Fast Approximate Inference in Higher Order MRF-MAP Labeling Problems

Chetan Arora *
The Hebrew University of Jerusalem
Jerusalem, Israel

Subhashis Banerjee Prem Kalra S.N. Maheshwari
Indian Institute of Technology Delhi
New Delhi, India

Abstract

Use of higher order clique potentials for modeling inference problems has exploded in last few years. The algorithmic schemes proposed so far do not scale well with increasing clique size, thus limiting their use to cliques of size at most 4 in practice. Generic Cuts (GC) of Arora et al. [9] shows that when potentials are submodular, inference problems can be solved optimally in polynomial time for fixed size cliques. In this paper we report an algorithm called Approximate Cuts (AC) which uses a generalization of the gadget of GC and provides an approximate solution to inference in 2-label MRF-MAP problems with cliques of size $k \geq 2$. The algorithm gives optimal solution for submodular potentials. When potentials are non-submodular, we show that important properties such as weak persistency hold for solution inferred by AC. AC is a polynomial time primal dual approximation algorithm for fixed clique size. We show experimentally that AC not only provides significantly better solutions in practice, it is an order of magnitude faster than message passing schemes like Dual Decomposition [19] and GTRWS [17] or Reduction based techniques like [10, 13, 14].

1. Introduction

Approaches for solving 2-label higher order MRF-MAP problems can be broadly categorized into two categories. The ones in the first category reduce any higher order terms in the energy function to an equivalent quadratic pseudo Boolean form [10, 13, 14, 22]. Graph cuts guarantee optimal solution if the reduced form is submodular. An approximate solution, using the roof dual based QPBO algorithm [18], is inferred if the reduced form is nonsubmodular. The algorithms in second category use variants of belief propagation and message passing with or without use of primal dual framework [19, 21, 23, 24].

The QPBO algorithm for optimizing nonsubmodular quadratic pseudo Boolean forms is attractive because it offers weak persistence guarantees for the approximate solution it outputs [12]. Quadratization is a well researched area. Ishikawa's reduction framework [13] encapsulates much of the earlier work and introduces general techniques for handling any higher order system. However, Ishikawa's reduction may add exponential number of terms during the transformation which makes its use problematic for larger clique sizes. Rother et al. [22] show that for pattern based potentials sparsity can be exploited and quadratic form size controlled. Quadratization, however, reduces structure and can introduces non-submodularity [9]. This may affect the number of weakly persistent nodes in the solution inferred by QPBO. A significant new insight is the idea of bisubmodular relaxation as a generalization of roof duality [16]. Kahl and Strandmark [14] have taken the idea further and have developed an LP based algorithm to work with 3-clique and and a subset of 4-clique potentials. Windheuser et al. [25] have extended generalized roof duality to multi-label potentials, giving a method to create submodular relaxation for multi-label 2-clique potential functions.

Dual decomposition and message passing based methods decompose the original problem into smaller problems which can be solved efficiently [20, 23]. Solutions to the independently solved decomposed problems are then combined to provide a consistent solution to the original problem. This cycle is repeated until convergence. The techniques guarantee convergence to optimal if the subproblems can be solved optimally. In practice, the convergence usually comes late and the algorithms are stopped after a few iterations or once the acceptable solution is reached. Komadakis and Paragios [19] have suggested an innovative methodology to extend these techniques to higher order cliques. Since the subproblems in their decomposition themselves can not solved optimally, convergence can not be guaranteed even though the technique gives good inference in practice.

Optimal inference in higher order problems is difficult not only due to issues involved in handling higher order cliques but also due to the non-submodularity of potentials involved. The Generic Cuts (GC) algorithm [9] provides efficient and practical algorithm for optimal solutions to

*Chetan Arora is now with IIIT Delhi.

higher order submodular potential functions and effectively isolates the issues of non-submodularity from that of higher order potentials. GC formulates the problem as flow in a gadget based flow graph and solves a maximum flow problem directly. We show in this paper that the Generic Cuts approach can be extended to model any higher order MRF-MAP problem, submodular or nonsubmodular. It turns out that the gadget introduced here realizes the submodular relaxation of Kahl and Strandmark for higher order potential of arbitrary clique sizes (clique size greater than 4 for the first time). Our experimental results validate the efficiency and quality of results produced by our algorithm. The algorithm has low order polynomial time complexity similar to GC. More importantly solutions given by our algorithm are provably weakly persistent.

Organization of the paper is as follows. Section 2 contains the primal dual underpinnings. Sections 3 and 4 give the framework and algorithmic details respectively for the max flow based solution. Section 5 describes the theoretical properties. Section 6 concentrates on weak persistence properties of the inferred solution. Section 7 contains details of comparative performance.

