Generic Cuts: An Efficient Algorithm for Optimal Inference in Higher Order MRF-MAP

Chetan Arora, Subhashis Banerjee, Prem Kalra, and S.N. Maheshwari

Indian Institute of Technology Delhi
http://www.cse.iitd.ac.in/~chetan

Abstract. We propose a new algorithm called Generic Cuts for computing optimal solutions to 2 label MRF-MAP problems with higher order clique potentials satisfying submodularity. The algorithm runs in time $O(2^k n^3)$ in the worst case (k is clique order and n is the number of pixels). A special gadget is introduced to model flows in a high order clique and a technique for building a flow graph is specified. Based on the primal dual structure of the optimization problem the notions of capacity of an edge and cut are generalized to define a flow problem. We show that in this flow graph max flow is equal to min cut which also is the optimal solution to the problem when potentials are submodular. This is in contrast to all prevalent techniques of optimizing Boolean energy functions involving higher order potentials including those based on reductions to quadratic potential functions which provide only approximate solutions even for submodular functions. We show experimentally that our implementation of the Generic Cuts algorithm is more than an order of magnitude faster than all algorithms including reduction based whose outputs on submodular potentials are near optimal.

Keywords: Higher Order MRF-MAP, Submodular Function Minimization, Optimal Algorithm.

1 Introduction

Many computer vision problems such as image segmentation, stereo matching, image restoration can be naturally modeled as labeling problems. Formulating such labeling problems as MRF-MAP problems converts them to a discrete optimization problem. Many optimization schemes based upon combinatorial optimization or linear programming [1–3] have been proposed for solving these problems, however most of the methods can handle unary and pairwise potentials only. It has been adequately shown by various authors [4–9] that encoding various structural and complex dependencies between pixels using higher order clique greatly improves the solution quality.

Techniques proposed for optimizing higher order potentials have ranged from Iterated Condition Modes [10] to Message Passing [11], to Dual Decomposition [9]. All output approximate solutions with slow convergence rates. Reduction of higher order problems to quadratic forms and then solve them using standard optimization techniques of graph cut if the resulting function is submodular, or use the

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part V, LNCS 7576, pp. 17–30, 2012.

QPBO [12] algorithm which is applicable for both non-submodular and submodular functions has recently attracted considerable attention [4, 7, 13–16]. Reduction is attractive as the higher order problem is reduced to the case where potentials are at most pairwise for which direct network flow based algorithms output optimal solutions when potentials are submodular. Reduction based approaches and algorithms derived from them, however, suffer from some drawbacks:

- It is known that a quadratic pseudo Boolean function can be optimized in polynomial time if it satisfies the submodularity constraint [16, 17]. However, not only reduction techniques do not preserve submodularity, not all higher order submodular functions can be reduced to quadratic submodular forms [18].
- Ishikawa [4] has noted that reduction of the higher order polynomial form to quadratic, for a k-clique, adds 2^k auxiliary variables per clique making this approach virtually unusable even for moderate sized cliques.
- QPBO algorithm when applied to reduced quadratic forms leaves many nodes unlabeled. This can happen even if the original pseudo Boolean function is submodular.

There have been attempts at addressing the above drawbacks. Rother et al. [7] exploit sparsity of preferred labelings by creating a submodular deviation function around them for which in some situations the reduction algorithm creates compact but non-submodular quadratic forms. There have also been attempts to take submodular relaxations directly to higher order terms [13, 19]. While for cubic potentials, generalized roof duality approximations can be obtained by solving a series of LP problems [13], for quartic potentials this gets limited to using only a subset of quartic submodular functions as not all submodular quartic polynomials can be reduced to equivalent quadratic form.

There is, therefore, a need to develop direct efficient algorithmic frameworks for optimizing MRF-MAP problems with higher order clique potentials. This is more so because submodular function optimization is known to be a polynomial time problem [20, 21], but the algorithms are effectively not usable (the most efficient polynomial time algorithm takes $O(n^5)$ steps [20]). Efficient direct optimal algorithms for higher order submodular potentials are important they can also be the basis of efficient direct approximation algorithms for MRF MAP optimization with non-submodular higher order clique potentials.

Contribution: We present in this paper an optimal algorithm based on primal dual frame work for two label higher order MRF-MAP problem with submodular costs. As in the two clique version of the problem in which such a primal can be viewed as a min cut and the dual as a max flow problem, we show that these concepts can be generalized for higher order clique problems. The dual framework has resulted in a novel flow problem in which both the capacity of an edge and the cost of a cut have a new but natural generalizations. We give an algorithm for optimizing two label higher order MRF-MAP problems with submodular costs based on solving a max flow problem in $O(n|\mathcal{C}|^2k^4 2^k)$ steps, where n is the number of pixels, $|\mathcal{C}|$ number of cliques and k the size of a clique. On an image of size 500×500 with clique size 4 the direct algorithm reported

here is 30 times faster than QPBO using Ishikawa's reduction. Also, like the max flow based graph cut optimizer for second order potentials the optimal solution outputted by our algorithm are integral which LP and other non combinatorial optimization theory based algorithms cannot guarantee.

