International Journal of Computer Vision manuscript No.
(will be inserted by the editor)

GRMA: Generalized Range Move Algorithms for the Efficient Optimization of MRFs

Kangwei Liu · Junge Zhang · Peipei Yang · Steve Maybank · Kaiqi Huang[*]

Received: date / Accepted: date

Abstract Markov Random Fields (MRF) have become an important tool for many vision applications, and the optimization of MRFs is a problem of fundamental importance. Recently, Veksler and Kumar et al. proposed the range move algorithms, which are some of the most successful optimizers. Instead of considering only two labels as in previous move-making algorithms, they explore a large search space over a range of labels in each iteration, and significantly outperform previous move-making algorithms. However, two problems have greatly limited the applicability of range move algorithms: 1) They are limited in the energy functions they can handle (i.e., only truncated convex functions); 2) They tend to be very slow compared to other move-making algorithms (e.g., α-expansion and $\alpha\beta$-swap). In this paper, we propose two generalized range move algorithms (GRMA) for the efficient optimization of MRFs. To address the first problem, we extend the GRMAs to more general energy functions by restricting the chosen labels in each move so that the energy function is submodular on the chosen subset. Furthermore, we provide a feasible sufficient condition for choosing these subsets of labels. To address the second problem, we dynamically obtain the iterative moves by solving set cover problems. This greatly reduces the number of moves during the optimization. We also propose a fast graph construction method for the GRMAs. Experiments show that the GRMAs offer a great speedup over previous range move algorithms, while yielding competitive solutions.

Keywords Markov random field · Discrete optimization · Energy minimization · Range move algorithms

[*] Prof. Kaiqi Huang is the correspondence author.
F. Author
Tel.: +123-45-678910
E-mail: fauthor@example.com

1 Introduction

Markov Random Fields (MRF) are an important and widely-used tool in many vision problems such as stereo reconstruction (Szeliski et al. 2008), image restoration (Boykov et al. 2001), segmentation (Boykov and Jolly 2001), medical image analysis (Boykov and Jolly 2000) and image matching (Liu et al. 2014). These problems are solved by finding the maximum a posteriori (MAP) estimate of a labeling, or equivalently obtaining the label assignment that minimizes the MRFs energy. The solution quality and the time complexity of the optimization significantly affect the applicability of MRF models. Therefore, optimizing the MRFs efficiently while ensuring the good quality of the solutions is a problem of fundamental importance.

In the last decades, many optimization approaches have been developed, such as *iterated conditional modes* (ICM) (Besag 1986), *sequential belief propagation* (Tappen and Freeman 2003; Szeliski et al. 2008), and *sequential tree-reweighted message passing* (TRW-S) (Kolmogorov 2006). Recently, graph-cut based algorithms (Kolmogorov and Zabin 2004; Ishikawa 2003; Boykov et al. 2001; Greig et al. 1989; Gridchyn and Kolmogorov 2013; Kumar and Torr 2009; Veksler 2012) have attracted significant attention because of their good optimality properties. Boykov et al. (2001) propose the popular α-*expansion* and $\alpha\beta$-*swap* algorithms, both of which optimize the MRFs by a series of iterative moves. In these two algorithms, each move involves solving the *st*-mincut problem of a corresponding graph. Although α-expansion and $\alpha\beta$-swap have been successfully used in many vision tasks, there is a limitation which has reduced the effectiveness of the algorithms. A choice between only two labels is provided for every vertex in each move. Veksler (Veksler 2007, 2012) effectively solves this problem by developing the so-called *range move algorithms*. In contrast with α-expansion and $\alpha\beta$-swap, they allow every vertex to have a choice of more than two labels. This yields better solutions in practice. In Kumar and Torr (2009) and Kumar et al. (2011), an improved range move algorithm is proposed and it is pointed out that the range move algorithm has the same guarantee as linear programming (LP) relaxation (Chekuri et al. 2004).

However, although the range move algorithms outperform α-expansion and $\alpha\beta$-swap in many cases, there are two major problems which have significantly limited their applicability in practice: (i) they are limited in the types of energy functions they can handle; (ii) the optimization is too slow. As a result, the range move algorithms are not as popular as α-expansion and $\alpha\beta$-swap. In the case of (i), previous range move algorithms (Veksler 2012, 2007; Kumar and Torr 2009; Kumar et al. 2011) are restricted to truncated convex functions. However, there are many more general energy functions, which have been successfully used in different vision problems. The piecewise linear function (Kohli et al. 2013) and the Geman-McClure function (Lempitsky et al. 2010) provide examples. Kumar (2014) modify the range expansion algorithm to general semi-metric pairwise energy functions and propose the rounding-based algorithm. However, the rounding-based algorithm chooses a subset aforehand and estimates a submodular overestimation to replace the original energy function, which leads to a solution that is not optimal for each range move. Veksler (2012) points out that range move algorithms can be extended to more general energies by restricting the set of labels so that the energy on the restricted subset is submodular. Unfortunately, it is still a challenging problem to judge which sets of labels satisfy the submodular condition, given an arbitrary energy function. In the case of (ii), previous range move algorithms execute all possible range moves, which contain many repeated labels and lead to computational inefficiency. As a result, the range move algorithms run much slower than α-expansion and $\alpha\beta$-swap. In theory (Veksler 2012), the larger the set of allowed moves, the better the optimization. However, in practice, we find that almost the same solution can be obtained with a much reduced set of moves. Therefore, we raise the following questions: (i) how to feasibly choose sets of labels that satisfy the submodular condition, given an arbitrary energy function? (ii) Whether we should execute all the possible moves, and if not, which moves are sufficient? (iii) How to schedule the moves to reduce the number of unnecessary moves, while ensuring the quality of the optimization is comparable to that produced by previous range move algorithms?

To solve the above problems, we propose two *generalized range move algorithms* (GRMA) that we call the *generalized range swap* (GRSA) and *generalized range expansion* (GREA) algorithm, respectively. Firstly, we extend the range

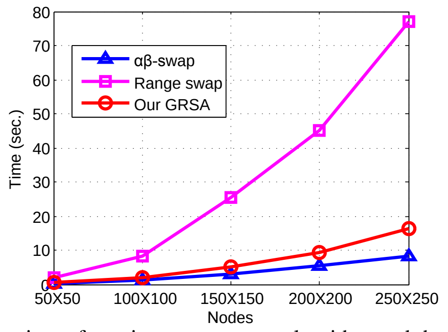


Fig. 1 The run time of previous range swap algorithm and the GRSA on MRFs with different sizes. The results are average run times tested on 50 MRFs with a truncated convex energy function.

move methods to more general functions[1] by requiring that the chosen set of labels in every move is submodular. Different from the method of Kumar (2014), whose strategy is to estimate a submodular overestimation and compute an approximate solution to the original range move, the GRMAs choose the subsets of the labels on which the submodular condition is satisfied on the original energy function. More importantly, we provide a sufficient condition for choosing sets of labels that satisfy the submodular condition. Secondly, we give conditions on the iterative moves to ensure that GRMAs produce a good estimate of the optimal solution. Both GRSA and GREA require that every vertex has the opportunity to change its current label to any other label during the optimization. This requirement guarantees that GRSA and GREA obtain at least as good solutions as $\alpha\beta$-swap or α-expansion. Then, to reduce the run time of the range move algorithms, we dynamically obtain a series of moves meeting the requirement by solving a *set cover problem*. Finally, we develop an improved graph construction method for the GRMAs. The proposed graph construction allows the GRMAs to handle more general energy functions than the truncated convex functions. Moreover, the new graph construction not only runs faster than previous graph constructions (Kumar et al. 2011), but also yields better solutions. We demonstrate the effectiveness of GRMAs on both synthetic data and real vision problems. Experimental results show that both GRSA and GREA greatly reduce the run time compared to previous range move algorithms (Fig. 1), while they obtain competitive solutions.

There are three main contributions in this paper:

– The GRMAs are applicable to arbitrary semimetric pairwise functions. We restrict the chosen labels to be a submodular set, and propose a method to choose the submodular sets feasibly.
– We formulate the iterative optimization using a solution to a set cover problem. This formulation avoids a large number of unnecessary moves and offers a great speedup

[1] In this work, we consider the optimization of arbitrary semimetric energy functions. Here, "semimetric" means that the pairwise function should satisfy $\theta(\alpha,\beta)=0 \Leftrightarrow \alpha=\beta$ and $\theta(\alpha,\beta)=\theta(\beta,\alpha)\geq 0$.

over previous algorithms. Furthermore, we show that although a much fewer number of moves are considered in our GRMAs, the methods do not lose much accuracy in practice.

– We develop an improved graph construction method for the graph cut technique. Compared to previous graph constructions (Ishikawa 2003; Kumar et al. 2011), there are two advantages in the new graph construction: (i) The graph construction needs fewer edges to model the pairwise potentials in each move, and thus the st-mincut problem can be solved faster than that in previous graph constructions (Ishikawa 2003; Kumar et al. 2011). (ii) The improved construction in the GREA guarantees better solutions than Kumar et al. (2011) both theoretically and practically.

The GRSA is a generalization of several graph-cut based algorithms including $\alpha\beta$-swap, range swap algorithm and the method of Ishikawa (2003), while the GREA is a generalization of α-expansion, the range expansion algorithm and the method of Ishikawa (2003). This provides new views of the relationships among the graph-cut based algorithms.

The preliminary version of the GRSA algorithm is proposed in Liu et al. (2015). This journal version significantly extends Liu et al. (2015) both theoretically and empirically. The most important extensions are the GREA algorithm and the improved graph construction method. Besides, this journal version reviews the related methods and summarizes their strengths and weaknesses in more details, enhances the theoretical analysis of the GRMA algorithms and presents more experimental analysis to demonstrate the effectiveness of the proposed methods.

The rest of the paper is organized as follows. We introduce the background and related work in Sec. 2. The generalized range swap algorithm is proposed and explained in Sec. 3, while the generalized range expansion algorithm is developed in Sec. 4. Then, the graph constructions for both GRSA and GREA are introduced in Sec. 5. The experimental results are demonstrated in Sec. 6, and finally, the conclusions are made in Sec. 7.

2 Background and related work

2.1 Preliminaries of MRFs

Many vision problems can be naturally formulated in terms of the *maximum a posteriori* (MAP) inference of an MRF. The MRF is defined as an undirected graph $\mathcal{G} = (\mathcal{P}, \mathcal{E})$, where $\mathcal{P}$ is the set of vertices, and $\mathcal{E}$ is the set of edges connecting neighboring vertices. Given an MRF, a *labeling* $f = \{f_p | p \in \mathcal{P}\}$ is the label assignment of all the vertices $p \in \mathcal{P}$. The probability of the labeling is given by the Gibbs distribution: $\mathbf{P}(f|\mathbf{D}) = \exp(-E(f))/Z$, where $\mathbf{D}$ is the observed data and Z is the partition function that depends on $\mathbf{D}$. The MAP estimation of the labeling can be solved by minimizing the Gibbs energy, which is typically given as follows:

$$E(f) = \sum_{p \in \mathcal{P}} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{E}} \theta_{pq}(f_p, f_q) \tag{1}$$

where f_p, f_q are in the set $\mathcal{L}$ of labels, and θ_p, θ_{pq} denote the unary and pairwise potential respectively. The neighborhood structure (i.e., elements of $\mathcal{E}$) are often derived from the spatial structure of an image. The objective of the optimization problem is to obtain the labeling f^*, which minimizes the sum of the unary terms θ_p and the pairwise terms θ_{pq}.

In the energy $E(f)$, the unary term θ_p encourages the consistency between the assigned label f_p and the observed data. In contrast, the pairwise term θ_{pq} encourages the labeling f to vary smoothly on neighboring vertices. The choice of θ_{pq} is of critical importance to the solution of different tasks. In some approaches (Poggio et al. 1989), θ_{pq} is chosen to be a convex function (e.g., linear or quadratic function), which makes the labeling smooth everywhere. However, convex functions do not perform well on the boundaries of objects, because the sharp label changes on the boundaries will increase the value of the energy function. To avoid the problem of overpenalizing, a large number of non-convex energies have been developed in the literature, e.g., truncated convex function (Boykov et al. 2001), piecewise linear functions (Kohli et al. 2013) and Geman-McClure function (Lempitsky et al. 2010).

2.2 Graph cut based optimization

In recent years, graph cut has been a standard technique for the optimization of MRFs. The GRMAs are also based on graph cut, thus, we give a brief review of graph cut based algorithms in this section.

The primary idea of the graph cut based algorithms is to construct a special graph $\mathcal{G}_C$ with a source vertex s and a sink vertex t, such that there is a one-to-one correspondence between the st-cut of $\mathcal{G}_C$ and of a labeling f. The cost of the st-cut of $\mathcal{G}_C$ is exactly equal to the value of energy $E(f)$. Thus, the minimization of $E(f)$ can be obtained by finding the st-mincut of the graph $\mathcal{G}_C$. Although the st-mincut can be found efficiently in polynomial time by max-flow algorithms, the weights of the edges in the st-mincut graph are required to be non-negative. Unfortunately, there are many energy functions for which such a corresponding graph cannot be exactly constructed. As a result, most algorithms optimize $E(f)$ by a series of moves, each of which only considers a subset of the labels. According to the number of labels considered in each move, the move-making algorithms can be divided into three categories:

The $\alpha\beta$-swap and α-expansion algorithms The α-expansion and $\alpha\beta$-swap methods (Boykov et al. 2001) are among the most popular graph cut based algorithms. Both of them optimize the energy by a series of iterations. In each iteration, the algorithms provide every vertex a choice either to keep the current label or to obtain a new label. Every such move leads to a lower energy. The algorithms terminate when no move can be found that lowers the energy $E(f)$. Unlike traditional standard move algorithms (i.e., ICM), which only allow one vertex to change its label at a time, both α-expansion and $\alpha\beta$-swap allow a large number of vertices to change their labels simultaneously. Due to their good optimality properties, both algorithms have been successfully applied in many vision tasks (Szeliski et al. 2008; Kappes et al. 2013), and there are a large number of improved move-making algorithms based on them. For example, Lempitsky et al. (2007, 2010) extend their application to arbitrary pairwise energy functions using QPBO (Kolmogorov and Rother 2007; Rother et al. 2007) to construct the graph. Kumar and Koller (2009) proposes an accurate hierarchical move making strategy and obtains the same guarantees as the standard linear programming relaxation. Batra and Kohli (2011) and Gould et al. (2009) improve the efficiency of the $\alpha\beta$-swap and α-expansion algorithms by reducing the label space to be searched in each move. However, all of these algorithms only provide each vertex a choice of two labels in each move.

The range move algorithms To obtain better solutions with the graph-cut techniques, Veksler (Veksler 2007, 2012) and Kumar et al. (Kumar and Torr 2009; Kumar et al. 2011) develop the range move algorithms for truncated convex functions (e.g., $\theta(f_p, f_q) = \min\{|f_p - f_q|, T\}$). The range move algorithms break the limitation of previous move making algorithms in which only two labels are considered in every move. In the algorithms, every iteration considers a consecutive label subset $\mathcal{L}_{\alpha\beta} = \{\alpha, \alpha+1, \cdots, \beta\}$ by imposing the restriction $|\alpha - \beta| = T$. However, there is a problem in the iterative moves of the previous range move algorithms: they execute the set of all possible range moves, and this leads to computational inefficiency. Although the range move algorithms significantly outperform $\alpha\beta$-swap and α-expansion on the quality of solutions, they perform much slower than $\alpha\beta$-swap and α-expansion. In contrast, the GRMAs do not suffer from this inefficiency, because the moves are based on a solution to a set cover problem. This greatly reduces the number of moves in the iterations.

The case of global optimization Although the optimization of MRFs is usually NP-hard, there are a few energy functions for which the exact solution can be obtained by considering all the labels in one st-mincut. Greig et al. (1989) first develop an exact method for the optimization of binary labeled MRFs. Ishikawa (2003) generalized the graph of Greig et al. (1989) to the optimization of multi-label MRFs. However, the pairwise potential θ_{pq} is restricted to be convex[2]. More generally, Schlesinger (Schlesinger and Flach 2006) points out that all the energies with *submodular* pairwise functions can be exactly minimized by the graph cut techniques. However, neither the convex nor submodular functions are widely used in practice because they cannot preserve the discontinuities at the boundaries (Boykov et al. 2001). Our GRMAs are based on the graph construction of (Schlesinger and Flach 2006). However, rather than requiring the energy function to be convex or submodular on the whole label set, GRMAs only require that the energy function be submodular on certain subsets of the labels.

2.3 Set cover problem

The set cover problem is an important question in computer science and complexity theory. Given an *universe* U of m elements and a collection of sets $\mathcal{S} = \{S_1, ..., S_k\}$ where $S_i \subseteq U$ and $\bigcup_{i=\in\{1,\cdots,k\}} S_i = U$, a *set cover* is a subcollection of the sets in $\mathcal{S}$ that covers all the elements in U. The goal of the set cover problem is to find the set cover with the smallest possible number of sets. In the weighted set cover problem, a cost function $c : \mathcal{S} \to \mathbf{R}$ is specified for each set $S_i \in \mathcal{S}$. Then, the objective is to find the set cover $\mathcal{S}' \subseteq \mathcal{S}$ that minimizes the costs $\sum_{S_i \in \mathcal{S}'} c(S_i)$. It is well known that the set cover problem is NP hard. Fortunately, the greedy algorithm (Slavłk 1996) solves the set cover problem approximately in polynomial time.

