Tighter continuous relaxations for MAP inference in discrete MRFs: A survey

Hariprasad Kannan, Nikos Komodakis, Nikos Paragios

Tighter continuous relaxations for MAP inference in discrete MRFs: A survey

Hariprasad Kannan, Team DataShape, Inria Saclay, Palaiseau,
Nikos Komodakis, Ecole des Ponts, Champs-sur-Marne,
Nikos Paragios, CentraleSupelec, Gif-sur-Yvette

Abstract: *Continuous relaxations are central to* MAP *inference in discrete Markov random fields (*MRF*s). In these methods, the intractable discrete optimization problem is approximated by a continuous relaxation. The relaxation could be based on different approaches, with the linear programming (*LP*) relaxation being the most well studied. For continuous relaxations, the important considerations are efficiently solving the optimization formulations and improving the approximation quality of the relaxations. In this chapter, we focus on the latter topic, which is referred to as tighter continuous relaxations. We present a comprehensive survey of techniques to achieve a tighter* LP *relaxation, followed by a discussion of semidefinite programming (*SDP*) relaxation techniques. We discuss these topics by covering both theoretical motivations and algorithmic considerations. We hope this will be a ready reference for researchers working on tighter relaxations and for practitioners who want to model their problem more accurately and efficiently. The foundation for the theoretical aspects comes from the discrete optimization community through the topics of polyhedral combinatorics and relaxation hierarchies. The structure of discrete* MRF*s lends a special perspective to these classic topics and we will see how several works have explored this path. Over the years, research efforts in discrete* MRF*s have characterized graph topologies and/or clique energies that lead to accurate* MAP *inference. In recent years, there have been similar efforts, to better understand the settings and their accompanying guarantees, for the algorithms that tighten the continuous relaxations. We will end this chapter by giving a taste for this emerging research topic.*

Probabilistic graphical models describe a set of random variables in a compact manner by exploiting the structured nature of their interaction Koller & Friedman (2009), Wainwright & Jordan (2008). In this chapter, we focus on undirected graphical models, which are referred to as Markov random fields (MRFs) Blake et al. (2011). We have a graph $\mathcal{G} = (\mathcal{V}, \mathcal{C})$, where $\mathcal{V}$ denotes the set of nodes, which are the random variables and $\mathcal{C}$ denotes the set of cliques, which represent the interactions. In this chapter, the random variables take discrete values or labels. A clique composed of two nodes is also called an edge. A clique with more than two nodes is called a hyperedge or a higher-order clique. We will refer to cliques, edges and hyperedges interchangeably, where the meaning will be clear based on the context. An important computation that we seek is finding the labelling of all the random variables, which will maximize the joint probability distribution,

$$P(\boldsymbol{X} = \boldsymbol{x}) = \prod_{i \in \mathcal{V}} \Psi_i(x_i) \prod_{c \in \mathcal{C}} \Psi_c(\boldsymbol{x}_c) = \exp\Big(- \sum_{i \in \mathcal{V}} \theta_i(x_i) - \sum_{c \in \mathcal{C}} \theta_c(\boldsymbol{x}_c) \Big). \tag{1}$$

Here, $\boldsymbol{X}$ denotes the complete set of n discrete variables $X_1, ..., X_n$. The set of variables corresponding to a clique c is indicated by $\boldsymbol{X}_c$. Without loss of generality, we assume that all variables take labels from a common set of size l. A particular realization (labelling) of X_i is denoted by x_i. Thus, the labelling of a clique, $\boldsymbol{x}_c$, is a vector of node labels. Thus, a clique could take one of $l^{|c|}$ labellings. Depending on the application, we define appropriate potential functions over nodes and cliques. They take the form $\Psi_i(x_i) = \exp\big(-\theta_i(x_i)\big)$ and $\Psi_c(\boldsymbol{x}_c) = \exp\big(-\theta_c(\boldsymbol{x}_c)\big)$, where $\theta_i(x_i)$ and $\theta_c(\boldsymbol{x}_c)$ denote the energies (also, called potentials) corresponding to node i taking label x_i and clique c taking label $\boldsymbol{x}_c$, respectively. The joint probability distribution factorizing according to the nodes and the cliques is a fundamental property of MRFs. Based on (1), maximum a posteriori inference or MAP inference, computes the labelling $\boldsymbol{x}$ that minimizes the following energy,

$$E(\boldsymbol{x};\boldsymbol{\theta}) \triangleq \sum_{i\in\mathcal{V}} \theta_i(x_i) + \sum_{c\in\mathcal{C}} \theta_c(\boldsymbol{x}_c). \tag{2}$$

However, except for some special graph topologies and/or clique potentials, this problem is NP-hard Shimony (1994), Li et al. (2016). Thus, there has been considerable research effort to develop approximate inference algorithms for MRFs. Several approaches exist to minimize this energy efficiently and approximately: especially methods based on graph cuts, belief propagation and continuous relaxation. Refer Szeliski et al. (2008), Blake et al. (2011), Kappes et al. (2015) for a good survey about various methods. Among these methods, continuous relaxation based approaches have led to state-of-the-art algorithms and also, provide a theoretical foundation to the topic of MAP inference Wainwright et al. (2005), Komodakis et al. (2016).

First, we observe that the energy in (2) has a linear form, with respect to the energies, $\theta_i(x_i)$'s and $\theta_c(\boldsymbol{x}_c)$'s. Thus, MAP inference can be equivalently represented as an intractable integer linear program (ILP) as follows,

$$\underset{\boldsymbol{\phi}}{\text{minimize}} \quad \sum_i \sum_{x_i} \theta_i(x_i)\phi_i(x_i) + \sum_c \sum_{\boldsymbol{x}_c} \theta_c(\boldsymbol{x}_c)\phi_c(\boldsymbol{x}_c) \tag{3a}$$

$$\text{subject to} \quad \sum_{\boldsymbol{x}_{c\setminus i}} \phi_c(\boldsymbol{x}_c) = \phi_i(x_i), \quad \forall (i,c) : i \in c,\ \forall x_i \tag{3b}$$

$$\begin{aligned} &\sum_{x_i} \phi_i(x_i) = 1, \quad \phi_i(x_i) \in \{0,1\} \quad \forall i \in \mathcal{V} \\ &\sum_{\boldsymbol{x}_c} \phi_c(\boldsymbol{x}_c) = 1, \quad \phi_c(\boldsymbol{x}_c) \in \{0,1\} \quad \forall c \in \mathcal{C}. \end{aligned} \tag{3c}$$

Here, for each node-label pair (i, x_i), there is an indicator variable $\phi_i(x_i)$. Similarly, for each clique-label pair (c, $\boldsymbol{x}_c$), there is an indicator variable $\phi_c(\boldsymbol{x}_c)$. Let us define a vector $\boldsymbol{\phi}$ that is obtained by stacking all the $\phi_i(x_i)$'s and $\phi_c(\boldsymbol{x}_c)$'s together. The normalization constraints, (3c), ensure that for a given labelling of all the nodes, for each node i, only one of $\phi_i(x_i)$ will be set to 1 and for each clique c, only one of $\phi_c(\boldsymbol{x}_c)$ will be set to 1. The node based indicator variables, $\phi_i(x_i)$'s, are enough to denote the labelling of the graphical model. However, we need the clique based indicator variables, $\phi_c(\boldsymbol{x}_c)$'s, in order to model our optimization problem. Shortly, we will see that this approach leads to a relaxation of the integer linear program that is convex in nature. Since, nodes and cliques are having separate indicator variables, we have to make sure that the labelling of the nodes and the labelling of the cliques agree. We ensure this through the marginalization constraints given by (3b). We will see how the summation $\sum_{\boldsymbol{x}_{c\setminus i}} \phi_c(\boldsymbol{x}_c)$ is performed. For a given clique c and node $i \in c$, there will be a set of clique labellings $\boldsymbol{x}_c$'s, where the label of node i is fixed to x_i. We will denote this set as $\boldsymbol{x}_{c\setminus i}$. With a slight abuse of notation, it is a more compact way of denoting the set, $\boldsymbol{x}_c : \boldsymbol{x}_c(i) = x_i$. Since, we will come across such summations often, a shorter notation is preferable. The summation of $\phi_c(\boldsymbol{x}_c)$'s is performed over this set and the constraint says that the sum should match with $\phi_i(x_i)$.

Using the toy graphical model given in Fig. 1, we will further understand these variables and constraints. In this graphical model, each clique is made of three nodes, i.e., $a = \{1, 2, 3\}$ and $b = \{2, 3, 4\}$. We will assume that each node could take one of three labels, $\{x_1, x_2, x_3\}$. Thus, both the $\phi(\boldsymbol{x}_a)$'s and the $\phi(\boldsymbol{x}_b)$'s consist of 27 elements. Consider the case where node 3 takes label x_2, i.e., $\phi_3(x_1) = 0$, $\phi_3(x_2) = 1$, $\phi_3(x_3) = 0$. We see that the normalization constraint (3c) is satisfied. Node 3 is part of both cliques a and b. There are nine elements in each set of $\phi_a(\boldsymbol{x}_a)$'s and $\phi_b(\boldsymbol{x}_b)$'s, which correspond to the case of node 3 taking a particular label. We denote these sets by $\boldsymbol{x}_{a\backslash 3}$ and $\boldsymbol{x}_{b\backslash 3}$, respectively. According to the marginalization constraint (3b), the summation of $\phi_a(\boldsymbol{x}_a)$'s (resp. $\phi_b(\boldsymbol{x}_b)$'s) over $\boldsymbol{x}_{a\backslash 3}$ (resp. $\boldsymbol{x}_{b\backslash 3}$) should equal 1.

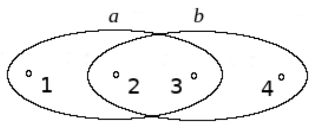

Figure 1: A toy graphical model with nodes 1, 2, 3 and 4 and cliques a and b.

In the quest for efficient algorithms for approximate MAP inference, one could derive continuous relaxation based formulations for the ILP given in (3). There have been relaxations based on linear programming (LP), quadratic programming (QP), semidefinite programming (SDP) and second-order cone programming (SOCP). Kumar et al. (2009) has presented an excellent analysis of these various relaxations. A continuous relaxation is naturally going to return an approximate solution and it is worthwhile to *tighten* the relaxation in order to attain more accurate results. In this chapter, we will survey several important aspects of this topic. First, we will see various algorithms that could help to achieve a tighter LP relaxation. Second, we will see inference algorithms based on the SDP relaxation that could offer better results than the LP relaxation for some problem instances. We will conclude by indicating research efforts to characterize graph topologies and clique potentials that govern the tightness of these relaxations.

1 LP relaxation

In this section, we will see how a tractable optimization problem could be obtained by LP relaxation and present the core concepts of marginal and local polytopes. First, we observe that the ILP in (3) could be equivalently expressed as an intractable linear program. In this LP, the objective is expressed in the same way as (3a) and the decision variables, $\phi_i(x_i)$'s and $\phi_c(\boldsymbol{x}_c)$'s, take continuous values within a polytope called the *marginal* polytope, denoted as $\mathcal{M}(\mathcal{G})$. The marginal polytope is the convex hull of the feasible set of the ILP (3). Thus, the vertices of $\mathcal{M}(\mathcal{G})$ are points taking $\{0, 1\}$ integer values, satisfying the constraints, (3b) and (3c). The optimal solution of any linear program consists of one of the vertices of the polytope over which it is optimized. Thus, the solution of this LP will automatically belong to the feasible set of (3). However, one needs an exponential number of constraints to represent the facets of $\mathcal{M}(\mathcal{G})$, thus this LP is as hard the original ILP. In order to obtain a more efficient situation, we could optimize over a feasible region that is simpler than the marginal polytope. In order to characterize such tractable feasible regions, we need to understand more about the marginal polytope. The marginal polytope contains those $\boldsymbol{\phi}$'s for which the $\phi_i(x_i)$'s and $\phi_c(\boldsymbol{x}_c)$'s will be valid marginals, with respect to some joint probability distribution over all the random variables. This property can be stated as follows,

$$\mathcal{M}(\mathcal{G}) = \Big\{ \boldsymbol{\phi} \;\mid\; \exists\, P \in \mathscr{P}(\{0,1,...,l-1\}^{|\mathcal{V}|}) \\ \text{s.t. } \phi_i(x_i) = P(X_i = x_i)\; \forall i \in \mathcal{V},\; \phi_c(\boldsymbol{x}_c) = P(\boldsymbol{X}_c = \boldsymbol{x}_c)\; \forall c \in \mathcal{C} \Big\} \tag{4}$$

where $\mathscr{P}(\{0,1,...,l-1\}^{|\mathcal{V}|})$ is the set of all valid probability distributions defined over the finite set $\{0,1,...,l-1\}^{|\mathcal{V}|}$. It takes an exponential number of linear constraints to ensure that a vector $\boldsymbol{\phi}$ belongs to $\mathcal{M}(\mathcal{G})$. The main idea behind LP relaxations is to enforce a tractable number of constraints, which will lead to simpler polytopes to optimize over. The most common LP relaxation takes the following form,

$$\underset{\boldsymbol{\phi}}{\text{minimize}} \quad \sum_i \sum_{x_i} \theta_i(x_i)\phi_i(x_i) + \sum_c \sum_{\boldsymbol{x}_c} \theta_c(\boldsymbol{x}_c)\phi_c(\boldsymbol{x}_c) \tag{5a}$$

$$\text{subject to} \quad \sum_{\boldsymbol{x}_{c \setminus i}} \phi_c(\boldsymbol{x}_c) = \phi_i(x_i), \quad \forall (i,c) : i \in c,\; \forall x_i \tag{5b}$$

$$\sum_{x_i} \phi_i(x_i) = 1, \quad \phi_i(x_i) \geq 0 \quad \forall i \in \mathcal{V} \\ \sum_{\boldsymbol{x}_c} \phi_c(\boldsymbol{x}_c) = 1, \quad \phi_c(\boldsymbol{x}_c) \geq 0 \quad \forall c \in \mathcal{C}. \tag{5c}$$

The marginalization constraints (5b) and the normalization constraints (5c) should be understood similar to (3b) and (3c), respectively. In this case, we work with continuous valued variables. For this LP, the constraint polytope is called the *local* polytope because the linear constraints are marginalization constraints (5b) that are only enforced locally with respect to the cliques. This represents a substantially lesser number of constraints compared to the marginal polytope. We will denote the local polytope as $\mathcal{L}(\mathcal{G})$. The local polytope is an outer bound on the marginal polytope. More precisely, it will contain all the points that satisfy (4), since, they automatically satisfy (5b) and (5c), also. Further, $\mathcal{L}(\mathcal{G})$ contains other points that satisfy constraints (5b) and (5c) but not (4). The points contained within the local polytope are called *pseudomarginals*. This is inspired by the fact that the marginal polytope contains true marginals. All the $\{0,1\}$ valued vertices of the marginal polytope are also vertices of the local polytope. Apart from these, there are several fractional valued vertices in the local polytope. It is difficult to visualize the marginal and local polytopes. Fig. 2 represents these polytopes in an abstract way. Suppose the LP relaxation returns a integer valued vertex, then we have the guarantee that we have recovered the exact MAP labelling of the graphical model. This is an important guarantee of the LP relaxation based approach Wainwright et al. (2005). However, the LP relaxation could return a fractional valued vertex. The desired MAP labelling is obtained by rounding these fractional values. Tightening the polytope over which the LP is optimized could lead to a fractional solution that could give a better integral solution after rounding or if possible an integer valued solution could be obtained. Since, the integral points belonging to the local polytope correspond to the vertices of the marginal polytope, an integral solution obtained through a tighter polytope will correspond to a vertex of the marginal polytope. We have already seen that the MAP solution will correspond to a vertex of the marginal polytope. Thus, we would have obtained the exact MAP solution, if we are lucky with our tighter relaxation.

So far, we have discussed about two linear programs: the intractable LP that is exactly equivalent to the ILP given in (3) and the LP relaxation given in (5). The formulation given in (5) looks similar to the ILP, except for one difference. The integer values constraints in (3c) have been relaxed to continuous values (5c). However, this simple difference leads to a much more tractable problem.

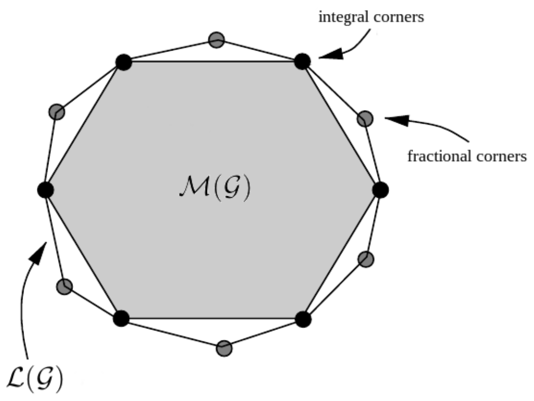

Figure 2: A cartoonish illustration of the relationship between the marginal polytope $\mathcal{M}(\mathcal{G})$ and the local polytope $\mathcal{L}(\mathcal{G})$. All the vertices of $\mathcal{M}(\mathcal{G})$ are also vertices of $\mathcal{L}(\mathcal{G})$. $\mathcal{L}(\mathcal{G})$ also has fractional vertices. The fact that $\mathcal{L}(\mathcal{G})$ has lesser number of facets than $\mathcal{M}(\mathcal{G})$ is difficult to visualize. *Image courtesy Wainwright & Jordan (2008).*

1.1 Tightening the polytope

Now, we will see how looser polytopes like the local polytope could be tightened to obtain more accurate results. Tighter relaxations for MAP inference are intimately related to *polyhedral combinatorics* Schrijver (2003). According to the combinatorial optimization community, a sequence of tighter LP relaxations called the Sherali-Adams hierarchy could approximate the ILP (3) to arbitrary precision Laurent (2003). In each level of the hierarchy, we obtain a polytope $\mathcal{SA}_r(\mathcal{G})$ by imposing marginalization constraints over all possible *clusters* of size r. A cluster is the most general data structure that we will come across in our discussions. In fact, any subset of nodes could be a cluster. In order to define the new set of marginalization constraints, we define a new set of variables, $\boldsymbol{\tau}_s$ for each cluster s. The marginalization constraints take the same form as (5b), with the summation computed over the cluster, while the label is held fixed for subsets (like cliques or individual nodes) of the cluster. The highest level of the hierarchy, $\mathcal{SA}_{treewidth}(\mathcal{G})$, corresponds to the marginal polytope and the size of the clusters that is required at this level is equal to the treewidth of the graph. Thus the highest level has intractable complexity. Even lower levels like $r = 2$ or 3 are computationally expensive. Since, it involves imposing constraints on all possible pairs or triplets of nodes. In practice, we consider smaller sets of clusters, like only pairs of nodes that correspond to edges of the graph could be considered. This is exactly what leads to the local polytope. In section (6), we will see more about efforts to understand such hierarchies and their guarantees with respect to graph topology and/or clique potentials Wainwright & Jordan (2004*b*), Thapper & Živný (2016), Weller et al. (2016), Rowland et al. (2017). For now, we will look at algorithms to tighten the relaxation, which work well in practice, while accompanied by a reasonable level of theoretical justification. We will see some of the pioneering works that algorithmically addressed the tighter relaxation problem Sontag & Jaakkola (2007), Komodakis & Paragios (2008), Sontag et al. (2008), Werner (2008). Before we go further, it is good to know that tightening the relaxation based on clusters has been pursued in the statistical physics community in the name of Kikuchi clustering (Wainwright & Jordan 2008, §4) and has been used within belief propagation algorithms Yedidia et al. (2005).

