# Joint Graph Decomposition & Node Labeling: Problem, Algorithms, Applications

Evgeny Levinkov[1], Jonas Uhrig[3,4], Siyu Tang[1,2], Mohamed Omran[1], Eldar Insafutdinov[1], Alexander Kirillov[5], Carsten Rother[5], Thomas Brox[4], Bernt Schiele[1] and Bjoern Andres[1]

[1]Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany
[2]Max Planck Institute for Intelligent Systems, Tübingen, Germany [3]Daimler AG R&D, Sindelfingen, Germany
[4]Computer Vision Lab, University of Freiburg, Germany [5]Computer Vision Lab, Technische Universität Dresden, Germany

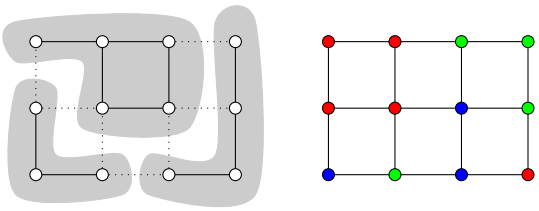
## Abstract

*We state a combinatorial optimization problem whose feasible solutions define both a decomposition and a node labeling of a given graph. This problem offers a common mathematical abstraction of seemingly unrelated computer vision tasks, including instance-separating semantic segmentation, articulated human body pose estimation and multiple object tracking. Conceptually, the problem we state generalizes the unconstrained integer quadratic program and the minimum cost lifted multicut problem, both of which are* NP*-hard. In order to find feasible solutions efficiently, we define two local search algorithms that converge monotonously to a local optimum, offering a feasible solution at any time. To demonstrate their effectiveness in tackling computer vision tasks, we apply these algorithms to instances of the problem that we construct from published data, using published algorithms. We report state-of-the-art application-specific accuracy for the three above-mentioned applications.*

## 1. Introduction and Related Work

In this article, we state a combinatorial optimization problem whose feasible solutions define both a decomposition and a node labeling of a given graph (Fig. 1). This problem that we call the minimum cost node labeling lifted multicut problem, NL-LMP, generalizes the NP-hard unconstrained integer quadratic program, UIQP, that has been studied intensively in the context of graphical models [14], and also generalizes the NP-hard minimum cost lifted multicut problem, LMP [16]. Unlike solutions of pure node labeling problems such as the UIQP, solutions of the NL-LMP can assign neighboring nodes with the same label to distinct components, and neighboring nodes with distinct labels to the same component. Unlike in pure decomposition problems such as the LMP, the cost of assigning nodes to the same component or distinct components can depend on node labels.

In order to find feasible solutions of the NL-LMP efficiently, we define and implement two local search algorithms that converge monotonously to a local optimum, offering a feasible solution at any time. These algorithms do not compute lower bounds. They output feasible solutions without approximation certificates. Hence, they belong to the class of primal feasible heuristics for the NL-LMP. The first algorithm we define and refer to as alternating Kernighan-Lin search with joins and node relabeling, KLj/r, is a generalization of the algorithm KLj of Keuper et al. [16] and of Iterated Conditional Modes (ICM). The second algorithm we define and refer to as joint Kernighan-Lin search with joins and node relabeling, KLj$*$r, is a generalization of KLj that transforms a decomposition and a node labeling jointly. Both algorithms build on work of Kernighan and Lin [15].



(a) Decomposition (b) Node Labeling

Figure 1: This article studies an optimization problem whose feasible solutions define both a decomposition (a) and a node labeling (b) of a given graph $G = (V, E)$. A decomposition of $G$ is a partition $\Pi$ of the node set $V$ such that, for every $V' \in \Pi$, the subgraph of $G$ induced by $V'$ is connected. A node labeling of $G$ is a map $f : V \to L$ from its node set $V$ to a finite, non-empty set $L$ of labels.

Toward applications, the NL-LMP offers a common mathematical abstraction of seemingly unrelated computer vision tasks, including multiple object tracking, instance-separating semantic segmentation and articulated human body pose estimation. For these three applications, the abstraction is introduced below and described in more detail in the later sections. Also for these three applications, benchmark data sets and feasible solutions found by our algorithms, we report state-of-the-art application-specific accuracy.

*Multiple object tracking* [3, 4, 6, 9, 17, 22, 24, 35, 36] can be seen as a task requiring two classes of decisions: For every point in an image, one needs to decide whether this point depicts an object or background. For every pair of points that depict objects, one needs to decide if the object is the same. Tang et al. [31, 32] abstract this task as a graph decomposition and node labeling problem w.r.t. a finite graph whose nodes are bounding boxes and w.r.t. 01-labels indicating that a bounding box depicts an object or background. Tang et al. solve the graph decomposition and the node labeling problem separately. By applying our proposed algorithms to the joint problem of [32], we obtain more accurate tracks for the multiple object tracking benchmark [21] than any published work.

*Instance-separating semantic segmentation* [7, 8, 20, 27, 28, 29, 37, 38] can be seen as a task requiring two classes of decisions: To every point in an image, one needs to assign a label that identifies a class of objects (e.g. human, car, bicycle, etc.). For every pair of points of the same class, one needs to decide if the object is the same. Kroeger et al. [19] state this problem as a multi-terminal cut problem w.r.t. a (super)pixel adjacency graph of the image. We generalize their problem to larger feasible sets. While Kroeger et al. [19] show qualitative results, we apply our algorithms to instances of the problem from the KITTI [11] and Cityscapes [7] benchmarks, obtaining more accurate results for Cityscapes than any published work.

*Articulated human body pose estimation* can be seen as a task requiring two classes of decisions: For every point in an image, one needs to decide whether it depicts a part of the human body. For every pair of points that depict body parts, one needs to decide if they belong to the same body. Pishchulin et al. [25] and Insafutdinov et al. [13] abstract this problem as a graph decomposition and node labeling problem w.r.t. a finite graph whose nodes are putative detections of body parts and w.r.t. labels that idenfity body part classes (head, wrist, etc.) and background. By substantially reducing the running time for this task compared to their branch-and-cut algorithm (that computes also lower bounds), we can tackle instances of the problem with more nodes. This allows us to obtain more accurate pose estimates for the MPII Human Pose Dataset [2] than any published work.

## 2. Problem

In this section, we define the minimum cost node labeling lifted multicut problem, NL-LMP. Sections 2.1–2.3 offer an intuition for its parameters, feasible solutions and cost function. Section 2.4 offers a concise and rigorous definition. Section 2.5 discusses special cases.

