Optimizing Binary MRFs via Extended Roof Duality

Carsten Rother[1], Vladimir Kolmogorov[2], Victor Lempitsky[3], Martin Szummer[1]

[1] Microsoft Research Cambridge {carrot, szummer}@microsoft.com
[2] University College London vnk@adastral.ucl.ac.uk
[3] Moscow State University victorlempitsky@gmail.com

Abstract

Many computer vision applications rely on the efficient optimization of challenging, so-called non-submodular, binary pairwise MRFs. A promising graph cut based approach for optimizing such MRFs known as "roof duality" was recently introduced into computer vision. We study two methods which extend this approach. First, we discuss an efficient implementation of the "probing" technique introduced recently by Boros et al. *[5]. It simplifies the MRF while preserving the global optimum. Our code is 400-700 faster on some graphs than the implementation of [5]. Second, we present a new technique which takes an arbitrary input labeling and tries to improve its energy. We give theoretical characterizations of local minima of this procedure.*

We applied both techniques to many applications, including image segmentation, new view synthesis, super-resolution, diagram recognition, parameter learning, texture restoration, and image deconvolution. For several applications we see that we are able to find the global minimum very efficiently, and considerably outperform the original roof duality approach. In comparison to existing techniques, such as graph cut, TRW, BP, ICM, and simulated annealing, we nearly always find a lower energy.

1. Introduction

Most early vision problems can be formulated in terms of Markov random fields (MRFs). Algorithms for MRF inference therefore are of fundamental importance for computer vision. The MAP-MRF approach (computing maximum a posteriori configurations in an MRF) has proven to be extremely successful for many vision applications such as stereo, image segmentation, image denoising, super-resolution, new view synthesis and others. We refer to [22] for an overview of MRF optimization techniques in vision.

Binary MRFs In this paper we focus on a special class of MRFs. Namely, we consider the problem of minimizing an energy function of the form

$$E(\mathbf{x}) = \theta_{\text{const}} + \sum_{p\in\mathcal{V}} \theta_p(x_p) + \sum_{(p,q)\in\mathcal{E}} \theta_{pq}(x_p, x_q)\ . \quad (1)$$

Here $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is an undirected graph. The set of nodes $\mathcal{V}$ usually corresponds to pixels, and $x_p \in \{0, 1\}$ denotes the label of node p. It is well-known that if the function E is submodular, i.e. every pairwise term θ_{pq} satisfies

$$\theta_{pq}(0,0) + \theta_{pq}(1,1) \leq \theta_{pq}(0,1) + \theta_{pq}(1,0), \quad (2)$$

then a global minimum of E can be computed in polynomial time as a minimum s-t cut in an appropriately constructed graph ("submodular graph cuts"). Submodular functions are very important, for example, for the image segmentation problem (see e.g. [6]). In many vision applications, however, condition (2) is not satisfied. We focus on the problem of minimizing non-submodular functions, which is a very challenging task (in general, NP-hard).

A promising approach for this problem called *roof duality* was proposed in [12] (see a review in [15]). It produces *part* of an optimal solution. Boros *et al.* [3] give an efficient algorithm for computing a roof dual. It can be viewed as a generalization of the standard graph cut algorithm used in vision: for submodular functions the two methods give the same answer and have exactly the same running time, except for a linear time overhead. We will refer to this method as the *QPBO* algorithm, which stands for *quadratic pseudo-boolean optimization* - this is what the minimization problem (1) is called in [12, 3]. Recently it was successfully applied to vision applications such as MR reconstruction [18] and texture restoration [15].

Our contributions In some cases the roof duality approach does not work very well, i.e. it leaves many nodes unlabeled. We investigate two extensions of the roof duality approach. The first one is the "probing" method introduced recently in [5]. This is an exact technique: it simplifies the energy by contracting and fixing nodes while preserving the global optimum. We describe an efficient implementation and observe that our code is 400-700 times faster than the software of [5] on several 4-connected grid graphs (such grids are common in vision.) Our experiments show that this makes the algorithm practical for vision applications.

Second, we develop a new approximate technique: it takes an input solution and tries to improve its energy. The energy is guaranteed not to increase, and experimentally often decreases. Both techniques can be combined; in some cases such combination outperformed other methods that

we tested (simulated annealing, ICM, max-product belief propagation, graph cut, and TRW).

Last but not least, we show the importance of the roof duality approach and its extensions for many vision applications, such as image segmentation, diagram recognition, new view synthesis, and image deconvolution. Note, experiments in [5] were outside computer vision.

Related work There is an extensive literature devoted to minimizing energy (1). Exact methods for this problem are usually branch-and-bound style methods, with different techniques for obtaining a lower bound. A large number of heuristic ideas have also been applied to this problem, e.g. tabu search, scatter search, simulated annealing, evolutionary algorithms. We refer to [1, 4] and references therein for an overview of different exact and approximate methods.

2. Optimizing Binary MRFs: Roof duality

In this section we give an overview of the roof duality approach for optimizing binary MRFs introduced in [12]. The idea is to solve a particular linear programming (LP) relaxation of the energy where integer constraints $x_p \in \{0,1\}$ are replaced with linear constraints $x_p \in [0,1]$. It can be shown that this LP has a half-integer optimal solution $\bar{\mathbf{x}}$, i.e. $\bar{x}_p \in \{0, 1, \frac{1}{2}\}$ for every node p. It is convenient to define the corresponding *partial* labeling $\mathbf{x}$ of the integer problem with $x_p \in \{0, 1, \varnothing\}$ where value $\varnothing$ means that the node is "unlabeled".

The LP relaxation above can be solved in several different ways. The algorithm in [3] is perhaps the most efficient. We review this method, which we call QPBO, in Section 2.1. As we mentioned, it produces a partial labeling $\mathbf{x}$. Properties of this labeling (in particular, *persistency*, or *partial optimality*) are discussed in Section 2.2.

2.1. The QPBO Algorithm

We describe the algorithm of [3] using the notion of *reparameterization*.