When one talks about tightening a polytope, the natural algorithmic approach that comes to mind is a *cutting-plane* method. In a cutting-plane method, one progressively tightens the polytope over which the optimization is performed by adding more constraints. This is the underlying principle behind Sontag & Jaakkola (2007), which starts with a loose relaxation of the marginal polytope, like the local polytope and iteratively adds constraints. In the previous paragraph, we came across marginalization constraints that could be enforced with respect to clusters of larger and larger sizes in order to tighten the polytope. In the combinatorial optimization literature, other valid constraints are also possible that lead to tighter relaxations. Sontag & Jaakkola (2007) worked with *cycle inequalities* Deza & Laurent (2009), which could be enforced for *pairwise binary* graphs, $\mathcal{G}_{\{0,1\}}$. For any labelling of such a graph, when one traverses a cycle of edges, one will encounter a change of node labelling, only an even (including zero) number of times. This fact leads to the following constraint for a cycle of edges C and any subset $F \subseteq C$ such that $|F|$ is odd,

$$\sum_{(i,j)\in C\setminus F} \Big(\phi_{ij}(1,0) + \phi_{ij}(0,1)\Big) + \sum_{(i,j)\in F} \Big(\phi_{ij}(0,0) + \phi_{ij}(1,1)\Big) \geq 1. \tag{6}$$

Here, the $\phi_{ij}(\cdot)$ are (pseudo)marginals corresponding to the edge ij and a labelling of the edge is indicated by a pair of binary values. The ϕ_{ij}'s will be integer valued only if they belong to a vertex of the marginal polytope. Otherwise, they could take fractional values. These constraints hold for any marginal $\boldsymbol{\phi} \in \mathcal{M}(\mathcal{G}_{\{0,1\}})$ and any cycle C of the pairwise binary graph $\mathcal{G}_{\{0,1\}}$. We have seen that for any Markov random field, one of the vertices of the marginal polytope corresponds to an optimal labelling of the nodes. While optimizing over a looser polytope (like the local polytope), the cutting-plane algorithm strives to add constraints in order to reach the integer valued vertex of the marginal polytope. The pseudomarginals in an outer bounding polytope like $\mathcal{L}(\mathcal{G})$ need not satisfy the above cycle inequality constraints. Thus, enforcing these constraints will lead to a tighter polytope. The important step in any cutting-plane algorithm is about finding the constraints to add in an efficient manner, which is called the *separation* problem. In the algorithm of Sontag & Jaakkola (2007), the cycle that most violates the cycle inequality (6) could be selected. Even though the number of cycles in a general graph is exponentially large, it is possible to identify the violated cycle inequalities in polynomial time. In fact, it is of complexity $O(n^2 \text{log} n + n|\mathcal{C}|)$ based on Dijkstra's shortest path algorithm along with a Fibonacci heap data structure.

So far, we have discussed an approach which will work for pairwise binary graphs. This will be a recurring setting in the study of tighter relaxations. Generally, extending these ideas to nodes taking multiple labels and also, to graphs with higher-order cliques lead to looser bounds, computationally more expensive algorithms and larger problem sizes due to introduction of auxiliary variables. For example, Sontag & Jaakkola (2007) has shown how to extend the cycle inequalities constraints to the multi-label scenario and also, to non-pairwise graphs. Again, they take inspiration from polyhedral combinatorics and propose an approach based on *projection* and *aggregation*. Consider a pairwise graph, with multi-label nodes. The set of labels of a node i, which we will denote as $\boldsymbol{l}_i$, can be partitioned into two sets. For ease of exposition, we will assume that all nodes take the same number of labels l. For l labels, there are $2^{(l-1)} - 1$ possible partitions and let π_i^s denote one such partition. Let $\boldsymbol{\pi}_i$ be a collection of partitions for node i. We define a binary *projection* *graph*, $\mathcal{G}^\pi = (\mathcal{V}^\pi, \mathcal{C}^\pi)$, where there is a node in $\mathcal{V}^\pi$ $\forall \pi_i^s \in \boldsymbol{\pi}_i$, $\forall i \in \mathcal{V}$ and there is an edge between π_i^s and π_j^t if $(i,j) \in \mathcal{C}$. If $\boldsymbol{\pi}_i$ consists of all possible partitions for all nodes $i \in \mathcal{V}$, then $\mathcal{G}^\pi$ is called the *full projection* graph. If each node's set of labels is partitioned only once, i.e., $|\boldsymbol{\pi}_i| = 1$ $\forall i$, we have the *single projection* graph. In the *l-projection* graph, each $\boldsymbol{\pi}_i$ consists of one versus the rest partitions for each of the l labels. Given a marginal with respect to the original marginal polytope, $\boldsymbol{\phi} \in \mathcal{M}(\mathcal{G})$, its values could be aggregated to define marginals with respect to the marginal polytope of the binary projection graph. For this marginal polytope, it is possible to define cycle inequalities as follows,

$$\begin{aligned} &\sum_{(i,j)\in c\setminus f} \phi_{ij}^\pi(x_i' \neq x_j') + \sum_{(i,j)\in f} \phi_{ij}^\pi(x_i' = x_j') \geq 1 \\ \text{where} \quad &\phi_{ij}^\pi(x_i' \neq x_j') = \sum_{\substack{x_i \in \boldsymbol{l}_i, x_j \in \boldsymbol{l}_j: \\ \pi_i(x_i) \neq \pi_j(x_j)}} \phi_{ij}(x_i, x_j) \\ &\phi_{ij}^\pi(x_i' = x_j') = \sum_{\substack{x_i \in \boldsymbol{l}_i, x_j \in \boldsymbol{l}_j: \\ \pi_i(x_i) = \pi_j(x_j)}} \phi_{ij}(x_i, x_j) \end{aligned} \tag{7}$$

where x_i' takes binary values $\{0,1\}$, the projection operation $\pi_i(x_i)$ maps a node label, x_i, to one of these binary values, C is a cycle with respect to the original graph $\mathcal{G}$ and any $F \subseteq C$ such that $|F|$ is odd. These equations are easy to understand for single projection graphs but could be derived for the full projection and l-projection cases, also. Thus, depending on whether the original graph has binary variables or multi-label variables, we obtain inequalities (6) or (**??**), respectively. Based on these cycle inequalities we could obtain cutting-planes to tighten the polytope. Thus, the algorithm given in Sontag & Jaakkola (2007) proceeds as follows. The first iteration involves solving an LP over a loose polytope, like the LP in (5). Then in each iteration, given the optimal pseudomarginals, cutting-planes (upto to n of them) are added and the tighter LP is solved to optimality. This is repeated until no cutting-planes are found. Even though there are specialized solvers to solve the LP relaxation in (5), specialized efficient solvers are not yet available to optimize the linear objective over a polytope which also has cycle inequalities constraints. For the binary case, it is possible to optimize using the Frank-Wolfe algorithm, using linear or semidefinite programming to optimize over the tighter polytope. For the multi-label case, a commercial solver like CPLEX is needed. Note, even though only the single projection graph leads to a surjective projection between the marginals of the multi-label graph to the marginals of the binary graph, it is preferable to work with the full projection graph. This is because we need several single projection graphs to obtain the same tightness as the full projection graph. Similarly, the full projection graph is tighter than the l-projection graph. Sontag & Jaakkola (2007) have shown experimental evidence, also, regarding better tightness of the full projection graph. Also, in order to extend this work to higher-order graphs, we need to create an intermediate pairwise graph which involves auxiliary variables corresponding to the higher-order clique potentials. (Wainwright & Jordan 2008, §E.3) presents one way to convert higher-order cliques to pairwise edges by adding auxiliary variables. Thus, we see how dealing with multi-label nodes and higher-order cliques lead to the addition of several auxiliary variables.

Even though the above work greatly helps to improve our understanding about tighter relaxations, it is not scalable. This is because we are not able to develop specialized algorithms for solving the LP relaxation with additional cycle inequalities based constraints. We are forced to use standard off-the-shelf LP solvers. The lack of scalability of standard LP solvers is well-known in the MAP inference literature Yanover et al. (2006). This is because they do not exploit the structure of the graphical model. Working in the dual space, leads to exploitation of the graph structure. Hence, there have been several scalable algorithms to solve the LP relaxation that are based on dual decomposition Komodakis et al. (2011), Sontag et al. (2011). We will see how by recalling the dual presented in Komodakis et al. (2011). This is derived for the LP relaxation shown in (5) based on the decomposition of the original graph into a set of subgraphs. Based on such a decomposition, (5) can be equivalently written as,

$$\underset{\phi_i,\phi_c,\phi_{\mathcal{S}i},\phi_{\mathcal{S}c}}{\text{minimize}} \sum_{\mathcal{S}\in\mathcal{S}(\mathcal{G})} \left(\sum_{i\in\mathcal{S}} \sum_{x_i} \theta_{\mathcal{S}i}(x_i)\phi_{\mathcal{S}i}(x_i) + \sum_{c\in\mathcal{S}} \sum_{\boldsymbol{x}_c} \theta_{\mathcal{S}c}(\boldsymbol{x}_c)\phi_{\mathcal{S}c}(\boldsymbol{x}_c) \right) \tag{8a}$$

$$\left.\begin{array}{l} \text{subject to } \sum_{x_{c\setminus i}} \phi_{\mathcal{S}c}(\boldsymbol{x}_c) = \phi_{\mathcal{S}i}(x_i), \quad \forall (i,c) : c\in\mathcal{S}, i\in c \\ \sum_{x_i} \phi_{\mathcal{S}i}(x_i) = 1, \quad \forall i\in\mathcal{S}; \quad \sum_{\boldsymbol{x}_c} \phi_{\mathcal{S}c}(\boldsymbol{x}_c) = 1, \quad \forall c\in\mathcal{S} \\ \phi_{\mathcal{S}i}(x_i) \geq 0 \ \ \forall i\in\mathcal{S}; \quad \phi_c(\boldsymbol{x}_c) \geq 0 \ \ \forall c\in\mathcal{S} \end{array}\right\} \forall\mathcal{S}\in\mathcal{S}(\mathcal{G}) \tag{8b}$$

$$\phi_{\mathcal{S}i}(x_i) = \phi_i(x_i) \quad \forall i\in\mathcal{V},\ \forall x_i; \quad \phi_{\mathcal{S}c}(\boldsymbol{x}_c) = \phi_c(\boldsymbol{x}_c) \quad \forall c\in\mathcal{C},\ \forall \boldsymbol{x}_c. \tag{8c}$$

Here $\mathcal{S}(\mathcal{G})$ is the set of subgraphs over which the original graph is decomposed. The subgraphs should be chosen such that all nodes and cliques of the original graph are covered by at least one subgraph. Also, each subgraph should share at least one node or clique with at least one other subgraph. The potentials $\theta_i^{\mathcal{S}}(x_i)$ and $\theta_c^{\mathcal{S}}(\boldsymbol{x}_c)$ are chosen such that $\sum_{\mathcal{S}\in\mathcal{S}(i)} \theta_i^{\mathcal{S}}(x_i) = \theta_i(x_i) \ \ \forall i\in\mathcal{V}$, $\sum_{\mathcal{S}\in\mathcal{S}(c)} \theta_c^{\mathcal{S}}(\boldsymbol{x}_c) = \theta_c(\boldsymbol{x}_c)$, $\forall c\in\mathcal{C}$. A simple way to achieve this is by setting the potentials as follows: $\theta_i^{\mathcal{S}}(x_i) = \frac{\theta_i(x_i)}{|\mathcal{S}(i)|} \ \forall\mathcal{S}\in\mathcal{S}(i)$ and $\theta_c^{\mathcal{S}}(\boldsymbol{x}_c) = \frac{\theta_c(\boldsymbol{x}_c)}{|\mathcal{S}(c)|} \ \forall\mathcal{S}\in\mathcal{S}(c)$. The consensus constraints (8c) ensure that this formulation is equivalent to (5). We could derive the dual by taking the consensus constraints (8c) into the objective and it takes the following form,

$$\max_{\boldsymbol{\delta}}. \sum_{\mathcal{S}\in\mathcal{S}(\mathcal{G})} \min_{\phi_{\mathcal{S}i},\phi_{\mathcal{S}c}}. \left(\sum_{i\in\mathcal{S}} \sum_{x_i} (\theta_{\mathcal{S}i}(x_i)+\delta_{\mathcal{S}i}(x_i))\phi_{\mathcal{S}i}(x_i) + \sum_{c\in\mathcal{S}} \sum_{\boldsymbol{x}_c} (\theta_{\mathcal{S}c}(\boldsymbol{x}_c)+\delta_{\mathcal{S}c}(\boldsymbol{x}_c))\phi_{\mathcal{S}c}(\boldsymbol{x}_c) \right) \tag{9a}$$

$$\left.\begin{array}{l} \text{subject to } \sum_{x_{c\setminus i}} \phi_{\mathcal{S}c}(\boldsymbol{x}_c) = \phi_{\mathcal{S}i}(x_i), \quad \forall (i,c) : c\in\mathcal{S}, i\in c \\ \sum_{x_i} \phi_{\mathcal{S}i}(x_i) = 1, \quad \forall i\in\mathcal{S}; \quad \sum_{\boldsymbol{x}_c} \phi_{\mathcal{S}c}(\boldsymbol{x}_c) = 1, \quad \forall c\in\mathcal{S} \\ \phi_{\mathcal{S}i}(x_i) \geq 0 \ \ \forall i\in\mathcal{S}; \quad \phi_c(\boldsymbol{x}_c) \geq 0 \ \ \forall c\in\mathcal{S} \end{array}\right\} \forall\mathcal{S}\in\mathcal{S}(\mathcal{G}) \tag{9b}$$

$$\sum_{\mathcal{S}:i\in\mathcal{S}} \delta_{\mathcal{S}i}(x_i) = 0, \quad \forall i\in\mathcal{V},\ \forall x_i; \quad \sum_{\mathcal{S}:c\in\mathcal{S}} \delta_{\mathcal{S}c}(\boldsymbol{x}_c) = 0, \quad \forall c\in\mathcal{C},\ \forall \boldsymbol{x}_c. \tag{9c}$$

Here, $\delta_{\mathcal{S}i}(x_i)$ are the dual variables with respect to the nodes and there is one such dual variable for each subgraph $\mathcal{S}$, each node $i\in\mathcal{S}$ and for each label x_i. $\delta_{\mathcal{S}c}(\boldsymbol{x}_c)$ are the dual variables with respect to the cliques and there is one such dual variable for each subgraph $\mathcal{S}$, each clique $c\in\mathcal{S}$ and for each possible clique labelling $\boldsymbol{x}_c$. The constraints given in (9c) ensure that we eliminate the global variables, $\phi_i(x_i)$'s and $\phi_c(\boldsymbol{x}_c)$'s, from the objective. We work only with the subgraph based variables, $\phi_{\mathcal{S}i}(x_i)$'s and $\phi_{\mathcal{S}c}(\boldsymbol{x}_c)$'s and the objective decomposes over the subgraphs. In other words, the minimization subproblems that we observe in the objective (9a) are with respect to individual subgraphs. Usually, the subgraphs are the individual pairwise edges or cliques or *hypertrees* of the original graphical model. A hypertree is a more general way of looking at a tree. It is a tree which could be made of hyperedges, not only pairwise edges. Hence, a tree made of only pairwise edges is also a hypertree. Note that nodes, edges and cliques are instances of hypertrees, too. The convenience with hypertrees is that the above minimization could be performed exactly and efficiently. This is because the minimization subproblem in a subgraph corresponds to performing MAP inference within a graph of the same topology as the subgraph and with suitably defined potentials. A fundamental result in graphical model is that we can perform MAP inference in hypertrees, exactly and efficiently Koller & Friedman (2009). In general, it is possible to decompose the graph according to subproblems that are not hypertrees. We will discuss more about this aspect towards the end of the next section. At this point, a natural question comes to mind. What happens to the LP relaxation if the original graphical model is a hypertree, i.e., it has no loops? It is not only efficient to perform MAP inference efficiently in a hypertree but in fact, the local polytope for such a graphical model is tight. Further, there are other graphical models where the local polytope is tight. A very important case is when the clique potentials are *submodular* Blake et al. (2011). Submodular clique potentials are an example of *attractive* potentials. An attractive clique potential encourages the nodes to take similar labels. Specifically, for pairwise binary graphs, submodular clique potentials satisfy the following condition,

$$\theta_{ij}(0,1) + \theta_{ij}(1,0) \geq \theta_{ij}(0,0) + \theta_{ij}(1,1) \tag{10}$$

from which we can see that similar labels are encouraged, when we try to minimize the energy (2). Another example of a graphical model for which the local polytope is tight, appears in the problem of maximum weight matching in a bipartite graph, which can be formulated as a MAP inference problem Schrijver (2003).

So far, we have seen three types of data structures: clusters, cycles and hypertrees. They are all subgraphs of the original graphical model. This will be a common theme to several ideas discussed in this chapter: tightening the relaxation through formulations involving subgraphs of the graphical model. These subgraphs will have more structure compared to hypertrees. At the same time, these subgraphs will have a structure that lends itself to more efficient computation. Just like hypertrees have an efficient MAP inference algorithm, there are other graph topologies and/or clique potentials that lead to efficient computation. We will see more of such special subgraphs in the coming sections and discuss insights and algorithms based on them. Even though we will encounter different graph topologies in practical problems, as a first step, we recommend the reader to imagine a two-dimensional grid graph (a planar graph, where nodes are arranged in the form of rows and columns) and visualize structures like clusters, cycles and higher-order cliques on such a graph. Grid graphs are common in computer vision problems like stereo depth estimation and image segmentation. Kappes et al. (2015) is a good starting point to explore more about various datasets for MAP inference in MRFs.

2 Cluster Pursuit Algorithms

We will now look further into cluster based methods and we will describe algorithms that achieve efficiency by working in the dual space. We have already mentioned relaxation hierarchies like Sherali-Adams that involve a series of tighter problem levels. At each level, marginalization constraints are enforced based on an exhaustive set of clusters of a particular size. As mentioned in section (1.1), a cluster could be any subset of nodes of the graph. In practice, we will consider connected subsets of nodes (formed by the union of a subset of cliques) as clusters. Let us see more clearly what it means to add cluster based marginalization constraints. Suppose, we have a graph, with pseudomarginals $\phi_i(x_i)$'s and $\phi_c(\boldsymbol{x}_c)$'s and we add marginalization constraints based on a set of clusters, $\mathcal{S}$, then we have a new polytope defined by the following constraints,

$$\begin{aligned} &\sum_{\boldsymbol{x}_{c\setminus i}} \phi_c(\boldsymbol{x}_c) = \phi_i(x_i) \\ &\sum_{\boldsymbol{x}_{s\setminus c}} \tau_s(\boldsymbol{x}_s) = \phi_c(\boldsymbol{x}_c) \quad \forall s \in \mathcal{S} \\ &\sum_{\boldsymbol{x}_s} \tau_s(\boldsymbol{x}_s) = 1 \end{aligned} \tag{11}$$

where the summation over $\boldsymbol{x}_{s\setminus c}$ is similar to the summation over $\boldsymbol{x}_{c\setminus i}$ in (3b). Note, the $\tau_s(\boldsymbol{x}_s)$'s are auxiliary variables and do not affect the dimension of the space in which the tighter polytope resides. The polytope is in the space of the $\phi_i(x_i)$'s and $\phi_{ij}(x_i, x_j)$'s as before and is a tighter polytope compared to the local polytope. However, we cannot practically add a lot of cluster based marginalization constraints, even if they are small ones like triplets. For many problems it is sufficient to add a smaller set of clusters, when the clusters are chosen intelligently. We could intuitively imagine that it is enough if the polytope is tightened preferentially in the neighbourhood of the optimum, instead of being tightened uniformly all-around. We need a way to judge how well a cluster tightens the relaxation in the neighbourhood of the optimum. Werner (2008) presented an approach based on the language of *constraint satisfaction problems* (CSPs), by showing that all clusters that do not satisfy a CSP subproblem could potentially tighten the relaxation. However, there can be many such clusters and we still need a way to rank them, so that a smaller subset of clusters could be added. Sontag et al. (2008) shows a greedy approach to rank the clusters and shows an approach to incrementally add clusters as the algorithm proceeds. At this point of time, we would like to point out that the problem of MAP inference in MRFs is intimately related to CSPs Werner (2010), Thapper & Živný (2016).

