Higher-order Segmentation via Multicuts

Jörg Hendrik Kappes[*1], Markus Speth [†2], Gerhard Reinelt[‡2] and Christoph Schnörr[§1]

[1]Department of Mathematics and Computer Science, Image and Pattern Analysis Group, Heidelberg University
[2]Department of Mathematics and Computer Science, Discrete and Combinatorial Optimization Group, Heidelberg University

Abstract

Multicuts enable to conveniently represent discrete graphical models for unsupervised and supervised image segmentation, based on local energy functions that exhibit symmetries. The basic Potts model and natural extensions thereof to higher-order models provide a prominent class of representatives, that cover a broad range of segmentation problems relevant to image analysis and computer vision. We show how to take into account such higher-order terms systematically in view of computational inference, and present results of a comprehensive and competitive numerical evaluation of a variety of dedicated cutting-plane algorithms. Our results reveal ways to evaluate a significant subset of models globally optimal, without compromising runtime. Polynomially solvable relaxations are studied as well, along with advanced rounding schemes for post-processing.

1 Introduction

1.1 Overview, Motivation

The segmentation problem, also known as partitioning, clustering, or grouping, is a fundamental problem of image analysis. Applications include unsupervised image partitioning [5,28], task-specific image partitioning [29], semantic image segmentation [24, 34], and modularity clustering in network analysis [12].

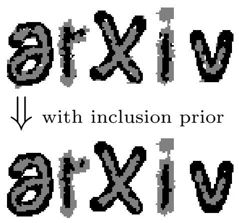
Common problem representations are based on a graph $G = (V, E)$, where nodes V relate to raw data on an image grid or extracted feature vectors, and edges E define a neighborhood structure of the nodes. A segmentation of a graph can be represented either by

(i) assigning to each node $v \in V$ a label, or by

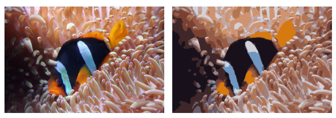
(ii) a multicut given by a subset of *active edges* $E' \subseteq E$, resulting in a partition of the set of nodes V.

One commonly distinguishes *supervised* and *unsupervised* segmentation. In the former case, the number of classes represented by labels is known, together with a function measuring how likely features associated with nodes belong to each class. In the latter unsupervised case, such information is absent. This introduces ambiguities of the representation (i) since permuting the labels results in the same segmentation. Representation (ii) does not exhibit such symmetries and is therefore particularly appealing in the unsupervised case.

(a) unsupervised segmentation



(b) inclusion prior


(c) supervised segmentation

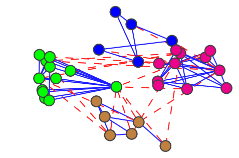

(d) modularity clustering

Figure 1: The presented framework covers **(a)** unsupervised and **(c)** supervised segmentation problems. In the former case, the number of components (clusters) of the partition is unknown. In the latter example, the image is partitioned (labelled) by assigning pixels to 12 predefined colors classes, taking spatial context into account. **(b)** By including higher order terms into the graphical model, segments can be enforced to include each other so as to respect topological prior knowledge. **(d)** Illustration of another example of a broad range of applications covered by the framework: graph partitioning by modularity clustering.

Accordingly, this paper focuses on the segmentation problem as a *multicut problem*, on the polyhedral representation of valid multicuts resulting in partitions of a given image [14,15,18], and on a computational approach to take into account the corresponding constraints efficiently.

Specifically, we consider objective functions for the segmentation problem of the form $J(x) = \sum_f \varphi_f(x_{ne(f)})$ – see Sec. 2 for details – where all higher-order terms are invariant to label permutations. For second-order terms this is equivalent to generalized Potts models that may involve negative couplings between adjacent nodes. Higher-order terms will be handled by additional auxiliary variables and few additional constraints that do not interfere with the constraints defining

[*]kappes@math.uni-heidelberg.de
[†]markus.speth@informatik.uni-heidelberg.de
[‡]gerhard.reinelt@informatik.uni-heidelberg.de
[§]schnoerr@math.uni-heidelberg.de

valid multicuts. Consequently, cutting-plane methods can be uniformly used for all models as demonstrated by comprehensive numerical evaluations.

In this connection, the present paper provides a systematic comparison of different separation strategies for computer vision applications. In particular, we find that

(i) odd-wheel inequalities do not tighten the relaxation as expected, in view of results for highly connected *non*-computer vision models [39],

(ii) integer linear programming subroutines work overall best, but

(iii) novel extensions for separation procedures as suggested in this paper are indispensable for efficient usage.

Taking these aspects into account improves runtime by at least a factor of 2.

The supervised segmentation problem will be considered as well in terms of finding an optimal multicut with at most k labels, which is known as the *multiway cut problem.* Compared to the standard (I)LP representation of such problems our approach is considerably more memory efficient and able to provide globally optimal solutions for many computer vision problems in reasonable runtime [24, 26].

Fig. 1 provides an overview and illustrations of the models studied in this paper.

1.2 Related Work

In the **unsupervised case**, the multicut polytope has recently become a focal point of research in computer vision. Major aspects of current work include closedness constraints for image segmentation [5,7], contour completion [36], ensemble segmentation [3,36], and the convex hull of feasible multicuts from the optimization point of view [25, 29, 43].

Regarding the latter viewpoint, some authors considered primal *linear program (LP)* relaxations solved by cutting-plane methods [28, 29]. Yarkony [43] suggested a Lagrangian relaxation for planar graphs based on a problem decomposition into binary planar max-cut problems. Others [3, 5, 25, 36] resorted to *integer linear programs (ILPs)* as inner-loop solver within the cutting-plane formulation. While this has exponential runtime in the worst case, it may be expected to work fast in many applications. A comparison of these methods and variants was missing so far, however.

In the **supervised case**, representation (i) above prevails for the image segmentation problem [30]. Accordingly, the *marginal polytope* has become a focal point of research with respect to relaxations and approximate inference for image labeling [32, 41, 42].

Alternatively, greedy move-making algorithms like α-expansion [11] or FastPD [33] have become established methods that are widely applied.

Methods that solve the *multiway cut problem* [14] have been considered somewhat misleadingly as computationally intractable for computer vision problems [10]. While in general this problem is known to be NP-hard [17], for few special cases, e.g., for planar graphs, exact polynomial-time algorithms are known [16, 35].

A connection of some relaxation of the second-order multiway cut problem to variational approaches using anisotropic variants of total variation, and to the linear programming relaxation over the local polytope, has been pointed out by Osokin et al. [40] and Nieuwenhuis et al. [37].

Recently, the authors [25] presented a cutting-plane approach to solve the multiway cut problem for various problem instances from computer vision. Globally optimal results for benchmark datasets were reported [24, 26] that compare remarkable well also in terms of runtime to state-of-the-art methods for approximate inference.

1.3 Contribution

We present a general framework for multicut problems, which includes Potts models as a special case. For the first time, we systematically compare different types of cutting-plane methods for the multicut problem in connection with computer vision applications.

Our framework also includes higher-order problems based on a new class of so-called *generalized higher-order Potts functions.* This class comprises all functions that are invariant to label permutations and thus provides a natural generalization of Potts functions.

We present several separation procedures and algorithmic variants that lead to significant speedups and either are able to solve the problems to optimality or to provide an approximative solution in guaranteed polynomial time with bounded integrality gap.

Comprehensive numerical evaluations demonstrate the basic properties of our approach and enable us to rank the different variants.

1.4 Organization

We start in Sec. 2 with the problem formulation followed by introducing multicuts and corresponding problem transformations in Sec. 3. In Sec. 4 we extend the framework to higher-order models and show how corresponding higher-order terms can be taken into account in a memory-efficient way by exploiting symmetries.

We detail separation procedures for finding violated constraints in Sec. 5 and show how they can be implemented efficiently. Rounding mechanisms will be discussed in Sec. 5.3. We conclude the framework with our cutting-plane method presented in Sec. 5.4.

Finally, we provide numerical evaluations for a large number of different models in Sec. 6, including second- and higher-order models in the supervised and unsupervised case, followed by concluding remarks in Sec. 7.

2 Problem Formulation

2.1 Basic Definitions

We consider discrete energy minimization problems given in terms of a *factor graph* $\mathcal{G} = (\mathcal{V}, \mathcal{F}, \mathcal{E})$, that is a bipartite graph with a set of variable nodes $\mathcal{V}$, a set of factors $\mathcal{F}$, and a corresponding relation $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{F}$ associating variables to factors, cf. [31].

Variable x_v assigned to node $v \in \mathcal{V}$ takes values in a discrete label-space X_v. We will use the shorthands $X_A = \bigotimes_{v \in A} X_v$ and $x_A = (x_v)_{v \in A}$ for $A \subseteq \mathcal{V}$, in particular $X = X_{\mathcal{V}}$ and $x = x_{\mathcal{V}}$. In cases where all X_v are equal we denote this label set by L.

Each factor $f \in \mathcal{F}$ has an associated function $\varphi_f : X_{ne(f)} \to \mathbb{R}$, where

$$ne(f) = \{v \in \mathcal{V} \mid (v, f) \in \mathcal{E}\} \tag{1}$$

denotes the neighborhood of the factor f, i.e., $x_{ne(f)}$ are the variables comprising f. We define the *order* of a factor by the cardinality $|ne(f)|$, e.g., pairwise factors have order 2, and the order of a model by the maximal order among all factors. We denote the set of all factors of order $\mathbb{N} \ni r \geq 1$ by $\mathcal{F}_r$. The energy function of the discrete labeling problem is then given by

$$J(x) = \sum_{f \in \mathcal{F}} \varphi_f(x_{ne(f)}), \tag{2}$$

where values of the variables x are also called *labelings.* We consider the problem to find a labeling with minimal energy, i.e.,

$$\hat{x} \in \arg\min_{x \in X} J(x), \tag{3}$$

for specific classes of energy functions.

