Lifted Disjoint Paths with Application in Multiple Object Tracking

Andrea Hornakova [*1] **Roberto Henschel** [*2] **Bodo Rosenhahn** [2] **Paul Swoboda** [1]

Abstract

We present an extension to the disjoint paths problem in which additional *lifted* edges are introduced to provide path connectivity priors. We call the resulting optimization problem the lifted disjoint paths problem. We show that this problem is NP-hard by reduction from integer multi-commodity flow and 3-SAT. To enable practical global optimization, we propose several classes of linear inequalities that produce a high-quality LP-relaxation. Additionally, we propose efficient cutting plane algorithms for separating the proposed linear inequalities. The lifted disjoint path problem is a natural model for multiple object tracking and allows an elegant mathematical formulation for long range temporal interactions. Lifted edges help to prevent id switches and to re-identify persons. Our lifted disjoint paths tracker achieves nearly optimal assignments with respect to input detections. As a consequence, it leads on all three main benchmarks of the MOT challenge, improving significantly over state-of-the-art.

1. Introduction

The disjoint paths problem, a special case of the network flow problem with flows constrained to be binary, is a classical combinatorial optimization problem for which fast combinatorial solvers exist. It is a natural model for the multiple object tracking problem (MOT) in computer vision (Zhang et al., 2008). In the form of the tracking-by-detection paradigm, MOT consists of two steps: First, an object detector is applied to each frame of a video sequence to find the putative locations of all objects appearing in the video. Then, in the data association step, false positive detections are removed while correct detections are associated to the corresponding identities, thereby forming trajectories. In this work, we concentrate on the latter task.

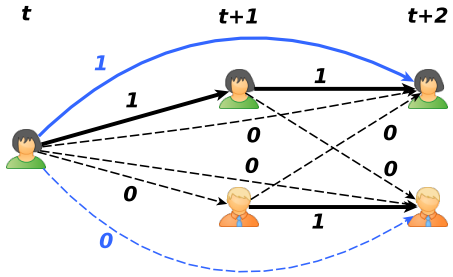

Figure 1. Illustration of connections between detections in three consecutive video frames together with a feasible labeling. We enhance the network flow model (black) with lifted edges (blue) representing whether two detections belong to the same track or not. Solid edges labeled by 1 represent active connections, dashed edges labeled by zero represent inactive connections between detections.

While for MOT even very large data association instances can be solved using the disjoint paths formulation, it has been shown that the basic disjoint paths problem alone is not sufficient to provide trajectories of high accuracy. The main limitation for MOT is the implicit assumption of a first-order Markov chain. In particular, costs only indicate whether two detections directly follow each other in a track.

Our contribution is three-fold: First, to overcome the limited expressiveness of disjoint paths, we propose to augment it with lifted edges which take into account long range interactions, see Figure 1. We call the resulting problem the *lifted disjoint paths problem*, see Section 3. We prove the problem to be NP-hard in Section 6. Second, we study the optimization problem from a polyhedral perspective, proposing a high-quality linear programming relaxation, see Section 4. Separation routines for the proposed constraints are described in Section 5. Third, we apply the lifted disjoint paths problem to MOT and show that our solver significantly outperforms state-of-the-art trackers on the popular MOT challenge, see Section 7.

We argue that our model has advantages from the modelling and optimization point of view. From the modelling standpoint, the lifted disjoint paths problem does not change the

[*]Equal contribution [1]Computer Vision and Machine Learning, Max Planck Institute for Informatics, Saarbrücken, Saarland, Germany [2]Institut for Image Processing, Leibniz University Hannover, Hannover, Niedersachsen, Germany. Correspondence to: Andrea Hornakova <andrea.hornakova@mpi-inf.mpg.de>, Roberto Henschel <henschel@tnt.uni-hannover.de>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

set of feasible solutions, but adds more expressive power to it. For MOT, this means that the set of feasible solutions, which naturally represent trajectories of objects, is preserved. The additional lifted edges represent connectivity priors. A lifted edge is active if and only if there is an active trajectory between its endpoints in the flow graph. For MOT, lifted edges take (dis-)similarity of object detection pairs represented by its endpoints into account. This allows to encourage or penalize an active path between the detections with possibly larger temporal distance. This helps to re-identify the same object and to prevent id-switches between distinct objects within long trajectories.

From the optimization point of view, we study several non-trivial classes of linear inequalities that result in a high-quality relaxation. The proposed inequalities depend non-trivially on the constraint structure of the underlying disjoint paths problem, see Section 4. We show that the polyhedral relaxation we consider is tighter than naively applying known inequalities. The proposed relaxation enables us to solve MOT problems via a global approach, in contrast to established approaches, which either use heuristics on complex models or global optimization on simpler models that do not exploit long range interaction. We present, to our knowledge, the first global optimization approach that incorporates long range interaction for MOT. This has several advantages: First, our optimization is not trapped in poor local optima or affected by initialization choices and is hence potentially more robust. Second, improvements in the discriminative power of features used to compute costs for the lifted disjoint paths problem directly correlate to better tracking performance, since no errors are introduced by suboptimal choices during optimization.

Finally, we note that the proposed lifted disjoint path formulation is not inherently tied to MOT and can potentially be applied to further problems not related to MOT.

Our code is available at `https://github.com/AndreaHor/LifT_Solver`.

2. Related Work

Disjoint paths problem. The disjoint paths problem can be solved with fast combinatorial solvers (Kovács, 2015). The shortest paths method for network flow specialized for the disjoint paths problem (Wang et al., 2019a) performs extremely well in practice. For the case of the two disjoint paths problem the specialized combinatorial algorithm by Suurballe's (Suurballe, 1974) can be used.

There exist several NP-complete extensions to the disjoint paths problem. The shortest disjoint paths problem with multiple source-sink pairs (Eilam-Tzoreff, 1998) is NP-complete, as is the more general integer multicommodity flow problem (Even et al., 1976). The special case of the disjoint paths problem with two distinct source/sink pairs can be solved in polynomial time, however (Tholey, 2012).