In order to develop the algorithm, it is possible to work in the primal space like Sontag & Jaakkola (2007). Working with the primal, introduces cutting-planes but no new decision variables. This is in contrast to working with the dual, which could lead to the introduction of new decision variables. However, there are advantages to working in the dual space. Working with the dual, helps in designing a principled criterion to choose the clusters. This is based on the fact that the dual, like the one shown in (9), provided a lower bound to the MAP energy in (2) Wainwright et al. (2005). Thus, it is possible to rank the clusters based on how they may increase this lower bound. A similar criterion to rank the clusters in the primal is hard to achieve. In both the primal and the dual, we have a new optimization problem each time we add a new set of cluster based marginalization conditions and in the case of the daul, we have new dual variables corresponding to the added clusters. Interestingly, the dual formulation is such that we can get a feasible starting point by setting the old dual variables to their final value in the previous problem and by setting the new dual variables to zero. Thus, the main advantages of working in the dual are summarized as follows:

- Principled condition to choose clusters.
- Warm-starting based on values from previous iteration.
- Scalable approach that readily works with multi-label and higher-order graphs.

Sontag et al. (2008) adds clusters from a predefined set of small clusters, i.e., it is based on dictionary enumeration. For ease of exposition, we will present the following equations considering a pairwise graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Then, we will mention how to extend these concepts to graphs with higher-order cliques. In case, the graph contains small *chordless* clusters like $\{(i,j,k) | (i,j), (j,k), (k,i) \in \mathcal{E}\}$, we will have a dictionary of such triplet clusters. In case of a grid graph, the dictionary will be made of the square faces of the graph. In order to rank the clusters exactly, one needs to define a dual optimization problem for each cluster, i.e., an optimization problem that will additionally have marginalization constraints based on that cluster. Such a dual optimization problem has to be solved to optimality to assess the benefit of adding that cluster. Higher the optimal objective, higher is the rank for the cluster. Doing this for all clusters in the dictionary is very expensive. Instead, one could exploit a property shared by various dual based algorithms. This property is that the objective of a dual based algorithm increases monotonically. Thus, we could run for only one iteration a dual optimization problem to which marginalization constraints w.r.t. one cluster have been added. The increase in the dual objective after one iteration is guaranteed to be a lower bound on the increase that is expected if the dual optimization problem is solved to optimality. Thus, the clusters are ranked based on this lower bound. In order to obtain a valid lower bound, we need a dual algorithm for which the objective increases monotonically. This rules out an algorithm based on supergradient ascent to optimize (9), which is not a strictly ascending algorithm. On the other hand, coordinate ascent based approaches are strictly ascending Meshi et al. (2014). Sontag et al. (2008) is based on an algorithm called MPLP Globerson & Jaakkola (2008). Note, the dual for MPLP is of a different form compared to (9) and its derivation involves the introduction of auxiliary variables, which denote duplicate copies of the edge based pseudomarginals, $\phi_{ij}(x_i, x_j)$.

We will denote the set of dual variables as $\boldsymbol{\lambda}$ and there are dual variables corresponding to each edge-node pair $(\lambda_{ij \to i}(x_i))$, each edge $(\lambda_{ij}(x_i, x_j))$ and each cluster-edge pair $(\lambda_{s \to ij}(x_i, x_j))$. Every iteration of the algorithm in Sontag et al. (2008) involves closed form updates of all these dual variables, similar to MPLP. We note that the size of $\boldsymbol{\lambda}$ keeps increasing as we add cluster based constraints. Suppose, at the end of iteration t, the dual variables are $\boldsymbol{\lambda}^t$. In iteration $t+1$, we would like to compute the criterion for some cluster s. In order to run one iteration of the dual problem that involves this additional cluster, we will warm start the dual variables corresponding to the already present clusters using $\boldsymbol{\lambda}^t$ and we will initialize the dual variables corresponding to cluster s to zero. Since, we are warm starting using $\boldsymbol{\lambda}^t$, the dual increases in value only based on updates from the cluster s. This quantity has a closed form expression, which is expressed in terms of edge based *beliefs*. For every edge $(i,j) \in s$, we could define a belief as,

$$b_{ij}(x_i, x_j) = \lambda_{ij}(x_i, x_j) + \sum_{s' : e \in s'} \lambda_{s' \to ij}(x_i, x_j). \tag{12}$$

The belief, $b_{ij}(x_i, x_j)$, is directly proportional to the marginal probability of edge ij taking label (x_i, x_j). The summation in (12) is not affected by the dual variables $\lambda_{s \to ij}(x_i, x_j)$ because they are set to zero. Thus, the increase in the dual objective due to the new cluster s takes the following form,

$$d(s) = \sum_{ij \in s} \max_{x_i, x_j} b_{ij}(x_i, x_j) - \max_{\boldsymbol{x}_s} \left[\sum_{ij \in s} b_{ij}(x_i, x_j) \right]. \tag{13}$$

$d(s)$ is a lower bound on the improvement to the dual objective if cluster s is added. Thus, the clusters are ranked with respect to $d(s)$ and are added in a greedy fashion. The overall approach based on these ideas is shown in algorithm (1).

Algorithm 1 Dual algorithm with iterative tightening

```
1: Run a dual algorithm like MPLP over the local polytope as close to the optimum as
   possible (say, 1000 iterations or convergence, whichever comes first).
2: Recover an integral solution and compute primal-dual gap. If the gap is less than a
   small threshold, then exit.
3: Add a fixed number of clusters according to the bound criterion (13).
4: Warm start the dual variables if possible and run the dual algorithm for a fixed limited
   number of iterations.
5: Return to step 2.
```

In algorithm (1) the integral solution could be recovered in several ways Ravikumar et al. (2010), Meshi et al. (2014), Savchynskyy & Schmidt (2014). Many times, a simple and efficient rounding scheme is preferable, like independently picking for each of the nodes the label with maximal pseudomarginal value. In step 4, it is sufficient to run the dual algorithm for a fixed number of iterations such that we have reached a neighbourhood of more optimal decision variables. In case, we run the dual algorithm to optimality in step 4, adding clusters is exactly equivalent to adding cutting-planes in the primal problem.

In order to extend these ideas to higher-order cliques, we first observe that we have seen three groups of nodes in our equations so far: clusters, edges and individual nodes. Also, in practice the clusters are built as unions of edges. The equations involved variables that associate a larger group of nodes with a smaller subset of that larger group. We could think about this in terms of tuples of the following form: $(A, B, \boldsymbol{x}_B)$. Here, A is a larger subset of nodes like a cluster or an edge. B is also a subset of nodes such that $B \subset A$. B could be an edge (resp. node), if A is a cluster (resp. cluster or edge). $\boldsymbol{x}_B$ is one possible labelling of the nodes in B. Such a tuple is called a *pencil* in graphical models literature Werner (2008). The dual variable $\lambda_{s\to ij}(x_i, x_j)$ which appeared a little earlier, is with respect to the pencil $(s, ij, (x_i, x_j))$. Thus, to use Sontag et al. (2008) for higher-order cliques, we will replace the edges by higher-order cliques. We recollect that a clique with two nodes is also called an edge and a clique with more than two nodes is called a higher-order clique or hyperedge. Thus, we will work with dual variables of the form $\lambda_{s\to c}(\boldsymbol{x}_c)$, which corresponds to the pencil $(s, c, \boldsymbol{x}_c)$. As before, s is a cluster which defines additional marginalization constraints and now, it is a union of higher-order cliques. c is one such higher-order clique and $\boldsymbol{x}_c$ is one possible labelling of c.

Another point to keep in mind is that the marginalization constraints with respect to a set of clusters $\mathcal{S}$, could be equivalently represented in another way. This equivalent representation involves a new graphical model Werner (2008). In this graphical model, the clusters $s \in \mathcal{S}$ are represented as cliques, apart from the original set of cliques $\mathcal{C}$. However, the clique potentials, $\theta_s(\boldsymbol{x}_s)$'s, of the clusters will all be set to zero. Thus, when we consider the LP relaxation equations (similar to (5)) for this graphical model, the objective will have the same value as the original graphical model because the additional cluster based potentials are zero valued. When we look at the constraints, there will be additional marginalization constraints present with respect to the new clusters, $s \in \mathcal{S}$, involving new pseudomarginals $\phi_s(\boldsymbol{x}_s)$'s. We can see that these pseudomarginals are exactly like the auxiliary variables $\tau_s(\boldsymbol{x}_s)$'s that we saw in (11). Thus, these new pseudomarginals have the same effect on the original pseudomarginals, $\phi_i(x_i)$'s and $\phi_c(\boldsymbol{x}_c)$'s. They get constrained to be in a tighter polytope. The advantage of imagining such a graphical model with additional zero valued potentials is to write the LP relaxation equations in a symmetric manner using pencils. For this new graphical model, the primal form of the LP relaxation will take the following form,

$$\underset{\boldsymbol{\phi}}{\text{minimize}} \quad \sum_{i\in\mathcal{V}}\sum_{x_i}\theta_i(x_i)\phi_i(x_i) + \sum_{c\in\mathcal{C}}\sum_{\boldsymbol{x}_c}\theta_c(\boldsymbol{x}_c)\phi_c(\boldsymbol{x}_c) + \sum_{s\in\mathcal{S}}\sum_{\boldsymbol{x}_s}\theta_s(\boldsymbol{x}_s)\phi_s(\boldsymbol{x}_s) \tag{14a}$$

$$\text{subject to} \quad \sum_{\boldsymbol{x}_{A\setminus B}}\phi_A(\boldsymbol{x}_A) = \phi_B(\boldsymbol{x}_B), \quad \forall (A,B,\boldsymbol{x}_B)\in J \tag{14b}$$

$$\begin{aligned} &\sum_{x_i}\phi_i(x_i) = 1, \quad \phi_i(x_i)\geq 0 \quad \forall i\in\mathcal{V} \\ &\sum_{\boldsymbol{x}_c}\phi_c(\boldsymbol{x}_c) = 1, \quad \phi_c(\boldsymbol{x}_c)\geq 0 \quad \forall c\in\mathcal{C} \\ &\sum_{\boldsymbol{x}_s}\phi_s(\boldsymbol{x}_s) = 1, \quad \phi_s(\boldsymbol{x}_s)\geq 0 \quad \forall s\in\mathcal{S}. \end{aligned} \tag{14c}$$

Here, the set J is made of pencils $(A, B, \boldsymbol{x}_B)$ of the following form: if A is a clique of the original graphical model, then B corresponds to the nodes of that clique; if A is a cluster added to tighten the polytope, then B corresponds to the cliques whose union is the cluster A. For example, we can see that for a pairwise graph, a cluster s could be a triplet and marginalization constraints are enforced between this triplet and the edges that make up the triplet.

2.1 More efficient cluster pursuit algorithms

Among the literature on tighter relaxations, a few of them have designed algorithms based on the dual of the primal problem given in (14) Werner (2008), Batra et al. (2011). That dual takes the following form,

$$\underset{\substack{h_A \;\; \forall A\in\mathcal{V}\cup\mathcal{C}\cup\mathcal{S} \\ y_{A\to B}(\boldsymbol{x}_B) \;\; \forall(A,B,\boldsymbol{x}_B)\in J}}{\text{maximize}} \quad \sum_{i\in\mathcal{V}} h_i + \sum_{c\in\mathcal{C}} h_c + \sum_{s\in\mathcal{S}} h_s \tag{15a}$$

$$\begin{aligned} \text{subject to} \quad & h_A \leq \theta_A(\boldsymbol{x}_A) + \sum_{Z|(Z,A,\boldsymbol{x}_A)\in J} y_{Z\to A}(\boldsymbol{x}_A) - \sum_{B|(A,B,\boldsymbol{x}_B)\in J} y_{A\to B}(\boldsymbol{x}_B) \\ & \forall A\in\mathcal{V}\cup\mathcal{C}\cup\mathcal{S}, \quad \forall(A,B,\boldsymbol{x}_B)\in J. \end{aligned} \tag{15b}$$

We observe that the added clusters influence the dual objective through the variables h_s, while they do not affect the primal objective because of the zero valued potentials. Using this dual formulation, Batra et al. (2011) has addressed a computational bottleneck in Sontag et al. (2008). The computational complexity of computing $d(s)$ given in (13) is $O(l^{|s|})$. Thus, computing this score for all the clusters enumerated in the dictionary is computationally very heavy. Batra et al. (2011) presents a heuristic criterion which could be used to first select a smaller subset of the clusters from the dictionary and compute $d(s)$ only for those clusters. Before we look at this heuristic criterion, we first observe that $\widetilde{\theta}_A(\boldsymbol{x}_A) \triangleq \theta_A(\boldsymbol{x}_A) + \sum_{Z|(Z,A,\boldsymbol{x}_A)\in J} y_{Z\to A}(\boldsymbol{x}_A) - \sum_{B|(A,B,\boldsymbol{x}_B)\in J} y_{A\to B}(\boldsymbol{x}_B)$ is a *reparameterization* of $\theta_A(\boldsymbol{x}_A)$. This means that for a given labelling of all the nodes in the graphical model, we will get the same energy with either the $\theta_A(\boldsymbol{x}_A)$'s or the $\widetilde{\theta}_A(\boldsymbol{x}_A)$'s. Reparameterization is a concept that we will come across often in graphical models literature, for example, Kolmogorov (2006). For each clique $c \in \mathcal{C}$, we could define a quantity,

$$l(c) = \widetilde{\theta}_c(\boldsymbol{x}_c^p) - \min_{\boldsymbol{x}_c}\left[\widetilde{\theta}_c(\boldsymbol{x}_c)\right] \tag{16}$$

where $\boldsymbol{x}_c^p$ is an integral labelling of the clique c that is obtained in step 2 of algorithm (1). This heuristic is derived based on the form of the complementary slackness condition for the constraint given in (15b). Then, for a cluster s, we define a criterion based on the cliques that it is made of,

$$l(s) = \sum_{c:c \in s} l(c). \tag{17}$$

Compared to $d(s)$ in (13), this criterion could be computed with complexity $O(l^{|c|})$. Just like we did with $d(s)$, the clusters could be ranked based on the value of $l(s)$. We note that this criterion comes with poorer guarantees compared to $d(s)$ and it is best to use it as an initial preprocessing step. $l(s)$ is computed for all clusters in the dictionary, in order to select a small subset of clusters and this smaller set is ranked according to $d(s)$. We also point out that the condition $l(s) = 0$ for a cluster s is equivalent to satisfaction of the CSP subproblem in Werner (2008).

The computation of $d(s)$ (13) need not be the only computationally heavy part of an algorithm for tighter relaxation. Depending on the dual algorithm, updating the dual variables could be costly, too. For example, the MPLP based approach of Sontag et al. (2008) updates the cluster-to-clique dual variables $(\lambda_{s \to c}(\boldsymbol{x}_c))$ with complexity that is exponential in the size of the cluster s. Sontag et al. (2009) shows a way to accelerate the update of these dual variables within the same MPLP based algorithm. This is achieved by defining a polytope, which is looser than the one defined by (11) but still tighter than $\mathcal{L}(\mathcal{G})$. This polytope is looser because the marginalization constraint between the added clusters and the cliques is over a coarser partitioning of the label space of the cluster, as follows,

$$\begin{aligned} \sum_{\boldsymbol{x}_{c \setminus i}} \phi_c(\boldsymbol{x}_c) &= \phi_i(x_i) \\ \sum_{\boldsymbol{z}_s \setminus \{z_i^s\}: i \in c} \tau_s(\boldsymbol{z}_s) &= \sum_{x_i \in z_i^s : i \in c} \phi_c(\boldsymbol{x}_c) \quad \forall s \in \mathcal{S} \\ \sum_{\boldsymbol{x}_c} \phi_c(\boldsymbol{x}_c) &= 1 \end{aligned} \tag{18}$$

where $\{\boldsymbol{z}_s\}$ is the set of labelling of the cluster s over a coarser label space. For each node in the cluster s, the set of labels is coarsened to a smaller set $\{z_i^s\}$. Then, marginalization constraints are enforced between a cluster and the clique it contains through these coarser labels. This will reduce the complexity of updating the cluster-to-clique dual variables $(\lambda_{s \to c}(\boldsymbol{x}_c))$. A better choice of the coarser label set $\{z_i^s\}$ is important. We want to ensure that the coarse partitioning behaves as closely as possible to what $d(s)$ guarantees with respect to increase in the dual objective. Sontag et al. (2009) presents a greedy approach with a threshold parameter, which leads to a trade-off between guaranteed improvement of the dual and the number of coarse states.

These cluster based ideas could be applied to practical problems, like the ones described in Yanover et al. (2006). For example, the problems of protein side chain prediction and protein design, have a graph structure that allows adding triplet based clusters. For stereo and segmentation problems, the graph is usually grid structured and we could work with clusters consisting of the four node faces of the graphs. So far, we have been looking at how to tighten the relaxation by enforcing additional cluster based constraints. At the same time, it is known in the graphical models literature that tighter relaxations are possible by using subgraphs which have loops (Komodakis et al. 2011, §6.1) within a dual decomposition formulation like (9). Is this approach related to tighter formulations using cluster based marginalization constraints? In fact, they are exactly the same concept. When we discussed about cluster based marginalization constraints, we presented a pair of problems: primal (14) and dual (15). The clusters had zero potentials in the primal, in order to ensure that the primal energy continues to be a function of the original node and clique potentials. However, we could define the cluster potentials in the following way: $\sum_{\mathcal{S}\in\mathcal{S}(c)} \theta_c^{\mathcal{S}}(\boldsymbol{x}_c) = \theta_c(\boldsymbol{x}_c), \ \ \forall c \in \mathcal{S}$. Then, by considering only node and cluster potentials, we ensure that the primal energy stays the same. Incidentally, this form of the cluster potentials is also a required condition for the dual in (9). Thus, having cluster based marginalization constraints in the primal, is equivalent to having cluster based subproblems in the dual, with suitably defined potentials. We will understand this concept more by considering cycles. We will see how tighter relaxation could be achieved either by imposing cycle based marginalization constraints within an algorithm like Sontag et al. (2008) or by decomposing the graphical model using cycle based subgraphs.

3 Cycles in the Graph

So far, we have been looking at clusters, which in theory could be any subset of the nodes. In practice, we seek small clusters which are connected to each other. The subset of nodes could have more structure, for example, they could form a *cycle*. A cycle is a chain of node pairs with the following form: $\{(x_0, x_1), (x_1, x_2), ..., (x_{n-2}, x_{n-1}), (x_{n-1}, x_0)\}$. Cycles play an important role in understanding about tighter relaxations for MAP inference. Suppose a given LP relaxation results in fractional values as the solution. So far, we have understood this as the relaxed polytope being not tight enough around the optimal integral vertex of the marginal polytope. Such fractional solutions could be directly characterized by the presence of *frustrated cycles* in the graph. To understand what these are and what to do about them, we will first consider a pairwise graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for ease of exposition. Later, we will see how to extend these works to higher-order graphs. Also, we will assume that all nodes take one of l labels.

So far, we have been made aware of the possibility of various dual formulations, (9), (15) and the MPLP based dual of Sontag et al. (2008). All these duals lower bound the MAP energy given in (2). There is yet another dual formulation that throws light on the problem of frustrated cycles. It is given in Komodakis & Paragios (2008) and is of the following form,

$$\underset{\boldsymbol{y}}{\text{maximize}} \quad \sum_{i\in\mathcal{V}} \min_{x_i}. \left(\theta_i(x_i) + \sum_{j:ij\in\mathcal{E}} y_{ij}(x_i) \right) \tag{19a}$$

$$\text{subject to} \quad \bar{\theta}_{ij}(x_i, x_j) - y_{ij}(x_i) - y_{ji}(x_j) \geq 0 \quad \forall ij \in \mathcal{E} \tag{19b}$$

$$\sum_{ij\in\mathcal{E}_S} \bar{\theta}_{ij}(x_i, x_j) \leq \sum_{ij\in\mathcal{E}_S} \theta_{ij}(x_i, x_j). \tag{19c}$$

We will denote this dual problem as $\mathcal{D}(\mathcal{E}_S)$, where, $\mathcal{E}_S$ is a subset of edges of the graph and $\mathcal{E}$ is the set of all edges. We would like to highlight the presence of $\bar{\boldsymbol{\theta}}$ in the formulation, which is the set of all $\bar{\theta}_{ij}(x_i, x_j)$'s and we will call them *virtual* potentials. Suppose, $\mathcal{E}_S = \mathcal{E}$ then we obtain a dual problem, $\mathcal{D}(\mathcal{E})$, that is equivalent to the intractable ILP (3). If $\bar{\theta}_{ij}(x_i, x_j) = \theta_{ij}(x_i, x_j) \quad \forall ij \in \mathcal{E}_S, \ \forall x_i, x_j$, then the dual, which we will call as $\mathcal{D}_{fixed}$, is equivalent to the LP relaxation (5). It comes with the same guarantees as optimizing LP relaxation over the local polytope. The optimal solutions of $\mathcal{D}_{fixed}$ are charaterized by the following theorem.