### 2.1. Parameters

Any instance of the NL-LMP is defined with respect to the following parameters:

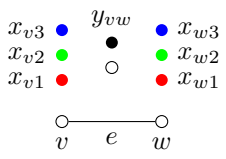
- A connected graph $G = (V, E)$ whose decompositions we care about, e.g. the pixel grid graph of an image.
- A graph $G' = (V, E')$ with $E \subseteq E'$. This graph can contain as edges pairs of nodes that are not neighbors in $G$. It defines the structure of the cost function.
- A digraph $H = (V, A)$ that fixes an arbitrary orientation of the edges $E'$. That is, for every edge $\{v, w\}$ of $G'$, the graph $H$ contains either the edge $(v, w)$ or the edge $(w, v)$. Formally, $H$ is such that for all $v, w \in V$:

$$\{v, w\} \in E' \Leftrightarrow (v, w) \in A \vee (w, v) \in A \quad (1)$$
$$(v, w) \notin A \vee (w, v) \notin A \quad (2)$$

- A finite, non-empty set $L$ called the set of *(node) labels*
- The following functions whose values are called *costs*:
  - $c : V \times L \to \mathbb{R}$. For any node $v \in V$ and any label $l \in L$, the cost $c_{vl}$ is payed iff $v$ is labeled $l$.
  - $c^{\sim} : A \times L^2 \to \mathbb{R}$. For any edge $vw \in A$ and any labels $ll' \in L^2$, the cost $c^{\sim}_{vw,ll'}$ is payed iff $v$ is labeled $l$ and $w$ is labeled $l'$ and $v$ and $w$ are in the same component.
  - $c^{\not\sim} : A \times L^2 \to \mathbb{R}$. For any edge $vw \in A$ and any labels $ll' \in L^2$, the cost $c^{\not\sim}_{vw,ll'}$ is payed iff $v$ is labeled $l$ and $w$ is labeled $l'$ and $v$ and $w$ are in distinct components.

### 2.2. Feasible Set

Every feasible solution of the NL-LMP is a pair $(x, y)$ of 01-vectors $x \in \{0,1\}^{V \times L}$ and $y \in \{0,1\}^{E'}$. More specifically, $x$ is constrained such that, for every node $v \in V$, there is precisely one label $l \in L$ with $x_{vl} = 1$. $y$ is constrained so as to well-define a decomposition of $G$ by the set $\{e \in E \,|\, y_e = 1\}$ of those edges that straddle distinct components. Formally, $(x, y) \in X_{VL} \times Y_{GG'}$ with $X_{VL}$ and $Y_{GG'}$ defined below.




- $X_{VL} \subseteq \{0,1\}^{V \times L}$, the set of all characteristic functions of maps from $V$ to $L$, i.e., the set of all $x \in \{0,1\}^{V \times L}$ such that

$$\forall v \in V : \quad \sum_{l \in L} x_{vl} = 1 \ . \quad (3)$$

  For any $x \in X$, any $v \in V$ and any $l \in L$ with $x_{vl} = 1$, we say that node $v$ is *labeled* $l$ by $x$.
- $Y_{GG'} \subseteq \{0,1\}^{E'}$, the set of all characteristic functions of multicuts of $G'$ lifted from $G$ [1]. For any $y \in Y_{GG'}$ and any $e = \{v, w\} \in E'$, $y_e = 1$ indicates that $v$ and $w$ are in distinct components of the decomposition

of $G$ defined by the multicut $\{e' \in E \,|\, y_{e'} = 1\}$ of $G$. Formally, $Y_{GG'}$ is the set of all $y \in \{0,1\}^{E'}$ that satisfy the following system of linear inequalities:

$$\forall C \in \text{cycles}(G)\, \forall e \in C : \; y_e \leq \sum_{e' \in C \setminus \{e\}} y_{e'} \tag{4}$$

$$\forall \{v,w\} \in E' \setminus E\, \forall P \in vw\text{-paths}(G) : \quad y_{\{v,w\}} \leq \sum_{e \in P} y_e \tag{5}$$

$$\forall \{v,w\} \in E' \setminus E\, \forall C \in vw\text{-cuts}(G) : \quad 1 - y_{\{v,w\}} \leq \sum_{e \in C} (1 - y_e) \;\; . \tag{6}$$

## 2.3. Cost Function

For every $x \in \{0,1\}^{V \times L}$ and every $y \in \{0,1\}^{A \times L^2}$, a cost $\varphi(x,y) \in \mathbb{R}$ is defined by the form

$$\begin{aligned} \varphi(x,y) = & \sum_{v \in V} \sum_{l \in L} c_{vl}\, x_{vl} \\ & + \sum_{vw \in A} \sum_{ll' \in L^2} c^{\sim}_{vw,ll'}\, x_{vl}\, x_{wl'}\, (1 - y_{\{v,w\}}) \\ & + \sum_{vw \in A} \sum_{ll' \in L^2} c^{\not\sim}_{vw,ll'}\, x_{vl}\, x_{wl'}\, y_{\{v,w\}} \;\; . \end{aligned} \tag{7}$$

## 2.4. Definition

We define the NL-LMP rigorously and concisely in the form of a linearly constrained binary cubic program.

**Definition 1** For any connected graph $G = (V,E)$, any graph $G' = (V,E')$ with $E \subseteq E'$, any orientation $H = (V,A)$ of $G'$, any finite, non-empty set $L$, any function $c : V \times L \to \mathbb{R}$ and any functions $c^{\sim}, c^{\not\sim} : A \times L^2 \to \mathbb{R}$, the instance of the *minimum cost node-labeling lifted multicut problem* (NL-LMP) with respect to $(G, G', H, L, c, c^{\sim}, c^{\not\sim})$ has the form

$$\min_{(x,y) \in X_{VL} \times Y_{GG'}} \varphi(x,y) \;\; . \tag{8}$$

## 2.5. Special Cases

Below, we show that the NL-LMP generalizes the UIQP. This connects the NL-LMP to work on graphical models with second-order functions and finitely many labels. In addition, we show that NL-LMP generalizes the LMP, connecting the NL-LMP to recent work on lifted multicuts. Finally, we show that the NL-LMP is general enough to express subgraph selection, connectedness and disconnectedness constraints.