In Section 2 we develop the basic primal dual frame work and the gadget which is used to model a clique in the flow graph created to work with the dual of the problem. Section 2.3 contains the algorithm which we call Generic-Cut. Section 3 reports comparison of performance with Ishikawa's reduction technique [4] using QPBO [12].

2 Primal Dual Schema

We denote the set of pixels in an image by $\mathcal{P}$, and the set of higher order cliques on the pixel set by $\mathcal{C}$. The label of a pixel p is denoted by l_p, the labeling configuration of clique $\mathbf{c}$ by $\mathbf{l_c}$, and $l_{\mathbf{c}}^p$ denotes the label of pixel p in clique $\mathbf{c}$ with labeling configuration $\mathbf{l_c}$. Finding a labeling with maximum *a posteriori* probability (MAP) assuming labeling to be a MRF can be shown to be equivalent to minimizing energy of following kind:

$$E(\mathbf{l_f}) = \sum_{p \in \mathcal{P}} D_p(l_p) + \lambda \sum_{\mathbf{c} \in \mathcal{C}} W_{\mathbf{c}}(\mathbf{l_c})), \tag{1}$$

where $D_p(l_p)$, called the *unary potential*, is the cost of assigning label l_p to p. $W_{\mathbf{c}}(\mathbf{l_c})$, called the *clique potential*, is the penalty/cost of any labeling configuration $\mathbf{l_c}$ on clique $\mathbf{c}$.

The LP formulation for MRF-MAP given below follows Kleinberg and Tardos [2]. Any pixel can take a label from the set $\mathcal{L} = \{a, b\}$ of possible labels, and $\mathbf{l}_{\mathbf{c},p,a}$ is a labeling in the labeling configurations of clique $\mathbf{c}$ in which the label of pixel p is a. We introduce a binary variable X_p^a whose value is 1/0 whenever pixel p is assigned label a/b respectively. Similarly the binary variable $Y_{\mathbf{c}}^{\mathbf{l_c}}$ takes value 1 whenever clique $\mathbf{c}$ is assigned label configuration $\mathbf{l_c}$ and is 0 otherwise. Let $W_{\mathbf{c}} : \mathcal{L}^k \to \mathcal{R}$ be the clique potential function giving the penalty of labeling pixels of clique $\mathbf{c}$ by $\mathbf{l_c}$. The MRF-MAP equation (1) can be equivalently written as the following relaxed linear program:

$$\min_{X_p^a, Y_{\mathbf{c}}^{\mathbf{l_c}}} \sum_{p \in \mathcal{P}} \sum_{a \in \mathcal{L}} C_p^a X_p^a + \sum_{\mathbf{c} \in \mathcal{C}} \sum_{\mathbf{l_c} \in \mathcal{L}^k} W_{\mathbf{c}}(\mathbf{l_c}) Y_{\mathbf{c}}^{\mathbf{l_c}} \tag{2}$$

subject to

$$\sum_{a \in \mathcal{L}} X_p^a = 1, \qquad p \in \mathcal{P},$$

$$\sum_{\forall \mathbf{l}_{\mathbf{c},p,a}} Y_{\mathbf{c}}^{\mathbf{l}_{\mathbf{c},p,a}} = X_p^a, \qquad \mathbf{c} \in \mathcal{C},\ p \in \mathbf{c},\ a \in \mathcal{L},$$

$$X_p^a \geq 0 \qquad , \qquad Y_{\mathbf{c}}^{\mathbf{l_c}} \geq 0.$$

The dual of the above can be shown to be [22]

$$\max_{U,V} \sum_{p \in \mathcal{P}} U_p \tag{3}$$

subject to

$$U_p \leq h_p^a, \qquad p \in \mathcal{P},\ a \in \mathcal{L}, \tag{4}$$

where

$$h_p^a = C_p^a + \sum_{\forall \mathbf{c}\ s.t.\ p \in \mathbf{c}} V_{\mathbf{c},p,a}, \tag{5}$$

and

$$\sum_{p \in \mathbf{c}} V_{\mathbf{c},p,l_{\mathbf{c}}^p} \leq W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}), \qquad \mathbf{c} \in \mathcal{C},\ \mathbf{l}_{\mathbf{c}} \in \mathcal{L}^k. \tag{6}$$