Theorem 1 (Weak Optimality Condition Werner (2007))**.** *After $\mathcal{D}_{fixed}$ has been solved to optimality, there must be at least one optimal label x_i^* for node i and at least one optimal label x_j^* for node j, such that $\bar{\theta}_{ij}(x_i^*, x_j^*) = y_{ij}(x_i^*) + y_{ji}(x_j^*)$.*

The above theorem is a necessary but not sufficient condition for an optimal labelling of the original MAP inference problem (2). This is expected because $\mathcal{D}_{fixed}$ is a tractable approximation to $\mathcal{D}(\mathcal{E})$, which is an NP-hard problem. Komodakis & Paragios (2008) showed how solving $\mathcal{D}(\mathcal{E}_S)$ presents a trade-off between these two extremes of tractability and accuracy. In order to see how it could be done, we need to understand the following stronger constraint that an optimal labelling for MAP inference is required to satisfy.

Theorem 2 (Strong Optimality Condition)**.** *An optimal labelling of the graph that minimizes the energy (1) is such that it assigns the labels (x_i^*, x_j^*) to the edge (i, j) then $\bar{\theta}_{ij}(x_i^*, x_j^*) = y_{ij}(x_i^*) + y_{ji}(x_j^*)$, for all edges.*

Solving $\mathcal{D}(\mathcal{E})$ ensures theorem 2. Suppose, $\mathcal{D}(\mathcal{E}_S)$ has been solved by some optimization algorithm and we obtain optimal labels for all nodes. We should keep in mind that there could be multiple optimal labels for some nodes. Suppose, we start at some node i on the graph and would like to take a step to a neighbouring node j. We could do it if $\exists\ x_i^*$ and x_j^* such that $\bar{\theta}_{ij}(x_i^*, x_j^*) = y_{ij}(x_i^*) + y_{ji}(x_j^*)$. Suppose we traverse a cycle of nodes in this manner and for each node i, the step towards and away from it are based on the same optimal label x_i^*, then that cycle is not a frustrated cycle. If it is possible to traverse all possible cycles of the graph based on this criterion, then the graph does not have any frustrated cycles. An optimal solution without frustrated cycles and an optimal solution satisfying the condition in theorem 2 are equivalent. In such a situation, $\mathcal{D}(\mathcal{E}_S)$ has managed to find the optimal MAP labelling. Suppose, a cycle in the graph could not be traversed according to the above mentioned criterion, then it is a frustrated cycle. Frustrated cycles could be of two different forms. One, where for all pairs of optimal labels for two adjacent nodes, $\bar{\theta}_{ij}(x_i^*, x_j^*) \neq y_{ij}(x_i^*) + y_{ji}(x_j^*)$. Hence, it is not possible to take a step between these two nodes. Two, where a node i is not able to satisfy $\bar{\theta}_{ij}(x_i^*, x_j^*) = y_{ij}(x_i^*) + y_{ji}(x_j^*)$ with all its neighbours based on the same optimal label x_i^*. An optimal labelling with frustrated cycles satisfy theorem 1 only. In practice, if the clique potentials are attractive, then frustrated cycles are discouraged and the opposite is true for non-attractive potentials. In section (1.1) we said that if the clique potentials are submodular, then the local polytope is tight. Thus, there will be no frustrated cycles in a graphical model with submodular potentials. For general clique potentials, tightening the LP relaxation leads to the elimination of such frustrated cycles. Thus, cycles play an important role in understanding and addressing the tightening of LP relaxations.

We had earlier mentioned that (19) involves the virtual potentials, $\bar{\boldsymbol{\theta}}$. If we can set $\bar{\boldsymbol{\theta}}$ properly then the optimal labelling obtained by solving (19) could have less or no frustrated cycles. Komodakis & Paragios (2008) has shown a way to iteratively modify $\bar{\boldsymbol{\theta}}$, followed by solving the dual problem (19), in order to obtain an optimal labelling with less or no frustrated cycles. They proposed to solve a series of dual problems, $\mathcal{D}(\mathcal{E}_S)$ with different $\mathcal{E}_S$. They solve these dual problems one after the other and repeatedly, until a stopping criterion is reached. Each $\mathcal{E}_S$ is picked from a dictionary of cycles. For example, in a grid graph, the dictionary could be composed of cycles that are the faces of the grid and also, the cycles going around two adjacent faces. In theory, the set of edges $\mathcal{E}_S$ in the optimization problem (19) could be any set of edges, not necessarily cycles. Since, it is frustrated cycles that we are trying to eliminate, it intuitively and algorithmically makes sense to choose $\mathcal{E}_S$ to be cycles. Thus the algorithm in Komodakis & Paragios (2008) repeatedly iterates over a dictionary of cycles.

At each iteration, we have a given set of virtual potentials, $\bar{\boldsymbol{\theta}}$ and a given cycle, $\mathcal{E}_S$. Based on the optimal dual variables $\boldsymbol{y}$ from the previous iteration, we check whether the current cycle is a frustrated cycle. Komodakis & Paragios (2008) shows a principled way to "repair" such a frustrated cycle and this procedure is the heart of the paper. The approach is to modify the values of $\bar{\boldsymbol{\theta}}$ corresponding to the cycle $\mathcal{E}_S$. While making this modification, the contraints (19b) and (19c) should be maintained. We do this by picking a pair of nodes in $\mathcal{E}_S$, for which there is at least one pair of labels for which the constraint (19b) is an equality. This will always be true if the previous dual problem (19) has been solved to optimality. According to Komodakis & Paragios (2008), we modify $\bar{\boldsymbol{\theta}}$ such that for that pair of nodes, the constraint (19b) becomes a strict inequality for all pairs of labels between the two nodes. This modification has to be done while continuing to satisfy the contraint (19c). This is possible by increasing the elements in $\bar{\boldsymbol{\theta}}$ w.r.t. to the pair of nodes, while decreasing elements in $\bar{\boldsymbol{\theta}}$ w.r.t. adjacent pairs of nodes in the cycle. In a frustrated cycle, it is always possible to perform this operation, while satisfying the constraints given in (19b) and (19c). Once the cycle has been "repaired" this way, the dual problem is now solved, with the constraint (19c) enforced w.r.t. this cycle. Based on the new optimal dual variables $\boldsymbol{y}$, this cycle will not be a frustrated cycle. These steps, "repair" cycle and solve the dual, are repeated for the next cycle in the dictionary and so on.

In this process, a cycle that was repaired earlier could again become a frustrated cycle. That is why the algorithm iterates over the cycles in the dictionary repeatedly. It is possible to warm-start $\boldsymbol{y}$ for the current dual optimization problem based on the previous optimal $\boldsymbol{y}$ because the virtual potentials are the same except for the two adjacent cycles in the dictionary. This leads to considerable speed-up. Interestingly, the dual objective increases monotonically as we solve one dual problem after the other. Thus, the sequence of optimal objective values are going closer to that of $\mathcal{D}(\mathcal{E})$, the intractable MAP inference problem, where contraint (19c) has to be satisfied over the entire set of edges, $\mathcal{E}$. Based on this monotonic increase property, this algorithm could be terminated in a principled manner. For example, a good way to terminate is when there are no frustrated cycles with respect to the dictionary. Even though larger frustrated cycles could still exist, this approach is sufficient to return the MAP labelling in several practical problems. In section (2), we have looked at tightening the relaxation using cluster based marginalization constraints. Optimizing $\mathcal{D}(\mathcal{E}_S)$, where $\mathcal{E}_S$ is a cycle, is similar to enforcing marginalization constraints with respect to that cycle. Earlier, we worked with additional dual variables $(\lambda_{s\to c}(\boldsymbol{x}_c))$ with respect to that cluster s. In this case, we work with $(\bar{\theta}_{ij}(x_i, x_j))$ corresponding to the edges ij of the cycle $\mathcal{E}_S$, which are adjusted if the cycle is frustrated.

3.1 Searching for frustrated cycles

The main bottleneck with the approach given in Komodakis & Paragios (2008) is w.r.t. efficiency. Their approach iterates over a big dictionary of short cycles present in the graph. Also, for some graphs like sparse graphs with few short cycles, it is not clear how to enumerate the dictionary of cycles. Thus, it is desirable to choose a cycle directly from the graph, instead of choosing cycles from a predefined dictionary. Sontag et al. (2012) has presented an approach to directly assess how a cycle will improve the dual bound. Their work combines ideas presented in Sontag & Jaakkola (2007), (Johnson 2008, §3.4) and Sontag et al. (2008). Just like Sontag et al. (2008) we greedily choose a cycle that leads to best dual objective improvement based on running one iteration of a dual optimization problem. This dual problem has additional dual variables corresponding to the cycle, just like we saw in section (2). Therefore, we have a function that serves a purpose similar to

$d(s)$ in (13) for choosing a cluster. In this case, we don't choose the cycle from a dictionary of cycles. At the same time, it is possible to characterize more desirable cycles. In order to do that we first construct a graph with edge weights that are defined based on edge beliefs of the original graph. We have already come across the concept of edge beliefs in (12) and they are obtained through an optimization algorithm like the dual based algorithm in Sontag et al. (2008). In this new graph, we seek the cycle, which among the cycles with an odd number of negative edge weights, maximizes the minimum absolute value of the edge weights present. This will lead us to cycles in the original graph that are more suited for tightening the relaxation. (Johnson 2008, §3.4) showed why this is true and also, showed an algorithm to obtain desirable cycles efficiently for graphs with binary valued nodes. For binary variables, the new graph that needs to be constructed has the same topology as the original graph. In order to obtain an approach that works for multi-label graphs, Sontag et al. (2008) observed that cycle inequalities (6, **??**) also, impose constraints on an odd number of edges within a cycle. Thus, for the multi-label case, a new projection graph is constructed, which was earlier used by Sontag & Jaakkola (2007) to define cycle inequalities for graphs with multi-label nodes. Sontag et al. (2012) then applies the algorithm of (Johnson 2008, §3.4) to this projection graph.

Once a cycle has been identified, it has to be used within an algorithm like that of Sontag et al. (2008). The algorithm in Sontag et al. (2008) is based on MPLP and the per iteration complexity will be high if the cycle is long. It is possible to bias the search algorithm to return cycles with shorter length. The cycle search algorithm of (Johnson 2008, §3.4) involves the construction of a spanning tree of the newly constructed graph and choosing an initial edge based on which the cycle is constructed. By constructing a shallow spanning tree and by using an efficient least common ancestor algorithm, cycles could be found, which are of shorter length and of higher rank w.r.t. the greedy criterion. In the next section, we will see an algorithm that further improves the efficiency when cycles are present within inference algorithms like MPLP.

Empirical results show the clear benefit of this approach for graphs with few or no short cycles. There are problem instances where the frustrated cycle is a short one, i.e., it is equivalent to a small cluster. In such cases, the cluster pursuit algorithm based on $d(s)$ is able to identify this cycle very quickly compared to the above cycle based greedy criterion. Thus, whenever possible, it is a good idea to add both cycles (found using the approach outlined above) and clusters (found using $d(s)$ in (13)) to tighten the relaxation. It should be noted that if $d(s)$ is used for finding cycles, then we need to work with a dictionary of cycles. For some graph topologies it could be difficult to construct this dictionary. Also, the approach of using $d(s)$ to a dictionary of cycles is accompanied by the following hardness results:

- Suppose, the graph has binary variables and the beliefs are obtained by running the optimization routine to optimum, then finding the cycle that maximizes $d(s)$ is equivalent to finding the most violated cycle inequality in the dual.
- Suppose, the graph has binary variables and the optimization is not run to optimum, then finding the cycle that maximizes $d(s)$ is NP-hard.
- Suppose, the graph is multi-label, then it is always NP-hard to rank cycles based on $d(s)$.

We will briefly discuss about cycles and higher-order MRFs. Many higher-order MRFs, involve both higher-order cliques and pairwise edges (for e.g., the datasets in Rother et al. (2009)). In such graphs, we could consider the cycles involving the pairwise edges. In case, only higher-order cliques are present, pairwise edges with zero valued potentials could be considered and suitable cycles could be defined. This is exactly like considering zero valued cluster based potentials in (14). We have seen that a cluster is any subset of nodes and a cycle is a special type of cluster. Thus, we could have algorithms that impose cycle based marginalization constraints, just like cluster based marginalization constraints that we came across in section (2). Towards the end of that section, we mentioned the equivalence between algorithms involving cluster based marginalization constraints and algorithms involving cluster based subproblems within dual decomposition. The same holds for cycles, too. It is very well possible to achieve tighter relaxation by considering cycles as subproblems within dual decomposition. The important works in this direction have been Yarkony et al. (2011), Wang & Koller (2013) and we will discuss them next.

3.2 Efficient MAP inference in a cycle

Suppose, one or more cycles are introduced as subgraphs in dual decomposition (9), then efficient minimization of the cycle subgraphs is a challenge. If the subgraph is a pairwise tree, with m nodes, the minimization could be performed with $O(ml^2)$ complexity using dynamic programming. This is equivalent to one pass of an algorithm called *max-product message passing*. In a tree, a node is chosen as the root and message computation starts at leaf nodes and message passing proceeds towards the root. Between two nodes, message computation takes $O(l^2)$ complexity, thus resulting in $O(ml^2)$ overall complexity. On the other hand, a naive approach for the cycle, with m nodes, would require $O(ml^3)$ complexity. This is because the complexity of inference in graphical models is exponential in the treewidth of the graph and the treewidths of a tree and a cycle are 1 and 2, respectively. Generally, inference in a cycle, involves forming a clique tree by triangulating the cycle, followed by max-product message passing on the clique tree. This involves several passes of the max-product algorithm and is of complexity $O(ml^3)$. There have been algorithms that have reduced the complexity of inference in cycles below $O(ml^3)$ Felzenszwalb & McAuley (2011), McAuley & Caetano (2011). Wang & Koller (2013) builds on these works and presents a more efficient algorithm. One way to triangulate a cycle is to enforce a node to be common across all the triangles, i.e., all the triangulating chords are from that node to the other nodes. This results in a clique tree in the form of a chain. Wang & Koller (2013) uses the presence of the common node across all the cliques to design a more efficient algorithm. Generally, as number of passes of the max-product algorithm increases fewer labels from each node participate in the message computation. Finally, the labels narrow down to the MAP labels for each node. In the naive approach, the l^3 factor appears in the complexity because all labels from each node participate in the message computation. At the same time, identifying the active labels for a node that are enough for computing the messages is a challenging task. Wang & Koller (2013) have derived inequality based bounds to decide which labels of a node participate in message computation. They derive these inequalities by exploiting the common node and they are able to identify the necessary labels for each node that are participating in message computation. They start with a low value for the bound that allows more labels to be selected in the initial iterations and the bound increases as the algorithm progresses. The algorithm is guaranteed to terminate and the exit condition involves the inactive labels and potentials. Thus, using their MAP algorithm for cycles within dual decomposition (9) leads to a more efficient overall algorithm.

If the optimization algorithm is like the MPLP based algorithm of Sontag et al. (2008), a cycle is treated just like a cluster and the algorithm will enforce marginalization constraints with respect to a cycle. The update of the dual variables correspond to *min-marginal* computation for the corresponding nodes. Given a graphical model, the min-marginal of a node w.r.t. to a label, is the minimum energy (of the form (1)) for that graphical model when the label of that node is fixed. Thus, each node-label pair is associated with a min-marginal value. For example, the max-product algorithm computes the MAP labelling of a tree by computing min-marginals. However, depending on the graph, min-marginal computation is more involved than computing the MAP labelling. For example, for cycles, it is more efficient to compute the MAP labelling compared to min-marginals. Apart from the algorithm for MAP inference in cycles, Wang & Koller (2013) has shown a heuristic way to compute min-marginals in cycles and has shown good empirical performance with that approach.

3.3 Planar subproblems

So far we have been discussing about individual cycles, either we compute marginalization constraints with respect to each of the cycles or optimize each cycle as a subproblem within dual decomposition. Yarkony et al. (2011) considers the possibility of optimizing multiple cycles together as one subproblem within dual decomposition. How does such a subproblem look like? Suppose that the original problem is a planar graph (like many computer vision problems) and consider a subproblem with the same topology as the original planar graph. Within dual decomposition, optimizing such a planar subproblem is equivalent to optimizing multiple cycles at the same time. Recall that dual decomposition requires the subproblems to be optimized exactly. The natural question is how is this planar subproblem going to be optimized exactly in an efficient manner? We will have to consider specific pairwise potentials that will make this possible. For example, Schraudolph & Kamenetsky (2009) showed how exact inference could be performed efficiently for a planar Ising model with binary variables. In an Ising model, the values taken by the edge potentials depend only on whether the two nodes take same or different labels.

Yarkony et al. (2011) uses planar subproblems which are of the above type. However, their original graphs are planar graphs with nodes taking multiple labels. They show that a node in the planar subproblem could be assigned binary labels, where the binary labels correspond to a binary partition of the label set of that node. They show one well motivated way to obtain a binary partition of a label set. Other approaches are also possible. In their algorithm, initially, the non-smooth dual (9) is solved to optimality using a decomposition based on trees. Then, labels are obtained for each node by rounding the fractional relaxed variables. Then, they partition the label set of each node in a one-vs-rest manner based on this labelling. Thus a binary planar subproblem is created and added to the dual decomposition. Now, we have a dual problem, with tree based subproblems and also, a planar subproblem. Again, this dual is solved to optimality and based on the resulting labels another binary planar subproblem is created. They fix a maximum limit on the number of binary subproblems that are added in this manner (say, 10). Their intuition is that each time a dual is solved to optimality and labels are decoded, the resulting energy is an upper bound to the actual energy (just like how the dual is a lower bound). The binary planar subproblems are encouraging the nodes to take labels corresponding to such upper bounds. Within each iteration of dual decomposition, the binary subproblem is solved to optimality based on an efficient algorithm like Schraudolph & Kamenetsky (2009) and the optimal labelling leads to a valid supergradient with respect to that subproblem (Komodakis et al. 2011, Lemma 1). As we said before this optimal binary labelling corresponds to a one-vs-rest partitioning of the original label set. The binary labels that are assigned to the nodes have to be lifted back to the original label space, in order to have a valid supergradient. If the optimal binary label of a node within this binary planar subproblem corresponds to the single label within the binary partition, then we have no problem. However, if the optimal label corresponds to the rest of the labels partition, then one of these labels is chosen at random and assigned as the optimal label according to the original label space. This could be a problem, when the total number of labels increases. Yarkony et al. (2011) has shown results with problems involving upto 3 labels per node. Nevertheless, this work is an innovative approach that could spark some stronger approach in the future.

Similar to clusters, cycle based ideas could be applied to several datasets like stereo, segmentation etc Kappes et al. (2015). In the case of cluster based approaches, the algorithms could work with a dictionary of small-sized clusters, like triplets. For cycle based approaches, it is more involved to define the set of cycles to consider. For example, a small cycle made of three nodes could as well be treated as a cluster Sontag et al. (2012). There are certain sparse graphs like those that model a social network or the web graph, where cycle based approaches stand out Sontag et al. (2012).

4 Tighter subgraph decompositions

So far, we have come across a data structure called a cluster. In the broadest sense, any subgraph could be a cluster. Usually, in the context of tighter relaxation, a cluster is formed as the union of cliques. Such clusters are directly related to relaxation hierarchies like Sherali-Adams. A cycle is also a cluster, with more special structure, which could be exploited for more efficient algorithms. Also, working with subgraphs in the form of cycles throughs light on the concept of frustrated cycles, which one will come across in loose relaxations. There are other subgraphs with special structure, too, which will lead to tighter relaxations and more efficient algorithms.