By using factor graph models we take the structural property of energy functions explicitly into account. Additionally, we will also consider properties of the functions φ_f. Specifically, we assume that any function with order greater than one is invariant to label permutations.

Definition 2.1 (Label permutation invariant functions)**.** *A function $\varphi : L^N \to \mathbb{R}$ is called* invariant to label permutations *if $\forall x, x' \in L^N$ with $x_i = x_j \Leftrightarrow x'_i = x'_j$ the equality $\varphi(x) = \varphi(x')$ holds.*

Many problems of interest are covered by models involving functions of this class.

Below, we will use for any predicate τ the corresponding indicator function

$$\mathbb{I}(\tau) = \begin{cases} 1, & \text{if } \tau \text{ is true,} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

2.2 Supervised Case

In the supervised case we deal with energy functions (2),

$$\min_{x \in X} \sum_{f \in \mathcal{F}_1} \varphi_f(x_{ne(f)}) + \sum_{r \geq 2} \sum_{f \in \mathcal{F}_r} \varphi_f(x_{ne(f)}), \tag{P1}$$

where $\varphi_f(\cdot)$ is permutation invariant for all factors $f \in \mathcal{F}_r$, $r \geq 2$. Second-order models of this kind are known as *Potts models*, with $\mathcal{F}_r = \emptyset$ for $r > 2$ and

$$\varphi_f(x_{ne(f)}) = \beta_f \, \mathbb{I}(x_{ne(f)_1} \neq x_{ne(f)_2}), \qquad \forall f \in \mathcal{F}_2,$$

where $\beta_f \in \mathbb{R}$ is the *coupling constant* of factor f, and $ne(f)_i$, $i = 1, 2$, denotes the i-th neighbor of f. We focus on related higher-order models separately in Sec. 4.

2.3 Unsupervised Case

Contrary to the supervised problem (P1), in the unsupervised case the set of first-order factors is empty and the number of labels equals the number of variables:

$$\min_{x \in \{1, \ldots, |V|\}^{|V|}} \sum_{r \geq 2} \sum_{f \in \mathcal{F}_r} \varphi_f(x_{ne(f)}). \tag{P2}$$

In the second-order case, (P2) is known as the *pairwise correlation clustering* problem, where a set of nodes $\mathcal{V}$ has to be partitioned into clusters such that the sum of the costs of node-pairs in different clusters is minimized. As shown in [25] for the second-order case, solving problem (P2) with solvers commonly used for problem (P1), e.g., TRWS [32], does not work, since the large state-space and label permutation invariant functions cause large sets of optimal solutions.

We study in this paper efficient methods for solving both (P1) and (P2) in the general case.

3 Multicuts

3.1 Basic Definitions

For an undirected graph $G = (V, E)$, $E \subseteq V \times V$, let $\{S_1, \ldots, S_k\}$ be a partition of V, i.e., $\bigcup_{i=1}^k S_i = V$, $S_i \cap S_j = \emptyset$, and $S_i \neq \emptyset$. We call the edge set

$$\delta(S_1, \ldots, S_k) := \{uv \in E \mid \exists i \neq j : u \in S_i \text{ and } v \in S_j\} \tag{5}$$

a *multicut* and the sets S_i the *shores* of the multicut. To obtain a polyhedral representation of multicuts, we define *incidence vectors* $\chi(E') \in \{0, 1\}^{|E|}$ for each subset $E' \subseteq E$:

$$\chi_e(E') = \begin{cases} 1, & \text{if } e \in E', \\ 0, & \text{if } e \in E \setminus E'. \end{cases}$$

The *multicut polytope* $\mathrm{MC}(G)$ then is given by the convex hull

$$\mathrm{conv}\,\{\chi(\delta(S_1, \ldots, S_k)) \mid \delta(S_1, \ldots, S_k) \text{ is a multicut of } G\}.$$

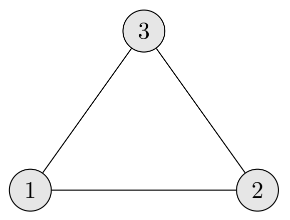
Fig. 2 shows an example. For further details on the geometry of this and related polytopes, we refer to [18].

The *multicut problem* is to find a multicut in a weighted undirected graph $G = (V, E, w)$, $w \in \mathbb{R}^{|E|}$, for which the sum of the weights of edges cut is minimal. Since all vertices (extreme points) of the multicut polytope correspond to multicuts, this amounts to solving the linear program

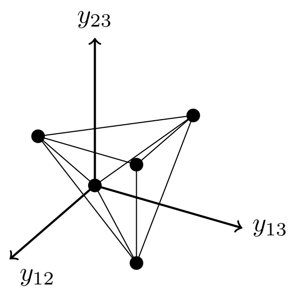
$$\min_{y \in \mathrm{MC}(G)} \sum_{e \in E} w_e \, y_e. \tag{P3}$$

In order to apply linear programming techniques, we have to represent $\mathrm{MC}(G)$ as intersection of half-spaces given by a system of affine inequalities. Since the multicut problem is NP-hard [21], we cannot expect to find a system of polynomial size. But, as we will see later, partial systems may already support effectively solving the multicut problem.

Before discussing how problem (P3) can be solved efficiently, we will show how the problems (P1) and (P2) can be transformed into problem (P3).

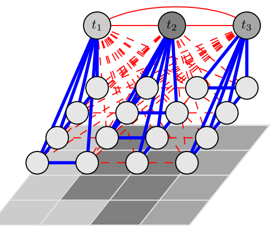


(a) Graph (K_3)



(b) Multicut polytope for K_3

Figure 2: **(a)** Illustration of the fully connected graph with three nodes K_3. **(b)** Illustration of the multicut polytope $MC(K_3)$, which has five vertices. Vertices of the polytope correspond to valid partitions and all other points of the polytope correspond to convex combinations of valid partitions. For large graphs the multicut polytope becomes huge and the describing system of inequalities intractable [18].



(a) Multicut graph for (P1)

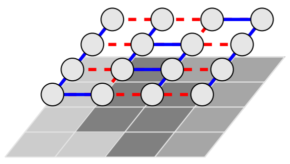

(b) Multicut graph for (P2)

Figure 3: Construction of $G = (V, E, w)$ for a 4×4-grid for **(a)** the supervised case with $L = \{1, 2, 3\}$ and **(b)** the unsupervised case. Red edges are part of the multicut, i.e., they separate shores. Blue edges join nodes of the same shore of the partition.

3.2 Multicuts for Second-order Models

To reformulate *problem (P2)* in the second-order case into a multicut problem we make use of the one-to-one correspondence between a partition and a multicut. A given factor graph $\mathcal{G}$ defines an undirected weighted graph $G = (V, E, w)$ with $V = \mathcal{V}$, $E = \{(ne(f)_1, ne(f)_2) \mid f \in \mathcal{F}_2\}$, and $w_e = \sum_{f\in\mathcal{F}_2, ne(f)=e} \beta_f$ for all $e \in E$. Accordingly, the cost of a multicut is the sum of all β_f over factors f connecting different shores, which equals the costs of (P2) – see [14] for a formal proof and Fig. 3(b) for an illustration.

Concerning *problem (P1)* for the second-order case we assume without loss of generality that $X_v = L = \{1, \dots, |L|\}$ for all $v \in \mathcal{V}$. Any labeling $x \in X$ defines a partition of $\mathcal{V}$. To write a second-order problem (P1) as a multicut problem (P3), we introduce additional terminal nodes $T = \{t_l \mid l \in L\} = \{t_1, \dots, t_{|L|}\}$ and define the undirected graph $G = (V, E)$ by $V = \mathcal{V} \cup T$, $E = \{(ne(f)_1, ne(f)_2) \mid f \in \mathcal{F}_2\} \cup \{(t, v) \mid t \in T, v \in V\} \cup \{(t_i, t_j) \mid 1 \le i < j \le |L|\}$, cf. Fig. 3(a). Thus each internal node $v \in \mathcal{V}$ is connected to all terminal nodes $t \in T$ by terminal-edges (t, v).

The terminal nodes represent the $|L|$ labels $l \in L$, and label l is assigned to variable x_v if the terminal-edge $t_l v$ is not part of the multicut, i.e., t_l and v are in the same shore. Since a single label only should be assigned to each variable, $|L|-1$ terminal-edges incident to each internal node v have to be part of the multicut. This is enforced by $|\mathcal{V}|$ additional constraints given by (22) below where we will take a closer look to classes of valid constraints. Edges between terminal nodes have weight 0 but are enforced to belong to different shores by additional constraints (23), which results in the so-called *multiway cut polytope*.

It remains to define the weights of terminal edges. Let $\mathbb{1}\mathbb{1}^\top$ be the matrix of all ones and I be the identity matrix, both of size $|L| \times |L|$ and

$$g_v(l) = \sum_{f\in ne(v)\cap\mathcal{F}_1} \varphi_f(l), \qquad l \in L. \tag{6}$$

Then the weights $w_{t_l v}$, $l \in L$, $v \in V$, are given by

$$\begin{pmatrix} w_{t_1 v} \\ \vdots \\ w_{t_{|L|} v} \end{pmatrix} = \frac{1}{|L|-1}(\mathbb{1}\mathbb{1}^\top - I) \begin{pmatrix} g_v(1) \\ \vdots \\ g_v(|L|) \end{pmatrix}. \tag{7}$$

As before we set $w_e = \sum_{f\in\mathcal{F}_2, ne(f)=e} \beta_f$ for internal edges e.