2. Primal Dual Framework

A MRF-MAP problem with set of pixels $\mathcal{P}$, and a set of cliques $\mathcal{C}$ can be formulated as finding a labeling which minimizes a function of the following form

$$\mathbf{l}_{\mathcal{P}}{}^{*} = \arg\min_{\mathbf{l}_{\mathcal{P}} \in \mathcal{L}^n} \left(\sum_{p \in \mathcal{P}} D_p(l_p) + \lambda \sum_{\mathbf{c} \in \mathcal{C}} W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}) \right), \quad (1)$$

where any pixel p can take a label l_p from the set $\mathcal{L} = \{a, b\}$. $D_p(l_p)$, called the unary potential, is the cost of assigning label l_p to p. $W_{\mathbf{c}} : \mathcal{L}^k \rightarrow \mathcal{R}$, called the clique potential function, assigns a cost for any labeling configuration $\mathbf{l}_{\mathbf{c}}$ on clique $\mathbf{c}$. We denote the number of pixels by n, and the size of a clique by k. $\mathbf{l}_{\mathbf{c},\mathbf{p},\mathbf{l}}$ denotes a labeling configuration on clique $\mathbf{c}$ in which the label of pixel p is l. Note that there can be many such labelings (corresponding to the same clique or the other cliques containing p) and the set of all such labelings is denoted as $\{\mathbf{l}_{\mathbf{c}}\}_{p,l}$.

X_p^l is a variable whose value is 1 whenever pixel p is assigned label l and is 0 otherwise. Similarly $Y_{\mathbf{c}}^{\mathbf{l_c}}$ takes value 1 whenever clique $\mathbf{c}$ is assigned labeling configuration $\mathbf{l}_{\mathbf{c}}$ and is 0 otherwise. With this notation, the MRF-MAP equation (1) can be written as the following integer program:

$$\min_{X_p^l, Y_{\mathbf{c}}^{\mathbf{l_c}}} \sum_{p \in \mathcal{P}} \sum_{l \in \mathcal{L}} D_p(l) X_p^l + \sum_{\mathbf{c} \in \mathcal{C}} \sum_{\mathbf{l}_{\mathbf{c}} \in \mathcal{L}^k} W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}) Y_{\mathbf{c}}^{\mathbf{l_c}} \quad (2)$$

subject to

$$\sum_{l \in \mathcal{L}} X_p^l = 1, \qquad p \in \mathcal{P}, \quad (3)$$

$$\sum_{z \in \{\mathbf{l}_{\mathbf{c}}\}_{p,l}} Y_{\mathbf{c}}^{z} = X_p^l, \qquad p \in \mathcal{P},\ l \in \mathcal{L}, \quad (4)$$

$$X_p^l \in \{0,1\} \quad , \quad Y_{\mathbf{c}}^{\mathbf{l_c}} \in \{0,1\}. \quad (5)$$

Equation (3) ensures that each pixel is assigned exactly one label while equation (4) enforces consistency between pixel and clique labelings. Replacing (5) by (6) we get the relaxed LP formulation of the optimization problem.

$$X_p^l \geq 0 \quad , \quad Y_{\mathbf{c}}^{\mathbf{l_c}} \geq 0. \quad (6)$$

The Lagrangian dual of the relaxed LP can be written as:

$$\max_{U} \sum_{p \in \mathcal{P}} U_p \quad (7)$$

subject to

$$U_p \leq h_p^l, \qquad p \in \mathcal{P},\ l \in \mathcal{L}, \quad (8)$$

where

$$h_p^l = D_p(l) + \sum_{\mathbf{c}: p \in \mathbf{c}} V_{\mathbf{c},p,l}, \quad (9)$$

and

$$\sum_{p \in \mathbf{c}} V_{\mathbf{c},p,\mathbf{l}_{\mathbf{c}}^p} \leq W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}), \qquad \mathbf{c} \in \mathcal{C},\ \mathbf{l}_{\mathbf{c}} \in \mathcal{L}^k. \quad (10)$$

$\mathbf{l}_{\mathbf{c}}^p$ denotes the label of pixel p in labeling $\mathbf{l}_{\mathbf{c}}$. Note that $\mathbf{l}_{\mathbf{c}}^p \in \mathcal{L}$. Complimentary slackness conditions can be written as:

$$X_p^l > 0 \quad \Rightarrow \quad U_p = h_p^l, \quad (11)$$

and

$$Y_{\mathbf{c}}^{\mathbf{l_c}} > 0 \quad \Rightarrow \quad \sum_{p \in \mathbf{c}} V_{\mathbf{c},p,\mathbf{l}_{\mathbf{c}}^p} = W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}). \quad (12)$$

Primal dual framework guarantees that any feasible primal and dual solutions which satisfy all complimentary slackness conditions (11,12) are optimal.

The primal dual framework developed here is very similar to the one used in GC [9]. We build our discussion taking as given concepts like gadget, conjugate edges, flow constraints, residual capacity, tight constraints, union and intersection of constraints, augmenting path as developed in GC and refer the reader to [9] for their details. We focus only on those issues here which are required for developing the new algorithmic framework.