For a pairwise graphical model, one of the most general subgraphs that could lead to a tighter relaxation is a hypertree (Wainwright & Jordan 2008, §4). For example, if the original graph is a pairwise grid graph, the local polytope relaxation decomposes the graph according to trees of the graph (for e.g., row chains and column chains). Instead, we could treat each face of the grid as a hyperedge/clique of 4 nodes and decompose the graph according to hypertrees, which are row chains and column chains of the faces of the grid. This decomposition will be tighter than the local polytope for a pairwise graph. From an algorithmic point of view, say, we are using a supergradient method to optimize (9), then the challenge is to perform the minimization efficiently within each of these hypertrees. We will recall that this minimization w.r.t. a hypertree is equivalent to MAP inference w.r.t. a hypertree of the same topology, with suitably defined clique potentials. Over the years there have been algorithms for efficient MAP inference in a hypertree having cliques with special structures, like cardinality and order Tarlow et al. (2010), linear constraint nodes Potetz & Lee (2008), matchings Duchi et al. (2007) and sparse, pattern based potentials Komodakis & Paragios (2009), Rother et al. (2009). Such specially structured clique potentials should be, and are often, leveraged to deal with higher-order structure in graphical models. Another interesting general approach for inference in trees is based on randomized algorithms Noorshams & Wainwright (2013).

Going beyond trees, Batra et al. (2010) decomposes the graph according to subgraphs that are outer-planar graphs. An outer-planar graph could be drawn as a planar graph such that no node is completely surrounded by edges. Another definition is that if a graph is formed from an outer-planar graph by adding an extra node and edges from that additional node to all other nodes, then that graph will be planar. Batra et al. (2010) point out that outer-planar graphs have treewidth of 2 and could be solved by the junction tree algorithm in general. Note that treewidth of 2 means $O(ml^3)$ complexity for the junction tree algorithm, where m is the number of nodes in the subgraph. This complexity could be prohibitive. For example, we saw similar complexity for inference in cycles (3.2) and discussed a more efficient approach. In the case of outer-planar graphs, Batra et al. (2010) use the efficient planar Ising model solver of Schraudolph & Kamenetsky (2009), whenever it is suitable. In this context, we would like to point out the work of Zheng et al. (2012), which also works with subgraphs of treewidth 2. They present an extended junction tree representation to solve such subgraphs.

Finally, one could decompose the graph according to subgraphs that are k-fans. In a graph with a k-fan structure, there are k inner nodes which are fully connected with each other. The rest of the nodes have edges to these k inner nodes only, not to each other. Suppose a k-fan subgraph has m nodes, the junction tree algorithm has complexity $O((m-k)l^{k+1})$. For k-fans, Bergtholdt et al. (2010), Kappes et al. (2010) work with a search based inference algorithm that performs faster on average, even though it has the same worst complexity. This k-fan based model is especially suited for problems with dense graphs like deformable parts based object localization.

4.1 Global higher-order cliques

Higher-order cliques come in different special forms. Usually, the special structure is exploited for efficiently optimizing the subproblems in dual decomposition or for efficient min-marginal computation. We came across some special clique potentials in the previous section. Apart from the nature of the clique potential, a higher-order clique is also characterized by its size. A special situation is a higher-order clique which involves all the nodes of the graph, i.e., it is a global higher-order clique. We will come across such cliques in applications like enforcing global cardinality constraint Li & Zemel (2014) for foreground/background segmentation, diverse M-best MAP estimation Batra et al. (2012), average-cut in image segmentation Mezuman & Weiss (2012). In such problems, apart from potentials corresponding to nodes and small cliques, we have a global clique, which we will denote as $\theta_\alpha(\boldsymbol{x})$. For such a problem, the LP relaxation takes the following form,

$$\underset{\boldsymbol{\phi}}{\text{minimize}} \quad \sum_i \sum_{x_i} \theta_i(x_i)\phi_i(x_i) + \sum_c \sum_{\boldsymbol{x}_c} \theta_c(\boldsymbol{x}_c)\phi_c(\boldsymbol{x}_c) + \sum_{\boldsymbol{x}} \theta_\alpha(\boldsymbol{x})\phi_\alpha(\boldsymbol{x}) \tag{20a}$$

$$\begin{aligned} \text{subject to} \quad & \sum_{\boldsymbol{x}_{c\setminus i}} \phi_c(\boldsymbol{x}_c) = \phi_i(x_i), \quad \forall (i,c) : i \in c \\ & \sum_{\boldsymbol{x}\setminus \boldsymbol{x}_c} \phi_\alpha(\boldsymbol{x}) = \phi_c(\boldsymbol{x}_c), \quad \forall c \in \mathcal{C} \end{aligned} \tag{20b}$$

$$\begin{aligned} & \sum_{x_i} \phi_i(x_i) = 1, \quad \phi_i(x_i) \geq 0 \quad \forall i \in \mathcal{V} \\ & \sum_{\boldsymbol{x}_c} \phi_c(\boldsymbol{x}_c) = 1, \quad \phi_c(\boldsymbol{x}_c) \geq 0 \quad \forall c \in \mathcal{C} \\ & \sum_{\boldsymbol{x}} \phi_\alpha(\boldsymbol{x}) = 1, \quad \phi_\alpha(\boldsymbol{x}) \geq 0. \end{aligned} \tag{20c}$$

The presence of the marginalization constraints allows us to write the same problem in the following simpler form,

$$\begin{aligned} &\underset{\boldsymbol{\phi}}{\text{minimize}} \quad \sum_{\boldsymbol{x}} \Big(\sum_i \theta_i(x_i) + \sum_c \theta_c(\boldsymbol{x}_c) + \theta_\alpha(\boldsymbol{x}) \Big) \phi_\alpha(\boldsymbol{x}) \\ &\text{subject to} \quad \sum_{\boldsymbol{x}} \phi_\alpha(\boldsymbol{x}) = 1, \quad \phi_\alpha(\boldsymbol{x}) \geq 0. \end{aligned} \tag{21}$$

Mezuman et al. (2013) show that this problem is tight. For their proof, they use the form given in (21). In this form, the objective is linear with respect to the $\phi_\alpha(\boldsymbol{x})$'s and the constraint is a simplex constraint over all $\phi_\alpha(\boldsymbol{x})$'s. Under such a simplex constraint, the optimal solution is to assign the value 1 entirely to that global labelling $\boldsymbol{x}$ that minimizes the objective in (21). Thus, we obtain the exact global labelling and the LP relaxation is tight.

As expected, solving (20) is intractable. This is because the marginalization constraint $\sum_{\boldsymbol{x}\setminus\boldsymbol{x}_c} \phi_\alpha(\boldsymbol{x}) = \phi_c(\boldsymbol{x}_c)$ is enforced with respect to all the smaller cliques, i.e., $\forall c \in \mathcal{C}$. Mezuman et al. (2013) try for a more tractable relaxation, which is still reasonably tight, by choosing a smaller set of the smaller cliques $\mathcal{S}$ and by enforcing the marginalization constraint with respect to $\phi_\alpha(\boldsymbol{x})$ in the following way,

$$\begin{aligned} &\sum_{\boldsymbol{x}\setminus x_i} \phi_\alpha(\boldsymbol{x}) = \phi_i(x_i), \quad \forall i \in \mathcal{V} - \mathcal{V}(\mathcal{S}) \\ &\sum_{\boldsymbol{x}\setminus \boldsymbol{x}_c} \phi_\alpha(\boldsymbol{x}) = \phi_c(\boldsymbol{x}_c), \quad \forall c \in \mathcal{S}. \end{aligned} \tag{22}$$

Mezuman et al. (2013) show that if $\mathcal{S}$ forms a low treewidth subgraph, the marginalization in (22) could be performed efficiently for special types of global potentials like cardinality based potential Tarlow et al. (2010) and sparse, pattern based potential Komodakis & Paragios (2009), Rother et al. (2009). They also show a way to iteratively build this set $\mathcal{S}$. Suppose, for a given $\mathcal{S}$, the algorithm has run to optimality and for some clique $c \notin \mathcal{S}$, the optimal labelling of those nodes w.r.t. the small clique θ_c and w.r.t. the global clique θ_α disagree, then adding c to $\mathcal{S}$ will lead to a tighter relaxation. Thus, the set $\mathcal{S}$ is grown step-by-step upto a maximum tractable size, where in each step the problem is run to optimality under marginalization constraints (22) defined by that $\mathcal{S}$.

By this stage, we have seen several algorithms that lead to a tighter LP relaxation. Some of these approaches are quite general, while some exploit special structure in the problem. We have also been exposed to fundamental concepts like the marginal and local polytopes, dual decomposition and frustrated cycles. It is time to broaden our understanding of continuous relaxations for MAP inference by discussing about semidefinite programming based relaxations.

5 Semidefinite programming based relaxation

Generally, a continuous relaxation for the MAP inference problem takes the form of an LP or a QP or an SDP or an SOCP. Among these relaxations, Kumar et al. (2009) showed that the LP relaxation formulation dominates a large class of QP and SOCP relaxations. Thus, there has been tremendous research focus on the LP relaxation and rightly so. As far as the SDP relaxation is concerned, the comparison with respect to the LP relaxation is more involved. We have seen that the LP relaxation could be tightened based on the

Sherali-Adams hierarchy. Similarly, the SDP relaxation could be tightened based on the *sum-of-squares* or the *Lasserre* hierarchy Lasserre (2001), Parrilo (2003). (Wainwright & Jordan 2008, §9.3) has pointed out that at a given level of their respective hierarchies, the LP and SDP relaxations are mutually incomparable, i.e., they are neither equivalent nor one strictly dominates the other. Thus, there will be problem instances where the SDP relaxation could indeed provide a tighter relaxation than the LP relaxation. However, SDP relaxations have been less attractive because solving an SDP is computationally intensive. An SDP involves a matrix as the decision variable. Therefore, the main challenge with solving SDPs is the heavy computational cost associated with linear algebraic operations (like, singular value decomposition (SVD)) involving that matrix decision variable.

In order to understand these computational challenges, we will first derive the general SDP relaxation for MAP inference. This could be derived from the QP relaxation for (2) . Suppose, we have a pairwise graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with n nodes and l labels per node. The QP relaxation takes the following form,

$$\underset{\boldsymbol{\phi}}{\text{minimize}} \quad \sum_{i \in \mathcal{V}} \sum_{x_i} \theta_i(x_i)\phi_i(x_i) + \sum_{i,j \in \mathcal{E}} \sum_{x_i, x_j} \theta_{ij}(x_i, x_j)\phi_i(x_i)\phi_j(x_j) \tag{23a}$$

$$\text{subject to} \quad \sum_{x_i} \phi_i(x_i) = 1, \quad \phi_i(x_i) \in [0,1] \quad \forall i \in \mathcal{V}. \tag{23b}$$

In the above formulation, if $\phi_i(x_i) \in \{0,1\}$ then we obtain an ILP that is equivalent to the one given in (3). In other words, the QP relaxation is obtained from an ILP formulation of (2), which involves only node based variables $\phi_i(x_i)$. On the other hand, the ILP in (3) involves clique based variables $\phi_c(\boldsymbol{x}_c)$, also. The reduction in the size of the decision variables comes with a price. The QP relaxation is non-convex. However, a remarkable property of the QP relaxation is that even though it is intractable, it is tight Ravikumar & Lafferty (2006). In other words, if the global optimum of the QP relaxation could be obtained, it will be the MAP labelling that we are seeking. However, direct attempts to solve this non-convex problem have shown poor empirical performance in terms of tighter results.

We could define a symmetric matrix variable $\Phi \in \mathcal{S}^{nl \times nl}$, such that $\Phi = \boldsymbol{\phi}\boldsymbol{\phi}^\top$. Here, ***phi*** is obtained by stacking the variables, $\phi_i(x_i)$, across all nodes i and labels x_i. In other words, an element of Φ is of the form, $\Phi(i; x_i, j; x_j) = \phi_i(x_i)\phi_j(x_j)$. We note that Φ has elements for all pairs of nodes, not just the edges of the graph. Thus, the problem in (23) could be rewritten as follows,

$$\underset{\boldsymbol{\phi},\, \Phi}{\text{minimize}} \quad \sum_{i} \sum_{x_i} \theta_i(x_i)\phi_i(x_i) + \sum_{i,j} \sum_{x_i, x_j} \theta_{ij}(x_i, x_j)\Phi(i; x_i, j; x_j) \tag{24a}$$

$$\text{subject to} \quad \sum_{x_i} \phi_i(x_i) = 1, \quad \phi_i(x_i) \in [0,1] \quad \forall i \in \mathcal{V} \tag{24b}$$

$$\Phi = \boldsymbol{\phi}\boldsymbol{\phi}^\top. \tag{24c}$$

It is still the same intractable but tight relaxation of the MAP inference problem, as (23). There is yet another way to represent this intractable problem, through a positive semidefinite constraint combined with a rank-1 constraint. It takes the following form,

$$\underset{\boldsymbol{\phi},\, \Phi}{\text{minimize}} \quad \sum_{i} \sum_{x_i} \theta_i(x_i)\phi_i(x_i) + \sum_{i,j} \sum_{x_i, x_j} \theta_{ij}(x_i, x_j)\Phi(i; x_i, j; x_j) \tag{25a}$$

$$\text{subject to} \quad \sum_{x_i} \phi_i(x_i) = 1, \quad \phi_i(x_i) \in [0,1] \quad \forall i \in \mathcal{V} \tag{25b}$$

$$\begin{bmatrix} 1 & \boldsymbol{\phi}^\top \\ \boldsymbol{\phi} & \Phi \end{bmatrix} \succeq 0 \tag{25c}$$

$$\operatorname{rank}\left(\begin{bmatrix} 1 & \boldsymbol{\phi}^\top \\ \boldsymbol{\phi} & \Phi \end{bmatrix}\right) = 1 \tag{25d}$$

$$\Phi(i; x_i, i; x_i) = \phi_i(x_i) \quad \forall i \in \mathcal{V}. \tag{25e}$$

We repeat that the problems (23), (24) and (25) are equivalent. We will look further into the motivation behind the formulation of the constraints in (25). First, we will see how the constraint $\Phi = \boldsymbol{\phi}\boldsymbol{\phi}^\top$ in (24) is equivalently represented by (25c) and (25d). In SDP relaxation literature, the presence of the equality constraint $\Phi = \boldsymbol{\phi}\boldsymbol{\phi}^\top$ (24c) in the stricter problem, usually leads to the semidefinite constraint $\Phi - \boldsymbol{\phi}\boldsymbol{\phi}^\top \succeq \mathbf{0}$ in the relaxation Goemans & Williamson (1995). Now, the following two inequalities are equivalent,

$$\Phi - \boldsymbol{\phi}\boldsymbol{\phi}^\top \succeq \mathbf{0} \tag{26}$$

$$\boldsymbol{M} \triangleq \begin{bmatrix} 1 & \boldsymbol{\phi}^\top \\ \boldsymbol{\phi} & \Phi \end{bmatrix} \succeq \mathbf{0}. \tag{27}$$

This equivalence is based on the Schur complement formula (Boyd & Vandenberghe 2004, §A.5.5). The semidefinite inequality (25c) along with the non-convex rank-1 constraint (25d) lead to an equivalent representation of $\Phi = \boldsymbol{\phi}\boldsymbol{\phi}^\top$. The SDP relaxation is obtained by dropping this rank-1 constraint. Further, the constraint $\Phi(i; x_i, i; x_i) = \phi_i(x_i)$ is inspired by the original ILP formulation of the problem. According to (24c), $\Phi(i; x_i, i; x_i) = \phi_i(x_i)^2$. If $\phi_i(x_i)$ takes integral values, i.e., $\{0, 1\}$, then $\Phi(i; x_i, i; x_i) = \phi_i(x_i)^2 = \phi_i(x_i)$. This constraint tries to compensate for the removal of the rank-1 constraint by encouraging the desired structure in the output variables. Thus, the SDP relaxation for MAP inference takes the following form,

$$\underset{\boldsymbol{\phi},\,\Phi}{\text{minimize}} \quad \sum_i \sum_{x_i} \theta_i(x_i)\phi_i(x_i) + \sum_{i,j} \sum_{x_i, x_j} \theta_{ij}(x_i, x_j)\Phi(i; x_i, j; x_j) \tag{28a}$$

$$\text{subject to} \quad \sum_{x_i} \phi_i(x_i) = 1, \quad \phi_i(x_i) \in [0, 1] \quad \forall i \in \mathcal{V} \tag{28b}$$

$$\begin{bmatrix} 1 & \boldsymbol{\phi}^\top \\ \boldsymbol{\phi} & \Phi \end{bmatrix} \succeq 0 \tag{28c}$$

$$\Phi(i; x_i, i; x_i) = \phi_i(x_i) \quad \forall i \in \mathcal{V}. \tag{28d}$$

In the SDP relaxation, the elements of Φ are equivalent to pseudomarginals of pairs of nodes, $\phi_{ij}(x_i, x_j)$, that we came across in the local polytope. The matrix $\boldsymbol{M}$ (27) contains pseudomarginals corresponding to nodes and pairs of nodes and it is called the *moment matrix* Wainwright & Jordan (2004*a*). $\boldsymbol{M}$ is a square matrix of side, $1 + nl$. Similar to the local polytope, $\mathcal{L}(\mathcal{G})$, the above constraints define a convex outer bound to the marginal polytope, $\mathcal{M}(\mathcal{G})$. We will denote it as $\mathcal{SDP}(\mathcal{G})$. Unlike, $\mathcal{L}(\mathcal{G})$, $\mathcal{SDP}(\mathcal{G})$ has a curved boundary. Note, in our formulation of the local polytope, we explicitly represented pseudomarginals only for pairs of nodes that correspond to the edges. However, it is possible to associate pseudomarginals with respect to all pairs of nodes and the SDP relaxation explicitly represents the pseudomarginals for all node pairs. The constraints (28c) and (28d) play a role similar to the marginalization constraints (5b) in the LP relaxation. Earlier we have seen that in the LP relaxation if the optimal variables are integer valued, we have obtained the exact MAP solution. Similarly, if the matrix $\boldsymbol{M}$ is of rank-1 in the SDP relaxation, we have obtained the exact MAP solution (Wang et al. 2016, Prop. 2).

Despite the computational challenge involving the semidefinite constraint (28c), there have been several works in recent years that have explored SDP relaxations for MAP inference. For several of these methods, the main thrust has been the efficient low-rank Burer-Monteiro approach Burer & Monteiro (2003). The Burer-Monteiro approach is an efficient way to solve semidefinite programs through a low-rank factorization based approximation of the matrix variable. Suppose, $\boldsymbol{M}$ is the matrix variable, it is substituted by $\boldsymbol{Y}\boldsymbol{Y}^\top$ everywhere in the problem formulation. We obtain an optimization problem with $\boldsymbol{Y}$, which is of low-rank, as the decision variable. One of the motivating reasons for this approach is that if an SDP satisfies certain mild conditions, the rank of the optimal decision variable is bounded by $\sqrt{2 \cdot \#constraints}$ Pataki (1998). Thus the optimal decision variable is guaranteed to be of considerably low-rank for many practical SDP formulations. Further, even though the Burer-Monteiro approach is non-convex, for several problems the local minimum returned by the approach is also a global minimum or is within a bounded value from the global minimum. Over the years, there has been progress in characterizing conditions which lead to these guarantees Burer & Monteiro (2005), Boumal et al. (2016), Mei et al. (2017), Boumal et al. (2018).

