Approximate Inference: Decomposition Methods with Applications to Networks

by

Kyomin Jung

B.S., Seoul National University, 2003

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2009

Author .....
Department of Mathematics
May 1, 2009

Certified by .....
Devavrat Shah
Jamieson Career Development Assistant Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Michel X. Goemans
Chairman, Applied Mathematics Committee

Accepted by .....
David Jerison
Chairman, Department Committee on Graduate Students

Approximate Inference: Decomposition Methods with Applications to Networks

by

Kyomin Jung

Submitted to the Department of Mathematics
on May 1, 2009, in partial fulfillment of the
requirements for the degree of
DOCTOR OF PHILOSOPHY

Abstract

Markov random field (MRF) model provides an elegant probabilistic framework to formulate inter-dependency between a large number of random variables. In this thesis, we present a new approximation algorithm for computing Maximum a Posteriori (MAP) and the log-partition function for arbitrary positive pair-wise MRF defined on a graph G. Our algorithm is based on decomposition of G into *appropriately* chosen small components; then computing estimates locally in each of these components and then producing a *good* global solution. We show that if either G excludes some finite-sized graph as its minor (e.g. planar graph) and has a constant degree bound, or G is a polynomially growing graph, then our algorithm produce solutions for both questions within arbitrary accuracy. The running time of the algorithm is linear on the number of nodes in G, with constant dependent on the accuracy.

We apply our algorithm for MAP computation to the problem of learning the capacity region of wireless networks. We consider wireless networks of nodes placed in some geographic area in an arbitrary manner under interference constraints. We propose a polynomial time approximate algorithm to determine whether a given vector of end-to-end rates between various source-destination pairs can be supported by the network through a combination of routing and scheduling decisions.

Lastly, we investigate the problem of computing loss probabilities of routes in a stochastic loss network, which is equivalent to computing the partition function of the corresponding MRF for the exact stationary distribution. We show that the very popular Erlang approximation provide relatively poor performance estimates, especially for loss networks in the critically loaded regime. Then we propose a novel algorithm for estimating the stationary loss probabilities, which is shown to always converge, exponentially fast, to the asymptotically exact results.

Thesis Supervisor: Devavrat Shah
Title: Jamieson Career Development Assistant Professor of Electrical Engineering and Computer Science

Acknowledgments

This dissertation would not have been possible without the help and encouragement from a number of people. I start by thanking my advisor, Prof. Devavrat Shah who made it all possible for me to go through all the exciting researches. During the course of my stay in MIT, I have learnt a lot from Devavrat. I have been with him from the beginning, and his passion for research has infected me immensely in my research. I cannot thank him enough for his support. I would like to thank Prof. Michael Sipser and Prof. Jonathan Kelner for being the committee of this dissertation. During my Ph.D. studies, I was fortunate to work with a number of people. I have enjoyed collaborating with Matthew Andrews, Alexander Stolyar, Ramakrishna Gummadi, Ramavarapu Sreenivas, Jinwoo Shin, Yingdong Lu, Mayank Sharma, Mark Squillante, Pushmeet Kohli, Sung-soon Choi, Jeong Han Kim, Byung-Ro Moon, Elena Grigorescu, Arnab Bhattacharyya, Sofya Raskhodnikova, and David Woodruff. I thank them for many enlightening discussions we have had in the last few years. I would also like to thank David Gamarnik, Santosh Vempala, Michael Goemans, Madhu Sudan, Ronitt Rubinfeld, David Sontag, Kevin Matulef, Victor Chen and many others for conversations which have influenced my research. My stay at MIT and Cambridge was made pleasant by numerous friends and colleagues whom I would like to thank: Tauhid Zaman, Srikanth Jagabathula, Shreevatsa Rajagopalan, Lynne Dell, Yoonsuk Hyun, Sooho Oh, Hwanchul Yoo, Donghoon Suk, Ben Smolen, Catherine Bolliet, Sangjoon Kim, Dongwoon Bae, Seohyun Choi, Jiye Lee, Sangmok Han, Heesun Kim and many others. Finally, I thank my family who have supported me in all my endeavors.

to my parents

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A Markov random field (MRF) is an abstraction that utilizes graphical representation to capture inter-dependency between a large number of random variables. MRF provides a convenient and consistent way to model context-dependent entities such as network nodes with interference and correlated features. This is achieved by characterizing mutual influences among such entities. The MRF based models have been utilized successfully in the context of coding (e.g. the low density parity check code [51]), statistical physics (e.g. the Ising model [11]), natural language processing [43] and image processing in computer vision [36, 42, 64]. The key property of this representation lies in the fact that the probability distribution over the random variables factorizes.

The key questions of interest in most of these applications are that of inferring the most likely, or Maximum A-Posteriori (MAP), assignment, and computing the normalizing constant of the probability distribution, or the partition function. For example, in coding, the transmitted message is decoded as the most likely code word from the received message. This problem corresponds to computing MAP assignment in an induced MRF in the context of graphical codes (aka LDPC codes). These two problems are NP-hard in general. However, in practice, the underlying graphical structure of MRF is not *adversarial* and hence can yield to simple structure aware approximations. In this dissertation, we identify new classes of structural properties of MRFs that yield to simple, and efficient algorithms for these problems. These

algorithms and presented in Chapter 2.

Then we present application of our algorithms to networks. In wireless network settings, in essence we have a network of n nodes where nodes are communicating over a common wireless medium using a certain standard communication protocol (e.g. IEEE 802.11 standard). Under any such protocol, transmission between a pair of nodes is successful if and only if none of the *nearby* or *interfering* nodes are transmitting simultaneously. Any such interference model is equivalent to an *independent set* interference model over the graph of interfering communication links. In this model, the well known "maximum weight scheduling" introduced by Tassiulas and Ephremides [61] is equivalent to the computation of *maximum weight independent set* by considering each node's queue size as its weight. In Chapter 3, by considering the maximum weight independent set problem as computation of MAP for the corresponding MRF and applying our approximate MAP algorithm, we propose an algorithm for learning the capacity region of a wireless network.

Finally, in Chapter 4, we investigate the problem of computing loss probabilities of routes in a stochastic loss network, which is equivalent to computing log-partition functions of the corresponding MRF for the exact stationary distribution. For this problem, we propose a novel algorithm for estimating the stationary loss probabilities in stochastic loss networks, and show that our algorithm always converges, exponentially fast, to the asymptotically exact solutions.

1.1 Markov random field

An MRF is a probability distribution defined on an undirected graph $G = (V, E)$ in the following manner. For each $v \in V$, let X_v be a random variable taking values in some finite valued space Σ_v. Without loss of generality, assume that $\Sigma_v = \Sigma$ for all $v \in V$. Let $\mathbf{X} = (X_1, \ldots, X_n)$ be collection of these random variables taking values in Σ^n. For any subset $A \subset V$, we let $\mathbf{X}_A$ denote $\{X_v | v \in A\}$. We call a subset $S \subset V$ a *cut* of G if by its removal from G the graph decomposes into two or more disconnected components. That is, $V \backslash S = A \cup B$ with $A \cap B = \emptyset$, $A, B \neq \emptyset$ and for

any $a \in A, b \in B$, $(a,b) \notin E$.

Definition 1 *The probability distribution* $\mathbf{X}$ *is called a Markov random field, if for any cut* $S \subset V$, $\mathbf{X}_A$ *and* $\mathbf{X}_B$ *are conditionally independent given* $\mathbf{X}_S$, *where* $V \backslash S = A \cup B$.

By the Hammersley-Clifford theorem [22], any positive Markov random field, i.e. $\mathbb{P}[\mathbf{X} = \mathbf{x}] > 0$ for all $\mathbf{x} \in \Sigma^n$, can be defined in terms of a decomposition of the distribution over cliques of the graph. That is, let C_G be the collection of the cliques of G, and Ψ_c be a positive real valued *potential function* defined on a clique $c \in C_G$. Then the MRF $\mathbf{X}$ can be represented by

$$\mathbb{P}[\mathbf{X} = \mathbf{x}] \propto \prod_{c \in C_G} \Psi_c(x_c).$$

In Chapters 2 and Chapter 3, we will restrict our attention to pair-wise MRFs, i.e. MRFs having the above potential functions decomposition with cliques only on the vertices and the edges of G. This does not incur loss of generality for the following reason. A distributional representation that decomposes in terms of distribution over cliques can be represented through a factor graph over discrete variables. Any factor graph over discrete variables can be transformed into a pair-wise Markov random field (see, [66] for example) by introducing auxiliary variables. Now, we present a precise definition of the pair-wise Markov random field.

Definition 2 *For each vertex* $v \in V$ *and edge* $(u,v) \in E$, *let there be a corresponding potential function* $\Psi_v : \Sigma \to \mathbb{R}_+$ *and* $\Psi_{uv} : \Sigma^2 \to \mathbb{R}_+$. *The distribution of* $\mathbf{X}$ *for* $\mathbf{x} \in \Sigma^n$ *which has the following form is called a pair-wise Markov random field.*

$$\mathbb{P}[\mathbf{X} = \mathbf{x}] \propto \prod_{v \in V} \Psi_v(x_v) \cdot \prod_{(u,v) \in E} \Psi_{uv}(x_u, x_v). \tag{1.1}$$

In Chapter 4, we will study a class of non pair-wise MRF with application to a stochastic loss network. Its definition is explained in Section 1.3.3.

1.2 Problems of interest

In this section, we present the two most important operational questions of interest for a pair-wise MRF : computing most likely assignment of unknown variables, and computation of probability of assignment given partial observations.

1.2.1 MAP assignment

The maximum a posteriori (MAP) assignment $\mathbf{x}^*$ is an assignment with maximal probability, i.e.

$$\mathbf{x}^* \in \arg\max_{\mathbf{x}\in\Sigma^n} \mathbb{P}[\mathbf{X} = \mathbf{x}].$$

In many inference problems, the best estimator is the Maximum Likelihood (ML) estimator. In the MRF model, this solution corresponds to a MAP assignment. Computing a MAP assignment is of interest in a wide variety of applications. In the context of error-correcting codes it corresponds to decoding the received noisy codeword, and in image processing, it can be used as the basis for image segmentation techniques. In the statistical physics applications, the MAP assignment corresponds to the ground state, or a state with minimum energy. In discrete optimization applications, including computation of a maximum weight independent set, the MAP assignment corresponds to the optimal solution.

In our pair-wise MRF setup, a MAP assignment $\mathbf{x}^*$ is defined as

$$\mathbf{x}^* \in \arg\max_{\mathbf{x}\in\Sigma^n} \left[\prod_{v\in V} \Psi_v(x_v) \cdot \prod_{(u,v)\in E} \Psi_{uv}(x_u, x_v)\right].$$

1.2.2 Partition function

The normalization constant in the definition (1.1) of distribution is called the *partition function* denoted by Z. Specifically,

$$Z = \sum_{\mathbf{x}\in\Sigma^n} \left[\prod_{v\in V} \Psi_v(x_v) \cdot \prod_{(u,v)\in E} \Psi_{uv}(x_u, x_v)\right].$$

Notice that Z is dependent on the potential function expression, which may not be unique for the given MRF distribution (for example, by adding constants to Ψs).

Clearly, the knowledge of Z is necessary in order to evaluate probability distribution or to compute marginal probabilities, i.e. $\mathbb{P}(X_v = x_v)$ for $v \in V$. Under the polynomial (over n) time computational power, computing Z is equivalent to computing $\mathbb{P}[\mathbf{X} = \mathbf{x}]$ for any $\mathbf{x} \in \Sigma^n$. In applications in statistical physics, logarithm of Z provides free-energy, and in reversible stochastic networks, Z provides loss probability for evaluating quality of service (see Chapter 4).

1.3 Examples of MRF

In this section we provide some examples of the MRF in the statistical physics and the network world. Some of which will be useful later in the thesis.

1.3.1 Ising model

The Ising model, introduced by Ernst Ising [20] is a mathematical model to understand structure of spinglass material in statistical physics [59]. In essence, the model is a binary pairwise MRF defined as follows. Let V be a collection of vertices, each corresponding to a "spin". For $v \in V$, let X_v denote its spin value in $\{-1, 1\}$. Let $E \in V \times V$ be collection of edges capturing pair-wise interaction between spins. Then the total energy of the system for each spin configuration $\mathbf{x} \in \{-1, 1\}^{|V|}$ is given as follows.

For $\mathbf{x} \in \{-1, 1\}^{|V|}$,

$$E(\mathbf{x}) \quad \propto \quad - \sum_{(u,v) \in E} J_{uv} x_u x_v, \tag{1.2}$$

where $J_{uv} \in \mathbb{R}$ are constants. For each pair $(u, v) \in E$, if $J_{uv} > 0$, the interaction is called *ferromagnetic*. If $J_{uv} < 0$, then the interaction is called *antiferromagnetic*.

Notice that the above energy function itself does not form an MRF, but $\exp(E)$

does form an pair-wise MRF defined on $G := (V, E)$. Let the *inverse temperature* be

$$\beta := \frac{1}{k_B T},$$

where k_B is the Boltzmann's constant, and T is the temperature of the system. Then the partition function of the following pair-wise MRF

$$Z = \sum_{\mathbf{x} \in \{-1,1\}^n} E(\beta \mathbf{x}) \tag{1.3}$$

corresponds to the thermodynamic total energy of the system [11], which allows us to calculate all the other thermodynamic properties of the system. For this MRF, the MAP assignment corresponds to the ground state, or the state with minimum energy. As an approximation model, Ising model defined on finite dimensional grid graphs, especially on 2-dimensional or 3-dimensional grid graphs are widely used. Generalized forms of Ising model which also contain potential functions for single spins, are widely used for applications in computer vision including image segmentation problems [36].

1.3.2 Wireless network

The primary purpose of a communication network is to satiate heterogeneous demands of various network users while utilizing network resources efficiently. The key algorithmic tasks in a communication network pertain scheduling (physical layer) and congestion control (network layer). Recent exciting progress has led to the understanding that good algorithmic solution for joint scheduling and congestion control can be obtained via the "maximum weight scheduling" of Tassiulas and Ephremides [61]. In Chapter 3, we will investigate this scheduling algorithm extensively.

Consider a wireless network on n nodes defined by a directed graph $G = (V, E)$ with $|V| = n, |E| = L$. For any $e \in E$, let $\alpha(e), \beta(e)$ denote respectively the origin and destination vertices of the edge e. The edges denote potential wireless links, and only the subsets of the edges so that any pairs of the edges in it do not interfere can

be simultaneously active. We say two edges interfere when they share a vertex. Let

$$\mathcal{S} = \{\mathbf{e} \in \{0,1\}^L : \mathbf{e} \text{ is the adjacency vector for a non-interfering subset of } E\}$$

Note that $\mathcal{S}$ is the collection of the *independent sets of* E by considering interference among $e_1 \in E$ and $e_2 \in E$ as the *edge between them.*

Consider m distinct source destination pairs of the network, $(s_1, d_1), \ldots, (s_m, d_m)$ and an *end to end* rate vector, $\mathbf{r} = (r_1, r_2, \ldots, r_m) \in [0,1]^m$. Let t be an index ranging over integers, to be interpreted as slotted time. Define $q_i^j(t) \in \mathbb{R}_+$ as the *packet mass* at node i destined for node d_j at time t (for $1 \leq i \leq n, 1 \leq j \leq m$). Define a *weight matrix* at time t, $\mathcal{W}(t)$, of dimension $L \times m$ via its $(l,j)^{th}$ element ($1 \leq l \leq L$ and $1 \leq j \leq m$):

$$\mathcal{W}_l^j(t) = q_{\alpha(l)}^j(t) - q_{\beta(l)}^j(t). \tag{1.4}$$

The weight vector of dimension L, $\mathbf{W}(t)$, is then defined with its l^{th} element (corresponding to link $l, 1 \leq l \leq L$) as

$$W_l(t) = \max_j \{\mathcal{W}_l^j(t)\}. \tag{1.5}$$

Let $\mathbf{a}\cdot\mathbf{b}$ denote the standard inner product of $\mathbf{a}$ and $\mathbf{b}$. Then let the maximum weight independent set (MWIS) of links be

$$\mathcal{M}(t) = \max_{\mathbf{e}\in\mathcal{S}} \mathbf{e} \cdot \mathbf{W}(t). \tag{1.6}$$

We will define an MRF so that computing MAP in the MRF is equivalent to computing MWIS of the network. In general, given a graph $G = (V, E)$ and node weights given by $\mathbf{w} = (w_1, \ldots, w_{|V|}) \in \mathbb{R}_+^{|V|}$, a subset $\mathbf{x}$ of V is said to be an *independent set* if no two vertices of $\mathbf{x}$ have common edge. let $\mathcal{I}(G)$ be set of all independent sets of G. A *maximum weight independent set* $\mathbf{x}^*$ is defined by $\mathbf{x}^* = \text{argmax}\left\{\mathbf{w}^T \cdot \mathbf{x} : \mathbf{x} \in \mathcal{I}(G)\right\}$, where we consider $\mathbf{w}$ as an element of $\{0,1\}^{|V|}$.

As explained in [61], the scheduling algorithm choosing the above maximum weight

independent set of links for each time t is throughput optimal. However, it is well known that finding maximum weight independent set in general graph is NP-hard [1] and even hard to approximate within $n^{1-o(1)}$ factor ($B/2^{O(\sqrt{\log B})}$ factor for degree B graph) [60].

Consider a subset of V as a binary string $\mathbf{x} \in \{0,1\}^n$. The following pair-wise MRF model is widely used to design approximate solution of the maximum weight independent set (MWIS) for many class of networks, where the weight W_v of node v is defined as in (1.5).

$$W(\mathbf{x}) \quad \propto \quad \exp\left[\sum_{v\in V} W_v(\tau)\cdot x_v\right] \cdot \prod_{(u,v)\in E} \Psi(x_u, x_v), \tag{1.7}$$

where Ψ is defined so that $\Psi(x_1, x_2) = 0$ if $x_1 = x_2 = 1$, and $\Psi(x_1, x_2) = 1$ otherwise.

For the above MRF, note that $W(\mathbf{x})$ is 0 if $\mathbf{x}$ is not an independent set of G. When $\mathbf{x}$ is an independent set of G, $W(\mathbf{x})$ is proportional to the exponent of total weight of $\mathbf{x}$. Hence, computation of MAP assignment for the above MRF corresponds to computation of the MWIS. Details of this model will be discussed in Chapter 3.

1.3.3 Stochastic loss network

For almost a century, starting with the seminal work of Erlang [8], stochastic loss networks have been widely studied as models of many diverse computer and communication systems in which different types of resources are used to serve various classes of customers involving simultaneous resource possession and non-backlogging workloads. Examples include telephone networks, mobile cellular systems, ATM networks, broadband telecommunication networks, optical wavelength-division multiplexing networks, wireless networks, distributed computing, database systems, data centers, and multi-item inventory systems. see, e.g., [21, 28–31, 44, 45, 53, 54, 68, 69]. Loss networks have been used recently for resource planning in the IT services industry, where a collection of IT service products are offered each requiring a set of resources with certain capabilities [4, 39]. In each case, the stochastic loss network

is used to capture the dynamics and uncertainty of the computer/communication application being modeled.

In Chapter 4, we investigate stochastic loss networks with fixed routing, by considering the Markov random field for its stationary distribution. Consider a network with J links, labeled $1, 2, \ldots, J$. Each link j has C_j units of capacity. There is a set of K distinct (pre-determined) routes, denoted by $\mathcal{R} = \{1, \ldots, K\}$. A call on route r requires A_{jr} units of capacity on link j, $A_{jr} \geq 0$. Calls on route r arrive according to an independent Poisson process of rate ν_r, with $\underline{\nu} = (\nu_1, \ldots, \nu_K)$ denoting the vector of these rates. The dynamics of the network are such that an arriving call on route r is admitted to the network if sufficient capacity is available on all links used by route r; else, the call is dropped (or lost). To simplify the exposition, we will assume that the call service times are i.i.d. exponential random variables with unit mean. It is important to note, however, that our results in this thesis are not limited to these service time assumptions since the quantities of interest remain unchanged in the stationary regime under general service time distributions due to the well-known *insensitivity property* [63] of this class of stationary loss networks.

Let $\underline{n}(t) = (n_1(t), \ldots, n_K(t)) \in \mathbb{N}^K$ be the vector of the number of active calls in the network at time t. By definition, we have that $\underline{n}(t) \in \mathcal{S}(C)$ where

$$\mathcal{S}(C) = \left\{ \underline{n} \in \mathbb{Z}^K | \underline{n} \geq 0, A\underline{n} \leq \underline{C} \right\},$$

and $\underline{C} = (C_1, \ldots, C_J)$ denotes the vector of link capacities. Within this framework, the network is Markov with respect to the state $\underline{n}(t)$. It has been well established that the network is a *reversible* multidimensional Markov process with a product-form stationary distribution [27]. Namely, there is a unique stationary distribution π on the state space $\mathcal{S}(C)$ such that for $\underline{n} \in \mathcal{S}(C)$,

$$\pi(\underline{n}) = G(C)^{-1} \prod_{r \in \mathcal{R}} \frac{\nu_r^{n_r}}{n_r!}, \tag{1.8}$$

where $G(C)$ is the normalizing constant, or the partition function

$$G(C) = \sum_{\underline{n} \in \mathcal{S}(C)} \prod_{r \in \mathcal{R}} \frac{\nu_r^{\underline{n}_r}}{\underline{n}_r!}.$$

Let M be an upper bound on the number of active route calls for all routes. Then the distribution (1.8) can be understood as the following Markov random field defined for $\underline{n} \in M^K$.

$$\pi(\underline{n}) \propto \prod_{r \in \mathcal{R}} \frac{\nu_r^{n_r}}{n_r!} \prod_{j=1}^{J} g_j(n_{\mathcal{R}_j}), \tag{1.9}$$

where $\mathcal{R}_j$ is the set of routes that uses link j, and $g_j(\cdot)$ is a potential function defined as

$$g_j(n_{\mathcal{R}_j}) = \begin{cases} 1 & \text{if } \sum_{r \in \mathcal{R}_j} A_{jr} n_r \leq C_j \\ 0 & \text{otherwise} \end{cases}. \tag{1.10}$$

The underlying graph of the above MRF is the factor graph of π.

Definition 3 *A factor graph is a bipartite graph that expresses the factorization structure of a multivariate function as π defined in (1.9): 1) it has a variable node for each variable n_r; 2) it has a factor node for each local function g_j; 3) there is an edge connecting variable node n_r to factor node g_j if and only if n_r is an argument of g_j.*

For this model, a main algorithmic question of interest is computing the probability over time that a call on a route will be lost. This probability is called the *loss probability* of the route. In Chapter 4, we will discuss a well known approach for this computation called Erlang's fixed-point approximation, and its limitation. Then we will provide a novel algorithm for this problem and its comparison with Erlang's approximation.

1.4 Main contributions of this thesis

Exact computation of MAP assignment and log-partition function for an MRF on general graph are known to be NP-hard, and sometimes they are even hard to approximate within $n^{1-o(1)}$ factor [1, 60]. However, applications require solving this problem using very simple algorithms. Hence designing efficient and easily implementable approximate algorithms for those problems for practical MRFs is of main algorithmic challenge.

One popular approach in the literature is that of the Belief Propagation algorithm and its deviations including Tree-reweighted algorithm, which exploit large "girth" (i.e. lack of short cycles) of the graphical structure. While these algorithms have been successful in situations like error-correcting codes where one has "freedom" to design the graph of code, in applications like wireless network, stochastic loss network, image processing, and statistical physics, this is not the case. Specifically, popular graphical models for wireless network, and image processing do have lots of short cycles.

In this thesis, we identify a new class of structural properties of MRF, that yield to simple, and efficient algorithms. Specifically, for MRFs defined on graphs that have some "geometry" (which we call as graphs with polynomial growth, see Section 2.3.1) or graphs which are minor-excluded (for example, planar graphs, see Section 2.3.2) with bounded degree, we design almost linear time algorithms for approximate computation of MAP assignment and log-partition function within arbitrary accuracy. The graphical models arising in wireless networks, statistical physics, and image processing do possess such graphical structure. We also provide a simple novel algorithm for MAP computation based on local updates of a MAP assignment. Our algorithm can be implemented in a distributed manner, and we show that for graphs with polynomial growth, our algorithm computes approximate MAP within arbitrary accuracy. These algorithms are presented in Chapter 2. A subset of the results of Chapter 2 appeared in [25].

Next, we turn our attention to the application of our algorithms to network problems. In Chapter 3, we consider a wireless network of n nodes placed in some ge-

ographic area in an arbitrary manner. These nodes communicate over a common wireless medium under some interference constraints. Our work is motivated by the need for an efficient algorithm to determine the n^2 dimensional unicast capacity region of such a wireless network. Equivalently, we consider the problem of determining whether a given vector of end-to-end rates between various source-destination pairs can be supported by the network through a combination of routing and scheduling decisions among more than exponentially many possible choices in n.

This question is known to be NP-hard and hard to even approximate within $n^{1-o(1)}$ factor for general graphs [57]. We consider wireless networks which are usually formed between nodes that are placed in a geographic area and come endowed with a certain "geometry", and show that such situations do lead to approximations to the MWIS problem by applying our approximate computation of MAP algorithm to the corresponding MRF. Consequently, this gives us distributed polynomial time algorithm to approximate the capacity of wireless networks to arbitrary accuracy. This result hence, is in sharp contrast with previous works that provide centralized algorithms with at least a constant factor loss. Results reported in this Chapter appeared in [13, 14, 26].

In Chapter 4, we investigate the problem of computing loss probabilities of routes in a stochastic loss network, which is equivalent to computing partition function of the corresponding MRF for the exact stationary distribution. We show that the very popular Erlang fixed-point approximation provides relatively poor performance estimates, especially when the loss network is critically loaded. Then based on the structural property of the corresponding MRF, we propose a novel algorithm for estimating the stationary loss probabilities in stochastic loss networks, and show that our algorithm always converges exponentially fast to the asymptotically exact solutions. Using a variational characterization of the stationary distribution, we also provide an alternative proof for an error upper bound of the Erlang approximation. A previous version of this Chapter appeared in [24].

Finally, Chapter 5 concludes with a summary and directions for the future work.

Chapter 2

Approximate Inference Algorithm

In this Chapter, we present new approximate inference algorithms for computation of MAP assignment and log-partition function for two important classes of pairwise MRFs based on the underlying graph structure of the MRF. The first class is the graphs that are polynomially growing, and the second one is the graphs that exclude a finite-sized graph as a minor and has a constant vertex degree bound. For the class of polynomially growing graph, we also provide an intuitively pleasing sequential randomized algorithm for efficient inference of MAP.

2.1 Previous work

A plausible approach for designing simple algorithms for computation of MAP and log-partition function is as follows. First, identify a wide class of graphs that have simple algorithms for computing MAP and log-partition function. Then, for any given graph, approximately compute a solution by possibly solving multiple sub-problems that have good graph structures and then combining the results from these sub-problems to obtain a global solution.

Such an approach has resulted in many interesting recent results starting from the Belief Propagation (BP) algorithm designed for tree graphs [49]. Since there is a vast literature on this topic, we will recall only few results. Two important algorithms are the generalized belief propagation (BP) [71] and the tree-reweighted

algorithm (TRW) [65–67]. Key properties of interest for these iterative procedures are the correctness of fixed points and convergence. Many results characterizing properties of the fixed points are known starting [71]. Various sufficient conditions for their convergence are known starting [62]. However, simultaneous convergence and correctness of such algorithms are established for only specific problems, e.g. [3,34,46].

Now, we discuss two results relevant to this chapter. The first result is about properties of TRW. The TRW algorithm provides provable upper bound on log-partition function for arbitrary graph [67]. However, to the best of our knowledge the error is not quantified. The TRW for MAP estimation has a strong connection to specific Linear Programming (LP) relaxation of the problem [66]. This was made precise in a sequence of work by Kolmogorov [33], Kolmogorov and Wainwright [34] for binary MRF. It is worth noting that LP relaxation can be poor even for simple problems.

Another work is an approximation algorithm proposed by Globerson and Jaakkola [12] to compute log-partition function using Planar graph decomposition (PDC). PDC uses techniques of [67] in conjunction with known result about exact computation of partition function for binary MRF when G is Planar and the exponential family has very specific form. Their algorithm provides provable upper bound for arbitrary graph. However, they do not quantify the error incurred. Further, their algorithm is limited to binary MRFs.

2.2 Outline of our algorithm

We propose a novel local algorithm for approximate computation of MAP and log-partition function. For any $\varepsilon > 0$, our algorithm can produce an ε-approximate solution for MAP and log-partition function for *arbitrary positive* MRF G as long as G has either of these two properties:

- G is a polynomially growing graph (see Section 2.3.1, Theorems 2 and 6),
- G excludes a finite-sized graph as a minor (see Theorems 3 and 7) and has constant maximum degree.

We say MRF $\mathbf{X}$ is a positive MRF when $\mathbf{X}$ is given as follows: for $\mathbf{x} \in \Sigma^n$,

$$\mathbb{P}[\mathbf{X} = \mathbf{x}] \quad \propto \quad \exp\left(\sum_{v \in V} \phi_v(x_v) + \sum_{(u,v) \in E} \psi_{uv}(x_u, x_v)\right), \tag{2.1}$$

where $\phi_v : \Sigma \to \mathbb{R}_+$ and $\psi_{uv} : \Sigma^2 \to \mathbb{R}_+$. We note that the assumption of ϕ_v, ψ_{uv} being non-negative does not incur loss of generality for the following reasons: (a) the distribution remains the same if we consider potential functions $\phi_v + C_v, \psi_{uv} + C_{uv}$, for all $v \in V, (u,v) \in E$ with constants C_v, C_{uv}; and (b) by selecting large enough constant, the modified functions will become non-negative as they are defined over finite discrete domain. The representation (2.1) is called a *exponential family distribution.*

Our algorithm is primarily based on the following steps.

- First, decompose G into small-size connected components say $G_1, \ldots, G_k$ by removing few edges of G.
- Second, compute estimates (either MAP or log-partition) in each of G_i separately.
- Third, combine these estimates to produce a global estimate while *taking care* of the effect induced by removed edges.

In general, our algorithm works for any G and we can quantify bound on the error incurred by our algorithm. It is worth noting that our algorithm provides a provable lower bound on log-partition function as well unlike many of previous works. We show that the error in the estimate depends only on the edges removed. This error bound characterization is applicable for arbitrary graph.

For obtaining sharp error bounds, we need good graph decomposition schemes. Specifically, we use a simple and very intuitive randomized decomposition scheme for graphs with polynomial growth. This decomposition is described in Section 2.4.2. For minor-excluded graphs, we use a simple scheme based on work by Klein, Plotkin and Rao [32] and Rao [50] that they had introduced to study the gap between max-flow

and min-cut for multicommodity flows. This decomposition scheme is described in Section 2.4.3. In general, as long as G allows for such good edge-set for decomposing G into small components, our algorithm will provide a good estimate.

To compute estimates in individual components, we use dynamic programming. Since each component is small, it is not computationally burdensome. However, one may obtain further simpler heuristics by replacing dynamic programming by other method such as BP or TRW for computation in the components.

The running time of our algorithms are $\Theta(n)$, with the constant dependent on ε and (a) growing rate for polynomially growing graph, or (b) maximum vertex degree and size of the graph that is excluded as minor for minor-excluded graphs. For example, for 2-dimensional grid graph, which has growth rate $O(1)$, the algorithm takes $C(\varepsilon)n$ time, where $\log\log C(\varepsilon) = O(1/\varepsilon)$. On the other hand, for a planar graph with constant maximum vertex degree, the algorithm takes $C'(\varepsilon)n$ time, with $\log\log C'(\varepsilon) = O(1/\varepsilon)$.

In Section 2.5.5, we develop an intuitively pleasing sequential randomized algorithm for approximate MAP computation for polynomially growing graphs. This is motivated by the property of the decomposition scheme for that graph class. This algorithm can be implemented in a distributed manner in a natural way. We strongly believe that this algorithm will have great practical impact.

In Section 2.7, as an unexpected consequence of these algorithmic results, we obtain a method to establish existence of asymptotic limits of free energy for a class of MRF. Specifically, we show that if the MRF is d-dimensional grid, and all node, edge potential functions are identical, then the free-energy (i.e. normalized log-partition function) converges to a limit as the size of the grid grows to infinity. In general, such approach is likely to extend for any *regular enough* MRF for proving existence of such limit: for example, the result will immediately extend when one replaces the node, edge potential being exactly the same by they being chosen from a common distribution in an i.i.d. fashion.

Finally, in Section 2.8, we present numerical experiments which convincingly show that our algorithms are very competitive to other recently successful algorithms in-

cluding TRW and PDC.

2.3 Graph classes

In this section, we explain the two class of graphs for which we obtain approximate inference algorithms.

2.3.1 Polynomially growing graph

Definition 4 *Let* $\mathbf{d}_G$ *be the shortest path distance metric of a given graph* G*, and let* $\mathbf{B}_G(v,r) = \{w \in V | \mathbf{d}_G(w,v) < r\}$*. If there are constants* $C > 0$ *and* $\rho > 0$ *so that for any* $v \in V$ *and* $r \in \mathbb{N}$*,*

$$|\mathbf{B}_G(v,r)| \leq C \cdot r^{\rho},$$

then we say G *is* polynomially growing *with* growth rate ρ *and corresponding constant* C*.*

Practical applications of MRF model including the following geometric network graphs and doubling dimensional graphs, satisfy the above property.

Example 1 : Geometric Graph. Consider a wireless network with n nodes represented by the vertices $V = \{1, \ldots, n\}$ placed in a 2-dimensional geographic region given by the $\sqrt{n} \times \sqrt{n}$ square of area n[1] in an *arbitrary* manner (not necessarily random). Let E be the set of edges between nodes indicating which pair of nodes can communicate. Let $\mathbf{d}_E(\cdot,\cdot)$ be the Euclidean distance of the Euclidean space. Given a vertex $v \in V$, let $\mathbf{B}_E(v,r) = \{u \in V : \mathbf{d}_E(u,v) < r\}$. We assume that the wireless network satisfies the following simple assumptions.

1. There is an $R > 0$ such that no two nodes having distance larger than R

[1]Placing the nodes in the specified square is for simple presentation. The same result holds when the nodes are placed in any Euclidean rectangle, and when the nodes are place in any region of k-dimensional Euclidean space.

can establish a communication edge with each other[2] where R is called the transmission radius.

2. Graph G has bounded density $D > 0$, i.e. for all $v \in V$, $\frac{|\mathbf{B}_E(v,R)|}{R^2} \leq D$.

A geometric random graph obtained by placing n nodes in the $\sqrt{n} \times \sqrt{n}$ square uniformly at random and connecting two nodes that are within distance $R = \Theta(\sqrt{\log n})$ of each other satisfies the previous assumptions with high probability.