Connectivity priors & lifted edges. For several combinatorial problems, special connectivity inducing edges, which we will call lifted edges for our problem, have been introduced to improve expressiveness of the base problem.

In the Markov Random Field literature, special connectivity inducing edges were studied from a polyhedral point of view in (Nowozin & Lampert, 2010). They were used in image analysis to indicate that two non-adjacent pixels come from the same object and hence they must be part of a contiguously labeled component of the underlying graph.

For multicut (a.k.a. correlation clustering), a classical graph decomposition problem, lifted edges have been introduced in (Keuper et al., 2015) to model connectivity priors. A lifted edge expresses affinity of two nodes to be in the same/different connected component of the graph partition. Lifted multicut has been used for image and mesh segmentation (Keuper et al., 2015), connectomics (Beier et al., 2017) and cell tracking (Rempfler et al., 2017). A combination of the lifted multicut problem and Markov Random Fields has been proposed in (Levinkov et al., 2017) with applications in instance-separating semantic segmentation (Kirillov et al., 2017). A polyhedral study of lifted multicut was presented in (Horňáková et al., 2017).

Yet, for the above problems, global optimization has only been reported for small instances.

Disjoint paths for MOT. The data association step of MOT has been approached using the disjoint path setup (Berclaz et al., 2011; Zhang et al., 2008), since disjoint paths through a graph naturally model trajectories of multiple objects. Extension of the plain disjoint paths problem that disallow certain pairs of detections to occur simultaneously have been used to fuse different object detectors (Chari et al., 2015) and for multi-camera MOT (Hofmann et al., 2013; Leal-Taixé et al., 2012). The drawback of these approaches is that they cannot integrate long range information, in contrast to our proposed formulation.

Other combinatorial approaches to MOT. The minimum cost arborescence problem, an extension of minimum spanning tree to directed graphs, has been used for MOT in (Henschel et al., 2014). In (Keuper et al., 2016; 2018; Kumar et al., 2014; Ristani & Tomasi, 2014; Tang et al., 2015; 2016) the multicut problem has been used for MOT and in (Babaee et al., 2018; Tang et al., 2017) additionally lifted edges have been used to better model long range temporal interactions. The maximum clique problem, which corresponds to multicut with complete graphs has been applied for MOT in (Zamir et al., 2012; Dehghan et al., 2015).

Maximum independent set, which corresponds to maximum clique on the complement graph, has been used for MOT in (Brendel et al., 2011). The multigraph-matching problem, a generalization of the graph matching problem, has been applied to MOT in (Hu et al., 2019). Consistency of individual matched detections is ensured by cycle-consistency constraints coming from the multi-graph matching. The works (Henschel et al., 2018; 2016) reformulate tracking multiple objects with long temporal interactions as a binary quadratic program. If the problem size is small, the optimization problem can be solved optimally by reformulating it to an equivalent binary linear program (Henschel et al., 2019a; von Marcard et al., 2018). For large instances, an approximation is necessary. To this end, a specialized non-convex Frank-Wolfe method can be used (Henschel et al., 2018). Common to the above state of the art trackers is that they either employ heuristic solvers or are limited in the integration of long range information, in contrast to our work.

Contribution w.r.t. existing combinatorial approaches. It is widely acknowledged that one crucial ingredient for obtaining high-quality MOT results is to incorporate long range temporal information to re-identify detections and prevent id-switches. However, from a theoretical perspective, we believe that long range information has not yet been incorporated satisfactorily in optimization formulations for the data association step in MOT.

In comparison to lifted multicut for MOT, we argue that from the modelling point of view, network flow has advantages. In multicut, clusters can be arbitrary, while in MOT, tracks are clusters that may not contain multiple detection hypotheses of distinct objects at the same time point. This exclusion constraint must be enforced in multicut explicitly via soft constraints, while the disjoint paths substructure automatically takes care of it. On the other hand, the lifted multicut approach (Tang et al., 2017) has used the possibility to cluster multiple detections in one time frame. This directly incorporates non-maxima suppression in the optimization, which however increases computational complexity.

From a mathematical perspective, naively using polyhedral results from multicut is also not satisfactory. Specifically, one could naively obtain a polyhedral relaxation for the lifted disjoint paths problem by reusing the known polyhedral structure of lifted multicut (Horňáková et al., 2017) and additionally adding network flow constraints for the disjoint paths substructure. However, this would give a suboptimal polyhedral relaxation. We show in Section 4 that the underlying structure of the disjoint paths problem can be used to derive new and tighter constraints for lifted edges. This enables us to use a global optimization approach for MOT. To our knowledge, our work is the first one to combine global optimization with long range interactions for MOT.

In comparison to works that propose non-convex algorithms or other heuristics for incorporating long range temporal edges (Henschel et al., 2018; Hu et al., 2019; Zamir et al., 2012; Dehghan et al., 2015) our approach yields a more principled approach and globally optimal optimization solutions via LP-based branch and bound algorithms.

3. Problem Formulation

Below we recapitulate the disjoint paths problem and extend it by defining lifted edges. We discuss how the lifted disjoint paths problem can naturally model MOT. Proofs for statements in all subsequent sections can be found in the Appendix, Section 9.

Flow network and lifted graph. Consider two directed acyclic graphs $G = (V, E)$ and $G' = (V', E')$ where $V' = V \backslash \{s, t\}$. The graph $G = (V, E)$ represents the *flow network* and we denote by G' the lifted graph. The two special nodes s and t of G denote source and sink node respectively. We further assume that every node in V is reachable from s, and t can be reached from it.

We define the set of paths starting at v and ending in w as

$$vw\text{-paths}(G) = \left\{ (v_1 v_2, \dots, v_{l-1} v_l) : \begin{array}{c} v_i v_{i+1} \in E, \\ v_1 = v, v_l = w \end{array} \right\} . \tag{1}$$

For a vw-path P we denote its edge set as P_E and its node set as P_V.

