Combinatorial persistency criteria for multicut and max-cut

Jan-Hendrik Lange
Max Planck Institute for Informatics
Saarland University

Bjoern Andres
Max Planck Institute for Informatics
Bosch Center for AI
University of Tübingen

Paul Swoboda
Max Planck Institute for Informatics

Abstract

In combinatorial optimization, partial variable assignments are called persistent if they agree with some optimal solution. We propose persistency criteria for the multicut and max-cut problem as well as fast combinatorial routines to verify them. The criteria that we derive are based on mappings that improve feasible multicuts, respectively cuts. Our elementary criteria can be checked enumeratively. The more advanced ones rely on fast algorithms for upper and lower bounds for the respective cut problems and max-flow techniques for auxiliary min-cut problems. Our methods can be used as a preprocessing technique for reducing problem sizes or for computing partial optimality guarantees for solutions output by heuristic solvers. We show the efficacy of our methods on instances of both problems from computer vision, biomedical image analysis and statistical physics.

1. Introduction

Partitioning graphs into meaningful clusters is a fundamental problem in combinatorial optimization with numerous applications in computer vision, biomedical image analysis, machine learning, data mining and beyond. The MULTICUT problem (a.k.a. correlation clustering) and MAX-CUT problem are arguably among the most well-known combinatorial optimization problems for partitioning graphs. They enable graph clustering purely based on costs between pairs of nodes and are thus commonly employed to model image processing and segmentation tasks occurring in computer vision [34, 3, 22, 18, 5]. The following factors contribute to the significance of the MULTICUT and MAX-CUT problem: The former allows for a graph clustering formulation that determines the number of clusters as part of the optimization process. The latter is essentially equivalent to binary quadratic programming, which has a variety of applications in image processing. However, as computer vision models are typically large-scale, standard solution techniques based on solving LP-relaxations do not scale well enough and are thus inapplicable. Even more so, finding globally optimal solutions with branch-and-cut is infeasible with off-the-shelf commercial solvers. Hence, the need arises for developing specialized heuristic solvers that output high-quality solutions for real-world problems, despite the worst-case NP-hardness of the MULTICUT and MAX-CUT problem. Unfortunately, although heuristic solvers often achieve a good empirical performance, they usually come without any optimality guarantees. Specifically, even if large parts of the variable assignments computed by a heuristic agree with globally optimal solutions, such optimality is not recognized.

In this work we consider combinatorial techniques for the MULTICUT and MAX-CUT problem by which we can efficiently find *persistency* (a.k.a. partial optimality). Persistent variable assignments come with a certificate that proves their agreement with a globally optimal solution. The potential benefits are twofold: (i) After running a primal heuristic, we can compute certificates which show that some variables are persistent. (ii) Even before running a heuristic, we may determine in a preprocessing step persistent variable assignments. In either case, the problem size can be reduced. In the first case, a subsequent optimization with exact solvers is accelerated. In the second case, possibly also the runtime of a heuristic algorithm is reduced and the solution quality improved.

A joint treatment of the MULTICUT and MAX-CUT problem seems instructive, since many criteria have a similar formulation and are based on analogous arguments. For the MAX-CUT problem we offer, to our knowledge, a novel approach for computing persistent variable assignments. For the MULTICUT problem our empirical evidence suggests that our method offers substantial improvement over prior work on persistency. Our empirical results are most significant for very large scale problems which current heuristics can barely handle, e.g. in biomedical image segmentation [5]. By reducing problem size via persistency, our method enables high quality solutions in such cases.

The paper is organized as follows. In Section 2 we review the related work. In Section 3 we introduce the MULTICUT and MAX-CUT problem mathematically in a shared compact formulation. In Section 4 we recap the concept of improving mappings in the context of persistency. Further, we introduce fundamental building blocks for the construction of improving mappings for the MULTICUT and MAX-CUT problem. In Section 5 and 6 we present our combinatorial persistency criteria and devise algorithms to check them. Finally, in Section 7 we evaluate our methods in numerical experiments on instances from the literature and compare to related work. Due to limited space, we provide the proofs for our results as well as running times for our experiments in the supplementary material. The supplements also contain technical improvements of our persistency criteria that were omitted from the main paper for the sake of clarity.

2. Related work

Persistency for Markov Random Fields (MRF) and, as a special case, for the binary quadratic optimization problem (a.k.a. Quadratic Pseudo-Boolean Optimization (QPBO)), has been well studied. It was observed in [31] that a natural LP-relaxation of the stable set problem has the *persistency* property: All integral variables of LP-solutions coincide with a globally optimal one. This result has been transferred to QPBO [16, 6, 7] and extended in [45] to find relational persistency, i.e. showing that some pairs of variables must have the same/different values. For higher order binary unrestricted optimization problems, the concept of roof duality can be extended to obtain further persistency results [34, 19, 26]. Going beyond the basic LP-relaxation for QPBO, persistency certificates involving tighter LP-relaxations for higher order polynomial 0/1-programs that do not possess the persistency property (i.e. integral variables need not be persistent) have been studied in [1].

For general MRFs, criteria that can be elementarily checked include Dead End Elimination (DEE) [12]. More powerful techniques generalizing DEE that still can be used for fast preprocessing can be found in [44]. The MQPBO method [24] consists of transforming multilabel MRFs to the QPBO problem and persistency results from QPBO can subsequently be used to obtain persistency for the original multilabel MRF. Persistency criteria for the multilabel Potts problem that can be efficiently checked with max-flow computations have been developed in [27, 28] and refined in [14]. More powerful criteria based on LP-relaxations have been proposed for the multilabel Potts problem in [41] and in [38, 42, 40] for general discrete MRFs. An in-depth exposition of the concept of improving mappings that is used implicitly or explicitly for all of the above MRF criteria can be found in [37]. A comprehensive theoretical discussion and comparison of the above persistency techniques can be found in [39].

There has been, to our knowledge, less work on persistency for the MULTICUT and MAX-CUT problem. For MULTICUT, the works [2, 29] proposed simple persistency criteria that allow to fix some edge assignments. We are not aware of any persistency results for MAX-CUT. Also it is not easily possible to transfer persistency results from QPBO to MAX-CUT, even though there exist straightforward transformations between these two problems. The underlying reason is that the transformation from MAX-CUT to QPBO introduces symmetries which current persistency criteria cannot handle. More specifically, known persistency criteria rely on an improving mapping, but in symmetric instances it is always possible to map a labeling to an equivalent one with the same cost by exploiting symmetries. Consequently, fixed-points of improving mappings, which amount to persistent variables, cannot be found. For the closely related (yet polynomial-time solvable) MIN-CUT problem, a family of persistency criteria were proposed in [32, 17]. They directly translate to the MAX-CUT problem and we derive them as special cases in our study below.