3. Approximate Cuts (AC) Framework

The gadget to model flows for nonsubmodular clique potential is a generalization of the one introduced in [9]. It consists of two copies of the gadget of GC connected as

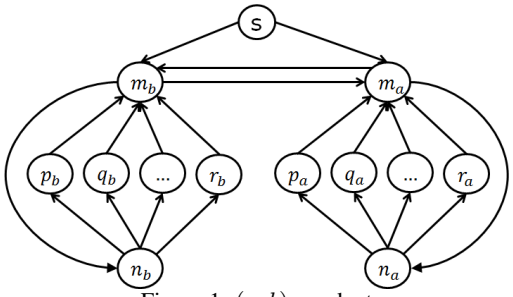


Figure 1: (a, b) - gadget

shown in Figure 1. Formally, for a clique of size k, The gadget consists of $2k + 4$ nodes. $2k$ nodes $p_a, q_a, ..., r_a$ and $p_b, q_b, ..., r_b$ correspond to labels a and b respectively of the k pixels and are called *a-pixel* and *b-pixel* nodes. There are four *auxiliary* nodes n_a, m_a, n_b and m_b. From n_a and n_b, there are directed edges to each a-pixel and b-pixel node respectively. Similarly there are directed edges from a-pixel and b-pixel nodes to m_a and m_b respectively. These edges are called conjugate edges. The two conjugate edges corresponding to a pixel node is referred to as conjugate edge pair. From m_a and m_b there are directed edges to n_a and n_b respectively. Also, there are edges in both directions between m_a and m_b. These edges are called auxiliary edges. The two copies of the gadget of GC are called a-gadget and b-gadget and together the structure is called (a, b)-gadget. For notational convenience we will use m, n, p (without the subscript) whenever the subscript a or b is obvious from the context.

Flow originating from node p_a to node q_a in the a-gadget is routed along the path $p_a \rightarrow m_a \rightarrow n_a \rightarrow q_a$. Similarly flow from node p_a to node q_b in a clique will be routed on path $p_a \rightarrow m_a \rightarrow m_b \rightarrow n_b \rightarrow q_b$. We refer to such a path between two pixel nodes as a *path fragment*. The node p_a and the conjugate edge corresponding to it is referred to as the *sending* node/edge. Node q_a is referred to as the *receiving* node. We denote flow in a conjugate edge, say $n_a \rightarrow p_a$, in the gadget corresponding to clique $\mathbf{c}$ by $f^{\mathbf{c}}_{n_a p_a}$. The relationship between a dual variable and flow in conjugate edge pairs is given by

$$V_{\mathbf{c},p,a} = f^{\mathbf{c}}_{n_a p_a} - f^{\mathbf{c}}_{p_a m_a} \text{ and } V_{\mathbf{c},p,b} = f^{\mathbf{c}}_{n_b p_b} - f^{\mathbf{c}}_{p_b m_b}. \quad (13)$$

We call the r.h.s. of equation (13) as effective flow in a conjugate edge pair. Using (13) dual feasibility constraints (equation 10) can be written as

$$\sum_{p \in \mathbf{c}: \mathbf{l}^p_{\mathbf{c}} = a} (f^{\mathbf{c}}_{n_a p_a} - f^{\mathbf{c}}_{p_a m_a}) + \sum_{p \in \mathbf{c}: \mathbf{l}^p_{\mathbf{c}} = b} (f^{\mathbf{c}}_{n_b p_b} - f^{\mathbf{c}}_{p_b m_b}) \leq W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}). \quad (14)$$

Equation (14) essentially says that the sum of effective flow in all the conjugate edge pairs corresponding to an $\mathbf{l}_{\mathbf{c}}$ can not exceed the value of $W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}})$. We refer to a constraint of the form (14) as a *Dual Feasibility Constraint* (DFC). Note that there is a DFC corresponding to every labeling $\mathbf{l}_{\mathbf{c}}$. The difference between the r.h.s. and l.h.s. of equation (14) is called the *slack* of the DFC. All the conjugate edge pairs in the l.h.s. of the equation and the pixel nodes corresponding to those pairs are said to be *contained/covered* by the DFC. Conversely all such edge pairs and pixel nodes are said to be *participating* in the DFC. Note that for a clique of size k there are *exactly* k conjugate edge pairs participating in any DFC. While in GC the participating conjugate edge pairs correspond to label a, in AC conjugate edge pairs corresponding to both labels a and b may participate with the provision that for a pixel only one of the two conjugate edge pairs participates.

We have, in effect, embedded a k-ary potential system in the $2.k$-ary framework of an (a, b)-gadget. One may look upon the 2^k DFCs corresponding to the k-ary system as the subset of valid DFCs from the 2^{2k} possible DFCs for an (a, b)-gadget. That the $2^{2k} - 2^k$ invalid DFCs play no role is ensured by assuming that the slack corresponding to them is ∞. As in GC, the slack of a DFC is the allowed capacity of each conjugate edge pair covered by it. Similarly *residual capacity* of a conjugate edge pair is the minimum of the slacks of *all* the DFCs in which it participates. Note that unlike GC where only slacks of DFCs corresponding to non uniform labelings contribute to the calculation of residual capacity of a conjugate edge pair, in AC all DFCs contribute to the calculation of residual capacity. In GC initial reparametrization ensures that both uniform labeling costs are zero and the gadget used in GC does not need to account for those costs. In AC this may not be so. We call a DFC *tight* if it's slack is zero. All conjugate edge pairs in a tight DFC are termed tight under that DFC. Note that the slack in a DFC should always be greater than or equal to zero. A DFC is called *violated* if its slack becomes negative.