Reparameterization Let us introduce the following notation. The energy of eq. (1) is specified by the constant term θ_{const}, unary terms $\theta_p(i)$ and pairwise terms $\theta_{pq}(i,j)$ ($i,j \in \{0,1\}$). It will be convenient to denote the last two terms as $\theta_{p;i}$ and $\theta_{pq;ij}$, respectively. We can concatenate all these values into a single vector $\theta = \{\theta_\alpha \,|\, \alpha \in \mathcal{I}\}$ where the index set is $\mathcal{I} = \{\text{const}\} \cup \{(p;i)\} \cup \{(pq;ij)\}$. Note that $(pq;ij) \equiv (qp;ji)$, so $\theta_{pq;ij}$ and $\theta_{qp;ji}$ are the same element. We will use the notation θ_p to denote a vector of size 2 and θ_{pq} to denote a vector of size 4.

Vector θ' is called a *reparameterization* of vector θ if the energy functions E' and E that they define are the same, i.e. $E'(\mathbf{x}) = E(\mathbf{x})$ for all labelings $\mathbf{x}$. As a particular example, we can subtract some constant from vectors θ_p or θ_{pq} and add the same constant to θ_{const}. Another possible transformation involves edge $(p,q) \in \mathcal{E}$ and label $j \in \{0,1\}$: we can subtract a constant from components $\theta_{pq;ij}$ for all $i \in \{0,1\}$ and add the same constant to $\theta_{p;j}$.

Normal form We will say that the vector θ is in a *normal form* if it satisfies the following:

(a) $\min\{\theta_{p;0}, \theta_{p;1}\} = 0$ for all nodes p.
(b) $\min\{\theta_{pq;0j}, \theta_{pq;1j}\} = 0$ for all $(p,q) \in \mathcal{E}$ and $j \in \{0,1\}$.

Normal form implies the following: $\theta_{pq;00} = \theta_{pq;11} = 0$, $\theta_{pq;01}, \theta_{pq;10} \geq 0$ if edge (p,q) is submodular; and $\theta_{pq;01} = \theta_{pq;10} = 0$, $\theta_{pq;00}, \theta_{pq;11} \geq 0$ if (p,q) is supermodular (see fig. 1 in [15]).

Algorithm The first step of the QPBO algorithm is to reparameterize vector θ into a normal form. This can be done in linear time (see e.g. [15]). Then a directed weighted graph $G = (V, A)$ is constructed. For each node $p \in \mathcal{V}$, two nodes $p, \bar{p}$ are added to V. (They correspond to variable x_p and its negation $\bar{x}_p = 1 - x_p$, respectively). In addition, there are two special nodes - the source s and the sink t which correspond to labels 0 and 1. Thus, $V = \{p, \bar{p} \,|\, p \in \mathcal{V}\} \cup \{s, t\}$. For each non-zero element θ_α (except for θ_{const}) two directed arcs are added to the graph with weight θ_α; details can be found in [2, 15].

Finally, a minimum s-t cut (S,T) in G is computed by computing a maximum flow from s to t. This cut gives an optimal solution to the LP relaxation and corresponding partial labeling $\mathbf{x}$ as follows: (i) if $p \in S, \bar{p} \in T$ then $x_p = 0$; (ii) if $p \in T, \bar{p} \in S$ then $x_p = 1$; (iii) otherwise, $x_p = \varnothing$. It is worth noting that the maximum flow in G defines a reparameterization of the energy. There are certain relations between this reparameterization and partial labeling $\mathbf{x}$ (*complementary slackness* conditions - see e.g. [20]).

Choosing a minimum cut One technical issue is that graph G may have several minimum cuts (S,T). They may correspond to different partial labelings $\mathbf{x}$ with different sets of labeled nodes. In general, there exist "extreme" cuts $(S^{\min}, T^{\min})$ and $(S^{\max}, T^{\max})$ such that for any other minimum cut (S,T) there holds $\text{dom}(\mathbf{x}^{\min}) \subseteq \text{dom}(\mathbf{x}) \subseteq \text{dom}(\mathbf{x}^{\max})$ where $\mathbf{x}^{\min}$, $\mathbf{x}^{\max}$, and $\mathbf{x}$ are the labelings defined by these cuts and $\text{dom}(\mathbf{x})$ denotes the set of labeled nodes in $\mathbf{x}$ ("domain of $\mathbf{x}$"). Details of how to compute these cuts can be found in [2, 15, 20]. Note that labeling $\mathbf{x}^{\min}$ is unique, but $\mathbf{x}^{\max}$ in general may depend on the cut.

2.2. Properties of QPBO

We now review properties of partial labeling $\mathbf{x}$ produced by the QPBO method ([12], see also [2]). Perhaps the most important one is the following:

[$\mathcal{P}1$] (Weak autarky) *Let $\mathbf{y}$ be an arbitrary complete labeling, and let $\mathbf{z} = FUSE(\mathbf{y}, \mathbf{x})$ be the "fusion" of $\mathbf{y}$ and $\mathbf{x}$: $z_p = x_p$ if $p \in \text{dom}(\mathbf{x})$, and $z_p = y_p$ otherwise. Then $E(\mathbf{z}) \leq E(\mathbf{y})$.*

If we take $\mathbf{y}$ to be a global minimum, then we see that $\mathbf{x}$ is a part of some optimal solution:

[$\mathcal{P}2$] (Weak persistency, *or* partial optimality) *There exists a global minimum* $\mathbf{x}^*$ *of energy* (1) *such that* $x^*_p = x_p$ *for all labeled nodes* $p \in \mathrm{dom}(\mathbf{x})$.

Strong persistency Properties [$\mathcal{P}1$] and [$\mathcal{P}2$] are valid for any partial labeling $\mathbf{x}$ produced by QPBO. If we take $\mathbf{x} = \mathbf{x}^{\min}$, then these properties can be strengthened:

[$\mathcal{P}1'$] (Strong autarky) *Let* $\mathbf{y}$ *be a complete labeling, and let* $\mathbf{z}$=*FUSE*($\mathbf{y}$,$\mathbf{x}$). *If* $\mathbf{z} \neq \mathbf{y}$ *then* $E(\mathbf{z}) < E(\mathbf{y})$.