The more involved constraints describing the MULTICUT and MAX-CUT problem make it difficult to directly transfer some of the powerful persistency techniques that are available for MRFs. In our work we show how the framework of improving mappings developed in [37] can be used to derive persistency criteria for combinatorial problems with more complicated constraint structures, such as the MULTICUT and MAX-CUT problem, once a class of mappings that act on feasible solutions is identified. Specifically, we show that the known MULTICUT persistency criteria from [29] and the persistency criteria from [17] (transferred to the MAX-CUT problem) can be derived in our theoretical framework. Moreover, we define more powerful criteria that can find significantly more persistent variables, as shown in the experimental Section 7, yet can be evaluated efficiently. We believe that our approach of composing improving mappings from elementary mappings is instructive in the search for more persistency criteria.

3. Multicut and max-cut

Let

$$\min \langle \theta, x \rangle \quad \text{s.t.} \quad x \in X \tag{P}$$

with $X \subseteq \{0,1\}^m$ be a linear combinatorial optimization problem. In this paper, we study specific instances of (P) known as the MULTICUT and the MAX-CUT problem, which are introduced mathematically in this section. To this end, let $G = (V, E, \theta)$ be a weighted graph, where $\theta \in \mathbb{R}^E$. We distinguish non-negative and negative edges via $E = E^+ \cup E^-$ with $E^+ = \{e \in E \mid \theta_e \geq 0\}$ and $E^- = \{e \in E \mid \theta_e < 0\}$. For any two disjoint subsets of vertices $U, W \subseteq V$ let $\delta(U, W) = \{uw \in E \mid u \in U, w \in W\}$ denote the set of edges between U and W. Further, we write $\delta(U) = \delta(U, V \setminus U)$ and $E(U) = \{uv \in E \mid u, v \in U\}$.

Definition 1 (Multicuts and Cuts)**.** Let $(U_1, \ldots, U_k)$ be a partition of V, i.e. $U_1 \cup \ldots \cup U_k = V$ and $U_i \cap U_j = \varnothing$ for all i, j with $i \neq j$. The set of edges M between any pair of components of the partition, defined by

$$M = \bigcup_{1 \le i < j \le k} \delta(U_i, U_j),$$

is called a *multicut* of G. If $k = 2$, then $M = \delta(U_1) = \delta(U_2)$ is called a *cut* of G. For any set of edges $F \subseteq E$ define the incidence vector $\mathbb{1}_F \in \{0,1\}^E$ of F via

$$(\mathbb{1}_F)_e = \begin{cases} 1 & \text{if } e \in F \\ 0 & \text{else.} \end{cases}$$

We write

$$\mathsf{MC} = \left\{ \mathbb{1}_M \mid M \text{ multicut of } G \right\},$$
$$\mathsf{CUT} = \left\{ \mathbb{1}_{\delta(U)} \mid U \subseteq V \right\} \subseteq \mathsf{MC}$$

for the set of incidence vectors of multicuts, respectively cuts of G.

The MULTICUT problem is to find a multicut of minimum weight w.r.t. θ and can be written as an instance of (P) as follows:

$$\min \langle \theta, x \rangle \quad \text{s.t.} \quad x \in \mathsf{MC}. \qquad (\mathrm{P}_{\mathsf{MC}})$$

The MAX-CUT problem is to find a cut $\delta(U)$, $U \subseteq V$, of maximum weight (or equivalently of minimum weight for $-\theta$). Therefore, it can w.l.o.g. be written as an instance of (P) as follows:

$$\min \langle \theta, x \rangle \quad \text{s.t.} \quad x \in \mathsf{CUT}. \qquad (\mathrm{P}_{\mathsf{CUT}})$$

Note that we use min instead of max to conform to (P).

4. Improving mappings

In this section, we introduce improving mappings as a concept to derive partial optimality results and define elementary building blocks to construct improving mappings for the MULTICUT and MAX-CUT problem.

Definition 2 ([38]). A mapping $p\colon X \to X$ with the property

$$\langle \theta, p(x) \rangle \le \langle \theta, x \rangle \quad \forall x \in X$$

is called *improving mapping*.

An improving mapping p that maps some variable x_i to a fixed value β provides *persistency* (a.k.a. partial optimality): For each feasible element $x \in X$, applying p to x and thus fixing $x_i = \beta$ gives another element that is at least as good.

Lemma 1 (Persistency). *Let $p\colon X \to X$ be an improving mapping and $\beta \in \{0,1\}$. If*

$$p(x)_i = \beta \quad \forall x \in X,$$

then $x^*_i = \beta$ in some optimal solution x^* of (P).

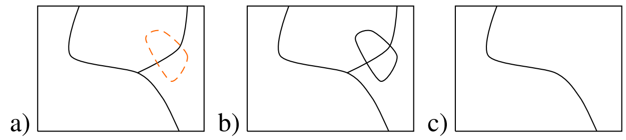

Figure 1. Illustration of elementary mappings. a) Original multicut $x \in \mathsf{MC}$ (solid lines) and connected region U (dashed line). b) Result of cut mapping $p_{\delta(U)}(x)$. c) Result of join mapping $p_U(x)$.

There are two trivial improving mappings: (i) The identity mapping $\mathrm{id}\colon x \mapsto x$. It does not provide any persistency at all, given that no variable is fixed by the constraint $x \in X$ alone. (ii) The mapping $p^*\colon x \mapsto x^*$ that maps any x to a fixed optimal solution $x^* \in \operatorname{argmin}_{x \in X} \langle \theta, x \rangle$. This mapping obviously provides the maximal persistency, i.e. it fixes all variables, but for NP-hard problems it is generally intractable to compute x^*.

We are hence interested in a middle ground: We want to find improving mappings that fix as many variables as possible (unlike id) but that are computable in polynomial time (unlike p^*). This allows us to simplify the original problem (P) by fixing the persistent variables. For the MULTICUT problem we can contract those edges that can be persistently set to 0, which allows to shrink the underlying graph. For the MAX-CUT problem, however, any value for persistent variables can be exploited for contractions, as we show below.

4.1. Elementary mappings

In order to construct improving mappings for the MULTICUT and MAX-CUT problem, we employ the elementary mappings defined in this section.