The flow variables in G are denoted by $y \in \{0,1\}^E$ for edges and $x \in \{0,1\}^V$ for nodes. Allowing only 0/1 values of vertex variables reflects the requirement of vertex disjoint paths. Variables on the lifted edges E' are denoted by $y' \in \{0,1\}^{E'}$. Here, $y'_{vw} = 1$ means that nodes v and w are connected via the flow y in G. Formally,

$$y'_{vw} = 1 \Leftrightarrow \exists P \in vw\text{-paths}(G) \text{ s.t. } \forall ij \in P_E : y_{ij} = 1 \, . \tag{2}$$

Optimization problem. Given edge costs $c \in \mathbb{R}^E$, node cost $\omega \in \mathbb{R}^V$ in flow network G and edge cost $c' \in \mathbb{R}^{E'}$ for the lifted graph G' we define the lifted disjoint paths problem as

$$\begin{aligned} \min_{\substack{y \in \{0,1\}^E, y' \in \{0,1\}^{E'}, \\ x \in \{0,1\}^V}} \quad & \langle c, y \rangle + \langle c', y' \rangle + \langle \omega, x \rangle \\ \text{s.t.} \quad & y \text{ node-disjoint } s,t\text{-flow in } G, \\ & x \text{ flow through nodes of } G \\ & y, y' \text{ feasible according to (2)} \end{aligned} \tag{3}$$

In Section 4, we present an ILP formulation of (3) by proposing several linear inequalities that lead to a high-quality linear relaxation.

Graph construction for multiple object tracking. We argue that the lifted disjoint paths problem is an appropriate way of modelling the data association problem for MOT. In MOT, an unknown number of objects needs to be tracked across a video sequence. This problem can be naturally formalized by a graph $G = (V, E)$ where its node set V represents either object detections or tracklets of objects. If V represents object detections, we can express it as follows: $V = s \cup V_1 \cup \ldots \cup V_T \cup t$, where T is the number of frames and V_i denotes the object detections in time i. We introduce edges between adjacent time frames. An active flow on such an edge denotes correspondences of the same object. We also introduce skip edges between time frames that are farther apart. An active flow on a skip edge also denotes correspondences between the same object that, in contrast, may have been been occluded or not detected in intermediate time frames. This classical network flow formulation has been commonly used for MOT (Zhang et al., 2008).

On top of the underlying flow formulation for MOT, we usually want to express that two detections belong to the same object connected by a possibly longer track with multiple detections in between. For that purpose, lifted edges with negative costs can be used. We say in such a case that an active lifted edge re-identifies two detections (Tang et al., 2017). If two detections with larger temporal distance should not be part of the same track, a positive valued lifted edge can be used. In this case the lifted edge is used to prevent id-switches.

4. Constraints

Below, we will first introduce constraints that give an integer linear program (ILP) of the lifted disjoint paths problem (3). The corresponding linear programming (LP) relaxation can be strenghtened by additional constraints that we present subsequently.

Many constraints considered below will rely on whether a node w is reachable from another node v in the flow network. We define to this end the *reachability relation* $\mathcal{R} \subset V^2$ via

$$vw \in \mathcal{R} \Leftrightarrow vw\text{-paths}(G) \neq \emptyset\,. \tag{4}$$

In the special case of $v = w$, we also allow empty paths, which means $\forall v \in V : vv \in \mathcal{R}$. This makes relation $\mathcal{R}$ reflexive.

Flow conservation constraints. The flow variables y obey, as in classical network flow problems (Ahuja et al., 1988), the flow conservation constraints

$$\forall v \in V \setminus \{s, t\} : \sum_{u:uv \in E} y_{uv} = \sum_{w:vw \in E} y_{vw} = x_v\,. \tag{5}$$

Constraining lifted edges. All the following constraints restrict values of lifted edge variables y'_{vw} in order to ensure that they satisfy (2). Despite their sometimes complex form, they always obey the two basic principles:

- If there is flow in G going from vertex v to vertex w, then $y'_{vw} = 1$. The constraints of this form are (8), (10).
- If there is a vw-cut in G with all edges labeled by zero (i.e. no flow passes through this cut), then $y'_{vw} = 0$. We will mainly look at cuts that are induced by paths, i.e. edges that separate a path from the rest of the graph. The paths of interest will either originate at v or end at w. The constraints of this form are (6), (7), (9), (11), (12).

Single node cut inequalities. Given a lifted edge $vw \in E'$, if there is no flow going from vertex v which can potentially go to vertex w, then $y'_{vw} = 0$. Formally,

$$y'_{vw} \leq \sum_{\substack{u:\ vu \in E,\\ uw \in \mathcal{R}}} y_{vu}\,. \tag{6}$$

Similarly, if there is no flow going to w that can originate from vertex v, then $y'_{vw} = 0$. Formally,

$$y'_{vw} \leq \sum_{\substack{u:uw \in E,\\ vu \in \mathcal{R}}} y_{uw}\,. \tag{7}$$

The number of constraints of the above type (5) is linear in the number of vertices, while (6) and (7) are linear in the number of lifted edges. Hence we add them into our initial constraint set during optimization.

Path inequalities. For lifted edge y'_{vw} it holds that if there is a flow in G going from v to w along a path P, then $y'_{vw} = 1$. This constraint can be expressed by the following set of inequalities:

$$\forall vw \in E'\ \forall P \in vw\text{-paths}(G) :$$
$$y'_{vw} \geq \sum_{vj:j \in P_V} y_{vj} - \sum_{i \in P_V \setminus \{v,w\}} \sum_{k \notin P_V} y_{ik} \tag{8}$$

Here the first sum expresses the flow going from v to any vertex of path P. The second sum is the flow leaving path vertices P_V before reaching w. In other words, if flow does not leave P_V, edge y'_{vw} must be active. Note that inequality (8) implicitly enforces y'_{vw} to be active if any path vw-path $\tilde{P}$ with $\tilde{P}_V \subset P_V$ is active.