Lemma 1 *Any geometric graphs satisfying the above two assumptions are polynomially growing with growth rate 2.*

Proof. Let G be a geometric graph with a transmission radius R and a bounded density D. First, note that in the Euclidean space, for any $r \geq R$, $\mathbf{B}_E(v,r)$ can be covered by $\Theta\left(\left(\frac{r}{R}\right)^2\right)$ many balls of radius R. Hence, together with the definition of the bounded density D, there is a constant $D' > 0$ so that for all $v \in V$ and $r \geq R$,

$$\frac{|\mathbf{B}_E(v,r)|}{r^2} \leq D'. \tag{2.2}$$

Now, for a given two connected vertices $v, w \in V$ of G, let $v = v_0, v_1, v_2 \ldots, v_\ell = w$ be a shortest path in G. By the definition of the transmission radius, for all $i = 0, 1 \ldots, (\ell - 1)$,

$$\mathbf{d}_E(v_i, v_{i+1}) \leq R.$$

By the triangular inequality in the Euclidean metric,

$$\mathbf{d}_E(v,w) \leq \sum_{i=0}^{\ell} \mathbf{d}_E(v_i, v_{i+1}) \leq R \cdot \ell.$$

So we obtain

$$\mathbf{d}_E(v,w) \leq R\ell = R \cdot \mathbf{d}_G(v,w).$$

[2]It does not imply that nodes within distance R must communicate.

Hence, for any $v \in V$ and $r \in \mathbb{N}$,

$$\mathbf{B}_G(v, r) \subset \mathbf{B}_E(v, Rr).$$

From (2.2),

$$|\mathbf{B}_G(v, r)| \leq |\mathbf{B}_E(v, Rr)| \leq \left(D'R^2\right) r^2,$$

which shows that the growth rate of G is 2.

Example 2 : Doubling Dimensional Graph. A graph is said to have a *doubling dimension* $\varphi > 0$ if any ball of radius $2r$ (w.r.t. the shortest path metric) in G can be covered by at most 2^{φ} many balls of radius r for any $r \in \mathbb{N}$. A graph with a constant doubling dimension is called *a doubling dimensional graph.* The notion of doubling dimensional graphs was first introduced in [2, 15, 17]. It is easy to check that a grid graph $\mathbb{Z}^d$ has doubling dimension d. Clearly, any graph with n nodes has doubling dimension at most $O(\log_2 n)$. The following Lemma shows that any doubling dimensional graph is polynomially growing.

Lemma 2 *A graph with a constant doubling dimension φ is polynomially growing with growth rate φ.*

Proof. First, we will show that for any $x \in V$ and any $t \in \mathbb{Z}_+$,

$$|\mathbf{B}_G(x, 2^t)| \leq 2^{t\varphi}. \tag{2.3}$$

The proof of (2.3) is by induction on $t \in \mathbb{Z}_+$. For the base case, consider $t = 0$. Now, $\mathbf{B}_G(x, 2^0)$ is essentially the set of all points which are at distance less than 1 from x by the definition. Since it is metric with distance being integer, this means that $\mathbf{B}_G(x, 1) = \{x\}$. Hence, $|\mathbf{B}(x, 1)| = 1 \leq 2^{0 \times \rho(\mathcal{M})}$ for all $x \in \mathcal{X}$.

Now suppose that the claim of Lemma is true for all $t \leq k$ and all $x \in \mathcal{X}$. Consider $t = k+1$ and any $x \in \mathcal{X}$. By definition of the doubling dimension, there exists $\ell \leq 2^{\varphi}$

balls of radius 2^k, say $\mathbf{B}_G(y_j, 2^k)$ with $y_j \in \mathcal{X}$ for $1 \leq j \leq \ell$, such that

$$\mathbf{B}_G(x, 2^{k+1}) \subset \cup_{j=1}^{\ell} \mathbf{B}_G(y_j, 2^k).$$

Therefore,

$$|\mathbf{B}_G(x, 2^{k+1})| \leq \sum_{j=1}^{\ell} |\mathbf{B}_G(y_j, 2^k)|.$$

By inductive hypothesis, for $1 \leq j \leq \ell$,

$$|\mathbf{B}_G(y_j, 2^k)| \leq 2^{k\varphi}.$$

Since we have $\ell \leq 2^{\varphi}$, we obtain

$$|\mathbf{B}_G(x, 2^{k+1})| \leq \ell \; 2^{k\varphi} \;\leq\; 2^{(k+1)\varphi}.$$

This completes the proof of inductive step, and that of (2.3).

Now, for any $r \in \mathbb{N}$, and any $x \in V$, let $2^t \leq r < 2^{t+1}$ for $t \in \mathbb{Z}_+$. From (2.3), we obtain that

$$|\mathbf{B}_G(x, r)| \leq |\mathbf{B}_G(x, 2^{t+1})| \leq 2^{(t+1)\varphi} = 2^{\varphi} \cdot (2^t)^{\varphi} \leq 2^{\varphi} \cdot r^{\varphi}, \tag{2.4}$$

which shows the Lemma.

Property of polynomially growing graphs. The following Lemma shows that any subgraph of a polynomially growing graph is also a polynomially growing graph.

Lemma 3 *If G is polynomially growing with growth rate ρ, any subgraph $\hat{G} = (\hat{V}, \hat{E})$ of G obtained by removing some edges and vertices of G is also polynomially growing with growth rate at most ρ.*

Proof. For any vertex $v, w \in \hat{V}$, note that

$$\mathbf{d}_{\hat{G}}(v, w) \geq \mathbf{d}_G(v, w),$$

since any path in $\hat{G}$ from v to w is also a path in G. Hence, for any $v \in \hat{V}$ and $r \in \mathbb{N}$,

$$\mathbf{B}_{\hat{G}}(v, r) \subset \mathbf{B}_G(v, r).$$

Hence,

$$|\mathbf{B}_{\hat{G}}(v, r)| \leq |\mathbf{B}_G(v, r)| \leq r^{\rho},$$

which shows the Lemma from the definition 4. For example, any subgraph of a grid graph is a polynomially growing graph.

2.3.2 Minor-excluded graph

Next, we introduce a class of graphs known as *minor-excluded* graphs (see a series of publications by Roberston and Seymour under *the graph minor theory project* [52]). A graph H is called minor of G if we can transform G into H through an arbitrary sequence of the following two operations:

- removal of an edge.
- merge two connected vertices u, v: that is, remove edge (u, v) as well as vertices u and v; add a new vertex and make all edges incident on this new vertex that were incident on u or v.

Now, if H is not a minor of G then we say that G excludes H as a minor.

The explanation of the following statement may help to understand the definition better: *any graph H with r nodes is a minor of K_r*, where K_r is a complete graph of r nodes. This is true because one may obtain H by removing edges from K_r that are absent in H. More generally, if G is a subgraph of G' and G has H as a minor, then G' has H as its minor. Let $K_{r,r}$ denote a complete bipartite graph with r nodes in each partition. Then K_r is a minor of $K_{r,r}$. Hence, any graph H with r nodes is a minor of $K_{r,r}$. An important implication of this is as follows: to prove property P for graph G that excludes H, of size r, as a minor, it is sufficient to prove that any graph that excludes $K_{r,r}$ as a minor has property P. This fact was cleverly used by

Klein et. al. [32]. In what follows and the rest of the paper, we will always assume r to be some finite number that does not scale with n (the number of nodes in G).

2.4 Graph decomposition

In this section, we introduce notion of our *graph decomposition.* We describe very simple algorithms for obtaining decomposition for graphs with polynomial growth and minor-excluded graphs.

2.4.1 (ε, Δ) decomposition

Given $\varepsilon, \Delta > 0$, we define notion of (ε, Δ) decomposition for a graph $G = (V, E)$. This notion can be stated in terms of vertex-based decomposition or edge-based decomposition.

Definition 5 *We call a random subset of vertices $\mathcal{B} \subset V$ as* (ε, Δ) vertex-decomposition *of G if the followings hold:*

(a) *For any $v \in V$, $\mathbb{P}(v \in \mathcal{B}) \leq \varepsilon$.*

(b) *Let $S_1, \ldots, S_\ell$ be the connected components of graph $G' = (V', E')$ where $V' = V \backslash \mathcal{B}$ and $E' = \{(u, v) \in E : u, v \in V'\}$. Then, $\max_{1 \leq k \leq \ell} |S_k| \leq \Delta$ with probability* 1.

Note that the (ε, Δ) vertex-decomposition $\mathcal{B}$ forms the union of boundary vertices of each connected components.

Definition 6 *Similarly, a random subset of edges $\mathcal{B} \subset E$ is called an (ε, Δ) edge-decomposition of G if the following holds:*

(a) *For any $e \in E$, $\mathbb{P}(e \in \mathcal{B}) \leq \varepsilon$.*

(b) *Let $S_1, \ldots, S_K$ be connected components of graph $G' = (V', E')$ where $V' = V$ and $E' = E \backslash \mathcal{B}$. Then, $\max_{1 \leq k \leq K} |S_k| \leq \Delta$ with probability* 1.

2.4.2 Graph decomposition for polynomially growing graphs

This section presents (ε, Δ) decomposition algorithm for polynomially growing graphs for various choice of ε and Δ. We will describe algorithm for node-based (ε, Δ) decomposition. This will immediately imply algorithm for edge-based decomposition for the following reason: given $G = (V, E)$ with growth rate $\rho(G)$, consider a graph of its edges $\mathcal{G} = (E, \mathcal{E})$ where $(e, e') \in \mathcal{E}$ if e, e' shared a vertex in G. It is easy to check that $\rho(\mathcal{G}) \leq \rho(G)$. Therefore, running algorithm for node-based decomposition on $\mathcal{G}$ will provide an edge-based decomposition.

The node-based decomposition algorithm for G will be described for the metric space on V with respect to the shortest path metric $\mathbf{d_G}$ introduced earlier. Clearly, it is not possible to have (ε, Δ) decomposition for any ε and Δ values. As will become clear later, it is important to have such decomposition for ε and Δ being not too large. Therefore, we describe algorithm for any $\varepsilon > 0$ and an operational parameter K, which will depend on ε and the growth rate ρ and the corresponding constant C of the graph. We will show that our algorithm will output (ε, Δ)-decomposition where Δ will depend on ε and K.

Given ε and K, define a random variable $\mathbf{Q}$ over $\{1, \ldots, K\}$ as

$$\mathbb{P}[\mathbf{Q} = i] = \begin{cases} \varepsilon(1-\varepsilon)^{i-1} & \text{if } 1 \leq i < K \\ (1-\varepsilon)^{K-1} & \text{if } i = K \end{cases}.$$

The graph decomposition algorithm **POLY-V** (ε, K) described next essentially does the following. The algorithm performs iteratively. Initially, all vertices are colored *white*. If there is any *white* vertex, choose any of them *arbitrarily*. Let u be the chosen vertex. Draw an independent random number Q as per distribution $\mathbf{Q}$. Select all *white* vertices that are at distance Q from u in $\mathcal{B}$ and color them *blue*; color all *white* vertices at distance $< Q$ from u (including u itself) as *red*. Repeat this process until no more *white* vertices are left. Output $\mathcal{B}$ (i.e. *blue* nodes).

POLY-V (ε, K)

(1) Initially, set $\mathcal{W} = V$, $\mathcal{B} = \emptyset$ and $\mathcal{R} = \emptyset$.

(2) Repeat the following till $\mathcal{W} \neq \emptyset$:

  (a) Choose an element $u \in \mathcal{W}$ uniformly at random.

  (b) Draw a random number Q independently according to the distribution $\mathbf{Q}$.

  (c) Update

    (i) $\mathcal{B} \leftarrow \mathcal{B} \cup \{w | \mathbf{d_G}(u, w) = Q \text{ and } w \in \mathcal{W}\}$,

    (ii) $\mathcal{R} \leftarrow \mathcal{R} \cup \{w | \mathbf{d_G}(u, w) < Q \text{ and } w \in \mathcal{W}\}$,

    (iii) $\mathcal{W} \leftarrow \mathcal{W} \cap (\mathcal{B} \cup \mathcal{R})^c$.

(3) Output $\mathcal{B}$.

Figure 2-1: Vertex-decomposition for polynomially growing graphs

Precise description of the algorithm is in Figure 2-1. We will set

$$K = K(\varepsilon, \rho, C) = \frac{8\rho}{\varepsilon} \log\left(\frac{8\rho}{\varepsilon}\right) + \frac{4}{\varepsilon} \log C + \frac{4}{\varepsilon} \log \frac{1}{\varepsilon} + 2.$$

This definition is exploited in the proof of Lemma 4. Figure 2-2 explains the algorithm **POLY-V** up to three choices of u.

Lemma 4 *Given graph G with growth rate $\rho = \rho(G)$ and the corresponding constant C, and $\varepsilon \in (0, 1)$, the output of the* **POLY-V**(ε, K) *becomes a $(2\varepsilon, CK^\rho)$ vertex-decomposition of G.*

Proof. To prove that the random output set $\mathcal{B} \subset V$ of the algorithm with parameters $(\varepsilon, K(\varepsilon, \rho))$ we need to establish properties (a) and (b) of Definition 5.

Proof of (a). To prove (a), we state and prove the following Claim.

Claim 1 *Consider metric space $\mathcal{G} = (V, \mathbf{d_G})$ with $|V| = n$. Let $\mathcal{B} \subset V$ be the random set that is output of* **POLY-V** *with parameter (ε, K) applied to $\mathcal{G}$. Then, for*

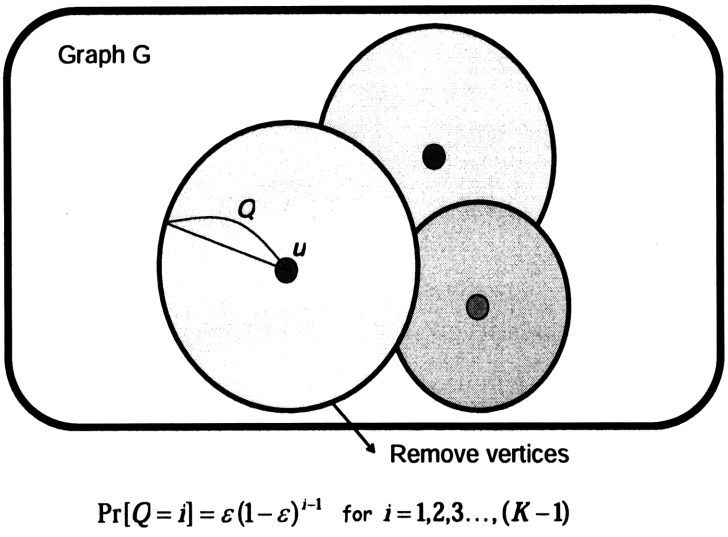

Figure 2-2: The first three iterations in the execution of **POLY-V**.

any $v \in V$,

$$\mathbb{P}[v \in \mathcal{B}] \leq \varepsilon + P_K |\mathbf{B}_G(v, K)|,$$

where $\mathbf{B}_G(v, K)$ *is the ball of radius* K *in* $\mathcal{G}$ *with respect to the* $\mathbf{d_G}$, *and* $P_K = (1 - \varepsilon)^{K-1}$.

Proof. The proof is by induction on the number of points n over which the metric space is defined. When $n = 1$, the algorithm chooses only point as u_0 in the initial iteration and hence it can not be part of the output set $\mathcal{B}$. That is, for this only point, say v,

$$\mathbb{P}[v \in \mathcal{B}] = 0 \leq \varepsilon + P_K |\mathbf{B}(v, K)|.$$

Thus, we have verified the base case for induction ($n = 1$).

As induction hypothesis, suppose that the Claim 1 is true for any metric space on n points with $n < N$ for some $N \geq 2$. As the induction step, we wish to establish that for a metric space $\mathcal{G} = (V, \mathbf{d_G})$ with $|V| = N$, the Claim 1 is true. For this, consider any point $v \in V$. Now consider the first iteration of the **POLY-V** applied to $\mathcal{G}$. The algorithm picks $u_0 \in V$ uniformly at random in the first iteration. Given

v, depending on the choice of u_0 we consider four different cases (or events). We will show that in these four cases,

$$\mathbb{P}[v \in \mathcal{B}] \leq \varepsilon + P_K|\mathbf{B}_G(v, K)|$$

holds.

Case 1. This case corresponds to event E_1 where the chosen random u_0 is equal to point v of our interest. By definition of the algorithm, under the event E_1, v will never be part of output set $\mathcal{B}$. That is,

$$\mathbb{P}[v \in \mathcal{B}|E_1] = 0 \ \leq \ \varepsilon + P_K|\mathbf{B}(v, K)|.$$

Case 2. Now, suppose u_0 is such that $v \neq u_0$ and $\mathbf{d_G}(u_0, v) < K$. Call this event E_2. Further, depending on choice of random number Q_0, define the following events

$$E_{21} = \{\mathbf{d_G}(u_0, v) < Q_0\}, \ E_{22} = \{\mathbf{d_G}(u_0, v) = Q_0\}, \text{ and } E_{23} = \{\mathbf{d_G}(u_0, v) > Q_0\}.$$

By definition of the algorithm, when E_{21} happens, v is selected as part of $\mathcal{R}_1$ and hence v can never be a part of output $\mathcal{B}$. When E_{22} happens, v is selected as part of $\mathcal{B}_1$ and hence it is definitely a part of output set $\mathcal{B}$. When E_{23} happens, v is neither selected in set $\mathcal{R}_1$ nor selected in set $\mathcal{B}_1$. It is left as an element of the set $\mathcal{W}_1$. This new set $\mathcal{W}_1$ has points less than N. The original metric $\mathbf{d_G}$ is still the metric on the points[3] of $\mathcal{W}_1$. By definition, the algorithm only cares about $(\mathcal{W}_1, \mathbf{d_G})$ in the future and it is not affected by its decisions in past. Therefore, we can invoke induction hypothesis which implies that if event E_{23} happens then the probability of $v \in \mathcal{B}$ is bounded above by $\varepsilon + P_K|\mathbf{B}(v, K)|$. Finally, let us relate the $\mathbb{P}[E_{21}|E_2]$ with $\mathbb{P}[E_{22}|E_2]$. Suppose $\mathbf{d_G}(u_0, v) = \ell < K$. By definition of probability distribution of $\mathbf{Q}$, we have

$$\mathbb{P}[E_{22}|E_2] \ = \ \varepsilon(1-\varepsilon)^{\ell-1}, \tag{2.5}$$

[3]Note the following subtle but crucial point. We are not changing the metric $\mathbf{d_G}$ after we remove points from original set of points as part of the **POLY-V**.

$$\begin{aligned}\mathbb{P}[E_{21}|E_2] &= (1-\varepsilon)^{K-1} + \sum_{j=\ell+1}^{K-1} \varepsilon(1-\varepsilon)^{j-1} \\ &= (1-\varepsilon)^{\ell}. \end{aligned} \tag{2.6}$$

That is,

$$\mathbb{P}[E_{22}|E_2] = \frac{\varepsilon}{1-\varepsilon}\mathbb{P}[E_{21}|E_2].$$

Let $q \triangleq \mathbb{P}[E_{21}|E_2]$. Then,

$$\begin{aligned}\mathbb{P}[v \in \mathcal{B}|E_2] &= \mathbb{P}[v \in \mathcal{B}|E_{21} \cap E_2]\mathbb{P}[E_{21}|E_2] + \mathbb{P}[v \in \mathcal{B}|E_{22} \cap E_2]\mathbb{P}[E_{22}|E_2] \\ &\quad + \mathbb{P}[v \in \mathcal{B}|E_{23} \cap E_2]\mathbb{P}[E_{23}|E_2] \\ &= 0 \times q + 1 \times \frac{\varepsilon q}{1-\varepsilon} + (\varepsilon + P_K|\mathbf{B}(v,K)|)\left(1 - \frac{q}{1-\varepsilon}\right) \\ &= \varepsilon + P_K|\mathbf{B}(v,K)| + \frac{q}{1-\varepsilon}(\varepsilon - \varepsilon - P_K|\mathbf{B}(v,K)|) \\ &= \varepsilon + P_K|\mathbf{B}(v,K)| - \frac{qP_K|\mathbf{B}(v,K)|}{1-\varepsilon} \\ &\leq \varepsilon + P_K|\mathbf{B}(v,K)|. \end{aligned} \tag{2.7}$$

Case 3. Now, suppose $u_0 \neq v$ is such that $\mathbf{d_G}(u_0, v) = K$. We will call this event E_3. Further, define the event $E_{31} = \{Q_0 = K\}$. Due to the independence of selection of Q_0, $\mathbb{P}[E_{31}|E_3] = P_K$. Under the event $E_{31} \cap E_3$, $v \in \mathcal{B}$ with probability 1. Therefore,

$$\begin{aligned}\mathbb{P}[v \in \mathcal{B}|E_3] &= \mathbb{P}[v \in \mathcal{B}|E_{31} \cap E_3]\mathbb{P}[E_{31}|E_3] + \mathbb{P}[v \in \mathcal{B}|E_{31}^c \cap E_3]\mathbb{P}[E_{31}^c|E_3] \\ &= 1 \times P_K + \mathbb{P}[v \in \mathcal{B}|E_{31}^c \cap E_3](1 - P_K). \end{aligned} \tag{2.8}$$

Under the event $E_{31}^c \cap E_3$, we have $v \in \mathcal{W}_1$, and the remaining metric space $(\mathcal{W}_1, \mathbf{d_G})$. This metric space has $< N$ points. Further, the ball of radius K around v with respect to this new metric space has at most $|\mathbf{B}(v,K)| - 1$ points (this ball is with respect to the original metric space $\mathcal{G}$ on N points). Now we can invoke the induction hypothesis for this new metric space to obtain

$$\mathbb{P}[v \in \mathcal{B}|E_{31}^c \cap E_3] \leq \varepsilon + P_K(|\mathbf{B}(v,K)| - 1). \tag{2.9}$$

From (2.8) and (2.9), we have

$$\begin{aligned}\mathbb{P}[v \in \mathcal{B}|E_3] &\leq P_K + (1-P_K)(\varepsilon + P_K(|\mathbf{B}(v,K)|-1)) \\ &= \varepsilon(1-P_K) + P_K|\mathbf{B}(v,K)| + P_K^2(1-|\mathbf{B}(v,K)|) \\ &\leq \varepsilon + P_K|\mathbf{B}(v,K)|.\end{aligned}$$

Case 4. Finally, let E_4 be the event that $\mathbf{d_G}(u_0, v) > K$. Then, at the end of the first iteration of the algorithm, we again have the remaining metric space $(\mathcal{W}_1, \mathbf{d_G})$ such that $|\mathcal{W}_1| < N$. Hence, as before, by induction hypothesis we have

$$\mathbb{P}[v \in \mathcal{B}|E_4] \leq \varepsilon + P_K|\mathbf{B}(v,K)|.$$

Now, the four cases are exhaustive and disjoint. That is, $\cup_{i=1}^4 E_i$ is the universe. Based on the above discussion, we obtain the following.

$$\begin{aligned}\mathbb{P}[v \in \mathcal{B}] &= \sum_{i=1}^4 \mathbb{P}[v \in \mathcal{B}|E_i]\mathbb{P}[E_i] \\ &\leq \left(\max_{i=1}^4 \mathbb{P}[v \in \mathcal{B}|E_i]\right)\left(\sum_{i=1}^4 \mathbb{P}[E_i]\right) \\ &\leq \varepsilon + P_K|\mathbf{B}(v,K)|. \qquad (2.10)\end{aligned}$$

This completes the proof of Claim 1.

Now, we will use Claim 1 to complete the proof of (a). The definition of growth rate implies that,

$$|\mathbf{B}(v,K)| \leq C \cdot K^\rho.$$

From the definition $P_K = (1-\varepsilon)^{K-1}$, we have

$$P_K\,|\mathbf{B}(v,K)| \leq C(1-\varepsilon)^{K-1}K^\rho.$$

Therefore, to show (a) of Definition 5, it is sufficient to show that our definition of K satisfies the following Lemma.

Lemma 5 *We have that*

$$C(1-\varepsilon)^{K-1}K^{\rho} \leq \varepsilon.$$

Proof. We will show the following equivalent inequality.

$$(K-1)\log(1-\varepsilon)^{-1} \geq \rho \log K + \log C + \log \frac{1}{\varepsilon}. \tag{2.11}$$

First, note that for all $\varepsilon \in (0,1)$,

$$\log(1-\varepsilon)^{-1} \geq \log(1+\varepsilon) \geq \frac{\varepsilon}{2}.$$

Hence to prove (2.11), it is sufficient to show that

$$K \geq \frac{2\rho}{\varepsilon}\log K + \frac{2}{\varepsilon}\log C + \frac{2}{\varepsilon}\log\frac{1}{\varepsilon} + 1. \tag{2.12}$$

Recall that

$$K = K(\varepsilon,\rho) = \frac{8\rho}{\varepsilon}\log\left(\frac{8\rho}{\varepsilon}\right) + \frac{4}{\varepsilon}\log C + \frac{4}{\varepsilon}\log\frac{1}{\varepsilon} + 2.$$

From the definition of K, we will show that

$$\frac{K}{2} \geq \frac{2\rho}{\varepsilon}\log K$$

and

$$\frac{K}{2} \geq \frac{2}{\varepsilon}\log C + \frac{2}{\varepsilon}\log\frac{1}{\varepsilon} + 1,$$

which will prove (2.12).

The following is trivial.

$$\frac{K}{2} \geq \frac{2}{\varepsilon}\log C + \frac{2}{\varepsilon}\log\frac{1}{\varepsilon} + 1. \tag{2.13}$$

Now, let $\hat{K} = \frac{8\rho}{\varepsilon}\log\left(\frac{8\rho}{\varepsilon}\right)$. Then

$$\frac{\hat{K}}{2} = \frac{4\rho}{\varepsilon}\log\left(\frac{8\rho}{\varepsilon}\right) \geq \frac{2\rho}{\varepsilon}\left(\log\left(\frac{8\rho}{\varepsilon}\right) + \log\log\left(\frac{8\rho}{\varepsilon}\right)\right) = \frac{2\rho}{\varepsilon}\log\hat{K}.$$

That is, $\frac{\hat{K}}{2} - \frac{2\rho}{\varepsilon}\log\hat{K} \geq 0$. Since the function $\phi(x) = \frac{x}{2} - \frac{2\rho}{\varepsilon}\log x$ is an increasing function of x when $x \geq \frac{4\rho}{\varepsilon}$, and from the fact that $K \geq \hat{K} \geq \frac{4\rho}{\varepsilon}$, we have

$$\frac{K}{2} \geq \frac{2\rho}{\varepsilon}\log K. \tag{2.14}$$

From (2.13) and (2.14), we have (2.12), which completes the proof of Lemma 5.

Proof of (b). First we give some notations. Define $R_t = \mathcal{R}_t - \mathcal{R}_{t-1}$, $B_t = \mathcal{B}_t - \mathcal{B}_{t-1}$, and

$$\partial R_t = \{v \in V : v \notin R_t \text{ and } \exists\, v' \in R_t \text{ s.t. } \mathbf{d_G}(v, v') = 1\}.$$

Then the followings are straightforward observations implied by the **POLY-V**. For any $t \geq 0$, (i) $R_t \cap \mathcal{R}_{t-1} = \emptyset$,

(ii) $B_t \cap \mathcal{B}_{t-1} = \emptyset$,

(iii) $R_t \subset \mathbf{B}(u_{t-1}, Q_{t-1})$,

(iv) $B_t \subset \mathbf{B}(u_{t-1}, Q_{t-1}+1) - \mathbf{B}(u_{t-1}, Q_{t-1})$.

Now, we state and prove a crucial claim for proving (b).

Claim 2 *For all $t \geq 0$, $\partial R_t \subset \mathcal{B}_t$.*

Proof. We prove the Claim 2 by induction. Initially, $\partial R_0 = \mathcal{B}_0 = \emptyset$ and hence the claim is trivial. At the end of the first iteration, by the definition of the algorithm,

$$R_1 = \mathcal{R}_1 = \mathbf{B}(u_0, Q_0), \text{ and } B_1 = \mathcal{B}_1 = \mathbf{B}(u_0, Q_0+1) - \mathbf{B}(u_0, Q_0).$$

Therefore, by definition, $\partial R_1 = \mathcal{B}_1$. Thus, the base case of induction is verified. Now, as the hypothesis for induction, suppose that $\partial R_t \subset \mathcal{B}_t$ for all $t \leq \ell$, for some $\ell \geq 1$. As induction step, we will establish that $\partial R_{\ell+1} \subset \mathcal{B}_{\ell+1}$.

Suppose to the contrary, that $\partial R_{\ell+1} \not\subset \mathcal{B}_{\ell+1}$. That is, there exists $v \in \partial R_{\ell+1}$ such that $v \notin \mathcal{B}_\ell$. By definition of the algorithm, we have

$$R_{\ell+1} = \mathbf{B}(u_\ell, Q_\ell) - (\mathcal{R}_\ell \cup \mathcal{B}_\ell).$$

Therefore,

$$\partial R_{\ell+1} \subset (\mathbf{B}(u_\ell, Q_\ell + 1) - \mathbf{B}(u_\ell, Q_\ell)) \cup \mathcal{R}_\ell \cup \mathcal{B}_\ell.$$

Again, by the definition of the algorithm we have

$$B_{\ell+1} = \mathbf{B}(u_\ell, Q_\ell + 1) - \mathbf{B}(u_\ell, Q_\ell) - \mathcal{R}_\ell - \mathcal{B}_\ell.$$

Therefore, $v \in B_{\ell+1}$ or $v \in \mathcal{R}_\ell \cup \mathcal{B}_\ell$. Recall that by the definition of algorithm $\mathcal{B}_\ell \cap \mathcal{R}_\ell = \emptyset$. Since we have assumed that $v \notin \mathcal{B}_{\ell+1}$, it must be that $v \in \mathcal{R}_\ell$. That is, there exists $\ell' \leq \ell$ such that $v \in R_{\ell'}$. Now since $v \in \partial R_{\ell+1}$ by assumption, it must be that there exists $v' \in R_{\ell+1}$ such that $\mathbf{d_G}(v, v') = 1$. Since by definition $R_{\ell+1} \cap R_{\ell'} = \emptyset$, we have $v' \in \partial R_{\ell'}$. By induction hypothesis, this implies that $v' \in \mathcal{B}_{\ell'} \subset \mathcal{B}_\ell$. That is, $\mathcal{B}_\ell \cap R_{\ell+1} \neq \emptyset$, which is a contradiction to the definition of our algorithm. That is, our assumption that $\partial R_{\ell+1} \not\subset \mathcal{B}_{\ell+1}$ is false. Thus, we have established the inductive step. This completes the induction argument and proof of the Claim 2. Now when the algorithm terminates (which must happen within n iterations), say the output set is $\mathcal{B}_T$ and $V - \mathcal{B}_T = \mathcal{R}_T$ for some T. As noted above, $\mathcal{R}_T$ is a union of disjoint sets $R_1, \ldots, R_T$. We want to show that R_i, R_j are disconnected for any $1 \leq i < j \leq T$ using Claim 2. Suppose to the contrary that they are connected. That is, there exists $v \in R_i$ and $v' \in R_j$ such that $\mathbf{d_G}(v, v') = 1$. Since $R_i \cap R_j = \emptyset$, it must be that $v' \in \partial R_i, v \in \partial R_j$. From Claim 2 and fact that $\mathcal{B}_t \subset \mathcal{B}_{t+1}$ for all t, we have that $R_i \cap \mathcal{B} \neq \emptyset, R_j \cap \mathcal{B} \neq \emptyset$. This is contrary to the definition of the algorithm. Thus, we have established that $R_1, \ldots, R_T$ are disconnected components whose union is $V - \mathcal{B}_T$. By definition, each of $R_i \subset \mathbf{B}(u_{i-1}, K)$. Thus, we have established that $V - \mathcal{B}_T$ is made of connected components, each of which is contained inside balls of radius K with respect to $\mathbf{d_G}$. From the definition of the growth rate of a graph, this completes

POLY-E(ε, K)

(1) Initially, set $\mathcal{W} = V$, $\mathcal{B} = \emptyset$ and $\mathcal{R} = \emptyset$.

(2) Repeat the following till $\mathcal{W} \neq \emptyset$:

  (a) Choose an element $u \in \mathcal{W}$ uniformly at random.

  (b) Draw a random number Q independently according to the distribution $\mathbf{Q}$.

  (c) Update

    (i) $\mathcal{B} \leftarrow \mathcal{B} \cup \{w \in V' | \mathbf{d_G}(u, w) = Q \text{ and } w \in \mathcal{W}\}$,

    (ii) $\mathcal{R} \leftarrow \mathcal{R} \cup \{w \in V | \mathbf{d_G}(u, w) < Q \text{ and } w \in \mathcal{W}\}$,

    (iii) $\mathcal{W} \leftarrow \mathcal{W} \cap \mathcal{R}^c$.

(3) Output $\mathcal{B}$.

Figure 2-3: Edge-decomposition for polynomially growing graphs

the proof of (b) and that of Lemma 4.

Now, in a similar fashion we obtain a $(2\varepsilon, CK^\rho)$ edge-decomposition of G. Let $G' = (V', E')$ be the graph obtained from G by adding one vertex at the center of each edges of G. If G is a polynomially growing graph ρ and the corresponding constant C, then G' is also a polynomially growing graph ρ and the corresponding constant C. As before we set

$$K = K(\varepsilon, \rho, C) = \frac{8\rho}{\varepsilon} \log\left(\frac{8\rho}{\varepsilon}\right) + \frac{4}{\varepsilon} \log C + \frac{4}{\varepsilon} \log \frac{1}{\varepsilon} + 2.$$

Then define a random variable $\mathbf{Q}'$ over $\{1, 3, 5, \ldots, 2K-1\}$ as

$$\mathbb{P}[\mathbf{Q} = i] = \begin{cases} \varepsilon(1-\varepsilon)^{(i-1)/2} & \text{if } 1 \leq i < 2K-1 \\ (1-\varepsilon)^{K-1} & \text{if } i = 2K-1 \end{cases}.$$

Let V be the set of vertices of G, which is a subset of V'. Then figure 2-3 describes the edge-decomposition algorithm **POLY-E**.

Note that the output of **POLY-E** consists of vertices of G' that does not belong

to V, i.e. vertices of G' that correspond to edges of G. By applying Lemma 4 to G', we obtain the following Lemma.

Lemma 6 *Given graph G with growth rate $\rho = \rho(G)$ and the corresponding constant C, and $\varepsilon \in (0,1)$, the output of the* **POLY-E** *(ε, K) becomes a $(2\varepsilon, CK^{\rho})$ edge-decomposition of G.*

2.4.3 Graph decomposition for minor-excluded graphs

In this section we describe a simple and explicit construction of decomposition for graphs that exclude certain finite sized graphs as their minor. This scheme is a direct adaptation of a scheme proposed by Klein, Plotkin, Rao [32] and Rao [50]. We describe an (ε, Δ) node-decomposition scheme. Later, we describe how it can be modified to obtain (ε, Δ) edge-decomposition.