Definition 3 (Multicut mappings). Let $U \subseteq V$ be a set of nodes that induce a connected component of G.

(i) The *elementary cut mapping* $p_{\delta(U)}$ is defined as

$$p_{\delta(U)}(x) = x \vee \mathbb{1}_{\delta(U)}\,.$$

In other words, this means that $p_{\delta(U)}(x)_e = 1$ for all edges $e \in \delta(U)$ and $p_{\delta(U)}(x)_e = x_e$ otherwise.

(ii) The *elementary join mapping* p_U is defined as

$$p_U(x)_{uv} = \begin{cases} 0, & uv \in E(U) \\ 0, & \exists uv\text{-path } P \text{ such that} \\ & \forall e \in E_P : \\ & x_e = 0 \text{ or } e \in E(U) \\ x_{uv}, & \text{otherwise.} \end{cases} \qquad (1)$$

Intuitively, the elementary cut mapping $p_{\delta(U)}$ adds the cut $\delta(U)$ to the multicut defined by x. The elementary join mapping p_U merges all components that intersect with U,

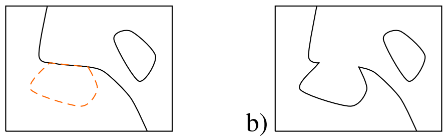

Figure 2. Illustration of symmetric difference mapping. a) Original cut $x \in \mathsf{CUT}$ (solid lines) and cut $\delta(U)$ (dashed orange line). b) Result of symmetric difference mapping $p^{\triangle}_{\delta(U)}(x)$.

cf. Figure 1. To show well-definedness of the elementary cut and join mapping rigorously, we need the following characterization of multicuts.

Fact 1 ([9])**.** A set $M \subseteq E$ is a multicut iff for every cycle C of G it holds that $|M \cap C| \neq 1$.

Lemma 2 (Well-definedness)**.** *The mappings $p_{\delta(U)}$ and p_U are well-defined, i.e.*

(i) $p_{\delta(U)} \colon \mathsf{MC} \to \mathsf{MC}$ *for any connected* $U \subseteq V$

(ii) $p_U \colon \mathsf{MC} \to \mathsf{MC}$ *for any connected* $U \subseteq V$.

The elementary mapping for the MAX-CUT problem exploits the well-known property of cuts that they are closed under taking symmetric differences (of edges).

Fact 2 ([36])**.** Let $x, y \in \mathsf{CUT}$. Then $x \triangle y \in \mathsf{CUT}$.

In particular, since $x \mapsto x \triangle y$ is an involution (i.e. its own inverse) for any cut $y \in \mathsf{CUT}$, it holds that $\mathsf{CUT} \triangle y = \{x \triangle y \mid x \in \mathsf{CUT}\} = \mathsf{CUT}$. Given an instance of MAX-CUT defined by $G = (V, E, \theta)$ and a cut $y \in \mathsf{CUT}$, this transformation of the feasible set corresponds to *switching* the signs of θ_e for all $e \in E$ with $y_e = 1$ and adding the constant $\sum_{e \in E} \theta_e y_e$ to the objective value. If y is optimal for the original instance, then $y \triangle y = 0$ is optimal for the transformed instance. Hence, whenever we want to compute persistency for $x_f = 1$, we can transform the instance to an equivalent one by applying the described switching for any cut that contains f and then checking whether $x_f = 0$ holds persistently.

Definition 4 (Symmetric Difference Mapping)**.** Let $U \subseteq V$. The *elementary symmetric difference mapping* $p^{\triangle}_{\delta(U)}$ w.r.t. $\delta(U)$ is defined as

$$p^{\triangle}_{\delta(U)}(x) = x \triangle \mathbb{1}_{\delta(U)} \,.$$

In other words, this means that $p^{\triangle}_{\delta(U)}(x)_e = 1 - x_e$ for all edges $e \in \delta(U)$ and $p^{\triangle}_{\delta(U)}(x)_e = x_e$ otherwise. The symmetric difference mapping is well-defined because of Fact 2. See Figure 2 for an illustration of $p^{\triangle}_{\delta(U)}$.

5. Persistency criteria

In this section, we propose subgraph-based criteria for finding improving mappings. We provide criteria for small connected subgraphs such as edges or triangles as well as criteria for general connected subgraphs. In Section 6, we present efficient heuristic algorithms to check the subgraph criteria proposed in this section.

First consider the instructive special case of a single edge subgraph. The following criterion has been evaluated by [29] for the MULTICUT problem.

Theorem 1 (Edge Criterion)**.** *Let $f \in E$ be an edge and $U \subseteq V$ be connected with $f \in \delta(U)$. Further, let $\beta = (1 - \operatorname{sign} \theta_f)/2$. If*

$$\begin{cases} \theta_f \geq \sum_{e \in \delta(U) \setminus \{f\}} |\theta_e|, & \mathsf{P} = \mathsf{P}_{\mathsf{MC}},\ \beta = 0 \qquad (2) \\ |\theta_f| \geq \sum_{e \in \delta(U) \cap E^+} \theta_e, & \mathsf{P} = \mathsf{P}_{\mathsf{MC}},\ \beta = 1 \qquad (3) \\ |\theta_f| \geq \sum_{e \in \delta(U) \setminus \{f\}} |\theta_e|, & \mathsf{P} = \mathsf{P}_{\mathsf{CUT}} \qquad (4) \end{cases}$$

then $x^*_f = \beta$ in some optimal solution x^* of (P).

The criteria of Theorem 1 are proved by showing that the mapping $p \colon X \to X$ that maps any $x \in X$ with $x_f \neq \beta$ to $(p_f \circ p_{\delta(U)})(x)$, $p_{\delta(U)}(x)$ or $p^{\triangle}_{\delta(U)}(x)$, respectively, is improving. Simple candidates for U in Theorem 1 are $\{u\}$ and $\{v\}$ where $f = uv$. Checking these for every edge $f \in E$ can be done in linear time. All u-v-cuts (the cuts that separate u from v) can be checked at once by minimizing the right-hand sides of (2) – (4) via max-flow techniques on the weighted graph $G^{|\cdot|} = (V, E, |\theta|)$, respectively $G^+ = (V, E^+, \theta)$ for (3). Note that the condition in (3) is less restrictive than (2). Computing a Gomory-Hu tree [13] of $G^{|\cdot|}$ or G^+ reduces the total computational effort of checking the criterion for all edges $f \in E$ to $|V| - 1$ max-flow problems.