Given a hierarchy of LP relaxations and a hierarchy of SDP relaxations, (Wainwright & Jordan 2008, §9.3) has shown that at any given level, the corresponding LP relaxation and the SDP relaxation are mutually incomparable. As mentioned earlier, this is an important motivation to pursue SDP relaxations. $\mathcal{L}(\mathcal{G})$ and $\mathcal{SDP}(\mathcal{G})$ are the lowest levels of the LP relaxation and the SDP relaxation hierarchies, respectively. Since, these two feasible regions are incomparable, for some problems $\mathcal{L}(\mathcal{G})$ could be tighter and for some others $\mathcal{SDP}(\mathcal{G})$ could be tighter. Thus, the intersection of $\mathcal{L}(\mathcal{G})$ and $\mathcal{SDP}(\mathcal{G})$ is a better bound on $\mathcal{M}(\mathcal{G})$ than either of them separately. Peng et al. (2012) has pursued this idea and presented a dual based algorithm. The variables $\boldsymbol{\phi}$ and Φ are pseudomarginals or beliefs and we will denote them together as one large sized variable $\boldsymbol{b}$. We could plug $\boldsymbol{b}$ within the LP relaxation objective (5a), which we will denote as $F(\boldsymbol{b})$. Then, Peng et al. (2012) solves the following problem,

$$\begin{aligned} \underset{\boldsymbol{b}^1 \in \mathcal{L}(\mathcal{G}), \boldsymbol{b}^2}{\text{minimize}} \quad & F(\boldsymbol{b}^1) + \mathbb{1}\big[\boldsymbol{b}^2 \in \mathcal{SDP}(\mathcal{G})\big] \\ \text{subject to} \quad & \boldsymbol{b}^1 = \boldsymbol{b}^2. \end{aligned} \tag{29}$$

In other words, the objective function in (5a) is minimized within the intersection of $\mathcal{L}(\mathcal{G})$ and $\mathcal{SDP}(\mathcal{G})$. The Lagrangian dual of this problem nicely decomposes according to $\boldsymbol{b}^1$ and $\boldsymbol{b}^2$, as follows,

$$\underset{\boldsymbol{\delta}}{\text{maximize}} \quad \min_{\boldsymbol{b}^1 \in \mathcal{L}(\mathcal{G})} \left(F(\boldsymbol{b}^1) + \boldsymbol{\delta}^\top \boldsymbol{b}^1 \right) + \min_{\boldsymbol{b}^2 \in \mathcal{SDP}(\mathcal{G})} -\boldsymbol{\delta}^\top \boldsymbol{b}^2. \tag{30}$$

Thus, while maximizing the dual w.r.t. $\boldsymbol{\delta}$, the two subproblems could be minimized w.r.t. $\boldsymbol{b}^1$ and $\boldsymbol{b}^2$ separately. Since, the dual problem is non-smooth, the minimizing $\boldsymbol{b}^{1*}$ of subproblem 1 and the minimizing $\boldsymbol{b}^{2*}$ of subproblem 2, give us a valid supergradient $(\boldsymbol{b}^{1*} - \boldsymbol{b}^{2*})$ to update the dual variable $\boldsymbol{\delta}$ (Komodakis et al. 2011, Lemma 1). The first subproblem is equivalent to the LP relaxation (5) with an additional linear term in the objective. Peng et al. (2012) solves it using convex belief propagation Hazan & Shashua (2008).

The second subproblem is a semidefinite program, a linear objective with a semidefinite constraint. It involves the decision variable $\boldsymbol{b}^2$, composed of $\boldsymbol{\phi}$ and Φ, which can be written as a moment matrix $\boldsymbol{M}$, like we saw in (27). Taking inspiration from the Burer-Monteiro approach, Peng et al. (2012) replace $\boldsymbol{M}$ by the low-rank factorization $\boldsymbol{Y}\boldsymbol{Y}^\top$, where $\boldsymbol{Y} \in \mathcal{R}^{(1+nl)\times r}$. Here, r is the low valued rank of $\boldsymbol{Y}$ and the moment matrix representing $\boldsymbol{b}^2$. Expecting the moment matrix in the SDP relaxation to be low-rank makes intuitive sense because we came across a rank-1 constraint in the intractable formulation given in (25). Further, this low-rank representation ensures that the moment matrix that is implicitly computed as $\boldsymbol{Y}\boldsymbol{Y}^\top$ is positive semidefinite by definition. Thus, the semidefinite constraint $\boldsymbol{M} \succeq 0$ in $\mathcal{SDP}(\mathcal{G})$ need not be explicitly enforced. This simplification afforded by the Burer-Monteiro approach appears repeatedly in SDP literature. The explicitly enforced constraints are (28d). Note that the constraints (28b) are also enforced in $\mathcal{L}(\mathcal{G})$ and they need not be part of the second subproblem. Peng et al. (2012) takes an augmented Lagrangian based approach to solve this subproblem. Generally, with the low-rank representation, the question of setting the rank r needs to be addressed. In Peng et al. (2012), r was set to various values and all the results are reported. r was at most 50, in their experiments.

Let us keep in mind that the decision variables of the SDP relaxation could either be represented as a moment matrix, $\boldsymbol{M}$ or explicitly through the parts, $\boldsymbol{\phi}$ and Φ, which make up $\boldsymbol{M}$. The underlying problem is the same in both cases. The constraints (28b), (28c) and (28d) could be expressed in a suitable manner in both cases. In case the decision variables are represented by the moment matrix, the constraint $\boldsymbol{M}(0,0) = 1$ should be explicitly enforced, also.

Intersecting $\mathcal{L}(\mathcal{G})$ with $\mathcal{SDP}(\mathcal{G})$ is one instance of a general way to get better results with the SDP relaxation. The general approach is about adding suitable linear equality and/or inequality constraints to the semidefinite program (28). This idea has been explored since the early works studying the SDP relaxation Wainwright & Jordan (2004*a*). Huang et al. (2014) presents one other approach along these lines. It adds to the formulation in (28), the following non-negativity constraints on the matrix Φ,

$$\Phi(i; x_i, j; x_j) \geq 0 \quad \forall (i,j) \in \mathcal{E}. \tag{31}$$

Empirically, these inequality constraints improve the performance of SDP relaxations, when submodular edge potentials are present. We have mentioned earlier that submodular potentials are a type of attractive potentials. Also, Huang et al. (2014) has shown that the semidefinite constraint (28c) along with the linear constraints (28d) and (28b), implicitly ensure the satisfaction of the marginalization constraints (3b) that we come across in the LP relaxation. For their formulation, Huang et al. (2014) has shown an ADMM based approach Boyd et al. (2011). The challenge with ADMM based algorithms is the task of proving convergence. Wen et al. (2010) has presented an ADMM based algorithm for solving semidefinite programs, where convergence is guaranteed only if equality constraints are present. Interestingly, the equality constraints and the inequality constraints in the SDP relaxation proposed by Huang et al. (2014) are decoupled. Specifically, the equality constraints concern the vector $\boldsymbol{\phi}$ and the diagonal blocks of Φ, while the inequality constraints concern the edges of the graph, i.e., off-diagonal blocks of Φ. Hence, Huang et al. (2014) is able to prove convergence by using the same ADMM approach as Wen et al. (2010).

A big challenge with SDPs is the memory intensive representation of the optimization variables. One needs to look for structure in the problem at hand, to reduce the number of variables. For example, Huang et al. (2014) has to deal with the primal decision variables, $\boldsymbol{\phi}$ and Φ and the dual variables corresponding to the augmented Lagrangian. These dual variables correspond to the positive semidefinite constraint, the linear equality constraints and the linear inequality constraints. Interestingly, for their problem formulation, the matrix valued dual variable corresponding to the positive semidefinite constraint need not be explicitly represented. Since, it could be expressed in terms of the other dual variables and the primal variables. Still, the representation of the matrix Φ is a challenge. Similar to Peng et al. (2012), ϕ and Φ are represented in terms of a moment matrix $\boldsymbol{M}$ (27) and Burer-Monteiro is invoked to represent $\boldsymbol{M}$ through a low-rank factorization $\boldsymbol{Y}\boldsymbol{Y}^\top$. In Peng et al. (2012), this low-rank representation led to the disappearance of the positive semidefinite constraint in the algorithm. However, in this case, one of the ADMM steps still involves projection of $\boldsymbol{Y}$ onto the positive semidefinite cone. However, $\boldsymbol{Y}$ is of rank r, only the top r eigenvalues and eigenvectors of an intermediate matrix needs to be computed using the Lanczos algorithm Golub & Van Loan (2013). The complexity of the Lanczos algorithm is governed by a matrix-vector multiplication operation, involving this intermediate matrix. Interestingly, the sparsity pattern of this matrix is driven by that of the graph $\mathcal{G}$, which can be advantageous for sparse graphs like grid graphs. Apart from these aspects, there is ample scope to parallelize this matrix-vector product. Similar to Peng et al. (2012), setting the rank parameter r is an open question. Huang et al. (2014) proposes to solve the problem to optimality repeatedly, doubling the value of r each time. The quality of the solution could be assessed based on the primal-dual gap of the original integral problem and the relaxed dual problem. Also, as the algorithm proceeds, the label for some of the nodes are fixed according to a reliability criterion. The label of a node is fixed if $\phi_i(x_i)$ is greater than a high threshold, like 0.99 or if $\phi_i(x_i)$ is the maximum $\forall i$ and $\forall x_i$. This heuristic improves the scalability of the algorithm.

Upon closer observation, one could notice that Peng et al. (2012) and Huang et al. (2014) are essentially having the same set of constraints. Both enforce the normalization and marginalization constraints of the LP relaxation and the constraints corresponding to the SDP relaxation. However, in the case of Huang et al. (2014) the marginalization constraints of the LP relaxation are enforced implicitly through other constraints. This enabled them to approach the optimization with a different algorithmic approach. Even though Huang et al. (2014) prove the convergence of their ADMM scheme when the moment matrix $\boldsymbol{M}$ is considered explicitly, they don't prove convergence when it is represented through the low-rank factorization, $\boldsymbol{Y}\boldsymbol{Y}^\top$. However, they report good empirical performance for the latter approach for various datasets.

So far, we have been seeing a way to tighten the SDP relaxation by adding linear constraints inspired by the LP relaxation. In the context of the LP relaxation, we saw that it could be increasingly tightened by a hierarchy of problems, called the Sherali-Adams hierarchy. A similar hierarchy exists for the SDP relaxation called the *sum-of-squares* or the *Lasserre* hierarchy Lasserre (2001), Parrilo (2003). Earlier we saw that even though, in theory, a given level of the Sherali-Adams hierarchy will involve all possible subsets of a given size, in practice, one could work with a smaller set of subsets like triplets or the faces of a grid. A similar idea to tighten the SDP relaxation in an efficient manner has been presented by Erdogdu et al. (2017). To understand their approach, we will look at a general way to characterize the moment matrix, $\boldsymbol{M}$. The following equations hold for the case, where the nodes are binary valued and take values 1 or -1. It is possible to extend them to the multi-label case based on larger moment matrices. In (27), the moment matrix carried the unary and pairwise pseudomarginals. In general, it could carry higher-order terms that will lead to a tighter relaxation Wainwright & Jordan (2004*a*). In more detail, given an even number d, we will consider all possible node subsets of size up to $\frac{d}{2}$, this included the null-set ($\varnothing$). We will have $\binom{n}{\leq \frac{d}{2}}$ such subsets. Thus, $\boldsymbol{M}$ is a square matrix of side $\left|\binom{n}{\leq \frac{d}{2}}\right|$. We see that the moment matrix grows exponentially in the size of d. The elements of $\boldsymbol{M}$ are indexed according to pairs of subsets of the nodes. Suppose, S and T are two such subsets, then, we index the corresponding element of $\boldsymbol{M}$ as $\boldsymbol{M}(S, T)$. The moment matrix $\boldsymbol{M}$ is desired to be a symmetric matrix. This constraints the elements of $\boldsymbol{M}$ to be a function of $S \Delta T$ only, where, Δ denotes the symmetric difference between two sets. The symmetry requirement can be expressed as follows,

$$\begin{aligned} S_1 \Delta T_1 = S_2 \Delta T_2 \quad &\Rightarrow \quad M(S_1, T_1) = M(S_2, T_2) \\ \forall S_1, T_1, S_2, T_2 \subseteq \mathcal{V}: \; &|S_1|, |T_1|, |S_2|, |T_2| \leq \frac{d}{2}. \end{aligned} \tag{32}$$

The moment matrix that we saw in (27) corresponds to the case of $d = 2$. In that case, we can see how the elements of $\boldsymbol{\phi}$ and Φ are indexed by (S, T), where $S, T \in \varnothing \cup \mathcal{V}$. Erdogdu et al. (2017) works with the case $d = 4$ and enforces the constraints on the elements of $\boldsymbol{M}$ dictated by (32). They achieve an efficient algorithm by invoking Burer-Monteiro and represent $\boldsymbol{M}$ through a low-rank factorization. They represent each row of the low-rank matrix as a vector $\boldsymbol{\sigma}_S \in \mathcal{R}^r$, where $S \subseteq \mathcal{V}: \; |S| \leq 2$. Thus, there will be $\left|\binom{n}{\leq 2}\right|$ such vectors. We will denote the set of node subsets as $\mathcal{S}$ and it will be made of nodes and pairs of nodes. We will see soon about the effect of the vector length r on the quality of the optimization result. Thus, we could express the constraints on the moment matrix $\boldsymbol{M}$ in terms of these vectors, as follows,

$$\|\boldsymbol{\sigma}_i\| = 1 \qquad \forall i \in \mathcal{S} \tag{33a}$$

$$\boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_j = \boldsymbol{\sigma}_{ij} \cdot \boldsymbol{\sigma}_\varnothing \qquad \forall i, j \in \mathcal{S} \tag{33b}$$

$$\boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_{ij} = \boldsymbol{\sigma}_j \cdot \boldsymbol{\sigma}_\varnothing \qquad \forall i, j \in \mathcal{S} \tag{33c}$$

$$\boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_{jk} = \boldsymbol{\sigma}_k \cdot \boldsymbol{\sigma}_{ij} \qquad \forall i, j, k \in \mathcal{S} \tag{33d}$$

$$\boldsymbol{\sigma}_{ij} \cdot \boldsymbol{\sigma}_{jk} = \boldsymbol{\sigma}_{ik} \cdot \boldsymbol{\sigma}_\varnothing \qquad \forall i, j, k \in \mathcal{S} \tag{33e}$$

$$\boldsymbol{\sigma}_{ij} \cdot \boldsymbol{\sigma}_{kl} = \boldsymbol{\sigma}_{ik} \cdot \boldsymbol{\sigma}_{jl} \qquad \forall i, j, k, l \in \mathcal{S} \tag{33f}$$

where $. \cdot .$ is the dot product operation and $i, j, k, l \in \mathcal{V}$. The constraint (33a) is a constraint on the diagonal elements of $\boldsymbol{M}$, which are all a function of the null-set ($\varnothing$). Similar to the ILP inspired constraint (28d), the diagonal elements of $\boldsymbol{M}$ are constrained to take the value 1, since, the nodes take labels in $\{-1, 1\}$. Note that the above equations are for the case of $d = 4$, when nodes take binary labels. When the same Burer-Monteiro approach is used for $d = 2$ with binary valued nodes, we will only have the constraint (33a), just like we saw in the case of Peng et al. (2012).

We would like to point out that the sets of nodes that are associated with the moment matrix, carry a similar meaning to clusters, which we came across in the LP relaxation. For example, if we are tightening the LP relaxation, we will be working with pseudomarginals corresponding to nodes, pairs of nodes, triplets of nodes etc. In an SDP (sum-of-squares) relaxation, we will have a moment matrix with elements corresponding to nodes, pairs of nodes, triplets of nodes, quadruplets of nodes etc. If we consider only upto quadruplets of nodes, we will denote the relaxation as SOS(4). We can see that SOS(2), leads to a moment matrix with elements corresponding to nodes and pairs of nodes. This is what we saw in (28). Also, SOS(2) corresponds to the pioneering SDP relaxation work of Goemans & Williamson (1995). It will be a good exercise to verify the structure of the moment matrix for SOS(2) and SOS(4).

The set of vectors $\boldsymbol{\sigma}_S$ satisfying the constraints in (33) is denoted as $\boldsymbol{\Omega}(\mathcal{S})$. Thus, the SOS(4) SDP relaxation for a binary valued pairwise graph takes the following form,

$$\underset{\boldsymbol{\sigma}_S, S \in \mathcal{S}}{\text{minimize}} \sum_{(i,j) \in \mathcal{E}} \theta_{ij} \boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_j + \sum_{i \in \mathcal{V}} \theta_i \boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_\varnothing \tag{34a}$$

$$\text{subject to } \boldsymbol{\sigma}_S \in \boldsymbol{\Omega}(\mathcal{S}) \ \ \forall S \in \mathcal{S} : |S| \leq 2. \tag{34b}$$

The set $\mathcal{S}$ is very big. Even when $d = 4$, it is of size $\left|\binom{n}{\leq 2}\right|$, i.e., we are considering all subsets $S : \ |S| \leq 2$. This is because we are considering all possible pairs of nodes. Just like we did in LP relaxation, while considering clusters, we could restrict the pairs of nodes that we consider. We could divide the graph into triplets of adjacent nodes. We can always create triplets by introducing edges with zero potential and we require these triplets to cover all the nodes and edges of the original graph. We will define a $\boldsymbol{\sigma}_S$, only if S, which is size 1 or 2, is contained in one of the triplets. We will call this restricted set as $\hat{\mathcal{S}}$. Erdogdu et al. (2017) attempts to solve the following more tractable problem,

$$\underset{\boldsymbol{\sigma}_S, S \in \hat{\mathcal{S}}}{\text{minimize}} \quad \sum_{(i,j)\in\mathcal{E}} \theta_{ij} \boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_j \ + \sum_{i\in\mathcal{V}} \theta_i \boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_\varnothing \tag{35a}$$

$$\text{subject to } \boldsymbol{\sigma}_S \in \boldsymbol{\Omega}(\hat{\mathcal{S}}) \ \ \forall S \in \hat{\mathcal{S}} : |S| \leq 2. \tag{35b}$$

We will now see more about the effect of the low-rank parameter r, which is the length of the vectors, $\boldsymbol{\sigma}_S$. If $r = \left|\binom{n}{\leq \frac{d}{2}}\right|$, then the SDP relaxation is convex. However, we could set r to low values and get acceptable results, while improving the memory footprint. Erdogdu et al. (2017) have obtained good results in their experiments when $r = 10$. Erdogdu et al. (2017) optimizes the resulting non-convex version of (35) by constructing an augmented Lagrangian, where only constraint (33a) is explicitly retained. This augmented Lagrangian is optimized in a coordinate descent fashion, where each time the minimization is with respect to one vector $\boldsymbol{\sigma}_S$. Once, optimality is reached, some of the nodes are assigned a label of 1 or -1 based on a rounding scheme that we will describe in section (5.1). Based on this, the $\boldsymbol{\sigma}_S$ vectors are fixed as $x_i^* \boldsymbol{\sigma}_\varnothing$ for those particular nodes and $x_i^* x_j^* \boldsymbol{\sigma}_\varnothing$ for edges between such nodes. Then, (35) is again optimized with respect to the remaining $\boldsymbol{\sigma}_S$ vectors. This process is repeated until all nodes are reliably labelled. We have seen similar shrinking of problem size in Huang et al. (2014).

There have been a few efforts to understand the Burer-Monteiro approach in the context of MAP inference. Frostig et al. (2014) has presented an analysis of the following setting: SOS(2) SDP relaxation of the binary pairwise graph. In other words, optimizing (28) over $\mathcal{SDP}(\mathcal{G}_{\{0,1\}})$. In this setting, they are able to prove that the output of the non-convex problem saturates for a low value of the rank r. Thus, it is possible to ensure that the non-convex Burer-Monteiro approach matches the convex SDP relaxation with a reasonably small rank r. Another work Mei et al. (2017) analysing SOS(2), has shown that it is possible to achieve $O(\frac{1}{r})$ relative error with respect to the global optimum for specific choices of the node and edge potentials. Generally, SDP relaxations become more useful with the addition of more linear constraints, either borrowed from LP relaxation or by going up the SOS hierarchy. It will be interesting to analyse the Burer-Monteiro approximation in more involved settings.