### 2.5.1 Unconstrained Integer Quadratic Program

**Definition 2** For any graph $G' = (V,E')$, any orientation $H = (V,A)$ of $G'$, any finite, non-empty set $L$, any $c : V \times L \to \mathbb{R}$ and any $c' : A \times L^2 \to \mathbb{R}$, the instance of the UIQP with respect to $(G', H, L, c, c')$ has the form

$$\min_{x \in X_{VL}} \sum_{v \in V} \sum_{l \in L} c_{vl}\, x_{vl} + \sum_{vw \in A} \sum_{ll' \in L^2} c'_{vw,ll'}\, x_{vl}\, x_{wl'} \;\; . \tag{9}$$

**Lemma 1** *For any graph $G' = (V,E')$, any instance $(G', H, L, c, c')$ of the* UIQP *and any $x \in X_{VL}$, $x$ is a solution of this instance of the* UIQP *iff $(x, 1_{E'})$ is a solution of the instance $(G', G', H, L, c, c', c')$ of the* NL-LMP.

PROOF Without loss of generality, we can assume that $G'$ is connected. (Otherwise, we add edges between nodes $v, w \in V$ as necessary and set $c'_{vw,ll'} = 0$ for any $l, l' \in L$.)

For any $x \in X_{GL}$, the pair $(x, 1_{E'})$ is a feasible solution of the instance of the NL-LMP because the map $1_{E'} : E' \to \{0,1\} : e \mapsto 1$ is such that $1_{E'} \in Y_{G'G'}$.

Moreover, $(x, 1_{E'})$ is a solution of the instance of the NL-LMP iff $x$ is a solution of the instance of the UIQP because, for $c^{\not\sim} = c^{\sim}$, the form (7) of the cost function of the NL-LMP specializes to the form (9) of the cost function of the UIQP.

### 2.5.2 Minimum Cost Lifted Multicut Problem

**Definition 3** [1] For any connected graph $G = (V,E)$, any graph $G' = (V,E')$ with $E \subseteq E'$ and any $c' : E' \to \mathbb{R}$, the instance of the minimum cost lifted multicut problem (LMP) with respect to $(G, G', c')$ has the form

$$\min_{y \in Y_{GG'}} \sum_{e \in E'} c'_e y_e \;\; . \tag{10}$$

**Lemma 2** *Let $(G, G', c')$ be any instance of the* LMP. *Let $(G, G', H, L, c, c^{\sim}, c^{\not\sim})$ be the instance of the* NL-LMP *with the same graphs and such that*

$$L = \{1\} \quad c = 0 \quad c^{\sim} = 0 \tag{11}$$

$$\forall (v,w) \in A : \quad c^{\not\sim}_{vw,11} = c'_{\{v,w\}} \;\; . \tag{12}$$

*Then, for any $y \in \{0,1\}^{E'}$, $y$ is a solution of the instance of the* LMP *iff $(1_{V \times L}, y)$ is a solution of the instance of the* NL-LMP.

PROOF Trivially, $y$ is a feasible solution of the instance of the LMP iff $(1_{V \times L}, y)$ is a feasible solution of the instance of the NL-LMP. More specifically, $y$ is a solution of the instance of the LMP iff $(1_{V \times L}, y)$ is a solution of the instance of the NL-LMP because, for any $x \in X_{VL}$, the cost function (7) of the NL-LMP assumes the special form below which is identical with the form in (10).

$$\varphi(x,y) \overset{(3),(11)}{=} \sum_{vw \in A} c^{\not\sim}_{vw,11} y_{\{v,w\}} \overset{(12)}{=} \sum_{e \in E'} c'_e y_e \;\; . \tag{13}$$

#### 2.5.3 Subgraph Selection

Applications such as [13, 25, 31, 32] require us to not only decompose a graph and label its nodes but to also select a subgraph. The NL-LMP is general enough to model subgraph selection. To achieve this, one proceeds in two steps: Firstly, one introduces a special label $\epsilon \in L$ to indicate that a node is not an element of the subgraph. We call these nodes *inactive*. All other nodes are called *active*. Secondly, one chooses a large enough $c^* \in \mathbb{N}$, a $c^\dagger \in \mathbb{N}_0$ and $c^\sim, c^\not\sim$ such that

$$\forall vw \in A \, \forall l \in L \setminus \{\epsilon\} : \quad c^\sim_{vw,l\epsilon} = c^\sim_{vw,\epsilon l} = c^* \tag{14}$$

$$c^\not\sim_{vw,l\epsilon} = c^\not\sim_{vw,\epsilon l} = 0 \tag{15}$$

$$\forall vw \in A : \quad c^\sim_{vw,\epsilon\epsilon} = c^\dagger \ . \tag{16}$$

By (14), inactive nodes are not joined with active nodes in the same component. By (15), cutting an inactive node from an active node has zero cost. By (16), joining inactive nodes has cost $c^\dagger$, possibly zero. Choosing $c^\dagger$ large enough implements an additional constraint proposed in [31] that inactive nodes are necessarily isolated. It is by this constraint and by a two-elementary label set that [31] is a specialization of the NL-LMP.

#### 2.5.4 (Dis-)Connectedness Constraints

Some applications require us to constrain certain nodes to be in distinct components. One example is instance-separating semantic segmentation where nodes with distinct labels necessarily belong to distinct segments [19]. Other applications require us to constrain certain nodes to be in the same component. One example is articulated human body pose estimation for a single human in the optimization framework of [25] where every pair of active nodes necessarily belongs to the same human. Another example is connected foreground segmentation [23, 26, 30, 34] in which every pair of distinct foreground pixels necessarily belongs to the same segment.

The NL-LMP is general enough to model a combination of connectedness constraints and disconnectedness constraints by sufficiently large costs: In order to constrain distinct nodes $v, w \in V$ with labels $l, l' \in L$ to be in *the same component*, one introduces an edge $(v, w) \in A$, a large enough $c^* \in \mathbb{N}$ and costs $c^\not\sim$ such that $c^\not\sim_{vw,ll'} = c^\not\sim_{vw,l'l} = c^*$. In order to constrain distinct nodes $v, w \in V$ with labels $l, l' \in L$ to be in *distinct components*, one introduces an edge $(v, w) \in A$, a large enough $c^* \in \mathbb{N}$ and costs $c^\sim$ such that $c^\sim_{vw,ll'} = c^\sim_{vw,l'l} = c^*$. Constraining nodes with the same label to the same component constrains the feasible decompositions to be $|L|$-colorable, For $|L| = 2$ in particular, a constrained NL-LMP specializes to the MAX-CUT problem.