[$\mathcal{P}2'$] (Strong persistency) Any *global minimum* $\mathbf{x}^*$ *of energy* (1) *satisfies* $x^*_p = x_p$ *for all nodes* $p \in \mathrm{dom}(\mathbf{x})$.

Which nodes are labeled? Clearly, the usefulness of the algorithm depends on how many nodes are labeled. In general, we cannot expect that the method will label all nodes since minimizing energy (1) is an NP-hard problem. In some special cases, however, the method is guaranteed to label all nodes [12]:

[$\mathcal{P}3$] *If the energy does not have any frustrated cycles then labelings* $\mathbf{x}^{\max}$ *produced by QPBO are complete, i.e.* $\mathrm{dom}(\mathbf{x}^{\max}) = \mathcal{V}$.

(A cycle is called *frustrated* if it contains an odd number of non-submodular terms). The condition of this property holds, in particular, for submodular functions.

The last property follows from the QPBO construction:

[$\mathcal{P}4$] *The algorithm is invariant with respect to "flipping" a subset of nodes* $\mathcal{U} \subseteq \mathcal{V}$, *i.e. swapping the meaning of 0 and 1 for pixels* $p \in \mathcal{U}$. *(This flipping transforms submodular terms between* $\mathcal{U}$ *and* $\mathcal{V} \backslash \mathcal{U}$ *into non-submodular, and vice versa).*

3. Extended Roof Duality

The roof duality works quite well in "simple" cases (e.g. when the number of non-submodular terms is small), but in more difficult cases it may leave many nodes unassigned (Sec. 4). In this paper we study two extensions of the roof duality approach and show that they outperform the basic algorithm for many vision applications. The first extension is the "probing" method introduced in [5]. We call it QPBOP where "P" stands for "probing". Its aim is to find the global optimum for nodes which QPBO failed to label. Sec. 3.1 reviews this method and also describes an efficient implementation. Then in Sec. 3.2 we propose a new algorithm which we call QPBOI, where "I" stands for "improve". Its aim is to efficiently improve a given reference solution. Unless noted otherwise, we will assume for simplicity that QPBO produces the (unique) strongly persistent solution $\mathbf{x} = \mathbf{x}^{\min}$, and write it as $\mathbf{x} = \mathrm{QPBO}(E)$. Note, in practice $\mathbf{x}^{\min}$ and $\mathbf{x}^{\max}$ are often the same [20].

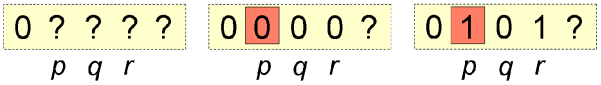

Figure 1. **Basic idea of QPBOP.** Left: QPBO labeling for the current energy, '?' means unlabeled. Middle & right: labelings after fixing node p (red). We can conclude that $x^*_q = 0$ and $x^*_r = x^*_p$ for any global minimum $\mathbf{x}^*$. Therefore, node q can be fixed to 0, and nodes p and r can be contracted.

We will use an operation called "fixing a node". Let $\mathbf{x} = \mathrm{QPBO}(E)$, and consider unlabeled node p with $x_p = \varnothing$ and given label $i \in \{0, 1\}$. Define energy $E' = E[p \leftarrow i]$ as follows: $E'(\mathbf{y}) = E(\mathbf{y}) + E_p(y_p)$ where E_p is a "hard constraint" term with $E_p(i) = 0, E_p(1-i) = C_p$ and C_p is a sufficiently large constant. If we run QPBO for energy E' then we obtain a new partial labeling $\mathbf{x}'$ in which more nodes may have become labeled. We refer to this behaviour as "spreading". It is easy to show the "monotonicity" property, i.e. that $x'_q = x_q$ for $q \in \mathrm{dom}(\mathbf{x})$ and $x'_p = i$ (see [20]).

Instead of adding term E_p, it is also possible to remove term θ_p along with all incident pairwise terms θ_{pq} while modifying unary terms θ_q. Note that adding the hard constraint term and removing node p are equivalent (see [20]).

3.1. QPBOP: Preserving Global Optimality

The basic idea of probing [5] is illustrated in Fig. 1. Let $\mathbf{x} = \mathrm{QPBO}(E)$, and consider unlabeled node p. Let us fix p to 0 and to 1 and run QPBO in each case. We will obtain two partial labelings $\mathbf{x}^0 = \mathrm{QPBO}(E[p \leftarrow 0])$ and $\mathbf{x}^1 = \mathrm{QPBO}(E[p \leftarrow 1])$. Let $\mathcal{U}$ be the set

$$\mathcal{U} = \left[\mathrm{dom}(\mathbf{x}^0) \cap \mathrm{dom}(\mathbf{x}^1)\right] \; - \; \left[\mathrm{dom}(\mathbf{x}) \cup \{p\}\right].$$

By the strong persistency property, we can draw the following information about global minima $\mathbf{x}^*$ of energy E:

$$x^*_p = i \quad \Rightarrow \quad x^*_q = x^i_q \qquad\qquad \forall\, i \in \{0,1\}, q \in \mathcal{U}.$$

Thus, nodes in $\mathcal{U}$ can be excluded from the energy without affecting the global minimum (or minima). Indeed, consider node $q \in \mathcal{U}$. Two cases are possible:

i) $x^0_q = x^1_q = j$. Then $x^*_q = j$ for all global minima $\mathbf{x}^*$, therefore x_q can be fixed to j.

ii) $x^0_q \neq x^1_q$. This means that either (a) $x^0_q = 0, x^1_q = 1$, or (b) $x^0_q = 1, x^1_q = 0$. In case (a) we know that $x^*_p = x^*_q$ for all global minima $\mathbf{x}^*$, therefore we can "contract" p and q. In case (b) there holds $x^*_p = 1 - x^*_q$ for all global minima $\mathbf{x}$, therefore we can "flip" variable x_q (change the meaning of 0 and 1) and then contract p and q.