4. Flow Graph Construction and Algorithm

The flow graph based on (a, b)-gadgets has two nodes corresponding to labels a and b for each pixel. For each clique the nodes corresponding to labels a and b get connected in a (a, b)-gadget using four auxiliary nodes introduced specifically for the clique. There are two additional nodes s and t (referred to as *terminal* nodes) which are connected to the gadget nodes as follows: Let the unary/singleton potential for assigning label a and b at pixel p be $D_p(a)$ and $D_p(b)$ respectively. If $D_p(a) > D_p(b)$ then node p_a in the flow graph has a directed edge from s and there is a directed edge from p_b to t. On the other hand, if $D_p(b) > D_p(a)$ then the directed edge from s is to p_b and the directed edge to t is from p_a. We refer to these edges as *terminal edges*. Capacity for the pair of such terminal edges introduced for a pixel p is defined to ensure such that sum of flows in the two edges cannot exceed $|D_p(b) - D_p(a)|$. Node s also has directed edges to all auxiliary nodes m_a and m_b of all the gadgets in the flow graph whose capacities are

constrained as follows:

Conservation of flow at node m_b of a clique $\mathbf{c}$ implies

$$f_{sm_b} + f^{\mathbf{c}}_{m_a m_b} + \sum_{p \in \mathbf{c}} f^{\mathbf{c}}_{p_b m_b} = f^{\mathbf{c}}_{m_b m_a} + f^{\mathbf{c}}_{m_b n_b},$$

$$\Rightarrow \quad f_{sm_b} + f^{\mathbf{c}}_{m_a m_b} + \sum_{p \in \mathbf{c}} f^{\mathbf{c}}_{p_b m_b} = f^{\mathbf{c}}_{m_b m_a} + \sum_{p \in \mathbf{c}} f^{\mathbf{c}}_{n_b p_b},$$

$$\Rightarrow \quad f_{sm_b} + f^{\mathbf{c}}_{m_a m_b} - f^{\mathbf{c}}_{m_b m_a} = \sum_{p \in \mathbf{c}} (f^{\mathbf{c}}_{n_b p_b} - f^{\mathbf{c}}_{p_b m_b}).$$

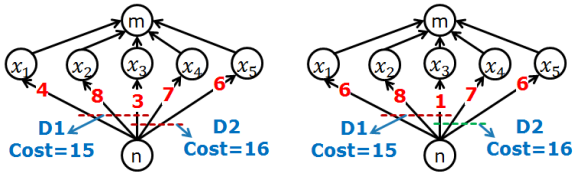
Note that r.h.s. of the above equation is bounded from above by the cost of uniformly labeling all nodes of clique $\mathbf{c}$ by b's. Since edges of type $m_a \rightarrow m_b$ and $m_b \rightarrow m_a$ are between the same pair of nodes, one can restrict, without loss of generality, that flow at any time is in only one of the two edges. Under this assumption we can say that the capacity constraint on edges $s \rightarrow m_b$ and $m_a \rightarrow m_b$ is that the sum of flow in them can not exceed the cost of uniformly labeling all nodes of clique $\mathbf{c}$ by b's, i.e., the clique potential for assigning label b to all nodes of a clique. Similarly the sum of flow in edges $m_b \rightarrow m_a$ and $s \rightarrow m_a$ should be less than or equal to the clique potential for assigning label a to all nodes of the clique. The capacities of edges $m_a \rightarrow n_a$ and $m_b \rightarrow n_b$ are set to infinity.

Algorithm 1 AC Maxflow Algorithm

1: **for** All cliques $\mathbf{c} \in \mathcal{C}$ **do**
2: Reparametrize the clique potential until the DFCs for uniform labeling become tight or all V variables become tight under some DFC;
3: **end for**
4: Build the residual graph R;
5: **while** There exists an s-t augmenting path in R find the lexicographically shortest augmenting path; **do**
6: Augment flow in that path;
7: Build the residual graph R;
8: **end while**

The generic algorithm for finding maximum flow in a flow graph based on (a, b)-gadgets is the traditional augmenting path method used in GC and is as given in Algorithm 1.

In the first step, we reparametrize clique potentials using the technique suggested in GC, with the objective of making uniform labeling costs zero if possible. In GC, when the clique potential function is submodular, uniform labeling costs are guaranteed to become zero. When clique potentials are non-submodular even if uniform costs do not become zero, the state arrived at results in a conjugate edge pair to be tight in at least one of the DFCs containing it. This makes a-pixel and b-pixel nodes in a gadget unreachable[1] from s via a path containing s to an auxiliary node edge. Note that flow can be augmented using any heuristic used in max flow algorithms. The augmenting path framework of steps 4 to 7 could even be replaced by the Push Relabel technique of [11].