So far, we have been looking at how Burer-Monteiro based low-rank approximation leads to scalable SDP algorithms. We will now see another algorithmic approach to achieve a scalable SDP algorithm that is suitable for tighter MAP inference. We will look into the dual based approach called SDCUT. It is an algorithm presented by Wang et al. (2013) to solve an SDP relaxation for *binary quadratic programs* (BQPs). A BQP is a special case of the quadratic programming problem in (24), where nodes take only binary labels. Their approach, which is called SDCUT, could be extended to multi-label nodes and thus used for MAP inference, as they show in Wang et al. (2015, 2016). The advantages of this approach are its scalability and flexibility to accommodate additional linear inequality constraints to tighten the relaxation

Wang et al. (2016). SDCUT has also been useful for MAP inference in fully connected graphs Wang et al. (2015), which we will not discuss in detail here. First, we will briefly see the derivation of SDCUT. We observe that the constraint, $\Phi(i; x_i, i; x_i) = \phi_i(x_i)$ (28d), along with the normalization constraint on $\phi_i(x_i)$ (28b), could be equivalently expressed as the constraint, $\text{tr}\left(\boldsymbol{M}\right) = n + 1$, where n is the number of nodes. Where $\boldsymbol{M}$ is the moment matrix (27). $\boldsymbol{M}$ is also constrained to be symmetric and positive semidefinite. As a result of all these constraints, the following result holds true: $\|\boldsymbol{M}\|_F \leq n + 1$ Malick (2007). This inequality condition could be made a part of the objective in the SDP (28a) as a regularization term, $\sigma(\|\boldsymbol{M}\|_F^2 - (n+1)^2)$ with $\sigma > 0$. Thus, we have a regularized problem with the following form,

$$\underset{\boldsymbol{M}\succeq\boldsymbol{0}}{\text{minimize}} \quad \text{tr}\left(\boldsymbol{M}^\top \boldsymbol{A}\right) + \sigma\|\boldsymbol{M}\|_F^2 - \sigma(n+1)^2 \tag{36a}$$

$$\text{subject to} \quad \text{tr}\left(\boldsymbol{M}^\top \boldsymbol{B}_i\right) = b_i, \quad \forall i = 1, ..., p \tag{36b}$$

$$\text{tr}\left(\boldsymbol{M}^\top \boldsymbol{B}_j\right) \leq b_j, \quad \forall j = p+1, ..., m. \tag{36c}$$

Here, $\boldsymbol{A}$ is a matrix such that $\text{tr}\left(\boldsymbol{M}^\top \boldsymbol{A}\right)$ is equal to the objective in (28a). The linear equality constraints (36b) and the linear inequality constraints (36c) are written in a generic manner. They represent the linear constraints that are imposed in the SDP. For MAP inference, they are necessarily composed of the SDP relaxation constraints, (28b) and (28d). They could also consist of constraints borrowed from the LP relaxation literature. The dual to this problem takes the following form,

$$\underset{\boldsymbol{u}}{\text{maximize}} \quad \frac{1}{4\sigma}\|\boldsymbol{C}(\boldsymbol{u})_-\|_F^2 - \boldsymbol{u}^\top \boldsymbol{b} - \sigma(n+1)^2 \tag{37a}$$

$$\text{subject to} \quad u_j \geq 0, \quad \forall j = p+1, ..., m \tag{37b}$$

where, $\boldsymbol{C}(\boldsymbol{u}) = \sum_{i=1}^{m} u_i \boldsymbol{B}_i + \boldsymbol{A}$. Computing the gradient for this problem, involves SVD, since, $\boldsymbol{C}(\boldsymbol{u})_-$ is computed by considering only the negative eigenvalues of $\boldsymbol{C}(\boldsymbol{u})$. This is an expensive part of other algorithms based on SDP relaxation like Huang et al. (2014), also. Huang et al. (2014) ameliorated this problem by computing only the top eigenvectors and eigenvalues that is dictated by low-rank representation and also, by exploiting sparse matrix-vector product driven by the graph structure. Wang et al. (2016) resorts to a parallel eigendecomposition algorithm to achieve speed-up of the SVD.

Wang et al. (2016) continues with the theme of adding more linear constraints to the basic SDP problem (28), in order to obtain a tighter relaxation. So far, we have come across various linear constraints that are part of the local polytope based LP relaxation. Additionally, Wang et al. (2016) considers the following linear constraints that could be enforced. For the main diagonal blocks of Φ, apart from the constraints (28d), we could enforce the following constraints,

$$\Phi(i; x_i, i; x_j) = 0 \quad \forall i \in \mathcal{V}, \; x_i \neq x_j. \tag{38}$$

Intuitively, the off-diagonal blocks of Φ correspond to the pseudomarginals of pairs of nodes and we could enforce marginalization constraints and non-negativity constraints on them, like it is done in the LP relaxation. However, the main diagonal blocks of Φ are not suitable for similar constraints. The main diagonal blocks are an artifact of the semidefinite relaxation. We have already seen the constraints (28d) that could be enforced on the main diagonal. In a similar spirit, (38) enforces desired structure on Φ.

Further Wang et al. (2016) enforces cycle based contraints that we came across in tighter LP relaxations (6, ??). These cycle based constraints were obtained by the combinatorial optimization community, while studying the polyhedral nature of graphs with binary variables ($\mathcal{G}_{\{0,1\}}$) Schrijver (2003), Deza & Laurent (2009). The cycle inequalities that we came across earlier (6) is an example of constraints that could be enforced on a graph with binary variables. Wang et al. (2016) has considered these cycle inequalities and also, *triangular* and *odd-wheel* inequalties. Consider, three nodes i, j, k and let $\delta_{i,j} \triangleq \mathbb{1}[x_i \neq x_j]$ indicate whether two nodes disagree in their labelling, then the triangle inequalities take the following form,

$$\begin{aligned} \delta_{i,j} + \delta_{j,k} + \delta_{k,i} &\leq 2 \\ \delta_{i,j} - \delta_{j,k} - \delta_{k,i} &\leq 0 \\ -\delta_{i,j} + \delta_{j,k} - \delta_{k,i} &\leq 0 \\ -\delta_{i,j} - \delta_{j,k} + \delta_{k,i} &\leq 0. \end{aligned} \tag{39}$$

Just like a cycle inequality, an odd-wheel inequality is defined on a cycle C with odd length of at least 3. These, inequalities take the following form,

$$\sum_{(i,j)\in C} \delta_{i,j} - \sum_{i \in \mathcal{V}(C)} \delta_{i,k} \leq \frac{|C|-1}{2} \tag{40}$$

where k is a node of the graph that is not in C but encircled by it. Like cycle inequalities, there are an exponential number of odd-wheel inequalities. Also, similar, to cycle inequalities there is a polynomial time separation algorithm for finding the cycle that most violate the odd-wheel inequality Deza & Laurent (2009).

Thus, Wang et al. (2016) has presented a tighter MAP inference algorithm based on the dual given in (37a), along with suitable linear equality and inequality constraints. One of the issues to address is making this approach work for a graph with multi-label nodes. Wang et al. (2016) finds a solution by constructing the l-projection graph Sontag & Jaakkola (2007) that we saw in section (1.1) and by considering the above constraints w.r.t. that graph. However, this leads to another issue to be addressed. It is the huge problem size that happens with large graphs and nodes with multiple labels. They take a branch and bound approach to reduce the set of active labels for each node Sun et al. (2012). Also, the huge number of constraints are handled in a cutting-planes framework. Thus, the dual problem is solved several times in this framework, where the label space is pruned each time, by updating lower and upper bounds for each node. The overall algorithm stops when these bounds are sufficiently close for all nodes.

To further improve the scalability, Wang et al. (2016) removes nodes based on persistency of a node label with respect to the optimal labelling. There have been several works regarding persistency in MRF inference Kovtun (2003), Kohli et al. (2008), Swoboda et al. (2016). Even though, obtaining the optimal labelling of all the nodes is an NP-hard problem, it is possible to identify whether a node has attained its optimal label or not. If it is true, we say this node has attained its persistent label. With this guarantee, it is possible to reduce the problem size and repeat the inference algorithm for the remaining nodes. Through this process, it is possible to obtain better overall labelling. Among various approaches, there is an algorithm that is referred to as QPBO (Blake et al. 2011, §2.3). It works with binary valued graphs and assigns labels that are one of $\{0, 0.5, 1\}$. Whenever, a node receives an integral label, it is guaranteed to be its optimal label. Wang et al. (2016) has chosen QPBO based on its simplicity and efficiency.

The third important issue to be addressed, is an efficient way to handle the various linear equality and inequality constraints. This step is the most important from a tighter relaxation point of view. Some of the constraints are limited in total number and the violated ones are found by enumeration. This is applicable to non-negativity (31), marginalization (3b) and triangle inequality (39) constraints. Cycle inequalities are added based on the seperation algorithm given in Deza & Laurent (2009). If no violated cycle inequalities are present, then the algorithm searches for violated odd-wheel inequalities. The separation algorithm for the latter is also described in Deza & Laurent (2009). Each added constraint corresponds to a new dual variable and it is important to control the problem size. Thus, it is important to remove inactive constraints. The heuristic used by Wang et al. (2016) is to remove constraints which are satisfied by the current primal variables and whose dual variables are zero. We would like to point out that Huang et al. (2014) has shown how the marginalization constraints are implicitly enforced by the semidefinite constraint along with the normalization and non-negativity constraints. Even though Wang et al. (2016) works with a perturbed problem, this condition will hold with more guarantee as the algorithm reaches the optimum. Thus, marginalization constraints need not be considered at later stages of the algorithm.

5.1 Rounding schemes for SDP relaxation

SDP relaxations have a rich array of rounding schemes to obtain integral labels for the nodes of the graph Goemans & Williamson (1995), Chazelle et al. (2004), Charikar et al. (2009), Raghavendra (2011). Some SDP relaxation algorithms are designed such that we obtain pseudomarginals for the nodes Peng et al. (2012), Huang et al. (2014), Wang et al. (2016). Then, we could assign a label to the node by sampling according to the probability distribution defined by the pseudomarginals. In fact, this is one of the possible rounding schemes for LP relaxations working with pseudomarginals Ravikumar et al. (2010). This *naive randomized* scheme works better when the distribution of the graphical model is more concentrated.

Several rounding schemes are based on low-rank representation of the moment matrix, i.e., $\boldsymbol{M} = \boldsymbol{Y}\boldsymbol{Y}^\top$, where $\boldsymbol{Y}$ is of rank r. So far, we have come across two kinds of node labels: binary ($\{1, -1\}$) or multi-label. For binary labels only one row of Y is needed per node. For the multi-label case, one row of $\boldsymbol{Y}$ is needed for each node and each label. While discussing the work of Erdogdu et al. (2017), we used $\boldsymbol{\sigma}_{i;x_i}$ to denote the row of Y corresponding to node i and label x_i. We could design rounding schemes based on projecting $\boldsymbol{\sigma}_{i;x_i}$ onto a suitable vector. For some of the rounding schemes, we also need the vector $\boldsymbol{\sigma}_\varnothing$ that corresponds to the first row of $\boldsymbol{Y}$.

First, we will discuss some rounding schemes for binary valued nodes. The seminal work Goemans & Williamson (1995) proposed to draw a random vector $\boldsymbol{w} \in \mathcal{R}^{1\times r}$ such that $\|\boldsymbol{w}\| = 1$ and assigns to node i the label $\text{sign}(\boldsymbol{\sigma}_i \cdot \boldsymbol{w})$. This is called the *random hyperplane rounding* scheme. *Shifted random projection rounding* extends this idea further and has proved to be a necessary step within approximation algorithms for MAX-2SAT Charikar et al. (2009). Peng et al. (2012) has shown results based on this scheme. A perturbation vector $\boldsymbol{p} \in \mathcal{R}^{1\times r}$ is constructed with i.i.d. normally distributed elements. Then node i is assigned the label $\text{sign}(\boldsymbol{\sigma}_i \cdot ((1-\beta)\boldsymbol{\sigma}_\varnothing + \beta\boldsymbol{p})$. We should refer Charikar et al. (2009) about setting β to a value between 0 and 1. Note that $\boldsymbol{\sigma}_\varnothing$ itself is of unit length. So, there are rounding schemes that work with $\boldsymbol{\sigma}_\varnothing$, instead of generating a random vector $\boldsymbol{w}$ of unit length.

Erdogdu et al. (2017) also works with $\boldsymbol{\sigma}_\varnothing$ and it fixes the label of a node as $\text{sign}(\boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_\varnothing)$, if $|\boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_\varnothing|$ exceeds a confidence threshold. We saw that Erdogdu et al. (2017) optimizes (35) several times, each time with a smaller set of $\boldsymbol{\sigma}_S$ vectors. Suppose there is an outer iteration, where no node exceeds the confidence threshold, then the nodes with the highest $|\boldsymbol{\sigma}_i \cdot \boldsymbol{\sigma}_\varnothing|$ value are picked for pruning. Erdogdu et al. (2017) presented their work for nodes taking binary labels but these rounding schemes could be extended to multi-label nodes. For example, we could assign the node i the label x_i that maximizes $|\boldsymbol{\sigma}_{i;x_i} \cdot \boldsymbol{\sigma}_{\varnothing}|$ or that maximizes $|\boldsymbol{\sigma}_{i;x_i} \cdot ((1-\beta)\boldsymbol{\sigma}_{\varnothing} + \beta \boldsymbol{p})|$. Also, there are other advanced rounding schemes, like those based on Grothendieck's inequality Braverman et al. (2013) which we will not discuss here.

5.2 Problems where SDP relaxation has helped

We will discuss problem instances where SDP relaxation based algorithms lead to better MAP labelling. Usually, these algorithms achieve a tighter relaxation by moving up the Lasserre/SOS hierarchy or by adding constraints that tighten the LP relaxation like cycle or cluster based constraints.

Wang et al. (2015, 2016) have shown that SDP relaxation based methods perform better for densely connected graphs with weak unary potentials and for graphs that have a large fraction of non-submodular (also, called non-attractive) pairwise potentials. Wang et al. (2016) was able to perform better than DAOOPT Otten & Dechter (2012) in a problem with a fully connected graph and no unaries from the Probabilistic Inference Challenge, 2011 PIC (2011). DAOOPT was a top performer in PIC-2011. Huang et al. (2014) has also observed that their method performs well on dense problems from PIC-2011. They were able to do better that ficolofo Cooper et al. (2010), a top performer with dense problems in PIC-2011.

Peng et al. (2012) has also observed that problems where edge potentials are strong are solved better by their approach. Also, when the edge potentials are not so strong, they observed that the cluster based tightening of LP relaxation by Sontag et al. (2008) performed better. It is readily possible to define a feasible set that is the intersection of the SDP relaxation with the cluster based tighter LP relaxation that we saw in section (2). To achieve this one could use the optimization scheme given in Sontag et al. (2008) instead of convex belief propagation to solve the subproblem within the dual optimization scheme of Peng et al. (2012).

Scalable and efficient SDP algorithms for MAP inference are still evolving. The four main algorithms that we have discussed Peng et al. (2012), Huang et al. (2014), Wang et al. (2016), Erdogdu et al. (2017) are good starting points to further explore this topic. At this moment, we would like to point out one particular area of research: the intersection between spectral graph theory and semidefinite programming. For the MAP labelling problems discussed so far, we could explore algorithms based on spectral methods Cour & Shi (2007), Olsson et al. (2007). Even though spectral methods are computationally more efficient, the optimum of the SDP relaxation can be characterized better. There have been attempts to combine the advantages of the two methods and it is an active area of research in combinatorial optimization Kolla (2009), ICERM (2014), SIMONS INSTITUTE (2017).

6 Characterizing Tight Relaxations

In section (4), we came across particular graph topologies and clique potentials, which by themselves or in combination lead to exact MAP inference. It will be interesting to understand more general settings. Especially, settings where the relaxation tightening algorithms that we have discussed, will lead to exact MAP inference.

So far, we have been discussing about various ways to tighten LP and SDP relaxations. The Sherali-Adams hierarchy and the Lasserre hierarchy provide a guaranteed approach to tighten the LP relaxation and the SDP relaxation, respectively. There have been efforts to characterize graph topologies and/or clique potentials that could be solved exactly with schemes that are at lower levels of these hierarchies. However, these hierarchies are computationally heavy even at the lower levels and they are enforced only partially. Surprisingly, there are many problem instances which are solved exactly with such partial schemes.

In the Sherali-Adams hierarchy, a level r corresponds to enforcing the marginalization constraint (5b) over all clusters of variables of size $\leq r$ and we will denote the resulting polytope as $\mathcal{SA}_r(\mathcal{G})$. The local polytope, $\mathcal{L}(\mathcal{G})$ is related to $\mathcal{SA}_2(\mathcal{G})$. Even though, $\mathcal{L}(\mathcal{G})$ enforces marginalization constraints only on edges of the graph and not on all possible pairs of nodes, it can be shown to be exactly equivalent to $\mathcal{SA}_2(\mathcal{G})$ (Rowland et al. 2017, §3.3). Rowland et al. (2017) have shown a similar result for $\mathcal{SA}_3(\mathcal{G})$: the polytope obtained by considering the marginalization constraints of $\mathcal{L}(\mathcal{G})$ together with the marginalization constraints of triplets obtained by triangulating the graph, is equivalent to $\mathcal{SA}_3(\mathcal{G})$.

Further, it is desirable to characterize the tightness of $\mathcal{SA}_r(\mathcal{G})$ for various values of r. Initially, there were efforts to study conditions for tightness based on the graph topology. Wainwright & Jordan (2004*b*) proved that a sufficient condition for $\mathcal{SA}_r(\mathcal{G})$ to be tight is to have a graph with treewidth $\leq r-1$. We can immediately see that $\mathcal{L}(\mathcal{G})$ is tight for trees. Since, trees have treewidth 1. Chandrasekaran et al. (2008) proved that in the more difficult problem of inferring the marginal probabilities, if there are no restrictions on the clique potentials, then bounded treewidth is necessary for tractability. Similar studies focusing on clique potentials have also been pursued. Thapper & Živný (2016) showed that for a given family of potentials (finite multi-label nodes and any clique size), either $\mathcal{L}(\mathcal{G})$ is tight, i.e., all such problems are tractable irrespective of the graph topology or the problem set is NP-hard. Tractability for graphical models containing only submodular clique potentials is a result that fits in this line of work.

In recent years, settings combining restrictions on graph topology and clique potentials have been studied Weller et al. (2016), Rowland et al. (2017). Rowland et al. (2017) and related works have considered binary pairwise models in their analysis. In this setting, every graph is associated with a *signed* graph of the same topology, where an edge is considered even if it is attractive and it is odd if it is not attractive. A *balanced* graph is one that could be partitioned into two exhaustive sets, such that all edges with both endpoints within any one set are even and edges with one endpoint in each set is odd. In other words, any cycle in a balanced graph, will have an even number of odd edges. Thus, a balanced graph does not have frustrated cycles. Many results that are satisfied by models with attractive potentials may be proven for balanced models, which form a wider class. An *almost balanced* graph contains no frustrated cycles except through one privileged node, s. Thus, removing that node makes the remaining graph balanced. Note, it is possible to efficiently test whether a graph is almost balanced. Weller et al. (2016), Rowland et al. (2017) have shown that for balanced graphs, $\mathcal{SA}_3(\mathcal{G})$ as defined in the above more compact manner is sufficient to achieve a tight relaxation. This is useful for applications like foreground/background subtraction, which involve binary labels and the foreground and background form the two sets of the balanced graph. For an almost balanced graph, we could define a polytope $\mathcal{SA}_3^s(\mathcal{G})$: the polytope obtained by considering the marginalization constraints of $\mathcal{L}(\mathcal{G})$ together with the marginalization constraints over only triplets which contain s, the privileged node. Rowland et al. (2017) has shown that $\mathcal{SA}_3^s(\mathcal{G})$ is tight for an almost balanced graph with a privileged node s.

