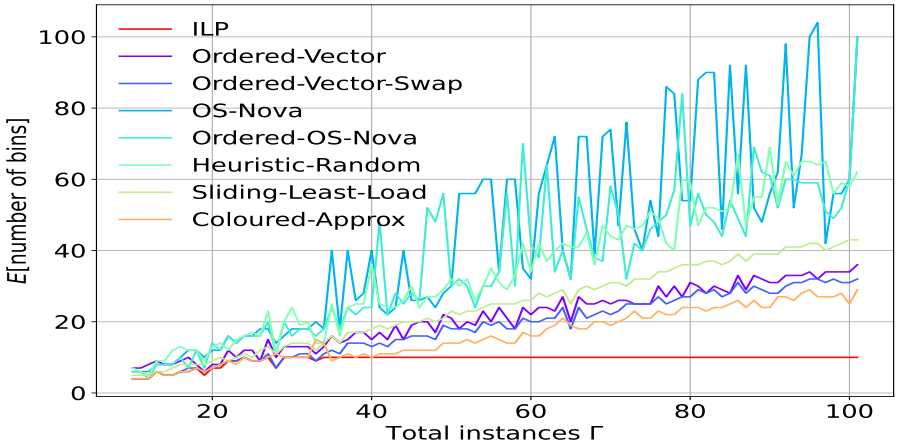


Figure 7.1 – Performance in terms of bins.

Metrics. We compare the different algorithms in terms of the expected number of bins obtained, the running time, and the distribution of free resources (referred to by wastage) across the different bins. Since the number of bins varies across the algorithms under the same slice, to analyze the wastage, we consider the variation coefficient C_v which is the ratio of the standard deviation to the mean and captures the level of dispersion around the mean of free resources of the compute servers after the placement.

7.4.2 Evaluation scenario

Generation. We run our experiments over a synthetic scenario based on a random generator of network slices. For a fixed number of zones (between 1 and 5), the generator provides a network slice made up from a random number of VNFCs. A VNFC has a given cost determined by its network vector where f^{cpu}, f^{ram} are selected uniformly at random between 0 and 1, the number of replicas T_c (which comprises the backup instances) randomly between 1 and 10, and the resiliency factor f^{res} randomly between 1 and T_c. Therefore, a given slice is made up of σ instances where $\sigma = \sum_{c=1}^{\Gamma} T_c$.

We also validate our results over a real scenario based on traces from a real ISP network.

Settings. We set $\gamma = 0.25$ and the time limit for the ILP solver to 10 hours. Moreover, we chose among two families of aggregators: $\mathcal{F}^d_{mean}(f_i, f_j) = \frac{f^d_i + f^d_j}{2}$ for each dimension d and $\mathcal{F}_{mean}(f_i, f_j) = \frac{\sum_d f^d_i + f^d_j}{4}$, in addition to $\mathcal{F}^d_{max}(f_i, f_j) = \max(f^d_i + f^d_j)$ for each dimension d and $\mathcal{F}_{max}(f_i, f_j) = \max(\{f^d_i, f^d_j, d \in \mathcal{D}\})$.

Results. As shown in Fig. 7.1 and Fig. 7.2, only a limited subset of our instances have been submitted to the ILP solver, which turned out to need a generous time-limit of 9 hours to be able to solve optimally a mere subset of 34 instances (the starting point of the red straight lines indicate the moments where we stopped taking into account the ILP algorithm). Therefore, the ILP does not scale and becomes quickly irrelevant as the size of the slice increases. Fig. 7.2 shows that, unlike the ILP, all the various algorithms remain very fast as the difficulty of the network slices increases.

We note that the quality of OS-NOVA regularly deteriorates in terms of bins. Moreover, ORDERED-OS-NOVA being relatively better than OS-NOVA shows the importance of sorting the

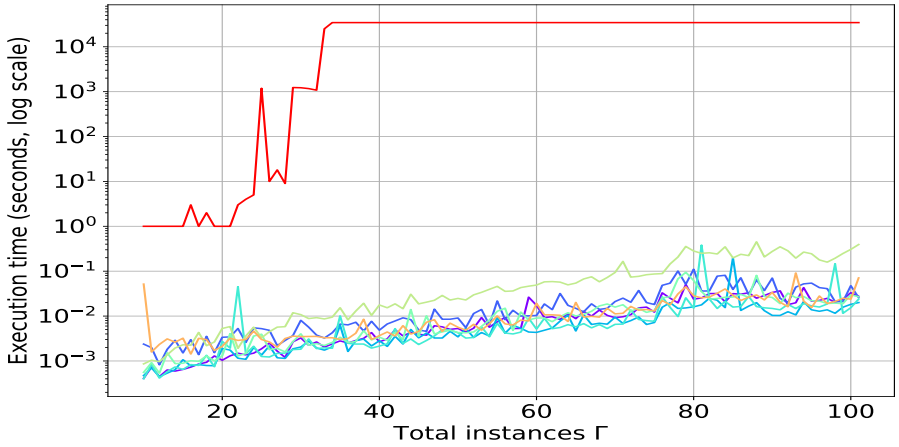


Figure 7.2 – Performance in terms of running time.

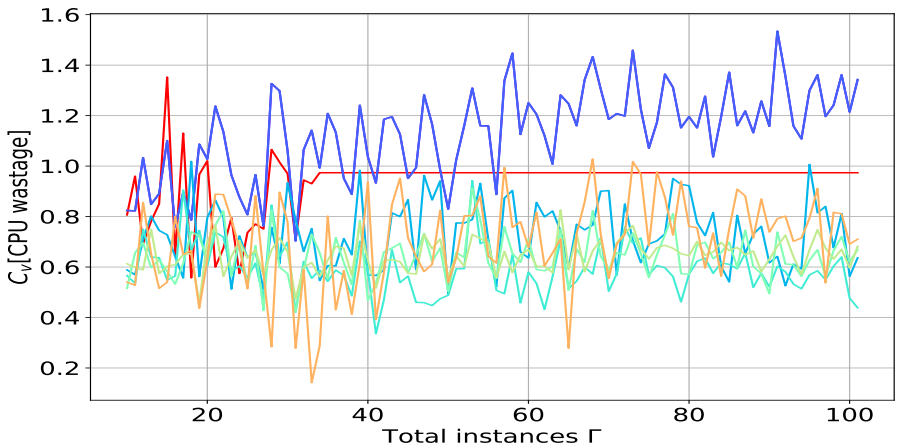


Figure 7.3 – Performance in terms of CPU wastage.

slice before the placement. On top of the VNFCs sorting, shuffling the top least loaded compute nodes provides an apparent improvement as demonstrated by HEURISTIC-RANDOM. This is explained by the fact that in the default version of OS-NOVA, the compute servers are sorted evenly by f^{cpu} and f^{ram}. However, the pre-sorting of the slice implies that sorting the computes by $F(\mathcal{R})$ of the ordering rule is more beneficial as SLIDING-LEAST-LOAD unveils. The latter records quite honorable results and, as expected, spreads the components evenly across the compute nodes (Fig. 7.3 considers the CPU as in the given slice $F(\mathcal{R}) = \text{CPU}$). While the algorithm remains fast, it is outperformed by the others (aside the ILP). This shows that even an overestimation L_B of the necessary number of compute notes does not affect the minimization problem of the least load paradigm.

As expected, the swap procedure improves over ORDERED-VECTOR while remaining fast. On the other hand, while COLORED-APPROX and ORDERED-VECTOR-SWAP obtain the best performances in bins, the coloring approach spreads the wastage evenly in contrast with the ORDERED-VECTOR-SWAP which obtains the worst performance with a relatively high C_v. We recall that in the one-dimensional case, the latter provides a guaranteed theoretical approximation perfor-

mance. The benefit of the weighting techniques lies in not giving the priority to resolving capacity constraints, but also taking into account in a joint manner the anti-affinity rule. We observed that the scenario studied under the real ISP traces provided a similar assessment with, in particular, the scalability and efficiency of the COLORED-APPROX as the number of zones increases. Last, we note that when the matching is empty, COLORED-APPROX becomes equivalent to ORDERED-VECTOR with the major difference that making the feasible coloring over the resiliency graph depend on $\mathcal{R}$ ensures that the distribution of wastage remains reasonable.

7.5 Conclusion and Future work

In this chapter, we presented a coloring based approach to carry out an efficient placement of network slices under a general protection mechanism which preserves failure tolerance of the virtual components. Our experiments over synthetic and real traces show that the proposed algorithms outperform the default placement algorithm in OpenStack while remaining very fast. A future work will investigate the asymptotic approximation performance of our algorithms and adapt them under an online provisioning of network slices.

CHAPTER 8

Resource Allocation with Deep RL

Contents

8.1 Introduction

To meet the growing and increasingly diverse demands of users and companies, networks have evolved, adopting new technologies to make their management more efficient and more responsive to dynamic changes in traffic. The first fundamental evolution to the management of modern networks is Software Defined Networking (SDN). SDN is a network paradigm that decouples the control plane from the data plane and enables centralized control of the network. The network becomes programmable and routing rules can be adjusted in real time leading to a better responsiveness of network management in case of traffic changes.

Network Function Virtualization (NFV) comes with SDN as a paradigm allowing the decoupling of network functions from the hardware. Functions can be virtualized on generic servers located in data centers dispersed all over the network. NFV allows first of all the reduction of capital expenditure (Capex) by avoiding the purchase of a dedicated equipment for each function. It also allows a reduction of operational costs (Opex) since a function can be easily stopped when not used. Finally, NFV allows a more flexible network management since functions can be instantiated on demand anywhere in the network when needed due to traffic dynamics.

By combining SDN and NFV technologies, network management thus is greatly enhanced by making the network programmable, dynamic, and flexible, and by allowing for controlled sharing of resources between different services and users. The increasing importance of wireless networks and the emergence of 5G bring out new needs such as massive device connectivity, high mobility and a great diversity in the quality of service (QoS) requirements. Network slicing has been proposed to meet this challenge and to satisfy these diversified service needs. By dividing the network infrastructure into multiple logical isolated networks, network slicing allows the support of a wide range of communication scenarios with a diversified set of service demands, requirements, and performance. To meet a demand, a slice needs to fulfill an end-to-end service which requires joint allocation of different types of resources. A slice must be deployed in real time and, thus, the corresponding provisioning of network, computing, and storage resources has to be done dynamically.

In 5G networks, traffic is considered to be highly dynamic and network requests may be subject to frequent changes such as arrivals and departures. This dynamicity may fragment the slice resource usage and make the use of network resources less efficient. To counter this effect, network operators need to regularly reconfigure the network slices. Indeed, thanks to SDN and NFV, the routing of flows and the allocation of the network functions can be easily modified. The slice allocation can thus be adjusted in order to reduce the resource utilization with the goal of minimizing operational costs.

In this work, *we propose a method to efficiently reconfigure the network without breaking the flows in order to avoid the interruptions of traffic*. We use a *make-before-break* mechanism in which we first allocate the resources for a secondary route while keeping the first one intact (i.e., two redundant routes are reserved in parallel). Then, we migrate the flow to the new route and release the resources of the old one. There is no disruption of the traffic and, therefore, no impact on the Quality of Service (QoS) of the slices. The computations of the new routes is done by using a *scalable optimization decomposition method making use of a column generation approach*.

Even when using such a mechanism to avoid degrading the QoS, network operators do not want to reconfigure their network too frequently, as it may lead to additional management costs. On the opposite, reconfiguring too rarely during the day may lead to a sub-optimal network usage. A simple policy with low computational cost is to regularly reconfigure after a fixed number

of minutes. However, reconfiguring in response to variation of traffic can reduce the number of reconfigurations required each day without impacting the overall improvement obtained.

In this chapter, we propose a *reconfiguration management agent* that chooses when to initiate reconfiguration as a function of different parameters such as the traffic dynamics and the level of network congestion. We use a *Deep Reinforcement Learning technique* (DRL) by implementing it with Tensorflow [Abadi et al., 2015] and their Deep Q-learning Network (DQN) agent. We then show that our agent improves the efficiency of reconfigurations by performing less reconfigurations while still minimizing the network operational costs compared to doing periodic and frequent reconfigurations.

The rest of this chapter is organized as follows. In Section 8.2, we discuss related work. Section 8.3 presents the formal definition of our problem, Section 8.4 the column generation model, and Section 8.5 the optimization model `Deep-REC` based on reinforcement learning. In Section 8.7, we validate `Deep-REC` by various numerical results. Finally, we draw our conclusion in Section 8.8.

8.2 Related Work

Routing and provisioning of slices. The VNF allocation and SFC provisioning problems have been widely studied in recent years. Some works focus on static scenarios such as [Huin et al., 2018, Tomassilli et al., 2018a] in which the authors develop efficient methods coupling chained allocation of VNFs and traffic routing within SFCs. The dynamic nature of network traffic raises a range of problems concerning the acceptance of incoming slices, resource management, and Service Level Agreement compliance. Cheng *et al.* [Cheng et al., 2020] use a method to deploy and manage slice provisioning using deep learning and Lyapunov stability theories. In [Harutyunyan et al., 2019], the authors present a Mixed Integer Linear Program and a heuristic to add new slices by minimizing the bandwidth consumption and the slice provisioning cost while taking into account the VNF migrations. In [Sharma et al., 2020], a method is presented to manage the creation, modification or deletion of slices by adapting to the traffic. Their goal is to minimize the number of slices while having enough bandwidth available to serve the traffic.

Reconfiguration using standard techniques. The reconfiguration of SFCs and/or slices aims to maintain a near-optimal state of the network over time in order to optimize the network usage and the acceptance of demands. Wang *et al.* [Wang et al., 2019] develop an algorithm that manages two types of reconfiguration to maximize the operator's profits. First, a reconfiguration to adapt the slices to the current traffic. Second, a reconfiguration modifying the flows traversing the slices. The algorithm then schedules the reconfigurations and reserves resources for future traffic to reduce the number of potential future reconfigurations. Each reconfiguration includes the service interruption and resource usage as costs. In [Pozza et al., 2020] authors proposed a slice reconfiguration technique in which the new state of the network is pre-computed. The reconfiguration is done in several steps in which the VNFs and routes are modified while taking into account capacities and delays. In [Gausseran et al., 2021], authors proposed an integer linear program and an heuristic to efficiently reconfigure Service Function Chains using a *make-before-break* strategy. In [Gausseran et al., 2021], column generation techniques are used to optimize the reconfiguration of hundreds of network slices in only few seconds.

Learning-based reconfiguration Some recent works use reconfiguration techniques based on reinforcement learning and try to predict the dynamicity of the network. Liu *et al.* [Liu et al.,

2020] propose a VNF migration strategy based on Double-Deep Q-Network. Their goal is to equally place VNFs between Mobile Edge clusters and core clouds in order to avoid congestion at the Edge. The migration takes into account future traffic and tries to reduce the number of migrations while minimizing the number of congested links. In [Troia et al., 2019] the authors use reinforcement learning to perform dynamic SFC resource allocation in optical networks. They define an agent that decides for each SFC when and which reconfiguration to perform (migration, scaling-up, scaling-down) in order to meet the QoS criteria. Each SFC has a bound on the maximum number of authorized reconfigurations and the optimization objective is the total number of reconfigurations. Each reconfiguration has a penalty for service interruption.

In [Wei et al., 2020] the authors use DRL to predict when to reconfigure in order to minimize the resources consumed. Unlike our work, the authors focus on intra-slice reconfigurations with a fixed number of requests throughout the experiment and with unchanging service sources and destinations. Their algorithm optimises slices only locally and uses a pre-computed set of paths using Depth-first search algorithm. Guan *et al.* [Guan et al., 2020] use a Markov Renewal Process to predict changes in the resource occupancy of slices and reserve resources for slices that obtain higher revenues at lower cost. They use Deep dueling neural network combined with Q-Learning to choose for each slice whether to reconfigure it or not. Their goal is to maximize long term revenue: the increased user utility minus a cost for the resource utilization and the service interruptions. Similar to the last two works mentioned, we establish an agent based on DRL to choose when to reconfigure the slices.

To the best of our knowledge, we are the first to propose a methodology to reconfigure a dynamic network with incoming and outgoing slices and not being dependent on a fixed number of slices throughout the day. Our deep reconfiguration learning algorithm adapts its behavior based on the variations of the whole network traffic and not to a fixed set of flows within a slice. Moreover, we do not fix a limit of the reconfigurations per slice or per day. The agent determines the best time to reconfigure and performs the needed number of reconfigurations depending on the traffic variations. The reconfiguration is computed independently using a column generation algorithm based on a *make-before-break* reconfiguration that chooses which slices to reconfigure. This allows our method to deal with a large number of slices, taking only a few seconds to compute the reconfiguration.

8.3 System Model and Problem Formulation

We consider the network as a directed capacitated graph $G = (V, L)$ where V represents the node set and L the link set. Using the resources available in this network, we must allocate a set of slices requests D.

A network slice request $d \in D$ is modeled as a Service Function Chain (SFC) (as in [Zhang et al., 2017]) with a quintuplet: (i) the source v_{SRC}, (ii) the destination v_{DST}, (iii) the required bandwidth BW_d in traffic units, (iv) the delay requirement γ_d, and, (v) the ordered sequence of network functions c_d that need to be performed, where $f_i^{c_d}$ is the $i-th$ function of chain c_d. Each network function instance $f \in F$ has a installation cost c_f accounting for all the VNF usage costs (licenses, energy consumption, etc). Each slice $d \in D$ provides a revenue u per bandwidth unit.

We aim to find an allocation of the slice requests such that the network operator profit accumulated over a certain time window is maximized. This cumulative profit is the sum of the instantaneous profits p_t at each observation time (i.e. minute). The profits p_t are computed as the

difference between the overall revenue of the allocated slices and the overall cost of the deployed VNFs at time t:

$$p_t = \sum_{d \in D_t} u \cdot \mathrm{BW}_d - \sum_{f \in F_t} c_f \tag{8.1}$$

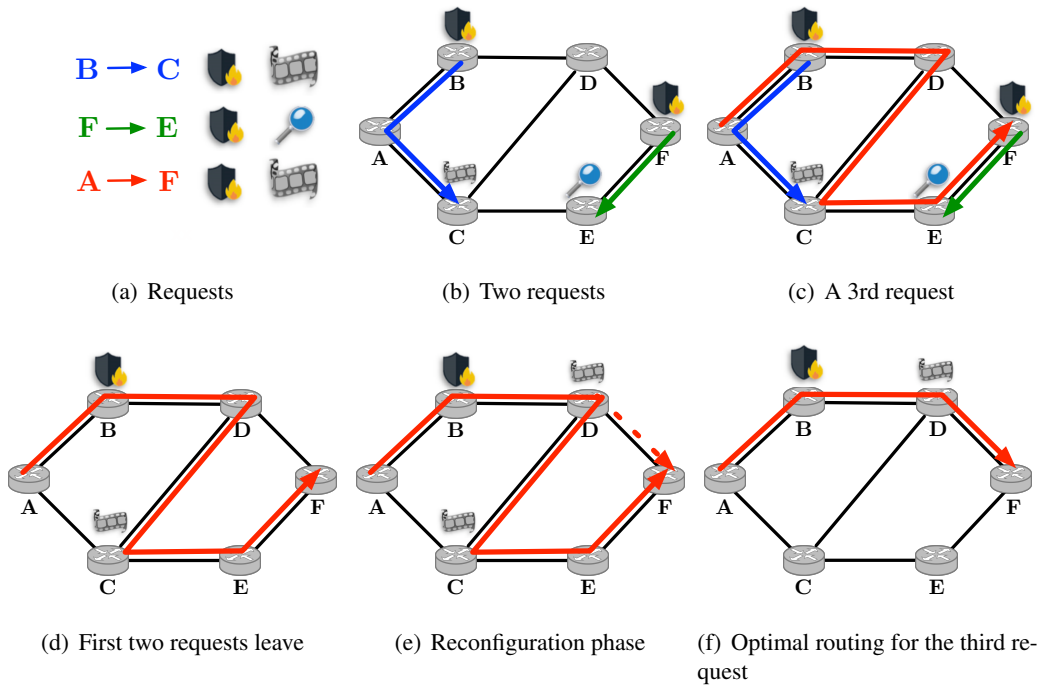
where D_t and F_t is the set of slices and function instances allocated at observation time t, respectively.

In a dynamic scenario with no information on future traffic, the impact of recently arrived requests onto the cumulative profit (at the operator time horizon) is still not known: slices routed using long paths will consume too many resources preventing the allocation of future requests. To take into account that, at each time, we target to place new requests on paths such that resources (i.e. bandwidth at each link and network functions at each node) are minimized.

As a consequence of this lack of information about the future, the trivial mechanics of requests coming and leaving over time will bring the network in a global sub-optimal state, since optimal allocations previously computed are not optimal anymore. Hence, we are forced to periodically reconfigure the network. To do that, we use the *make-before-break* mechanism [Gausseran et al., 2020] that avoids network service disruption due to traffic rerouting. We detail the process of the reconfiguration of a request in the following example of Figure 8.1.

8.3.1 Example

Two requests, *B* to *C* and *F* to *E* are routed during step (b). Four VNFs have been installed in *B*, *C*, *E* and *F* to satisfy the needs of these requests. To avoid the usage cost of new VNFs, the route from *A* to *F* with minimum cost is a long 5–hops route and the VNF already installed in node B is shared by the two slices (step (c)). When requests from *B* to *C* and from *F* to *E* leave, the request is routed on a non-optimal path (step (d)), which uses more resources than necessary. We compute one optimal 3-hop path and reroute the request on it (step (f)) with an intermediate *make-before-break* step (step (e)) in which both routes co-exist. During this intermediate step, traffic can follow both paths, resources are accordingly reserved. The old path is removed when all the allocation and provisioning are ready to be used. In doing so, no packet losses occur and the traffic is not interrupted. In this example, the reconfiguration can be done in only one step of reconfiguration, but we will consider in the following up to 3 steps of reconfiguration.



(a) Requests (b) Two requests (c) A 3rd request

(d) First two requests leave (e) Reconfiguration phase (f) Optimal routing for the third request

Figure 8.1 – An example of the reconfiguration of a request using a *make-before-break* approach with one step.

8.4 Column Generation Optimization models

In this section, we first describe the main principles of the column generation algorithm to solve the problem of reconfiguration of network slices, and then we show how we will leverage this for our deep reinforcement learning algorithm.