3 Generalized range swap algorithm

The optimization of MRFs is a NP-hard problem, while both the solution quality and time complexity are vitally important to applications. This forces the development of efficient approximation algorithms. Our algorithms generate a local minimum with respect to two types of moves: *range swap* and *range expansion*. The algorithms start from an initial labeling, and optimize $E(f)$ by making a series of moves, each of which refers to an st-mincut problem. They halt when no move can be found to decrease $E(f)$. In contrast to the α-expansion and $\alpha\beta$-swap algorithms, both the range swap and range expansion moves provide every vertex with a choice from a range of labels instead of only two labels.

In this section, we develop the so-called generalized range swap algorithm (GRSA), which is a generalization of $\alpha\beta$-swap, range swap and the method of Ishikawa (2003). Firstly, the notion of the generalized range swap move is

[2] A function $g(\cdot)$ is convex if it satisfies $g(x+1) - 2g(x) + g(x-1) \geq 0$ for any integer x. Note that convex is a special case of submodular.

introduced in Sec. 3.1. Every range swap move is executed on a subset of labels satisfying the submodular condition. However, it is still a hard problem to find subsets of labels satisfying the submodular condition. In Sec. 3.2, we propose a new sufficient condition for submodularity, and show how to choose the labels flexibly given this condition. Given a submodular set, the range swap move relies on solving the st-mincut of a graph which models the unary and pairwise potentials exactly. In Sec. 3.3, we focus on designing an iterative process to reduce the run time of GRSA, and obtain a series of moves by solving a set cover problem.

3.1 Generalized range swap move

Let $\mathcal{L} = \{0, \cdots, n\}$ be the label set, and $\mathcal{L}_s = \{l_1, \cdots, l_m\}$ ($l_i < l_{i+1}$) be a subset chosen from $\mathcal{L}$. Note that $\mathcal{L}_s$ is an arbitrary subset of $\mathcal{L}$, and is not necessary to be a consecutive sequence as in previous algorithms (Ishikawa 2003; Veksler 2007; Kumar and Torr 2009). Let $\mathcal{P}_l = \{p \in \mathcal{P} | f_p = l\}$ be the set of vertices assigned label l, and $\mathcal{P}_\mathcal{S} = \{p \in \mathcal{P} | f_p \in \mathcal{L}_s\}$ denote the set of vertices whose labels belong to $\mathcal{L}_s$. Then, a move from f to f' is called a *range swap move* (RSM) on $\mathcal{L}_s$, if $\mathcal{P}'_\mathcal{S} = \mathcal{P}_\mathcal{S}$, and $\mathcal{P}'_l = \mathcal{P}_l$ for any label $l \notin \mathcal{L}_s$. In other words, a RSM allows the vertices belonging to $\mathcal{P}_\mathcal{S}$ to swap their labels in $\mathcal{L}_s$. Each range swap move obtains the new labeling f' by minimizing the following energy:

$$E_s(f') = \sum_{p \in \mathcal{P}_\mathcal{S}} \theta_p(f'_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_\mathcal{S} \neq \varnothing} \theta_{pq}(f'_p, f'_q) \quad (2)$$

such that for any vertex $p \in \mathcal{P}$, we have

$$f'_p \in \mathcal{L}_s, \text{ if } p \in \mathcal{P}_\mathcal{S} \ ; \ \ f'_p = f_p, \text{ if } p \notin \mathcal{P}_\mathcal{S},$$

where f_p is the current label of vertex p.

Naturally, we have $E(f') = E_s(f') + E_{\hat{s}}(f')$, where

$$E_{\hat{s}}(f') = \sum_{p \notin \mathcal{P}_\mathcal{S}} \theta_p(f'_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_\mathcal{S} = \varnothing} \theta_{pq}(f'_p, f'_q).$$

With the minimization of $E_s(f')$, the RSM on $\mathcal{L}_s$ effectively decreases $E(f)$, since the move will not change the value of $E_{\hat{s}}(f)$. Theoretically, the RSM on $\mathcal{L}_s$ will lead to a better solution, if there are more labels considered in $\mathcal{L}_s$ (meanwhile, more vertices will be in $\mathcal{P}_\mathcal{S}$) (Veksler 2012). However, $\mathcal{L}_s$ cannot be chosen arbitrarily. It should satisfy the following submodular condition (Schlesinger and Flach 2006) to guarantee the optimal RSM can be obtained:

Definition 1 *Given a pairwise potential $\theta(\alpha, \beta)$, we call $\mathcal{L}_s$ a submodular set of labels, if it satisfies*

$$\theta(l_{i+1}, l_j) - \theta(l_{i+1}, l_{j+1}) - \theta(l_i, l_j) + \theta(l_i, l_{j+1}) \geq 0 \quad (3)$$

for any pair of labels l_i, $l_j \in \mathcal{L}_s (1 \leq i, j < m)$.

The optimal RSM on the submodular set $\mathcal{L}_s$ can be achieved by solving the st-mincut of a special graph (see Sec. 5.1).

3.2 Candidate submodular sets

Unfortunately, given an arbitrary energy function, it is still a challenging problem to obtain the submodular sets with these inequalities (3). Therefore, we propose a sufficient condition for submodularity, which allows the sets of labels satisfying (3) to be chosen feasibly in practice. The sufficient condition is given by the following theorem:

Theorem 1 *Given a pairwise function $\theta(\alpha, \beta) = g(x)$ ($x = |\alpha - \beta|$), assume there is an interval*[3] *$X_s = [a, b]$ ($0 \leq a < b$) satisfying: (i) $g(x)$ is convex on $[a, b]$, and (ii) $a \cdot (g(a+1) - g(a)) \geq g(a) - g(0) \geq 0$. Then $\mathcal{L}_s = \{l_1, \cdots, l_m\}$ is a submodular subset, if $|l_i - l_j| \in [a, b]$ for any pair of labels l_i, l_j such that $l_i \neq l_j$ and $l_i, l_j \in \mathcal{L}_s$ (Proof in Appendix A).*

In the follows, we focus on explaining the above theorem. For brevity, we call the interval X_s satisfying the conditions (i) and (ii) in Theorem 1 a *candidate interval*. It is obvious that any convex interval $[0, b]$ is a candidate interval, because $a \cdot (g(a+1) - g(a)) = g(a) - g(0)$ when $a = 0$.

General energy functions Most functions successfully applied in vision problems are neither linear functions nor convex functions, such as truncated convex functions and piecewise linear functions. Although these functions are neither convex nor submodular on the whole domain, there are usually some convex candidate intervals. Theorem 1 implies that submodular subsets $\mathcal{L}_s$ of labels can be obtained by restricting the differences between each pair of labels such that the differences belong to the same candidate interval. To explain this clearly, we use the example of the piecewise linear function shown in Fig. 2. There are two candidate intervals: $[0, 3]$ and $[5, 12]$ for this pairwise function. As in previous range move algorithms (Veksler 2007, 2012), we obtain a series of submodular sets $\{\alpha, \alpha+1, \alpha+2, \alpha+3\}$ where $\mathcal{L} = \{0, \cdots, n\}$, $0 \leq \alpha \leq n-3$ with the first candidate interval $[0, 3]$. Meanwhile, it can be seen that the subsets $\{\alpha, \alpha+2, \alpha+3\}$ and $\{\alpha, \alpha+1, \alpha+3\}$ are also submodular sets, because $|l_i - l_j| \in [0, 3]$ for any pair of labels. Furthermore, we can also obtain the submodular sets: $\{\alpha, \alpha+5, \alpha+10\}, \{\alpha, \alpha+6, \alpha+11\}, \cdots$ with the second candidate interval $[5, 12]$. More generally, we give the following corollary which is equivalent to Theorem 1:

Corollary 1 (Theorem 1) *Assuming the interval $[a, b]$ is a candidate interval, then $\{\alpha, \alpha+x_1, \alpha+x_1+x_2, \cdots, \alpha+x_1+\cdots+x_m\} \subseteq \mathcal{L}$ is a submodular set for any $\alpha \geq 0$, if $x_1, \cdots, x_m \in [a, b]$ and $x_1 + \cdots + x_m \leq b$ (Proof in Appendix A.3).*

[3] Here, the interval $[a, b]$ denotes the set of integers $\{x | a \leq x \leq b\}$.

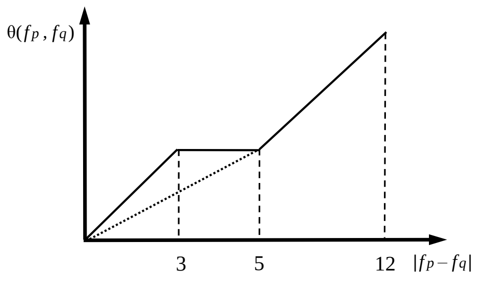


Fig. 2 An example of piecewise linear function $g(x)$. The function is locally convex on the intervals $[0, 3]$ and $[5, 12]$. Here, we assume that the function satisfies $\frac{g(6)-g(5)}{6-5} \geq \frac{g(5)-g(0)}{5-0}$. Thus, both $[0, 3]$ and $[5, 12]$ are candidate intervals, and the submodular sets can be obtained on these two candidate intervals.

Concave functions If the pairwise function is a concave function (e.g., $g(x) = \sqrt{x}$), then no convex interval can be found. It can be easily proved that in this case there is no submodular set that contains more than two labels. Thus, only two labels can be considered in every move. Therefore, GRSA is equivalent to the $\alpha\beta$ swap algorithm, when the energy function is concave.

Convex functions If the pairwise function is a convex function (e.g., $g(x) = x$), the domain $[0, n]$, where n is the number of labels, is a candidate interval. Therefore, the whole set $\mathcal{L}$ of labels is a submodular set. The optimal solution can be achieved in one move, and thus GRSA is equivalent to the global method of Ishikawa (Ishikawa 2003) in this case.

GRSA can be viewed as a generalization of several graph cut based algorithms with different energy functions. Given an arbitrary function, a series of candidate submodular sets can be obtained with Theorem 1 or Corollary 1. The range swap move executed on any of these submodular sets can be exactly solved by computing an st-mincut problem.

3.3 The iterative optimization

Before proposing the iterative optimization in GRSA, we review the iterative process in $\alpha\beta$-swap and previous range swap algorithms, and then give a condition on the moves to ensure that GRSA produces a good estimate of the optimal solution.

$\alpha\beta$-swap In the $\alpha\beta$-swap algorithm, the requirement of the swap moves is that each pair of labels should be visited in each *cycle*[4] of iterations. This guarantees that every vertex has a chance to swap its current label f_p with any other label in $\mathcal{L}$.

[4] In $\alpha\beta$-swap, we call these iteration moves considering all the pairs of labels once as a “cycle”. An $\alpha\beta$-swap algorithm usually takes several cycles to converge (Boykov et al. 2001).

Previous range swap In previous range swap algorithms, the moves are executed on all the subsets $\mathcal{L}_{\alpha\beta} = \{\alpha, \alpha+1, \cdots, \beta\}$, where $|\alpha - \beta| = T$, and T is the truncation factor in a truncated convex function (*e.g.* $\theta = \min\{|f_p - f_q|, T\}$). However, there are two problems on these moves. (i) There are many repeated labels. For example, in the two moves $\{\alpha, \alpha+1, \cdots, \beta\}$ and $\{\alpha+1, \alpha+2, \cdots, \beta+1\}$, all the label except α and β are repeated. Thus, these moves cost much time, and cannot efficiently decrease $E(f)$. This is why previous range swap algorithms run much slower than $\alpha\beta$-swap. The second problem (ii) is that some important moves are missing in these iterations, *i.e.*, the pairs of labels $\{\alpha, \alpha+T'\}$ where $T' > T$ are not considered in the moves. For a vertex p, whose current label is α and real label is $\alpha + T'$, there is unfortunately no move from α to $\alpha + T'$. Thus, previous range swap algorithms sometimes need a careful initialization to obtain good solutions.

The GRSA Given an arbitrary energy function, we usually can obtain a large number of submodular sets, each of which corresponds to one possible range swap move as described in Sec. 3.2. However, it is time-consuming and unnecessary to perform all possible sets of range moves. In practice, the quality of solutions is assured if *any pair of labels is considered once in any given cycle of iterative moves*, *i.e.*, every vertex should have chance to swap its current label with any other label. This is the same requirement as in $\alpha\beta$-swap.

The problem is how to choose a series of moves (*i.e.*, submodular sets) in each cycle, such that (i) these submodular sets cover all pairs of labels; (ii) submodular sets containing more labels are chosen preferentially, and (iii) there are as few repeated labels as possible in these submodular sets. This problem is naturally formulated as a *set cover problem* (SCP) (Feige 1998).

In the GRSA, $\mathcal{L} = \{0, \cdots, n\}$ is the set of labels, and let $\{\mathcal{L}_1, \mathcal{L}_2, \cdots, \mathcal{L}_k\}$ be the series of submodular sets. We define $\mathcal{C}(\mathcal{L}) = \{(0,1), (0,2), \cdots, (n-1, n)\}$ to be the set containing all the pairs of labels in $\mathcal{L}$. In the set cover formulation, the universe $U = \mathcal{C}(\mathcal{L})$, and the collection of subsets is defined by $S_i = \mathcal{C}(\mathcal{L}_i)$. Therefore, the moves are obtained by finding the set cover $\mathcal{S}' \subseteq \mathcal{S}$ in the following set cover problem:

$$\min_{\mathcal{S}' \subseteq \mathcal{S}} \sum_{S_i \in \mathcal{S}'} c(S_i) \qquad \text{s.t.} \bigcup_{S_i \in \mathcal{S}'} S_i = U \,. \tag{4}$$

where $c : \mathcal{S} \to \mathbf{R}$ is the cost function specified for each set $S_i \in \mathcal{S}$.

Although the SCP is an NP hard problem, fortunately, the greedy algorithm (Slavłk 1996) can obtain an approximate solution in polynomial time. Algorithm 1 describes the iterative process of the GRSA, where the moves are chosen dynamically by solving the SCP with the greedy algorithm.

Algorithm 1 The Generalized Range Swap Algorithm

```
Input:
1: The label set $\mathcal{L} = \{0, \cdots, n\}$, and the pairwise function $\theta(\alpha, \beta) = g(x)$ $(x = |\alpha - \beta|)$.
Initialization:
2: Find the series of submodular sets $\mathcal{L}_i$ with the form described in Corollary 1.
3: Define the collection of sets $\mathcal{S} = \{S_1, ..., S_k\}$ where $S_i = \mathcal{C}(\mathcal{L}_i)$ denotes the set containing all the pairs of labels in $\mathcal{L}_i$.
4: Initialize the labeling $f$.
Iteration:
5: repeat
6:     Initialize $U = \mathcal{C}(\mathcal{L})$, $S_c \leftarrow \varnothing$.
7:     while $S_c \neq U$ do
8:         Choose $S_i \in \mathcal{S}$, which minimizes the cost per element $\frac{c(S_i)}{|S_c \cup S_i| - |S_c|}$.
9:         Set $S_c := S_i \cup S_c$ and $\mathcal{L}_s := \mathcal{L}_i$ where $S_i = \mathcal{C}(\mathcal{L}_i)$.
10:        Obtain the new labeling $f' = \arg\min E(f)$ within the range swap move on $\mathcal{L}_s$.
11:        If $E(f') < E(f)$, set $f := f'$.
12:    end while
13: until No moves can be found to decrease $E(f)$.
Output:
14: Return the labeling $f$.
```

In the algorithm, steps 6-11 are a cycle of iterative moves. In each iteration, the following set is chosen in step 7:

$$S_i = \arg\min_{S_i \in \mathcal{S}} \frac{c(S_i)}{|S_c \cup S_i| - |S_c|} \tag{5}$$

where S_c denotes the set of elements which have been chosen in the greedy algorithm, and $|S_c \cup S_i| - |S_c|$ denotes the number of elements in S_i but not in S_c. The greedy algorithm always chooses the set which minimizes the cost per increased element.

Using the SCP, we can design different iterative processes by assigning different costs to the submodular sets. In this paper, we set[5] : $c(S_i) = 1$, if $S_i \cap S_c = \varnothing$; ∞, otherwise.

This ensures that at every iterative move in step 7 a set is chosen such that: (i) there is no repeated element in the sets which have been chosen in previous moves; (ii) the chosen set contains a maximal number of labels among the sets satisfying condition (i). In the case of a truncated convex function with truncation factor T, the GRSA can execute the moves on the following series of submodular sets: $\hat{\mathcal{L}} = \{\{0, \cdots, T\}, \{T, \cdots, 2T\}, \cdots, \{mT, \cdots, n\}\}$ and all the pairs of labels that are not visited in $\bigcup_{\mathcal{L}_i \in \hat{\mathcal{L}}} \mathcal{C}(\mathcal{L}_i)$.

[5] We chose this cost function for simplicity, but a better iterative process may be developed with other choices, for example $c(S_i) = 1 + |S_i|$, because a small number of repeated labels may lead to a better solution without a significant increasing in the run time. Another choice is to set $c(S_i)$ to be the estimate of improvement in energy as (Batra and Kohli 2011). However, the experiments show that GRSA yields promising results with the simple choice of cost function made here.