For details of the contract operation see [20]. In this operation edges (q, r) are replaced with edges (p, r), self-loops are deleted, and parallel edges are merged.

If set $\mathcal{U}$ is nonempty, then the operations above will modify the energy reducing the number of nodes; the new set

of nodes is $\mathcal{V} - \mathcal{U}$. Then we run QPBO again for the new energy obtaining a new partial labeling $\mathbf{x}'$. A "monotonicity" property holds, i.e. $x'_q = x_q$ for $q \in \mathrm{dom}(\mathbf{x})$ (see [20]). Thus, nodes in $\mathrm{dom}(\mathbf{x}) - \mathcal{U}$ are labeled in $\mathbf{x}'$ (and in fact other nodes may become labeled as well).

We can repeat these operations for other nodes of the new energy. In the end we obtain a new energy E' defined on a graph $(\mathcal{V}', \mathcal{E}')$ and function $f : 2^{\mathcal{V}'} \to 2^{\mathcal{V}}$ which maps configurations $\mathbf{y}$ of energy E' to configurations $\mathbf{x}$ of the original energy[1].

Proposition 1. *(a) Function f gives a one-to-one mapping between the sets of optimal solutions of energies E' and E. (b) For any labeling $\mathbf{y} \in 2^{\mathcal{V}'}$ there holds $E(f(\mathbf{y})) = E'(\mathbf{y})$.*

Adding directed constraints It often happens that $\mathrm{dom}(\mathbf{x}^0) \neq \mathrm{dom}(\mathbf{x}^1)$. Then it is possible to add directed constraints of the form $(x^*_p = i) \Rightarrow (x^*_q = x^i_q)$. More details and algorithm's summary can be found in [5, 20].

3.1.1 Efficient Implementation

We now describe an efficient implementation of QPBOP. As noted in [5], it is important to reuse previous calculations when the graph is updated (e.g. when the nodes are contracted). A correct implementation of the update operations, however, requires some care: the graph and flow constructed by the QPBO method have a certain property which must be preserved. Boros *et al.* [5] propose to maintain the *symmetry* condition which says that each arc holds the same flow as its "mate". This is achieved by modifying the maxflow algorithm[2]. We propose to maintain the "relaxed symmetry condition" instead (see [20]). One advantage of this is that the algorithm can work with integer capacities, whereas maintaining the symmetry condition requires floating point numbers (even if the original costs are integers).

We used the maxflow algorithm in [7], and reused flow and search trees as described in [13]. We modified the code so that it maintains a list of visited nodes; thus, set $\mathcal{U}$ can be traversed without going through the entire graph. Using the *relaxed* symmetry condition makes the update operation for nodes p, q quite fast: it involves only these nodes. In contrast, maintaining the symmetry condition using techniques in [7, 13] appears more difficult. After every maxflow computation we would need to go through all edges that have been accessed and restore the symmetry property. This could affect many edges, which would make reusing the search trees more complicated.

Our scheme also has a disadvantage: with the symmetry condition the lower bound on the function (represented by the amount of pushed flow) never decreases, which is not necessarily the case with the relaxed symmetry condition.

Order of processing nodes Experimentally, the order in which nodes are probed affects both the final result and the running time. The difference in results appears insignificant in practice (see [20]). However, optimizing the order is quite important for reducing the number of probed nodes (and, thus, the running time). We found that a particular heuristic performs consistently better than random orders[3].

Comparison of implementations Our tests on 3 random 4-connected grid graphs showed that our implementation is 400-700 times faster than the implementation of [5] (see [20]). It should be noted that there are many differences between the implementations: maxflow algorithm used, search tree recycling, symmetry vs. relaxed symmetry condition, integer vs. floating point capacities, data structures for storing the graph, and ordering of processing nodes. We believe that a major factor in the speed-up is the maxflow algorithm in [7] together with reusing search trees [13], whose use is simplified because of the relaxed symmetry condition. We make our code publicly available; we hope that this would have a significant practical impact in vision.

3.2. QPBOI: Improving a Given Solution

So far we have discussed exact methods (QPBO and QPBOP) which give information about global minima of the energy. We now turn to the problem of obtaining a good approximate solution. Let us assume that we have a (complete) input labeling $\mathbf{x}$ obtained via some method (e.g. random or max-product BP). Our goal is to try to improve this labeling using QPBO.

Let us pick an arbitrary subset of nodes $\mathcal{S} \subset \mathcal{V}$ and fix nodes in $\mathcal{S}$ to labels given by $\mathbf{x}$. We denote the obtained energy as $E[\mathcal{S} \leftarrow \mathbf{x}]$. Now we can run QPBO for this energy obtaining partial labeling $\mathbf{y}$. Obviously, $y_p = x_p$ for nodes $p \in \mathcal{S}$. Property $[\mathcal{P}1']$ immediately implies the following

Proposition 2. *Let $\mathbf{z} = FUSE(\mathbf{x}, \mathbf{y})$, i.e. $z_p = y_p$ if $x_p \neq \varnothing$, and $z_p = x_p$ otherwise. If $\mathbf{z} \neq \mathbf{x}$ then $E(\mathbf{z}) < E(\mathbf{x})$.*

Thus, we can set $\mathbf{x} := \mathbf{z}$ and repeat the procedure for a different subset $\mathcal{S}$. The construction guarantees that the energy of labeling $\mathbf{x}$ does not increase. We call this technique QPBOI, where "I" stands for improve.