4 Multicuts for Higher-order Models

We turn to higher-order models. First, we specify a class of higher-order functions that can be treated efficiently. Next, after detailing a reduction approach, we show how such functions can be incorporated into a multicut framework. Finally, a relevant subclass of functions will be considered that can be handled even when these functions comprise factors of orders larger than several hundreds.

4.1 Label Permutation Invariant Functions

4.1.1 Definition

An important class of functions are label permutation invariant functions, whose values only depend on the partitioning of the variables rather than on the labeling, as specified by Def. 2.1. They generalize Potts functions in a natural way and are especially suited to be handled by the multicut approach.

Each possible partition of N variables is uniquely represented by a binary vector over all $N(N-1)/2$ variable-pairs. But not each binary vector $\chi \in \{0,1\}^{N(N-1)/2}$ corresponds to a partition, cf. Fig. 2. The number of possible partitions is much smaller and given by the Bell numbers $B(N)$ [1]. This observation raises the issue of an efficient representation of these functions, independent of the number of labels.

Let us denote for $i = 1, \dots, B(N)$ by $\chi_i^N \in \{0,1\}^{N(N-1)/2}$ the indicator vector of the i-th partitioning of N variables. Furthermore, we define a mapping $\tau^N : \mathbb{N}^N \to \{0,1\}^{N(N-1)/2}$ from a variable-labeling to the partition indicator. With this we can represent any label permutation invariant function over $N = |A|$ variables parameterized by $\beta \in \mathbb{R}^{B(N)}$

$$\varphi_{GP}(x_A|\beta) = \beta_i \qquad \text{if } \tau^{|A|}(x) = \chi_i^{|A|}. \tag{8}$$

We call such functions *generalized higher-order Potts functions* since they generalize (second-order) Potts functions.

4.1.2 Reduction Theorem

In order to incorporate generalized higher-order Potts functions into our multicut framework, we introduce the following reduction theorem. The basic idea of this theorem is widely used in integer nonlinear optimization, dating back to the work of Glover and Woolsey [22].

Theorem 4.1 (Reduction Theorem)**.** *Any pseudo-Boolean function* $g : \{0,1\}^M \to \mathbb{R}$ *given by* $g(z) = \prod_{i\in B^+} z_i \cdot \prod_{i\in B^-}(1-z_i)$*, with* $|B^+ \cup B^-| = M$ *and* $B^+ \cap B^- = \emptyset$*, can be transformed into an optimization problem with*
(a) *a single Boolean auxiliary variable* $s \in \{0,1\}$ *and two linear inequalities*

$$\min_{z\in\{0,1\}^M,\, s\in\{0,1\}} s \tag{9}$$

$$s.t.\ Ms \le \sum_{i\in B^+} z_i + \sum_{i\in B^-}(1-z_i) \tag{10}$$

$$s \ge 1 - M + \sum_{i\in B^+} z_i + \sum_{i\in B^-}(1-z_i) \tag{11}$$

or **(b)** *a single auxiliary variable* $s \in [0,1]$ *and* $M+1$ *inequalities*

$$\min_{z\in\{0,1\}^M,\, s\in[0,1]} s \tag{12}$$

$$s.t.\ s \le z_i \qquad \forall i \in B^+ \tag{13}$$

$$s \le (1-z_i) \qquad \forall i \in B^- \tag{14}$$

$$s \ge 1 - M + \sum_{i\in B^+} z_i + \sum_{i\in B^-}(1-z_i). \tag{15}$$

Proof. The function $g(z)$ takes the value 1 if and only if $\forall i \in B^+ : z_i = 1$ and $\forall i \in B^- : z_i = 0$, and otherwise $g(z) = 0$. It remains to show that the systems of inequalities together with $s \in \{0,1\}$ or $s \in [0,1]$ restrict the feasible set such that $s = g(z)$.

Let k denote the number of vanishing terms of $g(z)$: $k = |\{i \in B^+ \mid z_i = 0\} \cup \{i \in B^- \mid z_i = 1\}|$.
(a) Inequalities (10) and (11) imply

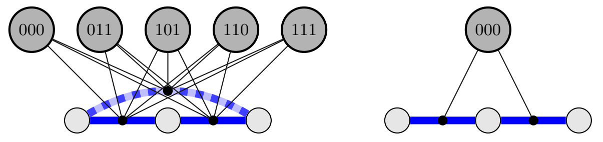
$$s \le 1 - \frac{k}{M},\ s \ge 1-k \quad \overset{s\in\{0,1\}}{\Rightarrow} \quad \begin{matrix} s=1 & \text{if } k=0, \\ s=0 & \text{if } k>0. \end{matrix}$$

(b) Inequalities (13)–(15) yield

$$\begin{matrix} (13)-(14) & \Rightarrow s \le 0 & \overset{s\in[0,1]}{\Rightarrow} & s=0 & \text{if } k>0, \\ (15) & \Rightarrow s \ge 1 & \overset{s\in[0,1]}{\Rightarrow} & s=1 & \text{if } k=0. \end{matrix}$$

□

A crucial observation is that case (b) of the reduction theorem implies integrality of s if all $z_i \in \{0,1\}$, whereas in case (a) this has to be enforced separately by $s \in \{0,1\}$. Consequently, case (b) leads to tighter relaxations by only enforcing $s \in [0,1]$.



(a) Generalized higher-order Potts (b) Higher-order Potts

Figure 4: Higher-order label permutation invariant functions are dealt with by problem reduction and additional binary auxiliary variables (Sec. 4). Corresponding constraints (black lines) enable to represent exactly the original higher-order problem. Panel **(a)** shows an example of a generalized Potts function of order three. Panel **(b)** shows an example of a Potts function of order three.

While reduction (b) thus seems to be preferable, due to a lower number of constraints, method (a) can be nevertheless appealing for some (I)LP-solver techniques, e.g., dual simplex. In our experiments, we therefore use all $M+2$ constraints (10),(11) and (13), (14) (note that (11) equals (15)), and let the solver choose the active constraint set.

4.1.3 Reduction

In order to apply Theorem 4.1 to a label permutation invariant function (8) of order $N = |A|$ we rewrite it as a pseudo-Boolean function

$$\varphi_{GP}(x_A|\beta) = \sum_{i=1}^{B(N)} \beta_i \cdot \underbrace{\prod_{j=1}^{\frac{N(N-1)}{2}} l\left([\chi_i^N]_j, [\tau^N(x)]_j\right)}_{g_i(\tau^N(x))} \tag{16}$$

with

$$l(b_1,b_2) = \begin{cases} 1-b_2, & \text{if } b_1 = 0, \\ b_2, & \text{if } b_1 = 1. \end{cases} \tag{17}$$

We apply the reduction theorem to each of the $B(N)$ binary functions $g_i(z)$, $z = \tau^N(x)$. Consequently, a function $\varphi_{GP}(x_A|\beta)$ of order N requires $B(N)$ auxiliary variables. These auxiliary variables are connected to the node-variables via the Boolean expressions $l(\cdot,\cdot)$ in (17) and correspond to the edge-variables y used in the multicut representation (P3). By this, we also get rid of difficulties caused by ambiguities of the node-label representation of a partition.

If an expression $l(\cdot,\cdot)$ has no corresponding edge e in G, we add this edge to G with weight zero.

Summing up, to include an label permutation invariant factor of order N into our multicut framework, we require at most $N(N-1)/2$ edge variables, $B(N)$ auxiliary variables, and $B(N)\cdot(N(N-1)/2+2)$ linear inequalities. These numbers are upper bounds, of course. In many cases more compact representations are obtained.

We observed in numerous experiments that additionally enforcing that all auxiliary variables corresponding to a higher-order term sum up to 1 significantly speeds up optimization. This entails to complement a single equality constraint for each higher-order term.

Fig. 4(a) illustrates an example of a factor of order three. The reduction requires $B(3) = 5$ auxiliary variables corresponding to possible partitions and, correspondingly, they are denoted by 000, ..., 111 in the figure. Constraints generated by the reduction theorem relate these auxiliary variables to the original higher-order problem. A single additional edge shown dotted in Fig. 4(a), has to be added to the graph G in this example.

4.2 Higher-order Potts Functions

4.2.1 Definition

A subclass of label permutation invariant functions are functions taking the value α_0 if all variables x_A with $A \subseteq V$ have the same label (are in the same shore) and α_1 otherwise. We call such functions *higher-order Potts functions* since they constitute the simplest generalization of (second-order) Potts functions to the higher-order case. Such functions are general enough to model the costs of a hyper-graph partitioning [28], in which the cost for a hyper-edge is included in the overall cost function if the hyper-edge connects at least two shores:

$$\varphi_{HOP}(x_A|\alpha) = \begin{cases} \alpha_0, & \text{if } \forall i,j \in A: \; x_i = x_j, \\ \alpha_1, & \text{else.} \end{cases} \tag{18}$$

4.2.2 Reduction

We can reformulate such functions in a pseudo-Boolean form:

$$\varphi_{HOP}(x_A|\alpha) = \alpha_1 + (\alpha_0 - \alpha_1) \prod_{e \in E_A} (1 - y_e) \tag{19}$$

where E_A is a subset of the edges of G that spans A. If $G_A = (A, E \cap (A \times A))$ is disconnected we have to add some edges with weight 0. We point out our empirical observation that using a spanning graph that includes all edges of G_A, instead of an arbitrary spanning-tree, leads to shorter runtimes.

As before, we apply the reduction theorem to add a higher-order Potts function as part of a model at hand. This only requires a single auxiliary variable. Fig. 4(b) provides a sketch for a function of order three.