Theoretical analysis The GRSA also generalizes $\alpha\beta$-swap and the method of Ishikawa (2003) in terms of the iterative process in the optimization. With the set cover formulation, the optimization process is exactly an $\alpha\beta$-swap or the method of Ishikawa (2003), when the pairwise potential is concave or convex, respectively. The GRSA has the following properties:

Proposition 1 *Let $\mathcal{L}_1, \cdots, \mathcal{L}_k$ be a set of range swap moves, which cover all pairs of labels $l_i, l_j \in \mathcal{L}$. Let $\hat{f}$ be a local minimum[6] obtained by these moves. Then, $\hat{f}$ is also a local minimum for $\alpha\beta$-swap (Proof in Appendix B).*

Proposition 2 *Let $\hat{f}$ be a local minimum obtained by $\alpha\beta$-swap. With the initial labeling $\hat{f}$, the range swap moves on $\mathcal{L}' = \{\mathcal{L}_1, \cdots, \mathcal{L}_k\}$ yield a local minimum $f^\dagger$ such that $E(f^\dagger) < E(\hat{f})$, unless the labeling $\hat{f}$ exactly optimizes the energy:*

$$E_s(f) = \sum_{p \in \mathcal{P}_{\mathcal{L}_i}} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_{\mathcal{L}_i} \neq \emptyset} \theta_{pq}(f_p, f_q)$$

for each $\mathcal{L}_i \subseteq \mathcal{L}'$ (Proof in Appendix C).

Proposition 1 implies that if the GRSA obtains a local minimum $\hat{f}$, it is also a local minimum for the $\alpha\beta$ swap and $E(\hat{f})$ cannot be decreased by any move in the $\alpha\beta$-swap. In contrast, Proposition 2 implies that if we obtain a local minimum $\hat{f}$ by $\alpha\beta$-swap, $E(\hat{f})$ still can be decreased by the moves in the GRSA. This shows that we can first get an initial labeling $\hat{f}$ by $\alpha\beta$-swap (or some other fast algorithm, *e.g.* α-expansion), and then achieve a better solution by applying GRSA to the labeling $\hat{f}$.

4 Generalized range expansion algorithm

The generalized range expansion algorithm (GREA) generalizes several graph-cut based algorithms including α-expansion, range expansion and the method of Ishikawa (2003). Unlike the GRSA which only considers the vertices $p \in \mathcal{P}$ whose current label f_p lies in $\mathcal{L}_s$, the GREA allows all the vertices to change their current labels simultaneously in one move.

We explain the idea of the generalized range expansion move (GREM) in Sec. 4.1. As for the GRSA, the chosen label set in the GREA should also satisfy the submodular conditions (3). In Sec. 4.2, we show how the iterative moves are obtained by solving set cover problems. This effectively reduces the number of unnecessary moves in the GREA.

4.1 Generalized Range Expansion Move

Let $\mathcal{L}_s = \{l_1, \cdots, l_m\}$ $(l_i < l_{i+1})$ be a subset chosen from the label set $\mathcal{L}$. Let $\mathcal{P}_{\mathcal{S}} = \{p \in \mathcal{P} | f_p \in \mathcal{L}_s\}$ be the

[6] If $\hat{f}$ is a local minimum, it means that $E(\hat{f})$ cannot be decreased by any of the moves $\mathcal{L}_i$.

set of vertices whose labels belong to $\mathcal{L}_s$. Then, a move from f to f' is called a *range expansion move* (REM), if $\mathcal{P}'_{\mathcal{S}} \supseteq \mathcal{P}_{\mathcal{S}}$, and $\mathcal{P}'_l \subseteq \mathcal{P}_l$ for any label $l \notin \mathcal{P}_{\mathcal{S}}$. In other words, a range expansion move allows any vertex $p \in \mathcal{P}$ to change its current label to any label $f'_p \in \mathcal{L}_s$. Each range expansion move obtains the new labeling f' by minimizing the following energy:

$$E_e(f') = \sum_{p \in \mathcal{P}} \theta_p(f'_p) + \sum_{(p,q) \in \mathcal{E}} \theta_{pq}(f'_p, f'_q) \tag{6}$$

such that for any vertex $p \in \mathcal{P}$, $f'_p \in \mathcal{L}_s$ or $f'_p = f_p$, where f_p is the current label of vertex p.

The REM effectively decreases the energy $E(f)$ by minimizing $E_e(f')$. The set $\mathcal{L}_s$ of labels in the GREA should also satisfy the submodular condition.

For general energy functions, the submodular set $\mathcal{L}_s$ in each move could be obtained according to Theorem 1 or Corollary 1. The energy functions may not satisfy submodularity (Schlesinger and Flach 2006), but there are usually subsets of labels that satisfy Theorem 1.

For a convex function, the whole set $\mathcal{L}$ of labels is a submodular set, and the optimization can be achieved by one range expansion move on $\mathcal{L}$. In this case, the range expansion algorithm is equivalent to the method of Ishikawa (2003). In contrast, if only one label is considered in each move, the range expansion algorithm reduces to α-expansion. Thus, the GREA can be viewed as the generalization of α-expansion, range expansion and the method of Ishikawa (2003).

4.2 Iterative Optimization

Before explaining the iterative moves of the GREA, we first review α-expansion and previous algorithms for range expansion.

α-expansion In each cycle of iterations, all the labels in $\mathcal{L}$ are visited once. This requirement guarantees that every vertex has the opportunity to change its current label to any one of the other labels in $\mathcal{L}$. Thus it ensures the solution quality.

Range expansion Previous range expansion algorithms execute all possible range expansion moves on the subsets $\mathcal{L}_{\alpha\beta} = \{\alpha, \alpha+1, \cdots, \beta\}$, where $|\alpha - \beta| = T$, and T is the truncation factor for a convex function. The problem is that there are many repeated labels in these moves, and thus, they cannot decrease the energy $E(f)$ efficiently. As a result, previous range expansion algorithms run much slower than α-expansion.

Algorithm 2 The Generalized Range Expansion Algorithm

```
Input:
 1: The label set $\mathcal{L} = \{0, \cdots, n\}$, and the pairwise function
    $\theta(\alpha, \beta) = g(x)$ $(x = |\alpha - \beta|)$.
Initialization:
 2: Find the series of submodular sets $\mathcal{L}_i$ with the form described in
    Corollary 1.
 3: Get the collection of sets $\mathcal{S} = \{S_1, ..., S_k\}$ where $S_i = \mathcal{L}_i$.
 4: Initialize the labeling $f$.
Iteration:
 5: repeat
 6:   Initialize $U = \mathcal{L}$, $S_c \leftarrow \varnothing$.
 7:   while $S_c \neq U$ do
 8:     Choose $S_i \in \mathcal{S}$, which minimizes the cost per element
        $\frac{c(S_i)}{|S_c \cup S_i| - |S_c|}$.
 9:     Set $S_c := S_i \cup S_c$ and $\mathcal{L}_s := \mathcal{L}_i$ where $S_i = \mathcal{L}_i$.
10:     Get the new labeling $f' = \arg\min E(f)$ within the range
        expansion move on $\mathcal{L}_s$.
11:     If $E(f') < E(f)$, set $f := f'$.
12:   end while
13: until No moves can be found to decrease $E(f)$.
Output:
14: Return the labeling $f$.
```

The GREA In the GREA, a series of candidate submodular sets can usually be found. However, it is inefficient to perform the range expansion moves on all the submodular sets. In practice, a high quality solution is found by requiring only that *all the labels in $\mathcal{L}$ are visited once in each cycle of iterative moves*, i.e., this guarantees that every vertex has an opportunity to change its current label to any one of the other labels.

To obtain enough expected moves to cover all the labels in $\mathcal{L}$, we formulate the following *set cover problem.*

Let $\{\mathcal{L}_1, \mathcal{L}_2, \cdots, \mathcal{L}_k\}$ be the series of submodular sets of labels. We define the universe $U = \mathcal{L}$, and the collection of setS $S_i = \mathcal{L}_i$. The moves are obtained by find the set cover $\mathcal{S}'$ which solves the following problem:

$$\min_{\mathcal{S}' \subseteq \mathcal{S}} \sum_{S_i \in \mathcal{S}'} c(S_i) \qquad \text{s.t.} \bigcup_{S_i \in \mathcal{S}'} S_i = U \,. \tag{7}$$

where $c(S_i)$ is the cost of the subset S_i.

Algorithm 2 describes the iterative process of the GRSA, where the moves are chosen by dynamically solving the SCP with the greedy algorithm. In the algorithm, step 6-11 is loop, In step 7, the set:

$$S_i = \arg\min_{S_i \in \mathcal{S}} \frac{c(S_i)}{|S_c \cup S_i| - |S_c|} \tag{8}$$

is chosen, where S_c is the set of elements already chosen. $|S_c \cup S_i| - |S_c|$ is the increase in the number of elements when set S_i is added into S_c. In this paper, we set: $c(S_i) = 1$ for simplicity.

5 Graph construction

In this section, we explain the graph construction for both GRSA and GREA. Although the graph construction methods proposed by Ishikawa (2003) and Kumar et al. (2011) can exactly model the submodular potentials, we develop improved graph construction methods that perform much faster than previous methods[7]. In the GRSA, the improved graph construction exactly models the range swap move as previous methods, but it needs much fewer edges to model the pairwise potentials compared to previous methods. In the GREA, the improved graph approximately model the range expansion moves as previous methods (Veksler 2007; Kumar et al. 2011), since it is difficult to model the pairwise potential $\theta(f'_p, f'_q)$ of the cases that f'_q is assigned a label in $\mathcal{L}_s$ and f'_p keep its current label that lies outside $\mathcal{L}_s$. Experimental results show that the improved graph construction models the pairwise potentials better than previous methods (Veksler 2007; Kumar et al. 2011).

5.1 Graph construction for GRSA

Given a submodular set $\mathcal{L}_s$, the range swap move can be exactly solved by calculating the st-mincut of a special graph $\mathcal{G}$. There is a one-to-one correspondence between the st-cut of $\mathcal{G}$ and the configuration of the new labeling f', while the cost of edges in the st-cut exactly models the energy $E(f)$.

Assume the submodular set $\mathcal{L}_s = \{l_1, \cdots, l_m\}$ contains m labels, and the current label of every vertex p is f_p. For the move on $\mathcal{L}_s$, we construct a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, such that a set of nodes $\{p_1, p_2, \cdots, p_m\} \subset \mathcal{V}$ is defined for each $p \in \mathcal{P}_s$ as shown in Fig. 3. In addition, there are two special nodes in $\mathcal{V}$, the *source* node s and the *sink* node t.

The edges in $\mathcal{E}$ with capacity are defined to represent the potentials in the energy $E(f)$. There are two types of edges in $\mathcal{E}$: i) *Unary edges*: the edges that are used to represent the unary potential θ_p for every vertex $p \in \mathcal{P}_s$; ii) *Pairwise edges*: the edges that are used to model the pairwise potentials θ_{pq} for $(p,q) \in \mathcal{E}$.

5.1.1 The Unary Edges

For the vertices $p \in \mathcal{P}$ that $f_p \notin \mathcal{L}_s$, we do not define any nodes or edges to model their unary potentials since they are not considered by the range swap move on $\mathcal{L}_s$, and will retain their current label f_p.

For all the vertices $p \in \mathcal{P}$ that $f_p \in \mathcal{L}_s$, we define the following edges in $\mathcal{E}$:

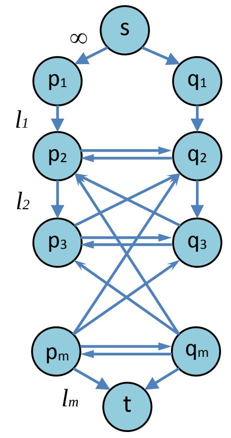


Fig. 3 The graph construction in the st-mincut problem to solve the range swap move.

- For all $i \in [1,m)$, edges (p_i, p_{i+1}) have the capacity $c(p_i, p_{i+1}) = \theta_p(l_i)$.
- For all $i \in [1,m)$, edges (p_{i+1}, p_i) have the capacity $c(p_{i+1}, p_i) = \infty$.
- Edges (p_m, t) have the capacity $c(p_m, t) = \theta_p(l_m)$.
- Edges (s, p_1) have the capacity $c(s, p_1) = \infty$.

There are two types of edges in the unary edges: i) the edges with finite capacity: $\mathcal{E}_1 = (p_m, t) \cup (p_i, p_{i+1}),\ \forall\, i \in [1,m)$; and ii) the edges with infinite capacity: $\mathcal{E}_2 = (s, p_1) \cup (p_{i+1}, p_i),\ \forall\, i \in [1,m)$.

Let C denote an st-cut of the graph $\mathcal{G}$ and $\mathcal{E}_c \subset \mathcal{E}$ denote the set of edges belonging to C. The edges in $\mathcal{E}_1$ model the unary potentials $\theta_p(l_i)$, that is the cost when vertex p is assigned label l_i. The edges in $\mathcal{E}_2$ are constructed to guarantee that only one label is assigned to each vertex p, i.e., there is only one unary edge $(p_i, p_{i+1}) \in \mathcal{E}_c$ for any $i \in [1,m)$. This is because the cost will be infinite if $\mathcal{E}_c$ contains more than one unary edges for the vertex p (Ishikawa 2003).

The new labeling f' can be obtained according to the st-cut C as follows:

$$f'_p = \begin{cases} l_i & \text{if } (p_i, p_{i+1}) \in \mathcal{E}_c, \forall i \in [1,m); \\ l_m & \text{if } (p_m, t) \in \mathcal{E}_c. \end{cases} \tag{9}$$

5.1.2 The Pairwise Edges

There are three cases for the pairwise potentials:

i) For $(p,q) \in \mathcal{E}$ where $f_p, f_q \notin \mathcal{L}_s$, the potential θ_{pq} is not represented by any edges in $\mathcal{E}$, because both vertices p, q will retain their current labels and θ_{pq} will remain unchanged.

ii) For $(p,q) \in \mathcal{E}$ where $f_p \in \mathcal{L}_s$, $f_q \notin \mathcal{L}_s$, we add the following edges[8] to $\mathcal{E}_1$:

[7] When θ_{pq} is a truncated linear function, the graph construction in the GRSA is the same as previous methods.

[8] The edges (p_i, p_{i+1}) for all $i \in [1,m)$ and (p_m, t) are already included in the unary edges $\mathcal{E}_1$. We can add the capacities that represent the pairwise potentials to these edges.

- For all $i \in [1, m)$, edges (p_i, p_{i+1}) have the capacity $c(p_i, p_{i+1}) = \theta_{pq}(l_i, f_q)$, that is, it represents the pairwise potential when p is assigned label l_i while q keeps its current label f_q.
- Edges (p_m, t) have the capacity $c(p_m, t) = \theta_{pq}(l_m, f_q)$.

The other case where $f_p \notin \mathcal{L}_s$, $f_q \in \mathcal{L}_s$ can be handled similarly.

iii) For $(p, q) \in \mathcal{E}$ where $f_p, f_q \in \mathcal{L}_s$, we define the following edges as shown in Fig. 3:

$$c(p_i, q_j) = \begin{cases} \frac{\psi(i,j)}{2} & \text{if } 1 < j = i \leq m; \\ \psi(i,j) & \text{if } 1 < j < i \leq m; \\ 0 & \text{if } j > i, \end{cases} \quad (10)$$

where

$$\psi(i,j) = \theta(l_i, l_{j-1}) - \theta(l_i, l_j) - \theta(l_{i-1}, l_{j-1}) + \theta(l_{i-1}, l_j)$$

for $1 < j \leq i \leq m$ (we have $0 \leq \psi(i,j)$ for a submodular set $\mathcal{L}_s$).

Let $\mathcal{E}_3$ represent the pairwise edges (p_i, q_j) for $(p, q) \in \mathcal{E}$. Note that the edges in $\mathcal{E}_1$ and $\mathcal{E}_2$ are defined similarly to previous graph constructions (Ishikawa 2003; Kumar et al. 2011). However, the set of edges $\mathcal{E}_3$ in the new construction contains only the edges (p_i, q_j) for $1 < j \leq i \leq m$ and $(p, q) \in \mathcal{E}$. In contrast, previous graph constructions contain all the edges (p_i, q_j) for $1 < i, j \leq m$. Therefore, the new graph construction requires fewer edges[9] to represent the pairwise potentials. As a result, the st-mincut problem with the new graph construction takes much less time to be solved. Fig. 4 compares the run time of different graph constructions on a truncated quadratic energy function. It can be seen that the new method can be two times faster than competing methods.

Furthermore, although the new construction contains fewer edges, it has the same properties as previous methods as follows.

Property 1 *The cost of the edges $\mathcal{E}_1$ in the st-cut C exactly represents the unary potentials $\theta_p(f'_p)$ for all vertices $p \in \mathcal{P}_s$ and the pairwise potentials $\theta_{pq}(f'_p, f'_q)$ for any $p \in \mathcal{P}_s$, $q \notin \mathcal{P}_s$ or $p \notin \mathcal{P}_s$, $q \in \mathcal{P}_s$.*

Lemma 1 *When edges (p_a, p_{a+1}) and (q_b, q_{b+1}) are in the st-cut C, that is, f_p, f_q are assigned the labels l_a, l_b respectively, let $cut(l_a, l_b)$ denote the cost of the pairwise edges in $\mathcal{E}_3$ in the st-cut. We have the following relationship*

$$cut(l_a, l_b) = \begin{cases} \sum_{i=b+1}^{a} \sum_{j=b+1}^{i} c(p_i, q_j), & \text{if } l_a \geq l_b\ ; \\ \sum_{i=a+1}^{b} \sum_{j=a+1}^{i} c(q_i, p_j), & \text{if } l_a < l_b \end{cases} \quad (11)$$

(Proof in Appendix D).