Using an Integer Linear Programming formulation to find the optimal solution for each make-before-break reconfiguration is impossible in an acceptable time for a dynamic scenario given the number of slices that we reconfigure. Column generation (CG) [Desaulniers et al., 2005] is a model allowing to solve an optimization model without explicitly introducing all variables, see Figure 8.2 for an explanation. It thus often allows to solve larger instances of the problem than a compact model, in particular, with an exponential number of variables. In this work, the master problem (MP) seeks a possible global reconfiguration for all slices with a path-formulation. This means that the problem decision variables (columns) correspond to paths on the layered graph. In the restricted master problem (RMP), only a subset of potential paths is used for each slice. At the initialization, the set of paths is the one used before reconfiguration. Each pricing problem (PP) then generates a new path for a request, together with the placement of the VNFs. During a reconfiguration, slices are migrated from one path to another. Note that, as the execution of each pricing problem is independent of the others, their solutions can be obtained in parallel. Table 8.1 summarizes the notations used for the column generation model.

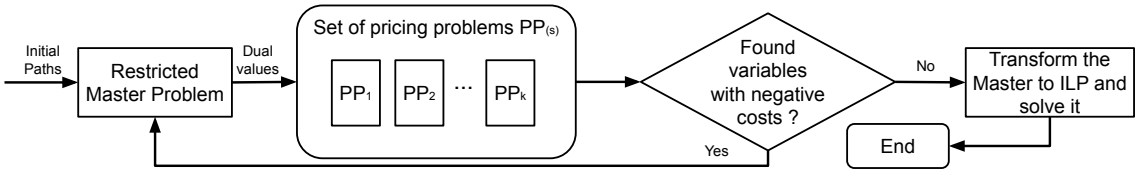


Figure 8.2 – CG is a decomposition method dividing an optimization model into two parts: a master problem and a (set of) pricing problem(s) (PP). The restricted master problem (RMP) solves a fractional relaxation of the problem with a restricted set of columns (i.e. paths on the layered graph). Then the PPs compute the best columns to be added, based on prices given by the dual variables of the RMP. The RMP and PP are then iteratively solved until no more columns can improve the solution of the RMP. Last, the original problem is solved with the integrality constraint using the columns of the RMP.

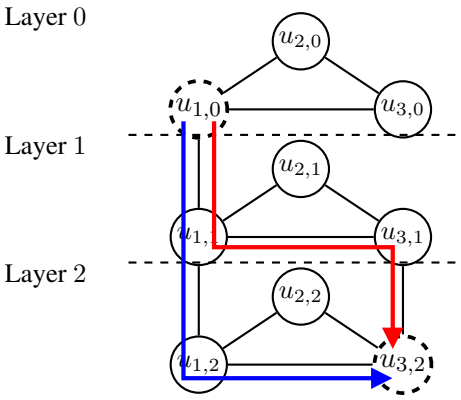


Figure 8.3 – The layered network $G^L(d)$ associated with a demand d such that $v_s = u_1$, $v_d = u_3$, and $c_d = f_1, f_2$, within a triangle network. f_1 is allowed be installed on u_1 and f_2 on u_1 and u_3. Source and destination nodes of $G^L(d)$ are $u_{1,0}$ and $u_{3,2}$. Two possible slices that satisfy d are drawn in red (f_1 is in u_1, f_2 in u_3) and blue (f_1 and f_2 are in u_1).

8.4.1 Layered graph

We make use of the concept of layered graph presented in Figure 8.3 with an example of a graph with three layers. In order to model the chaining constraint of a demand, we associate to each demand d a layered graph $G^L(d)$ with $|c_d|$ layers where $|c_d|$ denotes the number of VNFs in the chain of the demand. Each layer is a duplicate of the original graph and the capacities of both nodes and links are shared among layers. A path on the layered graph starts at layer 0 and ends at layer c_d and corresponds to an assignment of both a path and the locations where functions are being run (the links between layers).

Representing the original graph as a layered graph is a *modeling trick* first proposed in [Dwaraki and Wolf, 2016]. It allows to simplify the problem by reducing it to a routing problem with shared capacities. This allows a drastic reduction of computation time compared to usual strategies using a large number of binary variables due to the ordering constraints of VNFs in the slice.

8.4.2 Master Problem

The Master Problem of the column generation algorithm is described in this section.

Symbol	Description
$G = (V, L)$	Network: V represents the node set and L the link set.
C_ℓ	Bandwidth link capacity of $\ell \in E$.
Γ_ℓ	Link delay of $\ell \in L$.
C_v	Resource node capacity (e.g., CPU, memory, and disk) of node $v \in V$.
Δ_f	Number of compute units required by function $f \in F$ per unit of bandwidth processed.
$c_{u,f}$	Usage cost of function $f \in F$, which also depends on node u.
T	The number of reconfiguration steps.
V^{VNF}	Set of nodes where VNFs can be hosted
D	Set of slices requests
P_d	Set of paths for a demand d
Each demand $d \in D$ is modeled by a quintuplet:	
$(v_{\text{SRC}}, v_{\text{DST}})$	Source and destination nodes,
c_d	Ordered network function sequence for demand d,
$f_i^{c_d}$	$i-th$ function of chain c_d,
BW_d	Required bandwidth units,
γ_d	Maximum required delay for the slice.

TABLE 8.1 – Notations.

Parameters:
- δ_l^p is the number of times the link ℓ appears on path p.
- $\theta_{i,u}^p$ = 1 if node u is used as a VNF on path p on layer i.

Variables:
- $\varphi_p^{d,t} \in [0, 1]$ is the amount of flow of demand d on path p at time step t.
- $y_p^{d,t} \in [0, 1]$ is the maximum amount of flow of demand d on path p between time step $t-1$ and t.
- $z_{u,f} \in [0, 1]$, is equal to 1 if function f is activated on Node u at time step T in the final routing.

We assume an initial configuration is provided with fixed values for $\varphi_p^{d,0}$. The optimization model is written as follows.

Objective: minimize the amount of network resources consumed during the last reconfiguration time step T, which is the sum of the bandwidth used (BW) added to the sum of the costs of the deployed VNFs multiplied by a factor β.

$$\min \sum_{d \in D} \sum_{p \in P_d} \sum_{\ell \in E} \text{BW}_d \, \varphi_p^{d,T} \, \delta_\ell^p + \beta \sum_{u \in V^{\text{VNF}}} \sum_{f \in F} c_{u,f} \, z_{u,f} \tag{8.2}$$

Note that maximizing the accepted bandwidth in equation (8.1) implies minimizing the link bandwidth used by the paths (first term in equation (8.2)); whereas the second term in equation (8.2) represents the cost of the VNFs deployed at time t in equation (8.1).

We considered a value of β for which the bandwidth and the VNFs have the same weight in the objective: using 100% of the available bandwidth has the same cost as using 100% of the available VNFs.

Constraints:
One path constraint. For $d \in D$, time step $t \in \{0, ..., T\}$.

$$\sum_{p \in P_d} \varphi_p^{d,t} = 1 \tag{8.3}$$

Path usage over two consecutive time periods. For $d \in D$, $p \in P_d$, $t \in \{1, ..., T\}$.

$$\varphi_p^{d,t} \leq y_p^{d,t} \text{ and } \varphi_p^{d,t} \leq y_p^{d,t-1} \tag{8.4}$$

Make Before Break - Node capacity constraints. The capacity of a node u in V is shared between each layer and cannot exceed C_u considering the resources used over two consecutive time periods. Δ is the amount of computational units required by function $f \in F$ per unit of bandwidth processed. For $u \in V^{\text{VNF}}$, $t \in \{1, ..., T\}$.

$$\sum_{d \in D} \sum_{p \in P_d} \sum_{i=0}^{|c_d|-1} y_p^{d,t} \cdot \theta_{i,u}^p \cdot \text{BW}_d \cdot \Delta_{f_i^{c_d}} \leq C_u \tag{8.5}$$

Make Before Break - Link capacity constraints. The capacity of a link $\ell \in E$ is shared between each layer and cannot exceed C_ℓ considering the resources used over two consecutive time periods. For $\ell \in E$, $t \in \{1, ..., T\}$,

$$\sum_{d \in D} \sum_{p \in P_d} \text{BW}_d \, y_p^{d,t} \, \delta_\ell^p \leq C_\ell. \tag{8.6}$$

Function activation. To know which functions are activated on which nodes in the final routing. For $u \in V$, $f \in F$, $d \in D$, $i \in \{0, ..., |c_d| - 1\}$,

$$y_p^{d,T} \, \theta_{i,u}^p \leq z_{u,f_i^{c_d}}. \tag{8.7}$$

8.4.3 Pricing Problem

The pricing problem searches for a possible placement for the slice. Since a reconfiguration can be done in several steps, a pricing problem is launched for each demand, at each time step. The objective of the pricing problem for each demand d at time t is called the reduced cost and is expressed using the equation in [Desaulniers et al., 2005].

Parameters:
- μ are the dual values of the master's constraints. The number written in upperscript is the reference of the master's constraints.

Variables:
- $\varphi_{\ell,i} \in \{0, 1\}$ is the amount of flow on link ℓ in layer i.
- $\alpha_{u,i} \in \{0, 1\}$ is the amount of flow on node u in layer i.

Objective: minimize the amount of network resources consumed for the demand d at time t.

$$\min \sum_{\ell \in E} \sum_{i=0}^{|c_d|} \varphi_{\ell,i} \, \text{BW}(1 + \mu_{l,t}^{(8.6)}) + \text{BW} \sum_{u \in V^{\text{VNF}}} \mu_{u,t}^{(8.5)} \sum_{i=0}^{|c_d|-1} \Delta_{f_i^{c_d}} \, \alpha_{u,i} \\ - \mu_{d,t}^{(8.3)} + \beta \sum_{u \in V^{\text{VNF}}} \sum_{f \in F} c_{u,f} \, z_{u,f} \, \mu_{d,u,f}^{(8.7)} \tag{8.8}$$

where $\mu^{(8.7)}_{d,u,f} = 0$ when $t \neq T$, see constraints (8.7).

Constraints:

Flow conservation constraints for the demand d. For $u \in V^{\text{VNF}}$.

$$\sum_{\ell \in \omega^+(u)} \varphi_{\ell,0} \; - \sum_{\ell \in \omega^-(u)} \varphi_{\ell,0} + \alpha_{u,0} = \begin{cases} 1 \text{ if } u = v_s \\ 0 \text{ else} \end{cases} \tag{8.9}$$

$$\sum_{\ell \in \omega^+(u)} \varphi_{\ell,|c_d|} \; - \sum_{\ell \in \omega^-(u)} \varphi_{\ell,|c_d|} - \alpha_{u,|c_d|-1} = \begin{cases} -1 \text{ if } u = v_d \\ 0 \text{ else} \end{cases} \tag{8.10}$$

$$\sum_{\ell \in \omega^+(u)} \varphi_{\ell,i} \; - \sum_{\ell \in \omega^-(u)} \varphi_{\ell,i} + \alpha_{u,i-1} - \alpha_{u,i-1} = 0$$

$$0 < i < |c_d| \tag{8.11}$$

Delay constraints. The sum of the link delays of the flow must not exceed the delay requirement of demand d.

$$\sum_{i=0}^{|c_d|} \varphi_{\ell,i} \, \Gamma_\ell \leq \gamma_d \tag{8.12}$$

Function activation. To know which functions are activated on which nodes. For $u \in V^{\text{VNF}}$, $f \in F$, layer $i \in \{0, ..., |c_d| - 1\}$

$$\alpha_{u,i} \leq z_{u,f_i^{c_d}} \tag{8.13}$$

Location constraints. A node may be enabled to run only a subset of the virtual network functions. For $u \in V^{\text{VNF}}$, $i \in \{0, ..., |c_d| - 1\}$, if the $(i+1)^{\text{th}}$ function of c_d cannot be installed on u, we have

$$\alpha_{u,i} = 0. \tag{8.14}$$

8.4.4 Motivation for `Deep-REC`

Our algorithm reconfigures a given set of network slices from an initial routing and placement of network functions to another solution that improves the usage of the network resources (both in terms of link bandwidth and VNFs). This reconfiguration is done with a make-before-break approach to avoid interruptions of the flows.

In a dynamic scenario, due to the frequent arrival and departure of slices, the network is regularly in a sub-optimal state. Nevertheless, the frequency to run this reconfiguration algorithm was found on a empirical manner. Indeed, previous works [Gausseran et al., 2020, Gausseran et al., 2021] showed that reconfiguring every 15 minutes allowed a good ratio between cost reduction, quality of reconfigurations, acceptance rate and computation time.

A fixed frequency is easy to set up but in practice, at some specific time of a day, reconfiguration may not be needed as traffic remains stable and the network is already in an optimal state. A new reconfiguration at this time won't bring any gain. On the opposite, during high-dynamic traffic period, more frequent reconfigurations may be suitable to maintain an acceptable state of the network with efficient network resource usage. Therefore, a network operator might be interested to adapt the reconfiguration frequencies depending on the congestion of the network, and the nature of the traffic. This is the main goal of this work.

Indeed, even if we use a *make-before-break* reconfiguration model which allows to reconfigure without degrading the quality of service, reconfiguring generates network management costs to

migrate VNFs, and to compute and instantiate the intermediate and the new paths [Noghani et al., 2019].

We present in the new section our DRL model named Deep-REC to choose when to reconfigure in order to optimally adapt to the evolving network state. Our objective is to maximize the cumulative profit as presented before: the sum of the instantaneous profits p_t (See (8.1)).

8.5 Deep Reinforcement Learning (DRL) Algorithm: Deep-REC

The reinforcement learning paradigm formalises a discrete time stochastic control process (as our networking problem) where an *agent* interacts with an *environment* (in our case, the network). At each time step t, the *agent* interacts with its *environment* by (i) observing from the *environment* the current state s, (ii) accordingly, taking a decision (an *action*) a, (iii) receiving a reward $r(s, a)$, and, (iv) observing a new state s' (the network has transitioned from s to s'). The agent can repeat this process for a potential infinite number of time steps, giving rise to a *trajectory*. The sum of the discounted rewards over a trajectory from time t, or *discounted return*, is calculated as:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

where $\gamma \leq 1$. The expectation of G_t over all possible trajectories initiated at a state s after taking an action a is the so-called *Q-value function* $Q(s, a)$. We aim to take, at each state s, the action a maximizing the *Q-value function*. Then, we need to estimate $Q(s, a)$.

In [Watkins and Dayan, 1992], authors proposed the *Q-learning* algorithm to learn $Q(s, a)$ from a sequence of agent interactions with the environment. Unfortunately, when the state and action spaces are huge, Q-learning needs a prohibitive computation time. To overcome that, Deep Q-learning Network (DQN) [Mnih et al., 2015] makes use of a deep neural network to approximate the *Q-value function* for high-dimensional state-space problems, as our case. Finally, we also opt for DQN since, conversely to other reinforcement learning algorithms, it can learn efficiently from past experiences without introducing bias.

Description. For the implementation we use the DQN agent from *tf_agents.agents.dqn.dqn_agent*, and the neural network from *tf_agents.networks.q_network* [Abadi et al., 2015]. The network is composed of a pre-processing layer from keras [Chollet et al., 2015] used for batch normalization and 2 layers of 64 neurons each. The batch size is 288, which is large enough to properly normalize. We use Adam optimizer with a learning rate of 1e-3 and we update the network every 16 states.
The discount factor γ is set to 0.9, a value large enough to show the importance of future actions. We use an epsilon-greedy policy, where ϵ is set to 0.99 and decay to 0 in 200 instances. The replay buffer has a size of 50 instances and we train the agent on 250 instances.

Context. A 24-hour day consists of 1440 minutes. We decided to discretize it into 288 periods of 5 minutes in order to optimize the training of our agent. It is recalled that the objective is to maximize the profit, while reconfiguring as efficiently as possible. The agent can potentially choose to reconfigure 288 times. To make the agent aware of the implicit trade-off between reconfiguring now or later, an artificial cost per reconfiguration v_R is introduced. Reconfiguring a network will never decrease the profit, but the agent has to learn when a reconfiguration is really worth it.

The agent will then learn the optimal number of reconfigurations to maximize the profit with this artificial cost. This cost can be real (management cost) or it can be fixed to get a given number

of reconfigurations per day. The advantage of this technique compared to having a maximum number of reconfigurations allowed is that allows the agent to make more or less reconfigurations, adapting its behavior to the current period of the day.

State and Action Spaces. The network state can be described based on the next five quantities: (i) the number of minutes since the last reconfiguration ΔT, (ii) the number of slices added since the last reconfiguration λ, (iii) the number of slices released since the last reconfiguration μ, (iv) the current profit p_t (8.1) and, (v) the current time t. ΔT represent the current allocation oldness, λ and μ estimate the current network load.

The action space consists of two actions: to perform or not a reconfiguration at current time based on the decision of our agent.

Reward Function. If the agent has chosen not to perform a reconfiguration, the reward is 0.

Otherwise, the agent selects the reconfiguration, and two scenarios are possible:

1. *The reconfiguration was worth it.* A reconfiguration at time t is computed. To have a long-term vision, we simulate the network behaviour (slices arrivals and departures) with the new network configuration for the next three time slots (in training, we can simulate the future requests). Finally, we estimate the accumulated profit gained with the reconfiguration as:

$$\Delta p_R = \{\sum_{k=t}^{t+3} p_k | \text{reconf at } t\}$$

2. *The reconfiguration was actually not worth it.* The reward is estimated differently. We suppose that no reconfiguration was performed at t and we also simulate the network behaviour (slices arrivals and departures) for the next three time slots. Again, we estimate the accumulated profit gained, this case, without the reconfiguration as:

$$\Delta p_{NR} = \{\sum_{k=t}^{t+3} p_k | \text{no reconf at } t\}$$

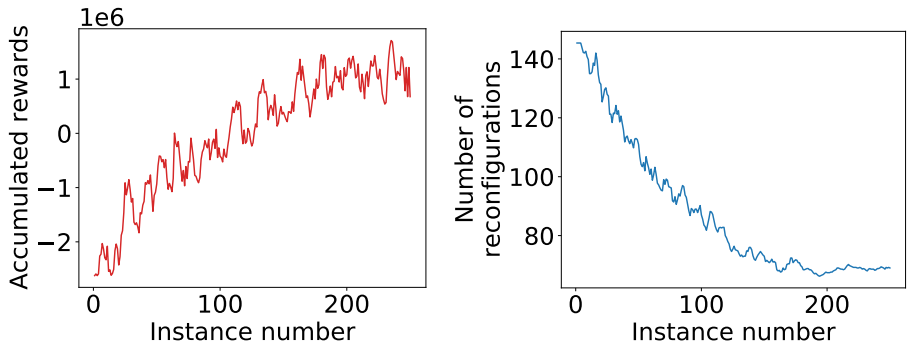
Finally, we compute the reward as:

$$r = \Delta p_R - \Delta p_{NR} - v_R.$$

We therefore have a positive reward when the profit increase Δp_R (if *reconfiguring* was the good decision) compensates both the profit increase Δp_{NR} (if *not reconfiguring* was the good decision) and the reconfiguration cost.

Training. We are now studying the efficiency of the learning of our agent trained on 250 instances. In the Figure 8.4(a), the return of the agent on the training environment increases during the 250 trained instances, which implies that it learns to maximize the accumulated rewards on each instance.

We should not reconfigure too often during a day, so in Figure 8.4(b), we study the variation of the number of reconfigurations made by the agent on each instance. The agent starts by reconfiguring randomly: 1 time out of 2 and thus about 144 times per instance, and it learns that it must reduce the number of reconfigurations to maximize the accumulated reward. When the agent has trained on around 200 instances, the epsilon reaches 0 and the number of reconfigurations converges to around 70 reconfigurations per instance.



(a) Accumulated rewards (b) Number of reconfigurations

Figure 8.4 – Learning Curves

8.6 Data Set

Topology. We conduct simulations on a real-world topology from SNDlib [Orlowski et al., 2010], `ta1` (24 nodes, 55 links), which includes 6 datacenters on which all VNFs can be instantiated. The cost of VNF c_f is equal to the revenue of 2000 times the revenue u of a megabyte served.

Slice demands Each slice is composed of a chain of up to 5 VNFs, requires a specific amount of bandwidth, and has latency constraints. We consider four different types of demands corresponding to four services: Video Streaming, Web Service, VoIP, and online gaming. The characteristics of each service are reported in Table 8.2 and have been already used by [Savi et al., 2015]. The bandwidth usage was chosen according to the distribution of Internet traffic described in [Cisco, 2015]. The latency requirements are expressed in milliseconds and represent the maximum delay between the source and destination.

Each minute, 1 to 5 slice requests arrive (uniform random distribution) and slices that have reached the end of their life are removed from the network. By varying the lifetime of the slices, we can vary the maximum number of slices present at the same time on the network and so that the load on the network follows the curve in Figure 8.5. This figure represents a real distribution of traffic measured on a dedicated network operator. We divided this traffic in five different periods, where D1 is a low-traffic period, and in D5, the network is highly congested. There are between 30 and 180 slices present at each moment on the network and in 24 hours, there are about 4320 arrivals of slices.

Reconfiguration Cost. To train `Deep-REC`, we define a fixed and artificial cost to the reconfiguration v_R. This cost can be adapted to reconfigure more or less. In our study, it is equal to the cost of deploying a VNF for 15 minutes, which implies that a reconfiguration is useful if it allows to shut down a VNF for at least 15 minutes. To be usable in practice, a reconfiguration must be done quickly. Thanks to the column generation, we can limit the computation time of each reconfiguration to 15 seconds without affecting its efficiency.

Slice Types	VNF chain	Latency	bw (Mbps)
Web Service	NAT-FW-TM-WOC-IDPS	10ms	100
Video Streaming	NAT-FW-TM-VOC-IDPS	5ms	256
VoIP	NAT-FW-TM-FW-NAT	3.5ms	64
Online Gaming	NAT-FW-VOC-WOC-IDPS	2.5ms	50

TABLE 8.2 – Characteristics of network slices

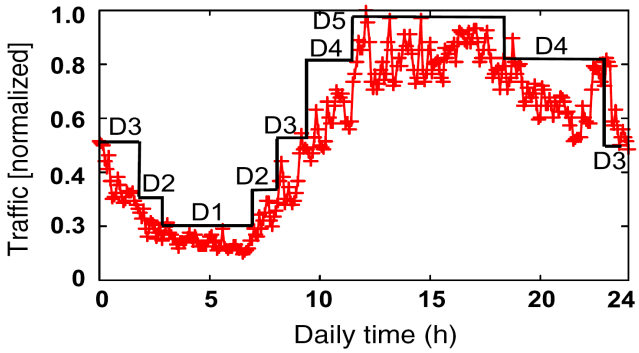


Figure 8.5 – Distribution of traffic

8.7 Numerical Results

We compare the results obtained with three solutions in this section:

— `No-REC`: the slices are added in and removed from the network over time, and no reconfiguration is performed,

— `REC-15`: the reconfiguration is carried out every 15 minutes using a *make-before-break* strategy,

— `Deep-REC`: our deep-learning *make-before-break* reconfiguration proposal.