Complimentary slackness conditions can be written as

$$X_p^a > 0 \qquad \Rightarrow \qquad U_p = h_p^a, \tag{7}$$

and

$$Y_{\mathbf{c}}^{\mathbf{l_c}} > 0 \qquad \Rightarrow \qquad \sum_{p \in \mathbf{c}} V_{\mathbf{c},p,l_{\mathbf{c}}^p} = W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}). \tag{8}$$

Assuming the cost of assigning uniform labeling (all a's or all b's) to clique as zero gives us the following constraint

$$\sum_{p \in \mathbf{c}} V_{\mathbf{c},p,a} = 0, \qquad \mathbf{c} \in \mathcal{C},\ a \in \mathcal{L}. \tag{9}$$

This assumption is not restrictive. It can be shown [22] that when costs are submodular, they can be normalized through reparametrization to an equivalent state where uniform labeling costs in a clique are zero.

2.1 Flow Interpretation of Dual

The above primal dual framework is very similar to the one derived by Komadakis and Tziritas [3]. We use their model of ball and wells to motivate flow interpretation of the dual. Corresponding to every pixel there is a well in which balls representing labels a and b float. The ball a in the well representing pixel p is represented by p^a and it floats at height h_p^a in the well. Since the dual is a maximization problem and U_p has to be less than both h_p^a and h_p^b (see equation (4)), it can be set equal to the height of the lower of the two balls in the well. If U_p is equal to, say, h_p^a in well p, complementary slackness condition (7) implies that primal variable X_p^a has to be set to 1 (since if X_p^b is set to 1 and given $h_p^b > h_p^a = U_p$, the complimentary slackness will not be satisfied). That is, in any feasible labeling configuration pixel p is assigned label a. We call the ball a as *active* in well p in such a scenario. In other words any dual optimization

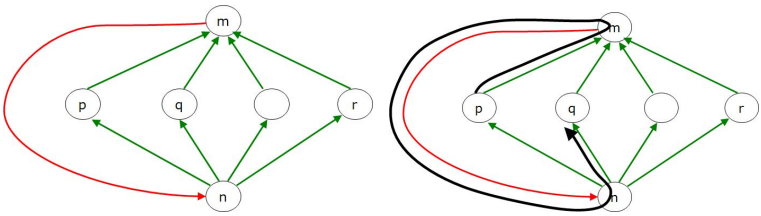
strategy which chooses the lower ball as active keeps equations (4) and (7) satisfied. Since any change in the value of variable $V_{\mathbf{c},p,a}$ impacts the height of ball a in well p and may change the ball with minimum height, therefore equation (4) essentially ensures the view that variables $V_{\mathbf{c},p,a}$ are the only free variables in the dual optimization problem.

Consider a clique $\mathbf{c}_1$ in which the labeling associated with pixels involved satisfies the uniform labeling constraint given in equation (9). Let p and q be two wells/pixels in $\mathbf{c}_1$. Consider the operation on dual variables in which ball p^a decreases its height by reducing variable $V_{\mathbf{c}_1,p,a}$ by δ without affecting the relative ordering of balls in well p. To continue satisfying the constraint given in equation (9) will require increase in the value(s) of other V variables associated with clique $\mathbf{c}_1$. One possibility is to increase variable $V_{\mathbf{c}_1,q,a}$ of q^a by the same amount. In effect the height of p^a decreases and that of q^a increases. We may view this change in heights of p^a and q^a as a consequence of sending δ flow from p^a to q^a in clique $\mathbf{c}_1$. Note that while decreasing the value of variable $V_{\mathbf{c}_1,p,a}$ fixes the well of flow origin in clique $\mathbf{c}_1$ the destination well could have been any of the other wells in clique $\mathbf{c}_1$. Consider another clique $\mathbf{c}_2$ containing wells/pixels q and r. If we further decrease variable $V_{\mathbf{c}_2,q,a}$ of q^a and increase $V_{\mathbf{c}_2,r,a}$, the combined effect will be that q^a remains at same height while height of p^a decreases by δ and r^a increases by δ. We can view this operation as if flow corresponding to ball a of amount δ originating at pixel p and ending at pixel r passing through pixel q as the intermediate node using *edges* in clique $\mathbf{c}_1$ and clique $\mathbf{c}_2$.

The dual objective function requires maximization of the sum of U_p over all pixels. The above discussion suggests that this would involve raising the heights of active balls in the pixel wells which can be achieved by sending flow to an active ball along a *path* from a non active ball with the same label. The non active ball comes down by an amount equal to the flow sent and the active ball increases its height by the same amount. How much flow can be sent is a function of the relative heights of balls in the two wells in question and the V variables associated with the *edges* along the path. The active ball should not float at a height higher than the non active ball when the flow is sent and the height of the non active ball should not become less than the active ball. This ensures that active ball configuration remains feasible. The changes in the values of V variables along the path should be such that no dual feasibility constraint of the type (6) becomes infeasible. Since this flow move can be between any two a balls or b balls it would seem that the equivalent flow graph should allow for both types of flows. We show that this is not necessary. If costs are submodular then the equivalent flow graph on which the max flow problem has to be solved needs to cater for flows in between balls of only one label. In the rest of our discussion we assume that flows of only balls of type a are taking place.