5 Cutting-Plane Approach and Separation Procedures

5.1 Approach

Determining a multicut with minimal costs is NP-hard in general [21]. However, if given data induce some structure then it is plausible to expect such problems to be easier solvable in practice, than problems without any structure.

We use a cutting-plane approach to iteratively tighten an outer relaxation of the form

$$\arg\min_{y \in Y} \sum_{e \in E} w_e \, y_e. \tag{20}$$

Here, $Y \supseteq \mathrm{MC}(G)$ is superset of the multicut polytope $\mathrm{MC}(G)$ (cf. (P3)) or $\{0,1\}^{|E|} \supset Y \supseteq \mathrm{MC}(G) \cap \{0,1\}^{|E|}$ in the integer case. In each step we solve a problem relaxation in terms of a linear or integer linear program, detect violated constraints from a pre-specified finite list (cf. Sec. 5.2) and augment the constraint system accordingly. This procedure is repeated until no more violated constraints are found.

After each iteration we obtain a lower bound as the solution of the (I)LP and an upper bound by mapping the obtained solution to the set of feasible points (rounding, cf. Sec. 5.3).

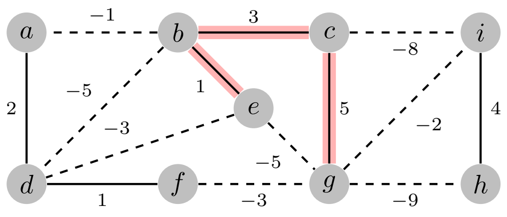


Figure 5: Edges labelings have to be constrained in order to be consistent. The active edges are shown dashed in the figure. This edge labeling is inconsistent since it does not respect the transitivity of the corresponding relation: being in the same segment. For example, the red path implies by transitivity that e and g are in the same segment, in conflict to the edge-label of the edge eg.

5.2 Relaxation, Constraints

5.2.1 Initial Constraints

We start with a polytope that enforces any edge-variable y_e to be lower and upper bounded by 0 and 1, respectively,

$$y_e \in [0,1], \qquad \forall e \in E \tag{21}$$

In presence of terminal nodes, we additionally enforce for each non-terminal node $v \in V \setminus T$ that exactly one incident edge is inactive, i.e.,

$$\sum\nolimits_{t \in T} y_{tv} = |T| - 1, \qquad \text{if } T \neq \emptyset, \; \forall v \in V \setminus T. \tag{22}$$

Furthermore, we add the compulsory constraints

$$y_{tt'} = 1, \qquad \forall t, t' \in T, t \neq t', \tag{23}$$

forcing different terminal nodes to belong to different shores.

5.2.2 Integer Constraints

A more restrictive alternative to (21) are the integer constraints

$$y_e \in \{0,1\}, \qquad \forall e \in E. \tag{24}$$

Note that not every vector $y \in \{0,1\}^{|E|}$ belongs to the multicut polytope. Hence, even enforcing Boolean variable values may lead to inconsistent edge-labelings, cf. Fig. 5. In general, using constraints (24) renders inference problems more difficult. On the other hand, finding violated constraints can be much simpler for Boolean-valued variables than for less tight non-Boolean relaxations. This may well compensate the additional costs[1] for solving an ILP instead of an LP.

[1]Note, sometimes solving the ILP is even faster than the LP.

5.2.3 Cycle Constraints

The problem of inconsistent edge-labelings has been considered in the literature, either motivated by closing contours [5, 36] or as tightening the multicut polytope relaxation via cycle constraints [15, 25, 28, 39]. In both cases inconsistent cycles are detected. If integer constraints are enforced an inconsistent cycle is a cycle that contains exactly a single active edge, which obviously violates transitivity. This can be generalized to the relaxed non-Boolean case $y_e \in [0, 1]$ [15].

A system of *cycle inequalities* that necessarily has to be satisfied by consistent labelings, is given by

$$\sum\nolimits_{e\in P} y_e \geq y_{uv} \qquad \forall uv \in E,\ P \in \text{Path}(u, v) \subseteq E. \tag{25}$$

It is well known [15] that if and only if the cycle $\{uv\} \cup P$ is chordless, then the constraint is facet-defining for the underlying polytope or, speaking less technically, “effective” for enforcing labeling consistency.

While for fully connected graphs, (25) can be represented by a polynomial number of triangle constraints [12, 15, 23], the separation procedure reduces to a sequence of shortest path problems in the general case [15]. Given y, the naive approach searches for each edge $uv \in E$ the shortest path from u to v in the weighted graph $G_y = (V, E, y)$. If this path is shorter than y_{uv}, then it represents the most violated constraint of the form (25) for uv. Using a basic implementation of Dijkstra (as we do) the cost for one search is $O(|V|^2)$. The cost can be reduced to $O(|E| + |V| \log |V|)$ by using Fibonacci heaps.

To reduce the number of shortest path searches we exploit the following three ideas:

Efficient Bounds on the Shortest Path (B): Instead of searching for each edge $uv \in E$ the shortest path from u to v in a positive weighted graph $G = (V, E, y)$, we can calculate a lower bound on the path length for all $uv \in E$ in $O(|E|+|V|)$. To this end, we determine the connected components in the graph $G' = (V, \{e \in E \mid y_e < \gamma\})$. If two nodes $u, v \in V$ are not in the same connected component, the shortest path from u to v is greater than or equal to γ. Choosing $\gamma = 1$ yields a preprocessing procedure that enables to omit many shortest path searches. Furthermore, if the edge between two nodes has weight 0, this is obviously the shortest path since all edge-weights y_e are non-negative.

Shortest Path in Binary Weighted Graph (I): If the edge weights are either 0 or 1, then simple breadth-first search can be applied instead of the Dijkstra algorithm. The computational effort can be further reduced, as before but without additional costs, by restricting the search to the graph $G_0 = (V, \{e \in E \mid y_e = 0\})$. Since any path including an edge with weight 1 cannot be shorter than the edge between the two nodes which is 0 or 1.

Finding Chordless Shortest Paths / Facet-Defining Constraints (F): A path between the two nodes forming an edge is called *chordless* if the cycle consisting of the path and the edge has no chord. Shortest path search can be easily extended so as to determine the shortest chordless paths: Every node except for the end-node is not updated by the Dijkstra algorithm if the path from this node to the starting node is chordal. This increases the costs by a factor bounded by $|V|$.

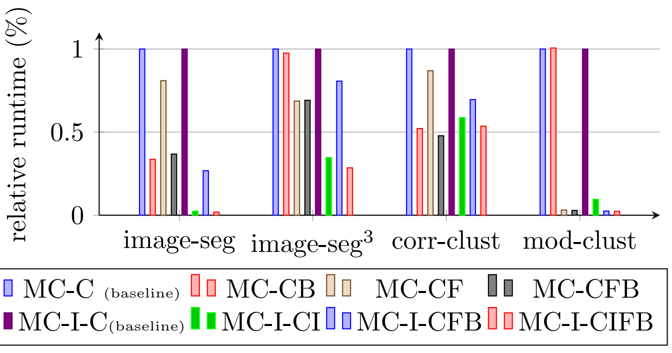


Figure 6: Comparison of the proposed extensions (marked by the postfixes B, I, and F) on the runtime. For the relaxed case (MC-C$*$) (first four bars) we observe that bounding clearly improves runtimes for image-based data, this is not true for third-order image segmentation and modularity clustering. Using only facet-defining constraints decreases the runtime for all four datasets, most significantly for modularity clustering. If we enforce integrality (MC-I-C$*$) during the cutting-plane procedure (last four bars), the use of specialized search methods (CI) reduces the runtime significantly.

In view of cycle constraints, the corresponding constraints are facet-defining.

Our experiments, discussed by Fig. 6 and in Sec. 6, spot that joint application of bounding procedures, facet-defining constraints (chordless paths) and dedicated search methods for binary weighted graphs, leads to better runtimes in nearly all cases.

5.2.4 Terminal Cycle Constraints

We can further reduce the costs for shortest path searches based on the following lemma.

Lemma 5.1. *In the presence of terminals there exists no cycle C with more than three nodes that is chordless and contains a terminal node.*

Proof. Let C be a cycle with more than three nodes that contains a terminal node t, and select an edge uv in C with $u, v \neq t$. The $tu, tv \in E$ by definition, hence the cycle is chordal. □

As a result of Lemma 5.1, we ignore all cycle constraint of a length greater than 3 that includes a terminal node. All facet-defining cycle constraints that include a terminal node are then given by

$$y_{tu} + y_{tv} \geq y_{uv}, \qquad \forall uv \in E,\ t \in T, \tag{26}$$
$$y_{tu} + y_{uv} \geq y_{tv}, \qquad \forall uv \in E,\ t \in T, \tag{27}$$
$$y_{tv} + y_{uv} \geq y_{tu}, \qquad \forall uv \in E,\ t \in T, \tag{28}$$

together with (23). As a consequence we only have to search for general cycle constraints on the graph without terminal nodes, that has $|T| \cdot |V|$ fewer edges!

5.2.5 Multi Terminal Constraints

Călinescu et al. [13] suggested another class of non-facet-defining linear inequalities that further tightens the outer polytope relaxation:

$$y_{uv} \geq \sum\nolimits_{t\in S} (y_{tu} - y_{tv}), \qquad \forall uv \in E, S \subseteq T. \tag{29}$$

Intuitively, these constraints enforce each non-terminal edge to be at least as active as all its terminal edge-pairs indicate. Since $\sum_{t\in T}(y_{tu} - y_{tv}) = 0$, we only consider differences in the direction $u \to v$. An alternative representation of (29) exploiting symmetry is

$$y_{uv} \geq \sum\nolimits_{t\in T} \frac{1}{2} |y_{tu} - y_{tv}|. \tag{30}$$

In order to see why multi terminal constraints are useful, let us consider a tiny toy example of a model with two variables and four labels. Overall, the multiway cut polytope has eight terminal edges $(t,1)_{t\in T}$, $(t,2)_{t\in T}$ and a single edge $(1,2)$ between the two nodes. We inspect few values of y and check if (29) is implied by (26)–(28) or not.