We first show that reconfiguring the network leads to significant gains in terms of profit. We then discuss the importance of selecting the best moments to carry out the reconfigurations, allowing to perform fewer reconfigurations while achieving similar gains.

8.7.1 Improved network operational Cost

Figure 8.6 presents the network cost ((second term in equation (8.2))) per megabyte of data sent over the network throughout the day. We observe that the costs achieved by `REC-15` and `Deep-REC` are very similar with a clear improvement compared to `No-REC`: `REC-15` allows an improvement of 36.82% when `Deep-REC` performs a little better with 38.05%. We also see that the cost is rather stable throughout the day while with `No-REC` the cost increases strongly during low-congestion period (periods D1 and D2, between 2am and 7am). The global cost improvement through a day of `REC-15` is 34.18% versus 35.55% with `Deep-REC`.

Figure 8.7 presents the network operational cost for the three strategies in terms of VNF costs (second term in equation (8.2)). This shows that both `Deep-REC` and `REC-15` reduce the network

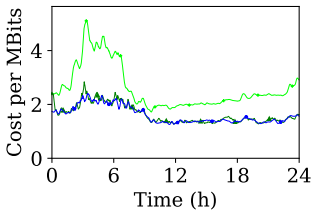


Figure 8.6 – Cost per MB

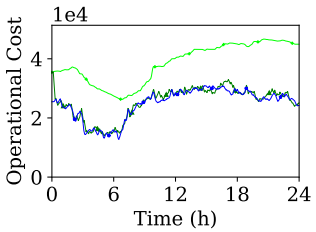


Figure 8.7 – Operational Cost

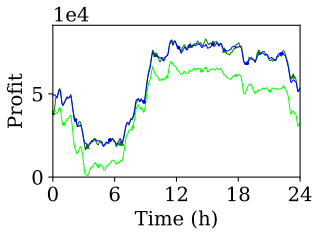


Figure 8.8 – Profit

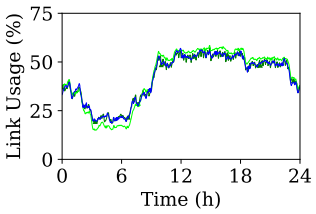


Figure 8.9 – Percentage of links capacity used

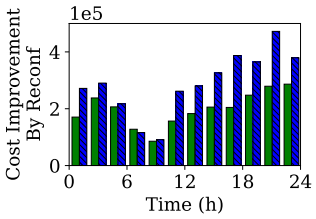


Figure 8.10 – Cost gain per Reconf

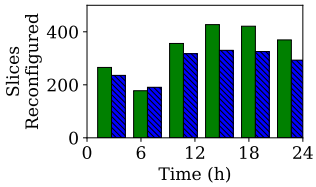


Figure 8.11 – Number of slices modified

operational cost compared to `No-REC`. This justifies the necessity of reconfiguring. Moreover, the two reconfiguring strategies are comparable for this parameter.

8.7.2 Improved Profit and link utilisation

Figure 8.8 shows the achieved profit, whose maximization is the objective of the reconfiguration: `Deep-REC` and `REC-15` have similar performance and improve the profit compared to `No-REC`. Indeed, the profit improvement of `REC-15` is 32.75% versus 32.53% for `Deep-REC`.

Finally, Figure 8.9 shows that even when minimizing VNF costs, reconfiguring the network does not lead to an increase of congestion: we observe a slightly higher utilization of links during periods D1-D2, but when the network is heavily loaded (periods D4-D5), there is a reduction of the congestion of the network.

As a conclusion, reconfiguring the network reduces congestion while reducing costs. Moreover, we validate with these results the performance of `Deep-REC` as it leads to similar profit as a periodic reconfiguration strategy such as `REC-15`. We show in the following that `Deep-REC`, by performing reconfigurations at the ideal moment, achieves this efficiency while reducing the total number of reconfigurations through a day compared to a regular and fixed reconfiguration strategy such as `REC-15`.

8.7.3 Number of reconfigurations

Figure 8.10 presents the cost improvement divided by the number of reconfigurations over periods of two hours. This shows that `Deep-REC` performs more efficient reconfigurations than `REC-15`. Each reconfiguration leads to a better improvement in terms of the network costs.

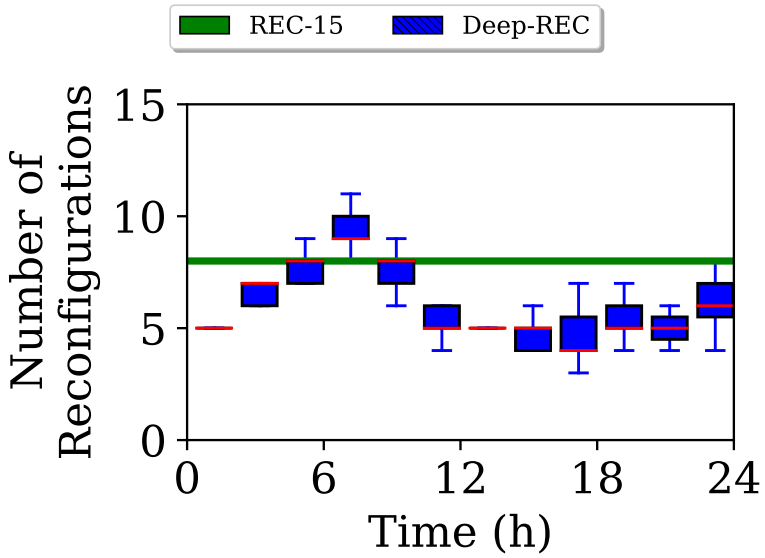


Figure 8.12 – Reconfiguration distribution for `Deep-REC` compared to `REC-15`.

Figure 8.11 presents the number of slices modified during the reconfiguration. Our algorithm `Deep-REC` modified approximately 20% less slices than `REC-15`, and thus there is less impact on the network (less modifications, less computation).

Finally, Figure 8.12 shows the distribution of the number of reconfigurations during a day over two-hour periods. The green line on the figure represents `REC-15` which does a constant number of reconfiguration, namely 8 (a reconfiguration every 15-minutes). In contrast, `Deep-REC` adapts its actions to the network load and does not carry out reconfigurations when they are not necessary, leading to a reduction of their number during the majority of the day, see the period between 10am and 4pm. Moreover, `Deep-REC` performs more reconfigurations than `REC-15` during the ascending phase (between D1 and D5) in order to react to the rapid change of the network and, thus, to maintain a good profit. With 96 reconfigurations during a day (against 73.2 in average for `Deep-REC`), `REC-15` has 31.15% more reconfigurations, for only 0.22% profit improvement.

8.8 Conclusion and Future Work

We presented in this chapter a DRL strategy to carry out an efficient reconfiguration of network slices with dynamic network demands. Our proposal, `Deep-REC`, chooses adequately the best time to reconfigure, it reconfigures few times during low-congestion periods compared to a fixed-frequency reconfiguration strategy. Moreover, when the network is highly congested, `Deep-REC` adapts his behavior, and reconfigures more in order to maximize the network profit. Finally, `Deep-REC` reduces by almost a quarter the number of reconfigurations needed over a day, while achieving similar performances in terms of network operational costs, achieved profit, and congestion of the network. As a future work we plan to extend the simulations of `Deep-REC` for several networks and to study the efficiency of our proposal by experimentation on a real platform.

CHAPTER 9

Conclusion and Future Work

Artificial Intelligence (AI) is a fast-growing research and development discipline which is rapidly changing and shaping several areas of society. However, together with opportunities, the field has also brought new challenges across different domains that need to be explored.

One of these challenges deals with finding efficient graph similarity measures to optimize the process of learning on graphs. In Chapter 2, we investigated the potential of graphs kernels in the field of networking by studying the problem of anomaly detection and traffic classification. We proposed a graph-based learning framework which aims at preventing anomalies from disrupting the network while providing assistance for traffic monitoring. At the core of the framework lies a graph kernel tailored for networking that was developed based on random walks and the Weisfeiler-Lehman hierarchy of isomorphism tests. The framework was made fast and scalable by using special purpose graphs along with an approximation algorithm for which we derived some theoretical guarantees. An interesting future research direction concerns the theoretical investigation of the interaction between graph kernels and graph neural networks towards graph learning.

Then, we studied the long-term behavior of multi-agent systems under deterministic and stochastic majority dynamics around the consensus task. We considered two different flavors of the problem, that have been widely studied in the literature.

The first, in Chapter 3, is a reversible majority dynamics. We studied a biased opinion dynamics where agents are influenced by the majority of their neighbors. We have shown that consensus on the preferred opinion exhibits a dichotomy by proving that convergence time is always polynomial for cubic graphs, whereas it becomes exponential (for a sufficiently small bias) in random Δ-regular graphs ($\Delta \geq 4$), answering an open conjecture in the literature. Our analysis of this dichotomy was based on a new approach to study the time of adoption of a new opinion or technology by exploiting structural properties of graphs in light of a generalized notion of graph domination. An interesting avenue for future research is to analyze the case of multiple opinions and the relationship with potential opinion diffusion models which reproduce complex properties observed in real-world networks such as social networks.

The second, in Chapter 4, is an irreversible majority dynamics. We studied a bootstrap percolation process in which, at every step, infected vertices with a majority of infected neighbors becomes infected and remain so forever. We determined almost tight bounds on the minimum size required by an initially infected vertices set in order to eventually lead to a full infection of the population in the cases of square grids and tori. Extensions for future work may cover the study of multidimensional grids and the speed of the infection.

Another challenge in the field of AI concerns overcoming the shortcomings induced by the complex and massive neural network architectures. Indeed, although neural networks have de-

monstrated impressive capabilities across several complicated tasks, such successes come at the cost of significant computational complexity. A fundamental line of research in this direction is aimed towards lowering the size of such networks while retaining good accuracy through compression techniques.

In Chapter 5, we first presented a simplified proof, with a more direct approach and resourcing to more elementary tools, for a fundamental result on the random subset sum problem that has gained renewed attention for its implications in the theory of compressing artificial neural networks with the aim of increasing their efficiency. Then, in Chapter 6, we studied a multidimensional generalization of the problem and demonstrated as an application that a recently proposed neural network model can, with high probability, approximate any neural network within a polynomial overhead in the number of parameters. Interesting future directions consist in investigating the real-world implications of these results and studying pruning approaches from an algorithmic perspective.

Finally, we looked at some other miscellaneous problems. In Chapter 7, we presented a multi-dimensional colored packing approach to carry out an efficient placement of network slices while protecting against failures of, e.g., compute servers, within an infrastructure that benefits from NFV and SDN technologies. We validated the performance of our algorithms on real-world ISP networks. Furthermore, in Chapter 8, we presented a deep reinforcement learning strategy to carry out an efficient reconfiguration of network slices with dynamic network demands, in the context of the 5G network slicing paradigm. Our aim is to determine the best time to reconfigure in order to maintain an efficient provisioning of the network.

In this thesis, we only addressed some of the challenges emerging within the AI field in its broad perspective, with a particular focus on investigating some foundations of networks spanning telecommunications networks, multi-agent systems and neural networks. While many of the challenges still need to be addressed, we believe there is tremendous potential of this field in this endeavor towards supporting efficient, effective and sustainable services.

Bibliography

[Abadi et al., 2015] Abadi, M. et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

[Adams et al., 2011] Adams, S. S., Troxell, D. S., and Zinnen, S. L. (2011). Dynamic monopolies and feedback vertex sets in hexagonal grids. *Computers Mathematics with Applications*, 62(11):4049–4057.

[Adler and Lev, 2003] Adler, J. and Lev, U. (2003). Bootstrap percolation: visualizations and applications. *Brazilian Journal of Physics*, 33(3):641–644.

[Afek et al., 2011] Afek, Y., Alon, N., Barad, O., Hornstein, E., Barkai, N., and Bar-Joseph, Z. (2011). A biological solution to a fundamental distributed computing problem. *science*, 331(6014):183–185.

[Agrawal et al., 1994] Agrawal, R., Srikant, R., et al. (1994). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499.

[Alimonti and Kann, 2000] Alimonti, P. and Kann, V. (2000). Some apx-completeness results for cubic graphs. *Theoretical Computer Science*, 237(1-2):123–134.

[Anagnostopoulos et al., 2020] Anagnostopoulos, A., Becchetti, L., Cruciani, E., Pasquale, F., and Rizzo, S. (2020). Biased opinion dynamics: When the devil is in the details. In *29th International Joint Conference on Artificial Intelligence (IJCAI 2020)*, pages 53–59. International Joint Conferences on Artificial Intelligence Organization.

[Arvind et al., 2020] Arvind, V., Fuhlbrück, F., Köbler, J., and Verbitsky, O. (2020). On weisfeiler-leman invariance: Subgraph counts and related graph properties. *Journal of Computer and System Sciences*, 113:42–59.

[Auletta et al., 2015] Auletta, V., Caragiannis, I., Ferraioli, D., Galdi, C., and Persiano, G. (2015). Minority becomes majority in social networks. In *International conference on web and internet economics*, pages 74–88. Springer.

[Auletta et al., 2018] Auletta, V., Ferraioli, D., and Greco, G. (2018). Reasoning about consensus when opinions diffuse through majority dynamics. In *IJCAI*, pages 49–55.

[Azuma, 1967] Azuma, K. (1967). Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal, Second Series*, 19(3):357–367.

[Babai, 2016] Babai, L. (2016). Graph isomorphism in quasipolynomial time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 684–697.

[Babai and Luks, 1983] Babai, L. and Luks, E. M. (1983). Canonical labeling of graphs. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 171–183.

[Bach, 2008] Bach, F. R. (2008). Graph kernels between point clouds. In *Proceedings of the 25th international conference on Machine learning*, pages 25–32.

[Baker and Coffman, 1981] Baker, B. S. and Coffman, Jr, E. G. (1981). A tight asymptotic bound for next-fit-decreasing bin-packing. *SIAM Journal on Algebraic Discrete Methods*, 2(2):147–152.

[Balogh and Bollobás, 2006] Balogh, J. and Bollobás, B. (2006). Bootstrap percolation on the hypercube. *Probability Theory and Related Fields*, 134(4):624–648.

[Balogh and Pete, 1998] Balogh, J. and Pete, G. (1998). Random disease on the square grid. *Random Structures & Algorithms*, 13(3-4):409–422.

[Bansal et al., 2006] Bansal, N., Caprara, A., and Sviridenko, M. (2006). Improved approximation algorithms for multidimensional bin packing problems. In *IEEE FOCS*, pages 697–708. IEEE.

[Becchetti et al., 2020a] Becchetti, L., Clementi, A., and Natale, E. (2020a). Consensus dynamics: An overview. *ACM SIGACT News*, 51(1):58–104.

[Becchetti et al., 2014] Becchetti, L., Clementi, A., Natale, E., Pasquale, F., and Silvestri, R. (2014). Plurality consensus in the gossip model. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 371–390. SIAM.

[Becchetti et al., 2017] Becchetti, L., Clementi, A., Natale, E., Pasquale, F., Silvestri, R., and Trevisan, L. (2017). Simple dynamics for plurality consensus. *Distributed Computing*, 30(4):293–306.

[Becchetti et al., 2016] Becchetti, L., Clementi, A., Natale, E., Pasquale, F., and Trevisan, L. (2016). Stabilizing consensus with many opinions. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 620–635. SIAM.

[Becchetti et al., 2020b] Becchetti, L., Clementi, A. E., Natale, E., Pasquale, F., and Trevisan, L. (2020b). Find your place: Simple distributed algorithms for community detection. *SIAM Journal on Computing*, 49(4):821–864.

[Becchetti et al., 2022] Becchetti, L., da Cuhna, A. C. W., Clementi, A., d'Amore, F., Lesfari, H., Natale, E., and Trevisan, L. (2022). On the multidimensional random subset sum problem. *arXiv preprint arXiv:2207.13944*.

[Bellman, 1966] Bellman, R. (1966). Dynamic programming. *Science*, 153(3731):34–37.

[Benevides et al., 2021] Benevides, F., Bermond, J.-C., Lesfari, H., and Nisse, N. (2021). *Minimum lethal sets in grids and tori under 3-neighbour bootstrap percolation*. PhD thesis, Université Côte d'Azur.

[Benevides and Przykucki, 2013] Benevides, F. and Przykucki, M. (2013). On slowly percolating sets of minimal size in bootstrap percolation. *The Electronic Journal of Combinatorics*, pages P46–P46.

[Benevides and Przykucki, 2015] Benevides, F. and Przykucki, M. (2015). Maximum percolation time in two-dimensional bootstrap percolation. *SIAM Journal on Discrete Mathematics*, 29(1):224–251.

[Berlinet and Thomas-Agnan, 2011] Berlinet, A. and Thomas-Agnan, C. (2011). *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media.

[Bernstein et al., 2013] Bernstein, D. J., Jeffery, S., Lange, T., and Meurer, A. (2013). Quantum algorithms for the subset-sum problem. In *International Workshop on Post-Quantum Cryptography*, pages 16–33. Springer.

[Bessy et al., 2019] Bessy, S., Ehard, S., Penso, L. D., and Rautenbach, D. (2019). Dynamic monopolies for interval graphs with bounded thresholds. *Discrete Applied Mathematics*, 260:256–261.

[Blalock et al., 2020] Blalock, D., Gonzalez Ortiz, J. J., Frankle, J., and Guttag, J. (2020). What is the state of neural network pruning ? *Proceedings of machine learning and systems*, 2:129–146.

[Bollobás, 1980] Bollobás, B. (1980). A probabilistic proof of an asymptotic formula for the number of labelled regular graphs. *European Journal of Combinatorics*, 1(4):311–316.

[Bollobás, 2004] Bollobás, B. (2004). *Extremal graph theory*. Courier Corporation.

[Bollobás, 2006] Bollobás, B. (2006). *The art of mathematics: Coffee time in Memphis*. Cambridge University Press.

[Bordes et al., 2005] Bordes, A., Ertekin, S., Weston, J., and Bottou, L. (2005). Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6(Sep):1579–1619.

[Borgs et al., 2004] Borgs, C., Chayes, J. T., Mertens, S., and Pittel, B. (2004). Phase diagram for the constrained integer partitioning problem. *Random Structures & Algorithms*, 24(3):315–380. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rsa.20001.

[Borgs et al., 2001] Borgs, C., Chayes, J. T., and Pittel, B. G. (2001). Phase transition and finite-size scaling for the integer partitioning problem. *Random Struct. Algorithms*, 19(3-4):247–288.

[Borgwardt et al., 2020] Borgwardt, K., Ghisu, E., Llinares-López, F., O'Bray, L., Rieck, B., et al. (2020). Graph kernels: State-of-the-art and future challenges. *Foundations and Trends® in Machine Learning*, 13(5-6):531–712.

[Borgwardt and Kriegel, 2005] Borgwardt, K. M. and Kriegel, H.-P. (2005). Shortest-path kernels on graphs. In *Fifth IEEE international conference on data mining (ICDM'05)*, pages 8–pp. IEEE.

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152.

[Boyd et al., 2006] Boyd, S., Ghosh, A., Prabhakar, B., and Shah, D. (2006). Randomized gossip algorithms. *IEEE transactions on information theory*, 52(6):2508–2530.

[Bringmann and Wellnitz, 2021] Bringmann, K. and Wellnitz, P. (2021). On near-linear-time algorithms for dense subset sum. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1777–1796. SIAM.

[Brown et al., 2020] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

[Bunke and Shearer, 1998] Bunke, H. and Shearer, K. (1998). A graph distance metric based on the maximal common subgraph. *Pattern recognition letters*, 19(3-4):255–259.

[Bürger, 2000] Bürger, R. (2000). *The mathematical theory of selection, recombination, and mutation*. John Wiley & Sons.

[Burkholz et al., 2022] Burkholz, R., Laha, N., Mukherjee, R., and Gotovos, A. (2022). On the existence of universal lottery tickets. In *International Conference on Learning Representations*.

[Cai et al., 2018] Cai, H., Zheng, V. W., and Chang, K. C.-C. (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1616–1637.

[Centeno et al., 2011] Centeno, C. C., Dourado, M. C., Penso, L. D., Rautenbach, D., and Szwarcfiter, J. L. (2011). Irreversible conversion of graphs. *Theoretical Computer Science*, 412(29):3693–3700.

[Chalupa et al., 1979] Chalupa, J., Leath, P. L., and Reich, G. R. (1979). Bootstrap percolation on a bethe lattice. *Journal of Physics C: Solid State Physics*, 12(1):L31.

[Chastain et al., 2014] Chastain, E., Livnat, A., Papadimitriou, C., and Vazirani, U. (2014). Algorithms, games, and evolution. *Proceedings of the National Academy of Sciences*, 111(29):10620–10623.

[Chazelle, 2009] Chazelle, B. (2009). Natural algorithms. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 422–431. SIAM.

[Chekuri and Khanna, 2004] Chekuri, C. and Khanna, S. (2004). On multidimensional packing problems. *SIAM journal on computing*, 33(4):837–851.

[Chen, 2009] Chen, N. (2009). On the approximability of influence in social networks. *SIAM Journal on Discrete Mathematics*, 23(3):1400–1415.

[Chen et al., 2022a] Chen, X., Jin, Y., Randolph, T., and Servedio, R. A. (2022a). Average-case subset balancing problems. In Naor, J. S. and Buchbinder, N., editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 743–778. SIAM.

[Chen et al., 2022b] Chen, X., Jin, Y., Randolph, T., and Servedio, R. A. (2022b). Average-case subset balancing problems. In Naor, J. S. and Buchbinder, N., editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 743–778. SIAM.

[Cheng et al., 2020] Cheng, X., Wu, Y., Min, G., Zomaya, A. Y., and Fang, X. (2020). Safeguard network slicing in 5g: A learning augmented optimization approach. *IEEE JSAC*.

[Chiang et al., 2013a] Chiang, C.-Y., Huang, L.-H., Li, B.-J., Wu, J., and Yeh, H.-G. (2013a). Some results on the target set selection problem. *Journal of Combinatorial Optimization*, 25(4):702–715.

[Chiang et al., 2013b] Chiang, C.-Y., Huang, L.-H., and Yeh, H.-G. (2013b). Target set selection problem for honeycomb networks. *SIAM Journal on Discrete Mathematics*, 27(1):310–328.

[Chollet et al., 2015] Chollet, F. et al. (2015). Keras. Technical report, GitHub.

[Chung and Graham, 1997] Chung, F. R. and Graham, F. C. (1997). *Spectral graph theory*. American Mathematical Soc.

[Cisco, 2015] Cisco, I. (2015). Cisco visual networking index: Forecast and methodology, 2014–2019. *CISCO White paper*.

[Coates et al., 2018] Coates, A., Han, L., and Kleerekoper, A. (2018). A unified framework for opinion dynamics. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems.