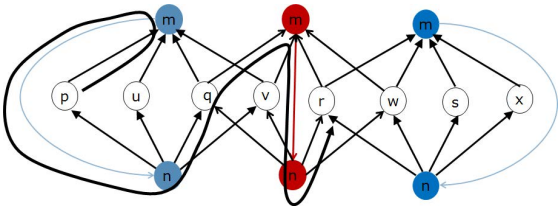
2.2 Flowgraph Construction

We model the flow carrying edges in a clique of size k by a gadget consisting of $k+2$ nodes. There are k nodes $p, q, \cdots, r$ corresponding to the k pixels called *pixel* nodes and two *auxiliary* nodes n and m. There are directed edges called



(a) Gadget corresponding to a 4 clique

(b) Flow in a single gadget



(c) Flow through multiple gadgets

Fig. 1.

conjugate edges from n to all pixel nodes, and from all pixel nodes to node m and a directed edge $m \to n$ from node m to n. The gadget corresponding to a clique of size 4 is given in Figure 1(a). Suppose pixel p sends δ flow to pixel q in the clique. This flow gets modeled by sending δ flow along the path $p \to m \to n \to q$ in the corresponding gadget as shown in Figure 1(b). We denote the flow in edge, say, $p \to m$ in the gadget corresponding to clique $\mathbf{c}$ by $f^{\mathbf{c}}_{pm}$. We can look upon the affect of this flow as bringing down ball a in well p by amount $f^{\mathbf{c}}_{pm}$ and pushing up ball a in well q by the same amount. Seen in this way the relationship between the dual variables and flow is defined by

$$V_{\mathbf{c},p,a} = f^{\mathbf{c}}_{np} - f^{\mathbf{c}}_{pm}. \tag{10}$$

The flow graph for the dual optimization problem using the gadget is created as follows: The set of nodes consists of two distinguished nodes s and t, a pixel node corresponding to each pixel, and two auxiliary nodes for each clique. The pixels and the auxiliary nodes corresponding to a clique are connected by the gadget edges.The edges from node s and to node t depend upon whether the flow graph models ball a's flow or ball b's flow. If flow is being modeled for the movement of ball a then there is a directed edge from s to each pixel node p in whose well the non-active ball is a and is above the active ball. Similarly there is a directed edge from each pixel node p in whose well ball a is active to node t. Figure 1(c) shows the flow path in a more general setting of flow pushing from pixel p to pixel r through pixel q of three different 4 cliques where the cliques have some pixels in common.

Since flow in conjugate edges controls the value of dual variables (equation (10)) and the value of dual variables impacts dual feasibility constraints, capacity

constraints on flow graph edges should be so chosen that under all legal flows all dual feasibility constraints are satisfied. The capacity of the $s \rightarrow p$ or the $p \rightarrow t$ edge, as the case may be, is set equal to the difference between the heights of the two balls in the well corresponding to pixel p because setting of source and sink edge capacities in this way ensures that an active ball can never go above the non active ball in its well. The dual feasibility constraints defined by equation (4) are always satisfied. The capacity of $m \rightarrow n$ edges is set to infinity as dual variable values are independent of flow in these edges.

Since the flow graph models movement of ball a we can assume that all dual variables $V_{\mathbf{c},p,b}$ are set to 0, and every instance of equation 6 simplifies to

$$\sum_{p \in \mathbf{c}: l^p_{\mathbf{c}} = a} V_{\mathbf{c},p,a} \leq W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}), \qquad \mathbf{c} \in \mathcal{C},\ \mathbf{l}_{\mathbf{c}} \in \mathcal{L}^k. \tag{11}$$

Equation (10) implies that in the corresponding flow graph the constraints defined by (11) take the form

$$\sum_{p \in \mathbf{c}: l^p_{\mathbf{c}} = a} (f^{\mathbf{c}}_{np} - f^{\mathbf{c}}_{pm}) \leq W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}), \qquad \mathbf{c} \in \mathcal{C},\ \mathbf{l}_{\mathbf{c}} \in \mathcal{L}^k. \tag{12}$$