(a) Before Redistribution (b) After Redistribution

Figure 2: Flow redistribution in a gadget

As in GC, the primary difficulty in pushing flow in a residual graph is due to the fact that as long as the sum of flows in the collection of conjugate edge pairs contained in a DFC remains the same, flows in the conjugate edge pairs involved can take any value without effecting the tightness of the DFC [9]. This allows for redistribution of flow in some edges of the flow graph to enable flow pushing in some other edges which were not involved in this redistribution. This can happen when two tight DFCs share a conjugate edge. For example, consider Figure 2 which shows two states of flow in a gadget of a flow graph.

Flow in each conjugate edge incident at a pixel node is shown in red and the cost of covering a DFC is shown in blue. In Figure 2(a) all conjugate edges incident at pixel nodes have zero residual capacity because the two DFCs, $D1$ covering conjugate edges incident at x_1, x_2, x_3, and $D2$ covering edges incident at x_3, x_4, x_5 are tight. Suppose it is possible to redistribute flow in the flow graph by increasing flow in the conjugate edge incident at x_1 to 6 and decreasing flow in conjugate edge incident at x_3 to 1. Total flow out of s remains the same but DFC $D2$ now has slack of 2 (see Figure 2(b)). Conjugate edges incident at x_4 and x_5 in this flow configuration may now have residual capacity of 2 (assuming other DFCs covering them have slack of 2 or higher) and there may now be a s to t path in the residual graph passing through these edges. When potentials are submodular this cannot take place because the DFC which represents the union of the two tight DFCs ($D1$ and $D2$) is per force tight. In effect no advantage by redistribution can be enabled for any conjugate edge involved as at least one of the DFCs which contributes to the computation of their residual capacities continues to remain tight.

Our heuristic is to attempt no redistribution when it is discovered that two tight DFCs are sharing edges. We declare the DFCs corresponding to union and intersection of two tight DFCs as tight at that stage. This can be seen as working with submodular approximations of the original

[1] A node p is called reachable from q in a flow graph if q can push flow to p without violating any DFC

Algorithm 2 Residual Graph Construction

for All cliques **c** **do**
  for All DFCs of **c** **do**
    Calculate the slack in presence of current flow;
    Declare a DFC as tight if it's slack is zero or if it is a union or intersection of two tight DFCs;
    Set the residual capacity of a conjugate edge as the minimum of slacks of all DFCs in which it participates;
  end for
  for All path fragments of type p_a to q_a, p_b to q_b, p_a to q_b, p_b to q_a **do**
    if Edge corresponding to q have have non zero residual capacity **then**
      Add the path fragment to the residual graph;
      Set the residual capacity of the path fragment as the residual capacity of receiving edge;
    else
      if The intersection of the tight DFCs covering receiving edge also cover sending edge **then**
        Add the path fragment to the residual graph;
        Set the residual capacity of the added fragment as minimum slack of all DFCs which cover receiving edge and not sending edge;
      end if
    end if
  end for
end for

potentials. The skeletal residual graph building algorithm, which reflects this heuristic, is given in Algorithm 2.

5. Theoretical Properties

Following the same proof methodology as for GC it can be shown that:

Lemma 5.1. *Flow in the two-gadget based flow graph cannot exceed the capacity of any (S, T) cut.*

Lemma 5.2. *If $W_{\mathbf{c}}(\cdot)$ is submodular, then for every two tight DFCs corresponding to labeling $\mathbf{l}_{\mathbf{c}}^{'}$ and $\mathbf{l}_{\mathbf{c}}^{''}$, the DFCs corresponding to labeling $\mathbf{l}_{\mathbf{c}}^{'} \cup \mathbf{l}_{\mathbf{c}}^{''}$ and $\mathbf{l}_{\mathbf{c}}^{'} \cap \mathbf{l}_{\mathbf{c}}^{''}$ are also tight.*

Theorem 5.3. *In the flow graphs corresponding to the dual optimization problem max flow is equal to min cut under the assumption that clique potential functions are submodular.*

We omit the proof of these lemmas as they follow directly from those given in [9]. Lemma 5.4 describe an important property satisfied by (a, b)-gadget based flow graphs. The proof of the lemma is included in the supplementary material.

Lemma 5.4. *In an (a, b)-gadget based residual flow graph created after flow augmentation process has terminated both p_a and p_b corresponding to a pixel p can not be reachable from s. In other words either one of p_a and p_b is reachable or both are unreachable from s.*

Lemma 5.4 shows that once the stage when flow cannot be augmented any further has been reached a strategy for labeling nodes can be developed that is consistent with the one developed for GC. This is because definition of set $\mathbf{S}$ as the set of all nodes reachable from s in the residual graph arrived at after the flow augmentation stage is over does not lead to any inconsistency. Lemma 5.4 ensures that only one of the nodes corresponding to a pixel will be reachable from s. We follow standard procedure to create set $\mathbf{S}$ and $\mathbf{T}$, but force all nodes p_b to be in $\mathbf{S}$, for which corresponding node p_a is not in $\mathbf{S}$ (this is to avoid the case when both p_a and p_b are not in $\mathbf{S}$). We label a pixel p to be a if p_b is in $\mathbf{S}$ and as b if p_a is in $\mathbf{S}$. The labeling strategy provides a label to all nodes.