[9] Note that when θ_{pq} is a truncated linear function, the numbers of edges in the new construction and in the previous method are the same, because $c(p_i, q_j) = 0$ for $i \neq j$ in this case.

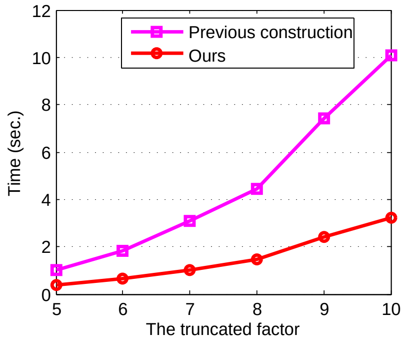


Fig. 4 The comparison of run time with previous graph constructions and our improved graph construction. We evaluate the different graph constructions on truncated quadratic function.

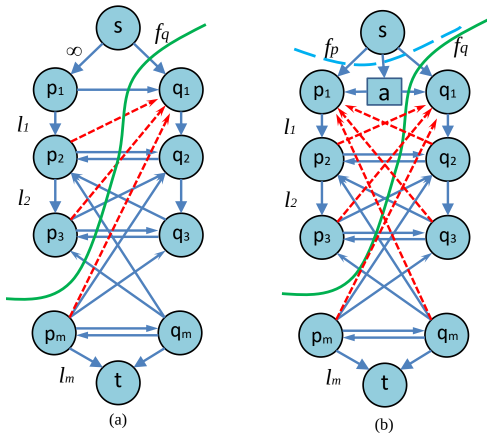


Fig. 5 The improved graph construction in the st-mincut problem to solve the range expansion move.

Property 2 *For $(p, q) \in \mathcal{E}$, if $f_p \in \mathcal{L}_s$, $f_q \in \mathcal{L}_s$, the cost of the st-cut exactly represents the pairwise potentials, ie,*

$$cut(f'_p, f'_q) = \theta_{pq}(f'_p, f'_q).$$

Lemma 2 *For the graph described in Sec. 5.1, Property 2 holds true (Proof in Appendix E).*

Property 1 and 2 implies that the cost of the st-cut on $\mathcal{G}$ exactly models the unary and pairwise potentials. The energy of the new labeling f' obtained by a range swap move is exactly equal to the cost of the st-mincut on $\mathcal{G}$.

5.2 Graph construction for GREA

The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in GREA is constructed similarly to that in GRSA. The difference is that we model the case when the label f'_p does not lie in $\mathcal{L}_s$.

5.2.1 Unary edges

For vertex $p \in \mathcal{P}$, the unary edges (p_m, t) and (p_i, p_{i+1}) for any $i \in [1, m)$ are defined as the cases in GRSA. The difference of the unary edges in GREA is that we change the capacity of edge (s, p_1) to model the unary potentials when p retains its current label $f_p \notin \mathcal{L}_s$:

$$c(s,p_1) = \begin{cases} \theta_p(f_p) & \text{if } f_p \notin \mathcal{L}_s; \\ 0 & \text{otherwise.} \end{cases}$$

The new labeling f' can be obtained according to the st-cut as follows:

$$f'_p = \begin{cases} l_i & \text{if } (p_i, p_{i+1}) \in \mathcal{E}_c, \forall i \in [1,m); \\ l_m & \text{if } (p_i, p_m) \in \mathcal{E}_c. \\ f_p & \text{if } (s, p_1) \in \mathcal{E}_c. \end{cases}$$

5.2.2 Pairwise edges

For the pairwise potentials $\theta_{pq}(f'_p, f'_q)$, if $f'_p, f'_q \in \mathcal{L}_s$, the edges are defined as Eq. 10 to represent $\theta_{pq}(f'_p, f'_q)$. However, we also have to model the cases that at least one vertex keeps its current label f_p that lies outside $\mathcal{L}_s$. In this case, the graph $\mathcal{G}$ for the range expansion move is complicated, since the pairwise potential $\theta(f'_p, f'_q)$ cannot be exactly represented by the cost of the st-cut. We develop the following graph construction method to model the pairwise potential approximately.

i) If $f_p \in \mathcal{L}_s$ and $f_q \notin \mathcal{L}_s$, we add the edges (p_i, q_1) in order (from 1 to m):

$$c(p_i,q_1) = \begin{cases} \theta(l_1, f_q) & i = 1; \\ \max(0, \theta(l_i, f_q) - \theta(l_i, l_1) & \\ - \sum_{j=1}^{i-1} c(p_j, q_1)) & 2 \le i \le m. \end{cases}$$

ii) If $f_q \in \mathcal{L}_s$ and $f_p \notin \mathcal{L}_s$, we add the following edges:

$$c(q_i,p_1) = \begin{cases} \theta(f_p, l_1) & i = 1; \\ \max(0, \theta(f_p, l_i) - \theta(l_1, l_i) & \\ - \sum_{j=1}^{i-1} c(q_j, p_1)) & 2 \le i \le m. \end{cases}$$

iii) If $f_p \notin \mathcal{L}_s$ and $f_q \notin \mathcal{L}_s$, we define

$$\delta = \max(0, \frac{\theta(f_p, f_q) - \theta(l_1, f_q) - \theta(f_p, l_1)}{2}), \tag{12}$$

and add an *auxiliary node* a with the following edges:

$$c(s,a) = \theta(f_p, f_q)$$
$$c(a,q_1) = \theta(l_1, f_q) + \delta,\ c(q_1, a) = \infty$$
$$c(a,p_1) = \theta(f_p, l_1) + \delta,\ c(p_1, a) = \infty$$

$$c(p_i,q_1) = \begin{cases} \theta(l_2, f_q) - c(a, q_1), & i = 2; \\ \max(0, \theta(l_i, f_q) - \theta(l_i, l_1) & \\ -c(a, q_1) - \sum_{j=2}^{i-1} c(p_j, q_1)), & 3 \le i \le m. \end{cases}$$

$$c(q_i,p_1) = \begin{cases} \theta(f_p, l_2) - c(a, p_1), & i = 2; \\ \max(0, \theta(f_p, l_i) - \theta(l_1, l_i) & \\ -c(a, p_1) - \sum_{j=2}^{i-1} c(q_j, p_1)), & 3 \le i \le m. \end{cases}$$

The above graph construction allows the GREA to handle more general functions instead of only truncated convex functions. The new graph construction has the following properties:

Property 3 *The cost of unary edges $\mathcal{E}_1$ in the st-cut exactly represents the unary potentials $\theta_p(f'_p)$ for all vertices $p \in \mathcal{P}_s$.*

Property 4 *For $(p,q) \in \mathcal{E}$, if $f'_p \in \mathcal{L}_s$ and $f'_q \in \mathcal{L}_s$, the cost of st-cut $cut(f'_p, f'_q)$ exactly represents the pairwise potentials $\theta_{pq}(f'_p, f'_q)$.*

Property 5 *For $(p,q) \in \mathcal{E}$, if $f'_p = f_p \notin \mathcal{L}_s$ and $f'_q = f_q \notin \mathcal{L}_s$, the cost of st-cut $cut(f'_p, f'_q)$ exactly represents the pairwise potentials $\theta_{pq}(f'_p, f'_q)$.*

The most complicated case is when one node is given a label inside $\mathcal{L}_s$ and the other node keeps its current label that lies outside $\mathcal{L}_s$. For brevity, we only consider the case $f'_p \in \mathcal{L}_s$, $f'_q = f_q \notin \mathcal{L}_s$. A similar argument applies when $f'_q \in \mathcal{L}_s$, $f'_p = f_p \notin \mathcal{L}_s$.

Property 6 *For $(p,q) \in \mathcal{E}$, if $f'_p \in \mathcal{L}_s$, $f'_q = f_q \notin \mathcal{L}_s$ and assume $f'_p = l_a$, then the cost of the st-cut is*

$$cut(f'_p, f'_q) = \begin{cases} \theta(l_a, l_1) + \sum_{i=2}^{a} c(p_i, q_1) + \theta(l_1, f_q), & f_p \in \mathcal{L}_s; \\ \theta(l_a, l_1) + \sum_{i=2}^{a} c(p_i, q_1) + \theta(l_1, f_q) & \\ +\delta, & f_p \notin \mathcal{L}_s. \end{cases}$$

where δ is defined in Eq. 12.

Property 3 implies that the cost of the st-cut on graph $\mathcal{G}$ exactly models the unary potentials $\theta_p(f'_p)$. Property 4 and 5 specify the cases when the cost of the st-cut $cut(f'_p, f'_q)$ exactly models the pairwise potentials, and Property 6 specified the remaining cases when $cut(f'_p, f'_q)$ overestimates the pairwise potentials. In other words, the energy on a new labeling f' is less than or equal to the cost of the st-mincut on $\mathcal{G}$.

To compare the properties of $\mathcal{G}$ and previous graph constructions, we consider the special case of truncated convex functions for which previous graph constructions are developed. Let $d(\cdot)$ denote a convex function and $\theta(f_p, f_q) = \min\{d(|f_p - f_q|), T\}$ be a truncated convex energy function. We consider a submodular set $\mathcal{L}_s = \{l_1, \cdots, l_m\}$ $(l_i < l_{i+1})$, where $d(l_m - l_1) \le T$ holds true. Using Property 6, we obtain the following results for the graph $\mathcal{G}$ in GREA.

Lemma 3 *When the pairwise function is a truncated function $\theta(f_p, f_q) = \min\{d(|f_p - f_q|), T\}$, for the case $f'_p \in \mathcal{L}_s$ and $f'_q = f_q \notin \mathcal{L}_s$, we have the following properties:*

– *If $f_q > l_m$, we have*

$$\theta(f'_p, f'_q) \le cut(f'_p, f'_q) \le d(|f'_p - l_1|) + T;$$

– *If $f_q < l_1$ and $f_p \in \mathcal{L}_s$ or $f_p < l_1$,*

$$\theta(f'_p, f'_q) \le cut(f'_p, f'_q) \le \min\{d(|f'_p - f'_q|), d(|f'_p - l_1|) + T\};$$

– *If $f_q < l_1$ and $f_p > l_m$, we have*

$$\theta(f'_p, f'_q) \le cut(f'_p, f'_q) \le \min\{d(|f'_p - f'_q|) + \frac{T}{2}, d(|f'_p - l_1|) + T\},$$

(Proof in Appendix E).

Lemma 3 tells us that if $f'_p \in \mathcal{L}_s$ and $f'_q = f_q \notin \mathcal{L}_s$, the cost of the st-cut $cut(f'_p, f'_q)$ is larger than or equal to the pairwise potentials, and at the same time, it gives the upper bound of $cut(f'_p, f'_q)$. In particular, for the case where if $f_q < l_1$ and $f_p \in \mathcal{L}_s$ or $f_p < l_1$, if $d(|f'_p - f'_q|) \le T$, we have $cut(f'_p, f'_q) = \theta(f'_p, f'_q) = d(|f'_p - f'_q|)$.

According to Property 9 in Kumar et al. (2011), for the case $f'_p \in \mathcal{L}_s$ and $f'_q = f_q \notin \mathcal{L}_s$, the graph construction of Kumar et al. (2011) always has that

$$\theta(f'_p, f'_q) < cut^K(f'_p, f'_q) = d(|f'_p - l_1|) + \hat{d}(|f'_p - l_1|) + T,$$

where $\hat{d}(x) = d(x+1) - d(x) - d(1) + \frac{d(0)}{2}$ and cut^K is the cost of the st-mincut in graph (Kumar et al. 2011). Thus,

$$\theta(f'_p, f'_q) \le cut(f'_p, f'_q) < cut^K(f'_p, f'_q). \tag{13}$$

In other word, the cost of the st-cut in our graph models the pairwise potentials more accurately.

As a result, if we perform the range expansion moves on the same submodular sets, the proposed graph construction will yield a better solution than previous methods. To prove this theoretically, let f^* denote an optimal labeling of $E(f)$. We consider a submodular set $\mathcal{L}_i$ and define the following sets:

$$\mathcal{P}^i = \{p | p \in \mathcal{P}, f^*_p \in \mathcal{L}_i\}$$
$$\mathcal{A} = \{(p,q) | (p,q) \in \mathcal{E}, f^*_p \in \mathcal{L}_i, f^*_q \in \mathcal{L}_i, \}$$
$$\mathcal{B}_1 = \{(p,q) | (p,q) \in \mathcal{E}, f^*_p \in \mathcal{L}_i, f^*_q \notin \mathcal{L}_i, \}$$
$$\mathcal{B}_2 = \{(p,q) | (p,q) \in \mathcal{E}, f^*_p \notin \mathcal{L}_i, f^*_q \in \mathcal{L}_i, \}$$

Using Lemma 3 of Kumar et al. (2011), we obtain the following results.

Lemma 4 *At an iteration of the GREA, given the current labeling f and a submodular set $\mathcal{L}_i$, the new labeling f' obtained by solving the st-mincut problem reduces the energy by at least the following:*

$$\sum_{p \in \mathcal{P}^i} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{A} \cup \mathcal{B}} \theta_{pq}(f_p, f_q) - (\sum_{p \in \mathcal{P}^i} \theta_p(f^*_p) + \sum_{(p,q) \in \mathcal{A}} \theta(f^*_p, f^*_q) + \sum_{(p,q) \in \mathcal{B}_1} cut(f^*_p, f_q) + \sum_{(p,q) \in \mathcal{B}_2} cut(f_p, f^*_q)). \tag{14}$$

where $cut(f^*_p, f^*_q)$ is the cost of st-cut when p and q are assigned label f^*_p, f^*_q respectively.

Let $\hat{f}$ be a local optimum obtained using range expansion moves on the series of submodular sets $\mathcal{L}' = \{\mathcal{L}_1, \cdots, \mathcal{L}_k\}$, where $\bigcup_{\mathcal{L}_i \in \mathcal{L}'} \mathcal{L}_i = \mathcal{L}$. It follows that

$$\sum_{p \in \mathcal{P}^i} \theta_p(\hat{f}_p) + \sum_{(p,q) \in \mathcal{A} \cup \mathcal{B}} \theta_{pq}(\hat{f}_p, \hat{f}_q) \le \sum_{p \in \mathcal{P}^i} \theta_p(f^*_p) + \sum_{(p,q) \in \mathcal{A}} \theta(f^*_p, f^*_q) + \sum_{(p,q) \in \mathcal{B}_1} cut(f^*_p, f_q) + \sum_{(p,q) \in \mathcal{B}_2} cut(f_p, f^*_q)),$$

for all $\mathcal{L}_i \subset \mathcal{L}'$, because the term 14 should be non-positive, otherwise the energy $E(\hat{f})$ can be further reduced, which contradicts the fact that $\hat{f}$ is a local optimum labeling.

We sum the above inequality over all $\mathcal{L}_i \subset \mathcal{L}'$, and obtain the following results

$$\begin{aligned} E(\hat{f}) &\le \sum_{\mathcal{L}_i \subset \mathcal{L}'} (\sum_{p \in \mathcal{P}^i} \theta_p(\hat{f}_p) + \sum_{(p,q) \in \mathcal{A} \cup \mathcal{B}} \theta_{pq}(\hat{f}_p, \hat{f}_q)) \\ &\le \sum_{\mathcal{L}_i \subset \mathcal{L}'} (\sum_{p \in \mathcal{P}^i} \theta_p(f^*_p) + \sum_{(p,q) \in \mathcal{A}} \theta(f^*_p, f^*_q) + \\ &\sum_{(p,q) \in \mathcal{B}_1} cut(f^*_p, f_q)) + \sum_{(p,q) \in \mathcal{B}_2} cut(f_p, f^*_q)) \end{aligned} \tag{15}$$

With Eq. 13, we have that $cut(f^*_p, f^*_q) < cut^K(f^*_p, f^*_q)$. Therefore, if we perform the range expansion moves on the same submodular sets, the graph construction described in Sec. 5.2 produces solutions with a tighter bound (see Eq. 15) than previous methods (Kumar et al. 2011).