Suppose, we are given a graph G that excludes the graph $K_{r,r}$ as minor. Recall that if a graph excludes some graph G_r of r nodes as its minor then it excludes $K_{r,r}$ as its minor as well. The algorithm described in Figure 2-4 for generating node-decomposition uses a parameter Λ. Later we will relate the parameter Λ to the decomposition property of the output.

As stated above, the basic idea is to use the following step recursively (upto depth r of recursion): in each connected component, say S, choose a node arbitrarily and create a breadth-first search tree, say $\mathcal{T}$. Choose a number, say L, uniformly at random from $\{0, \ldots, \Lambda - 1\}$. Remove (and add to $\mathcal{B}$) all nodes that are at level $L + k\Lambda, k \geq 0$ in $\mathcal{T}$. Clearly, the total running time of such an algorithm is $O(r(n + |E|))$ for a graph $G = (V, E)$ with $|V| = n$; with possible parallel implementation across different connected components.

Figure 2-5 explains the algorithm for a line-graph of $n = 9$ nodes, which excludes $K_{2,2}$ as a minor. The example is about a sample run of **MINOR-V**$(G, 2, 3)$ (Figure 2-5 shows the first iteration of the algorithm).

In [50], Rao proves the following Lemma.

MINOR-V(G, r, Λ)

(0) Input is graph $G = (V, E)$ and $r, \Lambda \in \mathbb{N}$. Initially, $i = 0$, $G_0 = G$, $\mathcal{B} = \emptyset$.

(1) For $i = 0, \ldots, r-1$, do the following.

  (a) Let $S_1^i, \ldots, S_{k_i}^i$ be the connected components of G_i.

  (b) For each $S_j^i, 1 \leq j \leq k_i$, pick an arbitrary node $v_j \in S_j^i$.

  - Create a breadth-first search tree $\mathcal{T}_j^i$ rooted at v_j in S_j^i.
  - Choose a number L_j^i uniformly at random from $\{0, \ldots, \Lambda - 1\}$.
  - Let $\mathcal{B}_j^i$ be the set of nodes at level $L_j^i, \Lambda + L_j^i, 2\Lambda + L_j^i, \ldots$ in $\mathcal{T}_j^i$.
  - Update $\mathcal{B} = \mathcal{B} \cup_{j=1}^{k_i} \mathcal{B}_j^i$.

  (c) set $i = i + 1$.

(3) Output $\mathcal{B}$ and graph $G' = (V, E \backslash \mathcal{B})$.

Figure 2-4: Vertex-decomposition for minor-excluded graphs

Lemma 7 *If G excludes $K_{r,r}$ as a minor. Let $\mathcal{B}$ be the output of* **MINOR-V**(G, r, Λ). *Then each connected component of $V - \mathcal{B}$ has diameter of size $O(\Lambda)$.*

Now using Lemma 7, we obtain the following Lemma.

Lemma 8 *Suppose G excludes $K_{r,r}$ as a minor. Let d^* be maximum vertex degree of nodes in G. Then algorithm* **MINOR-V**(G, r, Λ) *outputs $\mathcal{B}$ which is $(r/\Lambda, d^{*O(\Lambda)})$ node-decomposition of G.*

Proof. Let R be a connected component of $V - \mathcal{B}$. From Lemma 7, the diameter of R is $O(\Lambda)$. Since d^* is the maximum vertex degree of nodes of G, the number of nodes in R is bounded above by $d^{*O(\Lambda)}$.

To show that $\mathbb{P}(v \in \mathcal{B}) \leq r/\Lambda$, consider a vertex $v \in V$. If $v \notin \mathcal{B}$ in the beginning of an iteration $0 \leq i \leq r-1$, then it will present in exactly one breadth-first search tree, say $\mathcal{T}_j^i$. This vertex v will be chosen in $\mathcal{B}_j^i$ only if it is at level $k\Lambda + L_j^i$ for some integer $k \geq 0$. The probability of this event is at most $1/\Lambda$ since L_j^i is chosen uniformly at random from $\{0, 1 \ldots, \Lambda - 1\}$. By union bound, it follows that the probability that a vertex is chosen to be in $\mathcal{B}$ in any of the r iterations is at most r/Λ. This completes the proof of Lemma 8.

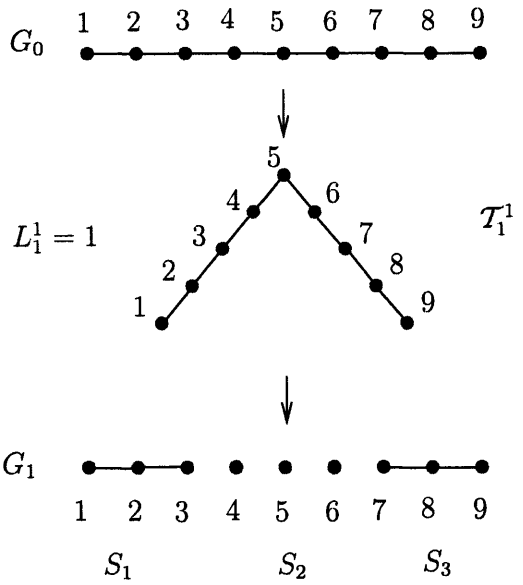

Figure 2-5: The first two iterations in execution of **MINOR-V**$(G, 2, 3)$.

It is known that Planar graph excludes $K_{3,3}$ as a minor. Hence, Lemma 8 implies the following.

Corollary 1 *Given a planar graph G with maximum vertex degree d^*, then the algorithm* **MINOR-V**$(G, 3, \Lambda)$ *produces* $(3/\Lambda, d^{*O(\Lambda)})$ *node-decomposition for any* $\Lambda \geq 1$.

We describe slight modification of **MINOR-V** to obtain algorithm that produces edge-decomposition in Figure 2-6. Note that the only change compared to **MINOR-V** is the selection of edges rather than vertices to create the decomposition.

Lemma 9 *Suppose G excludes $K_{r,r}$ as a minor. Let d^* be maximum vertex degree of nodes in G. Then algorithm* **MINOR-E**(G, r, Λ) *outputs $\mathcal{B}$ which is* $(r/\Lambda, d^{*\Lambda+1})$ *edge-decomposition of G.*

Proof. Let G^* be a graph that is obtained from G by adding center vertex to each edge of G. Then, execution of **MINOR-E**(G, r, Λ) can be viewed as executing **MINOR-V**$(G^*, r, 2\Lambda\text{-}1)$ so that we choose L_j^i's uniformly at random from $\{1, 3, 5, \ldots 2\Lambda - 1\}$. Hence by the same argument as in the proof of Lemma 8, we obtain Lemma 9.

MINOR-E(G, r, Λ)

(0) Input is graph $G = (V, E)$ and $r, \Lambda \in \mathbb{N}$. Initially, $i = 0$, $G_0 = G$, $\mathcal{B} = \emptyset$.

(1) For $i = 0, \ldots, r-1$, do the following.

  (a) Let $S_1^i, \ldots, S_{k_i}^i$ be the connected components of G_i.

  (b) For each $S_j^i, 1 \leq j \leq k_i$, pick an arbitrary node $v_j \in S_j^i$.

  - Create a breadth-first search tree $\mathcal{T}_j^i$ rooted at v_j in S_j^i.
  - Choose a number L_j^i uniformly at random from $\{0, \ldots, \Lambda - 1\}$.
  - Let $\mathcal{B}_j^i$ be the set of edges at level $L_j^i, \Lambda + L_j^i, 2\Lambda + L_j^i, \ldots$ in $\mathcal{T}_j^i$.
  - Update $\mathcal{B} = \mathcal{B} \cup_{j=1}^{k_i} \mathcal{B}_j^i$.

  (c) set $i = i + 1$.

(3) Output $\mathcal{B}$ and graph $G' = (V, E \backslash \mathcal{B})$.

Figure 2-6: Edge-decomposition for minor-excluded graphs

Figure 2-7 explains the algorithm for a line-graph of $n = 9$ nodes, which excludes $K_{2,2}$ as a minor. The example is about a sample run of **MINOR-E**$(G, 2, 3)$ (Figure 2-7 shows the first iteration of the algorithm).

2.5 Approximate MAP

Now, we describe our algorithm to compute MAP approximately. The algorithm uses an edge-decomposition algorithm as a sub-routine. Given G, decompose it into small components $S_1, \ldots, S_K$ by removing edges $\mathcal{B} \subset E$. Then, compute an approximate MAP assignment by computing exact MAP restricted to the components. We first describe our algorithm for any graph G in Figure 2-8; which will be specialized for graphs with polynomial growth and graphs that exclude minor by using the appropriate edge-decomposition schemes described in the previous sections. In Figure 2-8, we use term **DECOMP** for a generic edge-decomposition algorithm. For polynomially growing graph, we use algorithm **POLY-E** and for graph that excludes $K_{r,r}$ as minor for some r, we use the algorithm **MINOR-E**. The approximation guarantee of the output of the algorithm and its computation time depend on the property of

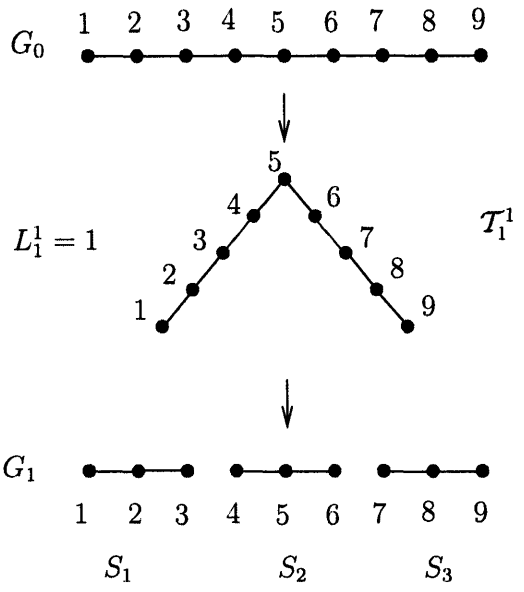

Figure 2-7: The first two iterations in execution of **MINOR-E**$(G, 2, 3)$.

DECOMP.

2.5.1 Analysis of MAP: General G

Here, we analyze performance of **MAP** for any G. Later, we will specialize our analysis for graph with polynomial growth and minor excluded graphs. Let $\psi_{ij}^U = \max_{(x_i,x_j)\in\Sigma^2} \psi_{ij}(x_i, x_j)$, and $\psi_{ij}^L = \min_{(x_i,x_j)\in\Sigma^2} \psi_{ij}(x_i, x_j)$.

Theorem 1 *Given an MRF G described by (2.1), the* **MAP** *algorithm produces output $\widehat{\mathbf{x}}^*$ such that:*

$$\mathcal{H}(\mathbf{x}^*) - \sum_{(i,j)\in\mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right) \leq \mathcal{H}(\widehat{\mathbf{x}}^*) \leq \mathcal{H}(\mathbf{x}^*).$$

The algorithm takes $O\left(|E|K|\Sigma|^{|S^*|}\right) + T_{\mathbf{DECOMP}}$ time to produce this estimate, where $|S^*| = \max_{j=1}^K |S_j|$ with **DECOMP** *producing a decomposition of G into $S_1, \ldots, S_K$ in time $T_{\mathbf{DECOMP}}$.*

MAP(G)

(0) Input is MRF $G = (V, E)$ with $\phi_i(\cdot), i \in V$, $\psi_{ij}(\cdot,\cdot), (i,j) \in E$.

(1) Use **DECOMP**(G) to obtain $\mathcal{B} \subset E$ such that

  (a) $G' = (V, E \backslash \mathcal{B})$ is made of connected components $S_1, \ldots, S_K$.

(2) For each connected component $S_j, 1 \leq j \leq K$, do the following:

  (a) Through dynamic programming (or exhaustive computation) find exact MAP $\mathbf{x}^{*,j}$ for component S_j, where $\mathbf{x}^{*,j} = (x_i^{*,j})_{i \in S_j}$.

(3) Produce output $\widehat{\mathbf{x}^*}$, which is obtained by assigning values to nodes using $\mathbf{x}^{*,j}, 1 \leq j \leq K$.

Figure 2-8: Algorithm for MAP computation

Proof. By definition of MAP $\mathbf{x}^*$, we have $\mathcal{H}(\widehat{\mathbf{x}^*}) \leq \mathcal{H}(\mathbf{x}^*)$. Now, consider the following.

$$
\begin{aligned}
\mathcal{H}(\mathbf{x}^*) &= \max_{\mathbf{x} \in \Sigma^n} \left[\sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E} \psi_{ij}(x_i, x_j) \right] \\
&= \max_{\mathbf{x} \in \Sigma^n} \left[\sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E \backslash \mathcal{B}} \psi_{ij}(x_i, x_j) + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}(x_i, x_j) \right] \\
&\overset{(a)}{\leq} \max_{\mathbf{x} \in \Sigma^n} \left[\sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E \backslash \mathcal{B}} \psi_{ij}(x_i, x_j) + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U \right] \\
&\overset{(b)}{=} \sum_{j=1}^{K} \left[\max_{\mathbf{x}^j \in \Sigma^{|S_j|}} \mathcal{H}(\mathbf{x}^j) \right] + \left[\sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U \right] \\
&\overset{(c)}{=} \sum_{j=1}^{K} \mathcal{H}(\mathbf{x}^{*,j}) + \left[\sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U \right] \\
&\overset{(d)}{\leq} \mathcal{H}(\widehat{\mathbf{x}^*}) + \left[\sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U - \psi_{ij}^L \right] .
\end{aligned}
\tag{2.15}
$$

We justify (a)-(d) as follows: (a) holds because for each edge $(i,j) \in \mathcal{B}$, we have replaced its effect by maximal value ψ_{ij}^U; (b) holds because by placing constant value ψ_{ij}^U over $(i,j) \in \mathcal{B}$, the maximization over G decomposes into maximization over the connected components of $G' = (V, E \backslash \mathcal{B})$; (c) holds by definition of $\mathbf{x}^{*,j}$ and (d) holds because when we obtain global assignment $\widehat{\mathbf{x}}^*$ from $\mathbf{x}^{*,j}, 1 \leq j \leq K$ and compute its global value, the additional terms get added for each $(i,j) \in \mathcal{B}$ which add at least ψ_{ij}^L amount.

For running time analysis, note that **MAP** performs two main tasks: (i) Decomposing G using **DECOMP** algorithm, which by definition take $T_{\mathbf{DECOMP}}$ time. (ii) Computing $\mathbf{x}^{*,j}$ for each component S_j through exhaustive computation, which takes $O(|E||\Sigma|^{|S_j|})$ time, and producing $\mathbf{x}^*$ takes addition $|V|$ operations at the most. Since there are K components in total with max-size of component being $|S^*|$, we obtain that the running time for task (ii) is $O(|E|K|\Sigma|^{|S^*|})$. Putting (i) and (ii) together, we obtain the desired bound. This completes the proof of Theorem 1.

2.5.2 Some preliminaries

This section presents some results about the property of MAP solution that will be useful in obtaining tight approximation guarantees later. First, consider the following.

Lemma 10 *If G has maximum vertex degree d^*, then*

$$\begin{aligned} \mathcal{H}(\mathbf{x}^*) &\geq \frac{1}{d^*+1}\left[\sum_{(i,j)\in E} \psi_{ij}^U\right] \\ &\geq \frac{1}{d^*+1}\left[\sum_{(i,j)\in E} \psi_{ij}^U - \psi_{ij}^L\right]. \end{aligned} \tag{2.16}$$

Proof. Assign weight $w_{ij} = \psi_{ij}^U$ to an edge $(i,j) \in E$. Since graph G has maximum vertex degree d^*, by Vizing's theorem there exists an edge-coloring of the graph using at most d^*+1 colors. Edges with the same color form a matching of the G. A standard application of Pigeon-hole's principle implies that there is a color with weight at least

$\frac{1}{d^*+1}(\sum_{(i,j)\in E} w_{ij})$. Let $M \subset E$ denote these set of edges. Then

$$\sum_{(i,j)\in M} \psi_{ij}^U \geq \frac{1}{d^*+1}\left(\sum_{(i,j)\in E} \psi_{ij}^U\right).$$

Now, consider an assignment $\mathbf{x}^M$ as follows: for each $(i,j) \in M$ set $(x_i^M, x_j^M) = \arg\max_{(x,x')\in\Sigma^2} \psi_{ij}(x,x')$; for remaining $i \in V$, set x_i^M to some value in Σ arbitrarily. Note that for above assignment to be possible, we have used matching property of M. Therefore, we have

$$\begin{aligned}
\mathcal{H}(\mathbf{x}^M) &= \sum_{i\in V} \phi_i(x_i^M) + \sum_{(i,j)\in E} \psi_{ij}(x_i^M, x_j^M) \\
&= \sum_{i\in V} \phi_i(x_i^M) + \sum_{(i,j)\in E\setminus M} \psi_{ij}(x_i^M, x_j^M) + \sum_{(i,j)\in M} \psi_{ij}(x_i^M, x_j^M) \\
&\overset{(a)}{\geq} \sum_{(i,j)\in M} \psi_{ij}(x_i^M, x_j^M) \\
&= \sum_{(i,j)\in M} \psi_{ij}^U \\
&\geq \frac{1}{d^*+1}\left[\sum_{(i,j)\in E} \psi_{ij}^U\right]. \qquad (2.17)
\end{aligned}$$

Here (a) follows because ψ_{ij}, ϕ_i are non-negative valued functions. Since $\mathcal{H}(\mathbf{x}^*) \geq \mathcal{H}(\mathbf{x}^M)$ and $\psi_{ij}^L \geq 0$ for all $(i,j) \in E$, we obtain the Lemma 10.

Lemma 11 *If G has maximum vertex degree d^* and the* **DECOMP***(G) produces $\mathcal{B}$ that is (ε, Δ) edge-decomposition, then*

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon(d^*+1)\mathcal{H}(\mathbf{x}^*),$$

where expectation is w.r.t. the randomness in $\mathcal{B}$. Further, **MAP** *takes time $O(nd^*|\Sigma|^\Delta) + T_{\textbf{DECOMP}}$.*

Proof. From Theorem 1, Lemma 10 and definition of (ε, Δ) edge-decomposition, we have the following.

$$\begin{aligned}
\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] &\leq \mathbb{E}\left[\sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L)\right] \\
&= \sum_{(i,j)\in E} \mathbb{P}((i,j)\in\mathcal{B})(\psi_{ij}^U - \psi_{ij}^L) \\
&\leq \varepsilon\left[\sum_{(i,j)\in E} (\psi_{ij}^U - \psi_{ij}^L)\right] \\
&\leq \varepsilon(d^*+1)\mathcal{H}(\mathbf{x}^*). \qquad (2.18)
\end{aligned}$$

Now to estimate the running time, note that under (ε, Δ) edge-decomposition $\mathcal{B}$, with probability 1 the $G' = (V, E\backslash\mathcal{B})$ is divided into connected components with at most Δ nodes. Therefore, the running time bound of Theorem 1 implies the desired result.

2.5.3 Analysis of MAP: polynomially growing G

Here we interpret result obtained in Theorem 1 and Lemma 11, for polynomially growing G and uses decomposition scheme **POLY-E**.

Theorem 2 *Let MRF graph G of n nodes with growth rate ρ and corresponding constant C be given. Consider any constant $\varepsilon \in (0,1)$, and define $\varphi = \frac{\varepsilon}{2^\rho C}$. Then* **MAP** *using* **POLY-E**$(\varphi, K(\varphi,\rho,C))$ *produces an assignment $\widehat{\mathbf{x}^*}$ such that*

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon\mathcal{H}(\mathbf{x}^*).$$

Further, the algorithm takes $O(n)$ amount of time.

Proof. First, **POLY-E**$(\varphi, K(\varphi,\rho))$ produces $(\varphi, O(1))$ edge-decomposition from Lemma 6.

Note that the maximum vertex degree d^* of G is less than $2^\rho C$ by the definition of polynomially growing graph. Therefore, by Lemma 11, the output produced by

the **MAP** algorithm is such that

$$\begin{aligned}\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] &\leq (d^*+1)\frac{\varepsilon}{2^\rho C}\mathcal{H}(\mathbf{x}^*) \\ &\leq \varepsilon\mathcal{H}(\mathbf{x}^*), \end{aligned} \tag{2.19}$$

Further, the running time of the algorithm is $O(nd^*|\Sigma|^{K(\varphi,\rho)}) + T_{\mathbf{DECOMP}} = O(n)$. This completes the proof of Theorem 2.

2.5.4 Analysis of MAP: Minor-excluded G

When G is a minor excluded graph with bounded vertex degree, we use **MINOR-E** for the edge-decomposition. We apply Theorem 1 and Lemma 11 to obtain the following result.

Theorem 3 *Let MRF graph G of n nodes exclude $K_{r,r}$ as its minor. Let d^* be the maximum vertex degree in G. Given constant $\varepsilon > 0$, use* **MAP** *algorithm with* **MINOR-E**(G, r, Λ) *where $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$. Then,* **MAP** *produces an assignment $\widehat{\mathbf{x}^*}$ such that*

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon\mathcal{H}(\mathbf{x}^*).$$

The algorithm takes $O(n)$ time.

Proof. From Lemma 9 about the **MINOR-E** algorithm, we have that with choice of $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$, the algorithm produces $(\frac{\varepsilon}{d^*+1}, \Delta)$ edge-decomposition where $\Delta = d^{*O(\Lambda)}$. Since its an $(\frac{\varepsilon}{d^*+1}, \Delta)$ edge-decomposition, from Lemma 11 it follows that

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon\mathcal{H}(\mathbf{x}^*).$$

Now, by Lemma 11 the algorithm running time is $O(nd^*|\Sigma|^{\Delta}) + T_{\mathbf{DECOMP}}$. As discussed earlier in Lemma 9, the algorithm **MINOR-E** takes $O(r|E|) = O(nrd^*)$ operations. That is, $T_{\mathbf{DECOMP}} = O(n)$. Now, $\Delta = d^{*O(\Lambda)}$ and $\Lambda \leq r(d^*+1)/\varepsilon + 1$. Therefore, the first term of the computation time bound $O(nd^*|\Sigma|^{\Delta}) = O(n)$. Hence the running time of **MAP** is $O(n)$. This completes the proof of Theorem 3.

2.5.5 Sequential MAP: tight approximation for polynomially growing G

The main drawback of the algorithm **MAP** is that it ignores the effects of the boundary edge potentials, instead of utilizing the boundary potentials. In this section, we provide a modification of **MAP** for the case when G is a polynomially growing graph, which utilizes the boundary information. The main idea of the modified algorithm **SEQ-MAP** is as follows. In **SEQ-MAP**, we will update our solution $\widehat{x^*}$ of MAP assignment over time. That is, instead of performing the whole edge-decomposition **POLY-E** of the graph and computing the MAP for each components at the same time, we will take balls of radius Q one by one, where Q is taken independently according to the distribution $\mathbf{Q}$ as in **POLY-E**. At each time, compute a MAP assignment within the chosen ball *while fixing all the other current assignment value of* $\widehat{x^*}$ *for the vertices outside the ball.* Then change the value of $\widehat{x^*}$ for the vertices within the chosen ball by the computed MAP. Do this process until all the vertices of G will be chosen as a center vertex of a ball at least once. By using a coupling argument, in Theorem 4 we will provide a similar error bound for **SEQ-MAP** as that of **MAP**.

Precise description of the algorithm is in Figure 2-9. Given a constant $\varepsilon \in (0,1)$ and a polynomially growing graph G with growth rate ρ and the corresponding constant C, we set $\varphi = \frac{\varepsilon}{5 \cdot 2^{\rho} C}$, and

$$K = K(\varphi, \rho, C) = \frac{8\rho}{\varphi} \log\left(\frac{8\rho}{\varphi}\right) + \frac{4}{\varphi} \log C + \frac{4}{\varphi} \log \frac{1}{\varphi} + 2,$$

and define a random variable $\mathbf{Q}'$ over $\{1, 2, 3, \ldots, K\}$ as

$$\mathbb{P}[\mathbf{Q} = i] = \begin{cases} \varphi(1-\varphi)^{i-1} & \text{if } 1 \leq i < K \\ (1-\varphi)^{K-1} & \text{if } i = K \end{cases}.$$

Figure 2-10 describes a difference between **POLY-E** and **SEQ-MAP**.

Theorem 4 *Let an MRF defined on a graph G of n nodes with growth rate ρ and the corresponding constant C be given. Consider any constant $\varepsilon \in (0,1)$. Then*

SEQ-MAP(ε, K)

(0) Input is an MRF $G = (V, E)$ with $\phi_i(\cdot), i \in V$, $\psi_{ij}(\cdot,\cdot), (i,j) \in E$.

(1) Set $\widehat{\mathbf{x}^*}$ with a random assignment.

(2) Let $\mathcal{W} \leftarrow V$.

(3) Do the following procedure while $\mathcal{W} \neq \emptyset$:

  (a) Choose an element $u \in V$ uniformly at random.

  (b) Draw a random number Q according to the distribution $\mathbf{Q}$.

  (c) Let $\mathcal{R} \leftarrow \{w \in V | \mathbf{d_G}(u, w) < Q\}$.

  (d) Through dynamic programming (or exhaustive computation) find an exact MAP $\mathbf{x}^{*,\mathcal{R}}$ for $\mathcal{R}$ while fixing all the other assignment of $\widehat{\mathbf{x}^*}$ value outside $\mathcal{R}$.

  (e) Change values of $\widehat{\mathbf{x}^*}$ for $\mathcal{R}$ by $\mathbf{x}^{*,\mathcal{R}}$.

  (f) Let $\mathcal{W} \leftarrow \mathcal{W} - \{u\}$.

(4) Output $\widehat{\mathbf{x}^*}$.

Figure 2-9: A Sequential Algorithm for MAP computation

SEQ-MAP *produces an assignment* $\widehat{\mathbf{x}^*}$ *such that*

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon \mathcal{H}(\mathbf{x}^*).$$

The algorithm takes $O(n \log n)$ *time in expectation.*

Proof. From the Coupon collector's problem [9], it immediately follows that with high probability as well as in expectation,the number of iterations of **SEQ-MAP** is $O(n \log n)$. The running time of one iteration of **SEQ-MAP** is $O(1)$ since ρ, C, and ε are constants. Hence we obtain that the expected running time of **SEQ-MAP** is $O(n \log n)$.

Now we show the error bound of **SEQ-MAP**. First, we prove the following lemma that will be used in the proof of the error bound.

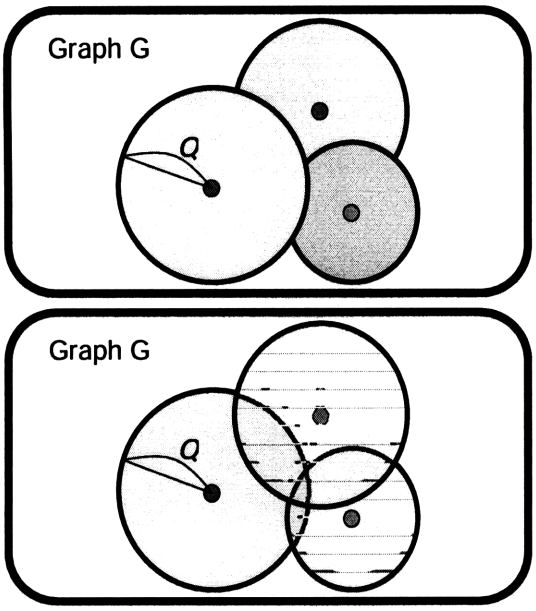

Figure 2-10: Comparison of **POLY-E** versus **SEQ-MAP**.

Lemma 12 *Let* $\mathbf{X}_1$ *and* $\mathbf{X}_2$ *be two MRFs defined on a same graph* $G' = (V', E')$. *Assume that the edge potential functions for* $\mathbf{X}_1$ *and* $\mathbf{X}_2$ *are identically given by* $\{\psi_{ij}(\cdot,\cdot)\}$, *and let* $\{\phi_i^1(\cdot)\}$ *and* $\{\phi_i^2(\cdot)\}$ *be the vertex potentials for* $\mathbf{X}_1$ *and* $\mathbf{X}_2$ *respectively. For each vertex* $i \in V'$, *let*

$$\phi_i^D = \max_{x_i \in \Sigma} \left| \phi_i^1(x_i) - \phi_i^2(x_i) \right|$$

be the maximum difference of ϕ_i. *For* $\ell = 1, 2$, *let*

$$\mathcal{H}_\ell(\mathbf{x}) = \sum_{i \in V'} \phi_i^\ell(x_i) + \sum_{(i,j) \in E'} \psi_{ij}(x_i, x_j),$$

and let $\mathbf{x}_\ell$ *be a MAP assignment of* $\mathbf{X}_\ell$. *Then we have*

$$|\mathcal{H}_1(\mathbf{x}_1) - \mathcal{H}_1(\mathbf{x}_2)| \leq 2 \sum_{i \in V'} \phi_i^D.$$

Proof. Let

$$\gamma = \max_{\mathbf{x} \in \Sigma^{|V'|}} |\mathcal{H}_1(\mathbf{x}) - \mathcal{H}_2(\mathbf{x})| .$$

Then we have

$$\begin{aligned}
\gamma &= \max_{\mathbf{x} \in \Sigma^{|V'|}} |\mathcal{H}_1(\mathbf{x}) - \mathcal{H}_2(\mathbf{x})| \\
&= \max_{\mathbf{x} \in \Sigma^{|V'|}} \left| \left[\sum_{i \in V'} \phi_i^1(x_i) + \sum_{(i,j) \in E'} \psi_{ij}(x_i, x_j) \right] - \left[\sum_{i \in V'} \phi_i^2(x_i) + \sum_{(i,j) \in E'} \psi_{ij}(x_i, x_j) \right] \right| \\
&= \max_{\mathbf{x} \in \Sigma^{|V'|}} \left| \sum_{i \in V'} \phi_i^1(x_i) - \sum_{i \in V'} \phi_i^2(x_i) \right| \\
&\leq \max_{\mathbf{x} \in \Sigma^{|V'|}} \sum_{i \in V'} \left| \phi_i^1(x_i) - \phi_i^2(x_i) \right| \\
&\leq \sum_{i \in V'} \max_{x_i \in \Sigma} \left| \phi_i^1(x_i) - \phi_i^2(x_i) \right| \\
&= \sum_{i \in V'} \phi_i^D . \qquad (2.20)
\end{aligned}$$

Now, from the definition of γ,

$$|\mathcal{H}_2(\mathbf{x}_2) - \mathcal{H}_1(\mathbf{x}_2)| \leq \gamma. \qquad (2.21)$$

From the fact that $\mathbf{x}_1$ is a MAP assignment of $\mathbf{X}_1$,

$$\mathcal{H}_1(\mathbf{x}_2) \leq \mathcal{H}_1(\mathbf{x}_1).$$

Hence we obtain that

$$\mathcal{H}_2(\mathbf{x}_2) \leq \mathcal{H}_1(\mathbf{x}_1) + \gamma. \qquad (2.22)$$

Similarly, from $|\mathcal{H}_1(\mathbf{x}_1) - \mathcal{H}_2(\mathbf{x}_1)| \leq \gamma$, and $\mathcal{H}_2(\mathbf{x}_1) \leq \mathcal{H}_2(\mathbf{x}_2)$, we obtain

$$\mathcal{H}_1(\mathbf{x}_1) \leq \mathcal{H}_2(\mathbf{x}_2) + \gamma. \qquad (2.23)$$

From (2.22) and (2.23), we have

$$|\mathcal{H}_1(\mathbf{x}_1) - \mathcal{H}_2(\mathbf{x}_2)| \leq \gamma. \tag{2.24}$$

Now from (2.20), (2.21) and (2.24),

$$|\mathcal{H}_1(\mathbf{x}_1) - \mathcal{H}_1(\mathbf{x}_2)| \leq |\mathcal{H}_1(\mathbf{x}_1) - \mathcal{H}_2(\mathbf{x}_2)| + |\mathcal{H}_1(\mathbf{x}_2) - \mathcal{H}_2(\mathbf{x}_2)| \leq 2\gamma \leq 2\sum_{i \in V'} \phi_i^D,$$

which proves Lemma 12.

To show the error bound of **SEQ-MAP**, we will imaginarily construct an edge decomposition that is similar to the output of **POLY-E**. The main idea is to consider the ball selection procedure (3) of **SEQ-MAP** in the *reverse order*. Formally, imagine that the procedure (3) of **SEQ-MAP** is done with an iteration parameter $t \in \mathbb{Z}_+$. Then for each vertex $w \in V$, we assign the largest iteration number t such that the chosen ball $\mathcal{R}$ at the iteration t contains w. That is,

$$T(w) = \max\{t \in \mathbb{Z}_+ | \textbf{ SEQ-MAP} \text{ chooses } w \text{ as a member of } \mathcal{R} \text{ at iteration } t.\}.$$

This is well defined, since we run the algorithm till each node becomes center of a ball at least once. Now define the *imaginary boundary set* of **SEQ-MAP** be

$$\mathcal{B} = \{(u, w) \in E | T(u) \neq T(w)\}.$$

Notice that all the vertices v in a connected component of $G' = (V, E - \mathcal{B})$ has same $T(v)$ value. Let $S_1, \ldots, S_K$ be the connected components of $G' = (V, E - \mathcal{B})$ so that vertices in S_1 has the largest $T(\cdot)$ value, vertices in S_2 has the second largest $T(\cdot)$ value, and so on. Let $T(S_k)$ be the corresponding iteration number of the vertices in S_k.

Now the above procedure that generates $\mathcal{B}$ is identical to the procedure **POLY-E** that generates an edge-decomposition of G, except the fact that the above procedure allows center vertices to be chosen from any vertices, whereas **POLY-E** allows only

the vertices in $\mathcal{W}$ to be chosen as center vertices. This fact does not affect the proof of Lemma 4 and Lemma 6. Hence from Lemma 4 and Lemma 6, we obtain that the random set $\mathcal{B}$ is a $(2\varepsilon, CK^\rho)$ edge-decomposition of G.

Recall that the maximum vertex degree d^* of G is less than $2^\rho C$ by the definition of polynomially growing graph. This fact together with Lemma 10, the following Lemma 13, and the fact that $\mathcal{B}$ is a $(2\varepsilon, CK^\rho)$ edge-decomposition of G, proves the error bound of **SEQ-MAP**.

Lemma 13 *Given an MRF* $\mathbf{X}$ *defined on graph* G *described by (2.1), the* **SEQ-MAP** *produces an output* $\widehat{\mathbf{x}}^*$ *such that:*

$$\left|\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}}^*)\right| \leq 5 \sum_{(i,j)\in\mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right),$$

where $\mathcal{B}$ *is the* imaginary boundary set *of* **SEQ-MAP**.

Proof. Let $\mathbf{x}^*$ be a MAP assignment of the MRF $\mathbf{X}$ defined on G. Given an assignment $\mathbf{x} \in \Sigma^{|V|}$ defined on a graph $G = (V, E)$ and a subgraph $S = (W, E')$ of G, let an assignment $\mathbf{x}' \in \Sigma^{|W|}$ be called *a restriction of* $\mathbf{x}$ *to* S if $\mathbf{x}'(v) = \mathbf{x}(v)$ for all $v \in W$. Let $\mathbf{x}_k^*$ be the restriction of $\mathbf{x}^*$ to the component S_k.