## 3. Algorithms

In this section, we define two local search algorithms that compute feasible solutions of the NL-LMP efficiently. Both algorithms attempt to improve the current feasible solution recursively by *transformations*. One class of transformations alters the node labeling of the graph by replacing a single node label. A second class of transformations alters the decomposition of the graph by moving a single node from one component to another. A third class of transformations alters the decomposition of the graph by joining two components.

As proposed by Kernighan and Lin [15] and generalized to the LMP by Keuper et al. [16], a local search is carried out not over the set of individual transformations of the current feasible solution but over a set of sequences of transformations. Complementary to this idea, we define and implement two schemes of combining transformations of the decomposition of the graph with transformations of the node labeling of the graph. This leads us to define two local search algorithms for the NL-LMP.

### 3.1. Encoding Feasible Solutions

To encode feasible solutions $(x, y) \in X_{VL} \times Y_{GG'}$ of the NL-LMP, we consider two maps: A *node labeling* $\lambda : V \to L$ that defines the $x^\lambda \in X_{VL}$ such that

$$\forall v \in V \, \forall l \in L : \quad x^\lambda_{vl} = 1 \Leftrightarrow \lambda(v) = l \ , \tag{17}$$

and a so-called *component labeling* $\mu : V \to \mathbb{N}$ that defines the $y^\mu \in \{0, 1\}^{E'}$ such that

$$\forall \{v, w\} \in E' : \quad y^\mu_{\{v,w\}} = 0 \Leftrightarrow \mu(v) = \mu(w) \ . \tag{18}$$

### 3.2. Transforming Feasible Solutions

To improve feasible solutions of the NL-LMP recursively, we consider three transformations of the encodings $\lambda$ and $\mu$:

For any node $v \in V$ and any label $l \in L$, the transformation $T_{vl} : L^V \to L^V : \lambda \mapsto \lambda'$ changes the label of the node $v$ to $l$, i.e.

$$\forall w \in V : \quad \lambda'(w) := \begin{cases} l & \text{if } w = v \\ \lambda(w) & \text{otherwise} \end{cases} \ . \tag{19}$$

For any node $v \in V$ and any component index $m \in \mathbb{N}$, the transformation $T'_{vm} : \mathbb{N}^V \to \mathbb{N}^V : \mu \mapsto \mu'$ changes the component index of the node $v$ to $m$, i.e.

$$\forall w \in V : \quad \mu'(w) := \begin{cases} m & \text{if } w = v \\ \mu(w) & \text{otherwise} \end{cases} \ . \tag{20}$$

For any component indices $m, m' \in \mathbb{N}$, the transformation $T'_{mm'} : \mathbb{N}^V \to \mathbb{N}^V : \mu \mapsto \mu'$ puts all nodes currently in the component indexed by $m$ into the component indexed by $m'$, i.e.

$$\forall w \in V : \quad \mu'(w) := \begin{cases} m' & \text{if } \mu(w) = m \\ \mu(w) & \text{otherwise} \end{cases} \ . \tag{21}$$

Not every component labeling $\mu$ is such that $y^\mu \in Y_{GG'}$. In fact, $y^\mu$ is feasible if and only if, for every $m \in \mu(V)$, the node set $\mu^{-1}(m)$ is connected in $G$. For efficiency, we allow for transformations (20) whose output $\mu'$ violates this condition, as proposed in [16]. This happens when an *articulation node* of a component is moved to a different component. In order to *repair* any $\mu'$ for which $y^\mu$ is infeasible, we consider a map $R : \mathbb{N}^V \to \mathbb{N}^V : \mu' \mapsto \mu$ such that, for any $\mu' : V \to \mathbb{N}$ and any distinct $v, w \in V$, we have $\mu(v) = \mu(w)$ if and only if the exists a $vw$-path in $G$ along which all nodes have the label $\mu'(v)$. We implement $R$ as connected component labeling by breadth-first-search.

### 3.3. Searching Feasible Solutions

We now define two local search algorithms that attempt to improve an initial feasible solution recursively, by applying the transformation defined above. Initial feasible solutions are given, for instance, by the finest decomposition of the graph $G$ that puts every node in a distinct component, or by the coarsest decomposition of the graph $G$ that puts every node in the same component, each together with any node labeling. We find an initial feasible solution for our local search algorithm by first fixing an optimal label for every node independently and by then solving the resulting LMP, i.e., (8) for the fixed labels $x \in X_{VL}$, by means of greedy agglomerative edge contraction [16].

**KLj/r Algorithm.** The first local search algorithm we define, alternating Kernighan-Lin search with joins and node relabeling, KLj/r, alternates between transformations of the node labeling and transformations of the decomposition. For a fixed decomposition, the labeling is transformed by Func. 1 which greedily updates labels of nodes independently. For a fixed labeling, the decomposition is transformed by Func. 2, *without those parts of the function that are written in green*, i.e., precisely the algorithm KLj of [16]. (All symbols that appear in the pseudo-code are defined above, except the iteration counter $t$, cost differences $\delta, \Delta$, and 01-vectors $\alpha$ used for bookkeeping, to avoid redundant operations.)

**KLj∗r Algorithm.** The second local search algorithm we define, joint Kernighan-Lin search with joins and node relabeling, KLj∗r, transforms the decomposition and the node labeling jointly, by combining the transformations (19)–(21) in a novel manner. It is given by Func. 2, *with those parts of the function that are written in green*.

Like the alternating algorithm KLj/r, the joint algorithm KLj∗r updates the labeling for a fixed decomposition (calls of Func. 1 from Func. 2). Unlike the alternating algorithm KLj/r, the joint algorithm KLj∗r updates the decomposition and the labeling also jointly. This happens in Func. 3 that is called from KLj∗r, *with the part that is written in green*.

Func. 3 looks at two components $V := \mu^{-1}(m)$ and $W := \mu^{-1}(' m)$ of the current decomposition. It attempts to improve the decomposition as well as the labeling by moving

Function 1: $(\Delta, \lambda') =$ update-labeling$(\mu, \lambda)$

$\lambda_0 := \lambda \quad \Delta := 0 \quad t := 0$
***repeat***
  ***choose*** $(\hat{v}, \hat{l}) \in \operatorname{argmin}_{(v,l) \in V \times L} \varphi(x^{T_{vl}(\lambda_t)}, y^{\mu_t}) - \varphi(x^{\lambda_t}, y^{\mu_t})$
  $\delta := \varphi(x^{T_{\hat{v}\hat{l}}(\lambda_t)}, y^{\mu_t}) - \varphi(x^{\lambda_t}, y^{\mu_t})$
  ***if*** $\delta < 0$
    $\lambda_{t+1} := T_{\hat{v}\hat{l}}(\lambda_t)$
    $\Delta := \Delta + \delta$
    $t := t + 1$
  ***else***
    ***return*** $(\Delta, \lambda_t)$

a node from $V$ to $W$ or from $W$ to $V$ *and by simultaneously changing its label*. As proposed by Kernighan and Lin [15], Func. 3 does not make such transformations greedily but first constructs a sequence of such transformations greedily and then executes the first $k$ with $k$ chosen so as to decrease the objective value maximally. KLj/r constructs a sequence of transformations analogously, but the node labeling remains fixed throughout every transformation of the decomposition. Thus, KLj∗r is a local search algorithm whose local neighborhood is strictly larger than that of KLj/r.