[Coja-Oghlan et al., 2014] Coja-Oghlan, A., Feige, U., Krivelevich, M., and Reichman, D. (2014). Contagious sets in expanders. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1953–1987. SIAM.

[Cooper et al., 2013] Cooper, C., Elsasser, R., Ono, H., and Radzik, T. (2013). Coalescing random walks and voting on connected graphs. *SIAM Journal on Discrete Mathematics*, 27(4):1748–1758.

[Cooper et al., 2014] Cooper, C., Elsässer, R., and Radzik, T. (2014). The power of two choices in distributed voting. In *International Colloquium on Automata, Languages, and Programming*, pages 435–446. Springer.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

[Cruciani et al., 2021] Cruciani, E., Mimun, H. A., Quattropani, M., and Rizzo, S. (2021). Phase transitions of the k-majority dynamics in a biased communication model. In *International Conference on Distributed Computing and Networking 2021*, pages 146–155.

[da Cunha et al., 2022a] da Cunha, A., d'Amore, F., Giroire, F., Lesfari, H., Natale, E., and Viennot, L. (2022a). Revisiting the Random Subset Sum problem. Number: arXiv:2204.13929 arXiv:2204.13929 [math].

[da Cunha et al., 2021] da Cunha, A., Natale, E., and Viennot, L. (2021). Proving the Lottery Ticket Hypothesis for Convolutional Neural Networks. In *International Conference on Learning Representations (ICLR)*.

[da Cunha et al., 2022b] da Cunha, A., Natale, E., and Viennot, L. (2022b). Proving the strong lottery ticket hypothesis for convolutional neural networks. In *International Conference on Learning Representations*.

[Dattorro, 2010] Dattorro, J. (2010). *Convex optimization & Euclidean distance geometry*. Lulu. com.

[De La Vega and Lueker, 1981] De La Vega, W. F. and Lueker, G. S. (1981). Bin packing can be solved within 1+ ε in linear time. *Combinatorica*, 1(4):349–355.

[Deng et al., 2020] Deng, L., Li, G., Han, S., Shi, L., and Xie, Y. (2020). Model compression and hardware acceleration for neural networks: A comprehensive survey. *Proceedings of the IEEE*, 108(4):485–532.

[Desaulniers et al., 2005] Desaulniers, G., Desrosiers, J., and Solomon, M. M., editors (2005). *Column Generation*. Number 978-0-387-25486-9 in Springer Books. Springer.

[Dijkstra, 1982] Dijkstra, E. W. (1982). Self-stabilization in spite of distributed control. In *Selected writings on computing: a personal perspective*, pages 41–46. Springer.

[Doerr et al., 2011] Doerr, B., Goldberg, L. A., Minder, L., Sauerwald, T., and Scheideler, C. (2011). Stabilizing consensus with the power of two choices. In *Proceedings of the twenty-third annual ACM symposium on Parallelism in algorithms and architectures*, pages 149–158.

[Donnelly and Welsh, 1983] Donnelly, P. and Welsh, D. (1983). Finite particle systems and infection models. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 94, pages 167–182. Cambridge University Press.

[Dreyer Jr and Roberts, 2009] Dreyer Jr, P. A. and Roberts, F. S. (2009). Irreversible k-threshold processes: Graph-theoretical threshold models of the spread of disease and of opinion. *Discrete Applied Mathematics*, 157(7):1615–1627.

[Du et al., 2019] Du, S. S., Hou, K., Salakhutdinov, R. R., Poczos, B., Wang, R., and Xu, K. (2019). Graph neural tangent kernel: Fusing graph neural networks with graph kernels. *Advances in neural information processing systems*, 32.

[Dubhashi and Panconesi, 2009] Dubhashi, D. P. and Panconesi, A. (2009). *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press.

[Duckworth and Wormald, 2006] Duckworth, W. and Wormald, N. C. (2006). On the independent domination number of random regular graphs. *Combinatorics, Probability and Computing*, 15(4):513–522.

[Dwaraki and Wolf, 2016] Dwaraki, A. and Wolf, T. (2016). Adaptive service-chain routing for virtual network functions in software-defined networks. In *Proceedings of the 2016 workshop on Hot topics in Middleboxes and Network Function Virtualization*, pages 32–37.

[Epstein and Levin, 2008] Epstein, L. and Levin, A. (2008). On bin packing with conflicts. *SIAM Journal on Optimization*, 19(3):1270–1298.

[Epstein et al., 2008] Epstein, L., Levin, A., and van Stee, R. (2008). Two-dimensional packing with conflicts. *Acta Informatica*, 45(3):155–175.

[Esser and May, 2019] Esser, A. and May, A. (2019). Low weight discrete logarithms and subset sum in $2^{0.65n}$ with polynomial memory. *Cryptology ePrint Archive*.

[Feinerman and Korman, 2013] Feinerman, O. and Korman, A. (2013). Theoretical distributed computing meets biology: A review. In *International Conference on Distributed Computing and Internet Technology*, pages 1–18. Springer.

[Feragen et al., 2013] Feragen, A., Kasenburg, N., Petersen, J., de Bruijne, M., and Borgwardt, K. (2013). Scalable kernels for graphs with continuous attributes. *Advances in neural information processing systems*, 26.

[Ferbach et al., 2022] Ferbach, D., Tsirigotis, C., Gidel, G., and Bose, A. J. (2022). A general framework for proving the equivariant strong lottery ticket hypothesis. *CoRR*, abs/2206.04270.

[Fischer and Burkholz, 2021] Fischer, J. and Burkholz, R. (2021). Towards strong pruning for lottery tickets with non-zero biases. *CoRR*, abs/2110.11150.

[Flocchini et al., 2004] Flocchini, P., Lodi, E., Luccio, F., Pagli, L., and Santoro, N. (2004). Dynamic monopolies in tori. *Discrete applied mathematics*, 137(2):197–212.

[Frankle and Carbin, 2018] Frankle, J. and Carbin, M. (2018). The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*.

[Gardner, 1970] Gardner, M. (1970). The fantastic combinations of john conway's new solitaire game'life. *Sc. Am.*, 223:20–123.

[Garey and Johnson, 1979a] Garey, M. R. and Johnson, D. S. (1979a). *Computers and intractability*, volume 174. freeman San Francisco.

[Garey and Johnson, 1979b] Garey, M. R. and Johnson, D. S. (1979b). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

[Gärtner et al., 2003] Gärtner, T., Flach, P., and Wrobel, S. (2003). On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*, pages 129–143. Springer.

[Gausseran et al., 2022] Gausseran, A., Alliche, R., Lesfari, H., Aparicio-Pardo, R., Giroire, F., and Moulerac, J. (2022). *When to Reconfigure my Network Slices ? A Deep Reinforcement Learning Approach*. PhD thesis, Université Côte d'Azur.

[Gausseran et al., 2020] Gausseran, A., Giroire, F., Jaumard, B., and Moulierac, J. (2020). Be scalable and rescue my slices during reconfiguration. In *IEEE ICC*.

[Gausseran et al., 2021] Gausseran, A., Giroire, F., Jaumard, B., and Moulierac, J. (2021). Be Scalable and Rescue My Slices During Reconfiguration. *The Computer Journal*, 64(10):1584–1599.

[Gausseran et al., 2021] Gausseran, A., Tomassilli, A., Giroire, F., and Moulierac, J. (2021). Don't interrupt me when you reconfigure my service function chains. *Computer Communications*.

[Gemmell and Johnston, 2001] Gemmell, P. and Johnston, A. M. (2001). Analysis of a subset sum randomizer. *IACR Cryptol. ePrint Arch.*, page 18.

[Gilmer et al., 2017] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

[Gorantla et al., 2019] Gorantla, S., Louis, A., Papadimitriou, C. H., Vempala, S., and Yadati, N. (2019). Biologically Plausible Neural Networks via Evolutionary Dynamics and Dopaminergic Plasticity. In *International Conference on Learning Representations (ICLR)*.

[Grohe, 2017] Grohe, M. (2017). *Descriptive complexity, canonisation, and definable graph structure theory*, volume 47. Cambridge University Press.

[Guan et al., 2020] Guan, W., Zhang, H., and Leung, V. C. M. (2020). Slice reconfiguration based on demand prediction with dueling deep reinforcement learning. In *IEEE GLOBECOM*.

[Guggiola and Semerjian, 2015] Guggiola, A. and Semerjian, G. (2015). Minimal contagious sets in random regular graphs. *Journal of Statistical Physics*, 158(2):300–358.

[Gureya et al., 2020] Gureya, D., Neto, J., Karimi, R., Barreto, J., Bhatotia, P., Quema, V., Rodrigues, R., Romano, P., and Vlassov, V. (2020). Bandwidth-aware page placement in numa. In *IEEE IPDPS*, pages 546–556. IEEE.

[Gutman and Trinajstić, 1972] Gutman, I. and Trinajstić, N. (1972). Graph theory and molecular orbitals. total φ-electron energy of alternant hydrocarbons. *Chemical Physics Letters*, 17(4):535–538.

[Halko et al., 2011] Halko, N., Martinsson, P.-G., and Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288.

[Hamilton et al., 2017a] Hamilton, W., Ying, Z., and Leskovec, J. (2017a). Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

[Hamilton et al., 2017b] Hamilton, W. L., Ying, R., and Leskovec, J. (2017b). Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

[Han et al., 2015] Han, S., Pool, J., Tran, J., and Dally, W. (2015). Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

[Harshaw et al., 2016] Harshaw, C. R., Bridges, R. A., Iannacone, M. D., Reed, J. W., and Goodall, J. R. (2016). Graphprints: Towards a graph analytic method for network anomaly detection. In *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, pages 1–4.

[Hart et al., 1968] Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107.

[Hartmanis, 1982] Hartmanis, J. (1982). Computers and intractability: a guide to the theory of np-completeness (michael r. garey and david s. johnson). *Siam Review*, 24(1):90.

[Harutyunyan et al., 2019] Harutyunyan, D., Fedrizzi, R., Shahriar, N., Boutaba, R., and Riggio, R. (2019). Orchestrating end-to-end slices in 5g networks. In *15th International Conference on Network and Service Management (CNSM)*.

[Hassin and Peleg, 1999] Hassin, Y. and Peleg, D. (1999). Distributed probabilistic polling and applications to proportionate agreement. In *International Colloquium on Automata, Languages, and Programming*, pages 402–411. Springer.

[Hayes, 2002] Hayes, B. (2002). The easiest hard problem. *American Scientist*, 90:113–117.

[Haynes, 2017] Haynes, T. (2017). *Domination in graphs: volume 2: advanced topics*. Routledge.

[Haynes et al., 2013] Haynes, T. W., Hedetniemi, S., and Slater, P. (2013). *Fundamentals of domination in graphs*. CRC press.

[Helm and May, 2018] Helm, A. and May, A. (2018). Subset sum quantumly in 1.17ˆ n. In *13th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[Herrera and Botero, 2016] Herrera, J. G. and Botero, J. F. (2016). Resource allocation in nfv: A comprehensive survey. *IEEE Transactions on Network and Service Management*, 13(3):518–532.

[Hindy et al., 2020] Hindy, H., Brosset, D., Bayne, E., Seeam, A., Tachtatzis, C., Atkinson, R., and Bellekens, X. (2020). A taxonomy of network threats and the effect of current datasets on intrusion detection systems. *IEEE Access*.

[Hoeffding, 1994] Hoeffding, W. (1994). Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*, pages 409–426. Springer.

[Hofmann et al., 2008] Hofmann, T., Schölkopf, B., and Smola, A. J. (2008). Kernel methods in machine learning. *The annals of statistics*, pages 1171–1220.

[Hoppen and Mansan, 2021] Hoppen, C. and Mansan, G. (2021). Asymptotic bounds on total domination in regular graphs. *Discrete Mathematics*, 344(4):112287.

[Huin et al., 2018] Huin, N., Jaumard, B., and Giroire, F. (2018). Optimal network service chain provisioning. *IEEE/ACM Transactions on Networking*.

[Jansen, 1999] Jansen, K. (1999). An approximation scheme for bin packing with conflicts. *Journal of combinatorial optimization*, 3(4):363–377.

[Jansen and Öhring, 1997] Jansen, K. and Öhring, S. (1997). Approximation algorithms for time constrained scheduling. *Information and computation*, 132(2):85–108.

[Janson, 2004] Janson, S. (2004). Large deviations for sums of partly dependent random variables: Large Deviations for Dependent Random Variables. *Random Structures & Algorithms*, 24(3):234–248.

[Jin et al., 2021] Jin, C., Vyas, N., and Williams, R. (2021). Fast low-space algorithms for subset sum. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1757–1776. SIAM.

[Jin and Wu, 2018] Jin, C. and Wu, H. (2018). A simple near-linear pseudopolynomial time randomized algorithm for subset sum. *arXiv preprint arXiv:1807.11597*.

[Johnson et al., 1991] Johnson, D. S., Aragon, C. R., McGeoch, L. A., and Schevon, C. (1991). Optimization by simulated annealing: an experimental evaluation ; part ii, graph coloring and number partitioning. *Operations research*, 39(3):378–406.

[Karagiannis et al., 2007] Karagiannis, T., Papagiannaki, K., Taft, N., and Faloutsos, M. (2007). Profiling the end host. In *International Conference on Passive and Active Network Measurement*, pages 186–196. Springer.

[Karmarkar and Karp, 1982] Karmarkar, N. and Karp, R. M. (1982). An efficient approximation scheme for the one-dimensional bin-packing problem. In *Symposium on Foundations of Computer Science*, pages 312–320. IEEE.

[Karmarkar et al., 1986] Karmarkar, N., Karp, R. M., Lueker, G. S., and Odlyzko, A. M. (1986). Probabilistic analysis of optimum partitioning. *Journal of Applied probability*, 23(3):626–645.

[Karp, 2011] Karp, R. M. (2011). Understanding science through the computational lens. *Journal of Computer Science and Technology*, 26(4):569–577.

[Kashima et al., 2003] Kashima, H., Tsuda, K., and Inokuchi, A. (2003). Marginalized kernels between labeled graphs. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 321–328.

[Kate and Goldberg, 2011] Kate, A. and Goldberg, I. (2011). Generalizing cryptosystems based on the subset sum problem. *Int. J. Inf. Sec.*, 10(3):189–199.

[Khatuya et al., 2018] Khatuya, S., Ganguly, N., Basak, J., Bharde, M., and Mitra, B. (2018). Adele: Anomaly detection from event log empiricism. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 2114–2122. IEEE.

[Kim and Feamster, 2013] Kim, H. and Feamster, N. (2013). Improving network management with software defined networking. *IEEE Communications Magazine*, 51(2):114–119.

[Kong et al., 2017] Kong, J., Kim, I., Wang, X., Zhang, Q., Cankaya, H. C., Xie, W., Ikeuchi, T., and Jue, J. P. (2017). Guaranteed-availability network function virtualization with network protection and vnf replication. In *IEEE GLOBECOM*, pages 1–6. IEEE.

[Krapivsky and Redner, 2003] Krapivsky, P. L. and Redner, S. (2003). Dynamics of majority rule in two-state interacting spin systems. *Physical Review Letters*, 90(23):238701.

[Kriege et al., 2016] Kriege, N. M., Giscard, P.-L., and Wilson, R. (2016). On valid optimal assignment kernels and applications to graph classification. *Advances in neural information processing systems*, 29.

[Kriege et al., 2020] Kriege, N. M., Johansson, F. D., and Morris, C. (2020). A survey on graph kernels. *Applied Network Science*, 5(1):1–42.

[Le et al., 2014] Le, Q. V., Sarlós, T., and Smola, A. J. (2014). Fastfood: Approximate kernel expansions in loglinear time. *arXiv preprint arXiv:1408.3060*.

[Le Bars and Kalogeratos, 2019] Le Bars, B. and Kalogeratos, A. (2019). A probabilistic framework to node-level anomaly detection in communication networks. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 2188–2196. IEEE.

[LeCun et al., 1989] LeCun, Y., Denker, J., and Solla, S. (1989). Optimal brain damage. *Advances in neural information processing systems*, 2.

[Lee et al., 2018] Lee, N., Ajanthan, T., and Torr, P. (2018). Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*.

[Leijnen and Veen, 2020] Leijnen, S. and Veen, F. v. (2020). The neural network zoo. *Multidisciplinary Digital Publishing Institute Proceedings*, 47(1):9.

[Lesfari and Giroire, 2022] Lesfari, H. and Giroire, F. (2022). Nadege: When graph kernels meet network anomaly detection. In *IEEE International Conference on Computer Communications (INFOCOM)*.

[Lesfari et al., 2021] Lesfari, H., Giroire, F., Di Lena, G., and Lac, C. (2021). A multidimensional colored packing approach for network slicing with dedicated protection. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pages 01–06. IEEE.

[Lesfari et al., 2022] Lesfari, H., Giroire, F., and Pérennes, S. (2022). Biased majority opinion dynamics: Exploiting graph k-domination. In *IJCAI 2022-International Joint Conference on Artificial Intelligence*.

[Levin and Peres, 2017] Levin, D. A. and Peres, Y. (2017). *Markov chains and mixing times*, volume 107. American Mathematical Soc.

[Levitin and Levitin, 2011] Levitin, A. and Levitin, M. (2011). *Algorithmic puzzles*. OUP USA.

[Lewis, 1983] Lewis, H. R. (1983). Computers and intractability. a guide to the theory of np-completeness.

[Li and Li, 2019] Li, Y. and Li, H. (2019). Improved quantum algorithm for the random subset sum problem. *arXiv preprint arXiv:1912.09264*.

[Liaskos et al., 2016] Liaskos, C., Kotronis, V., and Dimitropoulos, X. (2016). A novel framework for modeling and mitigating distributed link flooding attacks. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE.

[Liggett and Liggett, 1985] Liggett, T. M. and Liggett, T. M. (1985). *Interacting particle systems*, volume 2. Springer.

[Lin et al., 2017] Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., and Zhao, W. (2017). A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet of Things Journal*, 4(5):1125–1142.

[Liu et al., 2020] Liu, Y., Feng, G., Chen, Z., Qin, S., and Zhao, G. (2020). Network function migration in softwarization based networks with mobile edge computing. In *IEEE ICC*.

[Lueker, 1982] Lueker, G. S. (1982). On the average difference between the solutions to linear and integer knapsack problems. In *Applied Probability-Computer Science: The Interface Volume 1*, pages 489–504. Springer.

[Lueker, 1998a] Lueker, G. S. (1998a). Exponentially small bounds on the expected optimum of the partition and subset sum problems. *Random Structures & Algorithms*, 12(1):51–62. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291098-2418%28199801%2912%3A1%3C51%3A%3AAID-RSA3%3E3.0.CO%3B2-S.

[Lueker, 1998b] Lueker, G. S. (1998b). Exponentially small bounds on the expected optimum of the partition and subset sum problems. *Random Structures and Algorithms*, 12:51–62.

[Lynch, 1996] Lynch, N. A. (1996). *Distributed algorithms*. Elsevier.

[magazine, 2021] magazine, C. (2021). Annual report: Cyberwarfare in the c-suite. Technical report, Cybersecurity Ventures.

[Maimon and Rokach, 2005] Maimon, O. and Rokach, L. (2005). *Data mining and knowledge discovery handbook*. Springer.

[Malach et al., 2020] Malach, E., Yehudai, G., Shalev-Schwartz, S., and Shamir, O. (2020). Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*, pages 6682–6691. PMLR.

[man Jr et al., 1996] man Jr, E. C., Garey, M., and Johnson, D. (1996). Approximation algorithms for bin packing: A survey. *Approximation algorithms for NP-hard problems*, pages 46–93.

[Margulis, 1973] Margulis, G. A. (1973). Explicit constructions of concentrators. *Problemy Peredachi Informatsii*, 9(4):71–80.

[Maron et al., 2019] Maron, H., Ben-Hamu, H., Serviansky, H., and Lipman, Y. (2019). Provably powerful graph networks. *Advances in neural information processing systems*, 32.

[McKay, 1990] McKay, B. (1990). nauty user's guide (version 1.5). *Technical report*.

[Mertens, 1998] Mertens, S. (1998). Phase transition in the number partitioning problem. *Physical Review Letters*, 81:4281–4284.

[Mertens, 2001] Mertens, S. (2001). A physicist's approach to number partitioning. *Theor. Comput. Sci.*, 265(1-2):79–108.

[Mezard and Montanari, 2009] Mezard, M. and Montanari, A. (2009). *Information, physics, and computation*. Oxford graduate texts. Oxford University Press, Oxford ; New York. OCLC: ocn234430714.

[Mijumbi et al., 2015] Mijumbi, R., Serrat, J., Gorricho, J.-L., Bouten, N., De Turck, F., and Boutaba, R. (2015). Network function virtualization: State-of-the-art and research challenges. *IEEE Communications surveys & tutorials*, 18(1):236–262.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine learning*, volume 1. McGraw-hill New York.

[Mnih et al., 2015] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*.

[Morris et al., 2017] Morris, C., Kersting, K., and Mutzel, P. (2017). Glocalized weisfeiler-lehman graph kernels: Global-local feature maps of graphs. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 327–336. IEEE.

[Morris et al., 2019] Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. (2019). Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609.

[Morris, 2009] Morris, R. (2009). Minimal percolating sets in bootstrap percolation. *The Electronic Journal of Combinatorics*, 16(1):Research Paper 2,20.

[Morris, 2000] Morris, S. (2000). Contagion. *The Review of Economic Studies*, 67(1):57–78.

[Morrison and Noel, 2018] Morrison, N. and Noel, J. A. (2018). Extremal bounds for bootstrap percolation in the hypercube. *Journal of Combinatorial Theory, Series A*, 156:61–84.

[Mörters and Peres, 2010] Mörters, P. and Peres, Y. (2010). *Brownian motion*, volume 30. Cambridge University Press.

[Mossel and Tamuz, 2017] Mossel, E. and Tamuz, O. (2017). Opinion exchange dynamics. *Probability Surveys*, 14:155–204.

[Mozer and Smolensky, 1988] Mozer, M. C. and Smolensky, P. (1988). Skeletonization: A technique for trimming the fat from a network via relevance assessment. *Advances in neural information processing systems*, 1.

[Muandet et al., 2016] Muandet, K., Fukumizu, K., Sriperumbudur, B., and Schölkopf, B. (2016). Kernel mean embedding of distributions: A review and beyond. *arXiv preprint arXiv:1605.09522*.

[Mukhopadhyay et al., 2020] Mukhopadhyay, A., Mazumdar, R. R., and Roy, R. (2020). Voter and majority dynamics with biased and stubborn agents. *Journal of Statistical Physics*, 181(4):1239–1265.