We can interpret the above constraints as capacity constraints on the conjugate edges. It should be noted that the capacity constraints are no more limited to single edges as in traditional flow formulations. If we look upon $f^{\mathbf{c}}_{np} - f^{\mathbf{c}}_{pm}$ as effective flow in a pair of conjugate edges then an inequality of the form (12) essentially says that the sum of effective flow in a collection of conjugate edges can not exceed the cost of the configuration $\mathbf{l}_{\mathbf{c}}$ in which the pixels corresponding to the conjugate edges are labeled a. The quantity

$$W_{\mathbf{c}}(\mathbf{l}_{\mathbf{c}}) - \sum_{p \in \mathbf{c}: l^p_{\mathbf{c}} = a} (f^{\mathbf{c}}_{np} - f^{\mathbf{c}}_{pm}) \tag{13}$$

for a $\mathbf{c} \in \mathcal{C}$ and a $\mathbf{l}_{\mathbf{c}} \in \mathcal{L}^k$ is the slack in the corresponding dual feasibility constraint of type (11). This slack can be interpreted as the extent to which any pair of conjugate edges that is participating in the dual feasibility constraint of type (11) can have the flow increased without violating it. The slack can be looked upon as the allowed *capacity* of each of the conjugate edge pair by the corresponding dual feasibility constraint. We define the *residual capacity* in a pair of conjugate edges to be equal to the minimum of the slacks of all the dual feasibility constraints excluding those corresponding to uniform labeling in which it participates. The uniform labeling constraints are excluded because flow conservation constraints imply that for a clique $\mathbf{c}$

$$\sum_{p \in \mathbf{c}} f^{\mathbf{c}}_{pm} = \sum_{p \in \mathbf{c}} f^{\mathbf{c}}_{np},$$

or

$$\sum_{p \in \mathbf{c}} f^{\mathbf{c}}_{np} - \sum_{p \in \mathbf{c}} f^{\mathbf{c}}_{pm} = 0,$$

or

$$\sum_{p \in \mathbf{c}} V_{\mathbf{c},p,a} = 0. \tag{14}$$

In effect the constraints corresponding to uniform labeling given by equation (9) are always satisfied regardless of the flow in conjugate edges.

We make the implications of our definitions clear by an example (c.f. supplementary material for detailed working out). As mentioned earlier flow pushing in the flow graph simulates the movement of balls in two wells of a clique. If in a clique $\mathbf{c}$, ball a in well p comes down by amount δ and ball a of well q in the same clique goes up by the same amount, then in the flow graph this is simulated by "pushing" δ "flow" in the path fragment $p \to m \to n \to q$ of the gadget corresponding to clique c. The effect of this "flow push" is to reduce the value of dual variable $V_{\mathbf{c},p,a}$ by δ and increase the value of $V_{\mathbf{c},q,a}$ by δ. Note that this change in the values of the dual variables does not effect the dual feasibility constraints in which either both $V_{\mathbf{c},p,a}$ and $V_{\mathbf{c},q,a}$ participate or do not participate. However, the slack of those constraints in which only $V_{\mathbf{c},p,a}$ participates increases by δ and those in which only $V_{\mathbf{c},q,a}$ participates decreases by δ. Clearly the value of δ should not be allowed to increase by an amount which makes the slack of a constraint negative, i.e. "restrictions on residual capacity therefore only needs to take into account those dual constraints which can become infeasible by pushing flow". Now let us consider a situation in which residual capacity for the pair of conjugate edges for node q in clique c is 0. This implies that at least one constraint which contains the dual variable $V_{\mathbf{c},q,a}$ is tight. It should be noted that it will still be possible for node p to send flow to q provided all the tight constraints that contain $V_{\mathbf{c},q,a}$ also contain $V_{\mathbf{c},p,a}$. This is because any increase in $V_{\mathbf{c},q,a}$ is counter balanced by corresponding decrease in $V_{\mathbf{c},p,a}$ (path from p to q is $p \to m \to n \to q$) and the constraints that were tight prior to pushing of flow continue to remain tight. The flow from p to q is therefore limited by the constraints in which only $V_{\mathbf{c},q,a}$ is present (and variable $V_{\mathbf{c},p,a}$ is absent). Lemma 3 makes this more precise.

The fact that capacity constraints are on effective flow in a pair of conjugate edges rather than on flow on each individual edge is handled by requiring that flow is non zero in only one of the edges in the conjugate pair and associating the capacity constraint with the edge with non zero flow. Two cases arise: The conjugate edge with non zero flow emanates from n type auxiliary node. In this case the residual graph has two edges. The edge in direction from an auxiliary node (n type) to a pixel node has capacity equal to the residual capacity of the conjugate edge pair, and the edge in the direction from a pixel node to an auxiliary node has capacity equal to the flow in the conjugate edge pair. In the other case, the capacity of the residual edge from the pixel node to the auxiliary node (m type) is infinity, and the reverse direction edge has capacity equal to the flow towards the auxiliary node (m type) in the conjugate edge pair in question. When there is no flow in either of the edges of a conjugate pair, then the residual graph has two edges corresponding to the two conjugate

edges. One emanating from the auxiliary node (n type) has capacity equal to the capacity of the conjugate edge pair. Capacity of the other is infinity. In this case the requirement that only one of the conjugate edges has zero flow is ensured by restricting augmenting flow on paths from s to t that include only one of the edges of a conjugate pair in the residual graph.