[1] Function f can be described via two mappings $\alpha : \mathcal{V} \to \mathcal{V}' \cup \{0\}$, $\sigma : \mathcal{V} \to \{0, 1\}$. For configuration $\mathbf{y} \in 2^{\mathcal{V}'}$ labeling $\mathbf{x} = f(\mathbf{y})$ is defined as follows: if $\alpha_p = 0$ for node $p \in \mathcal{V}$ then $x_p = \sigma_p$, otherwise $x_p = (y_q + \sigma_p) \bmod 2$ where $q = \alpha_p$. Note that mapping α defines a partitioning of the set $\mathcal{V}$.

[2] The implementation of [5] is based on Dinic algorithm, but every augmentation is performed on a pair of "mate" paths (personal communications with G. Tavares).

[3] We split the execution into iterations; one iteration consists of processing nodes in a certain set. In a given iteration we record nodes which made progress, i.e. whose processing resulted in changes to the energy. We then erode this set of nodes by a fixed amount ($d = 3$), and in the next iteration test only nodes in the eroded set. When the set becomes empty, we process all nodes again.

To achieve efficiency, we propose to use a nested sequence of subsets $\mathcal{S}$ according to some ordering of nodes $\pi : \mathcal{V} \to \{1, \dots, |\mathcal{V}|\}$. Then flow and search trees can be reused as in [13]. One iteration of QPBOI is given below.

- Select an ordering of nodes π.
- Initialization: Compute $\mathbf{y} = \text{QPBO}(E)$, set $\mathbf{x} := \text{FUSE}(\mathbf{x}, \mathbf{y})$, $\mathcal{S} := \text{dom}(\mathbf{y})$.
- For nodes $p \in \mathcal{V}$ do in the order π:
  - If $p \notin \mathcal{S}$ compute $\mathbf{y} = \text{QPBO}(E[\mathcal{S} \cup \{p\} \leftarrow \mathbf{x}])$, set $\mathbf{x} := \text{FUSE}(\mathbf{x}, \mathbf{y})$, $\mathcal{S} := \text{dom}(\mathbf{y})$.

It is worth mentioning that the QPBOI procedure can be generalized. If, for example, $x_p = x_q$ in the current solution $\mathbf{x}$, then nodes p and q can be contracted. We informally tested one particular version of such operations (see [20]) but found that it performed worse than fixing nodes.

Local minima of QPBOI In order to understand the capabilities of QPBOI we now analyze local minima of this procedure.

Definition 3. *Labeling* $\mathbf{x}$ *is called* stable *(or* QPBO-stable*) if no QPBOI operation can change it, i.e. for any subset* $\mathcal{S} \subset \mathcal{V}$ *there holds* $y_p = x_p$ *for* $p \in \text{dom}(\mathbf{y})$ *where* $\mathbf{y} = \text{QPBO}(E[\mathcal{S} \leftarrow \mathbf{x}])$.

The theorem below exhibits a large class of functions for which stable labelings are essentially global minima.

Theorem 4. *Suppose energy* E *and labeling* $\mathbf{x}$ *satisfy at least one of the following conditions: (a)* E *does not have frustrated cycles. (b)* $\theta_{\text{const}} = 0$, $\theta_{p;i} \geq 0$ *for all indexes* $(p; i)$, $\theta_{pq;ij} \in \{0, C\}$ *for all indexes* $(pq; ij)$ *where* C *is a positive constant, and* $E(\mathbf{x}) < C$.

Then $\mathbf{x}$ *is stable iff it is a global minimum of* E.

Note that since in case (b) there exists a solution whose cost is smaller than C, pairwise terms act as hard constraints; for any solution $\mathbf{y}$ with $E(\mathbf{y}) \leq E(\mathbf{x})$ there must hold $\theta_{pq}(y_p, y_q) = 0$ for all edges (p, q). A proof of the theorem is given in [20]; it relies on characterization [$\mathcal{P}3$].

In general, however, there may be stable solutions which are not optimal. A simple example is the energy

$$E(x, y, z) = 3|x - y| + 3|y - z| + 2xy + (1 - x)(1 - y).$$

$(1, 1, 1)$ is a stable solution, and $(0, 0, 0)$ is the optimum.

Using theorem 4 it is not difficult to show the following negative result (see [20]):

Theorem 5. *Testing whether a labeling is stable is a co-NP complete problem (under Turing reductions).*

Thus, obtaining good orderings in the QPBOI procedure is a difficult task. Nevertheless, experimental results in Section 4 show that in many cases random permutations do decrease the energy, at least during the first few iterations.

3.3. Summary of Algorithms

There are several options for using the techniques described in Sec. 3. An important question for QPBOI is how to initialize it. Furthermore, QPBOP and QPBOI can be combined. We settled on the following four methods:

QPBOP This technique is designed for obtaining partial optimal solutions. We demonstrate that in many cases it significantly outperforms QPBO, i.e. it produces fewer unlabeled nodes.

BP+I First we run QPBO, then max-product BP (only for unlabeled nodes) and finally improve the solution using QPBOI with random permutations of nodes.

We used a "sequential" schedule of BP as in [14]. Before starting BP, we reparameterized the energy so that $\theta_{pq;00} = \theta_{pq;11}$, and $\theta_{pq;01} = \theta_{pq;10}$ for each edge (p, q). (Note that it does not make sense to start with the reparameterization obtained after running QPBO, since such reparameterization is a fixed point of BP).

QPBOI is stopped when the energy has not improved for 5 iterations, and BP is run for a large number of iterations (here 1000) and the best result is taken.

P+BP+I First we run QPBOP obtaining new energy E' and mapping $f : 2^{\mathcal{V}'} \to 2^{\mathcal{V}}$. Then we apply BP+I for energy E'; this gives solution $\mathbf{y} \in 2^{\mathcal{V}'}$. The output of P+BP+I is the labeling $\mathbf{x} = f(\mathbf{y})$.

P+I In some scenarios an input labeling $\mathbf{x}$ for energy E is available, and it is desirable that the method does not increase it. (An example is the expansion move algorithm [8]; the input labeling $(0, \dots, 0)$ corresponds to the current configuration.) This can be achieved by "tracking" the input solution during QPBOP; details are given in [20].