[Musco et al., 2017] Musco, C., Su, H.-H., and Lynch, N. A. (2017). Ant-inspired density estimation via random walks. *Proceedings of the National Academy of Sciences*, 114(40):10534–10541.

[Natale, 2017] Natale, E. (2017). *On the computational power of simple dynamics*. PhD thesis, Sapienza University of Rome.

[Neuhaus and Bunke, 2007a] Neuhaus, M. and Bunke, H. (2007a). Automatic learning of cost functions for graph edit distance. *Information Sciences*, 177(1):239–247.

[Neuhaus and Bunke, 2007b] Neuhaus, M. and Bunke, H. (2007b). *Bridging the gap between graph edit distance and kernel machines*, volume 68. World Scientific.

[NFV ISG, 2013] NFV ISG (2013). Network functions virtualisation (nfv) - network operator perspectives on industry progress. *Update White Paper*.

[Nikolentzos et al., 2021] Nikolentzos, G., Siglidis, G., and Vazirgiannis, M. (2021). Graph kernels: A survey. *Journal of Artificial Intelligence Research*, 72:943–1027.

[Noghani et al., 2019] Noghani, K. A., Kassler, A., and Taheri, J. (2019). On the cost-optimality trade-off for service function chain reconfiguration. In *IEEE CloudNet*.

[Orlowski et al., 2010] Orlowski, S., Wessäly, R., Pióro, M., and Tomaszewski, A. (2010). SNDlib 1.0–survivable network design library. *Wiley Networks*.

[Orseau et al., 2020] Orseau, L., Hutter, M., and Rivasplata, O. (2020). Logarithmic pruning is all you need. *Advances in Neural Information Processing Systems*, 33:2925–2934.

[Out and N. Zehmakan, 2021] Out, C. and N. Zehmakan, A. (2021). Majority vote in social networks: Make random friends or be stubborn to overpower elites. In *30th International Joint Conference on Artificial Intelligence (IJCAI 2021)*, pages 349–355. International Joint Conferences on Artificial Intelligence Organization.

[Panafieu et al., 2019] Panafieu, É. d., Lamali, M. L., and Wallner, M. (2019). Combinatorics of nondeterministic walks of the dyck and motzkin type. In *2019 Proceedings of the Sixteenth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, pages 1–12. SIAM.

[Peleg, 2002] Peleg, D. (2002). Local majorities, coalitions and monopolies in graphs: a review. *Theoretical Computer Science*, 282:231–257.

[Pendlebury et al., 2019] Pendlebury, F., Pierazzi, F., Jordaney, R., Kinder, J., and Cavallaro, L. (2019). Tesseract: Eliminating experimental bias in malware classification across space and time. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 729–746.

[Pensia et al., 2020a] Pensia, A., Rajput, S., Nagle, A., Vishwakarma, H., and Papailiopoulos, D. S. (2020a). Optimal lottery tickets via subset sum: Logarithmic over-parameterization is sufficient. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

[Pensia et al., 2020b] Pensia, A., Rajput, S., Nagle, A., Vishwakarma, H., and Papailiopoulos, D. S. (2020b). Optimal lottery tickets via subset sum: Logarithmic over-parameterization is sufficient. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

[Pete, 1997] Pete, G. (1997). *Disease processes and bootstrap percolation*. PhD thesis, Thesis for diploma at the Bolyai Institute, Jzsef Attila University, Szeged.

[Pippenger, 1977] Pippenger, N. (1977). Superconcentrators. *SIAM Journal on Computing*, 6(2):298–304.

[Pozza et al., 2020] Pozza, M., Nicholson, P. K., Lugones, D. F., Rao, A., Flinck, H., and Tarkoma, S. (2020). On reconfiguring 5g network slices. *IEEE JSAC*.

[Przykucki and Shelton, 2020] Przykucki, M. and Shelton, T. (2020). Smallest percolating sets in bootstrap percolation on grids. *The Electronic Journal of Combinatorics*.

[Rabin, 1983] Rabin, M. O. (1983). Randomized byzantine generals. In *24th annual symposium on foundations of computer science (sfcs 1983)*, pages 403–409. IEEE.

[Ramanujan et al., 2020] Ramanujan, V., Wortsman, M., Kembhavi, A., Farhadi, A., and Rastegari, M. (2020). What's hidden in a randomly weighted neural network ? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11893–11902.

[Randic, 1975] Randic, M. (1975). Characterization of molecular branching. *Journal of the American Chemical Society*, 97(23):6609–6615.

[Read and Corneil, 1977] Read, R. C. and Corneil, D. G. (1977). The graph isomorphism disease. *Journal of graph theory*, 1(4):339–363.

[Riedl, 2010] Riedl, E. (2010). Largest minimal percolating sets in hypercubes under 2-bootstrap percolation. *The Electronic Journal of Combinatorics*, 17(1).

[Riedl, 2012] Riedl, E. (2012). Largest and smallest minimal percolating sets in trees. *The Electronic Journal of Combinatorics*.

[Rousseeuw and Driessen, 1999] Rousseeuw, P. J. and Driessen, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223.

[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

[Ruml et al., 1996] Ruml, W., Ngo, J. T., Marks, J., and Shieber, S. M. (1996). Easily searched encodings for number partitioning. *Journal of Optimization Theory and Applications*, 89(2):251–291.

[Saigo et al., 2009] Saigo, H., Nowozin, S., Kadowaki, T., Kudo, T., and Tsuda, K. (2009). gboost: a mathematical programming approach to graph classification and regression. *Machine Learning*, 75(1):69–89.

[Sallam et al., 2019] Sallam, G., Zheng, Z., and Ji, B. (2019). Placement and allocation of virtual network functions: Multi-dimensional case. In *IEEE ICNP*, pages 1–11. IEEE.

[Sanfeliu and Fu, 1983] Sanfeliu, A. and Fu, K.-S. (1983). A distance measure between attributed relational graphs for pattern recognition. *IEEE transactions on systems, man, and cybernetics*, pages 353–362.

[Savi et al., 2015] Savi, M., Tornatore, M., and Verticale, G. (2015). Impact of processing costs on service chain placement in network functions virtualization. In *IEEE Conference NFV-SDN*.

[Schoenberg, 1938] Schoenberg, I. J. (1938). Metric spaces and positive definite functions. *Transactions of the American Mathematical Society*, 44(3):522–536.

[Schölkopf, 2000] Schölkopf, B. (2000). The kernel trick for distances. *Advances in neural information processing systems*, 13.

[Schölkopf et al., 1998] Schölkopf, B., Smola, A., and Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319.

[Seiden, 2002] Seiden, S. S. (2002). On the online bin packing problem. *Journal of the ACM*, 49(5):640–671.

[Sharma et al., 2020] Sharma, S., Gumaste, A., and Tatipamula, M. (2020). Dynamic network slicing using utility algorithms and stochastic optimization. In *IEEE HPSR*.

[Shervashidze and Borgwardt, 2009] Shervashidze, N. and Borgwardt, K. (2009). Fast subtree kernels on graphs. *Advances in neural information processing systems*, 22.

[Shervashidze et al., 2011] Shervashidze, N., Schweitzer, P., Van Leeuwen, E. J., Mehlhorn, K., and Borgwardt, K. M. (2011). Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9).

[Shervashidze et al., 2009] Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., and Borgwardt, K. (2009). Efficient graphlet kernels for large graph comparison. In *Artificial Intelligence and Statistics*, pages 488–495.

[Siglidis et al., 2020] Siglidis, G., Nikolentzos, G., Limnios, S., Giatsidis, C., Skianis, K., and Vazirgiannis, M. (2020). Grakel: A graph kernel library in python. *Journal of Machine Learning Research*, 21(54):1–5.

[Smola et al., 2007] Smola, A., Gretton, A., Song, L., and Schölkopf, B. (2007). A hilbert space embedding for distributions. In *International Conference on Algorithmic Learning Theory*, pages 13–31. Springer.

[Srinivas et al., 2019] Srinivas, J., Das, A. K., and Kumar, N. (2019). Government regulations in cyber security: Framework, standards and recommendations. *Future Generation Computer Systems*, 92:178–188.

[Sumpter et al., 2008] Sumpter, D. J., Krause, J., James, R., Couzin, I. D., and Ward, A. J. (2008). Consensus decision making by fish. *Current Biology*, 18(22):1773–1777.

[Sun and Das, 2019] Sun, S. and Das, K. C. (2019). On the second largest normalized laplacian eigenvalue of graphs. *Applied Mathematics and Computation*, 348:531–541.

[Sun, 2003] Sun, Z.-W. (2003). Unification of zero-sum problems, subset sums and covers of z. *Electronic Research Announcements of The American Mathematical Society*, 9:51–60.

[Szabó and Sriperumbudur, 2017] Szabó, Z. and Sriperumbudur, B. K. (2017). Characteristic and universal tensor product kernels. *The Journal of Machine Learning Research*, 18(1):8724–8752.

[Todeschini and Consonni, 2008] Todeschini, R. and Consonni, V. (2008). *Handbook of molecular descriptors*, volume 11. John Wiley & Sons.

[Tomassilli et al., 2021] Tomassilli, A., Di Lena, G., Giroire, F., Tahiri, I., Saucez, D., Pérennes, S., Turletti, T., Sadykov, R., Vanderbeck, F., and Lac, C. (2021). Design of robust programmable networks with bandwidth-optimal failure recovery scheme. *Computer Networks*, 192:108043.

[Tomassilli et al., 2018a] Tomassilli, A., Giroire, F., Huin, N., and Pérennes, S. (2018a). Provably efficient algorithms for placement of service function chains with ordering constraints. In *IEEE INFOCOM*.

[Tomassilli et al., 2018b] Tomassilli, A., Huin, N., Giroire, F., and Jaumard, B. (2018b). Resource requirements for reliable service function chaining. In *IEEE ICC*, pages 1–7. IEEE.

[Troia et al., 2019] Troia, S., Alvizu, R., and Maier, G. (2019). Reinforcement learning for service function chain reconfiguration in nfv-sdn metro-core optical networks. *IEEE Access*.

[Valiant, 1975] Valiant, L. G. (1975). On non-linear lower bounds in computational complexity. In *Proceedings of the seventh annual ACM symposium on Theory of computing*, pages 45–53.

[Vert, 2008] Vert, J.-P. (2008). The optimal assignment kernel is not positive definite. *arXiv preprint arXiv:0801.4061*.

[Vishwanathan et al., 2008] Vishwanathan, S., Borgwardt, K. M., Kondor, I. R., and Schraudolph, N. N. (2008). Graph kernels. *arXiv preprint arXiv:0807.0093*.

[Vishwanathan et al., 2010] Vishwanathan, S. V. N., Schraudolph, N. N., Kondor, R., and Borgwardt, K. M. (2010). Graph kernels. *The Journal of Machine Learning Research*, 11:1201–1242.

[Wagner and Fischer, 1974] Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1):168–173.

[Wang et al., 2021] Wang, C., Deng, J., Meng, X., Wang, Y., Li, J., Lin, S., Han, S., Miao, F., Rajasekaran, S., and Ding, C. (2021). A secure and efficient federated learning framework for NLP. In Moens, M., Huang, X., Specia, L., and Yih, S. W., editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7676–7682. Association for Computational Linguistics.

[Wang et al., 2019] Wang, G., Feng, G., Quek, T. Q. S., Qin, S., Wen, R., and Tan, W. (2019). Reconfiguration in network slicing—optimizing the profit and performance. *IEEE TNSM*.

[Wang et al., 2017] Wang, Y., Ramon, J., and Guo, Z.-C. (2017). Learning from networked examples. *arXiv:1405.2600 [cs, stat]*. arXiv: 1405.2600.

[Warnke, 2016] Warnke, L. (2016). On the method of typical bounded differences. *Combinatorics, Probability and Computing*, 25(2):269–299.

[Watkins and Dayan, 1992] Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*.

[Wei et al., 2020] Wei, F., Feng, G., Sun, Y., Wang, Y., Qin, S., and Liang, Y. C. (2020). Network slice reconfiguration by exploiting deep reinforcement learning with large action space. *IEEE Transactions on Network and Service Management*.

[Wiener, 1947] Wiener, H. (1947). Structural determination of paraffin boiling points. *Journal of the American chemical society*, 69(1):17–20.

[Winkler, 2003] Winkler, P. (2003). *Mathematical puzzles: a connoisseur's collection*. CRC Press.

[Woeginger, 1997] Woeginger, G. J. (1997). There is no asymptotic ptas for two-dimensional vector packing. *Information Processing Letters*, 64(6):293–297.

[Wolfram, 1984] Wolfram, S. (1984). Cellular automata as models of complexity. *Nature*, 311(5985):419–424.

[Wu et al., 2020] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.

[Xu et al., 2018] Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2018). How powerful are graph neural networks ? *arXiv preprint arXiv:1810.00826*.

[Yan and Han, 2002] Yan, X. and Han, J. (2002). gspan: Graph-based substructure pattern mining. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 721–724. IEEE.

[Yao et al., 2019] Yao, Y., Su, L., Zhang, C., Lu, Z., and Liu, B. (2019). Marrying graph kernel with deep neural network: A case study for network anomaly detection. In *International Conference on Computational Science*, pages 102–115. Springer.

[Ye et al., 2019] Ye, W., Wang, Z., Redberg, R., and Singh, A. (2019). Tree++: Truncated tree based graph kernels. *IEEE Transactions on Knowledge and Data Engineering*.

[Yi et al., 2015] Yi, S., Qin, Z., and Li, Q. (2015). Security and privacy issues of fog computing: A survey. In *International conference on wireless algorithms, systems, and applications*, pages 685–695. Springer.

[Zehmakan, 2021] Zehmakan, A. N. (2021). Majority opinion diffusion in social networks: An adversarial approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5611–5619.

[Zhang et al., 2017] Zhang, N., Liu, Y.-F., Farmanbar, H., Chang, T.-H., Hong, M., and Luo, Z.-Q. (2017). Network slicing for service-oriented networks under resource constraints. *IEEE JSAC*.

[Zhang et al., 2020] Zhang, Z., Cui, P., and Zhu, W. (2020). Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

[Zhang et al., 2018] Zhang, Z., Wang, M., Xiang, Y., Huang, Y., and Nehorai, A. (2018). Retgk: Graph kernels based on return probabilities of random walks. *Advances in Neural Information Processing Systems*, 31.

[Zuckerman, 2006] Zuckerman, D. (2006). Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 681–690.

Appendix

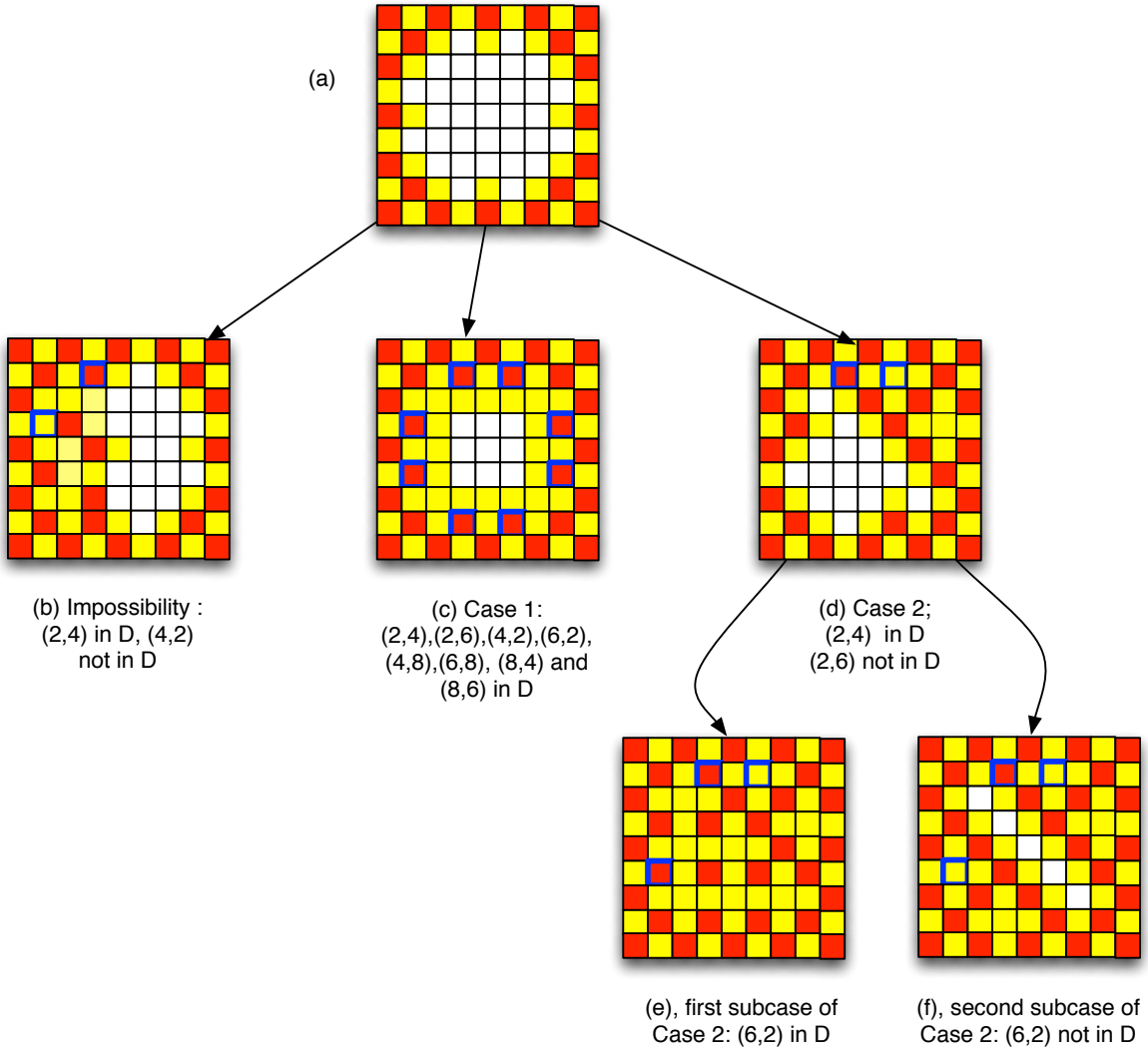


Figure A.1 – Illustrations of Proposition A.1 ($n = 9$). Red vertices are the one that are assumed to belong to the lethal set D of size 33, yellow vertices are assumed to belong to $V \setminus D$ and white vertices are undetermined yet. Vertices (squares) surrounded by blue correspond to the hypotheses done in the different cases of the proof. Figure (a) is a configuration that is certain if $|D| = 33$. Figure (b) is the configuration obtained if $(2, 4) \in D$ and $(4, 2) \notin D$ and leads to a contradiction (Claim A.3). Figure (c) corresponds to the first case and leads to a contradiction. Figure (d) corresponds to the second case. Figure (e) (resp., (f)) to the subcase $(6, 2) \in D$ (resp. $(6, 2) \notin D$) leading to a contradiction in both subcases.

A Chapter 4

We provide hereafter an ad-hoc combinatorial proof that $s_3(G_{9\times 9}) = LB_9 + 1$. We first state a proposition which gives more properties of the vertices of D and $V \setminus D$ when $n \equiv 1$ or $3 \pmod 6$.

Property 7. *Let $n \equiv 1$ or $3 \pmod 6$ and let D be a lethal set of size $LB_n = (n^2 + 2n)/3$; then a) there are no two adjacent vertices in D, and*

b) a vertex in $V \setminus D$ cannot have 4 neighbours in D.

Proof.

We reconsider the proof of Theorem 4.2.4 with a more careful analysis. If two vertices of D are adjacent, their perimeter is 6 and not 8 and so the initial perimeter is at most $4|D| - 2$. So we get $4|D| - 2 - 2(n^2 - |D|) \geq 4n$ and so $|D| \geq (n^2 + 2n + 1)/3 > LB_n$ and that proves property a). If a vertex in $V \setminus D$ has 4 neighbours in D, then when this vertex is infected the perimeter decreases by 4 and not 2. So in that case we get $4|D| - 2(n^2 - |D|) - 2 \geq 4n$ and so $|D| \geq (n^2 + 2n + 1)/3 > LB_n$ and that proves property b).

□

Proposition A.1. *We have $s_3(G_{9\times 9}) = LB_9 + 1 = 34$.*

Proof.

The upper bounds come from Theorem 4.3.6. To prove the proposition, by Theorem 4.2.4, it is sufficient to show that there is no lethal set for $G_{9\times 9}$ with 33 vertices.

Hence, for purpose of contradiction, let us assume that there exists a lethal set D of size 33 in the grid $G_{9\times 9}$. The basic idea of the proof is mainly to use Properties 5, 6 and 7 in order to progressively show that some vertices are forced to belong to D (vertices in red in Figure A.1) or to $V \setminus D$ (vertices in yellow in Figure A.1). There are few cases to be considered.

By Property 5 and 7, $D \cap \mathcal{B} = \{(1,1),(1,3),(1,5),(1,7),(1,9),\ (3,9),(5,9),(7,9),(9,9),\ (9,7),(9,5),(9,3),(9,1),(7,1),(5,1),(3,1)\}$. Then by Property 7(a), $(2,3),(2,5),(2,7),(3,8),\ (5,8),(7,8),(8,7),(8,5),(8,3),(7,2),(5,2),(3,2)$ are not in D. Then, Property 6 implies that $(2,2) \in D$ (otherwise there will the path $(1,2),(2,2),(2,1)$ in $V \setminus D$); similarly $(2,8),(8,2),(8,8) \in D$. So far, we may assume that D contains the vertices described in red and does not contain the ones in yellow as shown in Figure A.1(a).

Let us now prove two claims very useful for the proof.

Claim A.2. *At least one vertex in each of the pairs $\{(2,4),(2,6)\}$ $\{(4,2),(6,2)\}$, $\{(4,8),(6,8)\}$ and $\{(8,4),(8,6)\}$ belongs to D.*

Proof.

Suppose it is not true and that for example $(2,4)$ and $(2,6)$ are in $V \setminus D$. Then there will be the path $(1,4),(2,4),(2,5),(2,6),(1,6)$ in $V \setminus D$, contradicting Property 6.

□

Claim A.3. *For each pair $\{(2,4),(4,2)\}$ $\{(6,2),(8,4)\}$, $\{(2,6),(4,8)\}$ and $\{(6,8),(8,6)\}$, the two vertices of this pair have the same status (either both in D or both in $V \setminus D$).*

Proof.