6 Experiments

In this section, we evaluate our GRMAs on both synthetic data and the real vision applications including image restoration, stereo matching and image segmentation. The performance is compared with several state-of-the-art methods, including α-expansion, $\alpha\beta$-swap, BPS (Tappen and Freeman 2003), TRW-S (Kolmogorov 2006), as well as previous range swap and range expansion algorithms. We also compare the GRMAs with several variations of the range move algorithms to verify the effectiveness of the GRMAs. The variations including the range move algorithms with half moves (*R. swap Half* and *R. expan. Half*), randomized range move methods (*R. swap Rand1*, *R. swap Rand2*, *R. expan. Rand1* and *R. expan. Rand2*) and the range swap + t method (Veksler 2007). The range move algorithms with half moves perform the range moves $\{0, \cdots, T\}$,$\{\frac{T}{2}, \cdots, \frac{T}{2} + T\}$,$\cdots$,$\{\frac{mT}{2}, \cdots, \frac{mT}{2} + T\}$,$\cdots$ in the iterations, where T is the truncation factor in a truncated convex function. The first kind of randomized range move algorithms (*R. swap Rand1* and *R. expan. Rand1*) perform the range moves $\{0, \cdots, T\}$,$\{t_1, \cdots, t_1 +$

$T\}, \cdots, \{\sum_{i=1}^{m} t_i, \cdots, \sum_{i=1}^{m} t_i + T\}$, where t_i is randomly sampled from the interval $[0, T]$. The second kind of randomized range move algorithms (*R. swap Rand2* and *R. expan. Rand2*) perform the range moves $\{0, \cdots, T_0\}$, $\{t_1, \cdots, t_1+T_1\}, \cdots, \{\sum_{i=1}^{m} t_i, \cdots, \sum_{i=1}^{m} t_i+T_m\}$, where t_i, T_i are randomly sampled from the interval $[0, T]$ and $[T, T+4]$ respectively. In the range swap + t algorithm, every move considers the label set $\{\alpha - t, \alpha - t + 1, \cdots, \beta + t\}$, where $|\beta - \alpha| = T$ to obtain a better solution. We set $t = 2$ in all the experiments.

In the experiments, we use the codes provided by Kumar et al. (2011) for previous range swap and range expansion algorithms. We perform "*all swap moves*" plus "*range swap moves*" in the iterative moves in previous range swap, range swap + t and randomized range swap algorithms. This is because some important moves are missing in previous range swap algorithms, *i.e.*, the pairs of labels $\{\alpha, \beta\}$ are not considered if $|\alpha - \beta| > T$. For a vertex p, whose current label is α and real label is β ($|\alpha - \beta| > T$), unfortunately there is no move from α to β. This may lead to a poor solution especially when the label space is large (e.g. $|\mathcal{L}| = 256$ in image restoration).

We also evaluate the methods that perform all the range moves with our improved graph construction method. We call them "*RS + Our graph*" and "*RE + Our graph*", respectively. The only difference between "*RS + Our graph*", "*RE + Our graph*" and previous range move algorithms is that the methods of the graph construction are different. We verify the effectiveness of the proposed graph construction method by comparing "*RS + Our graph*" and "*RE + Our graph*" with previous range swap and range expansion algorithms.

We introduce the test data and experimental setting in Sec. 6.1. We evaluate the effectiveness of the GRSA in Sec. 6.3, and we evaluate the GREA in Sec. 6.4.

6.1 Data and experimental setting

Synthetic data The computation times of previous range move algorithms and the GRMAs are affected by multiple factors, such as the number of labels, the size of the MRF and the parameters of the pairwise functions. To give a detailed comparison of our GRMAs and previous range move algorithms under various cases, we evaluate them on the synthetic MRFs whose parameters are generated randomly. Following (Kumar and Torr 2009; Kumar et al. 2011), the data term $\theta_p(f_p)$ is sampled uniformly from the interval $[0, 10]$. For the pairwise term, we use the truncated convex function

$\theta_{pq} = 3\min\{(f_p - f_q)^2, T^2\}$.

In the experiments, we evaluate the influence of the label size, the size of the MRFs and the truncation factor T. For example, we first fix $T = 5$, and test the run time of the algorithms for a range of MRFs of increasing size. Then, we fix the size of the MRFs as 100×100, and evaluate the influence of the truncation factor T. In each group of experiments, we test the performance of the algorithms on 50 random fields, and compare the average run time and average energies.

Image restoration In image restoration, the given input images are corrupted with noise and the objective is to reconstruct the original images by removing the noise. We use two popular images from the Corel database: *penguin* and *house*. In the experiments, we set $\mathcal{L} = \{0, 1, \cdots, 255\}$, and test the GRMAs on two pairwise functions: 1) the truncated convex function $\theta_{pq}(f_p, f_q) = 25\min\{(f_p - f_q)^2, 200\}$ with parameters set as in Veksler (2012) and Kappes et al. (2013); and 2) the piecewise linear function:

$$\theta_{pq}(f_p, f_q) = \begin{cases} 25|f_p - f_q|, & \text{if } |f_p - f_q| \le 15; \\ 25 \times 15, & \text{if } 15 < |f_p - f_q| < 45; \\ 25(|f_p - f_q| - 35), & \text{if } 45 \le |f_p - f_q|. \end{cases}$$

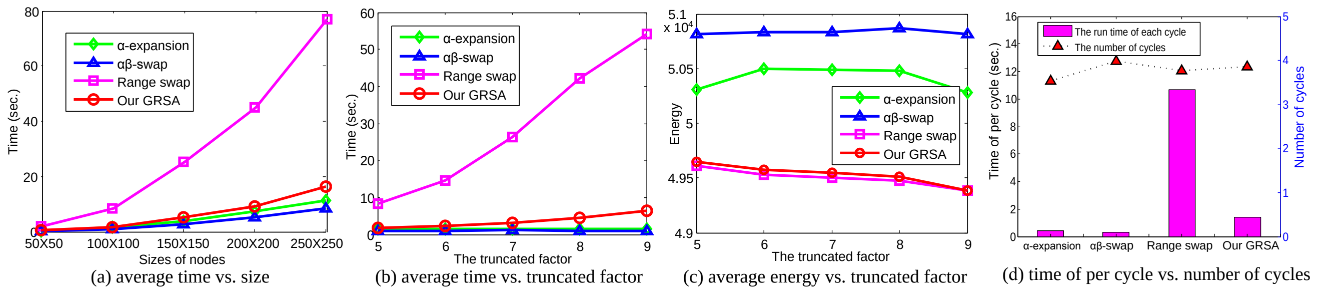
Stereo matching In stereo correspondence, the goal is to find corresponding pixels in the left and right images. In this experiment, we use the image pairs from the popular Middlebury Stereo Database. The size of the label space is equal to the number of values for the disparity for the image pairs. We use two kinds of energy function: the truncated function $\theta_{pq}(f_p, f_q) = 30\min\{(f_p - f_q)^2, T^2\}$ and the piecewise linear function

$$\theta_{pq}(f_p, f_q) = \begin{cases} 30|f_p - f_q|, & \text{if } |f_p - f_q| < T; \\ |f_p - f_q| + 30T, & \text{otherwise.} \end{cases}$$

We set $T = 8$ for *tsukuba*, and $T = 10$ for other image pairs. We also evaluate the accuracy of results on Middlebury On-line Evaluation[10] (Scharstein and Szeliski 2002) with two different error thresholds (ET).

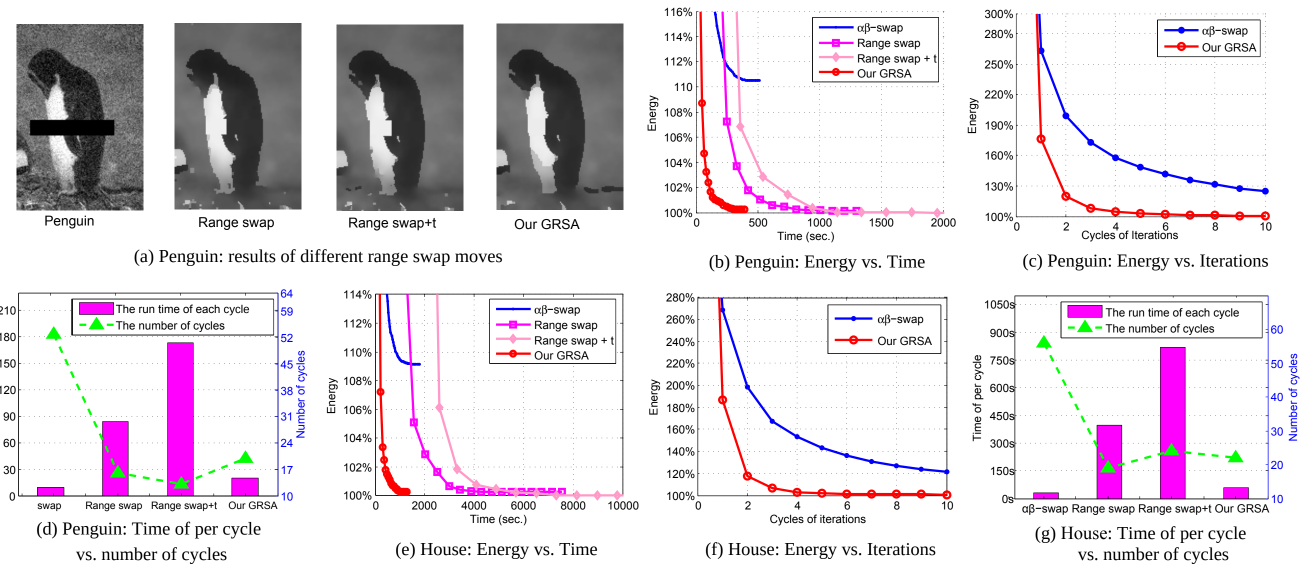
Intensity-based Segmentation The objective of segmentation (Boykov and Kolmogorov 2003; Nagarajan 2003) is to separate one or more regions of interest in an image. We test the algorithms on the Berkeley Segmentation Dataset (Martin et al. 2001) (see Fig. 11). In the experiments, we divide the image pixels into n classes according to image intensity such that $\mathcal{L} = \{0, 1, \cdots, n\}$. The unary term $\theta_p(f_p)$is set to be $(I_p - \frac{255 * f_p}{n})$, where I_p is the image intensity. For the pairwise energy function, we use $\theta_{pq}(f_p, f_q) = 500\min\{(f_p - f_q)^2, T^2\}$. In the experiments, we set $n = 10$ and $T = 7$.

[10] http://vision.middlebury.edu/stereo/eval/



(a) average time vs. size (b) average time vs. truncated factor (c) average energy vs. truncated factor (d) time of per cycle vs. number of cycles

Fig. 6 The evaluation of the GRSA on synthetic data. Each result is an average of the results obtained from 50 MRFs with truncated convex functions. (a) and (b) show the algorithms' run time with different sizes of MRFs or different truncation factors T. (c) shows the energy with different truncation factors T (size 100×100). (d) shows the number of cycles and the run time for each cycle when $T = 8$ and the MRFs' size is 100×100.



(a) Penguin: results of different range swap moves (b) Penguin: Energy vs. Time (c) Penguin: Energy vs. Iterations (d) Penguin: Time of per cycle vs. number of cycles (e) House: Energy vs. Time (f) House: Energy vs. Iterations (g) House: Time of per cycle vs. number of cycles

Fig. 7 The evaluation of the GRSA on image restoration. (a) shows the results obtained by range swap (Kumar and Torr 2009), range swap + t and the GRSA on *penguin*. (b) and (e) show the energy obtained by different algorithms as a function of run time on *penguin* and *house* respectively. (c) and (f) show the obtained energy as a function of the number of cycles on *penguin* and *house* respectively. The value of energy is plotted in percentage points, where 100% is set to be the lowest value achieved by any algorithm in (b), (c), (e) and (f). (d) and (g) show the run time taken by each cycle in different algorithms, and the number of cycles taken by each algorithm to converge.

6.2 Evaluation on the improved graph construction

We first evaluate the effectiveness of the proposed graph construction in this section. We compare it with previous construction method (Kumar et al. 2011) on both synthetic data and real problems. Fig. 8 shows the results for synthetic data, while Fig. 9, Tab. 2 and Tab. 3 show the results on the real applications of image restoration, stereo matching and image segmentation.

In the figures, ”*RS + our graph*” denotes that we perform all the range swap moves with our graph construction. From Tab. 3, we see that ”*RS + our graph*” yields the same results as previous range swap algorithms. However, the ”*RS + our graph*” performs about 3 times faster than previous range swap algorithm. Note that ”*RS + our graph*” and previous range swap algorithm take the same range swap moves in the optimization. Therefore, this shows that the improved graph construction in the GRSA is much faster than previous graph construction, while it yields the same solutions as previous graph construction.

”*RE + our graph*” denotes that we perform all the range expansion moves with our graph construction. In Fig. 8 and Fig. 9(b),(d), we see that our graph construction runs 2-5 times faster than previous methods on truncated quadratic functions, e.g., on image *penguin*, previous graph construction (*range expansion*) takes 68532.4 seconds, while our method (*RE + our graph*) takes 11907.8 seconds.

Fig 9(a), (c) and Tab. 1 show the quality of our graph construction for image restoration, while Tab. 2 and Tab. 3 show the results on stereo matching and image segmentation, respectively. We see that ”*RE + our graph*” performs much better than previous range expansion methods (e.g., on

Algorithm	Energy (penguin)	Time (penguin)	Energy (house)	Time (house)
$\alpha\beta$-swap	17367822	512.4	45114530	1789.8
Range swap	15740426	1448.7	41438765	7536.5
R. swap Half	15752587	898.6	41452186	3389.8
R. swap Rand1	15763214	2103.1	41443603	7646.3
R. swap Rand2	15724023	2367.4	41410447	9404.7
Range swap+t	**15716390**	2258.0	**41332881**	15753.3
Our GRSA	15758765	**392.0**	41452670	**1278.7**
α-expansion	16108436	**99.8**	41567659	**190.1**
Range expan.	15817602	68532.4	41290076	129280.4
R. expan. Half	15974414	10206.1	41491386	65700.6
R. expan. Rand1	15828659	16220.0	41333511	100722.1
R. expan. Rand2	15799637	27366.1	41293033	172447.8
RE+Our graph	**15624248**	11907.8	**40946750**	40378.3
Our GREA	15693819	1349.3	41159653	5448.4
$\alpha\beta$-swap	9395637	221.0	26648428	453.5
Our GRSA	**8675869**	**149.3**	**23827954**	**371.1**
Our GREA	9210063	652.7	25615002	3805.6

Table 1 The results for image restoration. The optimization algorithms are evaluated on two images. The 2-3 columns show the energy and run time on image *penguin*, respectively. The 4-5 columns show the results on image *house*. The 2-7 rows show the results on the truncated convex function, and 8-9 rows show the results on the piecewise linear function.

image penguin, previous graph construction obtains an energy 15817602, while "*RE + our graph*" obtains 15624248). In the experiments, "*RE + our graph*" and previous range expansion algorithm perform the range expansion moves on the same series of subsets. As a result, we see that the improved graph construction in the GREA not only runs much faster, but also yields better solutions than previous graph construction methods.

6.3 Evaluation on GRSA

Comparison with traditional range swap algorithm
Efficiency: To quantify the efficiency of the GRSA, we compare the run time of the GRSA with previous range swap algorithms on truncated convex functions. To avoid the influence of implementation details, we use the code[11] provided by Kumar and Torr (2009) for their range swap algorithm. Fig. 6 shows the run time of the algorithms on synthetic data with different influential factors. We observe that the run time of the previous range swap algorithm increases repidly as the size of the MRFs or the truncation factor increases. In contrast, the run time of the GRSA increases more slowly. The run time is reduced by more than 80% in many cases (e.g. $T = 8$, size 100×100). In real problems, the GRSA also runs much faster. As shown in Fig. 7 (b) and (e), and Tab. 1, we see that for image restoration the GRSA runs at least 3-6 times faster than previous range swap, and 5-14 times faster than the range swap + t algorithm. From Tab. 3, it is seen that for image segmentation the GRSA runs 6-10 times faster than previous range swap algorithm. The GRSA is much more efficient because the set cover formulation reduces a large number of unnecessary moves. Thus the GRSA takes much less time for each cycle of iterations. As shown in Fig. 6 (d), Fig. 7 (d) and (g), we see that the GRSA takes a similar number of cycles to converge, but runs several times faster in each cycle compared to previous range swap algorithms.

Performance: We quantify the performance of the GRSA and previous range swap algorithms on image restoration, stereo matching and image segmentation. In stereo matching, the range swap + t algorithm obtains the best results on the truncated convex function (Tab. 2). The GRSA and previous range swap algorithm also yield promising and similar results. For the piecewise linear function, we see that GRSA yields the best solutions. In image restoration, the range swap + t algorithm ("*all swap moves*" plus "*all range swap + t moves*") yields the best results. However, we can see that the GRSA obtains very similar solutions compared to both the range swap + t and range swap algorithms as shown in Fig. 7.

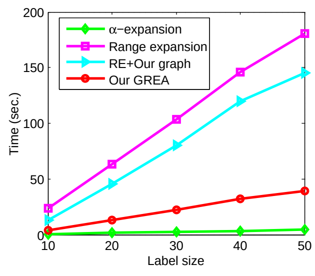
The above analysis shows that the GRSA offers a great speedup over previous range swap algorithms, and at the same time achieves competitive solutions without losing much in accuracy on both synthetic data and real problems.

Comparison with $\alpha\beta$-swap algorithm We compare the GRSA with $\alpha\beta$-swap on both synthetic data and real problems. As expected, the GRSA outperforms $\alpha\beta$-swap on both synthetic data and the real problems of image restoration, stereo matching and image segmentation, because the GRSA considers more labels in every move. The GRSA achieves not only a lower energy but also a better accuracy compared to $\alpha\beta$-swap. For example, as shown in Fig. 10, the error in the results obtained by $\alpha\beta$-swap on tsukuba is 10.3 (ET $= 2$), while the error of the GRSA is 4.5.