2.3 Max Flow Min Cut Relationship for Submodular Potentials and the Max Flow Algorithm

We consider a dual feasibility constraint to *cover* the dual variables (or equivalently pairs of conjugate edges) that are present in its l.h.s. and a set of dual constraints that covers the conjugate edges of an (S,T) cut of a flow graph as their conjugate edge cover. The cost of a conjugate edge cover is defined to be the sum of the r.h.s. of dual feasibility constraints constituting the conjugate edge cover. We define the capacity of an (S,T) cut to be equal to the sum of the value of the smallest cost conjugate edge cover covering the conjugate edge in the cut and the capacities of all the other edges in it (from s to nodes in T and from nodes in S to t). The general relationship between max flow and the capacity of (S,T) cuts is given below.

Lemma 1. *Let (S,T) be the cut in which S is the set of nodes reachable from s in the residual graph when flow is maximal. Value of max flow is equal to the sum of flow in saturated edges from s to nodes in T, and from nodes in S to t and flow in conjugate edges from auxiliary nodes in S to nodes in T. Value of max flow is less than or equal to the capacity of the (S,T) cut.*

It can be shown that [22] when dual feasibility constraints are submodular then max flow in a flow graph is always equal to the value of the (S,T) cut of Lemma 1. The following hold.

Lemma 2. *If $W_c(l_c)$ is submodular for all $\mathbf{c}$ in $\mathcal{C}$, then whenever in the flow graph there exists a conjugate edge that has residual capacity zero and is covered by two tight dual feasibility constraints corresponding to pixel sets $\mathbf{X}$ and $\mathbf{Y}$ of some clique, the dual feasibility constraint corresponding to pixel set $\mathbf{X} \cup \mathbf{Y}$ is also tight.*

Theorem 1. *When costs are submodular, in the flow graph corresponding to the dual optimization problem, max flow is equal to min cut and corresponding primal and dual solutions are optimal.*

Lemma 3. *A saturated conjugate edge $n \rightarrow p$ corresponding to a clique $\mathbf{c}$ in presence of flow f cannot be in an augmenting path if the only tight dual feasibility constraint covering it covers no other edge, or if the intersection of all tight dual feasibility constraints covering it contains no other edge.*

We now give details of an augmenting path based max flow algorithm for the gadget based flow graphs.

We add "include all saturated conjugate edges in the residual graph to which Lemma 3 does not apply" as an additional rule for adding edges to the residual graph. Such a saturated edge of the residual graph can be included in an $s-t$ augmenting path provided there is an immediately preceding conjugate edge in the path covered by the same tight dual constraint. The residual capacity of such a saturated edge is contextually defined. It is equal to the minimum of the slacks of all dual constraints which cover the saturated edge excluding those which cover both the saturated edge and the preceding edge. It can be shown [22] that if flow is not maximum in a flow graph corresponding to the dual optimization problem with submodular costs, then there will exist an $s-t$ augmenting path in the residual graph consistent with that flow.

For the purposes of determining flow that can be pushed in an augmenting path from s to t, we break the part of the augmenting path that excludes nodes s and t into path fragments. Each path fragment is between two pixel nodes p and q of a clique and has the form $p \rightarrow n \rightarrow q$ or $p \rightarrow m \rightarrow n \rightarrow q$. Note that each of these path fragments lies completely in the gadget corresponding to a clique. For what follows we define the length of an augmenting path to be the number of path fragments in it. With the augmenting path length defined as above, the property that the length of the shortest flow augmenting path in the residual graph is non decreasing, if flow is always augmented on the shortest length augmenting path, holds [22].

Lemma 4. *Let the shortest augmenting path length from s to t after l shortest augmenting path flow augmentations be denoted by $\delta^l(s,t)$. Then $\delta^{l+1}(s,t) \geq \delta^l(s,t)$.*

If we use Edmonds and Karp's shortest path augmentation strategy [23] then the complexity of the algorithm, assuming $|\mathcal{C}| \approx n$, can be shown to be $O(2^k k^2 n^3)$ [22]. The corresponding complexity for reduction bases algorithm can be shown to be $O(2^{5k} n^3)$ [22].