For purpose of contradiction, let us assume by symmetry that $(2,4) \in D$ and $(4,2) \notin D$ (see Figure A.1(b)). By Property 7(b), vertex $(3,3) \notin D$ (otherwise the vertex $(3,2)$ which belongs to $V \setminus D$ will have 4 neighbours in D). Then, Property 6 implies that vertex $(4,3) \in D$ (otherwise $V \setminus D$ would contain a 4-cycle). Moreover, by Claim A.2, as $(4,2) \notin D$, then $(6,2) \in D$. Now, by Property 7(a), vertices $(3,4),(4,4),(5,3)$ and $(6,3)$ are not in

D. Then, Property 6 implies $(5,4) \in D$ (since otherwise we will have in $V \setminus D$ the 8-cycle $(3,2),(3,3),(3,4),(4,4),(5,4),(5,3),(5,2),(4,2)$). Then, by Property 7(a), we have $(5,5),(6,4) \notin D$. Then, by Property 7(b) $(7,3) \notin D$ (since otherwise $(7,2)$ would be a vertex in $V \setminus D$ with four neighbours in D). Finally, Property 6 implies $(7,4) \in D$ (otherwise $V \setminus D$ contains the 4-cycle $((7,4),(7,3),(6,3),(6,4))$) and $(8,4) \in D$ otherwise $V \setminus D$ contains a path from $(4,1)$ to $(9,4)$. But then, there are two adjacent vertices in D namely $(7,4)$ and $(8,4)$, contradicting Property 7(a) (see Figure A.1(b)).

□

Now, let us come back to the partial configuration described in Figure A.1(a). Recall that, red vertices are known to belong to D and yellow vertices do not belong to D.

By Claim A.2, at least one vertex in each of the pairs $\{(2,4),(2,6)\}$ $\{(4,2),(6,2)\}$, $\{(4,8),(6,8)\}$ and $\{(8,4),(8,6)\}$ belongs to D. We consider two cases

— Case 1: all vertices $(2,4),(2,6),(4,2),(6,2),(4,8),(6,8),(8,4),(8,6)$ belong to D (see Figure A.1(c)).

By Property 7(a), this implies that vertices $(3,4),(3,6),(4,7),(6,7),(7,6),(7,4)$, $(6,3),(4,3)$ are not in D. Then, vertices $(3,3),(3,5),(3,7),(5,3),(5,7),(7,3),(7,5),(7,7)$ are not in D by Property 7(b). But, then there exists a 16-cycle $(3,3),(3,4),(3,5),(3,6),(3,7),(4,7)$, $(5,7),(6,7),(7,7),(7,6),(7,5),(7,4),(7,3),(6,3),(5,3),(4,3)$ in $V \setminus D$, a contradiction.

— case 2 : In at least one of the pairs $\{(2,4),(2,6)\}$ $\{(4,2),(6,2)\}$, $\{(4,8),(6,8)\}$ $\{(8,4),(8,6)\}$ one vertex belongs to D and the other not. By symmetry suppose $(2,4) \in D$ and $(2,6) \notin D$.

By Claim A.3, as $(2,4) \in D$, then $(4,2) \in D$. By Property 7(a) $(3,4) \notin D$ and $(4,3) \notin D$. By Claim A.3, as $(2,6) \notin D$, then $(4,8) \notin D$. Hence, by Claim A.2, $(6,8) \in D$ and again by Claim A.3, $(8,6) \in D$. Moreover, by Property 7(a) $(6,7)$ and $(7,6)$ are not in D. Now, Property 6 implies $(3,7) \in D$, since otherwise there would be a path from $(1,6)$ to $(4,9)$ in $V \setminus D$. Then, Property 7(a) implies that $(3,6)$ and $(4,7) \notin D$. Then, by Property 6, $(4,6) \in D$ again to avoid a path from $(1,6)$ to $(4,9)$ in $V \setminus D$ and then, by Property 7(a) $(4,5)$ and $(5,6) \notin D$. Then, vertices $(3,5)$ and $(5,7)$ need to be in D to avoid 4-cycle in $V \setminus D$. See Figure A.1(d) for the configuration reached so far. There are two subcases to be considered.

— Case $(6,2) \in D$ (see Figure A.1(e)). By Claim A.3, $(8,4) \in D$ and so by Property 7(a), $(6,3)$ and $(7,4) \notin D$. Then, Property 7(b), $(7,3) \notin D$ (otherwise $(8,3)$ would have 4 neighbours in D). Then $(6,4)$ must be in D to avoid a 4-cycle in $V \setminus D$ and so, by Property 7(a), $(5,4),(6,5) \notin D$ to avoid adjacent vertices in D. At this point, D already contains 31 vertices. Then to avoid 4-cycles in $V \setminus D$ either $(4,4) \in D$ or both $(3,3)$ and $(5,3) \in D$; similarly either $(6,6) \in D$ or both $(5,5)$ and $(7,5) \in D$. Therefore, to get $|D| = 33$ we need to have $(4,4) \in D$ and $(6,6) \in D$ and the other vertices $(5,3),(5,5),(7,5) \notin D$. But, then there is a 8-cycle in $V \setminus D$ $((5,3),(5,4),(5,5),(6,5),(7,5),(7,4),(7,3),(6,3))$, a contradiction.

— Case $(6,2) \notin D$ (See Figure A.1(f)). By Claim A.3, $(8,4) \notin D$. Hence $(7,3) \in D$ to avoid a path from $(6,1)$ to $(9,4)$ in $V \setminus D$. Then, by Property 7(a), $(6,3)$ and $(7,4) \notin D$. Hence, $(5,3),(7,5) \in D$ to avoid 4-cycles in $V \setminus D$, and $(6,4) \in D$ to avoid a path from $(6,1)$ to $(9,4)$ in $V \setminus D$. Moreover, by Property 7(a), $(5,4)$ and

$(6,5) \notin D$ to avoid adjacent vertices in D. There are already 32 vertices in D and the single remaining (undetermined yet) vertex of D is not sufficient to break all 4-cycles in $V \setminus D$, a contradiction.

□

B Chapter 6

Tools

Below we prove some inequalities we use and some omitted proofs.

Claim B.1. *Let $d, n \in \mathbb{N}$ and $\alpha \in \mathbb{R}_{>0}$. If $n \geq \frac{68d}{\alpha}$ and $\alpha \leq \frac{1}{6\sqrt{d}}$, then*

$$e^{\frac{4d}{\alpha n}} \cdot \frac{1}{\left(1-4\alpha^2\right)^{\frac{d}{2}}} \leq 1 + \frac{1}{8}.$$

Proof.

Since $e^x \leq (1-x)^{-1}$ for $x \leq 1$, for $n \geq \frac{4d}{\alpha}$, it holds that

$$e^{\frac{4d}{\alpha n}} \leq \frac{1}{1-\frac{4d}{\alpha n}} = 1 + \frac{4d}{\alpha n - 4d}.$$

Thus, having $n \geq \frac{68d}{\alpha}$ implies that

$$e^{\frac{4d}{\alpha n}} \leq 1 + \frac{1}{16}.$$

Moreover, by Bernoulli's inequality, since $\alpha < \frac{1}{2}$, it holds that,

$$\frac{1}{\left(1-4\alpha^2\right)^{\frac{d}{2}}} \leq \frac{1}{1-2d\alpha^2}.$$

Altogether, we need that

$$\frac{1+\frac{1}{16}}{1-2d\alpha^2} \leq 1 + \frac{1}{8},$$

which holds for $\alpha \leq \frac{1}{6\sqrt{d}}$.

□

Claim B.2. *Let $d, n \in \mathbb{N}$, $\varepsilon \in (0,1)$, and $\alpha \in (0, \frac{1}{6})$. If $n \geq \frac{144d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)$, then*

$$\frac{4e^{\frac{4d}{\alpha n}}}{2^{\frac{\alpha^2 n}{6}}} \cdot \left(\frac{\pi\alpha n}{2\varepsilon^2}\right)^{\frac{d}{2}} \leq \varepsilon.$$

Proof.

Consider the function

$$f(n) = \frac{n^d}{2^{\frac{\alpha^2 n}{6}}}.$$

We have that

$$\begin{aligned} f'(n) &= \frac{dn^{d-1}2^{\frac{\alpha^2 n}{6}} - \frac{\alpha^2 \ln 2}{6} \cdot n^d 2^{\frac{\alpha^2 n}{6}}}{2^{\frac{\alpha^2 n}{3}}} \\ &= \frac{n^{d-1}2^{\frac{\alpha^2 n}{6}}}{2^{\frac{\alpha^2 n}{3}}} \cdot \left(d - \frac{\alpha^2 n \ln 2}{6}\right), \end{aligned}$$

and, hence, f is non-increasing for $n \geq \frac{6d}{\alpha^2 \ln 2}$. Thus, since $f\left(\frac{6d}{\alpha^2 \ln 2}\right) = \left(\frac{6d}{e\alpha^2 \ln 2}\right)^d$, it holds that

$$\begin{aligned} \frac{4e^{\frac{4d}{\alpha n}}}{2^{\frac{\alpha^2 n}{6}}} \cdot \left(\frac{\pi\alpha n}{2\varepsilon^2}\right)^{\frac{d}{2}} &= \frac{4e^{\frac{4d}{\alpha n}}}{2^{\frac{\alpha^2 n}{12}}} \cdot \left(\frac{\pi\alpha}{2\varepsilon^2}\right)^{\frac{d}{2}} \sqrt{\frac{n^d}{2^{\frac{\alpha^2 n}{6}}}} \\ &\leq \frac{4e^{\frac{4d}{\alpha n}}}{2^{\frac{\alpha^2 n}{12}}} \cdot \left(\frac{\pi\alpha}{2\varepsilon^2}\right)^{\frac{d}{2}} \sqrt{\left(\frac{6d}{e\alpha^2 \ln 2}\right)^d} \\ &= \frac{4e^{\frac{4d}{\alpha n}}}{2^{\frac{\alpha^2 n}{12}}} \cdot \left(\frac{3\pi d}{\varepsilon^2 e\alpha \ln 2}\right)^{\frac{d}{2}} \\ &< \frac{8}{2^{\frac{\alpha^2 n}{12}}} \cdot \left(\frac{6d}{\varepsilon^2 \alpha}\right)^{\frac{d}{2}} \end{aligned}$$

where the last inequality comes from noting that $6 > \frac{3\pi}{e \ln 2}$ and that $n \geq \frac{8d}{\alpha}$ implies $e^{\frac{4d}{\alpha n}} < 2$. This is at most ε if

$$\frac{8}{\varepsilon^{d+1}} \cdot \left(\frac{6d}{\alpha}\right)^{\frac{d}{2}} \leq 2^{\frac{\alpha^2 n}{12}},$$

or, equivalently,

$$n \geq \frac{12}{\alpha^2}\left(\log 8 + (d+1)\log\frac{1}{\varepsilon} + \frac{d}{2}\log\frac{1}{\alpha} + \frac{d}{2}\log 6d\right).$$

The thesis follows from the bounds $d, n \geq 1$, $\varepsilon \in (0,1)$, and $\alpha < \frac{1}{6}$.

□

Claim B.3. *Let* A, B *be two centred normal random variables, and let* $\varphi_{\mathrm{B}}(x)$ *be the density function of* b*. Then, for any* $z \in \mathbb{R}$*, for any* $\varepsilon > 0$*, it holds that*

$$\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\left[\Pr\left[\mathrm{A} \in (z - x - \varepsilon, z - x + \varepsilon)\right]\right]^2 \mathrm{d}x \leq \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\left[\Pr\left[\mathrm{A} \in (-x - \varepsilon, -x + \varepsilon)\right]\right]^2 \mathrm{d}x.$$

Proof.

For any $x, z \in \mathbb{R}$, let

$$h(x,z) = \varphi_{\mathrm{B}}(x) \left[\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\right]^2 \mathrm{d}x,$$

and let

$$H(z) = \int_{\mathbb{R}} h(x,z)\,\mathrm{d}x.$$

Let $\varphi_{\mathrm{A}}(x)$ be the density function of a. Since

$$\begin{aligned}\left|\frac{\partial h(x,z)}{\partial z}\right| &= 2\left|\varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon) - \varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\right| \\ &\leq 2\varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon) + \varphi_{\mathrm{A}}(z-x-\varepsilon)\right),\end{aligned}$$

$h(x,z)$ meets the hypothesis of the Leibniz integral rule and we can write

$$\begin{aligned}\frac{\mathrm{d}H(z)}{\mathrm{d}z} &= \int_{\mathbb{R}} \frac{\partial h(x,z)}{\partial z}\,\mathrm{d}x \\ &= 2\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon) - \varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\,\mathrm{d}x\end{aligned}$$

If we prove that such a function is zero in $z = 0$, positive for $z < 0$ and negative for $z > 0$, then we have that the maximum of H is reached in $z = 0$.

First case: $z = 0$**.** Then

$$\begin{aligned}\frac{\mathrm{d}H(0)}{\mathrm{d}z} &= 2\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (x-\varepsilon, x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(x-\varepsilon) - \varphi_{\mathrm{A}}(x+\varepsilon)\right)\,\mathrm{d}x \qquad \text{(B.1)} \\ &= 2\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (x-\varepsilon, x+\varepsilon)\right]\varphi_{\mathrm{A}}(x-\varepsilon)\,\mathrm{d}x \\ &\quad - 2\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (x-\varepsilon, x+\varepsilon)\right]\varphi_{\mathrm{A}}(x+\varepsilon)\,\mathrm{d}x \\ &= 2\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (x-\varepsilon, x+\varepsilon)\right]\varphi_{\mathrm{A}}(x-\varepsilon)\,\mathrm{d}x \\ &\quad - 2\int_{\mathbb{R}} \varphi_{\mathrm{B}}(y)\Pr\left[\mathrm{A} \in (y-\varepsilon, y+\varepsilon)\right]\varphi_{\mathrm{A}}(y-\varepsilon)\,\mathrm{d}x \qquad \text{(B.2)} \\ &= 0,\end{aligned}$$

where in Equation (B.1) we exploited the symmetry of the integrand functions, Equation (B.2) we substituted in the second integral $y = -x$ and used again symmetry.

Second case: $z > 0$**.** Then

$$\begin{aligned}
&\frac{\mathrm{d}H(z)}{\mathrm{d}z} \\
&= 2\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&= 2\int_{-\infty}^{-z} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&\quad + 2\int_{-z}^{+z} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&\quad + 2\int_{+z}^{+\infty} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&= 2\int_{+z}^{+\infty} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z+x-\varepsilon, z+x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z+x+\varepsilon)-\varphi_{\mathrm{A}}(z+x-\varepsilon)\right)\mathrm{d}x \qquad \text{(B.3)} \\
&\quad + 2\int_{+3z}^{+\infty} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&\quad + 2\int_{+z}^{+3z} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&\quad + 2\int_{-z}^{+z} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&= 2\int_{+z}^{+\infty} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z+x-\varepsilon, z+x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z+x+\varepsilon)-\varphi_{\mathrm{A}}(z+x-\varepsilon)\right)\mathrm{d}x \\
&\quad - 2\int_{+z}^{+\infty} \varphi_{\mathrm{B}}(2z+x)\Pr\left[\mathrm{A} \in (z+x-\varepsilon, z+x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z+x+\varepsilon)-\varphi_{\mathrm{A}}(z+x-\varepsilon)\right)\mathrm{d}x \qquad \text{(B.4)} \\
&\quad - 2\int_{-z}^{+z} \varphi_{\mathrm{B}}(x-2z)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \qquad \text{(B.5)} \\
&\quad + 2\int_{-z}^{+z} \varphi_{\mathrm{B}}(x)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x \\
&= 2\int_{+z}^{+\infty} \left(\varphi_{\mathrm{B}}(x)-\varphi_{\mathrm{B}}(2z+x)\right)\Pr\left[\mathrm{A} \in (z+x-\varepsilon, z+x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z+x+\varepsilon)-\varphi_{\mathrm{A}}(z+x-\varepsilon)\right)\mathrm{d}x \qquad \text{(B.6)} \\
&\quad + 2\int_{-z}^{+z} \left(\varphi_{\mathrm{B}}(x)-\varphi_{\mathrm{B}}(x-2z)\right)\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\left(\varphi_{\mathrm{A}}(z-x+\varepsilon)-\varphi_{\mathrm{A}}(z-x-\varepsilon)\right)\mathrm{d}x, \qquad \text{(B.7)}
\end{aligned}$$

where in Equation (B.3) we substituted $x' = -x$ and used the symmetry of the integrand functions, in Equations (B.4) and (B.5) we substituted $x' = x-2z$ and $x' = 2z-x$, respectively, and used again the symmetry. The expression in Equation (B.6) is negative as $\varphi_{\mathrm{B}}(x) > \varphi_{\mathrm{B}}(2z+x)$ and $\varphi_{\mathrm{A}}(z+x+\varepsilon) < \varphi_{\mathrm{A}}(z+x-\varepsilon)$ for $x \geq z$; the expression in Equation (B.7) is negative as $\varphi_{\mathrm{B}}(x) > \varphi_{\mathrm{B}}(x-2z)$ and $\varphi_{\mathrm{A}}(z-x+\varepsilon) < \varphi_{\mathrm{A}}(z-x-\varepsilon)$ for $x \in (-z, z)$.

Third case: $z < 0$**.** This case is similar to the previous one: with the same arguments, we obtain

$$\frac{\mathrm{d}H(z)}{\mathrm{d}z}$$
$$= 2\int_{-\infty}^{+z} \left(\varphi_{\mathrm{B}}(x) - \varphi_{\mathrm{B}}(2z+x)\right) \Pr\left[\mathrm{A} \in (z+x-\varepsilon, z+x+\varepsilon)\right] \left(\varphi_{\mathrm{A}}(z+x+\varepsilon) - \varphi_{\mathrm{A}}(z+x-\varepsilon)\right) \mathrm{d}x \tag{B.8}$$

$$+ 2\int_{+z}^{-z} \left(\varphi_{\mathrm{B}}(x) - \varphi_{\mathrm{B}}(x-2z)\right) \Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right] \left(\varphi_{\mathrm{A}}(z-x+\varepsilon) - \varphi_{\mathrm{A}}(z-x-\varepsilon)\right) \mathrm{d}x. \tag{B.9}$$

The expression in Equation (B.8) is positive as $\varphi_{\mathrm{B}}(x) > \varphi_{\mathrm{B}}(2z+x)$ and $\varphi_{\mathrm{A}}(z+x+\varepsilon) > \varphi_{\mathrm{A}}(z+x-\varepsilon)$ for $x \leq z$; the expression in Equation (B.9) is positive as $\varphi_{\mathrm{B}}(x) > \varphi_{\mathrm{B}}(x-2z)$ and $\varphi_{\mathrm{A}}(z-x+\varepsilon) < \varphi_{\mathrm{A}}(z-x-\varepsilon)$ for $x \in (z, -z)$.

□

Claim B.4. *For all* $x \in \mathbb{R}$*,* $c \in \left(0, \frac{1}{162}\right)$*, and* $\varepsilon \in (0,1)$*, it holds that*

$$\left(\int_{-\varepsilon}^{\varepsilon} e^{-c(x+s)^2}\,\mathrm{d}s\right)^2 \leq \left(\int_{-\varepsilon}^{\varepsilon} \frac{e^{-c(x+\varepsilon)^2} + e^{-c(x-\varepsilon)^2}}{2} e^{c\varepsilon^2}\,\mathrm{d}s\right)^2.$$

Proof.

Let

$$f_x(s) = e^{-c(x+s)^2}.$$

Since

$$\int_{-\varepsilon}^{\varepsilon} \frac{e^{-c(x+\varepsilon)^2} + e^{-c(x-\varepsilon)^2}}{2} e^{c\varepsilon^2}\,\mathrm{d}s = \int_{-\varepsilon}^{\varepsilon} ms + \frac{e^{-c(x+\varepsilon)^2} + e^{-c(x-\varepsilon)^2}}{2} e^{c\varepsilon^2}\,\mathrm{d}s$$

for any $m \in \mathbb{R}$, we choose it to be the angular coefficient of the line passing through $f_x(-\varepsilon)$ and $f_x(\varepsilon)$, and prove the stronger result

$$e^{-c(x+s)^2} \leq \frac{e^{-c(x+\varepsilon)^2} - e^{-c(x-\varepsilon)^2}}{2\varepsilon} s + \frac{e^{-c(x+\varepsilon)^2} + e^{-c(x-\varepsilon)^2}}{2} e^{c\varepsilon^2} \tag{B.10}$$

for all $s \in (-\varepsilon, \varepsilon)$. In fact, the right hand side of Equation (B.10) is the equation for the line passing by the extrema of f_x in $(-\varepsilon, \varepsilon)$ lifted by a factor of $e^{c\varepsilon^2}$. Therefore, the results holds trivially if f_x is convex in the entire range $(-\varepsilon, \varepsilon)$, which is true when $|x| > 1 + \frac{1}{\sqrt{2c}}$. Moreover, the factor $e^{c\varepsilon^2}$ ensures the result for $x = 0$, so, we follow with the analysis of the case $x \in \left(0, 1 + \frac{1}{\sqrt{2c}}\right]$ and the remaining case $x \in \left[-1 - \frac{1}{\sqrt{2c}}, 0\right)$ follows by symmetry.

Dividing both side of Equation (B.10) by $e^{-c(x+s)^2}$, we obtain

$$1 \leq e^{2csx+cs^2} \left[\frac{e^{-c\varepsilon^2} s}{\varepsilon} \cdot \frac{e^{-2c\varepsilon x} - e^{2c\varepsilon x}}{2} + \frac{e^{-2c\varepsilon x} + e^{2c\varepsilon x}}{2}\right] \tag{B.11}$$
$$= e^{2csx+cs^2} \left[-\frac{e^{-c\varepsilon^2} s}{\varepsilon} \sinh(2c\varepsilon x) + \cosh(2c\varepsilon x)\right].$$

Let $g(x)$ be the right hand side of this inequality. Then

$$g'(x) = 2csg(x) + 2c\varepsilon e^{2csx+cs^2}\left[-\frac{e^{-c\varepsilon^2}s}{\varepsilon}\cosh(2c\varepsilon x) + \sinh(2c\varepsilon x)\right]$$
$$= 2ce^{2csx+cs^2}\left[\cosh(2c\varepsilon x)\left(s - se^{-c\varepsilon^2}\right) + \sinh(2c\varepsilon x)\left(\varepsilon - \frac{s^2}{\varepsilon}e^{-c\varepsilon^2}\right)\right].$$

If $s \in [0, \varepsilon)$, then $s \geq se^{-c\varepsilon^2}$ and $\varepsilon \geq \frac{\varepsilon^2}{\varepsilon}e^{-c\varepsilon^2} \geq \frac{s^2}{\varepsilon}e^{-c\varepsilon^2}$, hence $g'(x) \geq 0$. Since $g(0) \geq 1$, this ensures Equation (B.11).

The sub-case $s \in (-\varepsilon, 0)$ offers much more resistance. To analyse it we exploit that $x \in \left(0, 1 + \frac{1}{\sqrt{2c}}\right)$ implies that $cx \leq \sqrt{2c}$ for $c < \frac{1}{2}$ and make extensive use of Taylor's theorem to approximate the exponential functions.