Remark 1. *For the multicut problem, there exist path inequalities that enforce path properties in an analogous way. While the multicut path inequalities would yield the same set of feasible integral points, the resulting polyhedral relaxation would be weaker, see Proposition 3 in the Appendix.*

Path-induced cut inequalities. The path-induced cut inequalities generalize the single node cut inequalities (6) and (7) by allowing cuts induced by paths.

Let a lifted edge $vw \in E'$, a node u from which w is reachable and a vu-path P be given. Consider the cut given by edges ik with $i \in P_V$ and $k \notin P_V$ but such that w is reachable from k. If the flow does not take any edge of this cut, then $y'_{vw} = 0$. Formally,

$$\forall vw \in E' \; \forall P \in vu\text{-paths}(G) \text{ s.t. } uw \in \mathcal{R} \wedge u \neq w : \quad y'_{vw} \leq \sum_{i \in P_V} \sum_{\substack{k \notin P_V, \\ kw \in \mathcal{R}}} y_{ik} \,. \tag{9}$$

Lifted inequalities. The path inequalities (8) and the path-induced cut inequalities (9) only consider base edges on their right hand sides. We can generalize both (8) and (9) by including lifted edges in the paths as well. Conceptually, using lifted edges allows to represent all possible paths between their endpoints, which enables to formulate tighter inequalities, see Propositions 1 and 2.

To that end consider the multigraph $G \cup G' := (V, E \cup E')$. For any edge $ij \in E \cap E'$ we always distinguish whether $ij \in E$ or $ij \in E'$. For $P \in vw\text{-paths}(G \cup G')$, we denote by P_E and $P_{E'}$ edges of the path P in E and E' respectively. We require $P_E \cap P_{E'} = \emptyset$.

Lifted path inequalities. We generalize the path inequalities (8). Now the vw-path P may contain both edges in E and E'. Whenever a lifted edge y'_{ij} in the third sum in (10) is one, two cases can occur: (i) Flow goes out of P (uses vertices not in P_V) but reenters it again later. Then a base edge variable y_{ik} will be one in the second sum in (10) and the values of y'_{ij} and y_{ik} cancel out. (ii) A base edge $ij \in E \cap E'$ parallel to the lifted edge is active. Then the variable y_{ij} in the fourth sum in (10) cancels out y'_{ij}. The lifted path inequality becomes

$$\begin{aligned} \forall vw \in E' \; \forall P \in vw\text{-paths}(G \cup G') : \\ y'_{vw} \geq \sum_{j \in P_V} y_{vj} - \sum_{i \in P_V \setminus \{v,w\}} \sum_{k \notin P_V} y_{ik} \\ + \sum_{ij \in P_{E'}} y'_{ij} - \sum_{ij \in P_{E'} \cap E} y_{ij} \,. \end{aligned} \tag{10}$$

Whenever the path in (10) consists only of base edges P_E, the resulting inequality becomes a path inequality (8).

Proposition 1. *The lifted path inequalities* (10) *provide a strictly better relaxation than the path inequalities* (8)*.*

Lifted path-induced cut inequalities. We generalize the path-induced cut inequalities (9). Let a lifted edge $vw \in E'$ and a vu-path P in $G \cup G'$ be given. In contrast to the basic version (9), a lifted edge $ij \in P_{E'}$ can be taken. This can occur in two cases: Either the flow leaves P_V via a base edge ik, $k \notin P_V$ or a base edge $ij \in E \cap E'$ parallel to the lifted edge is taken. Both cases are accounted for by terms in the first and the third sum in (11) below.

$$\forall vw \in E' \; \forall P \in vu\text{-paths}(G \cup G') \text{ s.t. } uw \in \mathcal{R} \wedge u \neq w : \quad y'_{vw} \leq \sum_{i \in P_V} \sum_{\substack{k \notin P_V, \\ kw \in \mathcal{R}}} y_{ik} - \sum_{ij \in P_{E'}} y'_{ij} + \sum_{ij \in P_{E'} \cap E} y_{ij} \tag{11}$$

Assume that the last node u of path P is connected via a lifted edge with w. Then we can strengthen (11) by replacing the sum of base edges outgoing from u by y'_{uw}.

$$\begin{aligned} \forall vw \in E' \; \forall P \in vu\text{-paths } (G \cup G') \text{ s.t. } uw \in E' : \\ y'_{vw} \leq \sum_{i \in P_V \setminus u} \sum_{\substack{k \notin P_V, \\ kw \in \mathcal{R}}} y_{ik} - \sum_{ij \in P_{E'}} y'_{ij} \\ + \sum_{ij \in P_{E'} \cap E} y_{ij} + y'_{uw} \end{aligned} \tag{12}$$

Proposition 2. *The lifted path-induced cut inequalities* (11) *define a strictly tighter relaxation than the path-induced cut inequalities* (9)*.*

Furthermore the lifted path-induced cut inequalities (11) *and* (12) *define a strictly better relaxation than* (11) *alone.*

Symmetric cut inequalities. Inequalities (7) provide a symmetric counterpart to inequalities (6). We can also formulate symmetric counterparts to inequalities (9), (11) and (12) by swapping the role of v and w. All constraints (9), (11) and (12) concentrate on paths originating in v. The symmetric inequalities are obtained by studying all paths ending in w. These symmetric inequalities are described in Appendix Section 10.2. Relations analogous to those described in Proposition 2 hold for the symmetric counterparts as well. The symmetric inequalities also strengthen the relaxation strictly. For the exact statements, see propositions in Appendix Section 10.2.

5. Separation

We solve the lifted disjoint paths problem (3) with the state of the art integer linear program solver Gurobi (Gurobi Optimization, 2019). Since there are exponentially many constraints of the form (8), (9), (10), (11) and (12), we do not add them initially. Instead, we start with constraints (5), (6) and (7) and find the optimal integer solution. In the separation procedures described below we check if any of the advanced constraints are violated and add those that are to the active constraint set. We resolve the tightened problem and iterate until we have found a feasible solution to the overall problem (3).