$(y_{t,1})_{t\in T}$	$(y_{t,2})_{t\in T}$		(26)–(28)	(29)
$(1,1,1,0)$	$(1,1,0,1)$	$\Rightarrow$	$1 \leq y_{12} \leq 1$	$1 \leq y_{12}$
$(1,1,\frac{1}{2},\frac{1}{2})$	$(1,1,\frac{1}{2},\frac{1}{2})$	$\Rightarrow$	$0 \leq y_{12} \leq 1$	$0 \leq y_{12}$
$(\frac{1}{2},\frac{1}{2},1,1)$	$(1,1,\frac{1}{2},\frac{1}{2})$	$\Rightarrow$	$\frac{1}{2} \leq y_{12} \leq \frac{3}{2}$	$\mathbf{1} \leq y_{12}$
$(1,\frac{2}{10},\frac{3}{10},\frac{5}{10})$	$(1,\frac{1}{10},\frac{2}{10},\frac{7}{10})$	$\Rightarrow$	$\frac{2}{10} \leq y_{12} \leq \frac{3}{10}$	$\frac{2}{10} \leq y_{12}$

In the third example (row) above, multi terminal constraints tighten the relaxation. It can be shown that these constraints may tighten the relaxation only if at least four terminal nodes are present.

5.2.6 Odd-Wheel Constraints

While cycle constraints are only sufficient to obtain optimal solutions if integer constraints are enforced, we may tighten the relaxation in the case $y_e \in [0,1]$ by adding more complex constraints.

One such a class of constraints for which the separation procedure can be carried out efficiently, are *odd-wheel constraints*. A *wheel* $W = (V_W, E_W)$ is a graph with a selected center node $c \in V_W$. All other nodes are connected with the center, and the remaining edges build a cycle containing all nodes in $V_W \setminus \{c\}$. An *odd-wheel* is a wheel with an odd number of non-center nodes. The *odd-wheel constraints* are given by

$$\sum_{uv\in E_W, u,v\neq c} w_{uv} - \sum_{v\in V_W\setminus\{c\}} w_{cv} \leq \left\lfloor \frac{||V_W| - 1|}{2} \right\rfloor \tag{31}$$

for all odd-wheels $W = (V_W, E_W)$.

Deza et al. [19] proved that odd-wheel constraints are facet-defining for $||V_W| - 1| \geq 3$. As described in detail by Deza and Laurent [20] and Nowozin [38], the search for violated odd-wheel constraints can be reduced to a polynomial number of shortest path searches, if the current solution does not violate any cycle constraints.

In our experiments, we found that with increasing sparsity, odd-wheel constraints tighten the relaxation less. This is intuitively plausible since in densely connected graphs significantly more odd-wheels exist that could be violated. Since the overall gain was not better than with the previously proposed methods, we did not spend time to search for heuristics to speed up computation, as we did for the cycle inequalities.

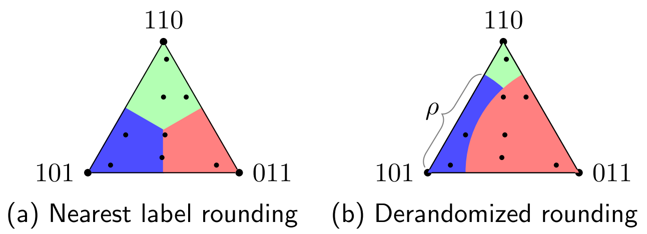


Figure 7: Illustration of the two rounding schemes for the multiway cut problem for the vector $(y_{tv})_{t\in T}$. *Nearest label rounding* (a) assigns each point in the simplex to the nearest vertex. Fig. (b) shows exemplarily one iteration of *derandomized rounding* for $\rho = 0.75$.

5.3 Rounding Fractional Solutions

Relaxations of the integer-valued multicut problem yield solutions that may be fractional and therefore infeasible. The objective value then will be a lower bound of the optimal value. The procedure to map an infeasible solution to the feasible set is called *rounding*. Furthermore, for the resulting multicut, a corresponding node-labeling has to be determined.

5.3.1 Supervised Case

In the presence of terminal nodes, we assign to each node-variable the label of the terminal node to which it is connected by means of $y_{tv} = 0$ in the integer-valued case. This idea extends to the general case by assigning to node v the label l with the lowest edge-value $y_{t_l v}$, i.e., the nearest corner in the corresponding simplex, cf. Fig. 7:

$$x_v = \arg\min\nolimits_{t\in T} y_{tv} \qquad \forall v \in V \setminus T. \tag{32}$$

This heuristic *nearest label rounding* method has two drawbacks, however. Firstly, it does not provide any performance guarantee. Secondly, nearby nodes that favor two or more labels nearly equally might be randomly assigned to different labels due to numerical inaccuracy. This is particularly problematic in case of positive coupling strengths where homogeneously labeled regions are preferred.

Contrary to this local procedure, Călinescu et al. [13] suggested a randomized rounding procedure that provides optimality bounds for Potts models with positive coupling strengths. Given a threshold $\rho \in [0,1]$, they iterate over all labels in a fixed order and assign label l to node v if $y_{t_l v} \leq \rho$ and no label was assigned to v before. In case no label was assigned to node v in the end, then the last label with respect to the ordering of the labels is assigned to v. This rounding procedure is sketched by Fig. 7(b).

A *randomized rounding* procedure would apply this for all $\rho \in [0,1]$ and select the labeling with the lowest energy. Since $[0,1]$ is uncountable, Călinescu et al. suggested a derandomized version. This is based on the observation that we only have to consider $|V \setminus T| \cdot |T|$ different threshold parameters, namely the

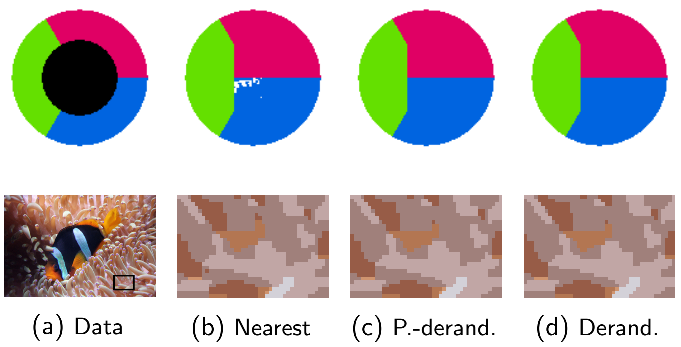


Figure 8: Illustration of the rounding results (nearest label, pseudo-derandomized and derandomized) after solving the LP relaxation with terminal, multi-terminal, and cycle inequalities for the instances inpainting and clownfish from [24]. Derandomized and pseudo-derandomized rounding gives similar results. Simple rounding to the nearest label can give inferior results (top row). But for real applications differences of the labelings are marginal (last row).

values of the terminal edge variables y_{tv}. Since this set can still be quite large, we also consider a heuristic approximation that we call *pseudo-derandomized rounding*, using a small number of equidistant thresholds, in practice: $0, 0.01, 0.02, \dots, 0.99, 1$.

Concerning tightness of the relaxation, Călinescu et al. [13] pointed out that the integrality ratio of the relaxed LP for the second-order multiway cut problem with positive coupling strengths, exploiting cycle, terminal and multi-terminal constraints, is $\frac{3}{2} - \frac{1}{k}$. This is superior to the α-expansion algorithm [11] and the work of Dahlhaus et al. [16], which guarantees only a ratio of $2 - \frac{2}{k}$.

Empirically, we observe for these types of models that derandomized rounding and pseudo-derandomized rounding usually lead to results that are slightly better than when using nearest label rounding. While pseudo-derandomization does empirically not give results worse than original derandomization, it is much faster, but does not come along with theoretical guarantees. Fig. 8 shows results for two instances taken from [24]. While for the synthetic instances rounding matters, for real world examples the differences are negligible.

5.3.2 Unsupervised Case

In absence of terminal nodes, we compute in the integer-valued case the connected components of $G_0 = (V, \{e \in E \mid y_e = 0\})$, enumerate them by $\#CC_{G_0}$, and assign to each node-variable as label the number of its connected component

$$x_v = \#CC_{G_0}(v), \qquad \forall v \in V. \tag{33}$$

It is easy to see that the labeling-costs $J(x)$ (2) are greater than or equal to the multicut costs $\langle w, y\rangle$ and equal if y is a valid multicut.

If y is not integral we first have to map y to a vertex of the multicut polytope. To this end, we determine the connected components of $G_{\leq\kappa} = (V, \{e \in E \mid y_e \leq \kappa\})$ and define the

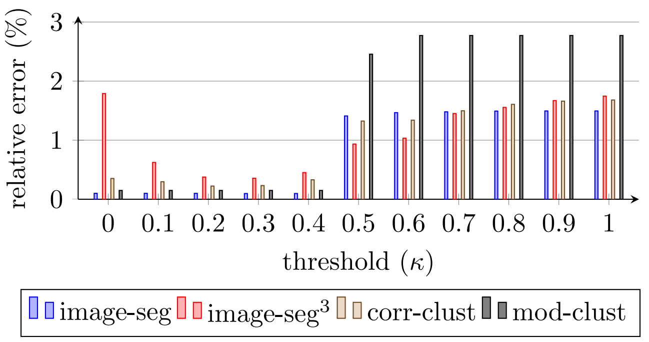


Figure 9: Illustration of the impact of the choice of κ on the distance of the energy of the integer solution obtained by rounding to the optimal value. For modularity clustering (mod-clust) and third-order image segmentation (image-seg^3) we scaled the bars by a factor of 0.1. The results show that one should choose $\kappa < 0.5$. Empirically the optimal value lies in $[0.2, 0.3]$ but also 0 (more precisely 10^{-8}) gives nearly similar results.

feasible projection $\hat{y}$ by

$$\hat{y}_{uv} = \begin{cases} 0, & \text{if } \#CC_{G_{\leq\kappa}}(u) = \#CC_{G_{\leq\kappa}}(v), \\ 1, & \text{else.} \end{cases} \tag{34}$$

The labeling then is given by

$$x_v = \#CC_{G_{\leq\kappa}}(v), \qquad \forall v \in V. \tag{35}$$

Since the connected component procedure will tend to remove dangling edges, it seems to be reasonable to select κ smaller than 0.5. This was empirically confirmed by our experiments. Fig. 9 shows the relative error of the rounded solutions after enforcing cycle constraints for different problem-classes with various values of κ.