5.1. General subgraph criteria

We give a technical lemma that allows to generalize the persistency criterion stated in Theorem 1.

Lemma 3. *Let $f \in E$ and $\beta \in \{0, 1\}$. Further, let $H = (V_H, E_H)$ be a connected subgraph of G such that $f \in E_H$. If for every $y \in \mathsf{CUT}(H)$ with $y_f = 1 - \beta$, there exists a mapping $p^y \colon X \to X$ such that for all $x \in X$ whose restriction to H agrees with y, i.e. $x_{|E_H} = y$, we have*

(i) $\langle \theta, p^y(x) \rangle \leq \langle \theta, x \rangle$

(ii) $p^y(x)_f = \beta$,

then $x^*_f = \beta$ in some optimal solution x^*.

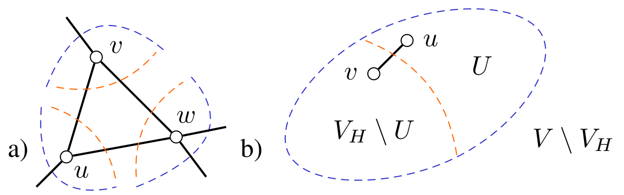

Figure 3. a) The conditions presented in Corollary 1 compare the weights of inner cuts (- -) and outer cuts (- -) around the triangle $\{u, v, w\}$. b) The conditions (9) and (10), presented in Theorem 2 and 3, compare the weights of the inner cut $\delta(U, V_H \setminus U)$ and the outer cut $\delta(V_H) = \delta(U, V \setminus V_H) \cup \delta(V_H \setminus U, V \setminus V_H)$.

The lemma follows by application of $p^y(x)$ whenever $x_{|E_H} = y$. Consider the following special case when H is a triangle subgraph. If the persistency criterion is satisfied, then for every assignment of $x_{|E_H}$ there is an improving combination of elementary mappings from Section 4.1.

Corollary 1 (Triangle Criterion)**.** *Let $\{uw, uv, vw\} \subset E$ be a triangle. Let $U \subset V$ be such that $uv, uw \in \delta(U)$, and $W \subset V$ be such that $uw, vw \in \delta(W)$.*

(i) If

$$\theta_{uw} + \theta_{uv} \geq \sum_{e \in \delta(U) \setminus \{uw, uv\}} |\theta_e|, \tag{5}$$

$$\theta_{uw} + \theta_{vw} \geq \sum_{e \in \delta(W) \setminus \{uw, vw\}} |\theta_e| \tag{6}$$

holds, then $x^*_{uw} = 0$ for some optimal solution of ($\mathsf{P_{CUT}}$).

(ii) If additionally

$$\theta_{uw} + \theta_{uv} + \theta_{vw} \geq \sum_{e \in \delta(\{u,v,w\}) \cap E^+} \theta_e \tag{7}$$

holds, then $x^*_{uw} = 0$ for some optimal solution of ($\mathsf{P_{MC}}$).

A straightforward choice for the cuts in Corollary 1 are $\delta(\{u\}), \delta(\{w\}), \delta(\{v, w\})$ and $\delta(\{u, v\})$, as depicted in Figure 3 a). It is possible to find better cuts w.r.t. costs $|\theta|$, but we are not aware of any more efficient technique than to explicitly compute them via max-flow for every triangle (unlike computing a Gomory-Hu tree to evaluate the single edge criterion for all edges).

We further employ Lemma 3 to state general subgraph criteria for the MULTICUT and MAX-CUT problem. They are proved by showing that the mapping $p\colon X \to X$, which suitably applies $p_{V_H} \circ p_{\delta(V_H)}$, respectively $p^{\triangle}_{\delta(U)}$, is improving. See Figure 3 b) for a schematic illustration.

Theorem 2 (Multicut Subgraph Criterion)**.** *Let $H = (V_H, E_H)$ be a connected subgraph of G and suppose $uv \in E_H$. If*

$$\min_{y \in \mathsf{MC}(H)} \langle \theta, y \rangle = 0 \tag{8}$$

and for all $U \subset V_H$ with $u \in U$ and $v \notin U$ it holds that

$$\sum_{e \in \delta(U, V_H \setminus U)} \theta_e \geq \sum_{e \in \delta(V_H) \cap E^+} \theta_e, \tag{9}$$

then $x^*_{uv} = 0$ in some optimal solution x^* of ($\mathsf{P_{MC}}$).

Note that the MULTICUT subgraph criterion stated in Theorem 2 is different from the edge and triangle criteria when evaluated on these special subgraphs. If $H = (f, \{f\})$ for some edge $f \in E$, then condition (9) translates to

$$\theta_f \geq \sum_{e \in \delta(f) \cap E^+} \theta_e.$$

If H is a triangle, i.e. $H = (\{u, v, w\}, \{uv, uw, vw\})$ for some vertices $u, v, w \in V$, then condition (9) translates to

$$\min\{\theta_{uv} + \theta_{uw}, \theta_{uv} + \theta_{vw}, \theta_{uw} + \theta_{vw}\} \geq \sum_{e \in \delta(\{u,v,w\}) \cap E^+} \theta_e.$$

Theorem 3 (Max-Cut Subgraph Criterion)**.** *Let $H = (V_H, E_H)$ be a connected subgraph of G and suppose $uv \in E_H$. If for all $U \subset V_H$ with $u \in U$ and $v \notin U$ it holds that*

$$\sum_{e \in \delta(U, V_H \setminus U)} \theta_e \geq \min\left\{ \sum_{e \in \delta(U, V \setminus V_H)} |\theta_e|, \sum_{e \in \delta(V_H \setminus U, V \setminus V_H)} |\theta_e| \right\}, \tag{10}$$

then $x^*_{uv} = 0$ in some optimal solution x^* of ($\mathsf{P_{CUT}}$).

Note that if H is a single edge or a triangle, the subgraph criterion stated in Theorem 3 specializes to the edge criterion, respectively triangle criterion, where only the cuts $\delta(\{u\}), \delta(\{v\})$, respectively $\delta(\{u\}), \delta(\{w\}), \delta(\{v, w\})$ and $\delta(\{u, v\})$ are considered.

6. Algorithms

In this section we devise algorithms that verify, for a given instance $G = (V, E, \theta)$ of the MULTICUT or MAX-CUT problem, the persistency criteria presented in Section 5.