Function 2: $(\Delta', \mu', \lambda') =$ update-lifted-multicut$(\mu, \lambda)$

$\mu_0 := \mu \qquad t := 0$
$(\delta, \lambda_0) :=$ update-labeling$(\mu_0, \lambda)$
***let*** $\alpha_0 : \mathbb{N} \to \{0, 1\}$ such that $\alpha_0(\mathbb{N}) = 1$
***repeat***
  $\Delta := 0 \qquad \mu_{t+1} := \mu_t \qquad \lambda_{t+1} := \lambda_t$
  ***let*** $\alpha_{t+1} : \mathbb{N} \to \{0, 1\}$ such that $\alpha_{t+1}(\mathbb{N}) = 0$
  ***for each*** $\{m, m'\} \in \binom{\mu(V)}{2}$
    ***if*** $\alpha_t(m) = 0 \wedge \alpha_t(m') = 0$
      ***continue***
    $(\delta, \mu_{t+1}, \lambda_{t+1}) :=$ update-2-cut$(\mu_{t+1}, \lambda_{t+1}, m, m')$
    ***if*** $\delta < 0$
      $\alpha_{t+1}(m) := 1 \quad \alpha_{t+1}(m') := 1 \quad \Delta := \Delta + \delta$
  ***for each*** $m \in \mu(V)$
    ***if*** $\alpha_t(m) = 0$
      ***continue***
    $m' := 1 + \max \mu(V)$ (new component)
    $(\delta, \mu_{t+1}, \lambda_{t+1}) :=$ update-2-cut$(\mu_{t+1}, \lambda_{t+1}, m, m')$
    ***if*** $\delta < 0$
      $\alpha_{t+1}(m) := 1 \quad \alpha_{t+1}(m') := 1 \quad \Delta := \Delta + \delta$
  $(\delta, \lambda_{t+1}) :=$ update-labeling$(\mu_{t+1}, \lambda_{t+1})$
  $\Delta := \Delta + \delta$
  ***if*** $y^{\mu_{t+1}} \notin Y_{GG'}$
    $\mu_{t+1} := R(\mu_{t+1})$ (repair heuristic)
    $\Delta := \varphi(x^{\lambda_{t+1}}, y^{\mu_{t+1}}) - \varphi(x^{\lambda_0}, y^{\mu_0})$
  $t := t + 1$
***while*** $\Delta < 0$

Our C++ implementation computes cost differences incrementally and solves the optimization problem over transformations by means of a priority queue, as described in detail in the supplement. The time and space complexities are identical to those of KLj and are established in [16], as transformations that take linear time in the number of labels take constant time in the size of the graph.

Function 3: $(\Delta', \mu', \lambda') = \text{update-2-cut}(\mu, \lambda, m, m')$

$\mu_0 := \mu \qquad \lambda_0 := \lambda \qquad t := 0$
***if*** $\mu^{-1}(m') = \emptyset$
  $V_0 := \mu^{-1}(m)$
***else***
  $V_0 := \{v \in \mu^{-1}(m) \,|\, \exists w \in \mu^{-1}(m') : \{v, w\} \in E\}$
***if*** $\mu^{-1}(m) = \emptyset$
  $W_0 := \mu^{-1}(m')$
***else***
  $W_0 := \{w \in \mu^{-1}(m') \,|\, \exists v \in \mu^{-1}(m) : \{v, w\} \in E\}$
***let*** $\alpha : \mathbb{N} \to \{0, 1\}$ such that $\alpha(\mathbb{N}) = 1$
***while*** $V_t \cup W_t \neq \emptyset$
  $\delta := \delta' := \infty$
  ***if*** $V_t \neq \emptyset$
    ***choose*** $(\hat{v}, \hat{l}) \in \underset{(v,l) \in V_t \times L}{\operatorname{argmin}} \varphi(x^{T_{vl}(\lambda_t)}, y^{T'_{vm'}(\mu_t)}) - \varphi(x^{\lambda_t}, y^{\mu_t})$
    $\delta := \varphi(x^{T_{\hat{v}\hat{l}}(\lambda_t)}, y^{T'_{\hat{v}m'}(\mu_t)}) - \varphi(x^{\lambda_t}, y^{\mu_t})$
  ***if*** $W_t \neq \emptyset$
    ***choose*** $(\hat{w}, \hat{l}) \in \underset{(w,l) \in W_t \times L}{\operatorname{argmin}} \varphi(x^{T_{wl}(\lambda_t)}, y^{T'_{wm}(\mu_t)}) - \varphi(x^{\lambda_t}, y^{\mu_t})$
    $\delta' := \varphi(x^{T_{\hat{w}\hat{l}}(\lambda_t)}, y^{T'_{\hat{w}m}(\mu_t)}) - \varphi(x^{\lambda_t}, y^{\mu_t})$
  ***if*** $\delta \leq \delta'$
    $\mu_{t+1} := T'_{\hat{v}m'}(\mu_t)$ (move node $\hat{v}$ to component $m'$)
    $\lambda_{t+1} := T_{\hat{v}\hat{l}}(\lambda_t)$ (label node $\hat{v}$ with label $\hat{\lambda}$)
    $\alpha(\hat{v}) := 0$ (mark $\hat{v}$ as inactive)
  ***else***
    $\mu_{t+1} := T'_{\hat{w}m}(\mu_t)$ (move node $\hat{w}$ to component $m$)
    $\lambda_{t+1} := T_{\hat{w}\hat{l}}(\lambda_t)$ (label node $\hat{w}$ with label $\hat{\lambda}$)
    $\alpha(\hat{w}) := 0$ (mark $\hat{w}$ as inactive)
  $V_{t+1} := \{v \in V \,|\, \mu_{t+1}(v) = m \wedge \alpha(v) = 1 \wedge \exists \{v, w\} \in E : \mu_{t+1}(w) = m'\}$
  $W_{t+1} := \{w \in V \,|\, \mu_{t+1}(w) = m' \wedge \alpha(w) = 1 \wedge \exists \{v, w\} \in E : \mu_{t+1}(v) = m\}$
  $t := t + 1$
$\hat{t} := \min \underset{t' \in \{0,\ldots,t\}}{\operatorname{argmin}} \varphi(x^{\lambda_{t'}}, y^{\mu_{t'}}) - \varphi(x^{\lambda_0}, y^{\mu_0})$
$\Delta_1 := \varphi(x^{\lambda_{\hat{t}}}, y^{\mu_{\hat{t}}}) - \varphi(x^{\lambda_0}, y^{\mu_0})$
$\Delta_2 := \varphi(x^{\lambda_0}, y^{T'_{mm'}(\mu)}) - \varphi(x^{\lambda_0}, y^{\mu_0})$ (join $m$ and $m'$)
***if*** $\min\{\Delta_1, \Delta_2\} \geq 0$
  ***return*** $(0, \mu, \lambda)$
***else if*** $\Delta_1 < \Delta_2$
  ***return*** $(\Delta_1, \mu_{\hat{t}}, \lambda_{\hat{t}})$
***else***
  ***return*** $(\Delta_2, T_{mm'}(\mu), \lambda)$