We start by rewriting Equation (B.11) as

$$\varepsilon e^{-2csx-cs^2} \leq e^{2c\varepsilon x}\left(\frac{\varepsilon}{2} - \frac{s}{2}e^{-c\varepsilon^2}\right) + e^{-2c\varepsilon x}\left(\frac{\varepsilon}{2} + \frac{s}{2}e^{-c\varepsilon^2}\right). \tag{B.12}$$

By Taylor's theorem, there exist $\lambda_1, \lambda_2 \in [0, 2c\varepsilon x] \subseteq [0, 2\sqrt{2c}\varepsilon]$, $\lambda_3 \in [0, -2csx] \subseteq [0, 2\sqrt{2c}\varepsilon]$, $\lambda_4 \in [0, cs^2]$, $\lambda_5 \in [0, c\varepsilon^2]$ such that

$$e^{+2c\varepsilon x} = 1 + 2c\varepsilon x + 2c^2\varepsilon^2x^2 + \frac{4}{3}c^3\varepsilon^3x^3e^{\lambda_1},$$
$$e^{-2c\varepsilon x} = 1 - 2c\varepsilon x + 2c^2\varepsilon^2x^2 - \frac{4}{3}c^3\varepsilon^3x^3e^{\lambda_2},$$
$$e^{-2csx} = 1 - 2csx + 2c^2s^2x^2 - \frac{4}{3}c^3s^3x^3e^{\lambda_3},$$
$$e^{-cs^2} = 1 - cs^2 + \frac{c^2s^4}{2}e^{-\lambda_4}, \tag{B.13}$$
$$e^{-c\varepsilon^2} = 1 - c\varepsilon^2 + \frac{c^2\varepsilon^4}{2}e^{-\lambda_5}, \tag{B.14}$$

where we used second order approximations for the first three terms and first order approximations for the last two. Plugging those in Equation (B.12) we obtain

$$\varepsilon e^{-cs^2}\left(1 - 2csx + 2c^2s^2x^2 - \frac{4}{3}c^3s^3x^3e^{\lambda_3}\right)$$
$$\leq \left(1 + 2c\varepsilon x + 2c^2\varepsilon^2x^2 + \frac{4}{3}c^3\varepsilon^3x^3e^{\lambda_1}\right)\left(\frac{\varepsilon}{2} - \frac{s}{2}e^{-c\varepsilon^2}\right)$$
$$+ \left(1 - 2c\varepsilon x + 2c^2\varepsilon^2x^2 - \frac{4}{3}c^3\varepsilon^3x^3e^{\lambda_2}\right)\left(\frac{\varepsilon}{2} + \frac{s}{2}e^{-c\varepsilon^2}\right).$$

The latter becomes

$$\varepsilon\left(1 - e^{-cs^2}\right) + 2cs\varepsilon x\left(e^{-cs^2} - e^{-c\varepsilon^2}\right) + 2c^2\varepsilon x^2\left(\varepsilon^2 - s^2e^{-cs^2}\right)$$
$$+ \frac{4}{3}c^3\varepsilon x^3\left(\varepsilon^2e^{\lambda_1}\left(\frac{\varepsilon}{2} - \frac{s}{2}e^{-c\varepsilon^2}\right) - \varepsilon^2e^{-\lambda_2}\left(\frac{\varepsilon}{2} + \frac{s}{2}e^{-c\varepsilon^2}\right) + s^3e^{\lambda_3 - cs^2}\right) \geq 0$$

Now, notice that

$$\varepsilon^2 e^{\lambda_1}\left(\frac{\varepsilon}{2}-\frac{s}{2}e^{-c\varepsilon^2}\right)-\varepsilon^2 e^{-\lambda_2}\left(\frac{\varepsilon}{2}+\frac{s}{2}e^{-c\varepsilon^2}\right)\geq 0,$$

as $\frac{\varepsilon}{2}-\frac{s}{2}e^{-c\varepsilon^2}\geq\frac{\varepsilon}{2}+\frac{s}{2}e^{-c\varepsilon^2}$ since $-\varepsilon\leq s<0$, and $\varepsilon^2e^{\lambda_1}\geq\varepsilon^2\geq\varepsilon^2e^{-\lambda_2}$. Furthermore, observe that $s^3e^{\lambda_3-cs^2}\geq 2s^3$ as $s<0$ and $\lambda_3\leq 2\sqrt{2c}\varepsilon\leq\frac{1}{2}$ if $c\leq\frac{1}{32}$. Thus, the inequality is true if

$$\varepsilon\left(1-e^{-cs^2}\right)+2cs\varepsilon x\left(e^{-cs^2}-e^{-c\varepsilon^2}\right)+2c^2\varepsilon x^2\left(\varepsilon^2-s^2e^{-cs^2}\right)+\frac{8}{3}c^3s^3\varepsilon x^3\geq 0.$$

Applying Equations (B.13) and (B.14), the latter inequality yields that

$$\begin{aligned}
&\varepsilon\left(cs^2-\frac{c^2s^4}{2}e^{-\lambda_4}\right)+2cs\varepsilon x\left(c\varepsilon^2-cs^2-\frac{c^2\varepsilon^4}{2}e^{-\lambda_5}+\frac{c^2s^4}{2}e^{-\lambda_4}\right)\\
&\quad+2c^2\varepsilon x^2\left(\varepsilon^2-s^2+cs^4-\frac{c^2s^6}{2}e^{-\lambda_4}\right)+\frac{8}{3}c^3s^3\varepsilon x^3\\
&=\varepsilon cs^2-\frac{c^2s^4\varepsilon}{2}e^{-\lambda_4}-c^3s\varepsilon^5xe^{-\lambda_5}+c^3s^5\varepsilon xe^{-\lambda_4}+\left(2c^3s^4\varepsilon x^2-c^4s^6\varepsilon x^2e^{-\lambda_4}\right)+\frac{8}{3}c^3s^3\varepsilon x^3\\
&\quad+2c^2\varepsilon x\left(\varepsilon^2-s^2\right)(x+s).
\end{aligned}$$

Now observe that

$$\left(2c^3s^4\varepsilon x^2-c^4s^6\varepsilon x^2e^{-\lambda_4}\right)\geq 0$$

as $c<1, s\leq\varepsilon\leq 1, e^{-\lambda_4}<1$; $-c^3s\varepsilon^5xe^{-\lambda_5}>0$ as $s<0$;

$$\begin{aligned}
&\varepsilon cs^2-\frac{c^2s^4\varepsilon}{2}e^{-\lambda_4}+c^3s^5\varepsilon xe^{-\lambda_4}+\frac{8}{3}c^3s^3\varepsilon x^3\\
&\geq cs^2\varepsilon-\frac{c^2s^2\varepsilon^3}{2}-c^2\sqrt{2c}s^2\varepsilon^4-\frac{8}{3}c^3s^2\varepsilon^2x^3\\
&>cs^2\varepsilon-\frac{c^2s^2\varepsilon^3}{2}-c^2\sqrt{2c}s^2\varepsilon^4-6c\sqrt{2c}s^2\varepsilon^2\\
&=cs^2\varepsilon\left(1-\frac{c\varepsilon^2}{2}-c^2\sqrt{2c}\varepsilon^3-6\sqrt{2c}\varepsilon\right) &\text{(B.15)}\\
&\geq cs^2\varepsilon\left(1-\frac{c}{2}-c^2\sqrt{2c}-6\sqrt{2c}\right) &\text{(B.16)}\\
&\geq\frac{cs^2\varepsilon}{5}, &\text{(B.17)}
\end{aligned}$$

where in Equation (B.15) we used that $cx\leq\sqrt{2c}$, in Equation (B.16) that $\varepsilon\leq 1$, and in Equation (B.17) we used i) $c^2\sqrt{2c}\leq\frac{c}{2}$ when $c\leq\frac{1}{\sqrt[3]{2}}$, ii) $c<\sqrt{2c}$ since $c<1$ and iii) $1-7\sqrt{2c}\geq\frac{1}{5}$, whenever $c\leq\frac{1}{162}$.
Going back to the inequality, we now have

$$\frac{cs^2\varepsilon}{5}+2c^2\varepsilon x\left(\varepsilon^2-s^2\right)(x+s).$$

If $x \geq |s|$ the latter is positive, otherwise it becomes

$$\begin{aligned}
&\frac{cs^2\varepsilon}{5} + 2c^2\varepsilon x\left(\varepsilon^2 - s^2\right)(x+s) \\
&\geq \frac{cs^2\varepsilon}{5} + 2c^2\varepsilon x^2\left(\varepsilon^2 - s^2\right) - 2c^2\varepsilon s^2\left(\varepsilon^2 - s^2\right) \\
&\geq \frac{cs^2\varepsilon}{5} - 2c^2\varepsilon s^2 + 2c^2\varepsilon x^2\left(\varepsilon^2 - s^2\right) \\
&\geq cs^2\varepsilon\left(\frac{1}{5} - 2c\right)
\end{aligned}$$

which is positive for $c < \frac{1}{10}$.

To sum-up, the thesis always holds for $c \leq \frac{1}{162}$.

□

Claim B.5. *For all $x \in \mathbb{R}$, $c \in \left(0, \frac{1}{10}\right)$, and $\varepsilon \in (0,1)$, it holds that*

$$\left(\int_{x-\varepsilon}^{x+\varepsilon} \exp\left(-cy^2\right)\,\mathrm{d}y\right)^2 \geq \int_{x-\varepsilon}^{x+\varepsilon} \exp\left(-c(x-\varepsilon)^2\right)\,\mathrm{d}y \cdot \int_{x-\varepsilon}^{x+\varepsilon} \exp\left(-c(x+\varepsilon)^2\right)\,\mathrm{d}y. \tag{B.18}$$

Proof.

We can express Equation (B.18) as

$$\begin{aligned}
&\left[\int_{x-\varepsilon}^{x+\varepsilon} \exp\left(-cy^2\right)\,\mathrm{d}y\right]^2 - \left[\int_{x-\varepsilon}^{x+\varepsilon} \exp\left(-c(x^2+\varepsilon^2)\right)\,\mathrm{d}y\right]^2 \\
&\quad = \left[\int_{x-\varepsilon}^{x+\varepsilon} \exp\left(-cy^2\right) - \exp\left(-c(x^2+\varepsilon^2)\right)\,\mathrm{d}y\right] \cdot \left[\int_{x-\varepsilon}^{x+\varepsilon} \exp\left(-cy^2\right) + \exp\left(-c(x^2+\varepsilon^2)\right)\,\mathrm{d}y\right] \\
&\quad \geq 0,
\end{aligned}$$

which holds if and only if

$$\int_{-\varepsilon}^{+\varepsilon} \exp\left(-c(x+s)^2\right)\,\mathrm{d}s \geq \int_{-\varepsilon}^{+\varepsilon} \exp\left(-c(x^2+\varepsilon^2)\right)\,\mathrm{d}s. \tag{B.19}$$

The result is immediate for $x = 0$, so we assume $x > 0$ and the claim follows by symmetry. Let

$$f_x(s) = \exp(-c(x+s)^2).$$

We provide distinct arguments depending on whether x is small or large.

Case $x \in (0,1)$**.** Since we assume $c < \frac{1}{8}$ and $\varepsilon < 1$, we have for any $x \leq 1$ that f_x is concave in $(-\varepsilon, \varepsilon)$. That is,

$$f_x(s) \geq \frac{f_x(\varepsilon) - f_x(-\varepsilon)}{2\varepsilon}s + \frac{f_x(\varepsilon) + f_x(\varepsilon)}{2},$$

for all $s \in (-\varepsilon, \varepsilon)$. Thus,

$$\begin{aligned}\int_{-\varepsilon}^{\varepsilon} f_x(s)\,\mathrm{d}s &\geq \int_{-\varepsilon}^{\varepsilon} \frac{f_x(\varepsilon) - f_x(-\varepsilon)}{2\varepsilon} s + \frac{f_x(\varepsilon) + f_x(-\varepsilon)}{2}\,\mathrm{d}s \\ &= \int_{-\varepsilon}^{\varepsilon} \frac{f_x(\varepsilon) + f_x(-\varepsilon)}{2}\,\mathrm{d}s \\ &= \int_{-\varepsilon}^{\varepsilon} \exp\left(-c(x^2+\varepsilon^2)\right) \cdot \frac{\exp\left(-2cx\varepsilon\right) + \exp\left(2cx\varepsilon\right)}{2}\,\mathrm{d}s \\ &\geq \int_{-\varepsilon}^{\varepsilon} \exp\left(-c(x^2+\varepsilon^2)\right)\,\mathrm{d}s.\end{aligned}$$

Case $x \geq 1$. The integral on the right hand side of Equation (B.19) has the same value for any affine integrand r_x for which $r_x(0) = \exp\left(-c(x^2+\varepsilon^2)\right)$. Thus, proving that $f_x(s) \geq r_x(s)$, for all $s \in (-\varepsilon, \varepsilon)$, concludes the proof.
In particular, we can choose

$$r_x(s) = f_x'(0) \cdot s + \exp\left(-c(x^2+\varepsilon^2)\right).$$

Since

$$f_x'(s) = -2c(x+s)\exp\left(-c(x+s)^2\right),$$

we aim to show that

$$\exp\left(-c(x+s)^2\right) \geq -2csx\exp\left(-cx^2\right) + \exp\left(-c(x^2+\varepsilon^2)\right)$$

for $s \in (-\varepsilon, \varepsilon)$. Dividing by $\exp(-c(x^2+s^2))$ and rearranging, we obtain

$$\exp(-2csx) + 2csx\exp\left(cs^2\right) - \exp\left(-c\left(\varepsilon^2 - s^2\right)\right) \geq 0. \tag{B.20}$$

Now, if $s \geq 0$, we have that

$$\begin{aligned}&\exp(-2csx) + 2csx\exp\left(cs^2\right) - \exp\left(-c\left(\varepsilon^2 - s^2\right)\right) \\ &\geq 1 - 2csx + 2csx(1+cs^2) - \exp\left(-c\varepsilon^2\right) \qquad \text{(B.21)} \\ &= 1 + 2c^2s^3x - \exp\left(-c\varepsilon^2\right) \\ &\geq 2c^2s^3x \\ &\geq 0,\end{aligned}$$

where in Equation (B.21) we used that $e^y \geq 1 + y$.
Now consider the sub-case $s < 0$. By Taylor's theorem,

$$\exp(y) = 1 + y + \frac{y^2}{2} + \frac{\exp(\xi_1)\cdot y^3}{6}$$

and

$$\exp(y) = 1 + y + \frac{\exp(\xi_2)\cdot y^2}{2},$$

for some $\xi_1, \xi_2 \in [0, y]$. Letting $\ell = -s \in (0, 1)$, we have

$$\exp(2c\ell x) \geq 1 + 2c\ell x + \frac{(2c\ell x)^2}{2} + \frac{(2c\ell x)^3}{6}$$

and

$$\begin{aligned}\exp(c\ell^2) &\leq 1 + c\ell^2 + \frac{\exp(c\ell^2)(c\ell^2)^2}{2} \\ &\leq 1 + c\ell^2 + \frac{\left(1 + 3(c\ell^2)\right)(c\ell^2)^2}{2}.\end{aligned}$$

since $e^y \leq 1 + 3y$ for $0 \leq y \leq 1$. Finally, applying this to Equation (B.20), we have

$$\begin{aligned}&\exp(-2csx) + 2csx \exp\left(cs^2\right) - \exp\left(-c\left(\varepsilon^2 - s^2\right)\right) \\ &\quad \geq \exp(2c\ell x) - 2c\ell x \exp\left(c\ell^2\right) - 1 \\ &\quad \geq 1 + 2c\ell x + \frac{(2c\ell x)^2}{2} + \frac{(2c\ell x)^3}{6} - 2c\ell x\left(1 + c\ell^2 + \frac{c^2\ell^4(1 + 3c\ell^2)}{2}\right) - 1 \\ &\quad = 2c^2\ell^2x^2 + \frac{4}{3}c^3\ell^3x^3 - 2c\ell x\left(c\ell^2 + \frac{c^2\ell^4(1 + 3c\ell^2)}{2}\right) \\ &\quad = 2c^2\ell^2x(x - \ell) + c^3\ell^3x\left(\frac{4}{3}x^2 - \ell^2(1 + 3c\ell^2)\right).\end{aligned}$$

The latter is non negative for $x \geq 1$ and $c \leq \frac{1}{9}$, since $\ell = -s \leq \varepsilon < 1$, so that $\frac{4}{3}x^2 - \ell^2(1 + 3c\ell^2) \geq \frac{4}{3} - 1 - \frac{1}{3} = 0$.

□

We provide hereafter the omitted proofs.

Proof of Lemma 6.5.1

By the distribution of $\mathbf{X}$,

$$\Pr\left[\mathbf{X} \in \mathcal{B}_\infty^d(\boldsymbol{z}, \varepsilon)\right] = \int_{\mathcal{B}_\infty^d(\boldsymbol{z}, \varepsilon)} \frac{1}{\left(2\pi\sigma^2\right)^{\frac{d}{2}}} \cdot \exp\left(-\frac{\|\boldsymbol{x}\|_2^2}{2\sigma^2}\right) \mathrm{d}\boldsymbol{x}.$$

Since $\mathcal{B}_\infty^d(\boldsymbol{z}, \varepsilon) \subseteq \mathcal{B}_\infty^d(\mathbf{0}, 2)$ and for all $\boldsymbol{x} \in \mathbb{R}^d$ it holds that $\|\boldsymbol{x}\|_2 \leq \sqrt{d} \cdot \|\boldsymbol{x}\|_\infty$, and, thus,

$$\exp\left(-\frac{2d}{\sigma^2}\right) \leq \exp\left(-\frac{\|\boldsymbol{x}\|_2^2}{2\sigma^2}\right) \leq 1.$$

The thesis follows by noting that the hypercube $\mathcal{B}_\infty^d(\boldsymbol{z}, \varepsilon)$ has measure $(2\varepsilon)^d$.

Proof of Lemma 6.7.1

Inheriting the setup from the proof of Lemma 6.5.3 and proceeding analogously we obtain that $\sigma_{\mathrm{A}}^2 = \alpha n\left(1-\frac{\alpha}{2}\right)$ and $\sigma_{\mathrm{B}}^2 = \frac{\alpha^2 n}{2}$. We diverge from that argument after Equation (6.2). Preserving equality for a bit longer, we have that

$$\begin{aligned}\left(\Pr\left[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1\right]\right)^{\frac{1}{d}} &= \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \left(\Pr\left[\mathrm{A} \in (z-x-\varepsilon, z-x+\varepsilon)\right]\right)^2 \mathrm{d}x \\ &= \int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \left(\int_{z-x-\varepsilon}^{z-x+\varepsilon} \varphi_{\mathrm{A}}(y)\,\mathrm{d}y\right)^2 \mathrm{d}x.\end{aligned}$$

The hypothesis on n implies that $2\sigma_a^2 \geq 10$, so, by Claim B.5,

$$\begin{aligned}\left(\int_{z-x-\varepsilon}^{z-x+\varepsilon} \varphi_{\mathrm{A}}(y)\,\mathrm{d}y\right)^2 &\geq (2\varepsilon)^2 \cdot \varphi_{\mathrm{A}}(z-x-\varepsilon) \cdot \varphi_{\mathrm{A}}(z-x+\varepsilon) \\ &= \frac{(2\varepsilon)^2}{2\pi\sigma_{\mathrm{A}}^2} \cdot \exp\left(-\frac{(z-x-\varepsilon)^2}{2\sigma_{\mathrm{A}}^2}\right) \cdot \exp\left(-\frac{(z-x+\varepsilon)^2}{2\sigma_{\mathrm{A}}^2}\right) \\ &= e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2} \cdot \frac{1}{\sqrt{2}} \cdot \frac{(2\varepsilon)^2}{\sqrt{2\pi\sigma_{\mathrm{A}}^2}} \cdot \frac{1}{\sqrt{\pi\sigma_{\mathrm{A}}^2}} \cdot \exp\left(-\frac{(z-x)^2}{\sigma_{\mathrm{A}}^2}\right) \\ &= e^{-\varepsilon^2/\sigma_{\mathrm{A}}^2} \cdot \frac{1}{\sqrt{2}} \cdot \frac{(2\varepsilon)^2}{\sqrt{2\pi\sigma_{\mathrm{A}}^2}} \cdot \varphi_{\mathrm{A}/\sqrt{2}}(z-x).\end{aligned}$$

Then, as before, we can reduce the main integral to a convolution. Namely, it holds that

$$\begin{aligned}\int_{\mathbb{R}} \varphi_{\mathrm{B}}(x) \cdot \varphi_{\mathrm{A}/\sqrt{2}}(z-x)\,\mathrm{d}x &= \varphi_{\mathrm{B}+\mathrm{A}/\sqrt{2}}(z) \\ &= \frac{1}{\sqrt{2\pi\sigma_{\mathrm{B}+\mathrm{A}/\sqrt{2}}^2}} \cdot \exp\left(-\frac{z^2}{2\sigma_{\mathrm{B}+\mathrm{A}/\sqrt{2}}^2}\right).\end{aligned}$$

Altogether, we have that

$$\begin{aligned}\left(\Pr\left[\mathrm{Y}_S = 1, \mathrm{Y}_T = 1\right]\right)^{\frac{1}{d}} &\geq \frac{(2\varepsilon)^2}{2\pi} \cdot \frac{1}{\sqrt{2\sigma_{\mathrm{A}}^2\sigma_{\mathrm{B}+\mathrm{A}/\sqrt{2}}^2}} \cdot \exp\left(-\frac{\varepsilon^2}{\sigma_{\mathrm{A}}^2} - \frac{z^2}{2\sigma_{\mathrm{B}+\mathrm{A}/\sqrt{2}}^2}\right) \\ &= \frac{(2\varepsilon)^2}{2\pi\alpha n} \cdot \frac{1}{\sqrt{1-\frac{\alpha^2}{4}}} \cdot \exp\left(-\frac{1}{\alpha n} \cdot \left(\frac{2\varepsilon^2}{2-\alpha} + \frac{2z^2}{2+\alpha}\right)\right).\end{aligned}$$

where the last equality follows from recalling that $\sigma_{\mathrm{B}}^2 = \frac{\alpha^2 n}{2}$ and $\sigma_{\mathrm{A}}^2 = \alpha n\left(1-\frac{\alpha}{2}\right)$, which implies that $\sigma_{\mathrm{B}+\mathrm{A}/\sqrt{2}}^2 = \frac{\alpha^2 n}{2} + \frac{\alpha n}{2}\left(1-\frac{\alpha}{2}\right)$. Finally, the hypotheses $z \in [-1,1]$, $\varepsilon \in (0,1)$, and $\alpha \in \left(0, \frac{1}{2}\right)$ imply that $\frac{2\varepsilon^2}{2-\alpha} + \frac{2z^2}{2+\alpha} < 3$.