Algorithms 1 and 2 describe the separation procedures for adding lifted path constraints (10), and lifted path-induced cut constraints (11) and (12). Since path constraints (8) and

path-induced cut inequalities (9) are special cases of those above, they are also accounted for.

Separation for path inequalities. Algorithm 1 iterates over all active st-paths. For every path P^1, labels of all lifted edges connecting two vertices in P^1_V are inspected. If the lifted edge variable is zero, Algorithm 1 will extract a path in $G \cup G'$ connecting the endpoints and add the resulting lifted path inequality (10) to the active constraint set.

Algorithm 1 Separation for lifted path inequalities (10)

Define $E^1 = \{e \in E : y_e = 1\}$, $G^1 = (V, E^1)$
for all $P^1 \in st$-paths (G^1) **do**
  for all $y'_{vw} = 0 : v \in P^1_V \wedge w \in P^1_V$ **do**
    $P :=$ Extract_path(P^1, v, w)
    Add constr. (10) for y'_{vw} with P.
  end for
end for

Separation for path-induced cut inequalities. Algorithm 2 iterates over all active st-paths. For every path P^1, lifted edges that start in P^1_V but do not end in P^1_V are inspected. If their label is one, Algorithm 2 will extract a subpath of P^1 for either (12) or (11) and add the respective inequality to the active constraint set.

Algorithm 2 Separation for lifted path-induced cut inequalities (11) and (12)

Define $E^1 = \{e \in E : y_e = 1\}$, $G^1 = (V, E^1)$
for all $P^1 \in st$-paths (G^1) **do**
  for all $y'_{vw} = 1 : v \in P^1_V \wedge w \notin P^1_V$ **do**
    if $\exists u \in P^1_V : y'_{uw} = 0 \wedge vu \in \mathcal{R}$ **then**
      $P :=$ Extract_path(P^1, v, u)
      Add constr. (12) for y'_{vw} with P.
    else
      $u :=$ last vertex of P^1 such that $uw \in \mathcal{R}$
      $P :=$ Extract_path (P_1, v, u)
      Add constr. (11) for y'_{vw} with P.
    end if
  end for
end for

Complexity of separation. Both Algorithms 1 and 2 can be implemented efficiently such that they are linear in $|E^1|$ (i.e. in the number of active edges of graph G). In our implementation, we traverse all active st-paths from the end to the beginning and directly store correctly labelled lifted edges that originate on the already processed subpaths. These lifted edges can be used later as edges in $P_{E'}$ in (10)-(12) or as $y'_{uw} = 0$ in (12).

Algorithm 3 Extract_path(P^1, v, w)

$P' := vw$-subpath of P^1, $P := \emptyset$
for $j \in P'_V$ from end of path to beginning **do**
  if $\exists$ edge $ij \in E', i \in P'_V, y'_{ij} = 1$ **then**
    Add ij to $P_{E'}$, skip to node $i \in P'_V$
  else
    Add ij from P' to P_E
  end if
end for
output $P = P_E \cup P_{E'}$

6. Complexity

Below, we show that the lifted disjoint paths problem (3) is NP-hard. The following Theorems state that even its restricted versions using only negative or only positive lifted edges are NP-hard. The proofs use reductions from two known NP-complete problems. Theorem 1 is proven by reduction from integer multicommodity flow (Even et al., 1976) and Theorem 2 by reduction from 3-SAT (Cook, 1971).

Theorem 1. *Lifted disjoint paths problem* (3) *with negative lifted edges only is NP-hard.*

Theorem 2. *Lifted disjoint paths problem* (3) *with positive lifted edges only is NP-hard.*

7. Experiments

We conduct several experiments on MOT showing the merit of using lifted disjoint paths for the tracking problem. Below, we describe our problem construction, cost learning for base and lifted edges, preprocessing and post-processing steps and report resulting performance. More details about our experiments are provided in Appendix, Section 9.

7.1. Graph Construction.

Two-step procedure. Due to the computational complexity of the problem, we cannot solve entire video sequences straightforwardly. In order to make the problem tractable, we apply the following two-step procedure. In the first step, the solver is applied on graphs over person detections but only for small time intervals consisting of a few dozen video frames. The tracks resulting from the first step are used for extracting tracklets. In the second step, the solver is applied on newly created graphs G and G' where vertices correspond to the obtained tracklets. Edges and edge costs between tracklets are obtained by aggregating original edges resp. edge costs between person detections. The tracks resulting from the second step may be suboptimal with respect to the original objective function defined over person detections. Therefore, we identify points where splitting a track leads to an improvement of the original objective value and

extract new tracklets from the divided tracks. Multiple iterations of the second step are performed until no improving split points are found in the output tracks. This two-step procedure improves the objective w.r.t. the original objective (3) in every iteration. Since there are only finitely many trackings, the procedure terminates finitely. In practice, only a few iterations are necessary.

Graph sparsification. For our experiments, we use edges between detections up to 2sec temporal distance. These long range edges cause high computational complexity for the first step. In order to reduce it, we apply sparsification on both base and lifted graphs. For the base edges, we select for every $v \in V'$ its K nearest (lowest-cost) neighbors from every subsequent time frame within an allowed time gap. Lifted edges with costs close to zero are not included, since they are not discriminative. Lifted edges connecting detections with high time gap are included more sparsely than lifted edges having lower time gaps. We use dense graphs in the second step.

Costs. Initially, in the first step, we set $\omega_v = 0$ for all vertices $v \in V$. For the second step, where V represents tracklets, ω_v is set to the cost of outputting tracklet v as a final trajectory. Specifically, ω_v is the sum of costs of base edges between consecutive detections in the tracklet and the cost of lifted edges between all pairs of detections contained in the tracklet. The cost of a base edge between two tracklets is given by the cost of the original base edge connecting the last detection in the first tracklet with the first detection in the subsequent tracklet. The cost of a lifted edge between two tracklets is obtained by summing up the costs of original lifted edges between detections contained in the tracklets. This ensures that the costs of the tracklet solution corresponds to the costs of the original problem. We set cost of all edges from the source node s and to the sink node t to zero. Setting of detection costs and in/out costs to zero reduces the number of hyperparameters that usually needs to be incorporated by other methods. Moreover, our method does not include temporal decay of edge costs since the formulation directly prefers short range base edges over the long range ones.