The edge and triangle criteria can be checked explicitly for all edges, respectively triangles of G. Note that listing all triangles of a graph can be done efficiently [35].

Therefore, we focus here on developing efficient algorithms that find subgraphs H which qualify for the criteria from Theorem 2 and 3. Specifically, we propose routines that (i) check for a given connected subgraph H whether some persistency criteria apply and (ii) find good candidates for H. Our method applies all subroutines repeatedly until no more persistent edges are found.

6.1. Subgraph evaluation

Let $H = (V_H, E_H)$ be a subgraph of G that we want to check for persistency condition (9), respectively (10). Now, for a given edge $uv \in E_H$, we can determine if (9) holds true for all $U \subset V_H$ with $u \in U$ and $v \notin U$ by minimizing the left-hand side w.r.t. U. In contrast, for (10), we also need to simultaneously maximize the right-hand side, since it depends on U as well. Obviously, minimizing the left-hand side (of either (9) or (10)) means finding a minimum u-v-cut w.r.t. θ. Further, since the right-hand sides are non-negative, the minimum u-v-cut must have non-negative weight. However, in general the weights θ on H may be negative, which renders both optimization problems hard in general.

For this reason, we simplify the problem by restriction to suitable subgraphs H that satisfy Assumption 1 below. We shall see subsequently how to utilize this condition. In order to state Assumption 1 rigorously, we need to briefly recap the following integer linear programming (ILP) formulation of the MULTICUT problem.

The MULTICUT problem can be stated equivalently to (P_{MC}) as finding the minimum weight edge set w.r.t. $|\theta|$ that covers every cycle with exactly one negative edge, the so-called *erroneous* or *conflicted* cycles [11, 29]. The corresponding ILP formulation reads

$$\begin{aligned} &\min_{\hat{x}} \langle |\theta|, \hat{x} \rangle + \sum_{e \in E^-} \theta_e && (11)\\ &\text{s.t.} \sum_{e \in E_C} \hat{x}_e \geq 1, \quad \forall \text{ conflicted } C \\ &\hat{x} \in \{0,1\}^E. \end{aligned}$$

The associated packing dual is the linear program

$$\begin{aligned} &\max_{\lambda} \langle \mathbb{1}, \lambda \rangle + \sum_{e \in E^-} \theta_e && (12)\\ &\text{s.t.} \sum_{C: e \in E_C} \lambda_C \leq |\theta_e| \qquad \forall e \in E, \\ &\lambda \geq 0. \end{aligned}$$

For any dual feasible $\lambda \geq 0$, the associated reduced costs for the primal problem (11) are given by

$$\tilde{\theta}_e = \Big(|\theta_e| - \sum_{C: e \in E_C} \lambda_C \Big) \operatorname{sign} \theta_e \qquad \forall e \in E. \qquad (13)$$

Assumption 1. Let $H = (V_H, E_H, \theta)$ be a weighted graph such that

i) The graph H has a trivial optimal MULTICUT solution $y^* = 0$, i.e. $\min_{y \in \mathsf{MC}(H)} \langle \theta, y \rangle = 0 = \langle \theta, y^* \rangle$.

ii) An optimal packing dual solution λ^* that corresponds to y^* for the MULTICUT problem on H is at hand.

Note that Assumption 1 i) also implies a trivial MAX-CUT solution, since $\mathsf{CUT} \subseteq \mathsf{MC}$. Assumption 1 has the following expedient consequence.

Lemma 4. *Let $H = (V_H, E_H, \theta)$ be a weighted graph that satisfies Assumption 1. Then the reduced costs $\tilde{\theta}$ defined by (13) satisfy $\tilde{\theta}_e \geq 0$ for all $e \in E_H$ and for any cut $\delta(U)$ of H it holds that*

$$0 \leq \sum_{e \in \delta(U)} \tilde{\theta}_e \leq \sum_{e \in \delta(U)} \theta_e. \qquad (14)$$

Our method exploits Assumption 1 and Lemma 4 as follows. First, we compute a heuristic solution to the packing dual (12) by the fast *Iterative Cycle Packing* (ICP) algorithm from [29]. Then, if the computed dual bound shows that H has a trivial MULTICUT solution (and thus the dual solution is optimal), we can compute lower bounds to the left-hand side of (9) and (10) by applying max-flow techniques on H with capacities $\tilde{\theta}$.

In the case of the MULTICUT problem, the right-hand side of (9) is constant w.r.t. U so it suffices to compute a Gomory-Hu tree on H. In the case of the MAX-CUT problem, however, this is not sufficient, since the right-hand side of (10) also depends on U. Here, after replacing θ_e by $\tilde{\theta}_e$ for all $e \in E_H$, we need to solve the following $\min \max$ problem

$$\begin{aligned} &\min_{\substack{U \subset V_H: \\ u \in U, v \notin U}} \Bigg(\sum_{e \in \delta(U, V_H \setminus U)} \tilde{\theta}_e \\ &\qquad - \min \Big\{ \sum_{e \in \delta(U, V \setminus V_H)} |\theta_e|, \sum_{e \in \delta(V_H \setminus U, V \setminus V_H)} |\theta_e| \Big\} \Bigg) \\ &= - \sum_{e \in \delta(V_H)} |\theta_e| + \min_{\substack{U \subset V_H: \\ u \in U, v \notin U}} \Bigg(\sum_{e \in \delta(U, V_H \setminus U)} \tilde{\theta}_e \\ &\qquad + \max \Big\{ \sum_{e \in \delta(U, V \setminus V_H)} |\theta_e|, \sum_{e \in \delta(V_H \setminus U, V \setminus V_H)} |\theta_e| \Big\} \Bigg). \end{aligned} \qquad (15)$$

As solving this problem exactly appears to be difficult, we propose to solve a relaxation that is obtained by replacing the inner $\max$ term with

$$\max_{\alpha \in [0,1]} \alpha \sum_{e \in \delta(U, V \setminus V_H)} |\theta_e| + (1 - \alpha) \sum_{e \in \delta(V_H \setminus U, V \setminus V_H)} |\theta_e|$$

and then swapping the order of $\min$ and $\max$. This yields

$$\begin{aligned} (15) \geq - \sum_{e \in \delta(V_H)} |\theta_e| + \max_{\alpha \in [0,1]} \min_{\substack{U \subset V_H: \\ u \in U, v \notin U}} \Bigg(\sum_{e \in \delta(U, V_H \setminus U)} \tilde{\theta}_e + \\ \alpha \sum_{e \in \delta(U, V \setminus V_H)} |\theta_e| + (1 - \alpha) \sum_{e \in \delta(V_H \setminus U, V \setminus V_H)} |\theta_e| \Bigg). \end{aligned}$$

The right-hand side is the maximization of a concave, non-smooth function on the unit interval, which can be solved efficiently with the bisection method. In every iteration, the inner minimization problem needs to be solved for a fixed $\alpha \in [0, 1]$, which can be formulated again as a max-flow problem.