4. Experiments

In this section we will first investigate the performance of the methods described above with respect to various MRF settings. Then we consider six different applications with non-submodular MRFs and compare them to a standard set of MRF optimization methods. Finally we show the usefulness of our new P+I method within the standard alpha-expansion procedure [8] to optimize a multi-labeled MRF.

4.1. Performance of QPBOP and QPBOI

In the following we measure the improvement of QPBOP over QPBO in terms of additionally labeled pixels. For QPBOI the improvement, with respect to a given reference solution, is measured in terms of lower energy. We are also interested in the runtime overhead for both methods with respect to QPBO. In general, the performance of QPBO (and extended versions) strongly depends on three factors: number of non-submodular terms (ideally, the number of frustrated cycles), connectivity (i.e. average degree of a node), and strength of unary versus pairwise terms. This strength

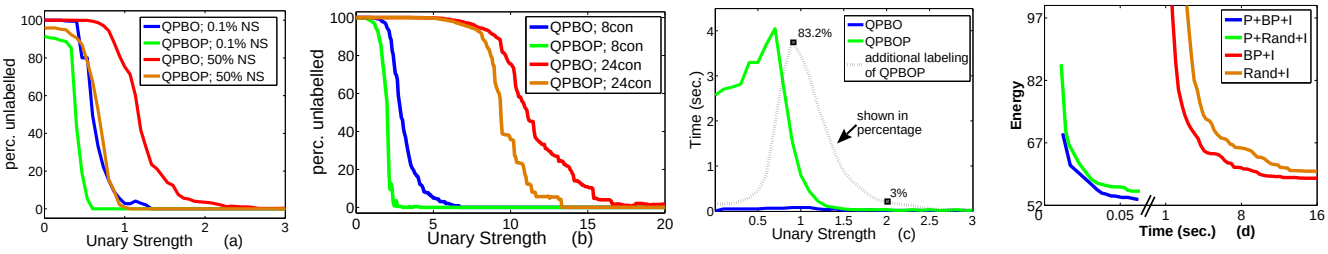

Figure 2. **Performance of QPBOP and QPBOI.** As default we use a 4-connected random graph in normal form of size 100×100 pixels with 50% non-submodular terms and unary strength 0.8. (a,b) compares QPBO and QPBOP with respect to varying non-submodularity (NS) and connectivity (con). The percentage of labeled nodes decreases for: (i) a large number of non-submodular terms, (ii) high connectivity, or (iii) a small unary strength. Note that the improvement of QPBOP over QPBO differs between 0% and 90% of additionally labeled nodes depending on the MRF settings. (c) compares the runtime of QPBO and QPBOP. As to be expected, the runtime increases significantly when QPBOP labels considerably more nodes than QPBO. For practical use an important range is the unary strength between 1.5 and 3, where QPBOP is able to compute (most of the time) the global minimum with only a small runtime overhead. (d) illustrates energy versus runtime for P+BP+I, BP+I, and both methods with a random starting point (P+Rand+I, Rand+I), i.e. a reference solution with random labeling. We see that using BP as the starting point consistently gives a better result. For this particular problem running QPBOP first gives a large improvement in runtime, however, for certain applications the runtime overhead of QPBOP can be considerable and therefore BP+I may sometimes be preferred in practice.

is computed as: $\text{mean}_{p,i}\theta_{p;i}/\text{mean}_{p,q,i,j}\theta_{pq;ij}$, after conversion into normal form (similar to [4]). Fig. 2 shows several variations of these factors.

4.2. Applications

In the following we will compare a standard set of optimization techniques [22] (ICM, BP, TRW-S, Graph Cut, QPBO, and Simulated Annealing) with the new methods on real world applications where binary non-submodular MRFs occur. We are interested to see which method achieves the best performance in a "reasonable" time, i.e. up to several seconds depending on the application. Therefore, we do not plot runtime versus energy but simply report the energy and runtime of the best result for each method. Some notes on the competitive methods: Iterative Conditional Modes (ICM) is run with a random traversal order until convergence. Details of Belief Propagation (BP) are described in Sec. 3.3. Since graph cut (GC) cannot handle non-submodular terms we truncated them as in [21]. Simulated annealing (SA) is capable of producing high quality results with potentially long runtimes. We tweaked the parameters of SA for each *individual* problem, to achieve best results. Finally, TRW-S is guaranteed to give the same answer as QPBO [16], therefore we omit it. (We verified experimentally that the result for labeled nodes is identical. Furthermore, running TRW-S until convergence of the lower bound is much slower than QPBO in practice.)

Table 1 lists the comparison of all methods for one or two examples of each of the six applications.

Diagram Recognition Shape recognition in hand-drawn diagrams is an application where the QPBOP method considerably outperforms standard QPBO. We tested 2700 diagram problems, with an average of 64 nodes and connectivity of 4.1. The MRF model is described in [23]. QPBOP returned the global minimum for **all** problems, whereas

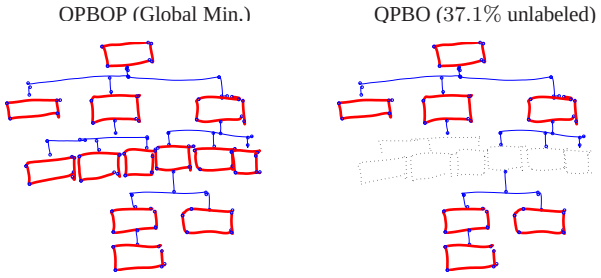

Figure 4. **Diagram recognition.** Given a raw unlabeled hand-drawing, the task is to classify whether each pen stroke is part of a container (red and bold) or a connector (blue). QPBOP finds the global minimum and labels all strokes correctly (left), whereas standard QPBO finds only part of the global solution and leaves 37.1% of the pen strokes unlabeled (dashed in the right diagram).

Figure 5. **New View Synthesis** where QPBO leaves 5731 pixels (3.9%) unlabeled (black), QPBOP finds the global minimum, and graph cut (best of all competitors) has visually noticeable artifacts.