3 Experiments and Discussion

All experiments were conducted on a computer with 2.5GHz dual core processor, 2GB of RAM running Windows 7 operating system with 64 bit addressability. Our algorithm has been implemented in C++. The comparison has been done using publicly available code of Ishikawa [24] which uses QPBO [25] and codes from Darwin framework [26] that we could run. These were Iterated Conditional Modes (ICM) [10], Max Product Inference (MPI) [11], Asynchronous Max Product Inference (AMPI) [11], TRWS [27] and Dual Decomposition MAP inference (DD) [9]. For the purpose of reporting we will refer to our algorithm as GC and the Ishikawa's reduction code using QPBO as IQ. It should be noted that GC gives optimal results for submodular potentials. Experiments, reported here, put in perspective comparative performance and quality of the prevalent techniques on 2 label multi clique problems when potentials are submodular. This focus is important because real life problems are multi label in general and get solved

either by repeatedly solving of 2 label subproblems or by direct techniques which generalize the 2 label algorithms. Our algorithmic technique has a natural generalization for handling non-submodular potentials directly. We will be reporting that algorithm and a comparative study separately later.

3.1 Binary Image Segmentation

Experiments have been conducted on a synthetic image with gaussian noise added to the image. The terminal weights (unary potential) for black and white labeling were chosen as difference of pixel intensity from their respective ideal values (0 and 255). The segmentation problem has been solved using rectangular cliques of various sizes with a clique anchored on every pixel resulting in number of cliques in the system equal to the number of pixels in the image. We have experimented with three clique potentials. They are SQRT which penalizes as per square root of the number of edges present in the labeling, NE which penalizes a labeling proportional to number of edges, and SQR which penalizes according to the square of number of edges. NE and SQRT are submodular for cliques of size 2×2, while SQR is not. Tables 1 and 2 provide details of energy outputted and time taken by various algorithms on 4 clique problems of various sizes. It should be noted that algorithms which output near optimal results take couple of order of magnitude more time. MPI is only two to three times slower than GC but has very poor energy output. MPI seems to move into a local minima fast and does have the ability to move out of it. Performance of AMPI on the potentials used was very similar to MPI in both energy outputted and time taken. Also, algorithms like DD and ICM which have good energy convergence started taking tens of minutes at image size of 150×150 and simply failed to run when image size was increased to 200×200.

Figure 2 shows the output image obtained on running various algorithms on the noisy image. Note that the quality of the output image is directly proportional to the energy outputted by the algorithms in general. Poor quality of output image for IQ is because the QBPO algorithm, as mentioned earlier, can

Table 1. Energy of the solution inferred with various methods for a 4 clique problem at different image sizes

Image Size	Clique Potential	DD	MPI	ICM	TRWS	IQ	GC
50×50	SQRT	219496	267018	231803	221815	232197	219161
50×50	NE	226956	293521	231483	228289	226424	226424
50×50	SQR	223668	300197	230690	226258	231101	223298
100×100	SQRT	845883	1056459	918196	848071	919249	843056
100×100	NE	854019	1131180	916320	858773	851585	851585
100×100	SQR	863277	1221611	920385	872259	920809	859448

Table 2. Inference time in ms for various methods for a 4 clique problem at different image sizes

Image Size	Clique Potential	DD	MPI	ICM	TRWS	IQ	GC
50×50	SQRT	2084	8	246	7769	86	4
50×50	NE	2207	7	244	8228	4	4
50×50	SQR	2215	7	247	7823	89	4
100×100	SQRT	14202	41	3988	107873	493	14
100×100	NE	14123	36	3986	107137	15	15
100×100	SQR	16466	40	5976	107483	577	15

(a) Orig (b) Input (c) GC 818763 (d) IQ 912134 (e) ICM 834381 (f) DD 819801 (g) MPI 967039

Fig. 2. Segmentation results from inference methods. Clique potential is SQRT. Algorithm name is followed by the energy outputted.

leave nodes unlabeled. In this case the grey nodes are those left unlabeled by QBPO. It is interesting to point out that IQ gives optimal results for NE potential. This is because for NE the energy function in the polynomial form has no monomial terms of order larger than two. IQ, in this case, does not introduce any auxiliary variables and leaves the polynomial form as is. Since for second order potentials submodularity and regularity are identical IQ terminates with the optimal solution. For other algorithms, NE and SQR potentials performance is along the lines for SQRT.

Table 3. Time comparison at various clique and image sizes

Image Size	Clique Size	DD	ICM	GC	IQ
100×100	4	15081	6061	14	519
100×100	6	36683	9294	103	7398
50×50	8	21290	377	159	32604
50×50	9	44872	431	435	206756
50×50	10	88577	447	1102	DNR
50×50	12	400543	535	7125	DNR

Table compares the algorithms from the perspective of time taken with varying clique sizes. Time taken by DD seems to be varying in a way similar to GC. For higher order cliques ICM's performance seems to be similar to MPI. However, for IQ time taken has increased at a much rapid rate. This increase is due to the exponentially large number of nodes added in reduction based algorithms. Our experiments indicate that for large clique sizes GC is hundreds of time faster than IQ.