5.4 Multicut Cutting-Plane Algorithm

Algorithm 1 provides a compact description of our complete multicut approach, summarizing the present section. In addition to the specification of the objective function in terms of a factor graph model $\mathcal{G}$, we expect a proper[2] list of separation procedure sets $\mathbb{S}$ as input parameters. For example, $\mathbb{S}_1$ could represent simple cycle constraints separation, $\mathbb{S}_2$ integrality constraints, and $\mathbb{S}_3$ cycle constraints separation specialized to integer solutions.

As specified by algorithm 1, we construct the weighted undirected graph G, introduce auxiliary variables for higher-order factors (as detailed in previous sections), and initialize the constraint set $\mathcal{C}$ by a simple outer relaxation of the feasible set.

For each separation procedure set in the list $\mathbb{S}$, we apply all separation procedures in $\mathbb{S}_i$ to find violated constraints and add these to $\mathcal{C}$ until no more are found. Then we proceed with the next set $\mathbb{S}_{i+1}$.

[2]A list of separation procedures is called proper if the separation procedures that are included once are also included when proceeding further down the list. For proper lists the obtained relaxation is well-defined. All lists used in our experiments are proper.

Algorithm 1 Multicut-Algorithm

```
1:  Given: $\mathcal{G}$ = factor graph model,
           $\mathbb{S}$ = proper list of separation procedure sets.
2:  Construct $G = (V, E, w)$ from $\mathcal{G}$.
3:  Initialize the constraint set $\mathcal{C}$ as described in Sec. 5.2.1.
4:  for $i = 1, \dots, |\mathbb{S}|$ do
5:    repeat
6:      Solve $\hat{y} \in \arg\min_{y \in \mathcal{C}} \langle w, y \rangle$,
7:      $\bar{\mathcal{C}}$ = violated constraints found by separation procedures $\mathbb{S}_i$ for $\hat{y}$,
8:      $\mathcal{C} = \mathcal{C} \cup \bar{\mathcal{C}}$,
9:    until $\bar{\mathcal{C}} == \emptyset$.
10: end for
11: Compute a labeling $x \in X$ based on $\hat{y}$.
```

The (integer) linear program in line 6 is solved by CPLEX 12.2, a standard off-the-shelf LP-solver. Finally, we compute an optimal node-labeling $x \in X$ from the multicut solution y.

The implementation of Alg. 1 turned out to be involved, due to several pitfalls necessitating some care. We will therefore make our code publicly available. Furthermore, when solving the (I)LP one should not expect that the solution is feasible. Sometimes we observe negative values of y_e and therefore project solutions always to $[0, 1]^{|E|}$. Also Boolean constraints were sometimes slightly violated. Most importantly, due to numerical reasons, constraints should only be added if they are significantly violated, i.e., the constraint $a \leq b$ is only added if $a \leq b - \epsilon$ does not hold. Ignoring this may not only lead to infinite loops for some instances, but may also significantly increase runtime. The parameter ϵ should be chosen depending on the precisions of the (I)LP solver. We use $\epsilon = 10^{-8}$.

6 Experiments

6.1 Set-Up, Implementation Details

We implemented the separation procedures and reduction methods described above using C++ and the OpenGM2-library [4] for the factor graph representation, and CPLEX for solving ILPs and LPs in the inner loop of the iteration.

Our multicut approach encompasses a variety of algorithms which differ in the used inequalities, in the separation procedures, and in the order these procedures are applied. The abbreviations for single separation procedures are listed as Tab. 1.

For example, MC-CFB-I-CIF indicates:

- application of the multicut algorithm (MC) based on
- searching for violated facet-defining cycle inequalities (CF) using bounding (B),
- enforcing integer constraints (I), and finally
- searching for facet-defining cycle inequalities violated by the current *Boolean* solution (CIF), based on Breadth-First-Search instead of the Dijkstra algorithm (cf. Sec. 5.2.3).

Table 1: Abbreviations for the separation procedures.

I	integer constraints
C	cycle inequalities separation
CF	facet-defining cycle inequalities separation
CI	cycle inequalities separation for ILP
CIF	facet-defining cycle inequalities separation for ILP
OW	odd-wheel inequalities separation
T	terminal inequalities separation
MT	multi terminal inequalities separation
TI	terminal inequalities separation for ILP
***B**	bounding for the shortest path search was used

We report for each dataset results averaged over all its instances:

1. the mean runtime: *runtime*,
2. the mean value of the integer solution after rounding: *value*,
3. the mean lower bound: *bound*,
4. how often the method found an integer solution with an objective value not larger than 10^{-6} compared to the overall best method for this instance: *best*, and
5. how often the method provided a gap between the objective value of the integer solution and the lower bound, that was smaller than 10^{-6}: *ver. opt*, which we interpret as globally optimal for our instances.

In the *unsupervised case*, we compared the proposed methods with our implementation of the Kernighan-Lin (KL) algorithm [27] for the second-order case, as well as with iterative conditional mode (ICM) [8] and Lazy Flipper (LF) [6]. For *planar graphs*, an optimal segmentation with only four labels exists, and methods for the supervised case can be applied.

In the *supervised case*, we compared with TRWS [32], α-expansion [11] and FastPD [33] – using in each case code provided by the respective authors of these papers. Furthermore, we compared to commercial LP- and ILP-solvers in the nodal domain, LBP, TRBP, and α-fusion, as provided by OpenGM2.

6.2 Probabilistic Image Segmentation

The probabilistic image segmentation framework was suggested by Andres et al. [5] and belongs to the class of unsupervised image segmentation problems. These problem instances involve $156 \cdots 3764$ superpixels. For all pairs of adjacent superpixels, the likelihood that their common part of the superpixel boundary is part of the segmentation, is learned offline by a random forest. This results in a Potts model with positive and negative coupling constraints. While the connection to Potts models is not mentioned in [5], they use a similar optimization scheme as in the present work. They introduced a higher-order model as well as a second-order one. Only the latter has been made publicly available in [24].

Second-order Case. As shown in Tab. 2, for this dataset, we profit from using ILP subproblems. This reduces the mean runtime to less than 3 seconds and is therefore empirically

Table 2: Second-order probabilistic image segmentation [5,24]

algorithm	runtime	value	bound	best	ver. opt
KL	4.96 s	4608.57	$-\infty$	0.0%	0.0%
ICM	6.03 s	4705.07	$-\infty$	0.0%	0.0%
LF1	2.35 s	4705.01	$-\infty$	0.0%	0.0%
LF2-L4	0.13 s	4627.38	$-\infty$	0.0%	0.0%
LF3-L4	3.16 s	4581.83	$-\infty$	0.0%	0.0%
LF4-L4	176.47 s	4555.73	$-\infty$	0.0%	0.0%
TRWS-L4	0.84 s	4889.23	4096.53	0.0%	0.0%
MC-C	14.02 s	4447.47	4442.34	35.0%	35.0%
MC-CB	**4.71 s**	4447.47	4442.34	35.0%	35.0%
MC-CF	11.35 s	4447.47	4442.34	35.0%	35.0%
MC-CFB	5.16 s	4447.47	4442.34	35.0%	35.0%
MC-C-OW	14.08 s	4447.41	4442.34	35.0%	35.0%
MC-CB-OW	**4.81 s**	4447.41	4442.34	35.0%	35.0%
MC-CF-OW	11.45 s	4447.41	4442.34	35.0%	35.0%
MC-CFB-OW	5.19 s	4447.41	4442.34	35.0%	35.0%
MC-I-CI	2.78 s	4442.64	4442.64	100.0%	100.0%
MC-I-CIF	**2.20 s**	4442.64	4442.64	100.0%	100.0%
MC-C-I-CI	15.00 s	4442.64	4442.64	100.0%	100.0%
MC-CFB-I-CIF	5.69 s	4442.64	4442.64	100.0%	100.0%

faster than LP-based cutting-plane methods and the heuristic KL-algorithm. ICM and LF perform worse than KL. With increasing search space LF outperforms KL. For a search-depth greater than 1 we make use of the fact that the instances are planar and an optimal solution with four labels exists. The same trick is used to make TRWS applicable. Additionally, we fix the first variable and initialize messages randomly. Even this does not help to prevent TRWS from running into poor local fix-points. In both cases the label reduction is marked by the postfix *L4*.

Concerning the multicut approach, odd-wheel constraints only marginally improve the results. LP-based cutting-plane methods find the optimal solution for 35 of 100 instances and are slower than ILP-based methods, too.

Higher-order Case. The third-order models from [5] are hard to solve with relaxations, hence rounding becomes more important, cf. Fig. 9. The additional third-order factors favor smooth boundary continuation. Since this sometimes conflicts with local boundary probabilities, the problem becomes more involved.