# 4. Applications

We show applications of the proposed problem and algorithms to three distinct computer vision tasks: articulated human body pose estimation, multiple object tracking, and instance-separating semantic segmentation. For each task, we set up instances of the NL-LMP from published data, using published algorithms.

## 4.1. Articulated Human Body Pose Estimation

We turn toward applications of the NL-LMP and the algorithms KLj/r and KLj∗r to the task of estimating the articulated poses of all humans visible in an image. Pishchulin et al. [25] and Insafutdinov et al. [13] approach this problem via a graph decomposition and node labeling problem that we identify as a special case of the NL-LMP with $c^{\not\sim} = 0$ and with subgraph selection (Section 2.5.3). We relate their notation to ours in the supplement. Nodes in their graph are putative detections of body parts. Labels define body part classes (head, wrist, etc.). In our notation, $x_{vl} = 1$ indicates that the putative detection $v$ is a body part of class $l$, and $y_{vw} = 1$ indicates that the body parts $v$ and $w$ belong to distinct humans. The test set of [13] consists of 1758 such instances of the NL-LMP.

To tackle these instances, Insafutdinov et al. define and implement a branch-and-cut algorithm in the integer linear programming software framework Gurobi. We refer to their published C++ implementation as B&C.

**Cost and time.** In Fig. 2, we compare the convergence of B&C (feasible solutions and lower bounds) with the convergence of our algorithms, KLj/r and KLj∗r (feasible solutions only). Shown in this figure is the average objective value over the test set w.r.t. the absolute running time. Thanks to the lower bounds obtained by B&C, it can be seen from this figure that KLj/r and KL+r arrive at near optimal feasible solutions after $10^{-1}$ seconds, five orders of magnitude faster than B&C. This result shows that primal feasible heuristics for the NL-LMP, such as KLj/r and KLj∗r, are practically useful in the context of this application.

**Application-specific accuracy.** In Tab. 1, we compare feasible solutions output by KLj/r and KLj∗r after convergence with those obtained by B&C after at most three hours. It can be seen from this table that the feasible solutions output by KLj/r and KLj∗r have lower cost and higher application-specific accuracy (Acc) on average. KLj∗r yields a lower average cost than KLj/r with slightly higher running time. The fact that lower cost does not mean higher application-specific accuracy is explained by the application-specific accuracy measure that does not penalize false positives.

The shorter absolute running time of KLj/r and KLj∗r allows us to increase the number of nodes from 150, as in [13], to 420. It can be seen from the last two rows of Tab. 1 that this increases the application-specific accuracy by 4%.

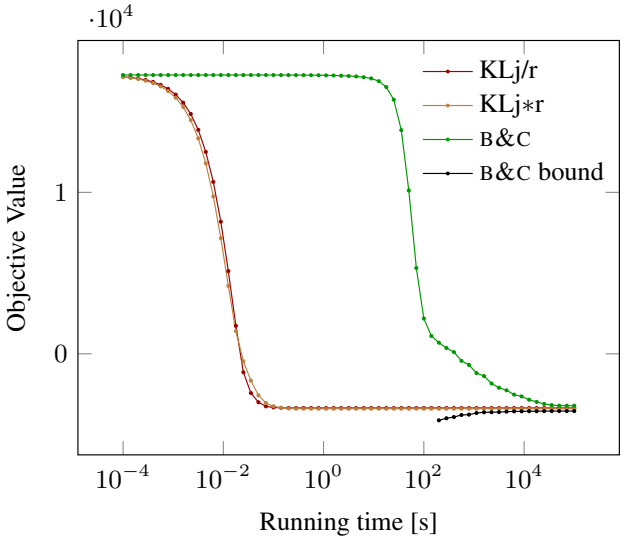



Figure 2: Convergence of B&C, KLj/r and KLj∗r in an application to the task of articulated human body pose estimation.

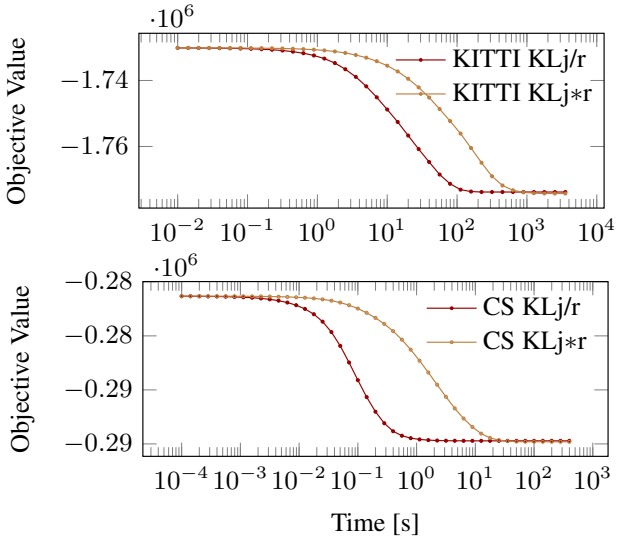



Figure 3: Convergence of KLj/r and KLj∗r in an application to the task of instance-separating semantic segmentation.

| $\|V\|$ | Alg. | AP | Mean cost | Mean time [s] | Median time [s] |
|---|---|---|---|---|---|
| 150 | [13] | 65.5 | -3013.30 | 9519.26 | 308.28 |
| | KLj/r | 66.5 | -3352.74 | **0.033** | **0.031** |
| | KLj∗r | **66.6** | **-3419.07** | 0.119 | 0.100 |
| 420 | KLj/r | **70.6** | -6184.36 | **0.098** | **0.053** |
| | KLj∗r | **70.6** | **-6608.53** | 0.534 | 0.254 |

Table 1: Comparison of B&C [13], KLj/r and KLj∗r in an application to the task of human body pose estimation.