Proof of Theorem 6.5.5

Let $n = k \cdot \frac{144d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)$ with $k \in \mathbb{N}$. By Lemma 6.5.4, for any $\boldsymbol{z} \in [-1,1]^d$, the probability than no subset-sum is sufficiently close to $\boldsymbol{z}$ is at most $\left(\frac{2}{3}\right)^k$. Leveraging the fact that it is possible to cover $[-1,1]^d$ by $\frac{1}{\varepsilon^d}$ hypercubes of radius ε, we can ensure that the probability of failing to 2ε-approximate any $\boldsymbol{z}$ is, by the union bound, at most

$$\begin{aligned}\frac{1}{\varepsilon^d}\cdot\left(\frac{2}{3}\right)^k &= 2^{-k\log\frac{3}{2}+d\log\frac{1}{\varepsilon}} \\ &= \exp\left[-\ln 2\cdot\frac{n-\frac{144d^2}{\alpha^2\log\frac{3}{2}}\log\frac{1}{\varepsilon}\left(\log\frac{1}{\varepsilon}+\log d+\log\frac{1}{\alpha}\right)}{\frac{144d}{\alpha^2\log\frac{3}{2}}\left(\log\frac{1}{\varepsilon}+\log d+\log\frac{1}{\alpha}\right)}\right]\end{aligned}$$

Thus, we can conclude the result for

$$n \geq \frac{144}{\log\frac{3}{2}}\cdot\frac{d^2}{\alpha^2}\log\frac{1}{\varepsilon}\cdot\left(\log\frac{1}{\varepsilon}+\log d+\log\frac{1}{\alpha}\right).$$

Generalisation of our result.

If the target value $\boldsymbol{z}$ lies in the hypercube $[-\lambda\sqrt{n},\lambda\sqrt{n}]^d$, for some $\lambda > \frac{1}{\sqrt{n}}$, we have slightly different bounds for the expectation and for the variance of Y. In particular, corollary 6.5.2 would give

$$e^{-\frac{2\lambda^2 d}{\alpha}}\frac{(2\varepsilon)^d|\mathcal{C}|}{(2\pi\alpha n)^{\frac{d}{2}}} \leq \mathbb{E}[\mathrm{Y}] \leq \frac{(2\varepsilon)^d|\mathcal{C}|}{(2\pi\alpha n)^{\frac{d}{2}}}. \tag{B.22}$$

On the other hand, as the proof of lemma 6.5.3 never uses that $\boldsymbol{z} \in [-1,1]^d$ but only exploits the bound on the expectation, it would yield

$$\mathrm{Var}[\mathrm{Y}] \leq \frac{(2\varepsilon)^{2d}|\mathcal{C}|^2}{(2\pi\alpha n)^d}\left[(1-4\alpha^2)^{-\frac{d}{2}-e^{-\frac{4\lambda^2 d}{\alpha}}}\right] + \frac{(2\varepsilon)^d|\mathcal{C}|}{(2\pi\alpha n)^{\frac{d}{2}}}. \tag{B.23}$$

We focus on the case $\lambda = \frac{1}{2}\sqrt{\frac{\alpha}{17d}}$ when $n > \frac{68d}{\alpha}$ (which implies $\lambda\sqrt{n} > 1$). Thus, we have a new estimation for the probability to hit a single value.

Lemma B.6. *Given $d, n \in \mathbb{N}$, $\varepsilon \in (0,1)$, and $\alpha \in (0, \frac{1}{6}]$, let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ i.d.d. following $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, $\boldsymbol{z} \in [-\lambda\sqrt{n},\lambda\sqrt{n}]^d$, with $\lambda = \frac{1}{2}\sqrt{\frac{\alpha}{17d}}$, and $\mathcal{C} \subseteq \binom{[n]}{\alpha n}$. If any two subsets in $\mathcal{C}$ intersect in at most $2\alpha^2 n$ elements, $\alpha \leq \frac{1}{6\sqrt{d}}$, and*

$$n \geq \frac{144d}{\alpha^2}\left(\log\frac{1}{\varepsilon}+\log d+\log\frac{1}{\alpha}\right),$$

then

$$\Pr[\mathrm{Y} \geq 1] \geq \frac{1}{3}.$$

Proof.

By Chebyshev's inequality, it holds that

$$\begin{aligned}\Pr\left[\mathrm{Y} \geq 1\right] &\geq \Pr\left[\left|\mathrm{Y} - \mathbb{E}\left[\mathrm{Y}\right]\right| < \frac{\mathbb{E}\left[\mathrm{Y}\right]}{2}\right] \\ &\geq 1 - \frac{4 \cdot \mathrm{Var}\left[\mathrm{Y}\right]}{\mathbb{E}\left[\mathrm{Y}\right]^2}.\end{aligned}$$

Notice that $\frac{4\lambda^2 d}{\alpha} = \frac{1}{17}$. Hence, using eqs. (B.22) and (B.23), we get that

$$\begin{aligned}\frac{4 \cdot \mathrm{Var}\left[\mathrm{Y}\right]}{\mathbb{E}\left[\mathrm{Y}\right]^2} &\leq 4 \cdot \frac{e^{\frac{1}{17}} \cdot (2\pi\alpha n)^d}{(2\varepsilon)^{2d}\left|\mathcal{C}\right|^2} \cdot \left[\frac{(2\varepsilon)^{2d}\left|\mathcal{C}\right|^2}{(2\pi\alpha n)^d} \cdot \left[(1-4\alpha^2)^{-\frac{d}{2}} - e^{-\frac{1}{17}}\right] + \frac{(2\varepsilon)^d\left|\mathcal{C}\right|}{(2\pi\alpha n)^{\frac{d}{2}}}\right] \\ &= 4 \cdot \left(\frac{e^{\frac{1}{17}}}{(1-4\alpha^2)^{\frac{d}{2}}} - 1\right) + \frac{4e^{\frac{1}{17}} \cdot (2\pi\alpha n)^{\frac{d}{2}}}{(2\varepsilon)^d\left|\mathcal{C}\right|}.\end{aligned}$$

Note that Claim B.1 holds exactly as it is for the ratio

$$\frac{e^{\frac{1}{17}}}{(1-4\alpha^2)^{\frac{d}{2}}}$$

obtaining the same bound for $n \geq \frac{68d}{\alpha}$ and $\alpha \leq \frac{1}{6\sqrt{d}}$, which yields

$$4 \cdot \left(\frac{e^{\frac{1}{17}}}{(1-4\alpha^2)^{\frac{d}{2}}} - 1\right) \leq \frac{1}{2}.$$

Furthermore, also Claim B.2 is true replacing $e^{\frac{4d}{\alpha n}}$ by $e^{\frac{1}{17}}$. Thus, as $n \geq \frac{144d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)$ and $\alpha \leq \frac{1}{6}$, Claim B.2 implies that

$$\frac{4e^{\frac{1}{17}} \cdot (2\pi\alpha n)^{\frac{d}{2}}}{(2\varepsilon)^d\left|\mathcal{C}\right|} \leq \varepsilon.$$

□

We remark that we cannot let λ be asymptotically greater than $\sqrt{\frac{\alpha}{d}}$ otherwise our method fails. Indeed, by remark 6.7.1, the term $\frac{4\mathrm{Var}[\mathrm{Y}]}{\mathbb{E}[\mathrm{Y}]^2}$ is at least

$$4 \cdot \left(\frac{e^{\frac{4\lambda^2 d}{\alpha} - \frac{3\lambda^2 d}{\alpha}}}{\left(1 - \frac{\alpha^2}{4}\right)^{\frac{d}{2}}} - 1\right).$$

The latter is greater than or equal to 1 if $\lambda \geq \sqrt{\frac{\alpha}{d}}$ since $e^{\frac{\lambda^2 d}{\alpha}} \geq 1 + \frac{\lambda^2 d}{\alpha}$.

We are ready to state our first generalised version of theorem 6.5.5.

Theorem B.7. *For given d and $\varepsilon \in (0,1)$, let $\mathbf{X}_1, \dots, \mathbf{X}_n$ be n independent standard normal d-dimensional random vectors and let $\alpha \in (0, \frac{1}{6\sqrt{d}}]$. There exist two universal constants $C > \delta > 0$ such that, if*

$$n \geq C\frac{d^2}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)^2,$$

the following holds with probability at least

$$1 - \exp\left[-\ln 2 \cdot \left(\frac{n}{\delta\frac{d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)} - d\log\frac{1}{\varepsilon}\right)\right]:$$

for all $\boldsymbol{z} \in \left[-\lambda\sqrt{n}, \lambda\sqrt{n}\right]^d$, with $\lambda = \frac{1}{2}\sqrt{\frac{\alpha}{17d}}$, there exists a subset $S_{\boldsymbol{z}} \subseteq [n]$, such that

$$\left\|\boldsymbol{z} - \sum_{i \in S_{\boldsymbol{z}}} \mathbf{X}_i\right\|_\infty \leq 2\varepsilon.$$

Moreover, the property above remains true even if we restrict to subsets of size αn.

Proof.

Let $\frac{n}{\frac{144d}{\alpha^2}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)} = k \geq 1$ with $k \in \mathbb{N}$. By lemma B.6, for any $\boldsymbol{z} \in [-\lambda\sqrt{n}, \lambda\sqrt{n}]^d$, the probability than no subset-sum is sufficiently close to $\boldsymbol{z}$ is at most $\left(\frac{2}{3}\right)^k$. Leveraging the fact that it is possible to cover $[-\lambda\sqrt{n}, \lambda\sqrt{n}]^d$ by $\left(\frac{\lambda\sqrt{n}}{\varepsilon}\right)^d$ hypercubes of radius ε, we can ensure that the probability of failing to 2ε-approximate any $\boldsymbol{z}$ is, by the union bound, at most

$$\begin{aligned}
\left(\frac{\lambda\sqrt{n}}{\varepsilon}\right)^d \cdot \left(\frac{2}{3}\right)^k &= 2^{-k\log\frac{3}{2} + d\left(\log\frac{1}{\varepsilon} + \frac{1}{2}\log n + \log\lambda\right)} \\
&= \exp\left[-\ln 2 \cdot \frac{n - \frac{144d^2}{\alpha^2\log\frac{3}{2}}\left(\log\frac{1}{\varepsilon} + \frac{1}{2}\log n + \log\lambda\right)\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)}{\frac{144d}{\alpha^2\log\frac{3}{2}}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)}\right] \\
&\leq \exp\left[-\ln 2 \cdot \frac{n - \frac{144d^2}{\alpha^2\log\frac{3}{2}}\left(\log\frac{1}{\varepsilon} + \frac{1}{2}\log n\right)\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)}{\frac{144d}{\alpha^2\log\frac{3}{2}}\left(\log\frac{1}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)}\right]
\end{aligned} \tag{B.24}$$

since $\lambda < 1$. Consider $\frac{n}{2} - \frac{144d^2}{2\alpha^2 \log \frac{3}{2}} \log n \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)$. Let $k = k' \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)$, which means that $n = \frac{144k'd}{\alpha^2} \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)^2$. Then

$$
\begin{aligned}
&\frac{n}{2} - \frac{144d^2}{2\alpha^2 \log \frac{3}{2}} \log n \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \\
&\quad = \frac{144d}{2\alpha^2} \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \left[k' \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \right. \\
&\qquad \left. - \frac{d}{\log \frac{3}{2}} \left(\log \frac{144}{\log \frac{3}{2}} + \log k' + \log d + 2 \log \frac{1}{\alpha} + 2 \log \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \right) \right] \\
&\quad \geq \frac{144d}{2\alpha^2} \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \left[k' \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \right. \\
&\qquad \left. - 2d \left(8 + \log k' + \log d + 2 \log \frac{1}{\alpha} + 2 \log \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \right) \right]
\end{aligned}
$$

If $k' = 17d$, we have that

$$
\begin{aligned}
&k' \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) - 2d \left(8 + \log k' + \log d + 2 \log \frac{1}{\alpha} + 2 \log \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \right) \\
&\quad \geq 4d \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} - \log \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right) \right) + 13d \log d + 13d \log \frac{1}{\alpha} \\
&\qquad - 16d - 2d \log c - 3d \log d - 4d \log \frac{1}{\alpha} \\
&\quad = 10d \log d + 9d \log \frac{1}{\alpha} - 16d - 2d \log 17 \geq 0,
\end{aligned}
$$

as $\alpha \leq \frac{1}{6}$. Thus, for $n \geq \frac{17 \cdot 144d^2}{\alpha^2} \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)^2$, we have that the expression in eq. (B.24) is at most

$$
\exp \left[- \ln 2 \cdot \frac{n - \frac{288d^2}{\alpha^2 \log \frac{3}{2}} \log \frac{1}{\varepsilon} \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)}{\frac{288d}{\alpha^2 \log \frac{3}{2}} \left(\log \frac{1}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)} \right].
$$

We have the thesis by setting $\delta = \frac{288}{\log \frac{3}{2}}$ and $C = 17 \cdot 144$.

□

Our analysis, that relies on fixed subset sizes, easily extends theorem B.7 for non-centred and non-unitary normal random vectors.

Corollary B.8. *Let $\sigma > 0$ and $\varepsilon \in (0, \sigma)$. Given $d, n \in \mathbb{N}$ let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be independent normal d-dimensional random vectors with $\mathbf{X}_i \sim \mathcal{N}(\boldsymbol{v}, \sigma^2 \cdot \boldsymbol{I}_d)$, for any vector $\boldsymbol{v} \in \mathbb{R}^d$. Furthermore, let $\alpha \in \left(0, \frac{1}{6\sqrt{d}}\right)$. There exist two universal constants $C > \delta > 0$ such that, if*

$$n \geq C \frac{d^2}{\alpha^2} \left(\log \frac{\sigma}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)^2,$$

then, with probability

$$1 - \exp\left[-\ln 2 \cdot \left(\frac{n}{\delta \frac{d}{\alpha^2} \left(\log \frac{\sigma}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)} - d \log \frac{\sigma}{\varepsilon} \right) \right],$$

for all $\boldsymbol{z} \in [-\sigma\lambda\sqrt{n}, \sigma\lambda\sqrt{n}]^d + \alpha n \boldsymbol{v}$, with $\lambda = \frac{1}{2}\sqrt{\frac{\alpha}{17d}}$, there exists a subset $S_{\boldsymbol{z}} \subseteq [n]$ for which

$$\left\| \boldsymbol{z} - \sum_{i \in S_{\boldsymbol{z}}} \mathbf{X}_i \right\|_\infty \leq 2\varepsilon.$$

Moreover, this remains true even when restricted to subsets of size αn.

Proof.

Simply apply theorem B.7 to the random vectors $\frac{\mathbf{X}_i - \boldsymbol{v}}{\sigma}$ with error $\frac{\varepsilon}{\sigma}$.

□

Following the line of [Lueker, 1998a], we also observe that our results extend to a wider class of probability distributions.

Definition B.1. Consider any two random variables X and Y having the same codomain, and let $\varphi_{\mathrm{X}}(x), \varphi_{\mathrm{Y}}(x)$ be their probability density functions. We say that X *contains* Y with probability p if a constant $p \in (0, 1]$ exists such that $\varphi_{\mathrm{X}}(x) = p \cdot \varphi_{\mathrm{Y}}(x) + (1 - p) f(x)$ for any function $f(x)$.

If X contains Y with probability p, we can describe the behaviour of X as follows: with probability p, draw Y; with probability $1 - p$, draw something else. An adapted version of our result holds for random variables containing Gaussian distributions.

Corollary B.9. *Let $\sigma > 0$, $\varepsilon \in (0, \sigma)$, and let $p \in (0, 1]$ be a constant. Given $d, n \in \mathbb{N}$ let $\mathbf{Y}_1, \ldots, \mathbf{Y}_n$ be independent d-dimensional random vectors containing d-dimensional normal random vectors $\mathbf{X} \sim \mathcal{N}(\boldsymbol{v}, \sigma^2 \cdot \boldsymbol{I}_d)$ with probability p, where $\boldsymbol{v}$ is any vector in $\mathbb{R}^d$. Furthermore, let $\alpha \in \left(0, \frac{1}{6\sqrt{d}}\right)$. There exist two universal constants $C > \delta > 0$ such that, if*

$$n \geq 2C \frac{d^2}{p\alpha^2} \left(\log \frac{\sigma}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)^2,$$

then, with probability

$$1 - 2\exp\left[-\ln 2 \cdot \left(\frac{pn}{2\delta \frac{d}{\alpha^2} \left(\log \frac{\sigma}{\varepsilon} + \log d + \log \frac{1}{\alpha} \right)} - d \log \frac{\sigma}{\varepsilon} \right) \right],$$

for all $\boldsymbol{z} \in \left[-\sigma\lambda\sqrt{\frac{pn}{2}}, \sigma\lambda\sqrt{\frac{pn}{2}}\right]^d + \frac{\alpha pn}{2}\boldsymbol{v}$*, with* $\lambda = \frac{1}{2}\sqrt{\frac{\alpha}{17d}}$*, there exists a subset* $S_{\boldsymbol{z}} \subseteq [n]$ *for which*

$$\left\|\boldsymbol{z} - \sum_{i \in S_{\boldsymbol{z}}} \mathbf{X}_i\right\|_\infty \leq 2\varepsilon.$$

Moreover, this remains true even when restricted to subsets of size $\frac{\alpha pn}{2}$.

Proof.

With a simple application of the Chernoff bound, we have that at least $\frac{pn}{2}$ random vectors are normal random vectors with probability $1 - e^{-\frac{pn}{8}}$. Conditional on this event, we can apply corollary B.8 to the $\frac{pn}{2}$ normal random vectors. Since $\Pr[A, B] \geq \Pr[A \mid B] \Pr[B]$ for any two events A, B, and $2\delta\frac{d}{\alpha^2}\left(\log\frac{\sigma}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right) \geq 8$, the thesis holds with probability at least

$$1 - \exp\left[-\ln 2 \cdot \left(\frac{pn}{2\delta\frac{d}{\alpha^2}\left(\log\frac{\sigma}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)} - d\log\frac{\sigma}{\varepsilon}\right)\right] - \exp\left[-\frac{pn}{8}\right]$$

$$\geq 1 - 2\exp\left[-\ln 2 \cdot \left(\frac{pn}{2\delta\frac{d}{\alpha^2}\left(\log\frac{\sigma}{\varepsilon} + \log d + \log\frac{1}{\alpha}\right)} - d\log\frac{\sigma}{\varepsilon}\right)\right].$$

□

C Chapter 7

We describe hereafter in detail the pseudocodes described in Sec. 7.3.4.

Algorithm 7 ORDERED-VECTOR

Input: VNFC set $\mathcal{C}$, Ordering rule $\mathcal{R}$
Output: A set of compute nodes $\mathcal{S}$ and a mapping of VNFC nodes to compute nodes $m_{\mathcal{R}} : \mathcal{C} \to \mathcal{S}$

```
Set L_B according to (7.9)                          ▷ Lower bound
Add L_B compute nodes to S
C_R ← sort C decreasingly using R
for each VNFC c in C_R do
    Mapped[c] ← False
    for each compute node s in S do
        if s can host c then                        ▷ Assignment constraints
            Assign c to s
            Mapped[c] ← True
            Break
    if not Mapped[c] then
        S ← S ∪ {t}                                 ▷ Activate a new compute node
        Assign c to t
return S
```

Algorithm 8 LEAST-LOAD

Input: VNFC set $\mathcal{C}$, Ordering rule $\mathcal{R}$
Output: A set of compute nodes $\mathcal{S}$ and a mapping of VNFC nodes to compute nodes $m_{\mathcal{R}} : \mathcal{C} \rightarrow \mathcal{S}$

Set L_B according to (7.9) ▷ Lower bound
Add L_B compute nodes to $\mathcal{S}$
$\mathcal{C}_{\mathcal{R}} \leftarrow$ sort $\mathcal{C}$ decreasingly using $\mathcal{R}$
for each VNFC c in $\mathcal{C}_{\mathcal{R}}$ **do**
    Mapped$[c] \leftarrow$ False
    for each compute node s in $\mathcal{S}$ **do**
        for each zone $z(s)$ **do**
            $\beta_z^{cpu} \leftarrow$ Free available CPU in z
            $\beta_z^{ram} \leftarrow$ Free available RAM in z
        $A(s) \leftarrow (\max_{z(s)} \beta_z^{cpu}, \max_{z(s)} \beta_z^{ram})$
    $\mathcal{S} \leftarrow$ sort $\mathcal{S}$ decreasingly per (A, $F_{\mathcal{R}}$) ▷ Availability
    for each compute node s in $\mathcal{S}$ **do**
        if s can host c **then**
            Assign c to s
            Mapped$[c] \leftarrow$ True
            Break
    if not Mapped$[c]$ **then**
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{t\}$
        Assign c to t
return $\mathcal{S}$

Procedure 1 SWAP

Input: A placement solution given by a mapping of VNFC nodes to compute nodes $m : \mathcal{C} \rightarrow \mathcal{S}$

Output: A set of compute nodes $\mathcal{S}$ and a new mapping of VNFC nodes to compute nodes $m : \mathcal{C} \rightarrow \mathcal{S}$

1: $(P_{min}, \mathcal{S}) \leftarrow$ partition $\mathcal{S}$ (See Sec. 7.3.4)
2: $C_{reallocate} \leftarrow$ collect the VNFCs assigned within P_{min}
3: **for each** VNFC c_1 in $C_{reallocate}$ **do**
4: **for** i in $\{1, ..., |\mathcal{S}|\}$ **do**
5: Choose at random a compute node s in $\mathcal{S}$
6: **for each** VNFC c_2 assigned to s **do**
7: **if** c_1 can be swapped with c_2 **then**
8: $C_{reallocate} \leftarrow C_{reallocate} \setminus \{c_1\}$
9: Assign c_1 to s
10: $C_{reallocate} \leftarrow C_{reallocate} \cup \{c_2\}$
11: **Break**
12: **for each** VNFC c in $C_{reallocate}$ **do**
13: Mapped$[c] \leftarrow$ False
14: **for each** compute node s in $\mathcal{S}$ **do**
15: **if** s can host c **then**
16: Assign c to s
17: Mapped$[c] \leftarrow$ True
18: **Break**
19: **if not** Mapped$[c]$ **then**
20: $\mathcal{S} \leftarrow \mathcal{S} \cup \{t\}$
21: Assign c to t
22: **return** $\mathcal{S}$

Fondements Réseaux et l'IA

Hicham LESFARI