Similar results have been obtained for the SOS hierarchy for SDPs. While discussing the work of Erdogdu et al. (2017), we discussed about the more tractable problem given in (35). Here, $\hat{\mathcal{S}}$ is made of a set of regions such that together they cover all the nodes of the graph. If each chordless cycle of the graph is contained in at least one region in $\hat{\mathcal{S}}$, we say that the set of regions is *circular*. To present their result, Erdogdu et al. (2017) makes the assumptions that the graph has zero valued unary potentials and its topology is not contractible to K_5. Given these assumptions they show that the problem in (35) defined over a circular set of covering regions is tight. They also show how to convert a graph with non-zero unary potentials to a graph with zero unary potentials but the resulting graph could be contractible to K_5. Note that the empirical results in Erdogdu et al. (2017) concern graphs which do not satisfy the assumptions, the set of covering regions do not contain all chordless cycles and they work with graphs having non-zero unary potentials for which the conversion to zero unary potentials could result in a graph that could be contractible to K_5. Their empirical results have exceeded the theoretical expectation and need further investigation.

We have given here only a taste for an active area of research. For example, Wainwright & Jordan (2004*b*) gave only a sufficient condition based on treewidth regarding the tightness of the Sherali-Adams hierarchy. Rowland et al. (2017) has presented analytical results which indicate that there are conditions beyond treewidth. Also, it is not necessary to study relaxation hierarchies based on exactness of the relaxation. Even if we manage to achieve low approximation error at a given level of the relaxation hierarchy, rounding schemes can be used to further improve the output Chlamtac & Tulsiani (2012).

7 Conclusion

Continuous relaxations for MAP inference in MRFs are an important topic of study with wide applications. Obtaining better MAP labelling in an efficient manner is the ultimate objective of this line of work. The possibility of better labelling will increase the scope of applications for MRFs with various graph topologies and clique potentials. In this survey we especially better understood approaches based on LP and SDP relaxations. We discussed various algorithmic steps that could lead to tighter relaxation in both cases. In the process, we understood the theoretical underpinnings better. This is a topic where there is scope for further research both from an algorithmic point of view and from a theoretical analysis point of view. A very important perspective is provided by applications, which explore novel clique potentials and graph topologies. The effect of tighter relaxation based inference within learning of structured prediction models is also an important avenue of research. Meshi et al. (2019) presents an interesting analysis about the empirical observation of tight LP relaxations within structured prediction applications. Learning a structured prediction model is computationally heavy, which involves inference in each iteration of the process (called loss augmented inference). Efficient algorithms for tighter relaxation could further the study of this topic. At this juncture, we would like to highlight the scope for further research in SDP relaxation based algorithms. The literature is much more evolving in that topic compared to algorithms based on LP relaxation.

References

Batra, D., Gallagher, A. C., Parikh, D. & Chen, T. (2010), Beyond trees: Mrf inference via outer-planar decomposition, *in* 'IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 21, 22

Batra, D., Nowozin, S. & Kohli, P. (2011), Tighter relaxations for map-mrf inference: A local primal-dual gap based separation algorithm, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 13

Batra, D., Yadollahpour, P., Guzman-Rivera, A. & Shakhnarovich, G. (2012), Diverse m-best solutions in markov random fields, *in* 'European Conference on Computer Vision (ECCV)'. 22

Bergtholdt, M., Kappes, J., Schmidt, S. & Schnörr, C. (2010), 'A study of parts-based object class detection using complete graphs', *International Journal of Computer Vision (IJCV)* **87**(1-2), 93. 22

Blake, A., Kohli, P. & Rother, C. (2011), *Markov random fields for vision and image processing*, Mit Press. 1, 2, 9, 32

Boumal, N., Voroninski, V. & Bandeira, A. (2016), The non-convex burer-monteiro approach works on smooth semidefinite programs, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 26

Boumal, N., Voroninski, V. & Bandeira, A. S. (2018), 'Deterministic guarantees for burer-monteiro factorizations of smooth semidefinite programs', *arXiv preprint arXiv:1804.02008* . 26

Boyd, S., Parikh, N., Chu, E., Peleato, B. & Eckstein, J. (2011), 'Distributed optimization and statistical learning via the alternating direction method of multipliers', *Foundations and Trends® in Machine learning* **3**(1), 1–122. 27

Boyd, S. & Vandenberghe, L. (2004), *Convex optimization*, Cambridge university press. 25

Braverman, M., Makarychev, K., Makarychev, Y. & Naor, A. (2013), The grothendieck constant is strictly smaller than krivine's bound, *in* 'Forum of Mathematics, Pi', Vol. 1, Cambridge University Press. 34

Burer, S. & Monteiro, R. D. C. (2003), 'A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization', *Mathematical Programming* **95**(2), 329–357. 26

Burer, S. & Monteiro, R. D. C. (2005), 'Local minima and convergence in low-rank semidefinite programming', *Mathematical Programming* **103**(3), 427–444. 26

Chandrasekaran, V., Srebro, N. & Harsha, P. (2008), Complexity of inference in graphical models, *in* 'Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI)'. 35

Charikar, M., Makarychev, K. & Makarychev, Y. (2009), 'Near-optimal algorithms for maximum constraint satisfaction problems', *ACM Transactions on Algorithms (TALG)* **5**(3), 32. 33

Chazelle, B., Kingsford, C. & Singh, M. (2004), 'A semidefinite programming approach to side chain positioning with new rounding strategies', *INFORMS Journal on Computing* **16**(4), 380–392. 33

Chlamtac, E. & Tulsiani, M. (2012), Convex relaxations and integrality gaps, *in* 'Handbook on semidefinite, conic and polynomial optimization', Springer, pp. 139–169. 36

Cooper, M. C., De Givry, S., Sánchez, M., Schiex, T., Zytnicki, M. & Werner, T. (2010), 'Soft arc consistency revisited', *Artificial Intelligence* **174**(7-8), 449–478. 34

Cour, T. & Shi, J. (2007), Solving markov random fields with spectral relaxation, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 34

Deza, M. M. & Laurent, M. (2009), *Geometry of cuts and metrics*, Vol. 15, Springer. 6, 32, 33

Duchi, J. C., Tarlow, D., Elidan, G. & Koller, D. (2007), Using combinatorial optimization within max-product belief propagation, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 21

Erdogdu, M. A., Deshpande, Y. & Montanari, A. (2017), Inference in graphical models via semidefinite programming hierarchies, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 28, 29, 30, 33, 34, 35, 36

Felzenszwalb, P. F. & McAuley, J. J. (2011), 'Fast inference with min-sum matrix product', *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **33**(12), 2549–2554. 19

Frostig, R., Wang, S., Liang, P. S. & Manning, C. D. (2014), Simple map inference via low-rank relaxations, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 30

Globerson, A. & Jaakkola, T. S. (2008), Fixing max-product: Convergent message passing algorithms for map LP-relaxations, *in* 'Advances in neural information processing systems (NIPS)'. 11

Goemans, M. X. & Williamson, D. P. (1995), 'Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming', *Journal of the ACM (JACM)* **42**(6), 1115–1145. 25, 29, 33

Golub, G. H. & Van Loan, C. F. (2013), *Matrix computations, 4th edition*, JHU Press. 28

Hazan, T. & Shashua, A. (2008), Convergent message-passing algorithms for inference over general graphs with convex free energies, *in* 'Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI)'. 26

Huang, Q., Chen, Y. & Guibas, L. (2014), Scalable semidefinite relaxation for maximum a posterior estimation, *in* 'International Conference on Machine Learning (ICML)'. 27, 28, 30, 31, 33, 34

ICERM (2014), 'Semidefinite programming and graph algorithms', `https://icerm.brown.edu/programs/sp-s14/w1/`. Accessed: 2018-08-07. 34

Johnson, J. K. (2008), Convex relaxation methods for graphical models: Lagrangian and maximum entropy approaches, PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science. 17, 18

Kappes, J. H., Andres, B., Hamprecht, F. A., Schnörr, C., Nowozin, S., Batra, D., Kim, S., Kausler, B. X., Kröger, T., Lellmann, J., Komodakis, N., Savchynskyy, B. & Rother, C. (2015), 'A comparative study of modern inference techniques for structured discrete energy minimization problems', *International Journal of Computer Vision* **115**(2), 155–184.
URL: *http://dx.doi.org/10.1007/s11263-015-0809-x* 2, 9, 21

Kappes, J. H., Schmidt, S. & Schnörr, C. (2010), Mrf inference by k-fan decomposition and tight lagrangian relaxation, *in* 'European Conference on Computer Vision (ECCV)'. 22

Kohli, P., Shekhovtsov, A., Rother, C., Kolmogorov, V. & Torr, P. (2008), On partial optimality in multi-label mrfs, *in* 'International Conference on Machine Learning (ICML)'. 32

Kolla, A. (2009), Merging Techniques for Combinatorial Optimization: Spectral Graph Theory and Semidefinite Programming, PhD thesis, UC Berkeley. 34

Koller, D. & Friedman, N. (2009), *Probabilistic graphical models: principles and techniques*, MIT press. 1, 9

Kolmogorov, V. (2006), 'Convergent tree-reweighted message passing for energy minimization', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **28**(10), 1568–1583. 13

Komodakis, N., Kumar, M. P. & Paragios, N. (2016), '(hyper)-graphs inference through convex relaxations and move making algorithms: Contributions and applications in artificial vision', *Foundations and Trends® in Computer Graphics and Vision* **10**(1), 1–102. 2

Komodakis, N. & Paragios, N. (2008), Beyond loose LP-relaxations: Optimizing mrfs by repairing cycles, *in* 'European conference on computer vision', pp. 806–820. 5, 15, 16, 17

Komodakis, N. & Paragios, N. (2009), Beyond pairwise energies: Efficient optimization for higher-order mrfs, *in* 'IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 21, 23

Komodakis, N., Paragios, N. & Tziritas, G. (2011), 'MRF energy minimization and beyond via dual decomposition', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **33**(3), 531–552. 8, 14, 20, 26

Kovtun, I. (2003), Partial optimal labeling search for a np-hard subclass of (max,+) problems, *in* 'Joint Pattern Recognition Symposium', Springer, pp. 402–409. 32

Kumar, M. P., Kolmogorov, V. & Torr, P. H. S. (2009), 'An analysis of convex relaxations for map estimation of discrete mrfs', *Journal of Machine Learning Research* **10**(Jan), 71–106. 3, 23

Lasserre, J. B. (2001), 'Global optimization with polynomials and the problem of moments', *SIAM Journal on optimization* **11**(3), 796–817. 24, 28

Laurent, M. (2003), 'A comparison of the sherali-adams, lovász-schrijver, and lasserre relaxations for 0–1 programming', *Mathematics of Operations Research* **28**(3), 470–496. 5

Li, M., Shekhovtsov, A. & Huber, D. (2016), Complexity of discrete energy minimization problems, *in* 'European Conference on Computer Vision (ECCV)'. 2

Li, Y. & Zemel, R. (2014), High order regularization for semi-supervised learning of structured output problems, *in* 'International Conference on Machine Learning (ICML)'. 22

Malick, J. (2007), 'The spherical constraint in boolean quadratic programs', *Journal of Global Optimization* **39**(4), 609–622. 31

McAuley, J. J. & Caetano, T. S. (2011), 'Faster algorithms for max-product message-passing', *Journal of Machine Learning Research* **12**(Apr), 1349–1388. 19

Mei, S., Misiakiewicz, T., Montanari, A. & Oliveira, R. I. (2017), Solving sdps for synchronization and maxcut problems via the grothendieck inequality, *in* 'Conference on Learning Theory', pp. 1476–1515. 26, 30

Meshi, O., Jaakkola, T. & Globerson, A. (2014), 'Smoothed coordinate descent for map inference', *Advanced Structured Prediction* pp. 103–131. 11, 12

Meshi, O., London, B., Weller, A. & Sontag, D. (2019), 'Train and test tightness of LP relaxations in structured prediction', *Journal of Machine Learning Research* **20**(13), 1–34. 36

Mezuman, E., Tarlow, D., Globerson, A. & Weiss, Y. (2013), Tighter linear program relaxations for high order graphical models, *in* 'Uncertainty in Artificial Intelligence (UAI)'. 23

Mezuman, E. & Weiss, Y. (2012), Globally optimizing graph partitioning problems using message passing, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 22

Noorshams, N. & Wainwright, M. J. (2013), 'Stochastic belief propagation: A low-complexity alternative to the sum-product algorithm', *IEEE Transactions on Information Theory* **59**(4), 1981–2000. 21

Olsson, C., Eriksson, A. P. & Kahl, F. (2007), Solving large scale binary quadratic problems: Spectral methods vs. semidefinite programming, *in* 'IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 34

Otten, L. & Dechter, R. (2012), 'Anytime and/or depth-first search for combinatorial optimization', *AI Communications* **25**(3), 211–227. 34

Parrilo, P. A. (2003), 'Semidefinite programming relaxations for semialgebraic problems', *Mathematical programming* **96**(2), 293–320. 24, 28

Pataki, G. (1998), 'On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues', *Mathematics of operations research* **23**(2), 339–358. 26

Peng, J., Hazan, T., Srebro, N. & Xu, J. (2012), Approximate inference by intersecting semidefinite bound and local polytope, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 26, 27, 28, 29, 33, 34

PIC (2011), 'Probabilistic inference challenge', `http://www.cs.huji.ac.il/project/PASCAL/index.php/`. Accessed: 2018-08-07. 34

Potetz, B. & Lee, T. S. (2008), 'Efficient belief propagation for higher-order cliques using linear constraint nodes', *Computer Vision and Image Understanding* **112**(1), 39–54. 21

Raghavendra, P. (2011), Generic techniques to round sdp relaxations, *in* 'International Symposium on Mathematical Foundations of Computer Science', Springer, pp. 34–34. 33

Ravikumar, P., Agarwal, A. & Wainwright, M. J. (2010), 'Message-passing for graph-structured linear programs: Proximal methods and rounding schemes', *Journal of Machine Learning Research* **11**(Mar), 1043–1080. 12, 33

Ravikumar, P. & Lafferty, J. (2006), Quadratic programming relaxations for metric labeling and markov random field map estimation, *in* 'International Conference on Machine learning (ICML)'. 24

Rother, C., Kohli, P., Feng, W. & Jia, J. (2009), Minimizing sparse higher order energy functions of discrete variables, *in* 'IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 18, 21, 23

Rowland, M., Pacchiano, A. & Weller, A. (2017), Conditions beyond treewidth for tightness of higher-order LP relaxations, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 5, 35, 36

Savchynskyy, B. & Schmidt, S. (2014), 'Getting feasible variable estimates from infeasible ones: Mrf local polytope study', *Advanced Structured Prediction* pp. 133–158. 12

Schraudolph, N. N. & Kamenetsky, D. (2009), Efficient exact inference in planar ising models, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 20, 22

Schrijver, A. (2003), *Combinatorial optimization: polyhedra and efficiency*, Vol. 24, Springer Science & Business Media. 5, 9, 32

Shimony, S. E. (1994), 'Finding maps for belief networks is np-hard', *Artificial Intelligence* **68**(2), 399–410. 2

Simons Institute (2017), 'Spectral graph theory and optimization symposium', `https://simons.berkeley.edu/workshops/spectral-graph-theory-and-optimization/`. Accessed: 2018-08-07. 34

Sontag, D., Choe, D. K. & Li, Y. (2012), Efficiently searching for frustrated cycles in map inference, *in* 'Uncertainty in Artificial Intelligence (UAI)'. 17, 18, 21

Sontag, D., Globerson, A. & Jaakkola, T. (2011), 'Introduction to dual decomposition for inference', *Optimization for Machine Learning* **1**, 219–254. 8

Sontag, D., Globerson, A. & Jaakkola, T. S. (2009), Clusters and coarse partitions in LP relaxations, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 14

Sontag, D. & Jaakkola, T. S. (2007), New outer bounds on the marginal polytope, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 5, 6, 7, 10, 17, 18, 32

Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T. & Weiss, Y. (2008), Tightening LP relaxations for map using message passing, *in* 'Uncertainty in Artificial Intelligence (UAI)'. 5, 10, 11, 12, 13, 14, 15, 17, 18, 19, 34

Sun, M., Telaprolu, M., Lee, H. & Savarese, S. (2012), Efficient and exact map-mrf inference using branch and bound, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 32

Swoboda, P., Shekhovtsov, A., Kappes, J. H., Schnorr, C. & Savchynskyy, B. (2016), 'Partial optimality by pruning for map-inference with general graphical models', *IEEE transactions on pattern analysis and machine intelligence (PAMI)* **38**(7), 1370–1382. 32

Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M. & Rother, C. (2008), 'A comparative study of energy minimization methods for markov random fields with smoothness-based priors', *IEEE transactions on pattern analysis and machine intelligence (PAMI)* **30**(6), 1068–1080. 2

Tarlow, D., Givoni, I. & Zemel, R. (2010), HOP-MAP: Efficient message passing with high order potentials, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 21, 23

Thapper, J. & Živný, S. (2016), 'The complexity of finite-valued csps', *Journal of the ACM (JACM)* **63**(4), 37. 5, 10, 35

Wainwright, M. J., Jaakkola, T. S. & Willsky, A. S. (2005), 'Map estimation via agreement on trees: message-passing and linear programming', *IEEE Transactions on Information Theory* **51**(11), 3697–3717. 2, 4, 10

Wainwright, M. J. & Jordan, M. I. (2004*a*), Semidefinite relaxations for approximate inference on graphs with cycles, *in* 'Advances in Neural Information Processing Systems (NIPS)'. 25, 27, 28

Wainwright, M. J. & Jordan, M. I. (2004*b*), Treewidth-based conditions for exactness of the sherali-adams and lasserre relaxations, Technical report, Technical Report 671, University of California, Berkeley. 5, 35, 36

Wainwright, M. J. & Jordan, M. I. (2008), 'Graphical models, exponential families, and variational inference', *Foundations and Trends® in Machine Learning* **1**(1–2), 1–305. 1, 5, 6, 7, 21, 24, 26

Wang, H. & Koller, D. (2013), A fast and exact energy minimization algorithm for cycle mrfs, *in* 'International Conference on Machine Learning (ICML)'. 19, 20

Wang, P., Shen, C. & Van Den Hengel, A. (2013), A fast semidefinite approach to solving binary quadratic problems, *in* 'IEEE conference on computer vision and pattern recognition (CVPR)'. 30

Wang, P., Shen, C. & van den Hengel, A. (2015), Efficient sdp inference for fully-connected crfs based on low-rank decomposition, *in* 'IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 30, 31, 34

Wang, P., Shen, C., van den Hengel, A. & Torr, P. H. S. (2016), 'Efficient semidefinite branch-and-cut for map-mrf inference', *International Journal of Computer Vision (IJCV)* **117**(3), 269–289. 26, 30, 31, 32, 33, 34

Weller, A., Rowland, M. & Sontag, D. (2016), Tightness of LP relaxations for almost balanced models, *in* 'International Conference on Artificial Intelligence and Statistics (AISTATS)'. 5, 35

Wen, Z., Goldfarb, D. & Yin, W. (2010), 'Alternating direction augmented lagrangian methods for semidefinite programming', *Mathematical Programming Computation* **2**(3-4), 203–230. 27

Werner, T. (2007), 'A linear programming approach to max-sum problem: A review', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**(7), 1165–1179. 16

Werner, T. (2008), High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (map-mrf), *in* 'IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 5, 10, 12, 13, 14

Werner, T. (2010), 'Revisiting the linear programming relaxation approach to gibbs energy minimization and weighted constraint satisfaction', *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **32**(8), 1474–1488. 10

Yanover, C., Meltzer, T. & Weiss, Y. (2006), 'Linear programming relaxations and belief propagation–an empirical study', *Journal of Machine Learning Research* **7**(Sep), 1887–1907. 7, 14

Yarkony, J., Morshed, R., Ihler, A. T. & Fowlkes, C. C. (2011), Tightening mrf relaxations with planar subproblems, *in* 'Uncertainty in Artificial Intelligence (UAI)'. 19, 20, 21

Yedidia, J. S., Freeman, W. T. & Weiss, Y. (2005), 'Constructing free-energy approximations and generalized belief propagation algorithms', *IEEE Transactions on information theory* **51**(7), 2282–2312. 6

Zheng, Y., Chen, P. & Cao, J.-Z. (2012), Map-mrf inference based on extended junction tree representation, *in* 'IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 22