QPBO failed to label all nodes in 97 cases, with between 5% and 56% of nodes unlabeled. One of those challenging examples is shown in fig. 4. Another difficult example is listed in table 1 where the new methods (P+BP+I and BP+I) attain the lowest energy, and QPBOP confirms that this is indeed the global minimum.

Super-resolution and new view synthesis For super-resolution we used the approach of [11] where a node label corresponds to a patch from a reference patch dictionary. The MRF pairwise terms encode the compatibility of overlapping patches of neighboring nodes. The amount

Applications	Sim. An.	ICM	GC	BP	BP+I	P+BP+I	QPBO	QPBOP
Diagram recognition (4.8con)	0 (0.28s)	999 (0s)	119 (0s)	25 (0s)	0 (0s)	0 (0s)	56.3% (0s)	0% (0s) **GM**
New View Synthesis (8con)	- (-s)	999 (0.2s)	2 (0.3s)	18 (0.6s)	0 (2.3s)	0 (1.2s)	3.9%(0.7s)	0% (1.4s) **GM**
Super-resolution (8con)	7 (52s)	68 (0.02s)	999 (0s)	0.03 (0.01s)	0.001 (0.06s)	0 (0.03s)	0.5% (0.016s)	0% (0.047s) **GM**
Image Segm. 9BC + 1 Fgd Pixel (4con)	983 (50s)	999 (0.07s)	0 (28s)	28 (0.2s)	0 (31s)	0 (10.5s)	99.9% (0.08s)	0% (10.5s) **GM**
Image Segm. 9BC; 4RC (4con)	900 (50s)	999 (0.04s)	0 (14s)	24 (0.2s)	0 (3s)	0 (1.48s)	1% (1.46s)	0% (1.48s) **GM**
Texture restoration (15con)	15 (165s)	636 (0.26)	999 (0.05s)	19 (0.18s)	0.01 (2.4s)	0 (14s)	16.5% (1.4s)	0% (14s) **GM**
Deconvolution 3×3 kernel (24con)	0 (0.4s)	14 (0s)	999 (0s)	5 (0.5s)	3.6 (1s)	0 (0.4s)	45% (0.01s)	43% (0.4s)
Deconvolution 5×5 kernel (80con)	0 (1.3s)	6 (0.03s)	999 (0s)	71 (0.9s)	8.1 (31s)	8.1 (31s)	80% (0.1s)	80% (9s)

Table 1. **Comparison table for different applications.** Results are given as: Energy (runtime in seconds). For each problem the energies are Scaled to the range of 0 to 999. The last two columns show the percentage of unlabeled nodes for QPBO and QPBOP, where GM means global minimum. For segmentation BC means boundary constraint and RC region constraint. Also, for segmentation, graph cut was run 2^n (n = number BC) times with flow and search tree recycling to obtain the global minimum. Note, ICM and simulated annealing do not perform well for applications with hard pairwise constraints (infinite links), such as segmentation and new view synthesis.

Figure 3. **Image deconvolution.** Given a blurry and noisy input image with 32 gray-levels, the task is to reconstruct the ground truth. The new P+I method within alpha-expansion improves results compared to graph cut and QPBO-based alpha expansion both in terms of energy and visually. Note, for all alpha expansion based methods the order is crucial. Energies of P+I differed between 26.3 and 27.9 and runtime between 12 and 31 sec, best result shown. Also, to improve runtime of P+I we initialized it with standard graph cut based alpha expansion. BP (E=103), TRW (E=112) and ICM (E=54) perform poorly. Simulated annealing achieved a similar result to P+I in 60 sec.

of non-submodularity can be high, e.g. 45%. The unary terms encode color consistency with the low-resolution image. To make it suitable for our purpose we use two labels and a 5×5 patch size which correspond to an 8-connected MRF (no overlap in the 3×3 center as in [11]). For New View Synthesis as introduced in [10] we may use the same MRF structure, where labels are now color modes derived from depth images (details omitted). We have tested several examples and parameter settings for both applications and may conclude that QPBO typically has a smaller number of unlabeled nodes, e.g. up to 3.9% (in total 5731 pixels) for example in fig. 5. QPBOP is able to find the global minimum most of the time with very little extra runtime (see examples in table 1).

Image segmentation An important issue for interactive image segmentation is the combination of boundary constraints, as in intelligent scissors [17], and region constraints, as in [6]. Here we show that this is possible by including a few non-submodular terms, see fig. 6. We have tested our system for many images, where two examples are listed in table 1. The conclusion is that QPBOP is able to give the global minimum for all examples we have tested, and outperforms QPBO considerably. The speed of QPBOP is affected by the number of brush strokes, the more the faster. All other methods perform very poorly for this application. Note, an alternative approach to compute the global minimum is to run standard graph cut 2^n times where n is the number of unconnected boundary constraints. For the example in fig. 6 where $n = 9$, running 512 graph cuts with

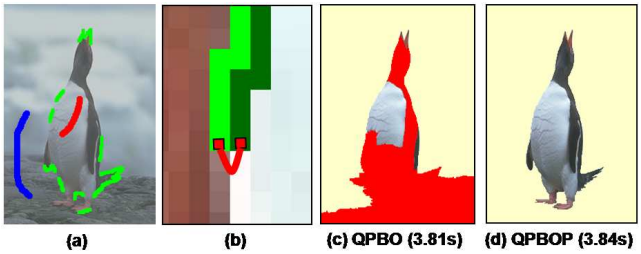

Figure 6. **Interactive Segmentation with Boundary and Region Constraints.** (a) Input image with superimposed user inputs: one inside brush (red), one outside brush (blue) and 9 unconnected boundary constraints (green), bold for better visibility. (b) Zoom into a boundary constraint: Pixels on each line (light and dark green) are constrained to have the same labeling and one extra non-submodular link (red) constrains both lines to have opposite labels. For optimal speed QPBOP first probes pixels at the red links. Note that alternative formulations, e.g. a "fat" intelligent scissors brush, with no specific start and endpoints, are possible and give similar results. The segmentation result of the penguin using QPBO (c) has 26.7% of unlabeled pixels (red), where QPBOP (d) finds the global minimum in about the same time as QPBO.

flow and tree recycling [13] and an optimized order took in the best case 16 sec (84 sec without recycling) which is considerably more than the 3.8 sec of QPBOP.