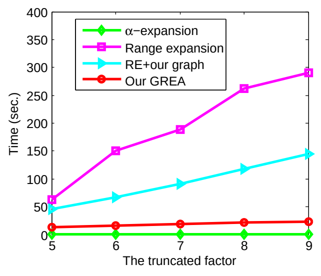
In the experiments, we find that the GRSA takes a similar time to converge as $\alpha\beta$-swap in most cases of synthetic data and stereo matching. However, in image restoration, the GRSA surprisingly converges faster than $\alpha\beta$-swap on both the truncated convex and piecewise linear functions. This is because although the GRSA takes more time in each cycle of iterations, it needs far fewer cycles to converge as shown in Fig. 7 (d) and (g). For example, the GRSA takes 21 cycles while $\alpha\beta$-swap takes 55 cycles to converge for the image *penguin*. In Fig. 7 (c) and (f), we also see that each cycle of the GRSA decreases $E(f)$ much more than $\alpha\beta$-swap does.

Comparison with the variations of range swap algorithms To evaluate the effectiveness of the GRSA, we compare it with different variations of range swap algorithms, including

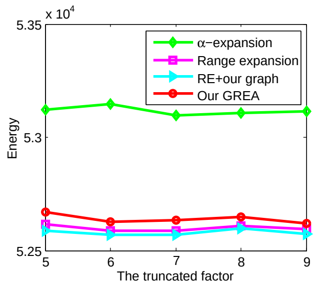
[11] http://cvn.ecp.fr/personnel/pawan/research/truncated-moves.html



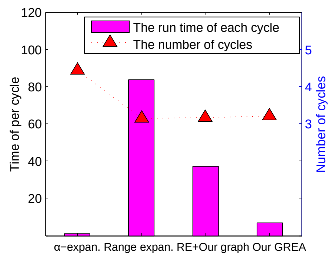
(a) time vs. label size



(b) time vs. truncation factor

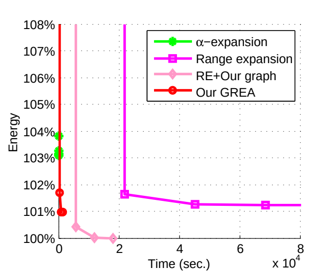


(c) energy vs. truncation factor

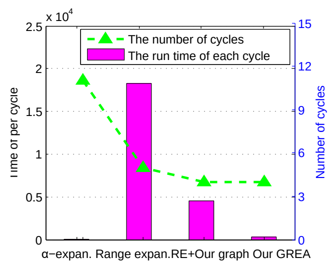


(d) time vs. cycles

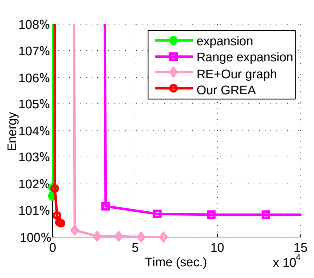
Fig. 8 The evaluation of the GREA on synthetic data. Each result is an average of the results obtained from 50 MRFs with truncated convex functions. (a) and (b) show the algorithms' run time with different label sizes or different truncation factors T. (c) shows the energy with different truncation factors T (size 100×100). (d) shows the number of cycles and the run time of each cycle when $T = 8$ and the MRFs' size is 100×100.



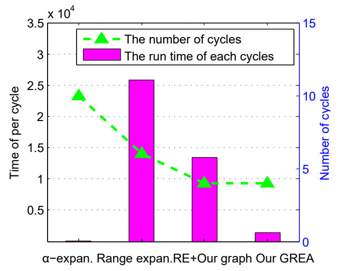
(a) Penguin: energy vs. time



(b) Penguin: time vs. cycles

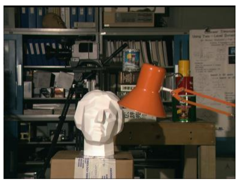


(c) House: energy vs. time

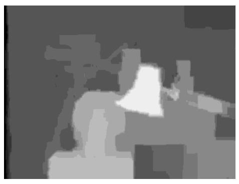


(d) House: time vs. cycles

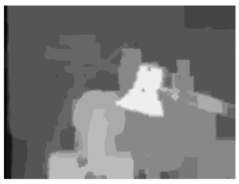
Fig. 9 The evaluation of the GREA on image restoration. (a) and (c) show the energy obtained by different algorithms with the run time on *penguin* and *house* respectively. The value of the energy is plotted in percentage points, where 100% is set to the lowest value achieved by any algorithm. (b) and (d) show the run time taken by every cycle of iterations in different algorithms, and the number of cycles that each algorithm takes to converge.



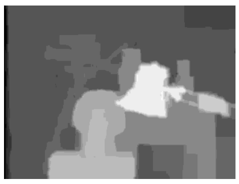
(a) Tsukuba



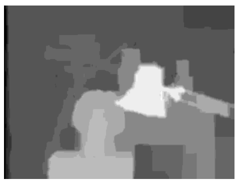
error1=13.9 # error2=6.22
Energy=1482258
(b) α-expansion



error1=16.1 # error2=10.3
Energy=2189043
(c) αβ-swap



error1=13.4 # error2=4.50
Energy=1449548
(d) TRWS



error1=12.9 # error2=4.48
Energy=1445585
(e) GRSA

Fig. 10 The stereo matching results obtained by α-expansion, $\alpha\beta$-swap, TRW-S and the GRSA with the truncated convex function. The errors (%) are tested on Middlebury Stereo Evaluation. Error1 is tested with the error threshold $ET = 1$, and error2 is tested with $ET = 2$.

R. swap Half and the randomized range move methods of *R. swap Rand1* and *R. swap Rand2*. From Tab. 1, we see that although *R. swap Rand1* and *R. swap Rand2* yield slightly better solutions than the GRSA, both of them are much slower than the GRSA and they are even much slower than the original range swap algorithm. For example, on the image *penguin R. swap Rand1* and *R. swap Rand2* take 2103.1 and 2367.4 seconds to converge, and the original range swap algorithm takes 1148.7 seconds. In contrast, the GRSA takes 392.0 seconds and it runs more than 5 times faster than both *R. swap Rand1* and *R. swap Rand2*, while it runs about 3 times faster than the original range swap algorithm. The GRSA also runs much faster than *R. swap Half*. For example, on the image *penguin* the GRSA runs more than 2 times faster than *R. swap Half*.

Comparison with other state-of-the-art algorithms In stereo matching, we compare the performance of the GRSA with α-expansion, BPS and TRW-S, all of which have many successful applications. Tab. 2 shows the performance obtained with both truncated convex functions and piecewise linear functions. We see that the range swap + t and RE + our graph algorithms obtain the best results on truncated convex functions. The GRSA, range swap algorithm and TRWS also have competitive results. On the piecewise linear functions, the GRSA obtains the best results compared to α-expansion, $\alpha\beta$-swap, BPS and TRW-S. Besides the energy, the GRSA also yields competitive results in terms of accuracy. Fig. 10 shows the solutions obtained by α-expansion, $\alpha\beta$-swap, TRW-S and the GRSA. It is seen that the GRSA not only obtains lower energy, but also yields promising results compared to these four algorithms.

Fig. 11 The images taken from the Berkeley Segmentation Dataset (Martin et al. 2001).

Algorithm	Energy 1 (Tsukuba)	Energy 2 (Tsukuba)	Energy 1 (Venus)	Energy 2 (Venus)	Energy 1 (Barn)	Energy 2 (Barn)	Energy 1 (Cones)	Energy 2 (Cones)	Energy 1 (Teddy)	Energy 2 (Teddy)
BPS	1686577	1460496	3299917	2633931	1348521	1173486	18272332	15226621	13883973	15214693
TRW-S	1449548	1371572	2630498	2603053	1194482	1172694	17765423	15076300	13530967	11331054
$\alpha\beta$-swap	2189043	1371721	2922213	2952128	1776277	1177890	18013820	15578786	13825543	11373350
Range swap	1449334	-	2629676	-	1193442	-	**17788537**	-	13520524	-
Range swap+t	**1431401**	-	**2629664**	-	**1189356**	-	**17788537**	-	**13519552**	-
Our GRSA	1445585	**1367681**	2630819	**2600259**	1202651	**1172674**	17799329	**15257479**	13521650	**11313025**
α-expansion	1482258	1369101	2712361	2611608	1218686	1175235	17977155	15378752	14016262	11342616
Range expan.	1433469	-	2635454	-	1193549	-	17799663	-	13526149	-
RE+Our graph	**1431655**	-	**2626416**	-	**1193442**	-	**17795254**	-	**13505214**	-
Our GREA	1434967	**1368658**	2637047	**2603042**	**1193442**	**1174693**	17825171	**15310238**	13526071	**11331918**

Table 2 The results for stereo matching. "Energy 1" denotes the energies obtained using the truncated convex function, while "Energy 2" denotes the energies obtained using the piecewise linear function.

Algorithm	Energy (Image1)	Time (Image1)	Energy (Image2)	Time (Image2)	Energy (Image3)	Time (Image3)	Energy (Image4)	Time (Image4)
$\alpha\beta$-swap	13345579	2.1	14497307	2.5	7698050	1.7	11220891	1.3
α-expansion	13524060	1.5	14580819	1.5	7710817	1.8	11238419	2.0
Range swap	**13254189**	113.2	**14411202**	60.5	**7657174**	75.8	**11135438**	60.8
RS+Our graph	**13254189**	47.0	**14411202**	21.4	**7657174**	26.2	**11135438**	21.0
Our GRSA	13259583	**9.9**	14411259	**9.9**	**7657174**	**11.1**	11135679	**9.2**
Range expan.	13587626	115.7	14476909	100.0	7657289	102.0	11145059	96.8
RE+Our graph	**13254189**	90.9	**14411202**	65.4	**7657174**	54.6	**11135438**	55.7
Our GREA	**13254189**	**29.3**	**14411202**	**20.2**	**7657174**	**15.2**	**11135438**	**13.3**

Table 3 The results for image segmentation. The optimization algorithms are evaluated on four images.

6.4 Evaluation on GREA

Comparison with traditional range expansion algorithm
Efficiency: To evaluate the efficiency of the GREA on different influential factors, we compare the run time of the GREA with previous range expansion algorithm on synthetic data. As shown in Fig. 8, we observe that the run time of the previous range expansion rapidly increases as the number of labels or the truncation factor increases. In contrast, the run time of the GREA increases much more slowly.

The GREA also runs much faster on real problems. In Fig. 9(a) and (b), we see that the GREA runs about 50 times faster than previous range expansion algorithm (e.g., on image *penguin*, previous range expansion algorithm takes 68532.4 seconds, while our GREA takes 1349.3 seconds). The GREA shows high efficiency compared to previous range expansion because of two important reason: 1) The set cover formulation in GREA greatly reduces the number of unnecessary moves and the run time in each cycle of iterations. In Fig. 8(d) and Fig. 9(b),(d), we see that the GREA takes similar number of cycles to converge, but runs several times faster in each cycle compared to previous range expansion algorithms. 2) The improved graph construction is much faster than previous graph construction methods. In Fig. 8(d) and Fig. 9(b),(d), we see that although the methods of range expansion and *RE + our graph* have the same number of moves in each cycle, the method RE + our graph runs much faster with our improved graph construction.

Performance: Fig 8(c) shows the comparison of the solution qualities of different algorithms. We see that our GREA performs better than α-expansion, while it performs similarly to the previous range expansion algorithm. Fig. 9 and Tab. 2 show the performance of our GREA on image restoration and stereo matching. We see that the method

RE + our graph obtains the best results among the algorithms. On stereo matching, previous range expansion and the GREA yield promising and similar results. On image restoration, the GREA obtains better solutions than previous range expansion because of the improved graph construction method (see Tab. 1), although it takes fewer moves in the optimization. On image segmentation, we see that the GREA yields the best results among the compared algorithms (see Tab. 3).

The above analysis show that the GREA runs much faster than previous range expansion algorithms, and also achieves competitive solutions on both synthetic data and real problems.

Comparison with the variations of range expansion algorithms To evaluate the effectiveness of the GREA, we compare it with different variations of range expansion algorithms, including *R. expan. Half* and the randomized range move methods of *R. expan. Rand1* and *R. expan. Rand2*. From Tab. 1, we see that the GREA not only runs much faster than these three variations of range expansion algorithms, but also yields better solutions than them. For example, on the image *penguin* the energies obtained by *R. expan. Half*, *R. expan. Rand1* and *R. expan. Rand2* are 15974414, 15828659 and 15799637, respectively, while the energy obtained by the GREA is 15693819. Meanwhile, on the image *penguin R. expan. Half*, *R. expan. Rand1* and *R. expan. Rand2* take 10206.1, 16220.0 and 27366.1 seconds to converge, respectively. In contrast, the GREA only takes 1349.3 seconds and it runs 7 times faster than *R. expan. Half*, 12 times faster than *R. expan. Rand1* and 20 times faster than *R. expan. Rand2*.

Comparison with other state-of-the-art algorithms First, we compare the performance of the GREA with α-expansion, which is one of the most successful algorithms. In Fig. 8(c), we see that GREA outperforms α-expansion on synthetic data, because GREA considers more labels in each move. GREA also performs better than α-expansion on real problems. In Fig. 7 and Tab. 1, Tab. 2, we see that GREA performs better than α-expansion on both image restoration and stereo matching tasks, e.g., on image *venus*, α-expansion obtains the energy 2712361, while GREA obtains 2637047.

We also compare the stereo matching performance of GREA with $\alpha\beta$-swap, BPS and TRW-S, all of which have many successful applications. Tab. 2 shows the performance obtained on both truncated convex and piecewise linear functions. We see that GREA performs much better than $\alpha\beta$-swap and BPS, e.g., on image *Venus*, $\alpha\beta$-swap obtains energy 2922213 with truncated convex function, BPS achieves 3299917, while the GREA achieves the best solution with energy 2637047. We also observe that the GREA outperforms TRW-S in most cases, e.g., on image *Tsukuba*, the TRW-S obtains energy 1449548, while the GREA obtains 1434967.

7 Conclusions and discussions

In this paper, we presented two generalized range move algorithms (GRMAs) for the approximate optimization of MRFs, and extended the GRMAs to more general energy functions than only truncated convex functions. In the GRMAs, we choose subsets of labels that satisfy the submodular condition for every move, and we propose a sufficient condition for choosing the submodular labels feasibly. In the iterative optimization, we dynamically obtain the range moves by solving a set cover problem (SCP) and greatly reduce the number of unnecessary moves. We also proposed an improved graph construction for the GRMAs. The new graph construction not only runs faster than previous graph constructions but also yields better solutions. The experiments show that the GRMAs run several times faster than previous range move algorithms, while they achieve competitive solutions. The GRMAs can be regarded as a generalization of several types of move-making algorithms.

In future work, the GRMAs will be further improved. For example, although Theorem 1 provides a sufficient condition to obtain the submodular sets, it is not a necessary condition. It is of interest to investigate whether there is a better way of finding submodular sets of labels. We also observe that there are still many moves that do not lower $E(f)$ significantly in the iterations of the GRMAs. It is of interest to find those moves which lead to the biggest decrease of $E(f)$ in each iteration.

Acknowledgements This work is funded by the National Basic Research Program of China (Grant No. 2012CB316302), National Natural Science Foundation of China (Grant No. 61322209, Grant No. 61175007 and Grant No. 61403387). We thank Olga Veksler for her great help to this work, and we thank Pushmeet Kohli for his valuable comments.