### 4.2. Instance-Separating Semantic Segmentation

We turn toward applications of the NL-LMP and the algorithms KLj/r and KLj∗r to the task of instance-separating semantic image segmentation. We state this problem here as an NL-LMP whose nodes correspond to pixels in a given image, and whose labels define classes of objects (human, car, bicycle, etc.). In our notation, $x_{vl} = 1$ indicates that the pixel $v$ shows an object of class $l$, and $y_{vw} = 1$ indicates that the pixels $v$ and $w$ belong to distinct objects.

Specifically, we apply the algorithms KLj/r and KLj∗r to instances of the NL-LMP for the task of instance-separating semantic segmentation posed by the KITTI [11] and Cityscapes [7] benchmarks. For KITTI, we construct instances of the NL-LMP from data published by Uhrig et al. [33] as described in detail in the supplement. For Cityscapes, we construct instances of the NL-LMP as follows. For costs $c^{\sim}$, we again use data of Uhrig et al. [33]. For costs $c$, we use a ResNet-50 [12] network with dilated convolutions [5]. We train the network in a fully convolutional manner with image crops (768 px·512 px) subjected to minimal data augmentation (horizontal flips). More details are in the supplement.

| Data | Algorithm | AP | $AP^{50\%}$ |
|---|---|---|---|
| KITTI validation [11] | KLj/r | **50.5** | **82.9** |
| | KLj∗r | 50.3 | 82.4 |
| KITTI test [11] | [33] | 41.6 | 69.1 |
| | KLj∗r | **43.6** | **71.4** |
| Cityscapes validation [7] | KLj/r | 11.3 | **26.8** |
| | KLj∗r | **11.4** | 26.1 |
| Cityscapes test [7] | MCG+R-CNN [7] | 4.6 | 12.9 |
| | [33] | 8.9 | 21.1 |
| | KLj∗r | **9.8** | **23.2** |

Table 2: Comparison of KLj/r and KLj∗r in an application to the task of instance-separating semantic segmentation.

**Cost and time.** In Fig. 3, we compare the convergence of KLj/r and KLj∗r. Shown in this figure w.r.t. the absolute running time are the average objective values over the KITTI and Cityscapes validation sets, respectively. It can be seen from this figure that KLj/r converges faster than KLj∗r. Both algorithms are practical for this application but not efficient enough for video processing in real-time.

**Application-specific accuracy.** In Tab. 2, we compare feasible solutions output by KLj/r and KLj∗r after convergence with the output of the algorithm of Uhrig et al [33]. It can be seen from this table that the application of KLj/r and KLj∗r improves the application-specific average precision, AP and $AP^{50\%}$. The AP of feasible solutions output by KLj∗r for the Cityscapes test set is higher than that of any published algorithm. A higher AP is reported by Kirillov et al. [18], who use the model and algorithms proposed in this paper with improved pairwise $c^{\sim}$ and unary $c$ costs.

| Method | MOTA ↑ | MOTP ↑ | FAF ↓ | MT ↑ | ML ↓ | FP ↓ | FN ↓ | ID Sw ↓ | Frag↓ | Hz ↑ | Detector |
|---|---|---|---|---|---|---|---|---|---|---|---|
| [10] | 41.0 | 74.8 | 1.3 | 11.6% | 51.3% | 7896 | 99224 | 430 | 963 | 1.1 | Public |
| [17] | 42.9 | 76.6 | 1.0 | 13.6% | 46.9% | **5668** | 97919 | 499 | 659 | 0.8 | Public |
| [6] | 46.4 | 76.6 | 1.6 | **18.3%** | 41.4% | 9753 | **87565** | **359** | **504** | 2.6 | Public |
| [32] | 46.3 | 75.7 | 1.1 | 15.5% | **39.7%** | 6373 | 90914 | 657 | 1114 | 0.8 | Public |
| KLj/r | **47.6** | **78.5** | 1.0 | 17.0% | 40.4% | 5844 | 89093 | 629 | 768 | **8.3** | Public |
| KLj∗r | **47.6** | **78.5** | **0.98** | 17.0% | 40.4% | 5783 | 89160 | 627 | 761 | 0.7 | Public |

Table 3: Comparison of the algorithms KLj/r and KLj∗r in an application to the task of multiple object tracking.

### 4.3. Multiple Object Tracking

We turn toward applications of the NL-LMP and the algorithms KLj/r and KLj∗r to the task of multiple object tracking. Tang et al. [31] approach this problem via a graph decomposition and node labeling problem that we identify as a special case of the NL-LMP with two labels and subgraph selection (Sec. 2.5.3). We relate their notation to ours rigorously in the supplement. Nodes in their graph are putative detections of persons. In our notation, $x_{vl} = 1$ indicates that the putative detection $v$ is active, and $y_{vw} = 1$ indicates that the putative detections $v$ and $w$ are of distinct persons. For the test set of the multiple object tracking benchmark [21], Tang et al. construct seven such instances of the NL-LMP.

To tackle these large instances, in [32] Tang et al. solve the subgraph suppression problem first and independently, by thresholding on the detections scores, and then solve the minimum cost multicut problem for the remaining subgraph by means of the algorithm KLj of [16], without re-iterating. Here, we apply to the joint NL-LMP the algorithms KLj/r and KLj∗r and compare their output to that of [32] and of other top-performing algorithms [6, 10, 17]. We use the same data as in [32], therefore the performance gain is due to our algorithms that solve the full problem [31].

**Cost and time.** The convergence of the algorithms KLj/r and KLj∗r is shown in Fig. 4. It can be seen from this figure that KLj/r converges faster than KLj∗r.