Let $\mathbf{X}_k$ be the restriction of the MRF $\mathbf{X}$ to $G_k = (S_k, E_k)$, where $E_k = \{(u, w) \in E | u, w \in S_k\}$. Then, since $\mathbf{x}^*$ is a MAP assignment of $\mathbf{X}$, $\mathbf{x}_k^*$ is a MAP assignment of $\mathbf{X}_k$.

Let $\widehat{\mathbf{x}}^*$ be the output of **SEQ-MAP**, and let $\widehat{\mathbf{x}}_k^*$ be the restriction of $\widehat{\mathbf{x}}^*$ to the component S_k. From the definition of S_k, note that $\hat{\mathbf{x}}_k^*$ is a MAP assignment of S_k under a condition of fixed assignment for all the vertices outside S_k. Let

$$\mathcal{B}_k = \{(u, w) \in \mathcal{B} | u \in S_k\}$$

be the edge boundary of S_k, and let

$$\partial S_k = \{u \in S_k | (u, w) \in \mathcal{B} \text{ for some } w \in V\}$$

be the set of boundary vertices of S_k. Then the condition that all the vertices outside S_k are fixed to an assignment is equivalent to changing the vertex potential functions on ∂S_k due to the fixed end point edge potentials of $\mathcal{B}_k$, and removing the edge potentials of $\mathcal{B}_k$. Let $\mathbf{X}_k$ be the MRF obtained from $\mathbf{X}$ by restricting to G_k, then changing the boundary vertex potentials due to the fixed assignment outside S_k at the iteration $T(S_k)$ of **SEQ-MAP**. Let $\{\phi_i^k(\cdot)\}_{i\in S_k}$ be the vertex potential functions of $\mathbf{X}_k$. For $i \in S_k$, let

$$\phi_i^{k,D} = \max_{x_i \in \Sigma} \left|\phi_i(x_i) - \phi_i^k(x_i)\right|$$

Then, from the potential change of ϕ_i^k, we have

$$\sum_i \phi_i^D \leq \sum_{(u,w)\in\mathcal{B}_j} (\psi_{ij}^U - \psi_{ij}^L). \tag{2.25}$$

For $\mathbf{x}_k \in \Sigma^{|S_k|}$, define

$$\mathcal{H}_k(\mathbf{x}_k) = \sum_{i\in} \phi_i^k(x_i) + \sum_{(i,j)\in E_k} \psi_{ij}(x_i, x_j).$$

Now, we have

$$\begin{aligned}
|\mathcal{H}(\hat{\mathbf{x}}^*) - \mathcal{H}(\mathbf{x}^*)| &\leq \sum_{k=1}^{K} |\mathcal{H}_k(\widehat{\mathbf{x}}_k^*) - \mathcal{H}_k(\mathbf{x}_k^*)| + \sum_{(i,j)\in\mathcal{B}} |\psi_{ij}(\widehat{x}_i^*, \widehat{x}_j^*) - \psi_{ij}(x_i^*, x_j^*)| \\
&\leq \sum_{k=1}^{K} |\mathcal{H}_k(\widehat{\mathbf{x}}_k^*) - \mathcal{H}_k(\mathbf{x}_k^*)| + \sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L) \\
&\overset{(a)}{\leq} \sum_{k=1}^{K} 2\left[\sum_{i\in S_k} \phi_i^{k,D}\right] + \sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L) \\
&\overset{(b)}{\leq} \sum_{j} 2\left[\sum_{(u,w)\in\mathcal{B}_j} (\psi_{ij}^U - \psi_{ij}^L)\right] + \sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L) \\
&\overset{(c)}{=} 5 \sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L).
\end{aligned} \tag{2.26}$$

Here, (a) follows from Lemma 12, (b) follows from (2.25), and (c) follows from the fact that each edge in $\mathcal{B}$ belongs to exactly two $\mathcal{B}_k$. This completes the proof of

Lemma 13.

2.6 Approximate $\log Z$

In this section, first we describe algorithm for approximate computation of $\log Z$ for any graph G. The algorithm uses an edge-decomposition algorithm as a sub-routine. Our algorithm provides provable upper and lower bound on $\log Z$ for any graph G. In order to obtain tight approximation guarantee, we will use specific graph structures as in the polynomially growing graph and the minor-excluded graph.

2.6.1 Algorithm

As in the **MAP**, we use the term **DECOMP** for a generic edge-decomposition algorithm. The approximation guarantee of the output of the algorithm and its computation time depend on the property of **DECOMP**. For polynomially growing graph, we use algorithm **POLY-E** and for graph that excludes $K_{r,r}$ as minor for some r, we use algorithm **MINOR-E**. Figure 2-11 describes the algorithm for computation of $\log Z$.

In words, **Log Partition**(G) produces upper and lower bound on $\log Z$ of MRF G as follows: decompose graph G into (small) components $S_1, \ldots, S_K$ by removing (few) edges $\mathcal{B} \subset E$ using **DECOMP**(G). Compute exact log-partition function in each of the components. To produce bounds $\log \hat{Z}_{\mathrm{LB}}, \log \hat{Z}_{\mathrm{UB}}$ take the summation of thus computed component-wise log-partition function along with minimal and maximal effect of edges from $\mathcal{B}$.

2.6.2 Analysis of Log Partition: General G

Here, we analyze performance of **Log Partition** for any G. Later, we will use property of the specific graph structures to obtain sharper approximation guarantees.

Theorem 5 *Given a pair-wise MRF G, the* **Log Partition** *produces* $\log \hat{Z}_{LB}, \log \hat{Z}_{UB}$

Log Partition(G)

(1) Use **DECOMP**(G) to obtain $\mathcal{B} \subset E$ such that

  (a) $G' = (V, E \backslash \mathcal{B})$ is made of connected components $S_1, \ldots, S_K$.

(2) For each connected component $S_j, 1 \leq j \leq K$, do the following:

  (a) Compute partition function Z_j restricted to S_j by dynamic programming (or exhaustive computation).

(3) Let $\psi_{ij}^L = \min_{(x,x') \in \Sigma^2} \psi_{ij}(x, x')$, $\psi_{ij}^U = \max_{(x,x') \in \Sigma^2} \psi_{ij}(x, x')$. Then

$$\log \hat{Z}_{\text{LB}} = \sum_{j=1}^{K} \log Z_j + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^L;$$

$$\log \hat{Z}_{\text{UB}} = \sum_{j=1}^{K} \log Z_j + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U.$$

(4) Output: lower bound $\log \hat{Z}_{\text{LB}}$ and upper bound $\log \hat{Z}_{\text{UB}}$.

Figure 2-11: Algorithm for log-partition function computation

such that

$$\log \hat{Z}_{LB} \leq \log Z \leq \log \hat{Z}_{UB},$$

$$\log \hat{Z}_{UB} - \log \hat{Z}_{LB} = \sum_{(i,j) \in \mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L \right).$$

It takes $O\left(|E| K |\Sigma|^{|S^*|}\right) + T_{\textbf{DECOMP}}$ *time to produce this estimate, where* $|S^*| = \max_{j=1}^{K} |S_j|$ *with* **DECOMP** *producing decomposition of* G *into* $S_1, \ldots, S_K$ *in time* $T_{\textbf{DECOMP}}$.

Proof. First, we prove properties of $\log \hat{Z}_{\text{LB}}, \log \hat{Z}_{\text{UB}}$ as follows:

$$\begin{aligned} \log \hat{Z}_{\text{LB}} &\overset{(o)}{=} \sum_{j=1}^{K} \log Z_j + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^L \\ &\overset{(a)}{=} \log \left[\sum_{\mathbf{x} \in \Sigma^n} \exp \left(\sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E \backslash \mathcal{B}} \psi_{ij}(x_i, x_j) + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^L \right) \right] \end{aligned}$$

$$
\begin{aligned}
&\overset{(b)}{\leq} \log\left[\sum_{\mathbf{x}\in\Sigma^n} \exp\left(\sum_{i\in V}\phi_i(x_i) + \sum_{(i,j)\in E\setminus\mathcal{B}}\psi_{ij}(x_i,x_j) + \sum_{(i,j)\in\mathcal{B}}\psi_{ij}(x_i,x_j)\right)\right] \\
&= \log Z \\
&\overset{(c)}{\leq} \log\left[\sum_{\mathbf{x}\in\Sigma^n} \exp\left(\sum_{i\in V}\phi_i(x_i) + \sum_{(i,j)\in E\setminus\mathcal{B}}\psi_{ij}(x_i,x_j) + \sum_{(i,j)\in\mathcal{B}}\psi^U_{ij}\right)\right] \\
&\overset{(d)}{=} \sum_{j=1}^{K}\log Z_j + \sum_{(i,j)\in\mathcal{B}}\psi^U_{ij} \\
&= \log\hat{Z}_{\mathrm{UB}}.
\end{aligned}
$$

We justify (a)-(d) as follows: (a) holds because by removal of edges $\mathcal{B}$, the G decomposes into disjoint connected components $S_1,\ldots,S_K$; (b) holds because of the definition of ψ^L_{ij}; (c) holds by definition ψ^U_{ij} and (d) holds for a similar reason as (a). The claim about difference $\log\hat{Z}_{\mathrm{UB}} - \log\hat{Z}_{\mathrm{LB}}$ in the statement of Theorem 5 follows directly from definitions (i.e. subtract RHS (o) from (d)). This completes proof of claimed relation between bounds $\log\hat{Z}_{\mathrm{LB}}, \log\hat{Z}_{\mathrm{UB}}$.

For running time analysis, note that **Log Partition** performs two main tasks: (i) Decomposing G using **DECOMP** algorithm, which by definition take $T_{\mathbf{DECOMP}}$ time. (ii) Computing Z_j for each component S_j through exhaustive computation, which takes $O(|E||\Sigma|^{|S_j|})$ time, and producing $\log\hat{Z}_{\mathrm{LB}}, \log\hat{Z}_{\mathrm{UB}}$ takes addition $|V|$ operations at the most. Since there are K components in total with max-size of component being $|S^*|$ we obtain that running time for task (ii) is $O(|E|K|\Sigma|^{|S^*|})$. Putting (i) and (ii) together, we obtain the desired bound. This completes the proof of Theorem 5.

2.6.3 Some preliminaries

Before stating precise approximation bound of **Log Partition** algorithm for polynomially growing graphs and graphs that exclude minors, we state two useful Lemmas about $\log Z$ for any graph.

Lemma 14 *If G has maximum vertex degree d^* then,*

$$\log Z \geq \frac{1}{d^*+1}\left[\sum_{(i,j)\in E} \psi_{ij}^U - \psi_{ij}^L\right].$$

Proof. Assign weight $w_{ij} = \psi_{ij}^U - \psi_{ij}^L$ to an edge $(i,j) \in E$. Since graph G has maximum vertex degree d^*, by Vizing's theorem there exists an edge-coloring of the graph using at most d^*+1 colors. Edges with the same color form a matching of the G. A standard application of Pigeon-hole's principle implies that there is a color with weight at least $\frac{1}{d^*+1}(\sum_{(i,j)\in E} w_{ij})$. Let $M \subset E$ denote these set of edges. That is,

$$\sum_{(i,j)\in M} (\psi_{ij}^U - \psi_{ij}^L) \geq \frac{1}{d^*+1}\left(\sum_{(i,j)\in E} (\psi_{ij}^U - \psi_{ij}^L)\right).$$

Now, consider a $Q \subset \Sigma^n$ of size $2^{|M|}$ created as follows. For $(i,j) \in M$ let $(x_i^U, x_j^U) \in \arg\max_{(x,x')\in\Sigma^2} \psi_{ij}(x,x')$. For each $i \in V$, choose $x_i^L \in \Sigma$ arbitrarily. Then,

$$\begin{aligned} Q \;=\; \{\mathbf{x} \in \Sigma^n : \forall\ (i,j) \in M,\ (x_i, x_j) = (x_i^U, x_j^U) \text{ or} \\ (x_i^L, x_j^L); \text{for all other } i \in V,\ x_i = x_i^L\}. \end{aligned}$$

Note that we have used the fact that M is a matching for Q to be well-defined.

By definition ϕ_i, ψ_{ij} are non-negative function (hence, their exponents are at least 1). Using this property, we have the following:

$$\begin{aligned} Z \;&\geq\; \left[\sum_{\mathbf{x}\in Q} \exp\left(\sum_{i\in V} \phi_i(x_i) + \sum_{(i,j)\in E} \psi_{ij}(x_i,x_j)\right)\right] \\ &\overset{(o)}{\geq}\; \left[\sum_{\mathbf{x}\in Q} \exp\left(\sum_{(i,j)\in M} \psi_{ij}(x_i,x_j)\right)\right] \end{aligned}$$

$$
\begin{aligned}
&\overset{(a)}{\geq} \quad 2^{|M|} \prod_{(i,j)\in M} \frac{\exp(\psi_{ij}^L) + \exp(\psi_{ij}^U)}{2} \\
&\overset{(b)}{=} \quad \prod_{(i,j)\in M} (1 + \exp(\psi_{ij}^U - \psi_{ij}^L)) \exp(\psi_{ij}^L) \\
&\overset{(c)}{\geq} \quad \prod_{(i,j)\in M} \exp(\psi_{ij}^U - \psi_{ij}^L) \\
&= \quad \exp\left(\sum_{(i,j)\in M} \psi_{ij}^U - \psi_{ij}^L \right). \qquad (2.27)
\end{aligned}
$$

Justification of (o)-(c): (o) follows since ψ_{ij}, ϕ_i are non-negative functions. (a) consider the following probabilistic experiment: assign (x_i, x_j) for each $(i,j) \in M$ equal to (x_i^U, x_j^U) or (x_i^L, x_j^L) with probability $1/2$ each. Under this experiment, the expected value of the $\exp(\sum_{(i,j)\in M} \psi_{ij}(x_i, x_j))$, which is $\prod_{(i,j)\in M} \frac{\exp(\psi_{ij}(x_i^L, x_j^L)) + \exp(\psi_{ij}(x_i^U, x_j^U))}{2}$, is equal to $2^{-|M|}[\sum_{\mathbf{x}\in Q} \exp(\sum_{(i,j)\in M} \psi_{ij}(x_i, x_j))]$. Now, use the fact that $\psi_{ij}(x_i^L, x_j^L) \geq \psi_{ij}^L$. (b) follows from simple algebra and (c) follows by using non-negativity of function ψ_{ij}. Therefore,

$$
\begin{aligned}
\log Z \quad &\geq \quad \sum_{(i,j)\in M} \left(\psi_{ij}^U - \psi_{ij}^L\right) \\
&\geq \quad \frac{1}{d^* + 1} \left(\sum_{(i,j)\in E} \left(\psi_{ij}^U - \psi_{ij}^L\right) \right), \qquad (2.28)
\end{aligned}
$$

using fact about weight of M. This completes the proof of Lemma 14.

Lemma 15 *If G has maximum vertex degree d^* and the* **DECOMP***(G) produces $\mathcal{B}$ that is (ε, Δ) edge-decomposition, then*

$$
\mathbb{E}\left[\log \hat{Z}_{UB} - \log \hat{Z}_{LB}\right] \leq \varepsilon(d^* + 1) \log Z,
$$

with respect to the randomness in $\mathcal{B}$, and the **Log Partition** *takes time $O(nd^*|\Sigma|^{\Delta}) + T_{\mathbf{DECOMP}}$.*

Proof. From Theorem 5, Lemma 14 and the definition of (ε, Δ) edge-decomposition, we have the following.

$$\begin{aligned}
\mathbb{E}\left[\log \hat{Z}_{\mathrm{UB}} - \log \hat{Z}_{\mathrm{LB}}\right] &\leq \mathbb{E}\left[\sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L)\right] \\
&= \sum_{(i,j)\in E} \mathbb{P}((i,j)\in\mathcal{B})(\psi_{ij}^U - \psi_{ij}^L) \\
&\leq \varepsilon\left[\sum_{(i,j)\in E} (\psi_{ij}^U - \psi_{ij}^L)\right] \\
&\leq \varepsilon(d^*+1)\log Z.
\end{aligned}$$

Now to estimate the running time, note that under (ε, Δ) decomposition $\mathcal{B}$, with probability 1 the $G' = (V, E\backslash\mathcal{B})$ is divided into connected components with at most Δ nodes. Therefore, the running time bound of Theorem 5 implies the desired result.

2.6.4 Analysis of Log Partition: Polynomially growing G

Here we interpret result obtained in Theorem 5 and Lemma 15, for polynomially growing G and uses edge-decomposition scheme **POLY-E**.

Theorem 6 *Let MRF graph G of n nodes with growth rate ρ and corresponding constant C be given. Consider any constant $\varepsilon \in (0,1)$, and define $\varphi = \frac{\varepsilon}{2^\rho C}$. Then* **Log Partition** *using* **POLY-E**$(\varphi, K(\varphi,\rho,C))$ *produces bounds* $\log \hat{Z}_{LB}, \log \hat{Z}_{UB}$ *such that*

$$\mathbb{E}\left[\log \hat{Z}_{UB} - \log \hat{Z}_{LB}\right] \leq \varepsilon \log Z.$$

Further, the algorithm takes $O(n)$ amount of time to obtain these estimates.

Proof. First, **POLY-E**$(\varphi, K(\varphi,\rho))$ produces $(\varphi, O(1))$ edge-decomposition from Lemma 6.

Note that the maximum vertex degree d^* of G is less than $2^\rho C$ by the definition of polynomially growing graph. Therefore, by Lemma 15, the output produced by

the **MAP** algorithm is such that

$$\begin{aligned}\mathbb{E}\left[\log \hat{Z}_{\mathrm{UB}} - \log \hat{Z}_{\mathrm{LB}}\right] &\leq \varepsilon 2^{-\rho-2}(d^*+1)\log Z \\ &\leq \varepsilon \log Z, \end{aligned} \tag{2.29}$$

Further, the running time of the algorithm is $O(nd^*|\Sigma|^{K(\varphi,\rho)}) + T_{\mathbf{DECOMP}} = O(n)$. This completes the proof of Theorem 6.

2.6.5 Analysis of Log Partition: Minor-excluded G

When G is a minor excluded graph with bounded vertex degree, we use **MINOR-E** for the edge-decomposition. We apply Theorem 1 and Lemma 11 to obtain the following result.

Theorem 7 *Let MRF graph G of n nodes exclude $K_{r,r}$ as its minor. Let d^* be the maximum vertex degree in G. Given a constant $\varepsilon > 0$, use* **Log Partition** *algorithm with* **MINOR-E**(G, r, Λ) *where $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$. Then,* **Log Partition** *produces bounds such that*

$$\log \hat{Z}_{LB} \leq \log Z \leq \log \hat{Z}_{UB}; \text{ and}$$

$$\mathbb{E}\left[\log \hat{Z}_{UB} - \log \hat{Z}_{LB}\right] \leq \varepsilon \log Z.$$

The algorithm takes $O(n)$ time.

Proof. From Lemma 9 about the **MINOR-E** algorithm, we have that with choice of $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$, the algorithm produces $(\frac{\varepsilon}{d^*+1}, \Delta)$ edge-decomposition where $\Delta = d^{*O(\Lambda)}$. Since it is an $(\frac{\varepsilon}{d^*+1}, \Delta)$ edge-decomposition, the upper bound and the lower bound, $\log \hat{Z}_{\mathrm{UB}}, \log \hat{Z}_{\mathrm{LB}}$, for the value produced by the algorithm are within $(1 \pm \varepsilon) \log Z$ by Lemma 15.

Now, by Lemma 15 the algorithm running time is $O(nd^*|\Sigma|^{\Delta}) + T_{\mathbf{DECOMP}}$. As discussed earlier in Lemma 9, the algorithm **MINOR-E** takes $O(r|E|) = O(nrd^*)$ operations. That is, $T_{\mathbf{DECOMP}} = O(n)$. Now, $\Delta = d^{*O(\Lambda)}$ and $\Lambda \leq r(d^*+1)/\varepsilon + 1$.

Therefore, the first term of the computation time bound $O(nd^*|\Sigma|^{\Delta}) = O(n)$. Hence the running time of **MAP** is $O(n)$. This completes the proof of Theorem 7.

2.7 An unexpected implication: existence of limit

This section describes an important and somewhat unexpected implication of our result, specifically Lemma 15. In the context of *regular* MRF, such as an MRF on $\mathbb{Z}_n^d$ (of n^d nodes) with same node and edge potential functions for all nodes and edges, we will show that (non-trivial) limit $\frac{1}{n^d} \log Z$ exists as $n \to \infty$. We believe that its proof method is likely to allow for establishing such existence for a more general class of problems. As an example, the theorem will hold even when node and edge potentials are not the same but are chosen from a class of such potential as per some distribution in an i.i.d. fashion. Now, we state the result.

Theorem 8 *Consider a regular MRF of n^d nodes on d-dimensional grid $\mathbb{Z}_d^n = (V_n, E_n)$: let $\psi_{ij} \equiv \psi, \phi_i \equiv \phi$ for all $i \in V_n, (i,j) \in E_n$ with $\psi : \Sigma^2 \to \mathbb{R}_+, \phi : \Sigma \to \mathbb{R}_+$. Let Z_n be partition function of this MRF. Then, the following limit exists:*

$$\lim_{n\to\infty} \frac{1}{n^d} \log Z_n = A(d, \phi, \psi) \in (0, \infty).$$

2.7.1 Proof of Theorem 8

The proof of Theorem 8 is stated for $d = 2$ and $\Sigma = \{0, 1\}$ case. Proof for $d \geq 3$ and Σ with $|\Sigma| \geq 2$ can be proved using exactly the same argument. The proof will use the following Lemmas.

Lemma 16 *Let $d = 2$ and $\phi^* = \max_{\sigma\in\{0,1\}} \phi(\sigma)$, $\psi^* = \max_{(\sigma,\sigma')\in\{0,1\}^2} \psi(\sigma, \sigma')$. Then,*

$$n^2 \leq \log Z_n \leq \alpha n^2,$$

where $\alpha = \log 2 + \log \phi^* + 4 \log \psi^*$.

Lemma 17 *Define* $a_n = \frac{1}{n^2} \log Z_n$. *Now, given* $k > 0$, *there exists* $n(k)$ *large enough such that for any* $m, n \geq n(k)$,

$$|a_m - a_n| = O\left(\frac{1}{k}\right) + O\left(\frac{k}{\min\{m,n\}}\right).$$

Proof. (Theorem 8) We state the proof of Theorem 8, before proving the above stated Lemmas. First note that, by Lemma 16, the elements of sequence $a_n = n^{-2} \log Z_n$ take value in $[1, \alpha]$. Now, suppose the claim of theorem is false. That is, sequence a_n does not converge as $n \to \infty$. That is, there exists $\delta > 0$ such for any choice of n_0, there are $m \geq n \geq n_0$ such that

$$|a_m - a_n| \geq \delta.$$

By Lemma 17, we can select k large enough and later $n_0 \geq n(k)$ large enough such that for any $m, n \geq n_0$,

$$|a_m - a_n| < \delta.$$

But this is a contradiction to our assumption that a_n does not converge to a limit. That is, we have established that a_n converges to a non-trivial limit in $[1, \alpha]$ as desired. This completes the proof of Theorem 8.

2.7.2 Proof of Lemmas

Proof. (Lemma 16) Consider the following.

$$\begin{aligned}
2^{n^2} &= \sum_{\mathbf{x} \in \{0,1\}^{n^2}} \prod_{i \in V_n} 1 \prod_{(i,j) \in E_n} 1 \\
&\overset{(a)}{\leq} \sum_{\mathbf{x} \in \{0,1\}^{n^2}} \prod_{i \in V_n} \exp(\phi(x_i)) \prod_{(i,j) \in E_n} \exp(\psi(x_i, x_j)) \\
&= Z_n \\
&\overset{(b)}{\leq} \sum_{\mathbf{x} \in \{0,1\}^{n^2}} \prod_{i \in V_n} \exp(\phi^*) \prod_{(i,j) \in E_n} \exp(\psi^*). \qquad (2.30)
\end{aligned}$$

Here, (a) follows from the fact that ψ, ϕ are non-negative valued functions and (b) follows from definitions of ϕ^*, ψ^*. Now, taking logarithm on both sides implies the Lemma 16.

Proof. (Lemma 17) Given $k > 0$, consider n large enough (will be decided later). Consider $\mathbb{Z}_n^2 = (V_n, E_n)$ and let it be laid out on $X - Y$ plane so that its node in V_n occupy the integral locations : (i, j), $0 \leq i \leq n-1, 0 \leq j \leq n-1$. Now, we describe a scheme to obtain a $(O(1/k), O(k^2))$ edge-decomposition of $\mathbb{Z}_n^2$. For this, choose $\ell_1, \ell_2 \in \{0, \ldots, k-1\}$ independently and uniformly at random. Select edges to form $\mathcal{B}$ to obtain edge-decomposition as follows: select vertical edges with bottom vertex having Y coordinate $\ell_2 + jk, j \geq 0$, and select horizontal edges with left vertex having X coordinate $\ell_1 + jk, j \geq 0$. That is,

$$\begin{aligned} \mathcal{B} &= \{(u,v) \in E_n : u = (i,j), v = (i+1, j), i \bmod k = \ell_1\} \\ &\cup \{(u,v) \in E_n : u = (i,j), v = (i, j+1), j \bmod k = \ell_2\}. \end{aligned}$$

It is easy to check that this is $(O(1/k), O(k^2))$ edge-decomposition due to uniform selection of ℓ_1, ℓ_2 from $\{0, \ldots, k-1\}$. Therefore, by Lemma 15, we can obtain estimates that are $(1 \pm O(1/k)) \log Z_n$ using our algorithm.

Let $m = \lceil n/k \rceil$. Under the decomposition $\mathcal{B}$ as described above, there are at least $(m-1)^2$ connected components that are MRF on $\mathbb{Z}_k$. Also, all the connected components can be covered by at most $(m+1)^2$ identical MRFs on $\mathbb{Z}_k$. Using arguments similar to those employed in calculations of Theorem 5 (using non-negativity of ϕ, ψ), it can be shown that the estimate produced by our algorithm is lower bounded as

$$(1 - O(1/k))(m-1)^2 \log Z_k = n^2 \frac{\log Z_k}{k^2} \times (1 - O(1/k) - O(k/n)),$$

and is upper bounded as

$$(1 + O(1/k))(m+1)^2 \log Z_k = n^2 \frac{\log Z_k}{k^2} \times (1 + O(k/n) + O(1/k)).$$

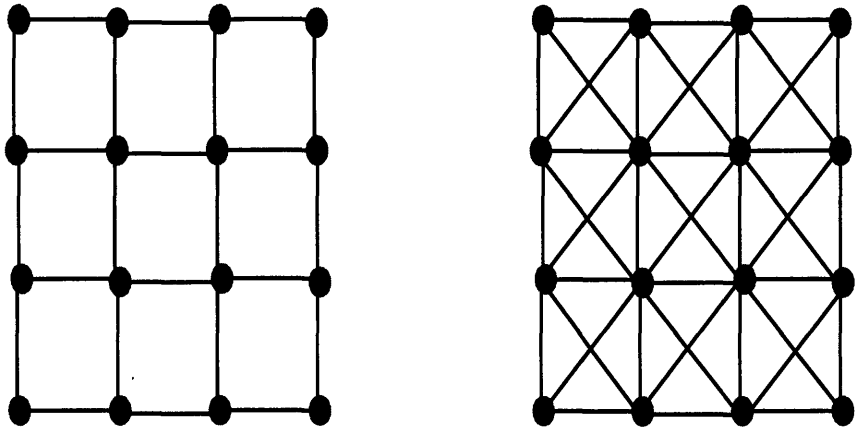

Figure 2-12: Example of grid graph (left) and cris-cross graph (right) with $n = 4$.

Therefore, from above discussion we obtain that

$$\frac{1}{n^2} \log Z_n = \frac{Z_k}{k^2} \left(1 \pm O(k/n) \pm O(1/k)\right).$$

Therefore, recalling notation of a_n, we have that

$$|a_m - a_n| = a_k O\left(\frac{k}{\min\{m,n\}}\right) + a_k O(1/k).$$

Since, $a_k \in [1, \alpha]$ for all k, we obtain the desired result of Lemma 17.

2.8 Experiments

Our algorithm provides provably good approximation for any positive MRF defined on a graph G that is either polynomially growing or minor-excluded with bounded degree. We present simulation results on the grid graphs, which satisfies both conditions. Grid graph appears in many application of MRF including Ising model and computer graphics. We will use the decomposition algorithm **MINOR-E** to obtain our results. Now we will present detailed setups and experimental results.

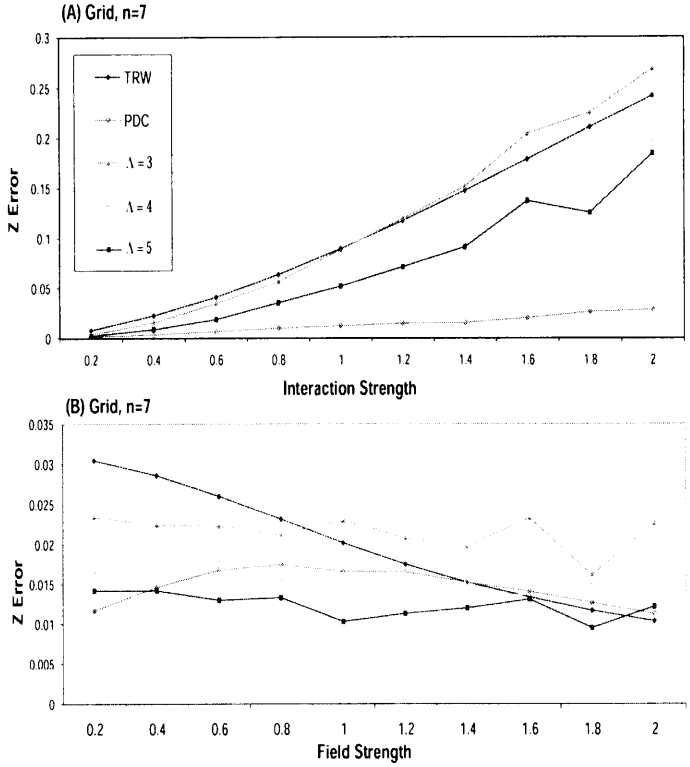

Figure 2-13: Comparison of TRW, PDC and our algorithm for grid graph with $n = 7$ with respect to error in $\log Z$. Our algorithm outperforms TRW and is competitive with respect to PDC.

2.8.1 Setup 1

Consider[4] the following binary (i.e. $\Sigma = \{0, 1\}$) MRF on an $n \times n$ grid $G = (V, E)$:

$$\mathbb{P}(\mathbf{x}) \propto \exp\left(\sum_{i \in V} \theta_i x_i + \sum_{(i,j) \in E} \theta_{ij} x_i x_j\right), \quad \text{for } \mathbf{x} \in \{0,1\}^{n^2}.$$

Figure 2-12 shows a lattice or grid graph with $n = 4$ (on the left side). There are two scenarios for choosing parameters (with notation $\mathcal{U}[a, b]$ being uniform distribution over interval $[a, b]$):

[4]Though this setup has ϕ_i, ψ_{ij} taking negative values, they are equivalent to the setup considered in the paper as the function values are lower bounded and hence *affine* shift will make them non-negative without changing the distribution.

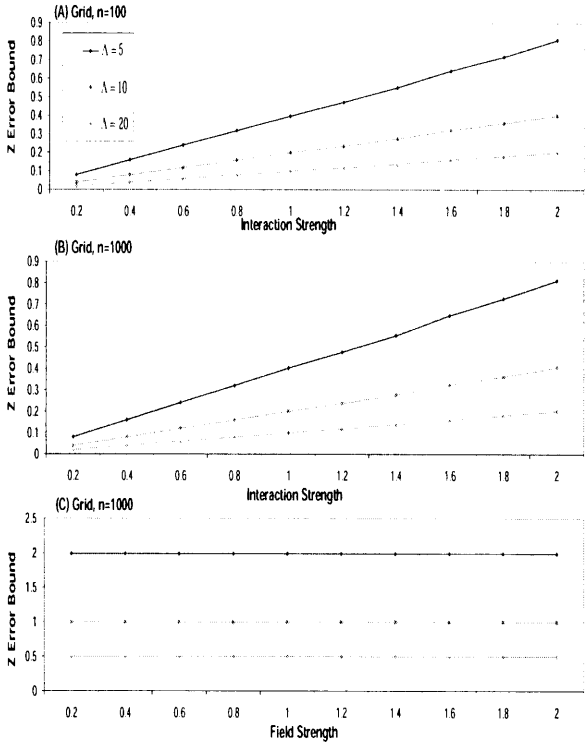

Figure 2-14: The theoretically computable error bounds for $\log Z$ under our algorithm for grid with $n = 100$ and $n = 1000$ under varying interaction and varying field model. This clearly shows scalability of our algorithm.

(1) *Varying interaction.* θ_i is chosen independently from distribution $\mathcal{U}[-0.05, 0.05]$ and θ_{ij} chosen independent from $\mathcal{U}[-\alpha, \alpha]$ with $\alpha \in \{0.2, 0.4, \ldots, 2\}$.

(2) *Varying field.* θ_{ij} is chosen independently from distribution $\mathcal{U}[-0.5, 0.5]$ and θ_i chosen independently from $\mathcal{U}[-\alpha, \alpha]$ with $\alpha \in \{0.2, 0.4, \ldots, 2\}$.

We run our algorithms **Log Partition** and **MAP**, with decomposition scheme **MINOR-E**$(G, 3, \Lambda)$, $\Lambda \in \{3, 4, 5\}$. We consider two measures to evaluate performance: error in $\log Z$, defined as $\frac{1}{n^2}|\log Z^{\text{alg}} - \log Z|$; and error in $\mathbb{E}(\mathbf{x}^*)$, defined as $\frac{1}{n^2}|\mathbb{E}(\mathbf{x}^{\text{alg}} - \mathbb{E}(\mathbf{x}^*)|$.

We compare our algorithm for error in $\log Z$ with the two recently very successful algorithms – Tree re-weighted algorithm (TRW) and planar decomposition algorithm

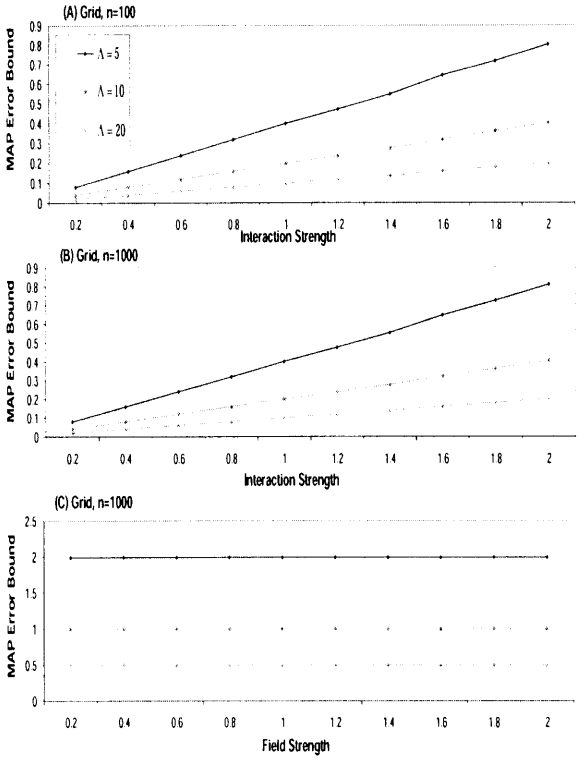

Figure 2-15: The theoretically computable error bounds for MAP under our algorithm for grid with $n = 100$ and $n = 1000$ under varying interaction and varying field model.

(PDC). The comparison is plotted in Figure 2-13 where $n = 7$ and the results are averages over 40 trials. The Figure (A) plots error with respect to varying interaction while Figure (B) plots error with respect to varying field strength. Our algorithm, essentially outperforms TRW for these values of Λ and perform very competitively with respect to PDC.

The key feature of our algorithm is scalability. Specifically, running time of our algorithm with a given parameter value Λ scales linearly in n, while keeping the relative error bound exactly the same. To explain this feature, we plot the upper bound on error in $\log Z$ in Figure 2-14 with tags (A), (B) and (C). Note that the error bound plot is the same for $n = 100$ (A) and $n = 1000$ (B). Clearly, actual error is likely to be smaller than these theoretically plotted bounds. We note that

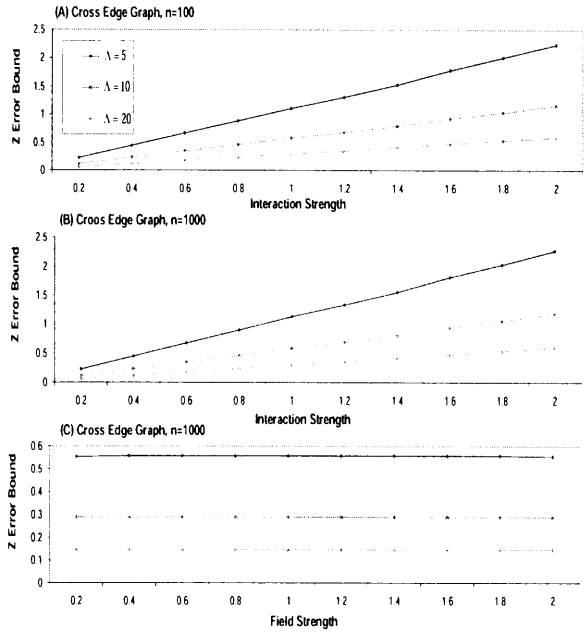

Figure 2-16: The theoretically computable error bounds for $\log Z$ under our algorithm for cris-cross with $n = 100$ and $n = 1000$ under varying interaction and varying field model. This clearly shows scalability of our algorithm and robustness to graph structure.

these bounds only depend on the interaction strengths and *not* on the values of fields strengths (C).

Results similar to of **Log Partition** are expected from **MAP**. We plot the theoretically evaluated bounds on the error in MAP in Figure 2-15 with tags (A), (B) and (C). Again, the bound on MAP relative error for given Λ parameter remains the same for all values of n as shown in (A) for $n = 100$ and (B) for $n = 1000$. There is no change in error bound with respect to the field strength (C).

2.8.2 Setup 2

Everything is exactly the same as the above setup with the only difference that grid graph is replaced by *cris-cross* graph which is obtained by adding extra four neighboring edges per node (exception of boundary nodes). Figure 2-12 shows cris-cross graph with $n = 4$ (on the right side). We again run the same algorithm as above

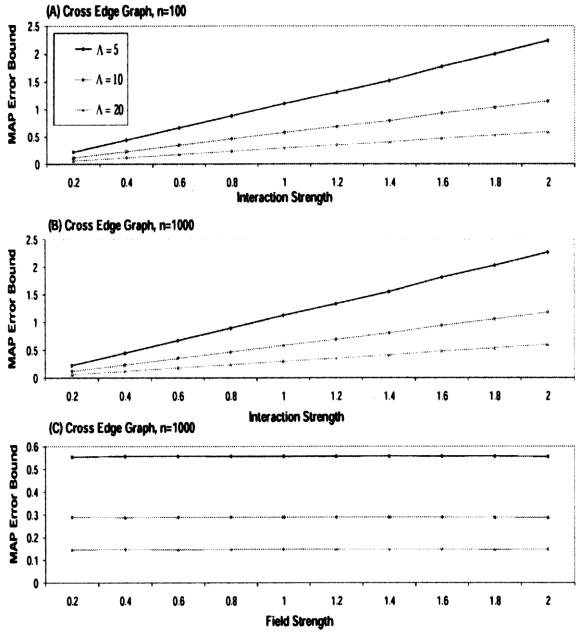

Figure 2-17: The theoretically computable error bounds for MAP under our algorithm for cris-cross with $n = 100$ and $n = 1000$ under varying interaction and varying field model.

setup on this graph. For cris-cross graph, which is a polynomially growing graph with growth rate 2, we obtained its graph decomposition from the same decomposition we have obtained for its grid sub-graph. Therefore , the running time of our algorithm remains the same (in order) as that of grid graph and error bound will become only 3 times weaker than that of the grid graph. We compute these theoretical error bounds for $\log Z$ and MAP which is plotted in Figure 2-16 and 2-17. These figures are similar to the Figures 2-14 and 2-15 for the grid graphs.

Chapter 3

Application of MAP : Wireless Network

3.1 Capacity region of wireless network

Wireless networks are becoming the architecture of choice for designing many of the emerging communication networks such as mesh networks to provide infrastructure in metro areas, peer-to-peer networks, and to provide infrastructure free interactions between handheld devices in popular locations like shopping malls or movie theaters, mobile ad-hoc network between vehicles for IVHS, etc. In all such settings, in essence we have a wireless network of n nodes where nodes are communicating over a common wireless medium using a certain standard communication protocol (e.g. IEEE 802.11 standard). Under any such protocol, transmission between a pair of nodes is successful iff none of the *nearby* or *interfering* nodes are transmitting simultaneously. Any such interference model is equivalent to an *independent set* interference model over the graph of interfering communication links.

A key operational question in any such network (e.g. a mesh network in a metro area) is that of determining whether a given set of end-to-end rates between various source destination pairs is simultaneously supportable by the network. That is, one wishes to determine the n^2 dimensional unicast capacity region of such a wireless network of n nodes. An algorithm for determining this feasibility must be distributed

(i.e. operation at a node utilizes only the information of the node's neighbors) and very efficient in order to be implementable.

However, an algorithm for determining feasibility of end-to-end rates has to explore over exponentially large space of joint routing and scheduling decisions under the wireless network interference constraints. This makes the question of designing such an efficient, distributed algorithm potentially very hard. Indeed, this question is known to be NP-hard and hard to approximation within $n^{1-o(1)}$ factor for general graphs [57].

But since wireless networks are usually formed between nodes that are placed in a geographic area, they possess a natural *geometry*. Therefore, a natural question arises: is it possible to design efficient algorithms for checking end-to-end rate feasibility for a wireless network arising in practice (i.e. possessing geometry)?

3.2 Previous works

In past decade or so, emergence of wireless network architectures have led various researchers to take two different approaches to design efficient algorithms for checking feasibility of end-to-end rates.

The first approach is inspired by the possibility of deriving explicit simple bounds. Specifically, starting work by Gupta and Kumar [16] significant effort has been put in to derive simple scaling laws for large random wireless network for random traffic demands. In essence, this result implies that under such a random regular setup, per source supportable rate scales like $1/\sqrt{n}$ in the network of n nodes. Thus, if such a random setting is a good approximation of the network operating in practice, then one can utilize this $1/\sqrt{n}$ formula to determine approximate feasibility. The possible effectiveness of such an approach has led to an extensive study of a related notion of *transport capacity*, introduced in [16], over the past decade. For example, see works by [10,23,41] and many others. We also refer an interested reader to a comprehensive survey by Xue and Kumar [70]. More recently, a complete information theoretic characterization of random traffic demand were obtained for random node placement

by Ozgur, Tse and Leveque [48] and for arbitrary node placement by Niesen, Gupta and Shah [47].

The second approach is based on determining the exact or approximate feasibility for a given arbitrary wireless network operating under a specific interference model. The question of determining feasibility of end-to-end rates is equivalent to checking feasibility of a solution of a certain Linear Program (LP). However, this LP is very high dimensional (due to exponentially many routing and scheduling choices) and hence exact solutions like simplex algorithm for this LP are inefficient. Various authors have provided approaches to design approximation algorithm with constant factor loss for such an LP with the constant factor loss being a function of the *degree* of nodes in the interference graph.

In general, given a specific network and a vector of end-to-end rates between various source destination pairs, the problem of determining their feasibility is NP-hard. In [13], the problem of determining feasibility of rates when sources wishes to send data to their neighbors directly was considered, i.e. the problem of feasibility of rates for a single hop network. A polynomial time approximation was developed for this problem when the network possesses geometry. As explained earlier, the single hop rate feasibility algorithm does not lead to feasibility of end-to-end rate feasibility primarily due to additional freedom of routing over exponentially many choices. It is also important to note that the multi-hop routing version is not a generalization of the single hop problem. This is because, even if we consider all source destination pairs as 1-hop neighbors, it is possible that the rates are feasible through a multi-hop routing scheme while the trivial single hop routing itself is infeasible. In that sense, the two problems, though related, are in fact, not generalizations of one another.

3.3 Our contributions

As our main result, we take a different approach to directly tackle the multihop problem. A well known result of Tassiulas and Ephremides [61], says that if the given end-to-end rates are feasible, then a network with i.i.d. arrivals of mean equal

to these rates (and of bounded second moment) under the maximum weight based combined scheduling and routing policy will lead to a stable Markov process of the queue lengths. The maximum weight scheduling algorithm is required to determine schedule of nodes subject to network scheduling constraints that are imposed by the network physical layer.

Although the maximum weight scheduling problem may become NP hard, the hardness result is an *adversarial* or a *worst-case* result. Therefore, it is natural to inquire whether there is a large class of network structure for which such impossibility result does not hold. Implicitly, such line of query was pursued in some of the recent works for very specific graph structures. Specifically, work by Hunt et. al. [19] provides an approximation algorithm for maximum weight independent set for unit disk graphs. This was utilized by Sharma, Mazumdar and Shroff [58] in their work to identify that max-weight scheduling with $K-$hop matching. A similar approach was employed in another recent work by Sarkar and Ray [55] for a very restricted class of graph structures. However, identification of a broad class of graph structures that posses such property has remained unknown.

As explained in Section 1.3.2, the following pair-wise MRF model is widely used to express the MWIS for network scheduling.

$$W(\mathbf{x}) \quad \propto \quad \exp\left[\sum_{v\in V} W_v(\tau)\cdot x_v\right]\cdot \prod_{(u,v)\in E} \Psi(x_u,x_v), \tag{3.1}$$

where Ψ is defined so that $\Psi(x_1,x_2)=0$ if $x_1=x_2=1$, and $\Psi(x_1,x_2)=1$ otherwise.

In most of the applications, practical network graphs have the following *geometry*. The size of the local neighborhood of each node in the network grows polynomially, i.e. they are polynomially growing graphs. As we have shown in Section 2.3.1, many of the popular graph models are special instances of our thus identified structure. For example, popular wireless network model of geometric random graph (e.g. Gupta and Kumar [16]) or popular computational geometric graph model of finite doubling dimension graphs are special instances of our polynomially growing graphs. Notice that (3.1) is not a positive MRF, hence we cannot apply our **MAP** algorithm directly.

In Section 3.5, we provide a randomized algorithm that computes approximate MWIS within any accuracy.

Now, provided the network constraint graph allows for efficient approximation to the MWIS problem, we suggest the following vaguely stated approach for a feasibility test of end to end rates : Simulate a network with fixed packet arrival of the given end-to-end rates, and use the given *approximate* MWIS algorithm for the packet transmit scheduling. Hopefully, if the queues remain "**stable**" then the rates are *approximately* feasible or else they are *approximately* infeasible. This is the basic idea behind our approach. However, in order to make this approach 'feasible', we need to deal with a host of non-trivial issues that are stated below:

- Even if one had an efficient exact MWIS algorithm, its popular analysis (as in [61]) provides bounds on the queue lengths of the stationary distribution, not for any time. Hence we need to characterize the queue lengths explicitly for any time in order to be able to design an algorithm.

- The queue length bounds obtained by using the standard Foster's criterion and moment bounds are only statements on the equilibrium distribution. This means that in order to establish an absolute bound, one might need to estimate the rates of convergence and we would have a polynomial algorithm if this convergence is quick enough. We take somewhat novel approach where we iterate analysis with the design along with the use of real valued queue lengths with deterministic fractional arrivals. By doing so, the queue-size vector does not remain a Markov chain on integer state space but, our direct analysis leads to bounds that are sufficient for our purposes and leads to the approximate correctness property of the algorithm that we propose.

3.4 Model and main results

In this section, we explain the mathematical model of the network we consider, and state our main results. Consider a wireless network on n nodes defined by a directed

graph $G = (V, E)$ with $|V| = n, |E| = L$. For any $e \in E$, let $\alpha(e), \beta(e)$ denote respectively the origin and destination vertices of the edge e. The edges denote potential wireless links, and only the subsets of the edges that do not interfere can be simultaneously active. We say two edges *interfere* when they share a vertex. Let

$$\mathcal{S} = \{\mathbf{e} \in \{0,1\}^L : \mathbf{e} \text{ is the adjacency vector for a non-interfering subset of } E\}$$

Note that $\mathcal{S}$ is the collection of the *independent sets of* E by considering interference among $e_1 \in E$ and $e_2 \in E$ as the edge between them. Given a graph $G = (V, E)$ on L nodes and node weights given by $\mathbf{w} = (w_1, \ldots, w_L) \in \mathbb{R}_+^L$, a subset $\mathbf{x}$ of V is said to be an *independent set* if no two vertices of $\mathbf{x}$ have a common edge. Let $\mathcal{I}(G)$ be the set of all independent sets of G. A *maximum weight independent set* (MWIS) $\mathbf{x}^*$ of a vertex weighted graph is defined by $\mathbf{x}^* = \text{argmax}\left\{\mathbf{w}^T\mathbf{x} : \mathbf{x} \in \mathcal{I}(G)\right\}$, where we consider $\mathbf{w}$ as an element of $\{0,1\}^{|V|}$.

The convex hull of $\mathcal{S}$, denoted by $co(\mathcal{S})$ in $\mathbb{R}^L$ represents the link rate feasibility region. Typically, the set $co(\mathcal{S})$ is complicated to describe (and exponential in size). Determining membership in $co(\mathcal{S})$ was a problem shown to be NP-hard by [1] under general interference constraints.

In this chapter, we consider m distinct source destination pairs, $(s_1, d_1), \ldots, (s_m, d_m)$ and an *end to end* rate vector, $\mathbf{r} = (r_1, r_2, \ldots, r_m) \in [0,1]^m$. We will use the index j to range over the Source-Destination pairs in the following discourse.

Definition 7 *The rate vector* $\boldsymbol{r} = (r_1, \ldots, r_m) \in [0,1]^m$ *corresponding to the Source-Destination pairs* $(s_1, d_1), \ldots, (s_m, d_m)$ *is said to be "**feasible**", if there exist flows,* $(\boldsymbol{f}^1, \ldots, \boldsymbol{f}^m)$ *such that*

- $\boldsymbol{f}^j$ *routes a flow of at least* r_j *from* s_j *to* d_j *for* $1 \leq j \leq m$
- *The net flow on the links induced,* $\hat{\boldsymbol{f}} \doteq \sum_{j=1}^m \boldsymbol{f}^j$ *belongs to* $co(\mathcal{S})$*, i.e. in other words, it can be scheduled under the interference constraints with a schedule of at most unit length.*

The equations that specify the notion of "flows routing $\mathbf{r}$" are given later in (3.14) via (3.6) and (3.12). Let

$$\mathcal{F} = \{\mathbf{r} \in [0,1]^m : \mathbf{r} \text{ is "feasible"}\}$$

be the set of all feasible end to end rate vectors. Our primary results are the followings.

Definition 8 *Given a constant $\varepsilon > 0$ and a graph G with a vertex weight $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{R}_+^n$, we say a randomized algorithm is an* ε-approximation algorithm of MWIS *of G if it outputs an independent set $\hat{\mathbf{x}} \in \mathcal{I}(G)$ that satisfies*

$$\mathbb{E}[\mathbf{w}^T \hat{\mathbf{x}}] \geq (1-\varepsilon)\mathbf{w}^T \mathbf{x}^*.$$

Theorem 9 *Let the interference graph of the network be a polynomially growing graph. Then for any constant $\varepsilon > 0$, there is an ε-approximation algorithm of MWIS that runs in $O(n)$ time.*

Theorem 10 *Assume we have a randomized ε-approximation algorithm of MWIS for a wireless network for $0 < \varepsilon < 1/4$. Then, there exists a randomized polynomial time algorithm to determine the approximate rate feasibility of a given end to end rate vector $\boldsymbol{r}$ in the following sense: If $(1+2\varepsilon)\boldsymbol{r} \in \mathcal{F}$, then the algorithm outputs a 'YES' with high probability. Conversely, if $(1-2\varepsilon)\boldsymbol{r} \notin \mathcal{F}$, then the algorithm outputs a 'NO' with high probability. Else, the answer could be arbitrary.*

3.5 Approximate MWIS for polynomially growing graphs

In this section, we present a randomized algorithm **MWIS** that computes an ε-approximation of MWIS for any polynomially growing graph and constant $\varepsilon > 0$. Let a graph $G = (V, E)$ is given with vertex weights $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{R}_+^n$. Assume that G is a polynomially growing graph with growth rate ρ and the corresponding constant C.

MWIS(ε, K)

(1) For the given graph G, use **POLY-V**(ε, K) to obtain a vertex decomposition of G.

  (a) Let $\mathcal{B}$ be the output of the decomposition algorithm.

  (b) $V - \mathcal{B}$ is divided into connected components with vertex sets $\mathcal{R}_1, \ldots, \mathcal{R}_j$ (j is some integer).

  (c) Let $G_1 = (\mathcal{R}_1, E_1), \ldots, G_L = (\mathcal{R}_j, E_j)$ be the corresponding disjoint subgraphs of G.

  (d) Let $\mathcal{I}(G_1), \ldots, \mathcal{I}(G_j)$ be set of independent sets of $G_1, \ldots, G_j$ respectively.

(2) For $\ell = 1, \ldots, j$ find

$$\mathbf{x}^*(G_\ell) \in \arg\max \left\{ \mathbf{w}^T \mathbf{x} : \mathbf{x} \in \mathcal{I}(G_\ell) \right\}.$$

  (a) The above computation can be done by dynamic programming or exhaustive computation in $O(2^{|\mathcal{R}_\ell|})$ operations for graph G_ℓ.

(3) Output $\hat{\mathbf{x}} = \cup_{\ell=1}^{j} \mathbf{x}^*(G_\ell)$ as an approximate MWIS of G.

Figure 3-1: Randomized algorithm for approximate MWIS

The main procedure of the algorithm **MWIS** is similar to that of **MAP**, except that **MWIS** uses vertex decomposition **POLY-V** for its graph decomposition subroutine unlike the fact that **MAP** uses **POLY-E**. This is to make sure that the merged solution of the independent sets computed within each components is also an independent set of G. Figure 3-1 describes the algorithm **MWIS**.

First, we state and prove a Lemma that evaluates the performance of **MWIS** for any graph.

Lemma 18 *Given a vertex weighted graph G, $\varepsilon > 0$ and K, the output $\hat{\mathbf{x}}$ of the* **MWIS** *(ε, K) has the following property:*

$$\mathbb{E}\left[\mathbf{w}^T \hat{\mathbf{x}}\right] \geq \mathbf{w}^T \mathbf{x}^* \left[1 - \varepsilon - P_K \left(\max_{v \in V} |\mathbf{B}(v, K)| \right)\right],$$

where $P_K = (1 - \varepsilon)^{K-1}$.

Proof. Let the **POLY-V** applied to G produce a random subset $\mathcal{B} \subset V$ as its output

and $V - \mathcal{B}$ be divided into connected components, $\mathcal{R}_1, \ldots, \mathcal{R}_j$. As described in the **POLY-V**, we have induced (disconnected) subgraphs $G_\ell = (\mathcal{R}_\ell, E_\ell)$, $1 \leq \ell \leq j$. The output of the **MWIS** is

$$\hat{\mathbf{x}} = \cup_{\ell=1}^{j} \mathbf{x}^*(G_\ell),$$

where $\mathbf{x}^*(G_\ell)$ is the solution of maximum weight independent set in the subgraph G_ℓ. In the above expression, the $\cup$ operation of $\mathbf{x}^*(G_\ell)$'s means the standard union of sets. Note that $\mathbf{x}^*(G_\ell)$ are disjoint sets since G_ℓ are disconnected graphs. Therefore,

$$\mathbf{w}^T \hat{\mathbf{x}} \;=\; \sum_{\ell=1}^{j} \mathbf{w}^T \mathbf{x}^*(G_\ell). \tag{3.2}$$

Now consider any maximum weight independent set $\mathbf{x}^*$, i.e. $\mathbf{x}^* \in \arg\max \left\{ \mathbf{w}^T \mathbf{x} : \mathbf{x} \in \mathcal{I}(G) \right\}$. The $\mathbf{x}^*$ corresponds to a subset of vertices V of G. The intersection of $\mathbf{x}^*$ and $\mathcal{R}_\ell$ induces an independent set in G_ℓ. Let

$$\mathbf{x}_\ell^* = \mathbf{x}^* \cap \mathcal{R}_\ell.$$

Since $\mathbf{x}^*(G_\ell)$ is a maximum weight independent set in G_ℓ, we have

$$\mathbf{w}^T \mathbf{x}^*(G_\ell) \;\geq\; \mathbf{w}^T \mathbf{x}_\ell^*. \tag{3.3}$$

Finally, denote $\mathbf{x}_{\mathcal{B}}^* = \mathbf{x}^* \cap \mathcal{B}$. Since $V = \mathcal{B} \cup \left(\cup_{\ell=1}^{j} \mathcal{R}_\ell \right)$ and all sets $\mathcal{B}, \mathcal{R}_1, \ldots, \mathcal{R}_L$ are disjoint, we have

$$\mathbf{w}^T \mathbf{x}^* \;=\; \mathbf{w}^T \mathbf{x}_{\mathcal{B}}^* + \sum_{\ell=1}^{L} \mathbf{w}^T \mathbf{x}_\ell^*. \tag{3.4}$$

From (3.2)-(3.4), we obtain

$$\mathbf{w}^T \hat{\mathbf{x}} \geq \mathbf{w}^T \mathbf{x}^* - \mathbf{w}^T \mathbf{x}_{\mathcal{B}}^*.$$

Therefore, with respect to the randomness in the **POLY-V** (using Lemma 4), we

have

$$\begin{aligned}\mathbb{E}\left[\mathbf{w}^T\hat{\mathbf{x}}\right] &\geq \mathbf{w}^T\mathbf{x}^* - \sum_{v\in\mathbf{x}^*} w_v\mathbb{P}[v\in\mathcal{B}] \geq \mathbf{w}^T\mathbf{x}^* - \left(\sum_{v\in\mathbf{x}^*} w_v\left(\varepsilon + P_K\left(\max_{v\in V}|\mathbf{B}(v,K)|\right)\right)\right)\\ &= \mathbf{w}^T\mathbf{x}^* - \left(\varepsilon + P_K\left(\max_{v\in V}|\mathbf{B}(v,K)|\right)\right)\left(\sum_{v\in\mathbf{x}^*} w_v\right)\\ &= \mathbf{w}^T\mathbf{x}^*\left[1-\varepsilon - P_K\left(\max_{v\in V}|\mathbf{B}(v,K)|\right)\right]. \qquad (3.5)\end{aligned}$$

This completes the proof of Lemma 18.

Before we state the error bound of **MWIS** for polynomially growing graph G, we give some definitions. Given a graph with growth rate ρ and corresponding constant C, and a constant $\delta\in(0,1)$, let

$$\varepsilon(\delta) = \frac{\delta}{2} \quad \text{and} \quad K(\delta) = \frac{16\rho}{\delta}\log\left(\frac{16\rho}{\delta}\right) + \frac{8}{\delta}\log C + \frac{8}{\delta}\log\frac{2}{\delta} + 2.$$

Lemma 19 *Let $\delta\in(0,1)$, ρ and C be constants. Then* **MWIS** *with parameters $(\varepsilon(\delta), K(\delta))$ computes an δ-approximation of MWIS of G in expectation, and its running time is $O(n)$.*

Proof. From Lemma 18, we obtain that our algorithm outputs a random independent set $\mathbf{x}$ such that

$$\mathbb{E}\left[\mathbf{w}^T\hat{\mathbf{x}}\right] \geq \mathbf{w}^T\mathbf{x}^*(1-\phi),$$

where $\phi = \varepsilon + P_K\left(\max_{v\in V}|\mathbf{B}(v,K)|\right)$.

For a graph G with growth ratio ρ and corresponding constant C, from the definition $K = K(\delta) = \frac{16\rho}{\delta}\log\left(\frac{16\rho}{\delta}\right) + \frac{8}{\delta}\log C + \frac{8}{\delta}\log\frac{2}{\delta} + 2$, and Lemma 5, we have

$$\phi = \varepsilon + P_K\left(\max_{v\in V}|\mathbf{B}(v,K)|\right) \leq 2\varepsilon.$$

Hence $\phi\leq 2\varepsilon = \delta$, which shows that our algorithm with parameter $(\varepsilon(\delta), K(\delta))$ generates independent set with average weight at least $(1-\delta)\mathbf{w}^T\mathbf{x}^*$.

To show that our algorithm runs in $O(n)$ time, note that by Definition 4 and Lemma 4, $(CK)^\rho$ is an upper bound on the size of each connected component obtained

by the **MWIS**. Then the running time for each connected component is at most $O\left(2^{(CK)^\rho}\right)$. Since K is a constant when ρ, C and δ are constants, the total running time of the **MWIS** is $O\left(n2^{(CK)^\rho}\right) = O(n)$. This completes the proof of Lemma 19.

Here we note that from the standard Markov's inequality, by running **MWIS** for $O(\log n)$ many times and taking the independent set with highest weight as the output, we can obtain an algorithm that outputs $\hat{\mathbf{x}}$ that satisfies

$$\mathbf{w}^T\hat{\mathbf{x}} \geq (1-\varepsilon)\mathbf{w}^T\mathbf{x}^*$$

with probability at least $1 - \frac{1}{n^h}$ for any constant h.

3.6 Proof of Theorem 10

To prove Theorem 10, we describe the algorithm first with some parameters the algorithm uses to compute its answer. Let t be an index ranging over integers, to be interpreted as slotted time. Define $q_i^j(t) \in \mathbb{R}$ as the 'packet mass' at node i destined for node d_j at time t (for $1 \leq i \leq n, 1 \leq j \leq m$). Define m 'routing matrices', each of dimension $n \times L$ with the j^{th} matrix, $\mathbf{R}^j$ defined as follows via its $(i,l)^{th}$ element ($1 \leq i \leq n$, and $1 \leq l \leq L$):

$$R_{i,l}^j = \begin{cases} -1 & \text{if } \alpha(l) = i, d_j \neq i \\ 1 & \text{if } \beta(l) = i, d_j \neq i \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

Define a 'weight matrix' at time t, $\mathcal{W}(t)$, of dimension $L \times m$ via its $(l,j)^{th}$ element ($1 \leq l \leq L$ and $1 \leq j \leq m$):

$$\mathcal{W}_l^j(t) = q_{\alpha(l)}^j(t) - q_{\beta(l)}^j(t). \tag{3.7}$$

In vector notation[1], we have for $1 \leq j \leq m$:

[1]Omitting a subscript for a previously defined scalar represents the corresponding column vector.

$$-\mathcal{W}^j(t)^T = \mathbf{q}^j(t)^T \mathbf{R}^j, \tag{3.8}$$

The weight vector of dimension L, $\mathbf{W}(t)$, is then defined with its l^{th} element (corresponding to link $l, 1 \leq l \leq L$) as

$$W_l(t) = \max_j \{\mathcal{W}_l^j(t)\}. \tag{3.9}$$

Finally, let the Maximum weight of the non interfering set of links be [2]

$$\mathcal{M}(t) = \max_{\mathbf{e} \in \mathcal{S}} \mathbf{e}.\mathbf{W}(t). \tag{3.10}$$

Let $\varepsilon-$**MWIS** be the given randomized ε-approximation algorithm of MWIS for the wireless network. The following is a direct consequence.

Property 11 *$\varepsilon-MWIS$ returns some $\hat{\mathbf{e}}(t) \in \mathcal{S}$ with the following property for each given t:*

$$\hat{\boldsymbol{e}}(t).\boldsymbol{W}(t) \geq (1-\varepsilon)\mathcal{M}(t). \tag{3.11}$$

The 'link activation matrix', $\mathcal{E}(t) \in \{0,1\}^{Lm}$, of dimension $L \times m$, will now be defined using the vector $\hat{\mathbf{e}}(t)$ obtained above for a fixed t. The $(l,j)^{th}$ element, $\mathbf{E}_l^j(t) = 1$ is to be interpreted as activating link l to transfer a unit packet mass corresponding to S-D pair j at the beginning of time slot $t+1$. Note that the MWIS approximation algorithm itself is oblivious to the various types of packets in the networks. So, we need to convert the set $\hat{\mathbf{e}}(t)$ into specific information on which class of S-D pair packets that it needs to serve, which will be accomplished while defining the link activation matrix below.

Definition 9 (Link Activation Matrix) *We write $\mathcal{E}(t) = [\boldsymbol{E}^1(t) \ldots \boldsymbol{E}^m(t)]$, and define the columns, $\boldsymbol{E}^j$'s in what follows. For $1 \leq j \leq m$, let:*

$$S^j = \{l : \hat{e}_l(t) = 1, W_l(t) = W_l^j(t) \text{ and } W_l^{j'}(t) < W_l(t), \forall j' < j\}.$$

[2] $\mathbf{a}.\mathbf{b}$ denotes the standard vector dot product of $\mathbf{a}$ and $\mathbf{b}$.

S^j's are all disjoint sets by definition and $\cup_{j=1}^{m} S^j$ is a subset of E with adjacency vector $\hat{\mathbf{e}}(t)$. $\boldsymbol{E}^j(t)$ is then defined to be the adjacency vector of some ***maximal subset of S^j that can be activated, subject to the following constraint:***

Property 12 (activation constraint) *The total number of activated links pointing out of node i in the activation set represented by $\boldsymbol{E}^j(t)$ is at most $q_i^j(t)$ for $1 \leq i \leq n$.*

Remark 13 *The above constraint is included to ensure that the queue sizes do not become negative because of activating too many links while having too less queue size at any given node. Because of this, the* **MWIS** *algorithm is supplied with positive weights, and the analysis below can assume that $q_i^j(t) \geq 0$, $\forall t$.*

Let $\mathbf{E}(t)$ be the net activation vector: $\mathbf{E}(t) = \sum_{j=1}^{m} \mathbf{E}^j(t)$. It has the following property:

Property 14 $\boldsymbol{W}(t).\boldsymbol{E}(t) \geq (1-\varepsilon)\mathcal{M}(t) - n^3$.

Proof. $\mathbf{W}(t).\mathbf{E}(t) = \mathbf{W}(t).\hat{\mathbf{e}}(t) - \mathbf{W}(t).(\hat{\mathbf{e}}(t) - \mathbf{E}(t)) \geq (1-\varepsilon)\mathcal{M}(t) - nL$ (the bound for the first term follows from Property 11, and for the second term since for any $1 \leq l \leq L$, $\hat{e}_l(t) - E_l(t) = 1 \Rightarrow W_l(t) \leq n$ based on the activation constraint above.)

Now, figure 3-2 describes the actual queue computations performed by the algorithm.

We'll now model the process specified above. Towards this, define m 'arrival vectors', $\mathbf{a}^j$ for $1 \leq j \leq m$, each of dimension L as (corresponding to step 1):

$$a_i^j = \begin{cases} r_j & \text{if } i = s_j \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

The queue dynamics then follows:

$$\mathbf{q}^j(t+1) = \mathbf{q}^j(t) + \mathbf{R}^j\mathbf{E}^j(t) + \mathbf{a}^j. \tag{3.13}$$

Queue computations

- Initialize all the mn queues, $q_i^j, 1 \leq i \leq n, 1 \leq j \leq m$ to zero mass at $t = 0$. Subsequently, at each discrete time slot, do the following:

(1) Add (a 'packet mass' of) r_k to $q_{s_k}^k$.

(2) Compute the weight matrix, $\mathbf{W}(t)$ via equations 3.7, 3.9.

(3) Invoke the $\varepsilon-$**MWIS** algorithm with weights $\mathbf{W}(t)$, which results in $\hat{\mathbf{e}}(t)$ satisfying Property 11.

(4) Decide the link activation matrix, $\mathcal{E}(t) \in \{0,1\}^{Lm}$ using the specification in definition 9 (such that it satisfies Property 12)

(5) For each activated link, l, with $\mathbf{E}_l^j(t) = 1$, move a unit queue mass from $q_{\alpha(l)}^j$ to $q_{\beta(l)}^j$. In other words, make the following updates:

$$q_{\alpha(l)}^j(t+1) = q_{\alpha(l)}^j(t+1) - 1$$

and

$$q_{\beta(l)}^j(t+1) = q_{\beta(l)}^j(t) + 1$$

Figure 3-2: Queue computations done by the algorithm

Let, the maximum queue size observed across the network at time t be:

$$q^{max}(t) = \max_{(i,j)} q_i^j(t).$$

We can then prove the following Lemma:

Lemma 20 *If* $(1+2\varepsilon)\boldsymbol{r} \in \mathcal{F}$, *then* $q^{max}(t) \leq \frac{10}{\varepsilon} n^{7.5}$ *for all* $t > 0$ *and* $0 < \varepsilon < 1/4$.

Proof. Consider the standard quadratic potential function $V(t) = \sum_{i,j} (q_i^j(t))^2$. Then,

$$\begin{aligned}\Delta V(t) &= V(t+1) - V(t) \\ &= \sum_j (\mathbf{R}^j\mathbf{E}^j(t) + \mathbf{a}^j).(\mathbf{R}^j\mathbf{E}^j(t) + \mathbf{a}^j + 2\mathbf{q}^j(t))\end{aligned}$$

$$
\begin{aligned}
&= \sum_j (\mathbf{R}^j\mathbf{E}^j(t) + \mathbf{a}^j).(\mathbf{R}^j\mathbf{E}^j(t) + \mathbf{a}^j) \\
&+2\sum_j \mathbf{q}^j(t).(\mathbf{R}^j\mathbf{E}^j(t) + \mathbf{a}^j) \\
&\leq (n+1)^2 m + 2\left(\sum_j \mathbf{q}^j(t).\mathbf{R}^j\mathbf{E}^j(t) + \sum_j \mathbf{q}^j(t).\mathbf{a}^j\right) \\
&\leq 2n^4 + 2(A+B),
\end{aligned}
$$

where, we now bound the terms A and B, starting with A first:

$$
\begin{aligned}
A &= \sum_{j=1}^{m} (\mathbf{q}^j(t))^T \mathbf{R}^j\mathbf{E}^j(t) = -\sum_{j=1}^{m} \mathcal{W}^j(t)^T\mathbf{E}^j(t) \\
&\overset{(a)}{=} -\sum_{j=1}^{m} \mathbf{W}(t)^T\mathbf{E}^j(t) \\
&\overset{(b)}{=} -\mathbf{W}(t).\mathbf{E}(t) \leq -(1-\varepsilon)\mathcal{M}(t) + n^3.
\end{aligned}
$$

Here (a) holds since $\mathbf{E}_l^j \neq 0 \Rightarrow \mathbf{W}_l = \mathcal{W}_l^j$, and (b) holds from Proposition 14.

Now we bound term B. Note that a 'flow vector', $\mathbf{f}^j \in \mathbb{R}^L$ routes a flow r_j for the S-D pair j, if the following holds:

$$
\mathbf{a}^j = -\mathbf{R}^j\mathbf{f}^j \text{ for } 1 \leq j \leq m. \tag{3.14}
$$

Let $(\mathbf{f}^1, \ldots \mathbf{f}^m)$ 'route' $\mathbf{r}$ for the m S-D pairs. The net flow on the links is given by:

$$
\hat{\mathbf{f}} = \sum_{j=1}^{m} \mathbf{f}^j.
$$

Claim 3 *If $(1+2\varepsilon)\boldsymbol{r} \in \mathcal{F}$, then there exist flows $(\boldsymbol{f}^1, \ldots, \boldsymbol{f}^n)$ that route $\boldsymbol{r}$ such that for the net flow, $\hat{\boldsymbol{f}}$, the following relation holds: $(1+2\varepsilon)\hat{\boldsymbol{f}} \in co(\mathcal{S})$.*

Proof. Since, $(1+2\varepsilon)\mathbf{r} \in \mathcal{F}$, there exist flows $(\mathbf{g}^1, \ldots \mathbf{g}^m)$ that route $(1+2\varepsilon)\mathbf{r}$ with $\hat{\mathbf{g}} \in co(S)$. Define $\mathbf{f}^j = \frac{1}{1+2\varepsilon}\mathbf{g}^j$. Since $\mathbf{a}^j$'s are linear in $\mathbf{r}$ (eq. 3.12), we see (from eq. 3.14) that, if $(\mathbf{g}^1, \ldots \mathbf{g}^m)$ route $(1+2\varepsilon)\mathbf{r}$, then $(\mathbf{f}^1, \ldots \mathbf{f}^m) = \frac{1}{1+2\varepsilon}(\mathbf{g}^1, \ldots \mathbf{g}^m)$ route $\mathbf{r}$. Also, from linearity of $\hat{\mathbf{f}}$ in terms of $(\mathbf{f}^1, \ldots, \mathbf{f}^m)$, it follows that $(1+2\varepsilon)\hat{\mathbf{f}} = \hat{\mathbf{g}} \in co(\mathcal{S})$.

Now, assuming that $(1+2\varepsilon)\mathbf{r} \in \mathcal{F}$,

$$\begin{aligned} B &= \sum_j \mathbf{q}^j(t).\mathbf{a}^j \\ &= -\sum_j (\mathbf{q}^j(t))^T \mathbf{R}^j \mathbf{f}^j \left(\text{ where } (1+2\varepsilon)\hat{\mathbf{f}} \in co(\mathcal{S}) \right) \\ &= \sum_j \mathcal{W}^j(t)^T \mathbf{f}^j \leq \mathbf{W}(t)^T \sum_j \mathbf{f}^j = \mathbf{W}(t).\hat{\mathbf{f}}. \end{aligned}$$

Since $(1+2\varepsilon)\hat{\mathbf{f}} \in co(\mathcal{S})$, let

$$(1+2\varepsilon)\hat{\mathbf{f}} = \sum_{i=1}^{|\mathcal{S}|} \lambda_i \mathbf{c}_i \text{ where each } \mathbf{c}_i \in \mathcal{S} \tag{3.15}$$

for some non negative λ_i such that $\sum_i \lambda_i \leq 1$. Then,

$$\begin{aligned} B &\leq \frac{1}{1+2\varepsilon} \sum_{i=1}^{|\mathcal{S}|} \lambda_i \mathbf{W}(t).\mathbf{c}_i \\ &\leq \frac{1}{1+2\varepsilon} \mathcal{M}(t). \end{aligned}$$

Therefore, we have the following bound on $\Delta V(t)$:

$$\begin{aligned} \Delta V(t) &\leq 2n^4 + 2\left(n^3 + \mathcal{M}(t)(\frac{1}{1+2\varepsilon} - (1-\varepsilon)) \right) \\ &\leq 3n^4 - \frac{\varepsilon}{3}\mathcal{M}(t) \text{ for } 0 < \varepsilon < 1/4. \end{aligned}$$

Next note that we could assume without loss of generality that the network graph is connected. If this is not the case, we can always analyze the capacity regions of each connected component separately. Upon this, we can get the simple bound that:

$$\mathcal{M}(t) \geq \frac{1}{L}\sqrt{\frac{V(t)}{mn}} \geq \frac{\sqrt{V(t)}}{n^{3.5}}.$$

To see the above bound, let $(a,b) = \arg max_{(i,j)} q_i^j$. Then $q_a^b \geq \sqrt{\frac{V(t)}{mn}}$ and since the network is connected, on any path from a to d_b (of length at most L), there exists at

least one link, l such that $\mathcal{W}_l^j(t) = q_{\beta(l)}^j(t) - q_{\alpha(l)}^j(t) \geq \frac{1}{L} q_a^b \geq \frac{1}{L}\sqrt{\frac{V(t)}{mn}}$, which is clearly a lower bound for $\mathcal{M}(t)$.

Therefore, we have the following bound

$$\Delta V(t) \leq 3n^4 - \frac{\varepsilon}{3}\frac{\sqrt{V(t)}}{n^{3.5}} \tag{3.16}$$

which in turn implies that

$$V(t) \leq 3n^4 + \left(\frac{9n^{7.5}}{\varepsilon}\right)^2 \leq \frac{100}{\varepsilon^2} n^{15} \quad \forall t > 0.$$

Hence,

$$\max_{(i,j)} q_i^j(t) \leq \frac{10}{\varepsilon} n^{7.5} \quad \forall t > 0.$$

Lemma 21 *If $(1-2\varepsilon)\boldsymbol{r} \notin \mathcal{F}$, then $q^{max}(t) \geq \frac{\varepsilon^2}{n^4} t$ for all $t \geq 0$.*

Proof. First, we begin by showing the following claim:

Claim 4 *if $(1-2\varepsilon)\boldsymbol{r} \notin \mathcal{F}$, then for any given m link rate vectors, $(\boldsymbol{g}^1, \ldots \boldsymbol{g}^m)$ with $\hat{\boldsymbol{g}} := \sum_j \boldsymbol{g}^j \in co(\mathcal{S})$, there is $j \in [m]$ such that the followings hold:*

(a) The graph with edge capacities given by $\boldsymbol{g}^j$ has a maximum flow of value at most $(1-\varepsilon)r_j$ from s_j to d_j.

(b) $r_j > \frac{\varepsilon}{n^3}$.

Proof. Suppose that Claim 4 is not true. Then, let

$$\mathbf{i} = \{i \in [m] | r_i \geq \frac{\varepsilon}{n^3}\}.$$

By the assumption, for all $i \in \mathbf{i}$, there exists a flow of value at least $(1-\varepsilon)r_i$ from s_i to d_i for edge capacities defined according to $\mathbf{g}^i$. Now consider m link rate vectors, $(\mathbf{h}^1, \ldots \mathbf{h}^m)$ with $\mathbf{h}^i \in \mathbb{R}^L$ defined below. Let R_i be some fixed arbitrary path from s_i to d_i:

$$\mathbf{h}_l^i = \begin{cases} (1-\varepsilon)\mathbf{g}_l^i & \text{if } i \in \mathbf{i}, l \in E \\ \frac{\varepsilon}{n^3} & \text{if } i \notin \mathbf{i} \text{ and } l \in R_i \\ 0 & \text{otherwise} \end{cases} \tag{3.17}$$

That is, for $i \in \mathbf{i}$, $\mathbf{h}^i = (1-\varepsilon)\mathbf{g}^i$ and otherwise, the value of $\mathbf{h}^i$ on all the edges in the path R_i is equal to $\frac{\varepsilon}{n^3}$, while the value on any edge not in R_i is defined to be 0. We will now argue that the net link rate, $\hat{\mathbf{h}} = \sum_j \mathbf{h}^j \in co(S)$ by producing a schedule for $\hat{\mathbf{h}}$ of unit length.

Note that for $i \in [m] \cap \mathbf{i}^c$, $\mathbf{h}^i$ can be scheduled under any interference constraint using $\frac{\varepsilon}{n^3} \times n$ amount of time, as there are at most n links in the path R_i. Thus, the link rate vector, $\sum_{i \in [m] \cap \mathbf{i}^c} \mathbf{h}^i$ can be scheduled in at most $m \times \frac{\varepsilon}{n^2} \leq \varepsilon$ time.

Next, $\sum_{i \in \mathbf{i}} \mathbf{h}^i = (1-\varepsilon) \sum_{i \in \mathbf{i}} \mathbf{g}^i \leq (1-\varepsilon) \sum_{i \in [m]} \mathbf{g}^i = (1-\varepsilon)\hat{\mathbf{g}}$. Since, $\hat{\mathbf{g}} \in co(\mathcal{S})$, this implies that $\sum_{i \in \mathbf{i}} \mathbf{h}^i$ can be scheduled in a total of $(1-\varepsilon)$ time.

Hence, $\hat{\mathbf{h}} = \sum_{i \in [m]} \mathbf{h}^i = \sum_{i \in \mathbf{i}} \mathbf{h}^i + \sum_{i \in [m] \cap \mathbf{i}^c} \mathbf{h}^i$ can be scheduled in $(1-\varepsilon) + \varepsilon$, which is unit time and thus we have $\hat{\mathbf{h}} \in co(\mathcal{S})$.

Now, consider the graph with edge capacities $\mathbf{h}^i$. If $i \in \mathbf{i}$, the max flow from s_i to d_i is at least $(1-\varepsilon)^2 r_i \geq (1-2\varepsilon) r_i$ by the definition of $\mathbf{h}^i$. Else, if $i \in [m] \cap \mathbf{i}^c$, then the max flow is again at least $\frac{\varepsilon}{n^3}$, which is bigger than $(1-2\varepsilon)r_i$ by the definition of $\mathbf{i}$. This implies that for each $i \in [m]$, there exists a flow ϕ^i which routes at least $(1-2\varepsilon)r_i$ from s_i to d_i, while satisfying (componentwise), $\phi^i \leq \mathbf{h}^i$. Thus, the vector of flows, $(\phi_1, \ldots, \phi_m)$ routes $(1-2\varepsilon)\mathbf{r}$ with the net flow, $\hat{\phi} \leq \hat{\mathbf{h}} \in co(\mathcal{S})$ Hence, $(1-2\varepsilon)\mathbf{r} \in \mathcal{F}$, contradicting our assumption and we obtain Claim 4.

Now, for any $t > 0$, define $(\mathbf{g}^1(t), \ldots \mathbf{g}^m(t))$, as the link rates for each packet type obtained by considering the actual schedules. More precisely, For $j \in [m]$, define

$$\mathbf{g}_l^j(t) = \frac{1}{t}(|\{t : \mathbf{E}_l^j(t) = 1\}|) \tag{3.18}$$

Clearly, $\hat{\mathbf{g}}(t) \in co(\mathcal{S})$, and we can apply Claim 4 to it. Thus, there exists $j \in [m]$ such that $r_j > \frac{\varepsilon}{n^3}$ and the max flow from s_j to d_j is at most $(1-\varepsilon)r_j$ in the graph

with edge capacities $\mathbf{g}^j(t)$. Applying the max-flow min-cut theorem, observe that there is a cut (S,T) of the vertex set V such that $s_j \in S$, $t_j \in T$ and $c(S,T) = \sum_{l \in E:\alpha(l) \in S, \beta(l) \in T} \mathbf{g}_l^j(t)$ is equal to the max flow from s_j to t_j, which is at most $(1-\varepsilon)r_j$.

From equation 3.18, observe that the total amount of packet mass of type j that was moved from S to T during the t time slots is at most $\sum_{l \in E:\alpha(l) \in S, \beta(l) \in T} t \times \mathbf{g}_l^j(t)$, which is at most $t \times (1-\varepsilon)r_j$. Since the amount of packets of type j that were added during these time slots is $t \times r_i$, we have:

$$\sum_{i \in S} q_i^j(t) \geq t.r_j - t.(1-\varepsilon)r_j.$$

Therefore,

$$\max_{(i,j)} q_i^j(t) \geq \frac{\sum_{i \in S} q_i^j(t)}{|S|} \geq \frac{\varepsilon \cdot t \cdot r_j}{n} \geq \frac{\varepsilon^2}{n^4} t.$$

Theorem 10 is now a direct consequence of Lemmas 20 and 21 since the max queue size grows at least linearly with time if $(1-2\varepsilon)\mathbf{r} \notin \mathcal{F}$ and is polynomially bounded if $(1+2\varepsilon)\mathbf{r} \in \mathcal{F}$, so the two cases can be clearly distinguished in the worst case before $t = \frac{10}{\varepsilon^3} n^{11.5}$ time slots of simulation. Note that the Maximum queue size can be spread across the network in a distributed manner easily. Further, the queue computation updates are also essentially distributed computations. Hence, the feasibility test is completely distributed.

3.7 Algorithm

Combining the above Lemmas, we have the following results based on the Maximum queue size observed at each time upon simulating the virtual queue computations using approximate max weight scheduling and routing described.

Let event E_1 be defined as observing

$$q^{max}(t) > \frac{10}{\varepsilon} n^{7.5} \text{ for some } t > 0.$$

Similarly, define event E_2 as observing

$$q^{max}(t) < \frac{\varepsilon^2}{n^4} t \text{ for some } t > 0.$$

We run the algorithm till a time $\hat{T}$ where:

$$\hat{T} = \min_t \{E_1 \text{ or } E_2 \text{ occurs}\}.$$

Note that either E_1 or E_2 has to occur eventually (in the worst case, before $t = \frac{10}{\varepsilon} n^{11.5}$ by definition of E_1 and E_2, so $\hat{T}$ is clearly polynomial)

We can then declare the following $2\varepsilon-$approximate statements (for arbitrarily small $\varepsilon > 0$,) on the membership of $\mathbf{r}$ in $\mathcal{F}$ by observing $q^{max}(\hat{T})$.

- If E_1, then declare $(1 + 2\varepsilon)\mathbf{r} \notin \mathcal{F}$.
- If E_2, then declare $(1 - 2\varepsilon)\mathbf{r} \in \mathcal{F}$.

The consistency of the above statements is a direct consequence of the definitions of E_1, E_2 and $\hat{T}$. Note that it is also possible that both E_1 and E_2 hold simultaneously without any contradiction, which just means that $\mathbf{r}$ is within an $1 \pm 2\varepsilon$ factor close to the boundary of the capacity region.

Alternately, one may not have any ε pre-specified to begin with and the interest is simply in making the best possible approximate statement after running the algorithm for a certain amount of time. We also have such a possibility resulting from the above analysis:

Towards this, define:

$$\varepsilon(t) = 2 \min \left(n^2 \sqrt{\frac{q^{max}(t)}{t}}, \frac{10n^{7.5}}{q^{max}(t)} \right)$$

Then, the above discussion implies that whenever $\varepsilon(t) < 1/2$, one can correctly declare the feasibility of a rate vector that is a $1 \pm \varepsilon(t)$ factor of the given vector.

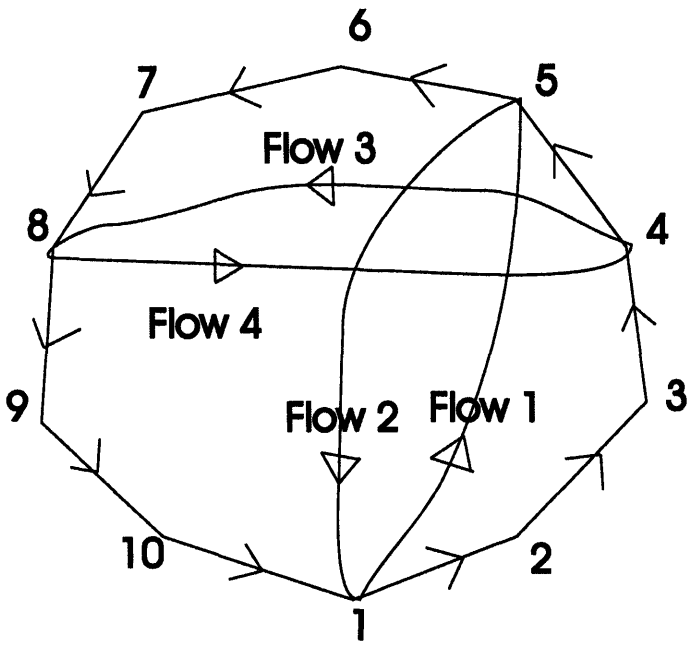

Figure 3-3: An illustration of the cyclical network with 2-hop interference.

3.8 Numerical experiment

The analysis guarantees that given any $\varepsilon > 0$, we will have $\varepsilon(t) < \varepsilon$ for a polynomially bounded t. We simulated the algorithm on a directed cyclical network of 10 nodes with 2-hop interference constraints, shown in Figure 3.8. In the network, 4 different rate vectors with coordinates corresponding to flows between nodes $1 \rightarrow 5, 5 \rightarrow 1, 4 \rightarrow 8, 8 \rightarrow 4$ were considered. We did simulation by using standard software for solving Integer Linear Programs for the approximate MWIS. We note that it solves the given MWIS exactly. There were assumed to be 4 flows as explained above. Figure 3.8 describes the maximum queue size $q^{max}(t)$ over time t on the network described in Figure 3.8 for 6 different rate vectors given by $r_1 = [0.2\ 0.1\ 0.2\ 0.1], r_2 = [0.23\ 0.13\ 0.2\ 0.1], r_3 = [0.1\ 0.15\ 0.1\ 0.15], r_4 = [0.1\ 0.16\ 0.1\ 0.16], r_5 = [0.1\ 0.17\ 0.1\ 0.17]$, and $r_6 = [0.1\ 0.18\ 0.1\ 0.18]$. As we can see, q^{max} grows roughly linearly for r_2, r_5, and r_6 whereas it stabilizes fairly quickly at around t=500 for r_1, r_3 and r_4. While our proofs give precise bounds and guarantees regarding polynomial time distinguishability, these experimental plots suggest that

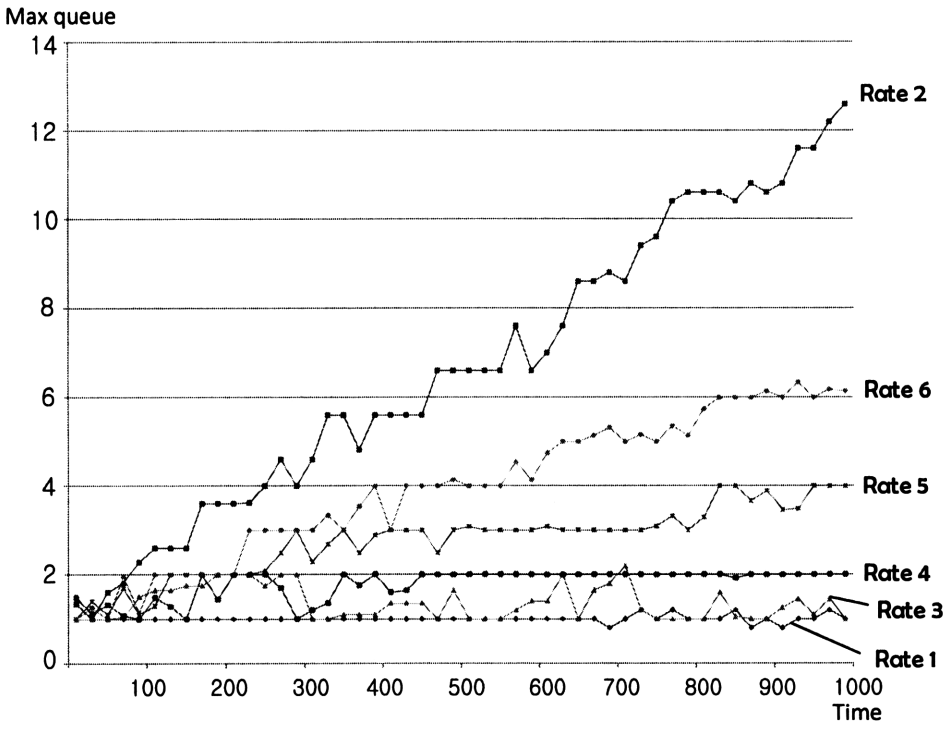

Figure 3-4: A plot of $q^{max}(t)$ versus t on the network in Figure 3.8 for 6 different rate vectors.

in practice they are likely to be distinguished fairly quickly.

Chapter 4

Application of Partition Function : Stochastic Loss Network

The question of allocating common resources for various entities accessing them, is central to many applications arising in diverse scenarios including telephone networks, broadband telecommunication networks, ATM networks, and workforce management. In such applications, the canonical dynamic model is the stochastic loss network. The key algorithmic question in this setup concerns evaluation of what is called "loss probability", or the inability to serve a request. This algorithmic question is equivalent to computation of the partition function of the corresponding MRF model for the stochastic loss network as described in Section 1.3.3.

Historically, a simple mechanism known as Erlang fixed-point approximation has been extensively utilized for application in telephone networks since early 1960's [29,30]. This approximation works very well when network is "underloaded" or "overloaded". However, we prove that when it is critically loaded, it performs very poorly. Indeed, in most applications one would like to operate a network near its capacity, i.e. under critical loading. Motivated by this, we also provide a novel approximation method which we call the "slice method" after its geometrical interpretation. We establish asymptotic correctness of our approach along with a exponentially fast convergence property.

4.1 Stochastic loss network

As the complexities of computer and communication systems continue to grow at a rapid pace, performance modeling, analysis and optimization are playing an increasingly important role in the design and implementation of such complex systems. Central to these research studies are the models of the various applications of interest.

For over decades, stochastic loss networks have been widely investigated as models of diverse computer and communication systems in which different types of resources are used to serve various classes of customers involving simultaneous resource possession and non-backlogging workloads. Such examples include telephone networks, mobile cellular systems, broadband telecommunication networks, ATM networks, optical wavelength-division multiplexing networks, wireless networks, distributed computing, database systems, data centers, and multi-item inventory systems [21, 28, 30, 31, 44, 45, 53, 54, 68, 69].

Loss networks also have even been used recently for resource planning within the context of workforce management in the information technology services industry, where a collection of IT service products are offered each requiring a set of resources with certain capabilities [4, 39].

4.2 Previous works

One of the most important objectives in analyzing such loss networks is to determine performance measures of interest, most notably the stationary loss probability for each customer class. The classical Erlang formula, which has been thoroughly studied and widely applied in many fields of research, provides the probabilistic characterization of these loss probabilities. More specifically, given a stochastic network and a multiclass customer workload, the formula renders the stationary probability that a customer will be lost due to insufficient capacity for at least one resource type. While the initial results of Erlang [8] were for the particular case of Poisson arrivals and exponential service times, Sevastyanov [56] demonstrates that the Erlang formula

holds under general finite-mean distributions for the customer service times. The results are also known to hold in the presence of dependencies among service times for a specific class [7]. Recent results [5] suggest that relaxations can be made to the customer arrival process, merely requiring that customers generate sessions according to a Poisson process and, within each session, blocked customers may retry with a fixed probability after an idle period of random length. A multi-period version of the Erlang loss model has also been recently studied [4].

Unfortunately, the computational complexity of the exact Erlang formula and related measures is known to be $\sharp P$-complete in the size of the network [38], thus rendering the exact formula of limited use for many networks in practice. The well-known Erlang fixed-point approximation has been developed to address this problem of complexity through a set of product-form expressions for the blocking probabilities of the individual resources that map the blocking probability of each resource to the blocking probabilities of other resources. In other words, it is as if customer losses are caused by independent blocking events on each of the resources used by the customer class based on the one-dimensional Erlang function. The Erlang fixed-point approximation has been frequently used and extensively studied as a tractable approach for calculating performance measures associated with the stochastic loss network, including estimates for stationary loss probabilities. Moreover, the Erlang fixed-point approximation has been shown to be asymptotically exact in two limiting regimes, one based on increasing the traffic intensities and resource capacities in a proportional manner [29,30], and the other based on increasing the number of resource types and number of customer classes [68, 72].

4.3 Our contributions

We make several important contributions in our work. Despite being asymptotically exact in certain limiting regimes, it is equally well known that the Erlang fixed-point approximation can provide relatively poor performance estimates in various cases. The stochastic loss networks that model many computer and communication systems

often operate naturally in a so-called "critically loaded" regime [18]. Somewhat surprisingly, first we find that, even though the Erlang fixed-point approximation can perform quite well in underloaded and overloaded conditions, the fixed-point approximation can provide relatively poor loss probability estimates when the network is critically loaded. We establish such qualitative results by means of estimating the convergence rate of the Erlang fixed-point approximation toward the exact solution under large network scalings. This motivates the need to design better algorithms for estimating loss probabilities.

Then we propose a novel algorithm for computing the stationary loss probabilities in stochastic loss networks, which we call the "slice method" because the algorithm exploits structural properties of the exact stationary distribution along "slices" of the polytope over which it is defined. Our algorithm is shown to always converge and to do so exponentially fast. Through a variational characterization of the stationary distribution, we establish that the results from our algorithm are asymptotically exact. We further estimate the convergence rate of our algorithm, where comparisons between the convergence rates of the Erlang fixed-point approximation and the slice method favors our approach.

Using this variational characterization, we also provide an alternative proof of the main Theorem in [29], which is much simpler and may be of interest in its own right. A collection of numerical experiments, especially for real world applications to IT services industry, further investigates the effectiveness of our algorithm where it convincingly outperforms the Erlang fixed-point approximation for loss networks in the critically loaded regime.

4.4 Setup

In this Section, we explain the mathematical model and the problem of interest. In our analysis of algorithms, we consider a scaling of the stochastic loss network to model the type of large networks that arise in various applications. In Section 4.4.3, we explain the scaled version of the network.

4.4.1 Model

We investigate general stochastic loss networks with fixed routing, using the standard terminology in the literature based on routes (customer classes) and links (resource types); see, e.g., [31]. Consider a network with J links, labeled $1, 2, \ldots, J$. Each link j has C_j units of capacity. There is a set of K distinct (pre-determined) routes, denoted by $\mathcal{R} = \{1, \ldots, K\}$. A call on route r requires A_{jr} units of capacity on link j, $A_{jr} \geq 0$. Calls on route r arrive according to an independent Poisson process of rate ν_r, with $\underline{\nu} = (\nu_1, \ldots, \nu_K)$ denoting the vector of these rates. The dynamics of the network are such that an arriving call on route r is admitted to the network if sufficient capacity is available on all links used by route r; else, the call is dropped. To simplify the exposition, we will assume that the call service times are i.i.d. exponential random variables with unit mean. However, our results are true under general service time distributions due to the well-known *insensitivity property* [63] of this class of stationary loss networks.

Let $\underline{n}(t) = (n_1(t), \ldots, n_K(t)) \in \mathbb{N}^K$ be the vector of the number of active calls in the network at time t. By definition, we have that $\underline{n}(t) \in \mathcal{S}(C)$ where

$$\mathcal{S}(C) = \left\{ \underline{n} \in \mathbb{Z}^K : \underline{n} \geq 0,\ A\underline{n} \leq \underline{C} \right\},$$

and $\underline{C} = (C_1, \ldots, C_J)$ denotes the vector of link capacities. Within this framework, the network is Markov with respect to state $\underline{n}(t)$. It has been well established that the network is a *reversible* multidimensional Markov process with a product-form stationary distribution [27]. Namely, there is a unique stationary distribution π on the state space $\mathcal{S}(C)$ such that for $\underline{n} \in \mathcal{S}(C)$,

$$\pi(\underline{n}) = G(C)^{-1} \prod_{r \in \mathcal{R}} \frac{\nu_r^{n_r}}{n_r!}, \tag{4.1}$$

where $G(C)$ is the partition function

$$G(C) = \sum_{\underline{n} \in \mathcal{S}(C)} \prod_{r \in \mathcal{R}} \frac{\nu_r^{n_r}}{n_r!}.$$

Let M be an upper bound on the number of active route calls for all routes. Then the distribution (4.1) can be understood as the following Markov random field defined on the factor graph of π for $\underline{n} \in M^K$.

$$\pi(\underline{n}) \propto \prod_{r \in \mathcal{R}} \frac{\nu_r^{n_r}}{n_r!} \prod_{j=1}^{J} g_j(n_{\mathcal{R}_j}), \tag{4.2}$$

where $\mathcal{R}_j$ is the set of routes that uses link j, and $g_j(\cdot)$ is defined as

$$g_j(n_{\mathcal{R}_j}) = \begin{cases} 1 & \text{if } \sum_{r \in \mathcal{R}_j} A_{jr} n_r \leq C_j \\ 0 & \text{otherwise} \end{cases}.$$

4.4.2 Problem

A primary performance measure in loss networks is the per-route stationary loss probability, the fraction of calls on route r in equilibrium that are dropped or lost, denoted by L_r. It can be easily verified that L_r is well-defined in the above model. This model can be thought of as a stable system where admitted calls experience an average delay of 1 (their service requirement) and lost calls experience a delay of 0 (their immediate departure). Therefore, the average delay experienced by calls on route r is given by

$$D_r = (1 - L_r) \times 1 + L_r \times 0 \; = \; (1 - L_r).$$

Upon applying Little's law [37] to this stable system (with respect to route r), we obtain

$$\nu_r D_r = \mathbb{E}[n_r]$$

which yields

$$1 - L_r = \frac{\mathbb{E}[n_r]}{\nu_r}. \tag{4.3}$$

Thus, computing L_r is equivalent to computing the expected value of the number of active calls on route r with respect to the stationary distribution of the network.

Even though we have an explicit formula, the computational complexity of the exact stationary distribution, known to be $\sharp P$-complete in general [38], renders its direct use of limited value in practice. Also notice that from the definition of the expectation of n_r over the stationary distribution (4.1),

$$\mathbb{E}[n_r] = \frac{\sum_{\underline{n}\in\mathcal{S}(C)} \prod_{s\in\mathcal{R}} n_r \frac{\nu_s^{n_s}}{n_s!}}{\sum_{\underline{n}\in\mathcal{S}(C)} \prod_{s\in\mathcal{R}} \frac{\nu_s^{n_s}}{n_s!}}.$$

Hence the computation of the $\mathbb{E}[n_r]$ is equivalent to computing the partition function of the corresponding MRF given by (4.2).

By exploiting these structural properties, we will design a family of simple iterative algorithms for approximate computation of the stationary loss probability that also have provably good accuracy properties.

4.4.3 Scaling

We consider a scaling of the stochastic loss network to model the type of large networks that arise in various applications. Although it has been well studied (see, e.g., [29]), we will use this scaling both to evaluate analytically the performance of different approximation algorithms for computing loss probabilities and to obtain the capacity region for satisfying a set of loss probabilities.

Given a stochastic loss network with parameters $\underline{C}, A$ and $\underline{\nu}$, a scaled version of the system is defined by the scaled capacities

$$\underline{C}_N = N\underline{C} = (NC_1, \ldots, NC_K)$$

and the scaled arrival rates

$$\underline{\nu}_N = N\underline{\nu} = (N\nu_1, \ldots, N\nu_K),$$

where $N \in \mathbb{N}$ is the system scaling parameter. The corresponding feasible region of

calls is given by $\mathcal{S}(N\underline{C})$. Now consider a normalized version of this region defined as

$$\mathcal{S}_N(C) = \left\{ \frac{1}{N}\underline{n} : \underline{n} \in \mathcal{S}(N\underline{C}) \right\}.$$

Then the following *continuous approximation* of $\mathcal{S}_N(C)$ emerges in the large N limit:

$$\bar{\mathcal{S}}(C) = \{\underline{x} : A\underline{x} \leq \underline{C},\ \underline{x} \in \mathbb{R}_+^K\}.$$

4.5 Algorithms

We now describe three algorithms for computing the stationary loss probabilities $\underline{L} = (L_r) \in [0,1]^K$. The well-known Erlang fixed-point approximation is presented first, followed by a "1-point approximation" based on the concentration of the stationary distribution around its mode in large networks. The third algorithm is our new family of "slice methods" that attempts to compute the average number of active calls on different routes via an efficient exploration through "slices" of the admissible polytope $\mathcal{S}(C)$.

4.5.1 Erlang fixed-point approximation

The well-known Erlang formula [8] for a single-link, single-route network with capacity C and arrival rate ν states that the loss probability, denoted by $E(\nu, C)$, is given by

$$E(\nu, C) = \frac{\nu^C}{C!} \left[\sum_{i=0}^{C} \frac{\nu^i}{i!} \right]^{-1}.$$

Based on this simple formula, the Erlang fixed-point approximation for multi-link, multi-route networks arose from the hypothesis that calls are lost due to *independent* blocking events on each link in the route. More formally, this hypothesis implies that the loss probabilities of routes $\underline{L} = (L_1, \ldots, L_K)$ and blocking probabilities of links

Erlang fixed-point approximation

(1) Denote by t the iteration of the algorithm, with $t = 0$ initially. Start with $E_j^{(0)} = 0.5$ for all $1 \leq j \leq J$.

(2) In iteration $t+1$, update $E_j^{(t+1)}$ according to

$$E_j^{(t+1)} = E(\rho_j^{(t)}, C_j),$$

where

$$\rho_j^{(t)} = (1 - E_j^{(t)})^{-1} \sum_{r: j \in r} \nu_r A_{jr} \prod_{i: i \in r} (1 - E_i^{(t)})^{A_{ir}}.$$

(3) Upon convergence per appropriate stopping conditions, denote the resulting values by $E_j^{\mathcal{E}}$ for $1 \leq j \leq J$. Compute the loss probabilities from the Erlang fixed-point approximation, $L_r^{\mathcal{E}}$, $r \in \mathcal{R}$, as

$$1 - L_r^{\mathcal{E}} = \prod_j (1 - E_j^{\mathcal{E}})^{A_{jr}}.$$

Figure 4-1: An iterative algorithm to obtain a solution to the fixed-point equations (4.4).

$\underline{E} = (E_1, \ldots, E_J)$ satisfy the set of *fixed-point* equations

$$\begin{aligned} E_j &= E(\rho_j, C_j), \\ \rho_j &= \frac{1}{1 - E_j} \left[\sum_r \nu_r A_{jr} \prod_i (1 - E_i)^{A_{ir}} \right], \\ 1 - L_r &= \prod_j (1 - E_j)^{A_{jr}}, \end{aligned} \tag{4.4}$$

for $j = 1, \ldots, J$ and $r \in \mathcal{R}$.

Figure 4-1 describes a natural iterative algorithm that attempts to obtain a solution to the above fixed-point equations.

4.5.2 1-point approximation

Kelly [29] established the asymptotic exactness of the Erlang fixed-point approximation in a large network limiting regime by showing that the stationary distribution

1-point approximation

(1) Denote by t the iteration of the algorithm, with $t = 0$ initially. Start with $z_j^{(0)} = 0.5$ for all $1 \leq j \leq J$.

(2) In iteration $t+1$, determine $\underline{z}^{(t+1)}$ as follows:

(a) Choose coordinates from $1, \ldots, J$ in a round-robin manner.

(b) Update $z_j^{(t+1)}$ by solving the equation

$$g_j^{(t)}(x) = \min\left\{C_j, g_j^{(t)}(1)\right\},$$

where $g_j^{(t)}(x) = \sum_r A_{jr}\nu_r \prod_i z_i^{A_{ir}}$ with

$$z_i = \begin{cases} z_i^{(t+1)} & \text{for } i < j, \\ x & \text{for } i = j, \\ z_i^{(t)} & \text{for } i > j. \end{cases}$$

Thus, $g_j^{(t)}(x)$ is the evaluation of part of the function $g(\cdot)$ corresponding to the j^{th} coordinate with values of components $< j$ being from iteration $t+1$, values of components $> j$ from iteration t, and component j being the variable itself.

(3) Upon convergence per appropriate stopping conditions, denote the resulting values by z_j^* for $1 \leq j \leq J$. Compute the loss probabilities from the 1-point approximation, L_r^*, $r \in \mathcal{R}$, as

$$1 - L_r^* = \prod_j (z_j^*)^{A_{jr}}.$$

Figure 4-2: An coordinate descent algorithm for obtaining a dual optimum.

concentrates around its mode $\underline{n}^*$ given by

$$\underline{n}^* \in \arg \max_{\underline{n} \in \mathcal{S}(C)} \pi(\underline{n}).$$

Such concentration suggests the following approach which is the premise of the *1-point approximation*: Compute the mode $\underline{n}^* = (n_r^*)$ of the distribution and use n_r^* as a surrogate for $\mathbb{E}[n_r]$ in the computation of L_r via equation (4.3). Before presenting our specific iterative algorithm, we consider some related optimization problems upon which it is based.

The definition of the stationary distribution $\pi(\cdot)$ suggests that the mode $\underline{n}^*$ corresponds to a solution of the optimization problem

$$\begin{aligned} \text{maximize} \quad & \sum_r n_r \log \nu_r - \log n_r! \\ \text{over} \quad & \underline{n} \in \mathcal{S}(C). \end{aligned}$$

By Stirling's approximation, $\log n_r! = n_r \log n_r - n_r + O(\log n_r)$. Using this and ignoring the $O(\log n_r)$ term, the above optimization problem reduces to

$$\begin{aligned} \text{maximize} \quad & \sum_r n_r \log \nu_r + n_r - n_r \log n_r \\ \text{over} \quad & \underline{n} \in \mathcal{S}(C). \end{aligned}$$

A natural continuous relaxation of $\underline{n} \in \mathcal{S}(C)$ is

$$\bar{\mathcal{S}}(C) = \left\{ \underline{x} \in \mathbb{R}_+^K : \ A\underline{x} \leq \underline{C} \right\},$$

which yields the following primal problem (P):

$$\begin{aligned} \text{maximize} \quad & \sum_r x_r \log \nu_r + x_r - x_r \log x_r \\ \text{over} \quad & \underline{x} \in \bar{\mathcal{S}}(C). \end{aligned}$$

The above relaxation becomes a good approximation of the original problem when

all components of $\underline{C}$ are large. In order to design a simple iterative algorithm, we consider the Lagrangian dual (D) to the primal problem P where standard calculations yield

$$\begin{aligned} \text{minimize} \quad & \sum_r \nu_r \exp\left[-\sum_j y_j A_{jr}\right] + \sum_j y_j C_j \\ \text{over} \quad & \underline{y} \geq 0. \end{aligned}$$

Define the dual cost function $g(\underline{y})$ as

$$g(\underline{y}) = \sum_r \nu_r \exp\left[-\sum_j y_j A_{jr}\right] + \sum_j y_j C_j.$$

By Slater's condition, the strong duality holds and hence the optimal cost of P and D are the same. Standard Karush-Kuhn-Tucker conditions imply the following: Letting $(\underline{x}^*, \underline{y}^*)$ be a pair of optimal solutions to P and D, then

(a) For each link j,

$$\frac{\partial g(\underline{y}^*)}{\partial y_j} = 0 \text{ or } y_j^* = 0 \;\&\; \frac{\partial g(\underline{y}^*)}{\partial y_j} \leq 0.$$

Equivalently,

$$\sum_r A_{jr}\nu_r \exp\left[-\sum_j y_j^* A_{jr}\right] = C_j \;\&\; y_j^* > 0,$$

$$\text{or, } \sum_r A_{jr}\nu_r \exp\left[-\sum_j y_j^* A_{jr}\right] \leq C_j \;\&\; y_j^* = 0.$$

(b) For each route r,

$$x_r^* = \nu_r \exp\left[-\sum_j y_j^* A_{jr}\right].$$

The above conditions suggest the following approach: Obtain a dual optimal solution, say $\underline{y}^*$, use it to obtain $\underline{x}^*$, and then compute the loss probability as $1 - L_r = x_r^*/\nu_r$. Figure 4-2 describes an iterative, coordinate descent algorithm for obtaining $\underline{y}^*$. In what follows, we will use the transformation $z_j = \exp(-y_j)$ given its similarity with the Erlang fixed-point approximation. Note that z_j is 1 minus the blocking

probability for link j, E_j.

4.5.3 Slice method

The Erlang fixed-point approximation and the 1-point approximation essentially attempt to use the mode of the stationary distribution as a surrogate for the mean, which works quite well when the distribution is concentrated (near its mode). While this concentration holds for asymptotically large networks, it otherwise can be an important source of error and therefore we seek to obtain a new family of methods that provide better approximations.

The main premise of our slice methods follows from the fact that computing the loss probability L_r is equivalent to computing the expected number of calls $\mathbb{E}[n_r]$ via equation (4.3). By definition,

$$\mathbb{E}[n_r] = \sum_{k=0}^{\infty} k\mathbb{P}[n_r = k]$$

and thus $\mathbb{E}[n_r]$ can be obtained through approximations of $\mathbb{P}[n_r = k]$ rather than by the mode value n_r^*. Note that $\mathbb{P}[n_r = k]$ corresponds to the probability mass along the "slice" of the polytope defined by $n_r = k$. An exact solution for $\mathbb{E}[n_r]$ can be obtained with our slice method by using the exact values of $\mathbb{P}[n_r = k]$, but obtaining the probability mass along a "slice" can be as computationally hard as the original problem. Hence, our family of slice methods is based on approximations for $\mathbb{P}[n_r = k]$. To do so, we will exploit similar insights from previous approaches: Most of the mass along each slice is concentrated around the mode of the distribution restricted to the slice. This approximation is better than the "1-point approximation" since it uses the "1-point approximation" many times (once for each slice) in order to obtain a more accurate solution. Figure 4-3 formally describes the algorithm, where the cost function of the primal problem P is denoted by

$$q(\underline{x}) = \sum_r x_r \log \nu_r + x_r - x_r \log x_r.$$

Slice method

Compute L_r for route $r \in \mathcal{R}$ as follows:

(1) For each value of $k \in \{n_r : \underline{n} \in \mathcal{S}(C)\}$, use the "1-point approximation" to compute $\underline{x}^*(k,r)$ as the solution of the optimization problem

$$\text{maximize} \quad q(\underline{x}) \quad \text{over} \quad \underline{x} \in \bar{\mathcal{S}}(C) \ \& \ x_r = k.$$

(2) Estimate $\mathbb{E}[n_r]$ as

$$\mathbb{E}[n_r] = \frac{\sum_k k \exp(q(\underline{x}^*(k,r)))}{\sum_k \exp(q(\underline{x}^*(k,r)))}.$$

(3) Generate $L_r = 1 - \frac{\mathbb{E}[n_r]}{\nu_r}$.

Figure 4-3: Description of the slice method.

4.5.4 3-point slice method

In the general slice method, for each route r, we apply the 1-point approximation to each slice defined by $n_r = k$, $k \in \{n_r : \underline{n} \in \mathcal{S}(C)\}$. In the scaled system, this requires $O(N)$ applications of the "1-point approximation" for each route. Recall that, in contrast, the Erlang approximation (or 1-point approximation) requires only $O(1)$ applications of the iterative algorithm. To obtain a variation of the general slice method with similar computational complexity, we introduce another slice method "approximation" whose basic premise is as follows: Instead of computing $\underline{x}^*(k,r)$ for all $k \in \{n_r : \underline{n} \in \mathcal{S}(C)\}$, we approximate $\underline{x}^*(k,r)$ by linear interpolation between pairs of 3 points.

For a given route r, first apply the 1-point approximation for the entire polytope $\bar{\mathcal{S}}(C)$ to obtain the mode of distribution $\underline{x}^*$. Define

$$n^{\max}(r) \triangleq \max\{n_r : \underline{n} \in \mathcal{S}(C)\}.$$

Next, obtain $\underline{x}^*(n^{\max}(r), r)$, the mode of distribution in the slice $n_r = n^{\max}(r)$, using the 1-point approximation as in the general slice method. Finally, obtain $\underline{x}^*(0,r)$, the

mode of distribution in the slice $n_r = 0$, using the 1-point approximation. Now for $k \in \{n_r : \underline{n} \in \mathcal{S}(C)\}$, unlike in the general slice method, we will use an interpolation scheme to compute $x^*(k,r)$ as follows:

(a) If $k \leq x_r^*$, then

$$\underline{x}^*(k,r) = \underline{x}^* \cdot \frac{k}{x_r^*} + \underline{x}^*(0,r) \cdot \frac{x_r^* - k}{x_r^*}.$$

That is, $x^*(k,r)$ is the point of intersection (in the space $\mathbb{R}^K$) of the slice $x_r = k$ with the line passing through the two points $\underline{x}^*$ and $\underline{x}^*(0,r)$.

(b) For $x_r^* < k \leq n_r^{\max}$, let

$$\underline{x}^*(k,r) = \underline{x}^*(n^{\max}(r),r) \cdot \frac{k - x_r^*}{n^{\max}(r) - x_r^*} + \underline{x}^* \cdot \frac{n^{\max}(r) - k}{n^{\max}(r) - x_r^*}.$$

Note that due to the convexity of the polytope $\bar{\mathcal{S}}(C)$, the interpolated $x^*(k,r)$ are inside the polytope. Now, as in the general slice method, we use these $x^*(k,r)$ to compute the approximation of $\mathbb{E}[n_r]$ and subsequently L_r. A pseudo-code for the 3-point slice method can be found in [24].

4.6 Error in 1-point approximation

Consider a stochastic loss network with parameters A, $\underline{C}$ and $\underline{\nu}$ that is scaled by N as defined in Section 4.4. Kelly [29] obtained a fundamental result which shows that, in the scaled system, the stationary probability distribution concentrates around its mode. Therefore, the results of the 1-point approximation are asymptotically exact. In this Section, we reprove this result using a variational characterization of the stationary distribution, which yields a much simpler (and possibly more insightful) set of arguments.

Theorem 15 *Consider a loss network scaled by parameter N. Let L_r^N be the exact loss probability of route $r \in \mathcal{R}$. Then*

$$\left|(1 - L_r^N) - \frac{x_r^*}{\nu_r}\right| = O\left(\sqrt{\frac{\log N}{N}}\right). \tag{4.5}$$

Kelly established the asymptotic exactness of the Erlang fixed-point approximation by using the above result together with the fact that the Erlang fixed-point approximation for a scaled system essentially solves the dual D as N increases.

Proof. Recall that the stationary distribution π is represented as

$$\pi(\underline{n}) := \frac{1}{G(C)} \exp(Q(\underline{n})), \quad \exp(Q(\underline{n})) = \prod_{r \in \mathcal{R}} \frac{\nu_r^{n_r}}{n_r!}$$

for $\underline{n} \in \mathcal{S}(C)$. Define $\mathcal{M}(C)$ as the space of distributions on $\mathcal{S}(C)$. Clearly, $\pi \in \mathcal{M}(C)$. For $\mu \in \mathcal{M}(C)$, define

$$\begin{aligned} F(\mu) &\triangleq \sum_{\underline{n} \in \mathcal{S}(C)} \mu(\underline{n}) Q(\underline{n}) - \sum_{\underline{n} \in \mathcal{S}(C)} \mu(\underline{n}) \log \mu(\underline{n}) \\ &= \mathbb{E}_\mu(Q) + H(\mu). \end{aligned}$$

Next, we state a variational characterization of π, which will be extremely useful throughout. This characterization essentially states that π is characterized uniquely as the maximizer of $F(\cdot)$ over $\mathcal{M}(C)$.

Lemma 22 *For all $\mu \in \mathcal{M}(C)$,*

$$F(\pi) \geq F(\mu).$$

The equality holds iff $\mu = \pi$. Further, $F(\pi) = \log G(C)$.

Proof. From the definition of $Q(\cdot)$, we have

$$Q(\underline{n}) = \log \pi(\underline{n}) + \log G(C).$$

Consider the following sequence of inequalities, which essentially use Jensen's inequality together with the above definition of $Q(\cdot)$:

$$
\begin{aligned}
F(\mu) &= \sum_{\underline{n}\in\mathcal{S}(C)} \mu(\underline{n})Q(\underline{n}) - \sum_{\underline{n}\in\mathcal{S}(C)} \mu(\underline{n})\log\mu(\underline{n}) \\
&= \sum_{\underline{n}\in\mathcal{S}(C)} \mu(\underline{n})(\log\pi(\underline{n}) + \log G(C)) \\
&\quad - \sum_{\underline{n}\in\mathcal{S}(C)} \mu(\underline{n})\log\mu(\underline{n}) \\
&= \sum_{\underline{n}\in\mathcal{S}(C)} \mu(\underline{n})\left(\log\frac{\pi(\underline{n})}{\mu(\underline{n})}\right) + \log G(C) \\
&\leq \log\left[\sum_{\underline{n}\in\mathcal{S}(C)} \mu(\underline{n})\frac{\pi(\underline{n})}{\mu(\underline{n})}\right] + \log G(C) \\
&= \log 1 + \log G(C) = F(\pi).
\end{aligned}
$$

The only inequality above is tight iff $\mu = \pi$. This completes the proof of the Lemma 22. Now, consider the scaled system with parameter N. For any $\underline{n} \in \mathcal{S}(NC)$, this is equivalent to considering $\frac{1}{N}\underline{n} \in \mathcal{S}_N(C)$. Then, π for a scaled system is equivalent to the distribution π_N on $\mathcal{S}_N(C)$ defined, for $\underline{x} \in \mathcal{S}_N(C)$, as

$$
\pi_N(\underline{x}) = \pi(N\underline{x}) = \frac{1}{G(NC)}\exp(Q(N\underline{x})).
$$

Upon considering $Q(N\underline{x})$, we have

$$
\begin{aligned}
&\exp(Q(N\underline{x})) = \prod_r \frac{(N\nu_r)^{Nx_r}}{(Nx_r)!} \\
&= \exp\left(\sum_r Nx_r \log N\nu_r - \sum_r \log(Nx_r)!\right) \\
&= \exp\left(N\log N\sum_r x_r + N\sum_r x_r\log\nu_r - \sum_r \log(Nx_r)!\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \exp\left(N \log N \sum_r x_r + N \sum_r x_r \log \nu_r \right. \\
&\qquad \left. - N \sum_r x_r \log N x_r + \sum_r N x_r + \sum_r O(\log N x_r) \right) \\
&= \exp\left(N \sum_r x_r \log \nu_r - N \sum_r x_r \log x_r \right. \\
&\qquad \left. + N \sum_r x_r + \sum_r O(\log N x_r) \right),
\end{aligned}
$$

where the above calculations make use of Stirling's approximation:

$$\log M! = M \log M - M + O(\log M).$$

It then follows from these calculations that

$$
\begin{aligned}
\frac{1}{N} Q(N\underline{x}) &= \sum_r x_r \log \frac{\nu_r e}{x_r} + \frac{1}{N} \left[\sum_r \log(N x_r) \right] \\
&= q(\underline{x}) + O\left(\frac{\log N}{N} \right),
\end{aligned}
$$

where

$$q(\underline{x}) = \sum_r x_r \log \frac{\nu_r e}{x_r}. \tag{4.6}$$

Given the above calculations, we further obtain the following concentration for the distribution π_N, which will be crucial in proving Theorem 15.

Lemma 23 *Given any $\varepsilon > 0$, define the set*

$$A_\varepsilon = \left\{ \underline{y} \in \mathcal{S}_N(C) : \|\underline{y} - \underline{x}^*\| > \varepsilon \right\}$$

where $\underline{x}^* = \arg\max_{\underline{x} \in \bar{\mathcal{S}}(C)} q(\underline{x})$. Then

$$\pi_N(A_\varepsilon) = O\left(\varepsilon^{-2} \frac{\log N}{N} \right). \tag{4.7}$$

Proof. From the definition of $q(\cdot)$, it can be verified that this is a strongly concave function on the set $\bar{\mathcal{S}}(C)$. Further, the constraint set $\bar{\mathcal{S}}(C)$ is closed and convex. Therefore, there exists a unique optimal solution $\underline{x}^*$ of the optimization problem

$$\text{maximize } q(\underline{x}) \text{ over } \underline{x} \in \bar{\mathcal{S}}(C).$$

By the optimality and uniqueness of $\underline{x}^*$, we have for any $\underline{y} \in \bar{\mathcal{S}}(C)$

$$\nabla q(\underline{x}^*)^T(\underline{y} - \underline{x}^*) \leq 0.$$

Now consider Taylor's expansion of $q(\cdot)$ at some $\underline{y} \in \bar{\mathcal{S}}(C)$ at $\underline{x}^*$

$$q(\underline{y}) = q(\underline{x}^*) + \nabla q(\underline{x}^*)^T(\underline{y} - \underline{x}^*) + (\underline{y} - \underline{x}^*)^T \nabla^2 q(\underline{z})(\underline{y} - \underline{x}^*),$$

where $z = \alpha \underline{x}^* + (1 - \alpha)\underline{y}$, $\alpha \in [0, 1]$. Using the optimality condition, we have

$$q(\underline{y}) \leq q(\underline{x}^*) + (\underline{y} - \underline{x}^*)^T \nabla^2 q(\underline{z})(\underline{y} - \underline{x}^*). \tag{4.8}$$

Next, in order to evaluate the bound of (4.8), we will compute the Hessian $\nabla^2 q(z)$. For this, recall that

$$q(\underline{x}) = \sum_r x_r(1 + \log \nu_r) - x_r \log x_r.$$

Then the first partial derivative is given by

$$\frac{\partial q(x)}{\partial x_r} = 1 + \log \nu_r - 1 - \log x_r = \log \nu_r - \log x_r,$$

and the second order partial derivatives are given by

$$\frac{\partial^2 q(x)}{\partial x_r \partial x_s} = \begin{cases} 0, & \text{if } r \neq s \\ -\frac{1}{x_r}, & \text{if } r = s \end{cases}.$$

Therefore, the Hessian $\nabla^2 q(\cdot)$ is a diagonal matrix of the form

$$\nabla^2 q(\underline{x}) = \left[\frac{\partial^2 q}{\partial x_r \partial x_s} \right] = \begin{bmatrix} -\frac{1}{x_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & -\frac{1}{x_K} \end{bmatrix}. \tag{4.9}$$

Next, for any $z \in \bar{\mathcal{S}}(C)$, by definition for any $r \in \mathcal{R}$

$$0 \leq z_r \leq C_* \Leftrightarrow \frac{1}{z_r} \geq \frac{1}{C_*},$$

where we recall that $C_* = \max_j C_j$. Using this bound, the definition of the Hessian and (4.8), we obtain the following lemma.

Lemma 24 *For any $y \in \bar{\mathcal{S}}(C)$,*

$$q(\underline{y}) \leq q(\underline{x}^*) - \frac{1}{C_*} \left\| \underline{y} - \underline{x}^* \right\|^2.$$

Proof.

$$\begin{aligned} q(\underline{y}) &\leq q(\underline{x}^*) - \sum_r \frac{(y_r - x_r^*)^2}{z_r} \\ &\leq q(\underline{x}^*) - \frac{1}{C_*} \left(\sum_r (y_r - x_r^*)^2 \right) \\ &= q(\underline{x}^*) - \frac{1}{C_*} \left\| \underline{y} - \underline{x}^* \right\|^2. \end{aligned} \tag{4.10}$$

Now, given $\underline{x}^* \in \bar{\mathcal{S}}(C)$, there exists $\bar{x}^* \in \mathcal{S}_N(C)$ such that

$$\|\bar{x}^* - \underline{x}^*\| = O\left(\frac{1}{N}\right) \text{ and } q(\bar{x}^*) \geq q(\underline{x}^*) - O\left(\frac{1}{N}\right).$$

Consider a special distribution $\bar{\mu}$ over $\mathcal{S}_N(C)$ as

$$\bar{\mu}(\underline{x}) = \mathbf{1}_{\underline{x} = \bar{x}^*}, \quad \text{for } \underline{x} \in \mathcal{S}_N(C).$$

Namely, $\bar{\mu}$ puts all of its mass on element $\bar{x}^*$. Note that the entropy of the single-point distribution is zero, i.e., $H(\bar{\mu}) = 0$. By Lemma 22 and the definition of π_N, we obtain that π_N optimizes $F(\cdot)$. Therefore,

$$\begin{aligned}
\frac{1}{N}F(\pi_N) &\geq \frac{1}{N}F(\bar{\mu}) \\
&= \frac{1}{N}Q(N\bar{x}^*) + H(\bar{\mu}) \\
&\geq q(\bar{x}^*) - O\left(\frac{\log N}{N}\right) \\
&= q(\underline{x}^*) - O\left(\frac{\log N}{N}\right). \qquad (4.11)
\end{aligned}$$

Next, suppose that π_N puts some mass, say $\pi_N(A_\varepsilon)$, over set A_ε as defined in Lemma 23. It then follows from Lemma 22 that

$$\frac{1}{N}F(\pi_N) = \frac{1}{N}\mathbb{E}\left[Q(N\cdot)\right] + \frac{1}{N}H(\pi_N). \qquad (4.12)$$

The support of π_N is over at most $O(N^K)$ elements. By standard bounds on entropy, we have $H(\pi_N) = O(\log N)$. Using this and inequality (4.10) in (4.12), it follows that

$$\frac{1}{N}F(\pi_N) \leq q(\underline{x}^*) + O\left(\frac{\log N}{N}\right) - \frac{\varepsilon^2 \pi_N(A_\varepsilon)}{C_*}. \qquad (4.13)$$

From (4.11) and (4.13), we have

$$\pi_N(A_\varepsilon) = O\left(\varepsilon^{-2}\frac{\log N}{N}\right), \qquad (4.14)$$

which completes the proof of Lemma 23.

Now, we will complete the proof of Theorem 15. Using $\varepsilon_k = k\sqrt{\frac{\log N}{N}}$ for the value of ε in the conclusion of Lemma 23, from (4.14) we obtain

$$\pi_N\left(|x_r - x_r^*| > \varepsilon_k\right) = O\left(\frac{1}{k^2}\right), \qquad (4.15)$$

which immediately implies

$$\mathbb{E}\left[|x_r - x_r^*|\right] = O\left(\sqrt{\frac{\log N}{N}}\right) \times O\left(\sum_k \frac{1}{k^2}\right) = O\left(\sqrt{\frac{\log N}{N}}\right).$$

Thus,

$$\mathbb{E}[|x_r - x_r^*|] = O\left(\sqrt{\frac{\log N}{N}}\right),$$

and since $L_r = 1 - \frac{\mathbb{E}[x_r]}{\nu_r}$, we have

$$|L_r - L_r^*| = \frac{\mathbb{E}[|x_r - x_r^*|]}{\nu_r} = O\left(\frac{1}{\nu_r}\sqrt{\frac{\log N}{N}}\right).$$

This completes the proof of Theorem 15.

4.7 Error in Erlang fixed-point approximation

The Erlang fixed-point approximation is quite popular due to its natural iterative solution algorithm and its asymptotic exactness in the limiting regime. However, it is also well known that the Erlang fixed-point approximation can perform poorly in various cases. In this Section, we prove that Erlang fixed-point approximation perform poorly especially when the load vector $\underline{\nu}$ is such that it falls on the boundary of $\bar{\mathcal{S}}(C)$, i.e., the stochastic loss network is in the critically loaded regime. More precisely, this means $\underline{\nu}$ is such that at least one of the constraints in $A\underline{\nu} \leq \underline{C}$ is tight. It can be readily verified (at least for simple examples) that, when $\underline{\nu}$ is strictly inside or strictly outside $\mathcal{S}(C)$, then the error in the Erlang fixed-point approximation for the scaled network is $O(1/N)$. However, for the boundary, the qualitative error behavior changes, and in particular we prove the following result.

Theorem 16 *When the vector ν lies on the boundary of $\bar{\mathcal{S}}(C)$,*

$$\|\underline{L}^{\mathcal{E},N} - \underline{L}^N\|_2 = \Omega\left(\sqrt{\frac{1}{N}}\right), \tag{4.16}$$

where $\underline{L}^{\mathcal{E},N} = (L_r^{\mathcal{E},N})$ is the vector of loss probabilities from the Erlang fixed-point

approximation and $\underline{L}^N = (L_r^N)$ *is the vector of exact loss probabilities, both for a loss network scaled by* N.

Kelly proves in [29] that, for any route r,

$$\|(1 - L_r^{\varepsilon,N}) - \frac{x_r^*}{\nu_r}\| = O\left(\frac{1}{N}\right). \tag{4.17}$$

Hence, the following Lemma together with (4.3) and (4.17) proves Theorem 16.

Lemma 25 *When the vector* ν *lies on the boundary of* $\bar{\mathcal{S}}(C)$,

$$\left\| \left(\frac{\mathbb{E}_N[x_r]}{\nu_r}\right)_r - \left(\frac{x_r^*}{\nu_r}\right)_r \right\|_2 = \Omega\left(\sqrt{\frac{1}{N}}\right), \tag{4.18}$$

where $(\frac{\mathbb{E}_N[x_r]}{\nu_r})_r = (\frac{\mathbb{E}_N[x_1]}{\nu_1}, \frac{\mathbb{E}_N[x_2]}{\nu_2}, \ldots)$ *is the vector consisting of the expectations for the routes in the scaled (discrete) system with parameter* N, *and* $(\frac{x_r^*}{\nu_r})_r = (\frac{x_1^*}{\nu_1}, \frac{x_2^*}{\nu_2}, \ldots)$.

Proof. We shall briefly summarize a few of the technical details, referring to [24] for the complete proof of Lemma 25. Let us start with the following Claim, the proof of which is provided in [24].

Claim 5 *For any* $r \in \mathcal{R}$,

$$|\mathbb{E}_{\bar{\pi}_N}[x_r] - \mathbb{E}_N[x_r]| = O\left(\frac{1}{N}\right).$$

Then from Claim 5, to prove Lemma 25, it suffices to show that

$$\|\mathbb{E}_{\bar{\pi}_N}[x_r] - \nu\|_2 = \Omega\left(\frac{1}{\sqrt{N}}\right). \tag{4.19}$$

Define

$$S \triangleq \{v \in S^K : \bar{\mathcal{S}}(C) \cap (\nu + tv) \neq \emptyset \text{ for some } t > 0\},$$

where S^K is the unit sphere in $\mathbb{R}^K$. Now, for a given $v \in S$ and $t \in [0, t_v)$ where $t_v = \sup\{t \in \mathbb{R}_+ : (\nu + tv) \in \bar{\mathcal{S}}(C)\}$, define

$$g_N(v,t) \triangleq \exp(q_N(N(\nu + tv))).$$

Then from spherical integration, we obtain

$$\begin{aligned}\mathbb{E}_{\bar{\pi}_N}[\underline{x}] &= \frac{\int_S \int_0^{t_v} (\nu + tv) g_N(v,t) t^{K-1} dt\ dv}{\int_S \int_0^{v_t} g_N(v,t) t^{K-1} dt\ dv} \\ &= \nu + \frac{\int_S v \int_0^{t_v} g_N(v,t) t^K dt\ dv}{\int_S \int_0^{v_t} g_N(v,t) t^{K-1} dt\ dv},\end{aligned}$$

and thus

$$\|\mathbb{E}_{\bar{\pi}_N}[\underline{x}] - \nu\|_2 = \left\| \frac{\int_S v \int_0^{t_v} g_N(v,t) t^K dt\ dv}{\int_S \int_0^{t_v} g_N(v,t) t^{K-1} dt\ dv} \right\|_2. \tag{4.20}$$

Next, we introduce the following Lemma, which will be crucial in proving (4.19).

Lemma 26 *Let the polytope $\bar{\mathcal{S}}(C)$ and an integer ℓ be given. If N is large enough, then for all $v \in S$*

$$\int_0^{t_v} g_N(v,t) t^\ell dt = \Theta\left(N^{-\frac{K}{2}} \exp(NK) \frac{\Gamma\left(\frac{\ell+1}{2}\right)}{N^{\frac{\ell+1}{2}}} \right),$$

where $\Gamma(\cdot)$ is the Gamma function, and the constant hidden in $\Theta(\cdot)$ is uniformly bounded over all $v \in S$.

Proof. First, note that since $\bar{\mathcal{S}}(C)$ is a polytope having finitely many faces, there exists a constant $\delta > 0$ such that for all $v \in S$, and $t \in (0, \delta)$, $\nu + tv \in \bar{\mathcal{S}}(C)$. Now,

$$\begin{aligned} g_N(v,t) &= \exp(N \sum_r (\nu_r + v_r t)(1 + \log(N\nu_r)) \\ &\quad -N \sum_r (\nu_r + v_r t)(\log N + \log(\nu_r + v_r t)) \\ &\quad -\frac{1}{2} \sum_r \log N(\nu_r + v_r t)) \\ &= \exp(N \sum_r (\nu_r + v_r t)(1 + \log \nu_r - \log(\nu_r + v_r t)) \\ &\quad -\frac{K}{2} \log N - \sum_r \log(\nu_r + v_r t)) \\ &= \exp(Nq(\nu + tv) - \frac{K}{2} \log N - \sum_r \log(\nu_r + v_r t)), \end{aligned}$$

where

$$q(\underline{x}) = \sum_r (x_r + x_r \log \nu_r - x_r \log x_r).$$

Then by Lemma 24, we obtain that for all $v \in S$, and $t \in (0, t_v)$,

$$\begin{aligned} g_N(v,t) &= N^{-K/2} \left(\prod_r (\nu_r + v_r t) \right) (\exp(Nq(\nu + tv))) \\ &= N^{-K/2} \left(\prod_r (\nu_r) + O\left(t + t^K\right) \right) (\exp(Nq(\nu + tv))) \\ &\leq N^{-K/2} \left(1 + O\left(t + t^K\right)\right) \exp(Nq(\nu)) \exp(-Nt^2/C^*) \\ &= g_N(v,0) \exp\left(-Nt^2/C^*\right) \left(1 + O\left(t + t^K\right)\right), \end{aligned}$$

where $C^* = \max\{C_j\}$. Now we will set $\delta = N^{-\frac{1}{3}-0.01}$. Then when N is sufficiently large, for all $v \in S$ and $t \in (\delta, t_v)$,

$$g_N(v,t) \leq g_N(v,0) \exp\left(-N\delta^2/(2C^*)\right). \tag{4.21}$$

Therefore,

$$\int_0^{t_v} g_N(v,t) t^\ell dt = \Theta\left(\int_0^{\delta} g_N(v,t) t^\ell dt \right). \tag{4.22}$$

Now for all $t \in [0, \delta]$,

$$\begin{aligned} g_N(v,t) &= \exp(N \sum_r (\nu_r + v_r t)(1 + \log \nu_r - \log(\nu_r + v_r t)) \\ &\quad - \frac{K}{2} \log N - \sum_r \log(\nu_r + v_r t)) \\ &= N^{-\frac{K}{2}} \prod_r (\nu_r + v_r t) \\ &\quad \cdot \exp\left(N \sum_r (1 + v_r t) \left(1 - v_r t + v_r^2 t^2/2 + O\left(t^3\right)\right) \right) \\ &= N^{-\frac{K}{2}} \left(1 + O(t)\right) \exp\left(N \sum_r (1 - v_r^2 t^2/2 + O\left(\delta^3\right)) \right) \\ &= N^{-\frac{K}{2}} \left(1 + O(\delta)\right) \exp\left(KN - t^2 N/2 + N \cdot O\left(\delta^3\right)\right). \end{aligned}$$

Hence the following holds for any sufficiently large N and for all $v \in S$.

$$\int_0^\delta g_N(v,t)t^\ell dt = \Theta\left(N^{-\frac{K}{2}}\exp(KN)\int_0^\delta \exp(-t^2N)t^\ell dt\right). \tag{4.23}$$

Now if N is sufficiently large,

$$\int_0^\delta \exp(-t^2N)t^\ell dt = \Theta\left(\int_0^\infty \exp(-t^2N)t^\ell dt\right). \tag{4.24}$$

And from the formula

$$\Gamma(z) = 2\int_0^\infty \exp(-y^2)y^{2z-1}dy,$$

by substituting $y = \sqrt{N/2}t$ we obtain that

$$\int_0^\infty \exp(-t^2N)t^\ell dt = \frac{\Gamma\left(\frac{\ell+1}{2}\right)}{N^{\frac{\ell+1}{2}}}. \tag{4.25}$$

From (4.22), (4.23), (4.24) and (4.25), we prove Lemma 26.

Finally, let T be a tangent plane of $\bar{\mathcal{S}}(C)$ at the point ν and let $w \in S^K$ be a unit vector that is perpendicular to T and that satisfies $v \cdot w \geq 0$, for any $v \in S$. Then, from (4.20), we have

$$\begin{aligned}
\|\mathbb{E}_{\bar{\pi}_N}[\underline{x}] - \nu\|_2 &= \left\| \frac{\int_S v \int_0^{t_v} g_N(v,t)t^K dt\ dv}{\int_S \int_0^{t_v} g_N(v,t)t^{K-1}dt\ dv} \right\|_2 \\
&\geq \left| w \cdot \frac{\int_S v \int_0^{t_v} g_N(v,t)t^K dt\ dv}{\int_S \int_0^{t_v} g_N(v,t)t^{K-1}dt\ dv} \right| \\
&= \left| \frac{\int_S w\cdot v \int_0^{t_v} g_N(v,t)t^K dt\ dv}{\int_S \int_0^{t_v} g_N(v,t)t^{K-1}dt\ dv} \right| \\
&= \frac{\Theta\left(N^{-\frac{K}{2}}\exp(NK)\frac{\Gamma\left(\frac{K+1}{2}\right)}{N^{\frac{K+1}{2}}}\right)\int_S v\cdot w\ dv}{\Theta\left(N^{-\frac{K}{2}}\exp(NK)\frac{\Gamma\left(\frac{K}{2}\right)}{N^{\frac{K}{2}}}\right)\int_S 1\ dv} \\
&= \Theta\left(\sqrt{\frac{1}{N}}\right),
\end{aligned} \tag{4.26}$$

where we used Lemma 26 and the facts that $\frac{\int_S v \cdot w \ dv}{\int_S 1 \ dv} = \Theta(1)$ and $\frac{\Gamma\left(\frac{K+1}{2}\right)}{\Gamma\left(\frac{K}{2}\right)} = \Theta(1)$. From (4.26) we obtain (4.19), which completes the proof of Lemma 25.

4.8 Accuracy of the slice method

The drastically poorer accuracy of the Erlang fixed-point approximation at the boundary (i.e., in the critical regime) from Theorem 16 strongly motivates the need for new and better loss probability approximations. This led to our development of the general "slice method" described in Section 4.5.3. In this Section, we establish its asymptotic exactness using the variational characterization of the stationary distribution.

Theorem 17 *For each route $r \in \mathcal{R}$, let $L_r^{S,N}$ be the loss probability estimate obtained from the general slice method for the system scaled with parameter N. Let L_r^N be the corresponding exact loss probability. Then, for any system parameter values, we have*

$$\left|L_r^{S,N} - L_r^N\right| = O\left(\sqrt{\frac{\log N}{N}}\right). \tag{4.27}$$

This result establishes the asymptotic exactness of the slice method over all ranges of parameters. The proven error bound, which essentially scales as $O(1/\sqrt{N})$, does not imply that it is strictly better than the Erlang fixed-point approximation. We are unable to establish strict dominance of the slice method, but numerical results in Section 4.10 illustrate that the slice method can convincingly outperform the Erlang fixed-point approximation under critical loading.

Proof. Theorem 15 implies that the actual loss probability L_r^N, $r \in \mathcal{R}$, is given by

$$L_r^N = 1 - \frac{x_r^*}{\nu_r} + O\left(\sqrt{\frac{\log N}{N}}\right).$$

Therefore, the proof of Theorem 17 will be implied by showing that for all $r \in \mathcal{R}$

$$L_r^{S,N} = 1 - \frac{x_r^*}{\nu_r} + O\left(\sqrt{\frac{\log N}{N}}\right). \tag{4.28}$$

This result is established next where the proof crucially exploits our concentration Lemma 23.

From the definition of the "slice method", the estimated loss probability $L_r^{S,N}$ is defined as

$$L_r^{S,N} = 1 - \frac{1}{\nu_r} \frac{\sum_k k \exp(q(\underline{x}^*(k,r)))}{\sum_k \exp(q(\underline{x}^*(k,r)))}. \tag{4.29}$$

Recall that $\underline{x}^*(k,r)$ is the solution of the optimization problem

$$\begin{aligned} \text{maximize} \quad & q(\underline{x}) \\ \text{over} \quad & \underline{x} \in \bar{\mathcal{S}}(C) \ \& \ x_r = k, \end{aligned}$$

further recalling the definition of the function $q(\cdot)$ as

$$q(\underline{x}) = \sum_r x_r \log \nu_r + x_r - x_r \log x_r.$$

Now, consider a route $r \in \mathcal{R}$. In the rest of the proof, we will use

$$\varepsilon = \sqrt{\frac{2C^* \log N}{N}},$$

where $C_* = \max_j C_j$. Further define the following useful subsets

$$\begin{aligned} S(r,N) &\triangleq \{n_r : \underline{n} \in \mathcal{S}_N(C)\}, \\ S_\varepsilon(r,N) &\triangleq \{k \in S(r,N) : \|\underline{x}^*(k,r) - \underline{x}^*\| \le \varepsilon\}, \\ S_\varepsilon^c(r,N) &\triangleq \{k \in S(r,N) : \|\underline{x}^*(k,r) - \underline{x}^*\| > \varepsilon\}. \end{aligned}$$

Next, we note two facts that will be used to prove appropriate lower and upper bounds which yield the desired result (4.28). First, Lemma 24 and the above definitions imply that, for $k \in S_\varepsilon^c(r,N)$,

$$\exp\left(Nq(\underline{x}_r^k)\right) \le \frac{1}{N^2} \exp\left(Nq(\underline{x}^*)\right). \tag{4.30}$$

Second, it is easy to see there exists $k \in S_\varepsilon(r,N)$ such that

$$\|\underline{x}^*(k,r) - \underline{x}^*\| = O\left(\frac{1}{N}\right).$$

For this k, we have

$$\exp\left(Nq(\underline{x}^*(k,r))\right) \;=\; \Theta\left(\exp(Nq(\underline{x}^*))\right). \tag{4.31}$$

Since $|S(r,N)| = O(N)$ in the scaled system, (4.30) and (4.31) imply that

$$\begin{aligned}\sum_{k\in S_\varepsilon^c(r,N)} &\exp(Nq(x^*(k,r))) \\ &= O\left(\frac{\exp(Nq(x^*))}{N}\right)\end{aligned}$$

$$\leq O\left(\frac{1}{N}\sum_{k\in S_\varepsilon(r,N)} \exp(Nq(x^*(k,r)))\right). \tag{4.32}$$

From (4.32), the value of ε and the above subset definitions, we obtain the following sequence of inequalities:

$$\begin{aligned}&\frac{\sum_{k\in S(r,N)} k\exp(Nq(x^*(k,r)))}{\sum_{k\in S(r,N)}\exp(Nq(x^*(k,r)))} \\ &\quad\geq \frac{\sum_{k\in S_\varepsilon(r,N)} k\exp(Nq(x^*(k,r)))}{\sum_{k\in S(r,N)}\exp(Nq(x^*(k,r)))} \\ &\quad\geq \frac{\sum_{k\in S_\varepsilon(r,N)} k\exp(Nq(x^*(k,r)))}{(1+O(1/N))\left(\sum_{k\in S(r,N)}\exp(Nq(x^*(k,r)))\right)} \\ &\quad\geq \frac{1}{1+O(1/N)}\left(x_r^* - \varepsilon\right) \\ &\quad= x_r^* - O\left(\sqrt{\frac{\log N}{N}}\right).\end{aligned} \tag{4.33}$$

For all $k \in S_\varepsilon^c(r,N)$, $|k|$ is bounded by some constant, and therefore we have

$$\begin{aligned}
&\sum_{k \in S_\varepsilon^c(r,N)} k \exp(Nq(x^*(k,r))) \\
&\quad = O\left(\frac{\exp(Nq(x^*))}{N}\right) \\
&\quad \leq O\left(\frac{1}{N} \sum_{k \in S_\varepsilon(r,N)} \exp(Nq(x^*(k,r))\right). \qquad (4.34)
\end{aligned}$$

From (4.34) and the definition of ε, we obtain

$$\begin{aligned}
&\frac{\sum_{k \in S(r,N)} k \exp(Nq(x^*(k,r)))}{\sum_{k \in S(r,N)} \exp(Nq(x^*(k,r)))} \\
&\qquad \leq \frac{\sum_{k \in S(r,N)} k \exp(Nq(x^*(k,r)))}{\sum_{k \in S_\varepsilon(r,N)} \exp(Nq(x^*(k,r)))}
\end{aligned}$$

$$\begin{aligned}
&\leq (1 + O(1/N)) \frac{\sum_{k \in S_\varepsilon(r,N)} k \exp(Nq(x^*(k,r)))}{\sum_{k \in S_\varepsilon(r,N)} \exp(Nq(x^*(k,r)))} \\
&\leq (1 + O(1/N))(x_r^* + \varepsilon) \\
&= x_r^* + O\left(\sqrt{\frac{\log N}{N}}\right). \qquad (4.35)
\end{aligned}$$

Finally, equations (4.33) and (4.35) together with (4.29) imply (4.28), thus completing the proof of Theorem 17.

4.9 Convergence of algorithms

So far, certain accuracy properties have been established for the iterative algorithms. In this Section, we establish the exponential convergence of the iterative algorithm for the general slice method. It is sufficient to state the convergence of the 1-point approximation, since this is used as a subroutine in our slice methods.

Theorem 18 *Given a loss network with parameter $A, \underline{C}$ and $\underline{\nu}$, let $\underline{z}^{(t)}$ be the vector produced by the 1-point approximation at the end of iteration t. Assume that each*

link is utilized by some route in the matrix A. Let $\mathcal{Y}^*$ be the set of optimal solutions of the dual problem D, and let $\mathcal{Z}^* = \{(z_j^*) : z_j^* = \exp(-y_j^*), y^* \in \mathcal{Y}^*\}$. Then,

$$d(\underline{z}^{(t)}, \mathcal{Z}^*) \leq \alpha \exp(-\beta t),$$

where α, β are positive constants which depend on the problem parameters, and $d(\cdot, \mathcal{Z}^*)$ is distance to the set $\mathcal{Z}^*$.

Moreover, there is a unique primal optimum x^* and for all $z^* \in \mathcal{Z}^*$, $x^* = \left(\nu_r \prod_j (z_j^*)^{A_{jr}}\right)$. And

$$\left\| \left(\nu_r \prod_j (z_j^{(t)})^{A_{jr}} \right) - x^* \right\|_2 \leq \alpha' \exp(-\beta' t),$$

for some positive constants α' and β'.

Proof. The proof of the convergence of the "round-robin" coordinate descent algorithm used to find the optimal solution of the dual problem D will follow from a result of Luo and Tseng [40]. We first recall their precise result and then show how it implies Theorem 18.

In order to state the result in [40], some notations need to be introduced. Consider a real valued function $\phi : \mathbb{R}^n \to \mathbb{R}$ defined as

$$\phi(\underline{x}) = \psi(E\underline{x}) + \sum_{i=1}^{n} w_i x_i, \tag{4.36}$$

where $E \in \mathbb{R}^{m \times n}$ is an $m \times n$ matrix with no zero column (i.e., all coordinates of $\underline{x}$ are useful), $\underline{w} = (w_i) \in \mathbb{R}^n$ is a given fixed vector, and $\psi : \mathbb{R}^m \to \mathbb{R}$ is a strongly convex function on its domain

$$D_\psi = \left\{ \underline{y} \in \mathbb{R}^m : \psi(\underline{y}) \in (-\infty, \infty) \right\}.$$

We have D_ψ being open and let ∂D_ψ denote its boundary. We also have that, along any sequence $\underline{y}_k$ such that $\underline{y}_k \to \partial D_\psi$ (i.e., approaches boundary of D_ψ), $\psi(\underline{y}^k) \to \infty$.

The goal is to solve the optimization problem

$$\begin{aligned} \text{minimize} \quad & \phi(\underline{x}) \\ \text{over} \quad & \underline{x} \in \mathcal{X}, \end{aligned} \tag{4.37}$$

where we assume that $\mathcal{X}$ is of box-type, i.e.,

$$\mathcal{X} = \prod_{i=1}^{n} [\ell_i, u_i], \quad \ell_i, u_i \in \mathbb{R} \cup \infty.$$

Let $\mathcal{X}^*$ be the set of all optimal solutions of the problem (4.37). The "round-robin" or "cyclic" coordinate descent algorithm for this problem has the following convergence property, as proved in Theorem 6.2 of [40].

Lemma 27 *There exist constants α_0 and β_0 which may depend on the problem parameters in terms of $g, E, \underline{w}$ such that starting from the initial value $\underline{x}^0$, we have in iteration t of the algorithm*

$$d(\underline{x}^t, \mathcal{X}^*) \leq \alpha_0 \exp(-\beta_0 t)\, d(\underline{x}^0, \mathcal{X}^*).$$

Here, $d(\cdot, \mathcal{X}^*)$ denotes distance to the optimal set $\mathcal{X}^*$.

Now, to prove Theorem 18, we will apply Lemma 27.

The coordinate descent algorithm is equivalent to running the "1-point approximation" algorithm without the transformation $z_j = \exp(-y_j)$. Therefore, we can consider the original dual problem D:

$$\begin{aligned} \text{minimize} \quad & g(\underline{y}) \\ \text{over} \quad & \underline{y} \geq 0. \end{aligned}$$

Now, the above problem, similar to (4.37), has a box-type domain set: $\mathbb{R}^K$.

The function $g(\cdot)$ has the form

$$g(\underline{y}) = \sum_r \nu_r \exp\left[-\sum_j y_j A_{jr}\right] + \sum_j y_j C_j.$$

Assuming that each link is utilized by some route in the matrix A, it can be easily verified that $g(\cdot)$ can be written in the desired form (4.36) of the cost function of the optimization problem (4.37). Therefore, the setup of Theorem 18 satisfies the conditions of Lemma 27. Since for each $j \in J$, the transform $y_j \rightarrow \exp(-y_j)$ has bounded derivative when $(y_j) \in \mathbb{R}_+^J$, we have

$$d(\underline{z}^{(t)}, \mathcal{Z}^*) \leq \alpha \exp(-\beta t)$$

for some positive constants α and β.

Since the primal cost function is strictly concave, there is a unique primal optimum x^*. Hence, by the strong duality, we have that for all $z^* \in \mathcal{Z}^*$, $x^* = (\nu_r \prod_j (z_j^*)^{A_{jr}})$. Now, for each $r \in \mathcal{R}$ the transform $(z_j)_j \rightarrow \nu_r \prod_j (z_j^*)^{A_{jr}}$ has bounded gradient when$(z_j)_j \in [0,1]^J$. Hence we have

$$\left\| \left(\nu_r \prod_j (z_j^{(t)})^{A_{jr}} \right) - x^* \right\|_2 \leq \alpha' \exp(-\beta' t),$$

for some positive constants α' and β'.

4.10 Experiments

The main contributions of this paper are the theoretical results presented in Sections 4.5 – 4.9. However, to illustrate and quantify the performance of our family of slice methods, in this Section, we consider two different sets of numerical experiments. The first is based on a small canonical loss network topology that is used to investigate the fundamental properties of our slice methods and previous approaches with respect to the scaling parameter N. Then we turn to consider a large set of numerical

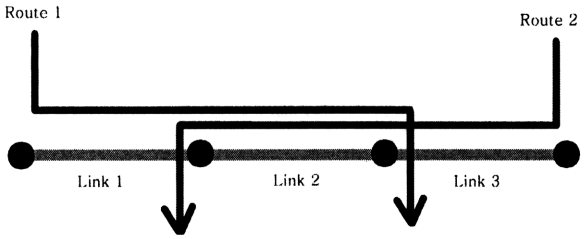

Figure 4-4: Illustration of the small canonical network model.

experiments based on workforce management applications in the IT services industry using real-world data.

4.10.1 Small loss networks

We consider a small canonical loss network topology comprised of two routes and three links, as illustrated in Figure 4-4. Both routes share link 2 with links 1 and 3 dedicated to routes 1 and 2, respectively. More precisely, the network is defined by

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad C = \begin{bmatrix} 2 \\ 3 \\ 2 \end{bmatrix}.$$

Figure 4-4 describes this network model.

In our first collection of experiments, we set $\rho_1 = 2$, $\rho_2 = 1$. The loss probabilities for this network model instance are then computed using our general slice method, the Erlang fixed-point method, and the 1-point approximation, where the loss probabilities in each case are considered as a function of the scaling parameter N. Note that, in this small model, the results from the 3-point slice method are identical to those from the general slice method, since the trace of the maximizer point for each slice in the general slice method indeed forms a linear interpolation of the three points. We also directly compute the exact loss probability by brute force and then obtain the average error (over both routes) for each method. These results are presented in Figure 4-5.

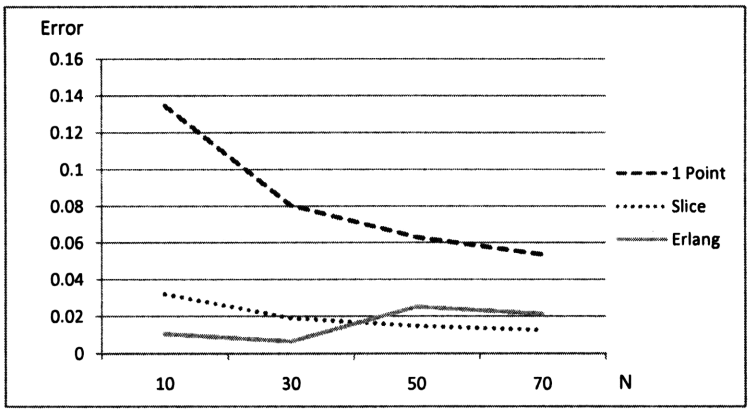

Figure 4-5: Average error of loss probabilities computed for each method as a function of the scaling parameter N.

First, we observe that the slice method performs better than the 1-point approximation method for every scaling value N. This result is as expected since the slice method utilizes more information about the probability distribution of the underlying polytope than the 1-point method. Second, it is quite interesting to observe that the Erlang fixed-point method initially performs better than the slice method in the small scaling region, whereas the slice method eventually provides the best performance among the approximation methods in the larger scaling region and the performance ordering among the methods shown for $N = 70$ continues to hold for $N > 70$. To understand this phenomena, note that as a function of the scaling with respect to N, the output of the Erlang fixed-point method converges to that of the 1-point approximation method on the order of $O\left(\frac{1}{N}\right)$ and the errors of the 1-point approximation method are given by $\Omega\left(\sqrt{\frac{1}{N}}\right)$, as established in Theorem 16. Moreover, when N becomes larger the error of the slice method becomes smaller than that of the Erlang fixed-point method because the error of the 1-point approximation

method is roughly a constant times that of the slice method for every sufficiently large N (as seen in Figure 4-5). Finally, while the asymptotic exactness of the Erlang fixed-point approximation is associated with the 1-point approximation method, Figure 4-5 also illustrates some of the complex characteristics of the Erlang fixed-point approximation in the non-limiting regime.

We also consider a second collection of experiments representing the symmetric case of $\rho_1 = \rho_2 = 1.5$. These results exhibit the same trends as in the asymmetric case, and hence are omitted.

4.10.2 Larger real-world networks

In this section, numerical experiments were also conducted for a large number of real-world loss network instances taken from various resource planning applications within the context of workforce management in the IT services industry. In each of these applications, the network routes represent various IT service products and the network links represent different IT resource capabilities. The various data sets comprising these model instances were obtained from actual IT service companies. First, we generally note that our results from such real-world model instances exhibit trends with respect to the scaling parameter N that are similar to those presented in Section 4.10.1 for a much simpler canonical model which captures fundamental properties of loss networks.

We shall focus on two representative model instances and present our comparative findings among the slice methods and previous approaches. The first model instance consists of 37 routes and 84 links, whereas the second model instance consists 110 routes and 132 links. In both data sets, the arrival rate vector ν happens to lie on the boundary of $\mathcal{S}(C)$.

The loss probabilities are computed for each loss network model instance using our general slice method, our 3-point interpolation slice method, and the Erlang fixed-point method. Since all of the real-world model instances are too large to numerically compute the exact solution, we use simulation of the corresponding loss network to estimate the exact loss probabilities within tight confidence intervals. The average

error (over all routes) and the individual per-route errors are then computed for each method in comparison with the exact loss probabilities, where the former results are summarized in Table 4.1.

	Erlang	slice method	3-point slice
Model instance 1	0.3357	0.1720	0.1720
Model instance 2	0.3847	0.0923	0.1148

Table 4.1: Average error of loss probabilities for each method.

It can be clearly observed from the results in Table 4.1 that the average relative improvements of our slice methods over the Erlang fixed-point method are quite significant. Even more importantly, we observe that the relative improvements for the individual routes are consistently and significantly better under both slice methods. In particular, the general (resp., 3-point) slice method provides the exact loss probabilities for 98 (resp., 93) of the 110 routes, while the Erlang fixed-point method never provides exact results, in model instance 2 and the 3-point slice method provides the exact loss probabilities for 10 of the 37 routes, while the Erlang fixed-point method provides the exact results for 5 of these 10 routes, in model instance 1. Note that when $L_r^S = L_r$, the relative error for the Erlang fixed-point method, $L_r^{\mathcal{E}}/L_r$, is equal to $1 + \mathcal{I}_r$ (resp., $1 - \mathcal{I}_r$) when $L_r^{\mathcal{E}} > L_r$ (resp., $L_r^{\mathcal{E}} < L_r$). In all of the cases where $\mathcal{I}_r = 1.0$, which represents a considerable number of routes in model instance 1 and the overwhelming majority of routes in model instance 2, both slice methods provide the exact loss probability for route r while the Erlang fixed-point method yields $L_r^E = 0$, even though the exact loss probabilities for these routes span the full range of values in $(0, 1)$. The loss probability estimates for a few routes are better under the Erlang fixed-point method than under the slice methods, but such routes are clearly in the minority representing a single route in model instance 1 and less than 6.5% of the routes in model instance 2.

Chapter 5

Conclusion

5.1 Summary

In this thesis, we have studied the Markov random field (MRF) model with respect to the two main problems of computing MAP and log-partition function. In Chapter 2, we have identified new classes of structural properties of MRF, that yield to simple, and efficient algorithms for approximate computation of MAP assignment and log-partition function. Specifically, for MRFs defined on graphs with polynomial growth, or graphs which are minor-excluded with bounded degree, we have designed efficient algorithms for approximate computation of MAP assignment and log-partition function within arbitrary accuracy. The running time of our algorithms are $\Theta(n)$, where n is the number of nodes in G, with constant dependent on the accuracy. The graphical models arising in wireless networks, statistical physics, and image processing do possess such graphical structure.

Then in Chapter 3, we applied our MAP computation algorithms to wireless network problems. We considered a wireless network of n nodes placed in some geographic area in an arbitrary manner. These nodes communicate over a common wireless medium under some interference constraints. We considered the problem of determining whether a given vector of end-to-end rates between various source-destination pairs can be supported by the network through a combination of routing and scheduling decisions among more than exponentially many possible choices in n.

We showed that such network graphs in geographic area do lead to a polynomial time algorithm to approximate the capacity of wireless networks to arbitrary accuracy, by applying our approximate computation of MAP algorithm to the corresponding MRF.

In Chapter 4, we studied the problem of computing loss probabilities of routes in a stochastic loss network, which is equivalent to computing the log-partition function of the corresponding MRF for the exact stationary distribution. We showed that when the loss network is critically loaded, the popular Erlang fixed-point approximation provides relatively poor performance. Then we proposed a novel algorithm, which we called "slice method", for estimating the stationary loss probabilities in stochastic loss networks based on structural properties of the exact stationary distribution, and showed that our algorithm always converges exponentially fast to the asymptotically correct solution. we also provided an alternative proof for an error upper bound of the Erlang fixed-point approximation using a variational characterization of the stationary distribution.

5.2 Future Work

We conclude this dissertation by providing some directions for future work. First we suggest an important class of problems for which application of our approximate inference algorithms is favorable. In the computer vision, the image segmentation problem has always remained an iconic problem [36]. Given a 2-dimensional or 3-dimensional (video) image, the goal of the problem is to cut, or segment, a specific part of the image. Examples include face recognition [64], disease detection problem in medical image processing [35], etc. The past few years have seen rapid progress made on this problem driven by the emergence of powerful optimization algorithms such as graph cuts [6]. The image segmentation problem is commonly formulated using the MRF model described in Chapter 1.1 defined on a 2-dimensional or 3-dimensional grid graph. In the model, segmenting an image with required property is expressed as the MAP computation problem of the MRF. In the context of image segmentation,

the vertex set corresponds to the set of all image pixels, and the value x_v denotes the labeling of the pixel v in the image. Similarly, other important problems in computer vision including image denoising are formulated by an MRF model defined on a grid graph [42]. Since the grid graphs are polynomially growing graphs, we can effectively apply our algorithms for these problems. As our future work we will apply our approximate MAP algorithms to these problems in computer vision.

In Section 2.7, we have shown that there exists a limit $\frac{1}{n^d} \log Z$ as $n \to \infty$, when the MRF is defined on a grid graph $\mathbb{Z}_n^d$ (of n^d nodes) with identical node and edge potential functions for all nodes and edges. We believe that our proof method can be applied to a more general class of MRFs where the underlying graph is a geometric random graph placed in an Euclidean space, and the node and the edge potential functions are drawn as per a distribution of the potential functions in an i.i.d. fashion. An important such example includes the Ising model of statistical physics. As our future work we will investigate the existence of a limit of $\log Z$ in this class of MRF.

Bibliography

[1] E. Arikan. Some complexity results about packet radio networks. *IEEE Trans. on Information Theory*, 30:681–685, July 1984.

[2] P. Assouad. Plongements lipschitziens dans $\mathbb{R}^n$. *Bull. Soc. Math. France*, 111(4):429–448, 1983.

[3] M. Bayati, D. Shah, and M. Sharma. Maximum weight matching via max-product belief propagation. In *IEEE ISIT*, 2005.

[4] S. Bhadra, Y. Lu, and M. S. Squillante. Optimal capacity planning in stochastic loss networks with time-varying workloads. *Proceedings of ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, June 2007.

[5] T. Bonald. The Erlang model with non-Poisson call arrivals. *Proceedings of Joint SIGMETRICS/Performance Conference on Measurement and Modeling of Computer Systems*, pages 276–286, June 2006.

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[7] D. Y. Burman, J. P. Lehoczky, and Y. Lim. Insensitivity of blocking probabilities in a circuit-switching network. *Journal of Applied Probability*, 21:850–859, 1984.

[8] A. K. Erlang. Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges. *E. Brockmeyer, H. L. Halstrom, and A. Jensen, editors,* The Life and Works of A. K. Erlang, *Paper first published in 1917.*, 1948.

[9] W. Feller. *An Introduction to Probability Theory and Its Applications.* Wiley, 1957.

[10] M. Franceschetti, O. Dousse, D. Tse, and P. Thiran. Closing the gap in the capacity of wireless networks via percolation theory. *IEEE Trans. on Information Theory*, 53:1009–1018, March 2007.

[11] H.-O. Georgii. Gibbs measures and phase transitions. Walter de Gruyter, 1988.

[12] A. Globerson and T. Jaakkola. Bound on partition function through planar graph decomposition. In *NIPS*, 2006.

[13] R. Gummadi, K. Jung, D. Shah, and R. Sreenivas. Feasible rate allocation in wireless networks. *IEEE International Conference on Computer Communications (INFOCOM)*, April 2008.

[14] R. Gummadi, K. Jung, D. Shah, and R. Sreenivas. Computing capacity region of a wireless network. *IEEE International Conference on Computer Communications (INFOCOM)*, April 2009.

[15] A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *FOCS '03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, page 534, Washington, DC, USA, 2003. IEEE Computer Society.

[16] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46:388–404, March 2000.

[17] J. Heinonen, editor. *Lectures on Analysis on Metric Spaces*. Springer, 2001.

[18] P. J. Hunt and F. P. Kelly. On critically loaded loss networks. *Advances in Applied Probability*, 21:831–841, 1989.

[19] H. Hunt-III, M. Marathe, V. Radhakrishnan, S. Ravi, D. Rosenkrantz, and R. Stearns. Nc-approximation schemes for np- and pspace-hard problems for geometric graphs. *J. Algorithms*, 26(2):238–274, 1998.

[20] E. Ising. Beitrag zur theore des ferromagnetismus. volume 31, pages 253–258. Zeitschr.. f. Physik, 1925.

[21] P. Jelenkovic, P. Momcilovic, and M. S. Squillante. Scalability of wireless networks. *IEEE/ACM Transactions on Networking*, 15:295–308, 2007.

[22] M. I. Jordan. Graphical models. *Statistical Science (Special Issue on Bayesian Statistics)*, 19:140–155, 2004.

[23] A. Jovicic, P. Viswanath, and S. Kulkarni. A network information theory for wireless communication: Scaling laws and optimal operation. *IEEE Trans. on Information Theory*, 50:2555–2565, November 2004.

[24] K. Jung, Y. Lu, D. Shah, M. Sharma, and M. S. Squillante. Revisiting stochastic loss networks: Structures and algorithms. *ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, June 2008.

[25] K. Jung and D. Shah. Local algorithms for approximate inference in minor-excluded graphs. *Annual Conference on Neural Information Processing Systems (NIPS)*, December 2007.

[26] K. Jung and D. Shah. Low delay scheduling in wireless network. *IEEE International Symposium on Information Theory (ISIT)*, June 2007.

[27] F. P. Kelly. Reversibility and stochastic networks. 1979.

[28] F. P. Kelly. Stochastic models of computer communication systems. *Journal of the Royal Statistical Society, Series B*, 47:379–395, 1985.

[29] F. P. Kelly. Blocking probabilities in large circuit-switched networks. *Advances in Applied Probability*, 18:473–505, 1986.

[30] F. P. Kelly. Routing in circuit-switched networks: Optimization, shadow prices and decentralization. *Advances in Applied Probability*, 20:112–144, 1988.

[31] F. P. Kelly. Loss networks. *Annals of Applied Probability*, 1:319–378, 1991.

[32] P. Klein, S. Plotkin, and S. Rao. Excluded minors, network decomposition and multicommodity flow. In *ACM STOC*, 1993.

[33] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. 2006.

[34] V. Kolmogorov and M. Wainwright. On optimality of tree-reweighted max-product message-passing. In *Uncertainty in Artificial Intelligence*, 2005.

[35] H. Li, M. Kallergi, L. Clarke, V. Jain, and R. Clark. Markov random field for tumor detection in digital mammography. *IEEE Transactions on Medical Imaging*, 14(3):565–576, 1995.

[36] S. Z. Li. Markov random field modeling in image analysis. Springer, 2001.

[37] J. D. C. Little. A proof of the queuing formula $L = \lambda W$. *Operations Research*, 9:383–387, 1961.

[38] G. Louth, M. Mitzenmacher, and F. Kelly. Computational complexity of loss networks. *Theoretical Computer Science*, 125:45–59, 1994.

[39] Y. Lu, A. Radovanović, and M. S. Squillante. Optimal capacity planning in stochastic loss networks. *Performance Evaluation Review*, 35, 2007.

[40] Z.-Q. Luo and P. Tseng. On the linear convergence of descent methods for convex essentially smooth minimization. *SIAM Journal on Control and Optimization*, 30:408–425, 1992.

[41] L.Xie and P.R.Kumar. A network information theory for wireless communication: Scaling laws and optimal operation. *IEEE Trans. on Information Theory*, 50:748–767, May 2004.

[42] M. Malfait and D. Roose. Wavelet-based image denoising using a markov random field a priori model. *IEEE Transactions on : Image Processing*, 6(4):549–565, 1997.

[43] C. D. Manning and H. Schutze. Foundations of statistical natural language processing. The MIT Press, 1999.

[44] D. Mitra, J. A. Morrison, and K. G. Ramakrishnan. Atm network design and optimization: A multirate loss network framework. *IEEE/ACM Transactions on Networking*, 4:531–543, 1996.

[45] D. Mitra and P. J. Weinberger. Probabilistic models of database locking: Solutions, computational algorithms and asymptotics. *Journal of the ACM*, 31:855–878, 1984.

[46] C. Moallemi and B. V. Roy. Convergence of the min-sum message passing algorithm for quadratic optimization. 2006.

[47] U. Niesen, P. Gupta, and D. Shah. On capacity scaling in arbitrary wireless networks. *submitted to IEEE Transactions on Information Theory*, November 2007. Available online at http://arxiv.org/abs/0711.2745.

[48] A. Özgür, O. Lévêque, and D. N. C. Tse. Hierarchical cooperation achieves optimal capacity scaling in ad hoc networks. *IEEE Transactions on Information Theory*, 53(10):3549–3572, October 2007.

[49] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Francisco, CA: Morgan Kaufmann, 1988.

[50] S. Rao. Small distortion and volume preserving embeddings for planar and euclidean metrics. In *SCG '99: Proceedings of the fifteenth annual symposium on Computational geometry*, pages 300–306, New York, NY, USA, 1999. ACM Press.

[51] T. Richardson and R. Ubanke. Modern coding theory. Cambridge University Press, 2008.

[52] N. Robertson and P. Seymour. The graph minor theory, started, 1984.

[53] K. W. Ross. Multiservice loss models for broadband telecommunication networks. 1995.

[54] A. Saleh and J. Simmons. Evolution toward the next-generation core optical network. *Journal of Lightwave Technology*, 24:3303–3321, 2006.

[55] S. Sarkar and S. Ray. Arbitrary throughput versus complexity tradeoffs in wireless networks using graph partitioning. *IEEE Transactions on Automatic Control*, 53(10):2307–2323, 2008.

[56] B. A. Sevastyanov. An ergodic theorem for Markov processes and its application to telephone systems with refusals. *Theoretical Probability Applications*, 2:104–112, 1957.

[57] D. Shah, D. Tse, and J. N. Tsitsiklis. On hardness of scheduling in wireless networks. personal communication, under preparation, 2008.

[58] G. Sharma, R. Mazumdar, and N. Shroff. On the complexity of scheduling in wireless networks. In *ACM Mobicom*, 2006.

[59] D. Sherrington and S. Kirkpatrick. Solvable model of a spin-glass. volume 35, pages 1792–1796. Physical Review Letter, 1975.

[60] K. Talwar. Bypassing the embedding: algorithms for low dimensional metrics. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 281–290, New York, NY, USA, 2004. ACM Press.

[61] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37:1936–1949, 1992.

[62] S. C. Tatikonda and M. I. Jordan. Loopy belief propagation and gibbs measure. In *Uncertainty in Artificial Intelligence*, 2002.

[63] H. Tijms. A first course in stochastic models. Wiley, 2003.

[64] P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[65] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. 2003.

[66] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. 2005.

[67] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. 2005.

[68] W. Whitt. Blocking when service is required from several facilities simultaneously. *AT&T Bell Laboratories Technical Journal*, 64:1807–1856, 1985.

[69] S. Xu, J. S. Song, and B. Liu. Order fulfillment performance measures in an assemble-to-order system with stochastic leadtime. *Operations Research*, 47:131–149, 1999.

[70] F. Xue and P. R. Kumar. Scaling laws for ad-hoc wireless networks: An information theoretic approach. *Foundation and Trends in Networking*, 1(2), 2006.

[71] J. Yedidia, W. Freeman, and Y. Weiss. Generalized belief propagation. 2000.

[72] I. B. Ziedins and F. P. Kelly. Limit theorems for loss networks with diverse routing. *Advances in Applied Probability*, 21:804–830, 1989.