6. Weak Persistence

For the purpose of discussion in this section we will assume that original energy function has been reparametrized such that singleton/unary costs have been absorbed in the higher order terms. Note that this reparametrization does not change the submodularity of the potential function. $W_c(\cdot)$ refers to such reparametrized function in this section. The overall energy function, referred to as $W(\cdot)$, is simply the summation of such $W_c(\cdot)$ over all cliques.

As discussed in Section 3, an (a, b)-gadget for a clique $\mathbf{c}$ can be looked upon as embedding a k-ary potential system in the $2.k$-ary framework of an (a, b)-gadget. One may look upon the 2^k DFCs corresponding to the k-ary system as the subset of valid DFCs from the 2^{2k} possible DFCs for an (a, b)-gadget. That the $2^{2k} - 2^k$ invalid DFCs play no role is ensured by assuming that the slack corresponding to them is ∞. Equivalently an (a, b)-gadget can also be looked upon as operating over a function $g_c(\cdot)$ of arity $2.k$ in which the first k Boolean attributes correspond to labeling of pixels of

the clique by a and the remaining k correspond to labeling of the pixels by b. Since the states associated with labels a and b are essentially complements of each other, the relationship between the k-ary function $W_c(\cdot)$ corresponding to the valid DFCs of clique c and $g_c(\cdot)$ is given by

$$W_{\mathbf{c}}(x) = g_{\mathbf{c}}(x, \bar{x}), \tag{15}$$

where x is any labeling on the pixels in $\mathbf{c}$. In line with the discussion above $g_c(x, y) = \infty$ whenever $x \neq \bar{y}$. The overall function, $g(\cdot)$ spanning all cliques is simply the summation of $g_{\mathbf{c}}(\cdot)$ over all cliques $\mathbf{c}$. If $W(\cdot)$ is the result of summation of all $W_{\mathbf{c}}(\cdot)$, then similar relationship holds between $W(\cdot)$ and $g(\cdot)$ as defined in (15). Note that $W(\cdot)$ and $g(\cdot)$ are n-ary and $2.n$-ary functions respectively (n is the number of pixels). We will now show that AC can be looked upon as starting with a nonsubmodular relaxation $g(\cdot)$ satisfying equation (15) and computing a submodular relaxation $g^*(\cdot)$ satisfying equation (15) and its minimizer. This is very much along the lines of Kahl and Strandmark [14] to compute the *generalized roof dual* approximation of a nonsubmodular function. The function $g^*(\cdot)$ is the sum of submodular functions $g^*_{\mathbf{c}}(\cdot)$ for all $\mathbf{c}$ which get defined during the execution of AC as follows.

The approximation heuristic in AC involves making the DFCs corresponding to union and intersection of two tight DFCs as tight in an (a, b)-gadget (corresponding to some clique c). Let the states corresponding to the two tight DFCs be $(x, \bar{x})$ and $(y, \bar{y})$. Declaring the DFCs corresponding to union and intersection tight is equivalent to defining the function $g^*_c(\cdot)$ for states $(x \cup y, \bar{x} \cup \bar{y})$, and $(x \cap y, \bar{x} \cap \bar{y})$. We set $g^*_c(x \cup y, \bar{x} \cup \bar{y})$ to *(flow in the conjugate edges corresponding to union of the two DFCs)* and $g^*_c(x \cap y, \bar{x} \cap \bar{y})$ to *(flow in the conjugate edges corresponding to intersection of the two DFCs)*. Note that setting the value of $g^*_c(\cdot)$ in this way does not cause any violation of residual capacity constraints of the conjugate edges of the gadget. What is important is that *invalid* DFCs (see Section 3 for definition of *invalid*) which have to be tight have been identified and their values set (to some value less than the original ∞). When AC terminates then for those $2^{2.k} - 2^k$ states for which $g^*_c(\cdot)$ has not been explicitly set during the execution of AC, we set the value of $g^*_c(\cdot)$ to the maximum flow possible in the conjugate edges corresponding to the states without violating any dual feasibility constraint. It can be verified that $g^*_c(\cdot)$ (as well as the overall function $g^*(\cdot)$) defined in this way is submodular. The flow computed by AC is equal to max flow in the flow graph for $g^*(\cdot)$ so defined. That is, we have minimized $g^*(\cdot)$. It should be noted that functions $g^*_c(\cdot)$ need not be explicitly calculated. AC essentially computes the minimum of a submodular function $g^*(\cdot)$ which is a sum of functions of type $g^*_c(\cdot)$, each of which satisfies equation (15).

Using a formulation similar to Windheuser et al. [25], we show that the solution outputted by AC is weakly persistent. Establishing weak persistence requires showing that there are parts in the AC's solution which will occur in a minimiser of function $g(\cdot)$. What is interesting is that we do not require $g^*(\cdot)$ to be symmetric.

Lemma 6.1. *For the $2.k$-ary submodular function $g^*_c(\cdot)$ defined during the execution of AC and for all k-ary states x, y of a and b-gadgets of a clique c the following is true:*

$$g^*_c(x, y) \geq g^*_c(x \cap \bar{y}, \bar{x} \cap y).$$

Proof of Lemma 6.1 is in the supplementary material. Since overall function $g^*(\cdot)$ is simply the summation of $g^*_c(\cdot)$, the lemma applies to $g^*(\cdot)$ as well.