4 Conclusions

We consider GC to be the first step towards development of a truly practical strongly polynomial algorithm for optimizing submodular higher order energy functions that arise in problems modeled by MRF-MAP. Its primary limitations are in the multiplicative factor of 2^k that shows up in the time complexity analysis. While this limits its realistic practical (in terms of time) use on images of 500×500 to at most 9 cliques, the worst case time complexity for this image size and $k = 16$ will be less than $O(n^5)$ the time complexity of the most efficient strongly polynomial algorithms known for general submodular function optimization. The gadget we have introduced for modeling a clique is quite powerful in that it can be generalized to handle non submodular higher order cost functions also. Of course, practical algorithms for non submodular functions will only find approximate solutions. We have developed approximate algorithms based on a generalization of the gadget introduced here. We can show that for submodular deviation functions introduced by Rother et al. [7] to handle sparse potentials GC needs to track only one constraint for every preferred labeling resulting low order strongly polynomial algorithms that are extremely efficient. These results will be reported in a future paper.

References

1. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. 23, 1222–1239 (2001)
2. Kleinberg, J., Tardos, E.: Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. J. ACM 49, 616–639 (2002)
3. Komodakis, N., Tziritas, G.: Approximate labeling via graph cuts based on linear programming. IEEE Trans. Pattern Anal. Mach. Intell. 29, 1436–1453 (2007)
4. Ishikawa, H.: Transformation of general binary MRF minimization to the first-order case. IEEE Trans. Pattern Anal. Mach. Intell. 33, 1234–1249 (2011)
5. Kohli, P., Ladický, L., Torr, P.H.: Robust higher order potentials for enforcing label consistency. International Journal of Computer Vision 82, 302–324 (2009)
6. Roth, S., Black, M.J.: Fields of experts. International Journal of Computer Vision 82, 205–229 (2009)
7. Rother, C., Kohli, P., Feng, W., Jia, J.: Minimizing sparse higher order energy functions of discrete variables. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1382–1389 (2009)

8. Woodford, O., Torr, P., Reid, I., Fitzgibbon, A.: Global stereo reconstruction under second order smoothness priors. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
9. Komodakis, N., Paragios, N.: Beyond pairwise energies: Efficient optimization for higher-order MRF's. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2985–2992 (2009)
10. Besag, J.: On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society B-48, 259–302 (1986)
11. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. In: Adaptive Computation and Machine Learning. MIT Press (2009)
12. Kolmogorov, V., Rother, C.: Minimizing nonsubmodular functions with graph cuts-a review. IEEE Trans. Pattern Anal. Mach. Intell. 29, 1274–1279 (2007)
13. Kahl, F., Strandmark, P.: Generalized roof duality for pseudo-boolean optimization. In: IEEE International Conference on Computer Vision, pp. 255–262 (2011)
14. Boros, E., Gruber, A.: On quadratization of pseudo-boolean functions. In: International Symposium on Artificial Intelligence and Mathematics (2012)
15. Fix, A., Gruber, A., Boros, E., Zabih, R.: A graph cut algorithm for higher-order markov random fields. In: IEEE International Conference on Computer Vision, pp. 1020–1027 (2011)
16. Freedman, D., Drineas, P.: Energy minimization via graph cuts: settling what is possible. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 939–946 (2005)
17. Kolmogorov, V., Zabin, R.: What energy functions can be minimized via graph cuts? IEEE Transactions on Pattern Analysis and Machine Intelligence 26, 147–159 (2004)
18. Živný, S., Jeavons, P.G.: Classes of submodular constraints expressible by graph cuts. Constraints 15, 430–452 (2010)
19. Kolmogorov, V.: Generalized roof duality and bisubmodular functions. Discrete Applied Mathematics 160, 416–426 (2012)
20. Iwata, S., Orlin, J.B.: A simple combinatorial algorithm for submodular function minimization. In: Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1230–1237 (2009)
21. Kolmogorov, V.: Minimizing a sum of submodular functions. CoRR abs/1006.1990 (2010)
22. (Supplementary material)
23. Edmonds, J., Karp, R.M.: Theoretical improvements in algorithmic efficiency for network flow problems. J. ACM 19, 248–264 (1972)
24. HOCR version 1.02, `http://www.f.waseda.jp/hfs/software.html`
25. QPBO version 1.3, `http://pub.ist.ac.at/~vnk/software.html`
26. Darwin framework, version 1.1.2, `http://drwn.anu.edu.au/`
27. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE Trans. Pattern Anal. Mach. Intell. 28, 1568–1583 (2006)