As shown in Tab. 3, local search methods give better results than relaxed solutions after rounding. Our exact multicut scheme was able to solve all instances to optimality. Notably, one instance was significantly harder than all others and took more than half of the overall runtime for MC-I-C and MC-I-CFB.

Overall, a few instances are significantly harder than others. This is apparent by the large difference of the mean runtime to the median runtime (the latter is shown in parentheses in Tab. 3).

6.3 Higher-order Hierarchical Image Segmentation

The hierarchical image segmentation framework was suggested by Kim et al. [28] and also belongs to the class of unsupervised image segmentation problems. Contrary to the work of Andres et al. [5], they learn their model-parameters by a structured support vector machine (S-SVM). Furthermore, higher-order Potts terms force selected regions to belong to the same cluster. The 715 instances of this dataset, published as part of [24], contain factors of order up to a few hundred and 122–651 variables.

The results are summarized as Table 4. Surprisingly, our LP-based methods perform better than the original algorithm used in [28], even though the algorithms are identical. Maybe this was caused by the different LP solver they used, or by some floating-point problems inside their separation procedure. The use of odd-wheel constraints marginally improves the results. Best results are obtained by using integer cutting-planes after having solved the LP. The use of the bounding as part of the post-processing reduces runtime by a factor of 2. The differences to only using facet-defining constraints are negligible.

Table 3: Third-order probabilistic image segmentation [5]

algorithm	runtime	value	bound	best	ver. opt
ICM	10.79 *(8.11)* s	6030.49	$-\infty$	0.0%	0.0%
LF	4.17 *(3.11)* s	6030.29	$-\infty$	0.0%	0.0%
MC-C	43.82 *(9.33)* s	6657.32	5465.15	0.0%	0.0%
MC-CB	42.86 *(9.26)* s	6657.32	5465.15	0.0%	0.0%
MC-CF	26.68 *(8.06)* s	6658.28	5465.15	0.0%	0.0%
MC-CFB	25.00 *(6.64)* s	6658.28	5465.15	0.0%	0.0%
MC-C-OW	43.71 *(11.16)* s	6657.12	5465.29	0.0%	0.0%
MC-CB-OW	43.38 *(9.66)* s	6657.12	5465.29	0.0%	0.0%
MC-CF-OW	27.62 *(8.15)* s	6658.08	5465.29	0.0%	0.0%
MC-CFB-OW	25.55 *(7.40)* s	6658.08	5465.29	0.0%	0.0%
MC-I-C	689.79 *(41.43)* s	5627.52	5627.52	100.0%	100.0%
MC-I-CFB	469.87 *(33.02)* s	5627.52	5627.52	100.0%	100.0%
MC-I-CI	119.64 *(31.73)* s	5627.52	5627.52	100.0%	100.0%
MC-I-CIF	72.81 *(27.39)* s	5627.52	5627.52	100.0%	100.0%
MC-C-I-CI	125.33 *(33.63)* s	5627.52	5627.52	100.0%	100.0%
MC-CFB-I-CIF	82.00 *(25.60)* s	5627.52	5627.52	100.0%	100.0%

Table 4: Higher-order hierarchical image segmentation [28].

algorithm	runtime	value	bound	best	ver. opt
ICM	1.90 s	−585.60	$-\infty$	0.0%	0.0%
LF	1.00 s	−585.60	$-\infty$	0.0%	0.0%
MC-C	0.23 s	−625.97	−628.89	19.9%	13.7%
MC-CB	0.12 s	−625.97	−628.89	19.9%	13.7%
MC-CF	0.20 s	−625.97	−628.89	19.9%	13.7%
MC-CFB	**0.11 s**	−625.97	−628.89	19.9%	13.7%
MC-C-OW	0.24 s	−625.98	−628.89	20.1%	14.0%
MC-CB-OW	0.14 s	−625.98	−628.89	20.1%	14.0%
MC-CF-OW	0.21 s	−625.98	−628.89	20.1%	14.0%
MC-CFB-OW	**0.13 s**	−625.98	−628.89	20.1%	14.0%
MCR [28]	0.38 s	−624.35	−629.03	16.4%	10.2%
MC-CI	1.14 s	−628.16	−628.16	100.0%	100.0%
MC-CIF	1.04 s	−628.16	−628.16	100.0%	100.0%
MC-C-CI	0.85 s	−628.16	−628.16	100.0%	100.0%
MC-CFB-CIF	**0.62 s**	−628.16	−628.16	100.0%	100.0%

6.4 Modularity Clustering

We also considered a clustering problem from outside the field of computer vision, which contrary to the previous models considered so far, involves a fully connected graph. Modularity clustering [12] means the problem of clustering an undirected unweighted graph into "meaningful" subsets, which amounts to optimization problems related to fully connected Potts model. For our experiments, we used the datasets[3] *dol-*

[3]`http://www-personal.umich.edu/~mejn/netdata/`

Table 5: Modularity clustering [12]

algorithm	runtime	value	bound	best	ver. opt
KL	0.01 s	−0.5251	$-\infty$	2/4	0/4
ICM	0.12 s	0.0000	$-\infty$	0/4	0/4
LF	0.05 s	0.0000	$-\infty$	0/4	0/4
MC-C	47.99 s	−0.5204	−0.5294	1/4	1/4
MC-CB	48.33 s	−0.5204	−0.5294	1/4	1/4
MC-CF	1.02 s	−0.5204	−0.5294	1/4	1/4
MC-CFB	0.91 s	−0.5204	−0.5294	1/4	1/4
MC-C-OW	72.05 s	−0.5282	−0.5282	4/4	4/4
MC-CB-OW	72.42 s	−0.5282	−0.5282	4/4	4/4
MC-CF-OW	12.26 s	−0.5282	−0.5282	4/4	4/4
MC-CFB-OW	11.60 s	−0.5282	−0.5282	4/4	4/4
MC-I-C	152.20 s	−0.5282	−0.5282	4/4	4/4
MC-I-CI	14.57 s	−0.5282	−0.5282	4/4	4/4
MC-I-CIF	6.31 s	−0.5282	−0.5282	4/4	4/4
MC-I-CCFDB	6.56 s	−0.5282	−0.5282	4/4	4/4
MC-C-I-CI	58.24 s	−0.5282	−0.5282	4/4	4/4
MC-CFB-I-CIF	1.31 s	−0.5282	−0.5282	4/4	4/4

phins, *football*, *karate*, and *lesmis*, with 62, 115, 34, and 77 data-points, respectively.

As shown in Tab. 5, for modularity clustering, the use of facet-defining inequalities as well as odd-wheel constraints significantly improves the results. We attribute this to the high connectivity of the graph. In such dense graphs more likely violated odd-wheel inequalities exist. Likewise, more *non*-facet-defining cycle inequalities exist as well, and adding those only blows up the system of inequalities.

As observed by Nowozin and Jegelka [39], odd-wheel inequalities usually tighten sufficiently the polytope. Furthermore, we observed for this dataset, as in [39], numerical problems if the allowed feasibility and optimality tolerances were set too large. However, the experiments showed that our proposed integer cycle inequalities perform better than odd-wheel separation, especially if we start from the LP-relaxation with cycle inequalities, cf. Tab. 5.

6.5 Supervised Image Segmentation

An elementary approach to supervised image segmentation, or image labeling, is to apply locally a statistical classifier, trained offline beforehand, to raw image data or to locally extracted image features. This is complemented by a non-local prior term, the most common form of which favours short boundaries of the segments partitioning the image domain. Such terms can be approximated by pairwise Potts terms [9] and lead to an energy function of the form

$$\sum_{f\in\mathcal{F}_1} -\log(p_{ne(f)}(x_{ne(f)}|I)) + \sum_{f\in\mathcal{F}_2} \beta\,\mathbb{I}(x_{ne(f)_1} \neq x_{ne(f)_2}).$$

As recently shown by Kappes et al. [26], such models can be evaluated globally optimal and very fast by first determining partial optimality, leading to a reduced inference problem in terms of remaining unlabelled connected image components, followed by solving each of these smaller problems independently.

We use the labels *"*"* to mark when these preprocessing steps were applied and "†" to mark whenever the memory requirement exceeded 12 GB.

Table 6: Supervised image segmentation [2]

algorithm	runtime	value	bound	best	ver. opt
MC-T-MT-I-T	149.43 s	308 472 274.3	308 472 274.3	3/3	3/3
MC*-T-MT-I-T	1.86 s	308 472 274.3	308 472 274.3	3/3	3/3
ILP	†	†	†	†	†
ILP*	1.91 s	308 472 274.3	308 472 274.3	3/3	3/3
MC-T-MT	115.14 s	308 472 274.3	308 472 274.3	3/3	3/3
MC*-T-MT	1.76 s	308 472 274.3	308 472 274.3	3/3	3/3
LP	†	†	†	†	†
LP*	2.17 s	308 472 274.3	308 472 274.3	3/3	3/3
TRWS [32]	150.47 s	308 472 310.6	308 472 270.4	2/3	1/3
TRWS*	3.90 s	308 472 274.3	308 472 274.3	2/3	2/3
FastPD [33]	0.45 s	308 472 275.0	$-\infty$	2/3	0/3
FastPD*	1.62 s	308 472 274.7	$-\infty$	2/3	0/3
α-Exp [11]	6.42 s	308 472 275.6	$-\infty$	2/3	0/3
α-Exp*	1.72 s	308 472 274.3	$-\infty$	3/3	0/3

As dataset we used the color segmentation instances of Alahari et al. [2]. The results are summarized as Table 6.