Lemma 6.2. *Let x^*, y^* be a minimizer of function $g^*(\cdot)$ outputted by AC (state corresponding to the $(\mathbf{S}, \mathbf{T})$ cut outputted by AC). For any state x, y*

$$g^*(x, y) \geq g^*\left((x \cup x^*) \cap \bar{y} \cap \bar{y}^*, (y \cup y^*) \cap \bar{x} \cap \bar{x}^*\right).$$

The Lemma 6.2 is identical to Lemma 6 in [25]. Its proof relies on Lemma 6.1 which we have proved here for AC. We refer the reader to [25] for the proof of Lemma 6.2.

Simplifying along the lines of [25], and using Lemma 6.2 we can show that $W(x) \geq g^*(x \cap \bar{y}^*, \bar{x} \cap \bar{x}^*)$, where x^*, y^* is the minimizer of $g^*(\cdot)$ outputted by AC. Similarly it can also be shown that $W(x) \geq g^*((x \cup x^*) \cap \bar{y}^*, (\bar{x} \cup y^*) \cap \bar{x}^*)$. Setting the overwrite similar to [25],

$$z = \bar{y}^* \cap (x^* \cup x), \text{ and } \bar{z} = \bar{x}^* \cap (y^* \cup \bar{x}),$$

we can be show that, $W(x) \geq g^*(z, \bar{z})$. Note that for any x the overwrite operator will output for a pixel p the same label as given by AC, as long as p_a and p_b are on different sides of the $(\mathbf{S}, \mathbf{T})$ cut as determined by AC. That is, the label of a pixel p inferred by AC is weakly persistent.

7. Experiments

Our experiments have focussed on time taken and errors in labeling observed using the algorithm reported here (referred to as AC) in comparison to message passing methods like Max Product Inference (MPI) [15], Dual Decomposition (DD) [19] and GTRWS [17] as well as reduction techniques proposed by Ishikawa (IQ) [13], Fix et al. (FZ) [10] and the Generalized Roof Duality based algorithm (KS) of [14]. The implementations of GTRWS, IQ, FZ and KS have been taken from [8], [2, 7], [3] and [6] respectively. The rest are from Darwin framework [1]. All experiments are conducted with default parameters in their respective implementations and no parameter tuning of any sort has been attempted for any algorithm.

Our first experiment has focussed on performance when clique potentials are submodular. We take denoising problem as used by GC [9]. The clique potential used penalizes

Input | AC(0.01s) 673535 673535 | IQ(0.4s) 775052 | FZ(0.2s) 775052 | DD(6.9s) 797675 | GTRWS(0.22s) 673535 673535 | MPI(0.2s) 824715 | KS(210.8s) 709283 668276

Figure 3: Denoising: σ of Gaussian noise added is 70. Time taken in seconds is in brackets, followed by primal and dual energies.

GT | AC (20.5%,2.1) | DD (40.9%,4457) | GTRWS (20.7%,17.1) | FZ (62.1%,12.7) | IQ (61.7%,16.5) | MPI (72%,0.38)

Figure 4: Stereo disparity results on Tsukuba dataset. Percentage error from ground truth followed by time taken in seconds is in parenthesis.

as per square root of number of edges in the labeling. This clique potential is submodular. Figure 3 shows the results. For IQ and FZ dual values are not reported as their code is not written to make dual values available. Unlabeled nodes are uniformly assigned label 0 in all methods. For AC and GTRWS primal and dual energies are equal. These results are optimal. While the unlabeled nodes have been shown as grey for energy calculation they have been uniformly labelled 0 in all methods. There is some work reported where the focus is to label such nodes "intelligently" rather than just label them uniformly 0 or 1. Grey in Fig. 3 only emphasizes the extent of pixels where intelligent probing is required. Note that while both KS and AC work with submodular relaxations which embed the original problem in cliques of larger order, the algorithmic technique proposed in KS is to reduce the computed submodular relaxations to second order systems and then use QPBO. The submodular approximation used in KS, therefore, may output non optimal labelings.

Working with nonsubmodular potentials, we first consider the problem of finding disparity from a pair of stereo images. Data energy terms along with Tsukuba images are as used by MRF code at the Middlebury test bench [5]. Cliques are of size 2×2 and image size is 130×130. We have run traditional α-expansion with clique potential which penalizes as per the square root of the number of edges in the multi-label labeling. In multi-label scenario this potential function need not be submodular. Also, while some of these methods under consideration can directly handle multi label inference problems we have run them in the α-expansion mode primarily to keep the experimental set up same in all cases. Error percentage is calculated by taking the difference of the labeling outputted from the ground truth image. Figure 4 shows comparative performance. Labels are not switched in α-expansion when AF and IQ give unlabeled nodes as output. All methods other than AC and GTRWS fail to move much from the initial solution (generated using unary potential).