References

Batra D, Kohli P (2011) Making the right moves: Guiding alpha-expansion using local primal-dual gaps. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1865–1872

Besag J (1986) On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society 48(3):259–302

Boykov Y, Jolly (2001) Interactive graph cuts for optimal boundary and region segmentation of objects in nd images. In: IEEE International Conference on Computer Vision, pp 105–112

Boykov Y, Jolly MP (2000) Interactive organ segmentation using graph cuts. In: Medical Image Computing and Computer-Assisted Intervention, pp 276–286

Boykov Y, Kolmogorov V (2003) Computing geodesics and minimal surfaces via graph cuts. In: IEEE International Conference on Computer Vision, pp 26–33

Boykov Y, Veksler O, Zabih R (2001) Fast approximate energy minimization via graph cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(11):1222–1239

Chekuri C, Khanna S, Naor J, Zosin L (2004) A linear programming formulation and approximation algorithms for the metric labeling problem. SIAM Journal on Discrete Mathematics 18(3):608–625

Feige U (1998) A threshold of ln n for approximating set cover. Journal of the ACM 45(4):634–652

Gould S, Amat F, Koller D (2009) Alphabet soup: A framework for approximate energy minimization. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 903–910

Greig D, Porteous B, Seheult AH (1989) Exact maximum a posteriori estimation for binary images. Journal of the Royal Statistical Society pp 271–279

Gridchyn I, Kolmogorov V (2013) Potts model, parametric maxflow and k-submodular functions. In: IEEE International Conference on Computer Vision, pp 2320–2327

Ishikawa H (2003) Exact optimization for markov random fields with convex priors. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(10):1333–1336

Kappes JH, Andres B, Hamprecht FA, Schnörr C, Nowozin S, Batra D, Kim S, Kausler BX, Lellmann J, Komodakis N, Rother C (2013) A comparative study of modern inference techniques for discrete energy minimization problem. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1328–1335

Kohli P, Osokin A, Jegelka S (2013) A principled deep random field model for image segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1971–1978

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

Kolmogorov V, Rother C (2007) Minimizing nonsubmodular functions with graph cuts-a review. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(7):1274–1279

Kolmogorov V, Zabin R (2004) What energy functions can be minimized via graph cuts? IEEE Transactions on Pattern Analysis and Machine Intelligence 26(2):147–159

Kumar MP (2014) Rounding-based moves for metric labeling. In: Advances in Neural Information Processing Systems, pp 109–117

Kumar MP, Koller D (2009) Map estimation of semi-metric mrfs via hierarchical graph cuts. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, pp 313–320

Kumar MP, Torr PH (2009) Improved moves for truncated convex models. In: Advances in Neural Information Processing Systems, pp 889–896

Kumar MP, Veksler O, Torr PH (2011) Improved moves for truncated convex models. The Journal of Machine Learning Research 12

Lempitsky V, Rother C, Blake A (2007) Logcut-efficient graph cut optimization for markov random fields. In: IEEE International Conference on Computer Vision, pp 1–8

Lempitsky V, Rother C, Roth S, Blake A (2010) Fusion moves for markov random field optimization. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(8):1392–1405

Liu K, Zhang J, Huang K, Tan T (2014) Deformable object matching via deformation decomposition based 2d label mrf. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 2321–2328

Liu K, Zhang J, Yang P, Huang K (2015) GRSA: Generalized range swap algorithm for the efficient optimization of mrfs. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1761–1769

Martin D, Fowlkes C, Tal D, Malik J (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: IEEE International Conference on Computer Vision, vol 2, pp 416–423

Nagarajan R (2003) Intensity-based segmentation of microarray images. IEEE Transactions on Medical Imaging 22(7):882–889

Poggio T, Torre V, Koch C (1989) Computational vision and regularization theory. Image understanding 3(1-18):111

Rother C, Kolmogorov V, Lempitsky V, Szummer M (2007) Optimizing binary mrfs via extended roof duality. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1–8

Scharstein D, Szeliski R (2002) A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision 47

Schlesinger D, Flach B (2006) Transforming an arbitrary minsum problem into a binary one. TU, Fak. Informatik

Slavłk P (1996) A tight analysis of the greedy algorithm for set cover pp 435–441

Szeliski R, Zabih R, Scharstein D, Veksler O, Kolmogorov V, Agarwala A, Tappen M, Rother C (2008) A comparative study of energy minimization methods for mrfs with smoothness-based priors. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(6):1068–1080

Tappen MF, Freeman WT (2003) Comparison of graph cuts with belief propagation for stereo using identical mrf parameters. In: IEEE International Conference on Computer Vision, pp 900–906

Veksler O (2007) Graph cut based optimization for mrfs with truncated convex priors. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1–8

Veksler O (2012) Multi-label moves for mrfs with truncated convex priors. International Journal of Computer Vision 98(1):1–14

A Proof of Theorem 1

A.1 Related lemmas and definitions

Before proving Theorem 1, we first give the following lemmas and the definition of *submodular set*.

Lemma 5 *For $b_1, b_2 > 0$, the following conclusion holds.*

$$\frac{a_1}{b_1} \geq \frac{a_2}{b_2} \Leftrightarrow \frac{a_1}{b_1} \geq \frac{a_1+a_2}{b_1+b_2} \geq \frac{a_2}{b_2}. \tag{16}$$

The proof is straightforward and we omit it.

Lemma 6 *Assuming that function $g(x)$ is convex on $[a,b]$ and there are three points $x_1, x, x_2 \in [a,b]$ satisfying $x_1 > x > x_2$, there is*

$$\frac{g(x_1)-g(x)}{x_1-x} \geq \frac{g(x_1)-g(x_2)}{x_1-x_2} \geq \frac{g(x)-g(x_2)}{x-x_2}. \tag{17}$$

Proof Since $x_1 > x > x_2$, there exists $\lambda \in (0,1)$ satisfying $x = (1-\lambda)x_1 + \lambda x_2$. Then by the definition of convex function, there is $(1-\lambda)g(x_1) + \lambda g(x_2) \geq g(x)$ and thus

$$(1-\lambda)(g(x_1)-g(x)) \geq \lambda(g(x)-g(x_2)) \tag{18}$$

Considering that $x_1 > x_2$ and $0 < \lambda < 1$, we can divide $\lambda(1-\lambda)(x_1-x_2)$ on both sides of (18) and obtain

$$\frac{g(x_1)-g(x)}{\lambda(x_1-x_2)} \geq \frac{g(x)-g(x_2)}{(1-\lambda)(x_1-x_2)} \tag{19}$$

$$\frac{g(x_1)-g(x)}{x_1-x} \geq \frac{g(x)-g(x_2)}{x-x_2}. \tag{20}$$

At last, the conclusion (17) can be proved by applying Lemma 5 to (20).

Lemma 7 *Assuming that $g(x)$ is convex on $[a,b]$ and there are four points $x_1,x_2,x_3,x_4 \in [a,b]$ satisfying $x_1 > x_3 \geq x_4$ and $x_1 \geq x_2 > x_4$, there is*

$$\frac{g(x_1)-g(x_3)}{x_1-x_3} \geq \frac{g(x_2)-g(x_4)}{x_2-x_4}. \tag{21}$$

Proof Since $g(x)$ is convex on $[a,b]$ and $x_1 > x_3 \geq x_4$, it is straightforward to obtain

$$\frac{g(x_1)-g(x_3)}{x_1-x_3} \geq \frac{g(x_1)-g(x_4)}{x_1-x_4} \tag{22}$$

by Lemma 6 (the case when $x_3 = x_4$ is trivial).

By the same way, there is

$$\frac{g(x_1)-g(x_4)}{x_1-x_4} \geq \frac{g(x_2)-g(x_4)}{x_2-x_4}. \tag{23}$$

Combining (22) and (23), we obtain the inequality (21) which completes the proof. □

Lemma 8 *Given a function $g(x)$ $(x=|\alpha-\beta|)$ on domain $X=[0,c]$, assume $g(x)$ is locally convex on interval $X_s=[a,b]$ $(0\leq a<b\leq c)$, and it satisfies $a\{g(a+1)-g(a)\} \geq g(a)-g(0)$. Then we have*

$$\frac{g(x_1)-g(x_3)}{x_1-x_3} \geq \frac{g(x_2)-g(0)}{x_2} \tag{24}$$

where x_1, x_2, $x_3 \in X_s$ where $x_3 < x_1$ and $x_2 < x_1$.

Proof Since $x_1 > x_3 \geq a$ and $x_1 \in \mathbb{N}$, we have $x_1 \geq a+1 > a$. Then considering that $x_1 > x_3 \geq a$, we can use Lemma 7 to obtain

$$\frac{g(x_1)-g(x_3)}{x_1-x_3} \geq \frac{g(a+1)-g(a)}{a+1-a} \geq \frac{g(a)-g(0)}{a}, \tag{25}$$

where the second inequality comes from $a\{g(a+1)-g(a)\} \geq g(a)-g(0)$.

If $x_2 = a$, the conclusion is obtained from (25). Otherwise, there is $x_1 > x_2 > a$ and $x_1 > x_3 \geq a$. Using Lemma 7, we obtain

$$\frac{g(x_1)-g(x_3)}{x_1-x_3} \geq \frac{g(x_2)-g(a)}{x_2-a}. \tag{26}$$

Combining (25) and (26), we can obtain

$$\begin{aligned}\frac{g(x_1)-g(x_3)}{x_1-x_3} \geq& \max\left\{\frac{g(x_2)-g(a)}{x_2-a}, \frac{g(a)-g(0)}{a}\right\}\\ \geq& \frac{g(x_2)-g(a)+g(a)-g(0)}{x_2-a+a}\\ =& \frac{g(x_2)-g(0)}{x_2},\end{aligned}$$

where the second inequality is due to Lemma 5 and this completes the proof.

Definition 1 *Given a pairwise potential $\theta(\alpha,\beta)$, we call $\mathcal{L}_s$ a submodular set of labels, if it satisfies*

$$\theta(l_{i+1},l_j)-\theta(l_{i+1},l_{j+1})-\theta(l_i,l_j)+\theta(l_i,l_{j+1}) \geq 0 \tag{27}$$

for any pair of labels l_i, $l_j \in \mathcal{L}_s (1 \leq i,j < m)$.

A.2 Proof of Theorem 1

Theorem 1 *Given a pairwise function $\theta(\alpha,\beta) = g(x)$ $(x = |\alpha-\beta|)$, assume there is an interval*[12] *$X_s = [a,b]$ $(0 \leq a < b)$ satisfying: (i) $g(x)$ is convex on $[a,b]$, and (ii) $a\cdot(g(a+1)-g(a)) \geq g(a)-g(0) \geq 0$. Then $\mathcal{L}_s = \{l_1,\cdots,l_m\}$ is a submodular subset, if $|l_i - l_j| \in [a,b]$ for any pair of labels l_i, l_j such that $l_i \neq l_j$ and $l_i, l_j \in \mathcal{L}_s$.*

Proof Since $\theta(\alpha,\beta)$ is semimetric and satisfies $\theta(\alpha,\beta) = \theta(\beta,\alpha)$, we only consider $l_i, l_{i+1}, l_j, l_{j+1} \in \mathcal{L}_s$ where $i \geq j$. Let

$$\begin{aligned}x_1 &= l_{i+1}-l_j, & x_2 &= l_{i+1}-l_{j+1},\\ x_3 &= l_i-l_j, & x_4 &= l_i-l_{j+1}\end{aligned}$$

We have $x_1 > x_2 \geq x_3 > x_4$, and $x_1-x_2 = x_3-x_4$. We can define

$$\lambda = \frac{x_3-x_4}{x_1-x_4} = \frac{x_1-x_2}{x_1-x_4},\ (0<\lambda<1) \tag{28}$$

then, we get

$$x_3 = \lambda x_1+(1-\lambda)x_4, \quad x_2 = \lambda x_4+(1-\lambda)x_1. \tag{29}$$

If $a = 0$, *i.e.* $X_s = [0,b]$ we have $x_1,x_2,x_3,x_4 \in X_s$ according to the assumption in Theorem 1. Since $g(x)$ is convex on X_s, with Eq. (29) we obtain

$$\begin{aligned}&g(x_3) \leq \lambda g(x_1)+(1-\lambda)g(x_4),\\ &g(x_2) \leq \lambda g(x_4)+(1-\lambda)g(x_1)\end{aligned} \tag{30}$$

Summing the two equations in Eq. (30), we can get

$$g(x_2)+g(x_3) \leq g(x_1)+g(x_4)$$

Thus, $\theta(l_{i+1},l_j)-\theta(l_{i+1},l_{j+1})-\theta(l_i,l_j)+\theta(l_i,l_{j+1}) \geq 0$ is satisfied for any pair of labels l_i, $l_j \in \mathcal{L}_s$.

If $a > 0$ ($X_s = [a,b]$), we prove the theorem in three cases: 1) $i = j$; 2) $i > j+1$; 3) $i = j+1$.

1. When $i = j$, we have

$$\begin{aligned}&\theta(l_{i+1},l_j)-\theta(l_{i+1},l_{j+1})-\theta(l_i,l_j)+\theta(l_i,l_{j+1})\\ =&\theta(l_{i+1},l_i)-\theta(l_{i+1},l_{i+1})-\theta(l_i,l_i)+\theta(l_i,l_{i+1})\\ =&2(g(l_{i+1}-l_i)-g(0))\\ \geq&2(g(a)-g(0)) \geq 0\end{aligned}$$

2. When $i > j+1$, we have $x_1,x_2,x_3,x_4 \in X_s$ according to the assumption in Theorem 1. Since $g(x)$ is convex on X_s, with Eq. (29) we obtain

$$\begin{aligned}&g(x_3) \leq \lambda g(x_1)+(1-\lambda)g(x_4),\\ &g(x_2) \leq \lambda g(x_4)+(1-\lambda)g(x_1)\end{aligned} \tag{31}$$

Summing the two equations in Eq. (31), we get

$$g(x_2)+g(x_3) \leq g(x_1)+g(x_4)$$

Thus, $\theta(l_{i+1},l_j)-\theta(l_{i+1},l_{j+1})-\theta(l_i,l_j)+\theta(l_i,l_{j+1}) \geq 0$ is satisfied for any pair of label $l_i, l_j \in \mathcal{L}_s$ and $i > j+1$.

3. When $i = j+1$, we have

$$\begin{aligned}x_1 &= l_{j+2}-l_j, & x_2 &= l_{j+2}-l_{j+1},\\ x_3 &= l_{j+1}-l_j, & x_4 &= 0\end{aligned}$$

[12] Here, the interval $[a,b]$ denotes the set of integers $\{x|a \leq x \leq b\}$.

Thus, we have $x_1 = x_2 + x_3$, and $x_1, x_2, x_3 \in X_s$ but $x_4 \notin X_s$.

With Lemma 8, we have

$$\frac{g(x_1) - g(x_3)}{x_1 - x_3} \geq \frac{g(x_2) - g(0)}{x_2}. \tag{32}$$

Thus we can get

$$g(x_2) + g(x_3) \leq g(x_1) + g(x_4)$$

and $\theta(l_{i+1}, l_j) - \theta(l_{i+1}, l_{j+1}) - \theta(l_i, l_j) + \theta(l_i, l_{j+1}) \geq 0$ is satisfied for any pair of labels $l_i, l_j \in \mathcal{L}_s$ and $i = j + 1$.

Therefore, $\theta(l_{i+1}, l_j) - \theta(l_{i+1}, l_{j+1}) - \theta(l_i, l_j) + \theta(l_i, l_{j+1}) \geq 0$ is satisfied for any pair of labels $l_i, l_j \in \mathcal{L}_s$. The proof is completed.

A.3 Proof of Corollary 1

Corollary 1 (Theorem 1) *Assuming the interval $[a, b]$ is a candidate interval, then $\{\alpha, \alpha + x_1, \alpha + x_1 + x_2, \cdots, \alpha + x_1 + \cdots + x_m\} \subseteq \mathcal{L}$ is a submodular set for any $\alpha \geq 0$, if $x_1, \cdots, x_m \in [a, b]$ and $x_1 + \cdots + x_m \leq b$.*

Proof Let $\mathcal{L}_s = \{\alpha, \alpha + x_1, \alpha + x_1 + x_2, \cdots, \alpha + x_1 + \cdots + x_m\}$. We consider a pair of labels α_1 and α_2, which can be any pair of distinct labels chosen in $\mathcal{L}_s$. According to the definition, there always exist p, q ($1 \leq p, q \leq m$) such that

$$|\alpha_1 - \alpha_2| = x_p + x_{p+1} + \cdots + x_q.$$

Since $x_i \in [a, b]$ for $\forall i \in [p, q]$, we have $|\alpha_1 - \alpha_2| \geq a$.
Since $x_1 + \cdots + x_m \leq b$, we have $x_p + x_{p+1} + \cdots + x_q \leq b$.
Thus, $|\alpha_1 - \alpha_2| \in [a, b]$ for any pair of labels $\alpha_1, \alpha_2 \in \mathcal{L}_s$
Thus, $\mathcal{L}_s$ is a submodular set according to Theorem 1.

B Proof of Proposition 1

Proposition 1 *Let $\mathcal{L}_1, \cdots, \mathcal{L}_k$ be a set of range swap moves, which cover all pairs of labels $l_i, l_j \in \mathcal{L}$. Let $\hat{f}$ be a local minimum obtained by these moves. Then, $\hat{f}$ is also a local minimum for $\alpha\beta$-swap.*

Proof Firstly, we assume that within one swap move on the pair of labels α, β, $\alpha\beta$-swap can achieve a better solution f^* such that $E(f^*) < E(\hat{f})$.

Let $\mathcal{L}_i$ be a move that covers α, β, *i.e.* $\alpha, \beta \in \mathcal{L}_i$. The range swap move on $\mathcal{L}_i$ minimizes:

$$E_s(f) = \sum_{p \in \mathcal{P}_{\mathcal{L}_i}} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_{\mathcal{L}_i} \neq \emptyset} \theta_{pq}(f_p, f_q) \tag{33}$$

where $\mathcal{P}_{\mathcal{L}_i}$ denotes the set of vertices whose labels belong to $\mathcal{L}_i$. We have $\mathcal{P}_\alpha, \mathcal{P}_\beta \in \mathcal{P}_{\mathcal{L}_i}$.

With the assumption, the swap move on α, β can achieve a better solution $E(f^*) < E(\hat{f})$. This means that the energy $E(\hat{f})$ can be decreased by changing the labels of some vertices $p \in \mathcal{P}_\alpha$ to β, or changing the labels of vertices $p \in \mathcal{P}_\beta$ to α. Therefore, the range swap move on $\mathcal{L}_i$ can decrease $E(\hat{f})$. This is inconsistent with the fact that $E(\hat{f})$ is a local minimum of the range swap moves.

Thus, $E(\hat{f})$ cannot be decreased by any move in the $\alpha\beta$-swap, and the proof is completed.

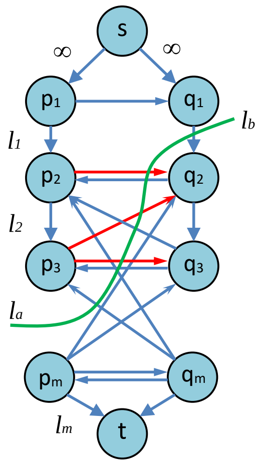


Fig. 12 The graph construction in the st-mincut problem to solve the range swap move. The edges in the st-cut are marked red when nodes p, q are assigned label l_a and l_b, respectively.