Parameter learning for binary texture restoration In this application we restore a noisy test image of a texture, based on an MRF model learned from a training image of the same texture type. We used the same learning procedure as described in [15] based on [9] with the only difference that QPBO is replaced by P+BP+I. We have done this for

one Brodatz texture D103 (see [15]) where the test error (averaged over 20 examples) reduces from 25.4 to 25.1 when using P+BP+I instead of QPBO. One example is listed in table 1 where P+BP+I achieved the global minimum. For this application BP+I achieved nearly the same result with a speed-up factor of 6.

Image deconvolution In [19] image deconvolution was formulated as a labeling problem with a pairwise MRF and solved using graph cut based alpha-expansion. Given an $n \times n$ convolution kernel the MRF connectivity is $(2n-1) \times (2n-1) - 1$. Fig. 3 shows an example of reconstructing an input image with 32 different gray-scales and convolution with a 3×3 kernel. To solve this 32 label problem we use alpha-expansion where the P+I method (Sec. 3.3) is used as the binary optimizer. Raj *et al.* [18] also use QPBO-based alpha expansion to reconstruct MR images, although with a sparsely connected MRF.

We have also used the deconvolution MRF with only two labels to reconstruct binary images. Table 1 gives two results with different convolution kernels. The main conclusion is that for highly connected MRFs, e.g. connectivity 80, simulated annealing outperforms all other methods including P+BP+I, and QPBOP performs similarly to QPBO.

5. Conclusions and Future Work

We presented an efficient implementation of the QPBOP method in [5] which is 2-3 orders of magnitude faster than the implementation of [5] on some vision related graphs. We introduced a new technique called QPBOI for optimizing binary non-submodular MRFs, and proved theoretical properties of this method. We have verified experimentally that QPBOP finds the global minimum for many vision applications and that QPBOI nearly always achieves a lower energy with respect to any given reference solution that we have tested. Both techniques are efficient due to graph cut with flow and search tree recycling.

We believe that the main impact of our work lies on the application side, where we plan to further investigate MRFs with high order cliques and multiple labels. Also, most handcrafted MRFs in computer vision are submodular, which is not necessarily true for learned MRFs. Consequently we believe that the demand for efficient, but general, optimizers both during MRF learning and inference, will increase considerably in the future. Finally, we will make the code and energies publicly available, as a step towards a benchmarking system for optimizing challenging, non-submodular MRFs in computer vision, similar to [22].

Acknowledgements We thank the anonymous reviewers for pointing out the recent work [5], Gabriel Tavares for running some tests on our datasets and sharing the implementation of "probing" used in [5], and Olly Woodford for providing us with energies for the new view synthesis problem.

References

[1] J. E. Beasley. Heuristic algorithms for the unconstrained binary quadratic programming problem. Technical report, Management School, Imperial College, London, UK, 1998.

[2] E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123(1-3):155 – 225, 2002.

[3] E. Boros, P. L. Hammer, and X. Sun. Network flows and minimization of quadratic pseudo-Boolean functions. Technical Report RRR 17-1991, RUTCOR, May 1991.

[4] E. Boros, P. L. Hammer, and G. Tavares. Local search heuristics for unconstrained quadratic binary optimization. Technical Report RRR 9-2005, RUTCOR, Feb. 2005.

[5] E. Boros, P. L. Hammer, and G. Tavares. Preprocessing of unconstrained quadratic binary optimization. Technical Report RRR 10-2006, RUTCOR, Apr. 2006.

[6] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, 2001.

[7] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9), Sept. 2004.

[8] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11), Nov. 2001.

[9] D. Cremers and L. Grady. Learning statistical priors for efficient combinatorial optimization via graph cuts, ECCV 06.

[10] A. W. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. In *ICCV*, 2003.

[11] W. Freeman, E. Pasztor, and O. Carmichael. Learning low-level vision. *IJCV*, 40(1):24–57, 2000.

[12] P. L. Hammer, P. Hansen, and B. Simeone. Roof duality, complementation and persistency in quadratic 0-1 optimization. *Mathematicl Programming*, 28:121–155, 1984.

[13] P. Kohli and P. H. S. Torr. Efficiently solving dynamic Markov random fields using graph cuts. In *ICCV*, Oct. 2005.

[14] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 28(10), Oct. 2006.

[15] V. Kolmogorov and C. Rother. Minimizing non-submodular functions with graph cuts - a review. *PAMI*, 2007. To appear. Online version at http://research.microsoft.com/ ˜carrot.

[16] V.Kolmogorov and M.Wainwright. On the optimality of tree-reweighted max-product message passing. In *UAI*, 2005.

[17] E. Mortensen and W. Barrett. Intelligent scissors for image composition. *SIGGRAPH*, 1995.

[18] A. Raj, G. Singh, and R. Zabih. MRF's for MRI's: Bayesian reconstruction of MR images via graph cuts. In *CVPR*, 2006.

[19] A. Raj and R. Zabih. A graph cut algorithm for generalized image deconvolution. In *ICCV*, 2005.

[20] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. Technical Report MSR-TR-2007-46, Microsoft Research, 2007.

[21] C. Rother, S. Kumar, V. Kolmogorov, and A. Blake. Digital tapestry. In *CVPR*, 2005.

[22] R. Szeliski *et al.* A comparative study of energy minimization methods for Markov random fields. In *ECCV*, 2006.

[23] M. Szummer and Y. Qi. Contextual recognition of hand-drawn diagrams with conditional random fields. In *9th Intl. Wkshp. Frontiers in Handwriting Recognition*, 2004.