For solving the max-flow problems that occur in our method, we use Boykov-Kolmogorov's algorithm with reused search trees [8, 25]. For computing Gomory-Hu trees, we use a parallelized implementation of Gusfield's algorithm [15, 10].

6.2. Finding candidate subgraphs

To efficiently find good candidate subgraphs, we employ the following strategy. First, we compute a primal feasible solution $\bar{x} \in X$ by a fast heuristic method such as greedy edge contraction algorithms [20, 23]. If the heuristic solution $\bar{x}$ is reasonably good, then many components defined by $\bar{x}$ should be close to optimal. Thus, in the case of the MULTICUT problem, the components may already serve as candidate subgraphs. In the case of the MAX-CUT problem, we use $\bar{x}$ to transform the instance by the switching operation described in Section 4.

Then, we compute a heuristic packing dual solution $\bar{\lambda}$ by ICP for the entire graph $G = (V, E, \theta)$. The candidate subgraphs are determined as the connected components of the positive residual graph $(V, \{e \in E \mid \tilde{\theta}_e > 0\})$, where $\tilde{\theta}$ is defined as before in (13). The intuition behind this strategy is that, by construction, the edges within the subgraphs have relatively higher weight than the outgoing edges. This facilitates the application of the conditions (9) and (10).

Reduced cost fixing. Further, whenever both a primal solution and dual solution are available, we use the following technique known as *reduced cost fixing* [4] to determine additional persistent variables. Let $\gamma = \langle \theta, \bar{x} \rangle - \langle \mathbb{1}, \bar{\lambda} \rangle - \sum_{e \in E^-} \theta_e$ denote the duality gap of the primal-dual solution pair and suppose $\gamma < \tilde{\theta}_f$ for some $f \in E$. Then, it follows that $x_f = 1$ cannot be optimal and thus we can fix $x_f = 0$.

7. Experiments

In order to study the effectiveness of our methods, we evaluate them on a collection of more than 200 instances from the literature. The size of the instances ranges from a few hundred to hundreds of millions of variables (edges). We measure and compare the average relative size reduction of test instances that is obtained by applying our algorithms.

Instances. For the MULTICUT problem we use segmentation and clustering instances from the OpenGM benchmark [21] as well as biomedical segmentation instances provided by the authors of [5] and [33]. The dataset *Image Segmentation* contains planar graphs that are constructed from superpixel adjacencies of photographs. The *Knott-3D* data sets contains non-planar graph arising from volume images acquired by electron microscopy. The set *Modularity Clustering* contains complete graphs constructed from clustering problems on small social networks. The *CREMI* data sets contain supervoxel adjacency graphs obtained from volume image scans of neural tissue. The *Fruit-Fly* instances were generated from volume image scans of fruit fly brain matter. The global problem is the largest instance in this study with roughly 650 million variables. It represents the current limit of what can be tackled by state-of-the-art local search algorithms. The instances *Level 1–4* are progressively simplified versions of the global problem obtained via block-wise domain decomposition [33].

Table 1. The table gathers for each data set the number of instances ($\#I$), the graph sizes and instance type (P).

Data set	$\#I$	$\lvert V\rvert$	$\lvert E\rvert$	P
Image Segmentation	100	156–3764	439–10970	P_{MC}
Knott-3D-150	8	572–972	3381–5656	P_{MC}
Knott-3D-300	8	3846–5896	23k–36k	P_{MC}
Knott-3D-450	8	15k–17k	94k–107k	P_{MC}
Knott-3D-550	8	27k–31k	173k–195k	P_{MC}
Modularity Clustering	6	34–115	561–6555	P_{MC}
CREMI-small	3	20k–35k	170k–235k	P_{MC}
CREMI-large	3	430k–620k	3.2M–4.1M	P_{MC}
Fruit-Fly Level 1–4	4	5M–11M	28M–72M	P_{MC}
Fruit-Fly Global	1	90M	650M	P_{MC}
Ising Chain	30	100–300	4950–44850	P_{CUT}
2D Torus	9	100–400	200–800	P_{CUT}
3D Torus	9	125–343	375–1029	P_{CUT}
Deconvolution	2	1001	11k–34k	P_{CUT}
Super Resolution	2	5247	15k–25k	P_{CUT}
Texture Restoration	4	7k–22k	59k–195k	P_{CUT}

Table 2. For each dataset the table reports the average fraction of remaining nodes and edges after applying our method, respectively the method from [29] (lower is better). †Results for *Fruit-Fly Global* are without ICP-based candidate subgraphs.

	Our		[29]	
Data set	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert V\rvert$	$\lvert E\rvert$
Image Seg.	27.7%	27.4%	63.7%	62.7%
Knott-3D-150	9.7%	9.6%	75.2%	88.3%
Knott-3D-300	54.8%	61.6%	76.7%	91.6%
Knott-3D-450	66.9%	77.6%	77.6%	92.4%
Knott-3D-550	67.8%	79.0%	77.8%	92.6%
Mod. Clustering	88.7%	80.6%	92.0%	85.1%
CREMI-small	33.8%	31.9%	76.6%	75.3%
CREMI-large	44.0%	44.2%	83.7%	86.6%
Fruit-Fly Level 1–4	8.7%	9.6%	24.6%	27.9%
Fruit-Fly Global†	56.3%	51.8%	77.9%	74.5%

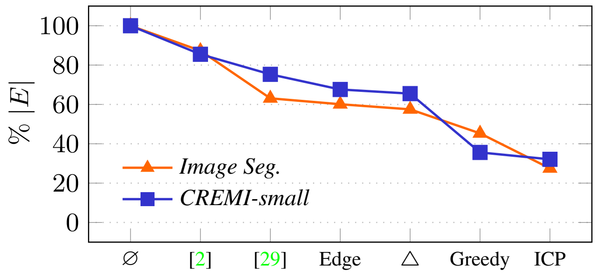

Figure 4. The figure shows the average fraction of remaining variables after shrinking the instance with progressively more expensive persistency criteria. The criteria added are from left to right: none [∅], connected components of G^+ [2], single node cuts [29], edge subgraphs [Edge], triangle subgraphs [△], greedy subgraphs [Greedy], ICP candidate subgraphs and reduced cost fixing [ICP].