7.2. Preprocessing and Post-processing

As is common for tracking by detection, we perform pre- and postprecessing to compensate for detector inaccuracies.

Input filtering. Given a set of input detections derived from a detector, we follow the approach of (Bergmann et al., 2019), a leading tracker for the MOT challenge, to reject false positive detections and to correct misaligned ones. For this, each input detection is send through the regression and classification part of their detector. In more detail, all tracking parts involved in the tracker Tracktor (Bergmann et al., 2019) are deactivated, such that it only reshapes and eventually rejects input detections, without assigning labels to them. Input detections are rejected if Tracktor's detector outputs a confidence score $\sigma_{\text{active}} \leq 0.5$.

Tracktor also applies a non-maxima-surpression on the reshaped input detections, where we use the threshold $\lambda_{\text{new}} = 0.6$.

Inter- and extrapolation. Even if all input detections have been assigned to the correct identities by our solver, there might still be missing detections in case that a person has not been detected in some frames. We recover missing detections within the time range of a trajectory, which we denote as interpolation. Further, we extend a trajectory in forward and backward directions, which we denote as extrapolation. To this end, we follow (Bergmann et al., 2019) and apply their object detector to recover missing positions based on the visual information at the last known position. Finally, for sequences filmed from a static camera, we perform linear interpolation on the remaining gaps. These sequences can be automatically detected using DeepMatching on the regions outside detection boxes.

To demonstrate the performance using traditional post-processing, we also evaluate our tracker using only linear interpolation as post-processing in all sequences.

7.3. Cost Learning

Costs for base edges E and lifted edges E' are computed equally, since they both indicate whether two detections are from the same object or not. For an edge $e = vw$, we denote with $d_{\text{wi}}(v)$ the detection width corresponding to node v.

Visual cues. We exploit two different appearance features: Given two detections, the *re-identification* descriptor utilizes global appearance statistics, while the *deep-matching* descriptor relies on fine-grained pixel-wise correspondences.

We employ the state-of-the-art re-identification network (Zheng et al., 2019) and train it on MOT17 train set (Milan et al., 2016) together with additional re-identification datasets (Zheng et al., 2015; Wei et al., 2018; Ristani et al., 2016). The obtained feature value $\mathfrak{f}_{\text{re-id}}(e) \in [-1, 1]$ is modified in order to better reflect the uncertainty of a connection. We truncate values smaller 0 (corresponding to improbable connections) and re-scale the rest. First, we normalize scores between each detection v and all detections in every time frame V_j through the score of the most probable connecting edge vw. Second, all other connections than vw are downscaled.

Our second visual cue utilizes DeepMatching (DM) (Weinzaepfel et al., 2013) to establishes pixelwise correspondences between two images. It thus serves as a reliable tracking feature (Tang et al., 2016; Henschel et al., 2018;

2019b).

We apply DM between boxes in two images and compute the DM intersection over union (Tang et al., 2016; Henschel et al., 2018) w.r.t. the whole detection boxes and on five subboxes (left/right, upper/middle/lower part). In addition, we measure for all points in a given subbox whether their matched endpoints are in the corresponding subbox again or not. This gives two additional error measures for deviation in x and y-directions. Thus, in total we obtain a feature vector $\mathfrak{f}_{DM}(e) \in [0,1]^8$. In order to assess the reliability of DM features, density of matching points is computed in each box and its subboxes. The smaller value is chosen for each box pair. This results in feature $\rho \in [0,1]^6$.

Motion constraints. We penalize for improbable motions by comparing the maximal displacement of DM endpoints within the sequence with the displacements of detection boxes. Assignment hypotheses of pairs of boxes representing improbable motions are penalized with a large cost.

Spatio-temporal cues. Our spatio-temporal cues utilize a simple motion compensation by computing the median DM displacement between correspondences of the background.

We assume a linear motion model, similar to (Ristani & Tomasi, 2018) and penalize deviations of detections from the estimated motion trajectory. This enforces spatio-temporally consistency of detections within one trajectory. Furthermore, we penalize improbable large person movements by relating velocities (in pixels per seconds) in horizontal direction to box width: $\mathfrak{f}_{\text{trans}}(e) = \log(\mathfrak{v}_x(e) / \min\{d_{\text{wi}}(v), d_{\text{wi}}(w)\})$.

Fusion of input features. We construct a neural network consisting of fully connected layers, batch normalization and relu units taking the above described features and time differences as input and outputting scores for assignment hypotheses. The final layer uses a sigmoid activation function for producing a score in $[0,1]$. We refer to the supplemental material for the exact structure of the neural network and details about the training procedure.

7.4. Experiment Setup

In order to assess the suitability of the proposed lifted disjoint paths formulation for MOT, we conduct extensive experiments on three challenging benchmarks: MOT15 (Leal-Taixé et al., 2015), MOT16 and MOT17 (Milan et al., 2016), resulting in 39 test sequences. The sequences are filmed from static and moving cameras. While MOT16 and MOT17 share the same sequences, MOT17 provides three different detectors in order to study the dependence of the tracking quality on the input detections. We perform analysis and parameter tuning for our tracker on the MOT17 train set, even when our tracker is applied to the MOT15 sequences to ensure that our tracker is not prone to overfitting. We follow the MOT challenge protocol and use the detections provided by the respective benchmarks. All experiments on the training set are evaluated using a leave-one-out cross-validation. This includes all of our training procedures, in particular also the training of the re-identification network.