While standard (I)LP solvers often suffer from their large memory requirements, the multicut approach outperformed all other approaches. Since for all instances the local polytope relaxation returned optimal integer solutions, MC-T-MT could solve them in polynomial time. When we resorted to the model reduction *, the subproblems became small for these problem instances, and (I)LP solvers could be conveniently applied. Our multicut approach then was only marginally faster. Despite global optimality, however, the runtime was comparable to algorithms for approximate inference that do not guarantee global optimality.

6.6 Higher-Order Supervised Image Segmentation

We studied image segmentation with junction regularisation as problem instances that benefit from the application of higher-order generalized Potts functions.

Rather than merely penalizing the boundary length of segments, this approach aims at improving segmentation results by additionally penalizing points where the boundaries of three or more segments meet:

$$\varphi^I(x_1,x_2,x_3,x_4) = \begin{cases} \lambda, & \text{if } |\{x_1,x_2,x_3,x_4\}| > 2, \\ 0, & \text{else.} \end{cases} \tag{36}$$

The overall cost for labeling then is given by

$$\sum_{f\in\mathcal{F}_1} \varphi^1_f(x_{ne(f)}) + \sum_{f\in\mathcal{F}_2} \varphi^2_f(x_{ne(f)}) + \sum_{f\in\mathcal{F}_4} \varphi^I(x_{ne(f)}),$$

where φ^1 denotes the L_1-norm of the difference between intensity of a pixel and a pixel-label, φ^2 the same second-order terms as in the pairwise case, and $\mathcal{F}_4$ the set of all factors over four pixels that build a cycle in the image grid.

Setting λ to 0 yields standard second-order model with boundary length regularization, whereas setting $\lambda \to \infty$ yields a model that enforces segments to be surrounded by one single segment.

Fig. 1(b) illustrates this property of the model. The standard second-order approach, cf. Fig. 1(b), top, produces many small artefacts, e.g., in "a", "x", and "v" , and often opens

Table 7: Supervised image segmentation with inclusion priors

algorithm	runtime	value	bound	best	ver. opt
ICM	0.03 s	1556.20	$-\infty$	0/10	0/10
LBP-LF2	12.20 s	1400.62	$-\infty$	8/10	0/10
α-FUSION	0.07 s	1587.13	$-\infty$	0/10	0/10
LBP	12.28 s	1800.67	$-\infty$	3/10	0/10
TRBP	13.93 s	2000.67	$-\infty$	2/10	0/10
LP	25.04 s	3900.59	1400.33	1/10	1/10
MC-T-MT	18.55 s	1739.29	1399.49	1/10	0/10
ILP	7.33 s	1400.57	1400.57	10/10	10/10
MC-T-MT-I-T	66.58 s	1400.57	1400.57	10/10	10/10

the surrounding segment (e.g., left of "a" and right of "i"). Invoking the fourth-order regularizer, cf. Fig. 1(b), bottom, eliminates many of these artefacts and results in a significantly better segmentation.

The results of an empirical evaluation for 10 synthetic 32×32 images are summarized as Table 7.

Approximate inference methods performed quite good, but among those only LBP-LF2 (Lazzy Flipper initialed with the solution of LBP) was able to provide nearly optimal results. While the multicut approach is on par when relaxations were considered, it became quite slow compared to a ILP applied to labeling in the nodal domain, when a globally optimal solution was enforced.

We believe there are two major reasons: First, the relaxation "prefers" less integral solutions due to the higher-order terms and therefore becomes harder to solve for LP-based methods. Second, we observe that CPLEX solves the ILP mainly by branching and probing in order to avoid solving LPs. This is also the reason why ILP is faster than LP.

While an in-depth study of such aspects is beyond the scope of the present paper, our findings indicate ways to further improve the multicut approach in such advanced settings.

7 Conclusion

We presented an approach based on multicuts to solve a broad range of supervised and unsupervised segmentation problems to optimality in reasonable runtime. We showed, in particular, how to extend the approach higher-order models based on a class of label invariant functions that generalize Potts functions in a natural way. Such models enable to model higher-order interactions concisely by taking its symmetries into account.

We devised several dedicated separation procedures and demonstrated a corresponding significant impact on runtime. A systematic comparison of different cutting-plane procedures for computer vision applications enabled us to improve runtimes for all models compared to the state of the art. A discussion of polynomially solvable relaxations of the unsupervised segmentation problems complemented our study, together with advanced rounding schemes.

Acknowledgements. This work has been supported by the German Research Foundation (DFG) within the program "Spatio-/Temporal Graphical Models and Applications in Image Analysis", grant GRK 1653.

References

[1] Martin Aigner. *Combinatorial Theory.* Springer, 1979.

[2] Karteek Alahari, Pushmeet Kohli, and Philip H. S. Torr. Dynamic hybrid algorithms for MAP inference in discrete MRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(10):1846–1857, 2010.

[3] Amir Alush and Jacob Goldberger. Ensemble segmentation using efficient integer linear programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10):1966–1977, 2012.

[4] Björn Andres, Thorsten Beier, and Jörg H. Kappes. OpenGM2, 2012. `http://hci.iwr.uni-heidelberg.de/opengm2/`.

[5] Björn Andres, Jörg H. Kappes, Thorsten Beier, Ullrich Köthe, and Fred A. Hamprecht. Probabilistic image segmentation with closedness constraints. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

[6] Björn Andres, Jörg H. Kappes, Thorsten Beier, Ullrich Köthe, and Fred A. Hamprecht. The lazy flipper: Efficient depth-limited exhaustive search in discrete graphical models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

[7] Björn Andres, Thorben Kröger, Kevin L. Briggman, Winfried Denk, Natalya Korogod, Graham Knott, Ullrich Köthe, and Fred A. Hamprecht. Globally optimal closed-surface segmentation for connectomics. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

[8] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, 48(3):259–302, 1986.

[9] Yuri Boykov and Vladimir Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2003.

[10] Yuri Boykov, Olga Veksler, and Ramin Zabih. Markov random fields with efficient approximations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998.

[11] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[12] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, 2008.

[13] Gruia Călinescu, Howard Karloff, and Yuval Rabani. An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences*, 60(3):564–574, 2000.

[14] Sunil Chopra and M. R. Rao. On the multiway cut polyhedron. *Networks*, 21(1):51–89, 1991.

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

[16] Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiway cuts (extended abstract). In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 1992.

[17] Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

[18] Michel M. Deza, Martin Grötschel, and Monique Laurent. Complete descriptions of small multicut polytopes. In Peter Gritzmann and Bernd Sturmfels, editors, *Applied Geometry and Discrete Mathematics: The Victor Klee Festschrift*. American Mathematical Society, 1991.

[19] Michel M. Deza, Martin Grötschel, and Monique Laurent. Clique-web facets for multicut polytopes. *Mathematics of Operations Research*, 17(4):981–1000, 1992.

[20] Michel M. Deza and Monique Laurent. *Geometry of Cuts and Metrics*. Springer, 1997.

[21] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[22] Fred Glover and Eugene Woolsey. Converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations Research*, 22(1):180–182, 1974.

[23] Martin Grötschel and Yoshiko Wakabayashi. A cutting plane algorithm for a clustering problem. *Mathematical Programming*, 45(1):59–96, 1989.

[24] Jörg H. Kappes, Björn Andres, Fred A. Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X. Kausler, Jan Lellmann, Nikos Komodakis, and Carsten Rother. A comparative study of modern inference techniques for discrete energy minimization problems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[25] Jörg H. Kappes, Markus Speth, Björn Andres, Gerhard Reinelt, and Christoph Schnörr. Globally optimal image partitioning by multicuts. In *Proceedings of the International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, 2011.

[26] Jörg H. Kappes, Markus Speth, Gerhard Reinelt, and Christoph Schnörr. Towards efficient and exact MAP-inference for large scale discrete computer vision problems via combinatorial optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[27] Brian W. Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. *The Bell Systems Technical Journal*, 49(2):291–307, 1970.

[28] Sungwoong Kim, Sebastian Nowozin, Pushmeet Kohli, and Chang D. Yoo. Higher-order correlation clustering for image segmentation. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2011.

[29] Sungwoong Kim, Sebastian Nowozin, Pushmeet Kohli, and Chang D. Yoo. Task-specific image partitioning. *IEEE Transactions on Image Processing*, 22(2):488–500, 2013.

[30] Jon Kleinberg and Éva Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1999.

[31] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

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

[33] Nikos Komodakis and Georgios Tziritas. Approximate labeling via graph cuts based on linear programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1436–1453, 2007.

[34] Victor Lempitsky, Andrea Vedaldi, and Andrew Zisserman. A pylon model for semantic segmentation. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2011.

[35] Dániel Marx. A tight lower bound for planar multiway cut with fixed number of terminals. In *International Colloquium on Automata, Languages and Programming (ICALP)*, 2012.

[36] Yansheng Ming, Hongdong Li, and Xuming He. Connected contours: A new contour completion model that respects the closure effect. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[37] Claudia Nieuwenhuis, Eno Töppe, and Daniel Cremers. A survey and comparison of discrete and continuous multilabel segmentation approaches. *International Journal of Computer Vision*. to appear.

[38] Sebastian Nowozin. *Learning with Structured Data: Applications to Computer Vision*. PhD thesis, Technische Universität Berlin, 2009.

[39] Sebastian Nowozin and Stefanie Jegelka. Solution stability in linear programming relaxations: Graph partitioning and unsupervised learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[40] Anton Osokin, Dmitry Vetrov, and Vladimir Kolmogorov. Submodular decomposition framework for inference in associative Markov networks with global constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[41] David Sontag and Tommi Jaakkola. New outer bounds on the marginal polytope. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.

[42] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

[43] Julian Yarkony, Alexander Ihler, and Charless C. Fowlkes. Fast planar correlation clustering for image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.