For the MAX-CUT problem we use two different types of instances. (i) The datasets *Ising Chain*, *2D Torus* and *3D Torus* contain instances that stem from applications in statistical physics [30]. The instances in *Ising Chain* assume a linear order on the nodes. For any pair of nodes there is an edge with an associated weight. The absolute values of the weights decrease exponentially with the distance of the nodes in the linear order. The instances in *2D Torus* and *3D Torus* are defined on toroidal grid graphs in two, resp. three dimensions with Gaussian distributed weights. (ii) The datasets *Deconvolution*, *Super Resolution* and *Texture Restoration* contain QPBO instances originating from image processing applications [34, 43] that are converted to our formulation of the MAX-CUT problem. The transformation introduces an additional node that is connected with all other nodes. A cut (uncut) edge to the additional node signifies label 0 (resp. 1). The instance size statistics for all data sets are summarized in Table 1.

Results. In Table 2, we report for the MULTICUT instances the average graph sizes after shrinking the instances with our algorithms from Section 6. In Figure 4, the contributions of the individual persistency criteria are separated and compared. It can be seen from Table 2 and Figure 4 that our criteria enable finding substantially more persistent variables than the prior work [2, 29]. In relation to the graph sizes after shrinking with the baseline [29], our method achieves an additional size reduction of about 30–60% for the large *CREMI* and *Fruit-Fly* instances. This shows that our algorithms find persistent variable assignments that are harder to detect than with the criteria from prior work.

In Table 3 we report for the MAX-CUT instances the average graph size reduction on each dataset. For the QPBO instances we compare to the QPBO method [34]. For the original MAX-CUT instances we are unaware of any baseline method and the QPBO method is not applicable. In Figure 5 we compare the contribution of the different subgraph criteria. It can be seen that our method solves all *Ising Chain* instances to optimality, which is facilitated by their particular distribution of the weights. On *2D Torus* we achieve substantial size reductions and on the denser *3D Torus* instances we find few persistencies. Our results on the QPBO instances are on a par with [34] for *Deconvolution* and *Super Resolution* while our method is less effective for *Texture Restoration*.

Table 3. For each dataset, the table reports the average fraction of remaining nodes and edges after applying our method, respectively the QPBO method [34] (lower is better). Note that the latter is not applicable to original MAX-CUT instances due to symmetries.

	Our		[34]	
Data set	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert V\rvert$	$\lvert E\rvert$
Ising Chain	0.0%	0.0%	n/a	n/a
2D Torus	23.6%	27.9%	n/a	n/a
3D Torus	94.8%	98.1%	n/a	n/a
Deconvolution	61.0%	56.5%	61.0%	56.5%
Super Resolution	0.0%	0.0%	0.2%	0.1%
Texture Restoration	98.4%	98.5%	58.8%	57.3%

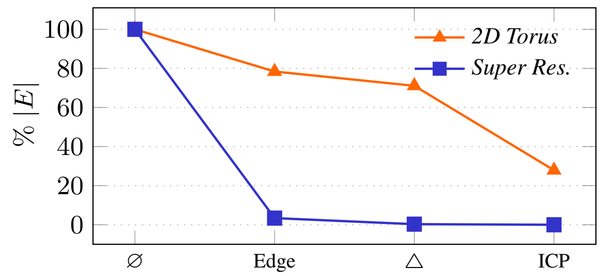

Figure 5. The figure shows the average fraction of remaining variables after shrinking the instance with progressively more expensive persistency criteria. The criteria added are from left to right: none [∅], edge subgraphs [Edge], triangle subgraphs [△], ICP candidate subgraphs and reduced cost fixing [ICP].

8. Conclusion

We have presented combinatorial persistency criteria for the MULTICUT and MAX-CUT problem. Moreover, we have devised efficient algorithms to check our criteria. For MULTICUT our method achieves a substantial improvement over prior work when evaluated on common benchmarks as well as practical instances. For MAX-CUT we are, to the best of our knowledge, the first to propose an algorithm that computes persistent variable assignments for the general problem. For the special case of QPBO problems, our method matches the performance of prior work on some instances. Our results demonstrate the feasibility of computing persistent variable assignments for NP-hard graph cut problems in practice. Besides acquiring partial optimality guarantees, our approach is a helpful tool for shrinking problem sizes and thus essential toward identifying globally optimal solutions.

References

[1] Warren P. Adams, Julie Bowers Lassiter, and Hanif D. Sherali. Persistency in 0-1 polynomial programming. *Mathematics of Operations Research*, 23(2):359–389, 1998.

[2] Amir Alush and Jacob Goldberger. Ensemble segmentation using efficient integer linear programming. *TPAMI*, 34(10):1966–1977, 2012.

[3] Bjoern Andres, Jörg H. Kappes, Thorsten Beier, Ullrich Köthe, and Fred A. Hamprecht. Probabilistic image segmentation with closedness constraints. In *ICCV*, 2011.

[4] Egon Balas and Clarence H. Martin. Pivot and complement-a heuristic for 0-1 programming. *Management Science*, 26(1):86–96, 1980.

[5] Thorsten Beier, Constantin Pape, Nasim Rahaman, Timo Prange, Stuart Berg, Davi D. Bock, Albert Cardona, Graham W. Knott, Stephen M. Plaza, Louis K. Scheffer, Ullrich Koethe, Anna Kreshuk, and Fred A. Hamprecht. Multicut brings automated neurite segmentation closer to human performance. *Nature Methods*, 14(2):101–102, 2017.

[6] Endre Boros and Peter L. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 123(13):155 – 225, 2002.

[7] Endre Boros, Peter L. Hammer, Richard Sun, and Gabriel Tavares. A max-flow approach to improved lower bounds for quadratic unconstrained binary optimization (qubo). *Discrete Optimization*, 5(2):501 – 529, 2008. In Memory of George B. Dantzig.

[8] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *TPAMI*, 26(9):1124–1137, 2004.

[9] Sunil Chopra and M.R. Rao. The partition problem. *Mathematical Programming*, 59(1–3):87–115, 1993.

[10] Jaime Cohen, Luiz A. Rodrigues, Fabiano Silva, Renato Carmo, André L. P. Guedes, and Elias P. Duarte. Parallel implementations of gusfield's cut tree algorithm. In *Algorithms and Architectures for Parallel Processing*, pages 258–269, 2011.