To measure the tracking quality, the multiple object tracking accuracy (MOTA) (Bernardin & Stiefelhagen, 2008) and the IDF1 metric (Ristani et al., 2016) are regarded as the most meaningful ones. The first incorporates the number of false negatives (FN), false positives (FP) and identity switches (IDS), thereby focusing on the coverage of persons. The latter assesses the consistency w.r.t. identities. Further tracking metrics (MT, ML) are defined in (Li et al., 2009).

	0.3s	0.5s	1s	1.5s	2s	∞
MOTA (ours)↑	52.6	52.7	52.8	52.8	**52.8**	-
MOTA (optimal)↑	53.0	53.1	53.3	53.3	**53.4**	**53.4**
IDF1 (ours) ↑	55.7	57.8	61.8	63.8	**64.3**	-
IDF1 (optimal)↑	56.0	58.6	63.2	65.7	66.8	**69.9**
IDP (ours) ↑	79.8	82.9	88.5	91.4	**92.1**	-
IDP (optimal)↑	80.4	84.2	90.8	94.3	95.9	**100.0**
IDR (ours) ↑	42.7	44.5	47.4	49.0	**49.4**	-
IDR (optimal)↑	42.9	45.0	48.5	50.4	51.3	**53.4**

Table 1. Assignment quality of our solver without interpolation or extrapolation on the MOT17 train set with different maximal time gaps in seconds. Rows 1,3,5 and 7 show the results by our solver, rows 2,4,6 and 8 show the maximally achievable bounds with admissible assignment hypotheses up to the specified time gap. Bold numbers represent the best values per row.

7.5. Benefit of Long Range Edges

We investigate the importance of using long range information for MOT. To this end, we apply our proposed tracker on the MOT17 training sequence with varying maximal time gap, for which base and lifted edges are created between nodes. In order to assess the influence of the time gap on the tracking quality, we measure the *assignment* quality in terms of the MOTA and IDF1 metrics, without performing any inter- or extrapolation. To assess how well the *assignment part* is solved by our tracker, we compute the maximum achievable metrics given the filtered input detections and admissible assignment hypotheses within maximal time gaps. A detailed description of how we obtain the optimal assignments are given in the appendix in Section 10.6. From the result in Table 1, we see essentialy constant MOTA scores. This is due the fact that selecting correct connections does not change MOTA significantly except after inter- and extrapolation (which we have excluded in Table 1). However, we see a significant improvement in the IDF1 score, which directly penalizes wrong connections. Here, long range edges help greatly. Moreover, both metrics, ID precision and ID recall, clearly increase with increasing

	Method	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Frag↓
MOT17	Lif_T (ours)	**60.5**	**65.6**	27.0	33.6	14966	**206619**	1189	3476
	Lif_TsimInt (ours)	58.2	65.2	**28.6**	33.6	16850	217944	**1022**	**2062**
	Tracktor17	53.5	52.3	19.5	36.6	**12201**	248047	2072	4611
	JBNOT	52.6	50.8	19.7	35.8	31572	232659	3050	3792
	FAMNet	52.0	48.7	19.1	**33.4**	14138	253616	3072	5318
	eTC17	51.9	58.1	23.1	35.5	36164	232783	2288	3071
	eHAF17	51.8	54.7	23.4	37.9	33212	236772	1834	2739
MOT16	Lif_T (ours)	**61.3**	**64.7**	**27.0**	34.0	4844	**65401**	389	1034
	Lif_TsimInt (ours)	57.5	64.1	25.4	**34.7**	4249	72868	**335**	604
	Tracktor16	54.4	52.5	19.0	36.9	**3280**	79149	682	1480
	NOTA	49.8	55.3	17.9	37.7	7248	83614	614	1372
	HCC	49.3	50.7	17.8	39.9	5333	86795	391	**535**
	eTC	49.2	56.1	17.3	40.3	8400	83702	606	882
	KCF16	48.8	47.2	15.8	38.1	5875	86567	906	1116
2D MOT15	Lif_T (ours)	**52.5**	**60.0**	**33.8**	**25.8**	6837	**21610**	730	1047
	Lif_TsimInt (ours)	47.2	57.6	27.0	29.8	7635	24277	**554**	**803**
	Tracktor15	44.1	46.7	18.0	26.2	6477	26577	1318	1790
	KCF	38.9	44.5	16.6	31.5	7321	29501	720	1440
	AP_HWDPL_p	38.5	47.1	8.7	37.4	**4005**	33203	586	1263
	STRN	38.1	46.6	11.5	33.4	5451	31571	1033	2665
	AMIR15	37.6	46.0	15.8	26.8	7933	29397	1026	2024

Table 2. We compare our tracker Lif_T with the five best performing competing solvers w.r.t. MOTA from the MOT challenge. Tracktor (Bergmann et al., 2019), JBNOT (Henschel et al., 2019b), FAMNet (Chu & Ling, 2019), eTC (Wang et al., 2019b), eHAF (Sheng et al., 2018), NOTA (Chen et al., 2019), HCC (Ma et al., 2018), KCF (Chu et al., 2019), AP_HWDPL_p (Chen et al., 2017), STRN (Xu et al., 2019) and AMIR15 (Sadeghian et al., 2017). In addition, we compare the results to our tracker Lif_TsimInt that uses only a simple interpolation method (linear interpolation) as post-processing in all sequences. We outperform competing solvers on most metrics on all three MOT Challenge benchmarks, using Lif_T and Lif_TsimInt. Arrows indicate whether low or high metric values are better.

time gap. This shows that improvements by incorporating more temporal information come from using longer skip edges (impact on IDR) but most importantly, precision increases greatly. This means that ID switches are avoided thanks to lifted edges. Furthermore, the experiment shows that our designed features together with the lifted disjoint paths formulation (3) are well-suited for the MOT problem delivering nearly optimal assignments.