C Proof of Proposition 2

Proposition 2 *Let $\hat{f}$ be a local minimum obtained by $\alpha\beta$-swap. With the initial labeling $\hat{f}$, the range swap moves on $\mathcal{L}' = \{\mathcal{L}_1, \cdots, \mathcal{L}_k\}$ yield a local minimum $f^\dagger$ such that $E(f^\dagger) < E(\hat{f})$, unless the labeling $\hat{f}$ exactly optimizes the energy:*

$$E_s(f) = \sum_{p \in \mathcal{P}_{\mathcal{L}_i}} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_{\mathcal{L}_i} \neq \emptyset} \theta_{pq}(f_p, f_q)$$

for each $\mathcal{L}_i \subseteq \mathcal{L}'$.

Proof Assume the labeling $\hat{f}$ does not exactly optimize the energy

$$E_s(f) = \sum_{p \in \mathcal{P}_{\mathcal{L}_i}} \theta_p(f_p) + \sum_{(p,q) \in \mathcal{E}, \{p,q\} \cap \mathcal{P}_{\mathcal{L}_i} \neq \emptyset} \theta_{pq}(f_p, f_q) \tag{34}$$

where $\mathcal{L}_i \subseteq \mathcal{L}'$.

Obviously, $E(\hat{f})$ can be decreased by the range swap move on $\mathcal{L}_i$, since this move can obtain a labeling f^* which is a global minimization of $E_s(f)$ in (34).

Thus, the proof of Proposition 2 is completed.

D Proof of Lemma 1

Lemma 1 *When edges (p_a, p_{a+1}) and (q_b, q_{b+1}) are in the st-cut C, that is, f_p, f_q are assigned the labels l_a, l_b respectively, let $cut(l_a, l_b)$ denote the cost of the pairwise edges in $\mathcal{E}_3$ in the st-cut. We have the following relationship*

$$cut(l_a, l_b) = \begin{cases} \sum_{i=b+1}^{a} \sum_{j=b+1}^{i} c(p_i, q_j), & if\, l_a \geq l_b\,; \\ \sum_{i=a+1}^{b} \sum_{j=a+1}^{i} c(q_i, p_j), & if\, l_a < l_b\,. \end{cases}$$

Proof We will show the proof of the case where $l_a \geq l_b$, and the similar argument applies when $l_a < l_b$.

As the definition of st-cut, an st-cut only consists of edges going from the source's (s) side to the sink's (t) side. The cost of an st-cut is the sum of capacity of each edge in the cut. As shown in Fig. 12, if nodes p, q are assigned l_a and l_b, respectively, the st-cut is specified by the edges

$$(p_a, p_{a+1}) \cup (q_b, q_{b+1}) \cup \{(p_i, q_j), b+1 \leq i \leq a, b+1 \leq j \leq i\}.$$

where (p_a, p_{a+1}) and (q_b, q_{b+1}) are the unary edges, while (p_i, q_j) denote the pairwise edges.

As a result, the cost of the pairwise edges in the st-cut is

$$cut(l_a, l_b) = \sum_{i=b+1}^{a} \sum_{j=b+1}^{i} c(p_i, q_j), \text{ where } l_a \geq l_b. \tag{35}$$

The proof is completed.

E Proof of Lemma 2

Lemma 2 *For the graph described in Sec.5.1, Property 2 holds true.*

Proof As described in Sec.5.1, we construct a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, such that a set of nodes $\{p_1, p_2, \cdots, p_m\} \subset \mathcal{V}$ is defined for each $p \in \mathcal{P}_s$. In addition, a set of edges are constructed to model the unary and pairwise potentials.

Assume that p and q are assigned the labels l_a, l_b respectively. In other word, we have $f'_p = l_a$ and $f'_q = l_b$. For brevity, we only consider the case where $l_a \geq l_b$, and a similar argument applies when $l_a < l_b$.

We observe that the st-cut will consist of only the following edges:

$\{(p_i, p_j), b+1 \leq i \leq a, b+1 \leq j \leq i\}$.

Using Eq. 10 to sum the capacities of the above edges, we obtain the cost of the st-cut

$$\begin{aligned} cut(f'_p, f'_q) &= \sum_{i=b+1}^{a} \sum_{j=b+1}^{i} c(p_i, q_j) \\ &= \sum_{i=b+1}^{a} \sum_{j=i}^{i} c(p_i, q_j) + \sum_{i=b+1}^{a} \sum_{j=b+1}^{i-1} c(p_i, q_j) \\ &= \sum_{i=b+1}^{a} \sum_{j=i}^{i} \frac{\psi(i,j)}{2} + \sum_{i=b+1}^{a} \sum_{j=b+1}^{i-1} \psi(i,j) \end{aligned} \tag{36}$$

where $\psi(i,j) = \theta(l_i, l_{j-1}) - \theta(l_i, l_j) - \theta(l_{i-1}, l_{j-1}) + \theta(l_{i-1}, l_j)$ for $1 < j \leq i \leq m$.

Since the pairwise potentials satisfy $\theta(\alpha, \beta) = 0 \Leftrightarrow \alpha = \beta$ and $\theta(\alpha, \beta) = \theta(\beta, \alpha) \geq 0$, when $i = j$, $\psi(i,j)$ can be simplified as

$\psi(i,j) = 2\theta(l_i, l_{j-1}), i = j$.

Using the above equation, we have

$$\sum_{i=b+1}^{a} \sum_{j=i}^{i} \frac{\psi(i,j)}{2} = \sum_{i=b+1}^{a} \theta(l_i, l_{i-1}), \tag{37}$$

and

$$\begin{aligned} &\sum_{i=b+1}^{a} \sum_{j=b+1}^{i-1} \psi(i,j) \\ &= \sum_{i=b+1}^{a} \{\theta(l_i, l_b) - \theta(l_{i-1}, l_b) - \theta(l_i, l_{i-1})\} \\ &= \theta(l_{b+1}, l_b) - \theta(l_b, l_b) - \theta(l_{b+1}, l_b) + \\ &= \theta(l_a, l_b) - \sum_{i=b+1}^{a} \theta(l_i, l_{i-1}) \end{aligned} \tag{38}$$

Using Eq. 36 37 and 38, we have

$cut(f'_p, f'_q) = \theta(l_a, l_b)$

This proves that Property 2 holds true.

F Proof of Lemma 3

Lemma 3 *When the pairwise function is a truncated function $\theta(f_p, f_q) = \min\{d(|f_p - f_q|), T\}$, for the case $f'_p \in \mathcal{L}_s$ and $f'_q = f_q \notin \mathcal{L}_s$, we have the following properties:*

– *If $f_q > l_m$, we have*

$$\theta(f'_p, f'_q) \leq cut(f'_p, f'_q) \leq d(|f'_p - l_1|) + T.$$

– *If $f_q < l_1$ and $f_p \in \mathcal{L}_s$ or $f_p < l_1$,*

$$\theta(f'_p, f'_q) \leq cut(f'_p, f'_q) \leq \min\{d(|f'_p - f'_q|), d(|f'_p - l_1|) + T\}$$

– *If $f_q < l_1$ and $f_p > l_m$, we have*

$$\theta(f'_p, f'_q) \leq cut(f'_p, f'_q) \leq \min\{d(|f'_p - f'_q|) + \frac{T}{2}, d(|f'_p - l_1|) + T\}$$

Proof With Property 6, we have the cost of the st-cut is

$$cut(f'_p, f'_q) = \begin{cases} \theta(l_a, l_1) + \sum_{i=2}^{a} c(p_i, q_1) + \theta(l_1, f_q), & f_p \in \mathcal{L}_s; \\ \theta(l_a, l_1) + \sum_{i=2}^{a} c(p_i, q_1) + \theta(l_1, f_q) & \\ +\delta, & f_p \notin \mathcal{L}_s. \end{cases}$$

where $f'_p = l_a$ and $\delta = \max(0, \frac{\theta(f_p, f_q) - \theta(l_1, f_q) - \theta(f_p, l_1)}{2})$.

For brevity, we define

$$\eta = \begin{cases} 0, & f_p \in \mathcal{L}_s; \\ \delta, & f_p \notin \mathcal{L}_s, \end{cases}$$

and the cost of the st-cut can be rewritten as

$$cut(f'_p, f'_q) = \theta(l_a, l_1) + \sum_{i=2}^{a} c(p_i, q_1) + \theta(l_1, f_q) + \eta. \tag{39}$$

As described in Sec.5.2.2, the graph $\mathcal{G}$ is constructed with the following edges

$$\begin{aligned} c(p_1, q_1) &= \theta(l_1, f_q) & f_p \in \mathcal{L}_s; \\ c(a, q_1) &= \theta(l_1, f_q) + \delta & f_p \notin \mathcal{L}_s, \end{aligned}$$

$$c(p_i, q_1) = \max(0, \theta(l_i, f_q) - \theta(l_i, l_1) - \theta_\eta - \sum_{j=2}^{i-1} c(p_j, q_1)), \tag{40}$$

where we define $\theta_\eta = \theta(l_1, f_q) + \eta$ for brevity. We have

$$\begin{aligned} \theta(l_1, f_q) &\leq T \\ \theta(l_1, f_q) + \delta &= \max(\theta(l_1, f_q), \frac{\theta(f_p, f_q) + \theta(l_1, f_q) - \theta(f_p, l_1)}{2}) \\ &\leq T, \end{aligned}$$

and thus

$$\theta_\eta = \theta(l_1, f_q) + \eta \leq T. \tag{41}$$

Firstly, we prove the following inequality holds true for the case $f'_p \in \mathcal{L}_s$ and $f'_q = f_q \notin \mathcal{L}_s$.

$$\theta(l_a, f_q) \leq cut(f'_p, f'_q) \leq d(|l_a - l_1|) + T \tag{42}$$

where $f'_p = l_a$ and $f'_q = f_q$.

When $a = 1$, we have

$cut(l_1, f'_q) = \theta(l_1, f_q) + \eta$.

Using the equation above and Eq. 41, we obtain

$$\theta(l_1, f_q) \leq cut(l_1, f'_q) \leq d(|l_1 - l_1|) + T$$

and thus, the inequality (42) is true when $a = 1$.

We assume inequality (42) holds true when $a = k$ ($k \geq 2$), and we obtain the following results

$$\theta(l_k, f_q) \leq cut(l_k, f'_q) \leq d(|l_k - l_1|) + T \tag{43}$$

where

$$cut(l_k, f'_q) = \theta(l_k, l_1) + \sum_{i=2}^{k} c(p_i, q_1) + \theta_\eta. \tag{44}$$

When $a = k + 1$,

$$cut(l_{k+1}, f'_q) = \theta(l_{k+1}, l_1) + \sum_{i=2}^{k+1} c(p_i, q_1) + \theta_\eta. \tag{45}$$

Using Eq. 40, 44 and 45,

$$\begin{aligned} cut(l_{k+1}, f'_q) &= cut(l_k, f'_q) + \theta(l_{k+1}, l_1) + c(p_{k+1}, q_1) - \theta(l_k, l_1) \\ &= cut(l_k, f'_q) + \theta(l_{k+1}, l_1) - \theta(l_k, l_1) \\ &\quad + \max(0, \theta(l_{k+1}, f_q) - \theta(l_{k+1}, l_1) \\ &\quad + \theta(l_k, l_1) - cut(l_k, f'_q)) \end{aligned}$$

If $\theta(l_{k+1}, f_q) - \theta(l_{k+1}, l_1) - cut(l_k, f'_q) + \theta(l_k, l_1) \leq 0$, we have

$$\begin{aligned} cut(l_{k+1}, f'_q) &= cut(l_k, f'_q) + \theta(l_{k+1}, l_1) - \theta(l_k, l_1), \\ &\leq \theta(l_{k+1}, l_1) - \theta(l_k, l_1) + d(|l_k - l_1|) + T \\ &\leq d(|l_{k+1} - l_1|) + T \end{aligned}$$

and

$$\theta(l_{k+1}, f_q) \leq cut(l_k, f'_q) + \theta(l_{k+1}, l_1) - \theta(l_k, l_1),$$
$$\theta(l_{k+1}, f_q) \leq cut(l_{k+1}, f'_q).$$

If $\theta(l_{k+1}, f_q) - \theta(l_{k+1}, l_1) - cut(l_k, f'_q) + \theta(l_k, l_1) \geq 0$, we have

$$\theta(l_{k+1}, f_q) \leq cut(l_{k+1}, f'_q) = \theta(l_{k+1}, f_q) \leq d(|l_{k+1} - l_1|) + T.$$

Inequality (42) is true and the proof is completed.

Then, we prove if $f_q < l_1$ it holds true that

$$cut(f'_p, f'_q) \leq d(|l_a - f'_q|) + \eta, \tag{46}$$

where $f'_q = l_a$.

When $a = 1$, it holds true that

$$cut(l_1, f'_q) = \theta(l_1, f_q) + \eta \leq d(|l_a - f'_q|) + \eta.$$

We assume when $a = k$ ($k \geq 2$), it hold true that

$$\begin{aligned} cut(l_k, f'_q) &= \theta(l_k, l_1) + \sum_{i=2}^{k} c(p_i, q_1) + \theta(l_1, f_q) + \eta \\ &\leq d(|l_k - f'_q|) + \eta. \end{aligned}$$

When $a = k + 1$,

$$cut(l_{k+1}, f'_q) = \theta(l_{k+1}, l_1) + \sum_{i=2}^{k+1} c(p_i, q_1) + \theta_\eta. \tag{47}$$

Using Eq. 40 and 47,

$$\begin{aligned} cut(l_{k+1}, f'_q) &= cut(l_k, f'_q) + \theta(l_{k+1}, l_1) - \theta(l_k, l_1) \\ &\quad + \max(0, \theta(l_{k+1}, f_q) - \theta(l_{k+1}, l_1) \\ &\quad + \theta(l_k, l_1) - cut(l_k, f'_q)) \end{aligned}$$

If $\theta(l_{k+1}, f_q) - \theta(l_{k+1}, l_1) - cut(l_k, f'_q) + \theta(l_k, l_1) \leq 0$, we have

$$\begin{aligned} cut(l_{k+1}, f'_q) &= cut(l_k, f'_q) + \theta(l_{k+1}, l_1) - \theta(l_k, l_1), \\ &\leq \theta(l_{k+1}, l_1) - \theta(l_k, l_1) + d(|l_k - f'_q|) + \eta \end{aligned}$$

As $l_1, l_k, l_{k+1} \in \mathcal{L}_s$, we have $\theta(l_k, l_1) = d(|l_k - l_1)|$, and $\theta(l_{k+1}, l_1) = d(|l_{k+1} - l_1)|$. As $d(\cdot)$ is a convex function and $f'_q < l_1 < l_k < l_{k+1}$, using Lemma 7, we have

$$\frac{d(|l_k - f'_q|) - d(|l_k - l_1|)}{l_k - f'_q - l_k - l_1} \leq \frac{d(|l_{k+1} - f'_q|) - d(|l_{k+1} - l_1|)}{l_{k+1} - f'_q - l_{k+1} + l_1}$$
$$d(|l_k - f'_q|) - d(|l_k - l_1|) \leq d(|l_{k+1} - f'_q|) - d(|l_{k+1} - l_1|)$$

.

Therefore,

$$\begin{aligned} cut(l_{k+1}, f'_q) &\leq \theta(l_{k+1}, l_1) - \theta(l_k, l_1) + d(|l_k - f'_q|) + \eta \\ &\leq d(|l_{k+1} - f'_q|) + \eta \end{aligned}$$

If $\theta(l_{k+1}, f_q) - \theta(l_{k+1}, l_1) - cut(l_k, f'_q) + \theta(l_k, l_1) \geq 0$, we have

$$cut(l_{k+1}, f'_q) = \theta(l_{k+1}, f_q) \leq d(|l_{k+1} - f'_q|) + \eta.$$

Therefore, if $f_q < l_1$ and $f_p \in \mathcal{L}_s$ or $f_p < l_1$,

$cut(f'_p, f'_q) \leq d(|l_a - f'_q|) + \eta$, where $f'_q = l_a$

holds true and the proof is completed.

If $f_p \in \mathcal{L}_s$, we have $\eta = 0$.

If $f_q < l_1$ and $f_p < l_1$, we have $\theta(f_p, f_q) < \theta(l_1, f_q)$ and

$$\frac{\theta(f_p, f_q) - \theta(l_1, f_q) - \theta(f_p, l_1)}{2} < 0.$$

Therefore, $\delta = 0$, and $\eta = 0$.

Using the above results and inequalities (42) and (46), we obtain the following results

$$\theta(f'_p, f'_q) \leq cut(f'_p, f'_q) \leq \min\{d(|f'_p - f'_q|), d(|f'_p - l_1|) + T\}$$

for the case where $f_q < l_1$ and $f_p \in \mathcal{L}_s$ or $f_p < l_1$.

If $f_q < l_1$ and $f_p > l_m$, we have

$$\frac{\theta(f_p, f_q) - \theta(l_1, f_q) - \theta(f_p, l_1)}{2} < \frac{T}{2}.$$

Using the above results and inequalities (42) and (46), we obtain the following results

$$\theta(f'_p, f'_q) \leq cut(f'_p, f'_q) \leq \min\{d(|f'_p - f'_q|) + \frac{T}{2}, d(|f'_p - l_1|) + T\}$$

for the case where $f_q < l_1$ and $f_p > l_m$.

The proof of Lemma 2 is completed.