We have chosen deblurring/deconvolution of binary images as the next experiment. We have taken binary images from [4], blurred them using 3×3 averaging kernel and added Gaussian noise with zero mean and different values of σ (values given below each input figure). The images are restored with the potential being distance of average value in the 3×3 windows in the proposed labeling from the observation corresponding to the center of the window. Unary cost is the absolute difference of current pixel intensity from the ideal $\{0, 255\}$ values. Figure 5 reports outputs obtained from various methods. Error in IQ's output is the largest as most of the nodes have been left unlabeled (grey nodes). Error in AC's output is the least. Image size is 3000 pixels. This is the largest size at which all the algorithms ran to completion.

Since KS can not handle cliques of size larger than 4 we re-ran the de-convolution experiment with deblurring using 2×2 blur kernel. Figure 6 shows the results. While the speedup of AC over FZ is around $5X$ for clique size 4, it shoots up to $40X - 50X$ for clique size 9. Similar results have been reported by GC as well [9].

Note that while quality of the output produced by GTRWS is competitive with AC in all experiments it is an order or more slower than AC.

8. Conclusion

AC is the first direct combinatorial method for not only solving higher order MRF-MAP problems with non-submodular clique potentials but also outputting weakly

Figure 5: Non Blind Deconvolution (9 clique). Percentage error from ground truth followed by time taken in seconds is in parenthesis.

Figure 6: Non Blind Deconvolution (4 Clique). Percentage error from ground truth followed by time taken in seconds is in parenthesis.

persistent solutions. AC's gadget structure captures generalized roof duality in a natural way. Since no additional auxiliary nodes are introduced, it is expected that the structure of the labeling problem can be exploited by other application specific heuristics to improve the labeling solution in AC. Modeling using gadgets allows visualizing source of approximation (union or intersection constraint of two tight constraints assumed to be tight) and can be an additional way to develop new heuristics for better empirical labeling solutions in future.

Acknowledgement: Chetan Arora is supported by post-doctoral fellowship of the Council for Higher Education in Israel.

References

[1] Darwin Framework, Version 1.1.2. http://drwn.anu.edu.au/.

[2] Higher-Order Clique Reduction software Version 1.02. www.f.waseda.jp/hfs/software.html.

[3] Higher order energy reduction. www.cs.cornell.edu/~afix/.

[4] MPEG7 CE Shape-1 Part B. http://www.imageprocessingplace.com.

[5] MRF energy minimization software - Version 2.1. http://vision.middlebury.edu/MRF/code/.

[6] Pseudo-Boolean Optimization. www.maths.lth.se/matematiklth/personal/petter/pseudoboolean.php.

[7] QPBO Version 1.3. http://pub.ist.ac.at/~vnk/software.html.

[8] SRMP- Version 1.01. http://pub.ist.ac.at/~vnk/software.html#SRMP.

[9] C. Arora, S. Banerjee, P. Kalra, and S. Maheshwari. Generic Cuts: An efficient optimal algorithm for submodular MRF-MAP problems with higher order cliques. In *ECCV*, 2012.

[10] A. Fix, A. Gruber, E. Boros, and R. Zabih. A graph cut algorithm for higher-order markov random fields. In *ICCV*, 2011.

[11] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. In *STOC*, 1986.

[12] P. Hammer, P. Hansen, and B. Simeone. Roof duality, complementation and persistency in quadratic 0-1 optimization. *Math. Programming*, 1984.

[13] H. Ishikawa. Transformation of General Binary MRF Minimization to the First-Order Case. *TPAMI*, 2011.

[14] F. Kahl and P. Strandmark. Generalized roof duality for pseudo-boolean optimization. In *ICCV*, 2011.

[15] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. Adaptive Computation and Machine Learning. MIT Press, 2009.

[16] V. Kolmogorov. Generalized roof duality and bisubmodular functions. *Discrete Applied Mathematics*, 2012.

[17] V. Kolmogorov. Reweighted message passing revisited. *CoRR*, abs/1309.5655, 2013.

[18] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts-a review. *TPAMI*, 2007.

[19] N. Komodakis and N. Paragios. Beyond pairwise energies: Efficient optimization for higher-order MRFs. In *CVPR*, 2009.

[20] N. Komodakis, N. Paragios, and G. Tziritas. MRF Energy Minimization and Beyond via Dual Decomposition. *TPAMI*, 2011.

[21] X. Lan, S. Roth, D. P. Huttenlocher, and M. J. Black. Efficient belief propagation with learned higher-order markov random fields. In *ECCV*, 2006.

[22] C. Rother, P. Kohli, W. Feng, and J. Jia. Minimizing sparse higher order energy functions of discrete variables. In *CVPR*, 2009.

[23] D. Sontag, T. Meltzer, A. Globerson, Y. Weiss, and T. Jaakkola. Tightening LP relaxations for MAP using message-passing. In *UAI*, 2008.

[24] D. Tarlow, I. E. Givoni, and R. S. Zemel. HOP-MAP: Efficient Message Passing with High Order Potentials. *JMLR*, 2010.

[25] T. Windheuser, H. Ishikawa, and D. Cremers. Generalized roof duality for multi-label optimization: optimal lower bounds and persistency. In *ECCV*, 2012.