7.6. Benchmark Evaluations

Finally, we compare our tracking performance on the MOT15, MOT16 and MOT17 benchmarks with all trackers listed on the MOTChallenge which have been peer-reviewed and correspond to published work. The three benchmark datasets consist of 11/7/7 training and test sequences for MOT15/16/17 respectively. They are the standard benchmark datasets for MOT. The results in Table 2 show the tracking performance of our tracker together with the best 5 performing trackers, accumulated over all sequences of the respective benchmarks. The evaluations show that we outperform all tracking systems by a large margin on all considered benchmarks. On MOT17, we improve the MOTA score from 53.5 to 60.5 and the IDF1 score from 52.3 to 65.6, which corresponds to an improvement of 13% in terms of MOTA and almost 25% in terms of the IDF1 score, indicating the effectiveness of the lifted edges. We observe similar improvements across all three benchmarks. These results reflect the near-optimal assignment performance observed on the MOT17 train set in Sect. 7.5. Finally, using only simple linear interpolation as post-processing (Lif_TsimInt), our tracker achieves 58.2 MOTA and 65.2 IDF1. Even then, our system clearly outperforms existing tracking systems. On average, the ILP solver needs 26.6 min. per sequence. Detailed runtimes are available in Table 5 in Appendix.

8. Conclusion

We have shown that for the MOT challenge datasets we reach nearly optimal data association performance. We conjecture that further improvements would have to come from better detectors, better inter- and extrapolation and more powerful solvers for our formulation to take into account even longer time-gaps. Our polyhedral work offers the basis for writing such more powerful solvers.

9. Acknowledgement

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy within the Cluster of Excellence PhoenixD (EXC 2122). We thank Laura Leal-Taixé for initiating the collaboration. We thank all reviewers for their valuable comments.

References

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. *Network flows*. Cambridge, Mass.: Alfred P. Sloan School of Management, Massachusetts, 1988.

Babaee, M., Athar, A., and Rigoll, G. Multiple people tracking using hierarchical deep tracklet re-identification. *arXiv preprint arXiv:1811.04091*, 2018.

Beier, T., Pape, C., Rahaman, N., Prange, T., Berg, S., Bock, D. D., Cardona, A., Knott, G. W., Plaza, S. M., Scheffer, L. K., et al. Multicut brings automated neurite segmentation closer to human performance. *Nature Methods*, 14 (2):101, 2017.

Berclaz, J., Fleuret, F., Turetken, E., and Fua, P. Multiple object tracking using k-shortest paths optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819, 2011.

Bergmann, P., Meinhardt, T., and Leal-Taixé, L. Tracking without bells and whistles. In *IEEE International Conference on Computer Vision*, pp. 941–951, 2019.

Bernardin, K. and Stiefelhagen, R. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008: 1–10, 2008.

Brendel, W., Amer, M., and Todorovic, S. Multiobject tracking as maximum weight independent set. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1273–1280. IEEE, 2011.

Chari, V., Lacoste-Julien, S., Laptev, I., and Sivic, J. On pairwise costs for network flow multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5537–5545, 2015.

Chen, L., Ai, H., Shang, C., Zhuang, Z., and Bai, B. Online multi-object tracking with convolutional neural networks. In *IEEE International Conference on Image Processing*, pp. 645–649. IEEE, 2017.

Chen, L., Ai, H., Chen, R., and Zhuang, Z. Aggregate tracklet appearance features for multi-object tracking. *IEEE Signal Processing Letters*, 26(11):1613–1617, 2019.

Chu, P. and Ling, H. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *IEEE International Conference on Computer Vision*, pp. 6172–6181, 2019.

Chu, P., Fan, H., Tan, C. C., and Ling, H. Online multi-object tracking with instance-aware tracker and dynamic model refreshment. In *IEEE Winter Conference on Applications of Computer Vision*, pp. 161–170. IEEE, 2019.

Cook, S. A. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pp. 151–158. ACM, 1971.

Dehghan, A., Modiri Assari, S., and Shah, M. GMMCP tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4091–4099, 2015.

Eilam-Tzoreff, T. The disjoint shortest paths problem. *Discrete Applied Mathematics*, 85(2):113–138, 1998.

Even, S., Itai, A., and Shamir, A. On the complexity of timetable and multicommodity flow problems. *SIAM J. Comput.*, 5:691–703, 12 1976. doi: 10.1137/0205048.

Gurobi Optimization, L. Gurobi optimizer reference manual, 2019. URL http://www.gurobi.com.

Henschel, R., Leal-Taixé, L., and Rosenhahn, B. Efficient multiple people tracking using minimum cost arborescences. In *German Conference on Pattern Recognition*, pp. 265–276. Springer, 2014.

Henschel, R., Leal-Taixé, L., Rosenhahn, B., and Schindler, K. Tracking with multi-level features. *arXiv preprint arXiv:1607.07304*, 2016.

Henschel, R., Leal-Taixé, L., Cremers, D., and Rosenhahn, B. Fusion of head and full-body detectors for multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2018.

Henschel, R., von Marcard, T., and Rosenhahn, B. Simultaneous identification and tracking of multiple people using video and imus. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019a.

Henschel, R., Zou, Y., and Rosenhahn, B. Multiple people tracking using body and joint detections. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019b.

Hofmann, M., Wolf, D., and Rigoll, G. Hypergraphs for joint multi-view reconstruction and multi-object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3650–3657, 2013.

Horňáková, A., Lange, J.-H., and Andres, B. Analysis and optimization of graph decompositions by lifted multicuts. In *International Conference on Machine Learning*, 2017.

Hu, W., Shi, X., Zhou, Z., Xing, J., Ling, H., and Maybank, S. Dual L1-normalized context aware tensor power iteration and its applications to multi-object tracking and multi-graph matching. *International Journal*

of Computer Vision, Oct 2019. ISSN 1573-1405. doi: 10.1007/s11263-019-01231-y. URL https://doi.org/10.1007/s11263-019-01231-y.

Keuper, M., Levinkov, E., Bonneel, N., Lavoué, G., Brox, T., and Andres, B. Efficient decomposition of image and mesh graphs by lifted multicuts. In *IEEE International Conference on Computer Vision*, 2015. doi: 10.1109/ICCV.2015.204.

Keuper, M., Tang, S., Zhongjie, