Rectangular Permanent Approximation

A Project Report

Submitted by:

Ativ Joshi (201501040)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION AND COMMUNICATION TECHNOLOGY (ICT)

at

School of Engineering and Applied Sciences (SEAS)

Ahmedabad, Gujarat

May 2019

DECLARATION

I hereby declare that the project entitled “Rectangular Permanent Approximation” submitted for the B. Tech. (ICT) degree is my original work and the project has not formed the basis for the award of any other degree, diploma, fellowship or any other similar titles.

Signature of the Student

Place: Chennai Mathematical Institute

Date:

CERTIFICATE

This is to certify that the project titled “Rectangular Permanent Approximation” is the bona fide work carried out by Ativ Joshi, a student of B Tech (ICT) of School of Engineering and Applied Sciences at Ahmedabad University during the academic year 2018-19, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (Information and Communication Technology) and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

Signature of the Guide

Place: Chennai Mathematical Institute

Date:

ACKNOWLEDGEMENT

First and foremost I wish to express my deepest gratitude to my project mentor Dr. Partha Mukhopadhyay for his constant motivation and support. I would like to thank Rajit Datta and Abhranil Chatterjee for meaningful and productive discussions that helped me complete my project. A special thanks to Dr. Ratnik Gandhi and Dr. Bireswar Das who provided me with much needed guidance during my B.Tech.

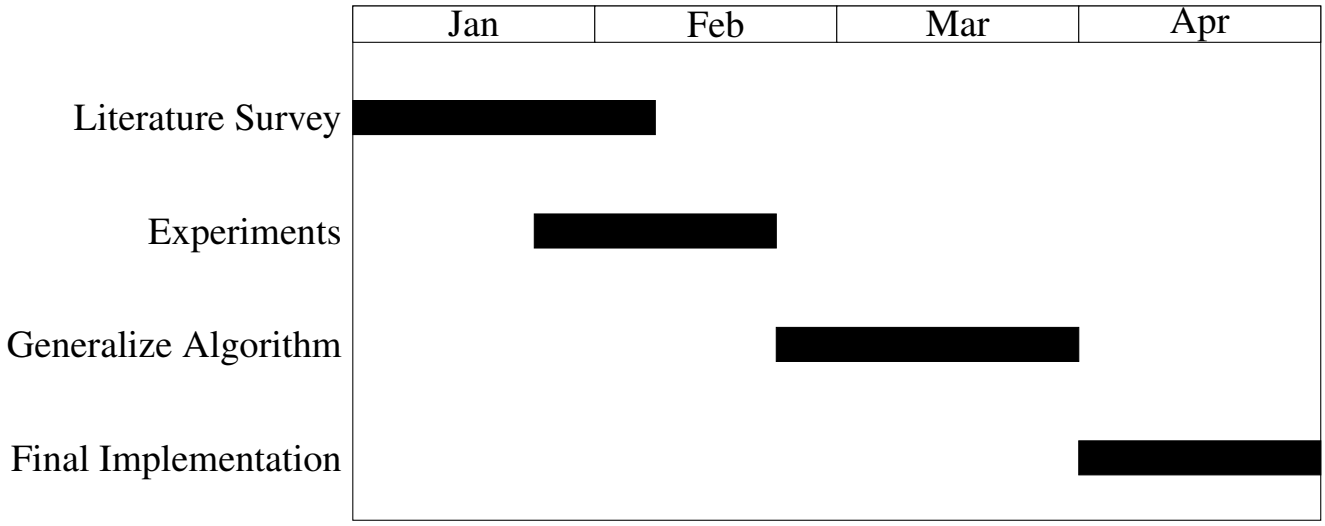
I would like to thank all my friends who made the undergrad journey the most memorable experience of my life. Finally, I would like to thank my family members who supported me during all my peaks and troughs.

ABSTRACT

Permanent of a matrix is an operation similar to determinant and has applications in combinatorics and statistical physics. Although determinant is easy to compute, the problem of deterministically computing a permanent is proved to be $\#P - Complete$. Balasubramanian–Bax–Franklin–Glynn formula and Ryser's Formula are the best known exact methods to compute permanent. We look at approximation algorithm which has its roots in statistical physics. The algorithm creates the probability distribution whose partition function is the permanent of a square matrix. Then it uses belief propagation to compute bethe free energy which approximates the permanent. We generalize the algorithm to approximate the permanent of rectangular matrices.

Keywords: permanent, belief propagation, bethe free Energy, graphical models, approximation algorithms

Gantt Chart


Jan
Feb
Mar
Apr
Literature Survey
Experiments
Generalize Algorithm
Final Implementation

Contents

List of Figures

1 Introduction

1.1 Definition

Permanent of a square matrix $P_{n\times n} = (p_{ij})$ is defined as

$$\text{perm}(P) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} p_{i,\sigma(i)}.$$

Where S_n is the symmetric group of permutations of n elements. The permanent is similar in definition to determinant (permanent is determinant where "subtraction is replaced by addition"). Permanent of rectangular matrix $P_{n\times m}$ is defined as sum of permanents of all the $\binom{m}{n}$ square sub-matrices.

1.2 Permanent and Matching in Bipartite Graph

Consider a bipartite graph $K = (L, R, E)$ with n elements in left and right partition. Define its adjacency matrix $P = (p_{ij}) \in \{0, 1\}^{n\times n}$ whose rows represents the left partition and column represents the right partition (i.e. if $p_{ij} = 1$, it means that i^{th} element of left partition and j^{th} element of right partition are connected).

It is easy to see that permanent of such a matrix P will compute the number of perfect matchings in K. This is because for any permutation $\sigma \in S_n$, the term $\prod_{i=1}^{n} p_{i,\sigma(i)}$ is 1 only if the selected edges form a perfect matching. If the graph is weighted, permanent will compute the weighted sum of perfect matchings where weight of a matching is product of weights of all the edges in the matching. This relation between bipartite graphs and permanent is the key part of the approximation algorithm.

Section 2 describes general graphical models and probability distributions over them. *Section 3* describes an iterative algorithm to compute marginal probabilities for graphical models. *Section 4* describes the bethe free energy whose minimum is exactly the partition function of some particular probabilistic graph models. The ideas described in this section are inspired from statistical physics. *Section 5* describes an algorithm which computes minimum of bethe free energy efficiently using belief propagation which in turn approximates the permanent. In *section 6*, we generalize the algorithm to compute the permanent of rectangular matrix. The flow of the literature survey will follow [9].

2 Graphical Models

Graphical models describes the relation between random variables (and functions of the random variables). There are several types of graphical models like Bayesian networks, factor graphs, ising models, random fields etc. Here we briefly describe two types of graphical models which are relevant.

2.1 Pairwise Markov Random Fields (PMRF)

PMRF consists of two types of variable: "observed variables" (y_i) and a corresponding "hidden variables" (x_i). We define $\phi_i(x_i, y_i)$ as a compatibility function. As y_i is fixed, we abbreviate it as $\phi_i(x_i)$. For two adjacent hidden nodes, we define $\psi_{ij}(x_i, x_j)$ as the compatibility function. The joint probability function is given by

$$p(\{x\}, \{y\}) = \frac{1}{Z} \prod_{(ij)} \psi_{ij}(x_i, x_j) \prod_i \phi_i(x_i) \tag{1}$$

. Here (ij) is the product over adjacent nodes and Z is the normalization constant.

2.2 Factor Graph

A factor graph is a bipartite graph where one partition represent the nodes corresponding to random variable x_i and the other represents the factors (functions) $\psi_a(\{x_a\})$ of these random variables. A random variable is connected to a factor if it is an argument of that factor. (Note that $\{x_a\}$ means the set of random variables which are arguments of ψ_a. The joint probability function is given by

$$p(\{x\}) = \frac{1}{Z} \prod_{a=1}^{M} \psi_a(\{x_a\}) \tag{2}$$

.

2.3 Conversion of Graphical Models (Optional)

The above graphical models are equivalent, i.e. one model can be converted to other model.

PMRF $\rightarrow$ Factor Graph: A factor graph function $\psi_a(\{x_a\})$ will be equivalent to the two node function $\psi_{ij}(x_i, x_j)$ if it links two hidden nodes or it will correspond to the single node function $\phi_i(x_i)$ if it connects a hidden node and observed node.

Factor Graph $\rightarrow$ PMRF: Each $\psi_a(\{x_a\})$ will be converted to a hidden node x_a with an observable node y_a linked to it. x_a can take as many states as the product of all the variables of the corresponding function ψ_a. $\psi_a(\{x_a\})$ is converted to $\phi_i(x_i)$. The variable of the Factor Graph remain unchanged and each of them are connected to the new variable x_a. $\psi_{aj}(x_a, x_j)$ is used to enforce the consistency between the new and old variables (as the state of x_a is related to the states of its linked variables which were the arguments of the function in the factor graph).

For a detailed explanation, see section 1.5 of [9].

3 Belief Propagation (BP)

3.1 Belief Propagation Algorithm

We consider the PMRF for the Belief Propagation algorithm. BP is a message passing algorithm. A message $m_{ij}(x_i)$ can be viewed as a message from hidden node i to hidden node j which describes what state should the node j be in *according* to node i. The message is a vector whose length is same as the number of states that the node j can take.

The beliefs approximate the marginal probability of every node (the approximation is exact if the graph is a tree).

Description: The joint distribution of PMRF is given by

$$p(\{x\}) = \frac{1}{Z} \prod_{(ij)} \psi_{ij}(x_i, x_j) \prod_i \phi_i(x_i)$$

.

The belief of a node is given by

$$b_i(x_i) = k\phi_i(x_i) \prod_{j \in N(i)} m_{ji}(x_i) \tag{3}$$

. $N(i)$ is the set of neighbouring node of i and k is a normalization constant (all beliefs should sum to 1).

The message from node i to j is given by

$$m_{ij}(x_j) \leftarrow \sum_{x_i} \phi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{ki}(x_i). \tag{4}$$

Observation. *BP gives exact marginal probabilities if the graph is singly-connected (does not have any loops)*

Proof. Let $\theta_{ij} := \phi_i(x_i)\psi_{ij}(x_i, x_j)$.

Now,

$$
\begin{aligned}
b_i(x_i) &= k\phi_i(x_i) \prod_{j \in N(i)} m_{ji}(x_i) \\
&= k\phi_i(x_i) m_{j_1 i} m_{j_2 i} \dots \\
&= k\phi_i(x_i) \Big(\sum_{x_{j_1}} \theta_{j_1 i} \prod_{l \in N(j_1) \setminus i} m_{l j_1}(x_{j_1}) \Big) \Big(\sum_{x_{j_2}} \theta_{j_2 i} \prod_{l \in N(j_2) \setminus i} m_{l j_2}(x_{j_2}) \Big) \dots \\
&= k\phi_i(x_i) \Bigg(\sum_{x_{j_1}} \theta_{j_1 i} \Big(\Big(\sum_{x_{l_1}} \theta_{l_1 i_1}(\dots) \Big) \Big(\sum_{x_{l_1}} \theta_{l_1 i_1}(\dots) \Big) \dots \Big) \Bigg) \dots \\
&= k \sum_{\{x\} \setminus x_i} \prod_{(ab)} \psi_{ab}(x_a, x_b) \prod_a \phi_a(x_a)
\end{aligned}
$$

Here $\{x\} \setminus x_i$ means summation over all variables except x_i. Last equality is the of marginal probability of x_i (by definition). We can take all the summation out in the last step because none of the variables will repeat as there are no loops. □

In practice, for trees we can start computing messages from leaf nodes and go upwards and once we reach the root, go downwards. There is no need to initialize the messages as by definition, messages from leaf nodes to other nodes will be its corresponding function. The algorithm will terminate once we go from leaves to root and back. Each message is only calculated once.

The BP algorithm does not assume the structure of the graph by itself, and can be used on graphs with loops by starting with some initial value of messages and iterating (and hoping for convergence). The beliefs will not give the exact marginals in this case. In fact there are cases when the algorithm does not even converge. However, it is observed that in practice, the BP approximates the marginal probabilities very well most of the time.

3.2 Two Node Beliefs and Marginals

For singly-connected PMRF, it is convenient to define 2-node marginals for two adjacent node i and j:

$$
p_{ij}(x_i, x_j) = \sum_{z : z_{ij} \setminus (x_i, x_j)} P(\{z\}) \tag{5}
$$

The corresponding belief will be analogous to the one node belief:

$$b_{ij}(x_i, x_j) = k\psi_{ij}(x_i, x_j)\phi_i(x_i)\phi_j(x_j) \prod_{k \in N(i)\setminus j} m_{ki}(x_i) \prod_{l \in N(j)\setminus i} m_{lj}(x_j) \tag{6}$$

The marginalization condition $b_i(x_i) = \sum_{x_j} b_{ij}(x_i, x_j)$ is satisfied by the 2-node marginals. Analogous to the previous proof, the 2-node beliefs give exact 2-node marginals for singly-connected graphs.

4 Free Energy

4.1 Kullback-Leibler Distance (KL)

Every graph has a corresponding joint probability distribution function $p(\{x\})$. Suppose if we have some other approximate joint probability distribution $b(\{x\})$, we can find out how "close" the approximated distribution is to the original distribution using KL Distance. The KL Distance between $p(\{x\})$ and $b(\{x\})$ is given by,

$$D(b(\{x\})||p(\{x\}) = \sum_{\{x\}} b(\{x\}) ln \frac{b(\{x\})}{p(\{x\})} \tag{7}$$

The KL distance is always non-negative and it is 0 if and only if both the distributions are exactly same. It does not satisfy triangle inequality.

4.2 Energy

Assuming the Boltzmann's Law from statistical physics,

$$p(\{x\}) = \frac{1}{Z} e^{-E(\{x\})/T}$$

we can define the "energy" $E(\{x\})$ directly (we take $T = 1$ as it is just a parameter that defines scale of units of energy). So the energy is given by

$$E(\{x\}) = -ln\ (Z\ p(\{x\})) \tag{8}$$

By substituting energy (8) into our distance measure (7), we have

$$D(b(\{x\})||p(\{x\}) = \sum_{\{x\}} b(\{x\}) E(\{x\}) + \sum_{\{x\}} b(\{x\}) ln(b(\{x\})) + ln\ Z \tag{9}$$

The KL distance will be zero when

$$\begin{aligned} G(b(\{x\})) &= \sum_{\{x\}} b(\{x\}) E(\{x\}) + \sum_{\{x\}} b(\{x\}) ln(b(\{x\}) \\ &= U(b(\{x\})) - S(b(\{x\})) \end{aligned} \tag{10}$$

will achieve its minimum value of $-ln\ Z$. We call G as "Gibbs free energy", U as "average energy" and S as "entropy".

Observe that we can approximate (or compute exactly in case of singly connected graph) the partition Z by computing the minimum of the Gibbs free energy. Computing the

partition exactly in case of an arbitrary graph is exponentially hard, just like the marginals. (Optional: We will see that the permanent can be approximated by modeling it as a partition of a bipartite graph).

4.3 Mean Field Energy (MF)

The Gibbs Free Energy still has exponentially many variable as its argument (as $b(\{x\})$ represents all the joint probability functions in the power set of variables). We can simplify this by assuming that the approximate joint probability distribution $b(\{x\})$ is factorized over the site, i.e.

$$b(\{x\}) = \prod_i b_i(x_i) \tag{11}$$

where $\sum_i b_i(x_i) = 1$. Here the one-node beliefs will be $b_i(x_i)$ and the two-node beliefs will be the the products of corresponding one-node beliefs, $b_{ij}(x_i, x_j) = b_i(x_i)b_j(x_j)$.

Now, considering the PMRF model, the energy will be

$$E(\{x\}) = -\sum_{ij} ln\ \psi_{ij}(x_i, x_j) - \sum_i ln\ \phi_i(x_i)$$

average energy will be

$$U_{MF}(\{b_i\}) = -\sum_{(ij)} \sum_{x_i, x_j} b_i(x_i)b_j(x_j) ln\ \psi_{ij}(x_i, x_j) - \sum_i \sum_{x_i} b_i(x_i) ln\ \phi_i(x_i)$$

and entropy will be

$$S_{MF}(\{b_i\}) = -\sum_i \sum_{x_i} b_i(x_i) ln\ b_i(x_i)$$

.

Obtaining the expression of energy and average energy is trivial. The entropy is derived as follows:

Claim. $S_{MF}(\{b_i\}) = -\sum_i \sum_{x_i} b_i(x_i) ln\ b_i(x_i)$

Proof. Substituting the assumption (11) into the definition of entropy (10)

$$\begin{aligned}
S_{MF}(\{b_i\}) &= -\sum_{x_1,x_2\ldots} b(x_1,x_2,\ldots) ln\ b(x_1,x_2,\ldots) \\
&= -\sum_{x_1,x_2\ldots} b(x_1,x_2,\ldots) ln \prod_i b_i(x_i) \\
&= -\sum_{x_1,x_2\ldots} b(x_1,x_2,\ldots) ln\ b_1(x_1) - -\sum_{x_1,x_2\ldots} b(x_1,x_2,\ldots) ln\ b_2(x_2) - \ldots \\
&= -\sum_{x_1} b_1(x_1) ln\ b_1(x_1) - \sum_{x_2} b_2(x_2) ln\ b_2(x_2) - \ldots \\
&= -\sum_i \sum_{x_i} b_i(x_i) ln\ b_i(x_i)
\end{aligned}$$

□

4.4 Bethe Free Energy (BFE)

The Mean Field energy is a function of only one-node beliefs. Now we want the approximation of Gibbs free energy which is function of both one-node and two-node beliefs. The beliefs should obey the normalization condition $\sum_{x_i} b_i(x_i) = \sum_{x_i,x_j} b_{ij}(x_i,x_j) = 1$ and the marginalization condition $b_i(x_i) = \sum_{x_j} b_{ij}(x_i,x_j)$. (Note that the marginalization condition is only for one and two node beliefs. It does not need to obey for arbitrary number of variables. These are also called "pseudo-marginals" in [5]).

For PMRF the one-node and two-node marginals are sufficient to compute the average energy. For an approximate joint probability distribution, if the one-node and two-node marginals are exactly same as the original probability distribution, the average energy will be exact. This means that if the one-node and two-node beliefs are exact, the average energy will be exact. The average energy is given by

$$U_{Bethe} = -\sum_{(ij)} \sum_{x_i,x_j} b_{ij}(x_i,x_j) ln\ \psi_{ij}(x_i,x_j) - \sum_i \sum_{x_i} b_i(x_i) ln\ \phi_i(x_i)$$

The calculation of entropy requires all joint probability distributions, so it is not easy to calculate entropy exactly. However, if the graph is singly-connected, we can express the joint distribution in terms of one node and two node marginals [6],

$$b(\{x\}) = \frac{\prod_{(ij)} b_{ij}(x_i,x_j)}{\prod_i b_i(x_i)^{q_i-1}} \tag{12}$$

where q_i is the number of neighbouring nodes of node i. Using this assumption, the entropy is given by

$$S_{Bethe} = -\sum_{(ij)} \sum_{x_i,x_j} b_{ij}(x_i, x_j) ln\ b_i(x_i, x_j) + \sum_i (q_i - 1) \sum_{x_i} b_i(x_i) ln\ b_i(x_i)$$

From our previous claim that BP algorithm gives exact one-node and two-node marginals in case of singly connected graph, it follows that for such graphs the values of those beliefs that minimize the bethe free energy $G_{Bethe} = U_{Bethe} - S_{Bethe}$ correspond to the exact marginal probabilities.

Note:
We can write U_{Bethe} in a form similar to S_{Bethe} by defining "local energies" $E_i(x_i) = -ln\ \phi_i(x_i)$ and $E_{ij}(x_i, x_j) = -ln\ \psi_{ij}(x_i, x_j) - ln\ \phi_i(x_i) - ln\ \phi_j(x_j)$. So average energy can be written as

$$U_{Bethe} = \sum_{(ij)} \sum_{x_i,x_j} b_{ij}(x_i, x_j) E_{ij}(x_i, x_j) + \sum_i (q_i - 1) \sum_{x_i} b_i(x_i) E_i(x_i)$$

So, G_{Bethe} can be written as

$$\begin{aligned} G_{Bethe}(b_i(x_i), b_{ij}(x_i, x_j)) = &\sum_{(ij)} \sum_{x_i,x_j} b_{ij}(x_i, x_j)(E_{ij}(x_i, x_j) + ln\ b_i(x_i, x_j)) \\ &+ \sum_i (q_i - 1) \sum_{x_i} b_i(x_i)(E_i(x_i) + ln\ b_i(x_i)) \end{aligned} \tag{13}$$

4.5 Equivalence of BP and BFE

From the previous discussion, we know the following about BP and BFE in case of **singly-connected** PMRF:

- Bethe Free Energy is equal to the exact Gibbs Free Energy
- BP gives correct one-node and two-node marginals.
- Beliefs computed using BP algorithm are the global minima of Bethe Free Energy (and Gibbs Free Energy as they are equal).

Now we claim the following,

Claim. *Fixed points of the Belief Propagation algorithm are the local stationary points of Bethe Free Energy.*

Proof Sketch. We add three lagrangian multipliers to G_{Bethe} to convert it into a lagrangian L:

- $\lambda_{ij}(x_j)$ for the marginalization constraint $b_i(x_i) = \sum_{x_j} b_{ij}(x_i, x_j)$.
- γ_{ij} for the normalization condition $\sum_{x_i,x_j} b_{ij}(x_i, x_j) = 1$.
- γ_i for the normalization condition $\sum_{x_i} b_i(x_i) = 1$.

So the Lagrangian will be

$$\begin{aligned} L = G_{Bethe} &- \sum_{ij} \Big(\big(\lambda_{ij}(x_j)\big) \big(b_i(x_i) - \sum_{x_j} b_{ij}(x_i, x_j)\big) \Big) \\ &- \sum_{ij} \Big(\gamma_{ij} \big(\sum_{x_i,x_j} b_{ij}(x_i, x_j) - 1 \big) \Big) \\ &- \sum_{i} \Big(\gamma_i \big(\sum_{x_i} b_i(x_i) - 1 \big) \Big) \end{aligned} \tag{14}$$

Now, differentiating L w.r.t. $b_{ij}(x_i, x_j)$ and equating to 0 gives

$$ln\ b_{ij}(x_i, x_j) = -E_{ij}(x_i, x_j) + \lambda_{ij}(x_j) + \lambda_{ji}(x_i) + \gamma_{ij} - 1 \tag{15}$$

differentiating L w.r.t. $b_i(x_i)$ and equating to 0 gives

$$(q_i - 1)(ln\ b_i(x_i) + 1) = (1 - q_i)E_i(x_i) + \sum_{j \in N(i)} \lambda_{ji}(x_i) + \gamma_i \tag{16}$$

and differentiating L w.r.t. Lagrangian multipliers will give back the constraints.

Now, if we have a set of messages and beliefs which are the fixed points of BP and if we define $\lambda_{ji}(x_j)$ as

$$\lambda_{ji}(x_j) = ln \prod_{k \in N(j) \setminus i} m_{kj}(x_j) \tag{17}$$

we can show that $\lambda_{ji}(x_j)$ and the beliefs satisfy the stationary conditions (15) and (16). Similarly, given the beliefs and Lagrange multipliers that satisfy the stationary conditions (15) and (16), we can use (17) to define messages such that the messages and beliefs satisfy the BP fixed points. □

5 Permanent of a Positive Square Matrix

We use theory discussed above to model the problem of approximating permanent of a positive square matrix as computing the partition Z of joint probability of the graphical model. Recall that the minimum of the Gibbs free energy is $-ln\ Z$, so if we compute the minimum , we can recover Z. We describe two ways in which permanent can be modeled. We also discuss the bounds and guarantees given by the algorithm.

5.1 Using PMRF

This is analogous to [5] and we will follow a similar notation. We want to approximate the permanent of a $n \times n$ matrix $P = (p_{ij})$.

Take a complete bipartite graph $K_{n,n}$ of $2n$ elements. Let the nodes in the left partition be $x_1, \ldots, x_n$ and right partition be $y_1, \ldots, y_n$. Each node can take n states {1, …, n}. If a node x_i is in state k, we can say that x_i "chooses" the node y_k and vice versa. If node $x_i = j$ and $y_j = i$, we can say that they are "matched" (this will be used below).

The matrix can be viewed as an adjacency matrix with p_{ij} representing the weight of an edge between x_i and y_j. If we define the weight of a matching as product of all the edges, then the permanent is the sum of weights of all the perfect matchings.

Now we define the functions of PMRF as follows:

- $\phi_i(x_i) = \sqrt{p_{i,x_i}}$
- $\phi_j(y_j) = \sqrt{p_{y_j,j}}$
- $\psi_{ij}(x_i, y_j) = I(\neg(x_i = j \oplus y_j = i))$. $I(.)$ is 0 if the argument is false, and 1 if true. This will be used to check if all the variables are "matched" perfectly.

So,

$$\prod_{i,j} \psi_{ij}(x_i, y_j) \prod_i \phi_i(x_i) \prod_j \phi_i(y_j)$$

will be 0 if the the variables x_i and y_j are not matched perfectly, and will give the weight of a perfect matching otherwise. The partition is

$$Z = \sum_{x_1,\ldots,x_n,y_1,\ldots,y_n} \prod_{i,j} \psi_{ij}(x_i, y_j) \prod_i \phi_i(x_i) \prod_j \phi_i(y_j)$$

which is also the definition of permanent. This can be approximated by minimizing the corresponding G_{Bethe} using BP (the q_i will be n in (13)).

The bethe function for the square matrix can be written as

$$\begin{aligned} G_{Bethe} = &- \sum_{ij} \sum_{x_i, y_j} b_{ij}(x_i, y_j) ln\ \psi_{ij}(x_i, y_j) \phi_i(x_i) \phi_j(y_j) + \sum_{ij} \sum_{x_i, y_j} b_{ij}(x_i, y_j) ln\ b_{ij}(x_i, y_j) \\ &- (n-1) \sum_{i} \sum_{x_i} b_i(x_i) ln\ b_i(x_i) - (n-1) \sum_{j} \sum_{y_j} b_j(y_j) ln\ b_j(y_j) \\ &+ (n-1) \sum_{i} \sum_{x_i} b_i(x_i) ln\ \phi_i(x_i) + (n-1) \sum_{j} \sum_{y_j} b_j(y_j) ln\ \phi_j(y_j) \end{aligned} \tag{18}$$

Bi-stochasticity: Note that ψ_{ij} enforces he following condition

$$b_{ij}(x_i = j, y_j = i) = b_i(x_i = j) = b_j(y_j = i)$$

as $\sum_{y_j} b_{ij}(x_i = j, y_j) = b_i(x_i = j) = b_{ij}(x_i = j, y_j = i)$ because only $\psi_{ij}(x_i = j, y_j = i)$ will be 1 and all others will be 0. So all $b_{ij}(x_i = j, y_j = i) = B_{ij}$ can be written as the entries of a bi-stochastic matrix $B = (B_{ij})$. This fact is exploited in [2], [7], [1] and [8].

As evident from previous paragraph $b_i(x_i)$ and $b_j(y_j)$ are matrices containing n^2 elements and they are transpose of each other. $b_{ij}(x_i, y_j)$ is a set of n^2 matrices each corresponding to a fixed i and j. An entry of $b_{ij}(x_i, y_j)$ can be written as

$$b_{ij}(x_i = k, y_j = l) = \frac{b_i(x_i = k) b_j(y_j = l) \psi_{ij}(x_i = k, y_j = l)}{m_{x_i}(y_j = l) m_{y_j}(x_i = k)}.$$

Thus, it can be seen that we only need to compute a bi-stochastic matrix and all the terms for beliefs can be derived from it.

5.2 Using Normal Factor Graphs (NFG)

The following explanation is analogous to [8]. The NFG is similar to a factor graph where the variables are represented as edges instead of separate node. This enforces a restriction that each variable appears in exactly two function.

Any factor graph can be trivially converted to an NFG by introducing a dummy function in the place of a variable node and connecting it to the functions in which the variable is an argument (the edges can be seen as replicas of the original variable). The new dummy function is 1 only if value of all the variables (edges) is same, else it is 0. (For a detailed explanation, see Section 1-A of [3]).

As before, consider a $n \times n$ matrix $P = (p_i^j)$ whose permanent can be modeled using perfect matchings on $K_{n,n}$, which can written as a $n \times n$ binary matrix σ define as

$$\left\{ \sigma = (\sigma_i^j) \middle| \forall i \sum_{j=1}^{n} \sigma_i^j = 1, \forall j \sum_{i=1}^{n} \sigma_i^j = 1 \right\}. \tag{19}$$

Now we define

$$\mathcal{P}(\sigma) = \frac{1}{Z} P^\sigma; \quad P^\sigma \equiv \prod_{i,j \in E} (p_i^j)^{\sigma_i^j}; \quad Z \equiv \sum_{\sigma \in PM} (p_i^j)^{\sigma_i^j} = Perm(P) \tag{20}$$

where E is the set of edges of the bipartite graph.

The KL distance will be written as

$$D(b(\sigma)) = \sum_{\sigma} b(\sigma) ln \, \frac{b(\sigma)}{P^\sigma}$$

and if the graph is bipartite, the beliefs can be written as

$$b(\sigma) = \frac{\prod_i b_i(\sigma_i) \prod_j b^j(\sigma^j)}{\prod_{(i,j) \in E} b_i^j(\sigma_i^j}$$

where $\forall i, j : \sigma_i = (\sigma_i^j \in \{0,1\} | j = 1, \ldots, n)$ s.t. $\sum_j \sigma_i^j = 1$ and $\forall i, j : \sigma^j = (\sigma_i^j \in \{0,1\} | i = 1, \ldots, n)$ s.t. $\sum_i \sigma_i^j = 1$. This is analogous to (12).

Here, b_i and b^j are beliefs related to vertices and b_i^j are beliefs related to edges. They satisfy the marginalization condition

$$\forall (i,j) \in E : b_i^j(\sigma_i^j) = \sum_{\sigma_i \setminus \sigma_i^j} b_i(\sigma_i) = \sigma_{\sigma^j \setminus \sigma_i^j} b^j(\sigma^j), \tag{21}$$

and normalization condition

$$\forall (i,j) \in E : b_i^j(1) + b_i^j(0) = 1 \tag{22}$$

Now, the average energy will be

$$U_{bethe} = \sum_{(i,j) \in E} b_i^j(1)$$

as for all $b_i^j(0)$, the corresponding σ_i^j will be 0, else it will be 1.

Entropy will be

$$S_{bethe} = \sum_{(i,j)} \sum_{\sigma_i^j} b_i^j(\sigma_i^j) ln \, b_i^j(\sigma_i^j) - \sum_i \sum_{\sigma_i} b_i(\sigma_i) ln \, b_i(\sigma_i) - \sum_j \sum_{\sigma^j} b^j(\sigma^j) ln \, b^j(\sigma^j)$$

As explained in the previous section, we can write beliefs as entries of a bi-stochastic matrix $\beta = \beta_i^j = b_i^j(1)$ (here also this follows from marginalization (21) and normalization (22) conditions. E.g. $b_i(\sigma_i = (0\ 0\ 1\ 0)) = b_i^3(1)$).

So the entropy can be written as

$$\begin{aligned} S_{bethe}(\beta_i^j) &= \sum_{i,j} \Big(\beta_i^j ln\ \beta_i^j + (1-\beta_i^j) ln\ (1-\beta_i^j) \Big) \\ &- \sum_i \sum_j \beta_i^j ln\ \beta_i^j - \sum_j \sum_i \beta_i^j ln\ \beta_i^j \\ &= \sum_{i,j} \Big((1-\beta_i^j) ln\ (1-\beta_i^j) - \beta_i^j ln\ \beta_i^j \Big) \end{aligned} \tag{23}$$

This the bethe free energy is

$$G_{Bethe} = \sum_{i,j} \Big(\beta_i^j ln\ \frac{\beta_i^j}{p_i^j} - (1-\beta_i^j) ln\ (1-\beta_i^j) \Big) \tag{24}$$

The method of minimizing and finding the permanent is same as before. Although the two models look very different, they are equivalent. In fact, the bi-stochastic matrix computed by the PMRF model can directly be used in (24) to compute its correct minima.

5.3 Complexity and Bounds

Huang and Jebara experimentally shows in [5] that the belief propagation algorithm converges in constant number of iterations (with some tolerance). Vontobel proves theoretically in [7] that belief propagation converges in polynomial time and that the bethe free energy is a convex function in the bi-stochastic matrix β discussed above. Yedidia et. al. shows in [9] that fixed points of belief propagation correspond to stationary points of the Bethe free energy function. Gurvits shows in [4] that permanent is lower bounded by minima of bethe free energy. Anari and Rezaei proves in [1] that the permanent is approximated within a factor of $\sqrt{2}^n$ by computing the minima of bethe free energy. This implies that belief propagation efficiently approximates the permanent of non negative square matrices within tight bounds.

6 Application to Rectangular Permanent

6.1 Modeling the Rectangular Permanent

We use the same notation as in Section 5.1. Take a complete bipartite graph $K_{n,m}$ of $n+m$ elements where $n < m$. Let the nodes in the left partition be $X = (x_1, \dots x_n)$ and right partition be $Y = (y_1, \dots y_m)$. If a node x_i is in state k, we can say that x_i "chooses" the node y_k and vice versa. If node $x_i = j$ and $y_j = i$, we can say that they are "matched".

Here we will consider the set of all maximal matching instead of perfect matching as in the square case. The rectangular permanent can be viewed as the sum of weights of all the maximal matchings.

As opposed to the square case, in the rectangular case there will always be more than one y_j's that point to the same i. So for the old definition of ψ in Section 5.1, the product $\prod_{i,j} \psi_{ij}(x_i, y_j)$ will always be zero. Thus, to approximate rectangular permanent we need new definition of $\psi_{ij}(x_i, y_j)$, $\phi_i(x_i)$ and $\phi_j(y_j)$ such that the normalization term

$$Z = \sum_{x_1,\dots,x_n,y_1,\dots,y_m} \prod_{i,j} \psi_{ij}(x_i, y_j) \prod_i \phi_i(x_i) \prod_j \phi_i(y_j)$$

is the rectangular permanent (scaled by some factor).

We define the functions as follows

- $\phi_i(x_i) = p_{i,x_i}$
- $\phi_j(y_j) = 1$
- $\psi_{ij}(x_i, y_j) = I(\neg(x_i = j)\ or\ (y_j = i))$. $I(.)$ is 0 if the argument is false, and 1 if true. This will be used to check if the graph forms a maximal matching for a given assignment of X and Y.

Using the new definitions of $\psi_{ij}(x_i, y_j)$, $\phi_i(x_i)$ and $\phi_j(y_j)$, the normalization term Z will give the permanent of a $n \times m,\ (n \leq m)$ matrix scaled by n^{m-n}. This scaling factor comes from the fact that for each maximal matching, there are $(m-n)$ many y_j variables which are "free". So for each assignment of these $(m-n)$ variables, the weight of a particular maximal matching is over-counted.

The bethe free energy in this case will be

$$
\begin{aligned}
G_{Bethe} = &- \sum_{ij} \sum_{x_i,x_j} b_{ij}(x_i,y_j) ln\ \psi_{ij}(x_i,y_j)\phi_i(x_i)\phi_j(y_j) + \sum_{ij} \sum_{x_i,x_j} b_{ij}(x_i,y_j) ln\ b_{ij}(x_i,y_j) \\
&- (n-1) \sum_{i} \sum_{x_i} b_i(x_i) ln\ b_i(x_i) - (m-1) \sum_{j} \sum_{y_j} b_j(y_j) ln\ b_j(y_j) \\
&+ (n-1) \sum_{i} \sum_{x_i} b_i(x_i) ln\ \phi_i(x_i) + (m-1) \sum_{j} \sum_{y_j} b_j(y_j) ln\ \phi_j(y_j)
\end{aligned}
\tag{25}
$$

Note that as $\phi_j(y_j) = 1$, the term $(m-1) \sum_j \sum_{y_j} b_j(y_j) ln\ \phi_j(y_j)$ will be 0.

As explained in Section 3, belief propagation algorithm is a general algorithm and can be used to compute beliefs for any well defined graphical model. So, the method to compute beliefs remains same.

6.2 Explanation

The naïve way of approximating permanent of a rectangular matrix would be to approximate permanent of each $n \times n$ sub-matrix. It is trivial to see that this will take exponential time as there are $\binom{m}{n}$ sub-matrices.

Analogous to correspondence between permanent of square matrix and number of matchings in a bipartite graph with n elements in each partition, we can easily establish a relation between the permanent of a rectangular matrix and number of maximal matchings in a bipartite graph with n and m elements in each partition ($n \leq m$). A maximal matching is nothing but a perfect matching for a selected subset of n nodes from m nodes. As the number of subset of size n in a bipartite graph is $\binom{m}{n}$, the number of maximal matchings is same as the permanent of the adjacency matrix of the graph.

As it would be difficult to compute the bethe free energy (and by extension, run the belief propagation algorithm) by iterating over each maximal matching, we introduce a function $\psi_{ij}(x_i, y_j)$ which is 1 only if the assignment of the set of X and Y variables correspond to a maximal matching. But by doing this, we count each maximal matching $n^{(m-n)}$ times. This is because a maximal matching is identified by checking the assignment of n many Y variables. For each assignment of the rest $m-n$ variables, the same matching is counted. As there are $n^{(m-n)}$ many possible assignments of $m-n$ variables (because each variable in Y can take n values independent of each other), we get the number of

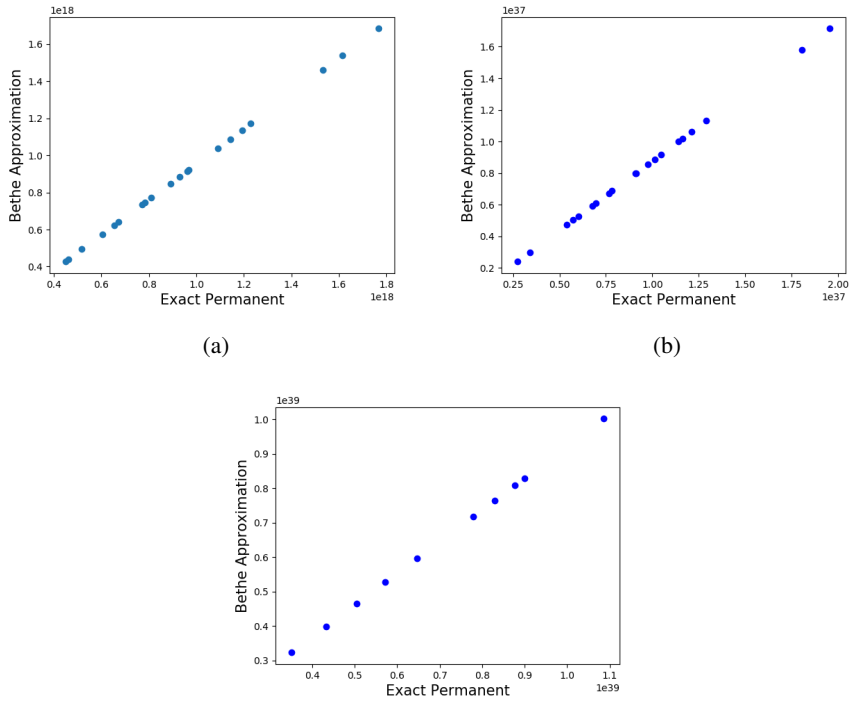


Figure 1: Bethe approximation of random (a) 5×10, (b) 10×15 and (c) 10×20 matrices with coefficients smaller than 1000.

maximal matchings scaled by $n^{(m-n)}$. As the factor is independent of the entries of the input matrix, it can be easily divided to get the weighted number of perfect matchings.

6.3 Experiments

Figure 1 shows relation between permanent and approximation of matrices of three different sizes. The approximated values are very close and the data points form a straight line at almost $45°$. The experiments are conducted on Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz and 8 GB DDR3 RAM. The algorithm was executed for a 250 iterations.

Figure 2 displays time (in sec) taken to approximate the permanent of three different sizes of matrices. The experiment was conducted on the same machine as in previous case.

Figure 3 shows the time (in sec) and number of iterations taken by $n \times 15$ matrices where $n \in [2 \ldots 14]$ to converge. We show a 2^n graph for reference, to infer that the time or number of iterations taken to converge does not grow exponentially. Although it

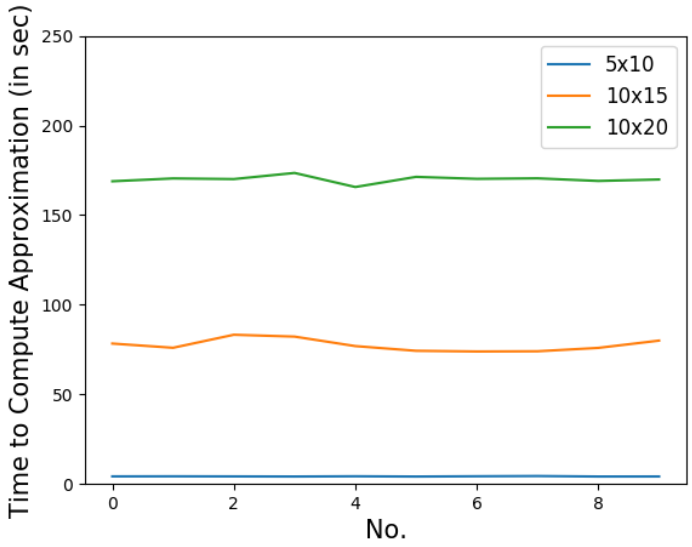


Figure 2: Time to compute bethe approximation of matrices of three fixed dimension

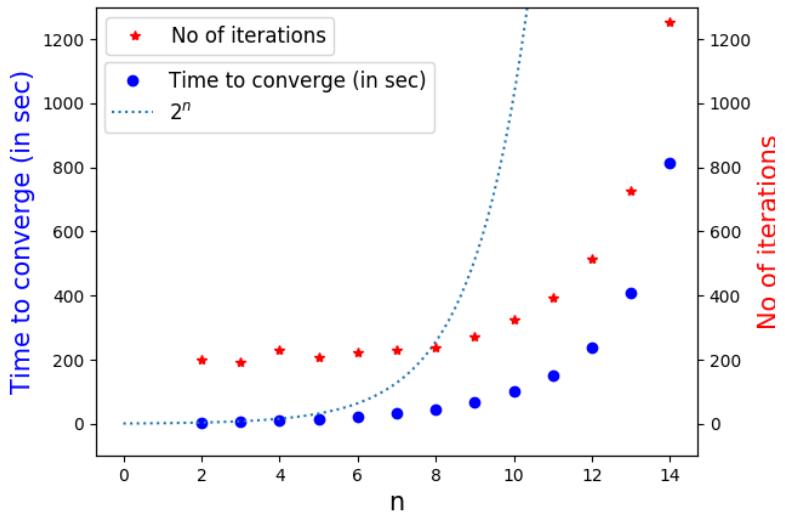


Figure 3: Time to compute bethe approximation of $n \times 15$ matrices

is known that the belief propagation converges in polynomial time for square matrices, no theoretical bound is known for rectangular matrices. Note that polynomial time convergence for rectangular matrices is non-trivial as each permanent of a rectangular matrix contains (exponentially many) permanents of square matrix. We conducted also conducted an experiment for $n \times 20$ matrices with similar result. The experiment was conducted on Google Colab VM with Intel(R) Xeon(R) CPU @ 2.30GHz and 12 GB RAM.

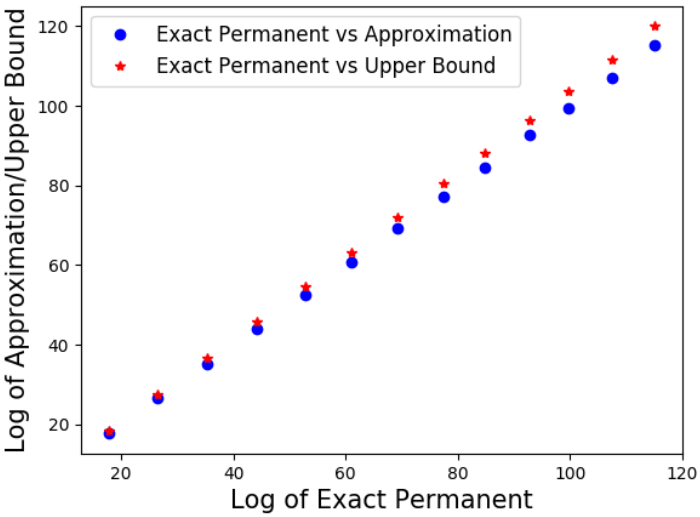


Figure 4: Log of permanent vs bethe approximation of $n \times 15$ matrices

Figure 4 shows log of permanent vs bethe approximation and the (conjectured) upper bound of $\sqrt{2}^n * Bethe$ of $n \times 15$ matrices for $n \in [2 \dots 14]$. The belief propagation ran for fixed number of iterations. (Logarithm is plotted as value of permanent vary largely as n grows). Note that the (conjectured) upper bound is dependent on n and not m (which is fixed and greater than n).

7 Conclusion and Future Work

We studied the existing probabilistic graphical algorithm to approximate the permanent of square matrices with positive entries and extended the algorithm to compute the permanent of rectangular matrices. The polynomial run-time and tight bounds for approximation have been proved for the square case. Observing the experimental data, we conjecture similar bounds for permanent of rectangular matrices. The next step is to prove these bounds theoretically.

References

[1] Anari, Nima, and Alireza Rezaei. "A Tight Analysis of Bethe Approximation for Permanent." *arXiv preprint arXiv:1811.02933* (2018).

[2] Chertkov, Michael, and Adam B. Yedidia. "Approximating the permanent with fractional belief propagation." *The Journal of Machine Learning Research* 14.1 (2013): 2029-2066.

[3] Forney Jr, G. David, and Pascal O. Vontobel. "Partition functions of normal factor graphs." *arXiv preprint arXiv:1102.0316* (2011).

[4] Gurvits, Leonid. "Unleashing the power of Schrijver's permanental inequality with the help of the Bethe approximation." *arXiv preprint arXiv:1106.2844* (2011).

[5] Huang, Bert, and Tony Jebara. "Approximating the permanent with belief propagation." arXiv *preprint arXiv:0908.1769* (2009).

[6] Pearl, Judea. "*Probabilistic reasoning in intelligent systems: networks of plausible inference. 1988.*" (1988).

[7] Vontobel, Pascal O. "The Bethe permanent of a nonnegative matrix." *IEEE Transactions on Information Theory* 59.3 (2013): 1866-1901.

[8] Watanabe, Yusuke, and Michael Chertkov. "Belief propagation and loop calculus for the permanent of a non-negative matrix." *Journal of Physics A: Mathematical and Theoretical* 43.24 (2010): 242002.

[9] Yedidia, Jonathan S., William T. Freeman, and Yair Weiss. "Understanding belief propagation and its generalizations." *Exploring artificial intelligence in the new millennium* 8 (2003): 236-239.