**Application-specific accuracy.** We compare the feasible solutions output by KLj/r and KLj∗r to the state-of-the-art for the benchmark [21]. To this end, we report in Tab. 3 the standard CLEAR MOT metric, including: multiple object tracking accuracy (MOTA), multiple object tracking precision (MOTP), mostly tracked object (MT), mostly lost (ML) and tracking fragmentation (FM). MOTA combines identity switches (ID Sw), false positives (FP) and false negatives (FN) and is most widely used. Our feasible solutions are published also at the benchmark website unser the names NLLMP (KLj/r) and NLLMPj (KLj∗r). It is can be seen from Tab. 3 that the feasible solutions obtained by KLj/r and KLj∗r rank first in MOTA and MOTP. Compared to [32], KLj/r and KLj∗r reduce the number of false positives and false negatives. The average inverse running time per frame of a video sequence (column "Hz" in the table) is better for KLj/r by a margin than for any other algorithm. Overall, these results show the practicality of the NL-LMP in conjunction with the local search algorithms KLj/r and KLj∗r for applications in multiple object tracking.

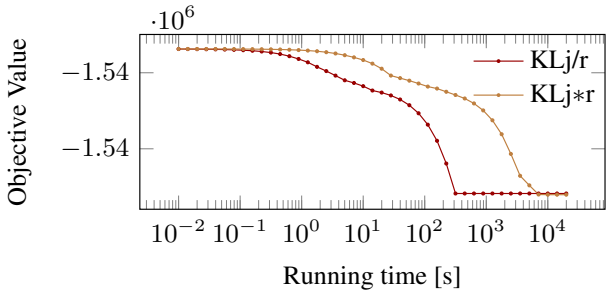



Figure 4: Convergence of the algorithms KLj/r and KLj∗r in an application to the task of multiple object tracking.

## 5. Conclusion

We have stated the minimum cost node labeling lifted multicut problem, NL-LMP, an NP-hard combinatorial optimization problem whose feasible solutions define both a decomposition and a node labeling of a given graph. We have defined and implemented two local search algorithms, KLj/r and KLj∗r, that converge monotonously to a local optimum, offering a feasible solution at any time. We have shown applications of these algorithms to the tasks of articulated human body pose estimation, multiple object tracking and instance-separating semantic segmentation, obtaining state-of-the-art application-specific accuracy. We conclude that the NL-LMP is a useful mathematical abstraction in the field of computer vision that allows researchers to apply the same optimization algorithm to diverse computer vision tasks. To foster collaboration between the fields of computer vision and combinatorial optimization, we make our code publicly available at https://github.com/bjoern-andres/graph

## References

[1] B. Andres, A. Fuksová, and J.-H. Lange. Lifting of multicuts. *CoRR*, abs/1503.03791, 2016. 2, 3

[2] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2D human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014. 2

[3] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *TPAMI*, 33(9):1806–1819, 2011. 2

[4] W. Brendel, M. Amer, and S. Todorovic. Multiobject tracking as maximum weight independent set. In *CVPR*, 2011. 2

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv:1606.00915*, 2016. 7

[6] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*, 2015. 2, 8

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes Dataset for semantic urban scene understanding. In *CVPR*, June 2016. 2, 7

[8] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016. 2

[9] L. Fagot-Bouquet, R. Audigier, Y. Dhome, and F. Lerasle. Improving multi-frame data association with sparse representations for robust near-online multi-object tracking. In *ECCV*, 2016. 2

[10] L. Fagot-Bouquet, R. Audigier, Y. Dhome, and F. Lerasle. Improving multi-frame data association with sparse representations for robust near-online multi-object tracking. In *ECCV*, 2016. 8

[11] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012. 2, 7

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[13] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele. DeeperCut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*, 2016. 2, 4, 6, 7

[14] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, T. Kröger, J. Lellmann, N. Komodakis, B. Savchynskyy, and C. Rother. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision*, 115(2):155–184, 2015. 1

[15] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell Systems Technical Journal*, 49:291–307, 1970. 1, 4, 5

[16] M. Keuper, E. Levinkov, N. Bonneel, G. Lavoué, T. Brox, and B. Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *ICCV*, 2015. 1, 4, 5, 6, 8

[17] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited. In *ICCV*, 2015. 2, 8

[18] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. Instancecut: from edges to instances with multicut. *CoRR*, abs/1611.08272, 2016. 7

[19] T. Kroeger, J. H. Kappes, T. Beier, U. Koethe, and F. A. Hamprecht. Asymmetric cuts: Joint image labeling and partitioning. In *GCPR*, 2014. 2, 4

[20] X. Liang, Y. Wei, X. Shen, J. Yang, L. Lin, and S. Yan. Proposal-free network for instance-level object segmentation. *CoRR*, abs/1509.02636, 2015. 2

[21] A. Milan, L. Leal-Taixé, I. D. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831*, 2016. 2, 8

[22] A. Milan, S. Roth, and K. Schindler. Continuous energy minimization for multitarget tracking. *TPAMI*, 36(1):58–72, 2014. 2

[23] S. Nowozin and C. H. Lampert. Global interactions in random field models: A potential function ensuring connectedness. *SIAM Journal on Imaging Sciences*, 3(4):1048–1074, 2010. 4

[24] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011. 2

[25] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. Gehler, and B. Schiele. DeepCut: Joint subset partition and labeling for multi person pose estimation. In *CVPR*, 2016. 2, 4, 6

[26] M. Rempfler, B. Andres, and B. Menze. The minimum cost connected subgraph problem in medical image analysis. In *MICCAI*, 2016. 4

[27] M. Ren and R. Zemel. End-to-end instance segmentation and counting with recurrent attention. In *arXiv:1605.09410 [cs.CV]*, 2016. 2

[28] B. Romera-Paredes and P. H. S. Torr. Recurrent instance segmentation. In *ECCV*, 2016. 2

[29] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 2

[30] J. Stühmer and D. Cremers. A fast projection method for connectivity constraints in image segmentation. In *EMMCVPR*, 2015. 4

[31] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph decomposition for multi-target tracking. In *CVPR*, 2015. 2, 4, 8

[32] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Multi-person tracking by multicut and deep matching. *CoRR*, abs/1608.05404, 2016. 2, 4, 8

[33] J. Uhrig, M. Cordts, U. Franke, and T. Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In *GCPR*, 2016. 7

[34] S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. In *CVPR*, 2008. 4

[35] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *ICCV*, 2015. 2

[36] A. R. Zamir, A. Dehghan, and M. Shah. GMCP-Tracker: Global multi-object tracking using generalized minimum clique graphs. In *ECCV*, 2012. 2

[37] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation with deep densely connected MRFs. In *CVPR*, 2016. 2

[38] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. Monocular object instance segmentation and depth ordering with CNNs. In *ICCV*, 2015. 2