[11] Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2–3):172–187, 2006.

[12] Johan Desmet, Marc De Maeyer, Bart Hazes, and Ignace Lasters. The dead-end elimination theorem and its use in protein side-chain positioning. *Nature*, 356(6369):539, 1992.

[13] R. E. Gomory and T. C. Hu. Multi-Terminal Network Flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

[14] Igor Gridchyn and Vladimir Kolmogorov. Potts model, parametric maxflow and k-submodular functions. In *ICCV*, 2013.

[15] Dan Gusfield. Very simple methods for all pairs network flow analysis. *SIAM J. Comput.*, 19(1):143–155, 1990.

[16] Peter L. Hammer, Pierre Hansen, and Bruno Simeone. Roof duality, complementation and persistency in quadratic 0–1 optimization. *Mathematical Programming*, 28(2):121–155, 1984.

[17] Monika Henzinger, Alexander Noe, Christian Schulz, and Darren Strash. Practical minimum cut algorithms. In *Proceedings of the Twentieth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 48–61. SIAM, 2018.

[18] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*, pages 34–50, 2016.

[19] Fredrik Kahl and Petter Strandmark. Generalized roof duality. *Discrete Applied Mathematics*, 160(16-17):2419–2434, 2012.

[20] Sera Kahruman-Anderoglu, Elif Kolotoglu, Sergiy Butenko, and Illya V. Hicks. On greedy construction heuristics for the MAX-CUT problem. *International Journal of Computational Science and Engineering (IJCSE)*, 3(3):211–218, 2007.

[21] Jörg H. Kappes, Björn Andres, Fred A. Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X. Kausler, Thorben Kröger, Jan Lellmann, Nikos Komodakis, Bogdan Savchynskyy, and Carsten Rother. A comparative study of modern inference techniques for structured discrete energy minimization problems. *IJCV*, 115(2):155–184, 2015.

[22] Margret Keuper, Bjoern Andres, and Thomas Brox. Motion trajectory segmentation via minimum cost multicuts. In *ICCV*, 2015.

[23] Margret Keuper, Evgeny Levinkov, Nicolas Bonneel, Guillaume Lavoué, Thomas Brox, and Bjoern Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *ICCV*, 2015.

[24] P. Kohli, A. Shekhovtsov, C. Rother, V. Kolmogorov, and P. Torr. On partial optimality in multi-label MRFs. In *ICML*, 2008.

[25] Pushmeet Kohli and Philip H. S. Torr. Effciently solving dynamic markov random fields using graph cuts. In *ICCV*, pages 922–929, 2005.

[26] Vladimir Kolmogorov. Generalized roof duality and bisubmodular functions. *Discrete Applied Mathematics*, 160(4):416 – 426, 2012.

[27] Ivan Kovtun. Partial optimal labeling search for a np-hard subclass of (max,+) problems. In Bernd Michaelis and Gerald Krell, editors, *Pattern Recognition*, pages 402–409, 2003.

[28] Ivan Kovtun. Sufficient condition for partial optimality for (max,+)-labeling problems and its usage. *Control systems and machines*, (2):35–42, 2011.

[29] Jan-Hendrik Lange, Andreas Karrenbauer, and Bjoern Andres. Partial optimality and fast lower bounds for weighted correlation clustering. In *ICML*, 2018.

[30] Frauke Liers, Michael Jünger, Gerhard Reinelt, and Giovanni Rinaldi. *Computing Exact Ground States of Hard Ising Spin Glass Problems by Branch-and-Cut*, chapter 4, pages 47–69. Wiley-Blackwell, 2005.

[31] George L Nemhauser and Leslie Earl Trotter. Vertex packings: structural properties and algorithms. *Mathematical Programming*, 8(1):232–248, 1975.

[32] M. Padberg and G. Rinaldi. An efficient algorithm for the minimum capacity cut problem. *Mathematical Programming*, 47(1):19–36, May 1990.

[33] Constantin Pape, Thorsten Beier, Peter Li, Viren Jain, Davi D. Bock, and Anna Kreshuk. Solving large multicut problems for connectomics via domain decomposition. In *ICCV Workshops*, 2017.

[34] Carsten Rother, Vladimir Kolmogorov, Victor S. Lempitsky, and Martin Szummer. Optimizing binary MRFs via extended roof duality. In *CVPR*, 2007.

[35] Thomas Schank and Dorothea Wagner. Finding, counting and listing all triangles in large graphs, an experimental study. In *Experimental and Efficient Algorithms, 4th International Workshop, (WEA), Proceedings*, pages 606–609, 2005.

[36] Alexander Schrijver. *Combinatorial Optimization*. Springer, 2003.

[37] Alexander Shekhovtsov. *Exact and Partial Energy Minimization in Computer Vision*. PhD Thesis CTU–CMP–2013–24, Center for Machine Perception, K13133 FEE Czech Technical University in Prague, Prague, Czech Republic, 2013.

[38] Alexander Shekhovtsov. Maximum persistency in energy minimization. In *CVPR*, 2014.

[39] Alexander Shekhovtsov. Higher order maximum persistency and comparison theorems. *Computer vision and image understanding*, 143(C):54–79, 2 2016.

[40] A. Shekhovtsov, P. Swoboda, and B. Savchynskyy. Maximum persistency via iterative relaxed inference with graphical models. *TPAMI*, PP(99), 2017.

[41] Paul Swoboda, Bogdan Savchynskyy, Jörg H. Kappes, and Christoph Schnörr. Partial optimality via iterative pruning for the Potts model. In *Scale Space and Variational Methods (SSVM)*, 2013.

[42] Paul Swoboda, Alexander Shekhovtsov, Jörg H. Kappes, Christoph Schnörr, and Bogdan Savchynskyy. Partial optimality by pruning for MAP-inference with general graphical models. *TPAMI*, 38(7):1370–1382, 7 2016.

[43] Tanmay Verma and Dhruv Batra. Maxflow revisited: An empirical comparison of maxflow algorithms for dense vision problems. In *BMVC*, 2012.

[44] Chen Wang and Ramin Zabih. Relaxation-based preprocessing techniques for Markov Random Field inference. In *CVPR*, 2016.

[45] Di Wang and Robert Kleinberg. Analyzing quadratic unconstrained binary optimization problems via multicommodity flows. *Discrete Applied Mathematics*, 157(18):3746 – 3753, 2009.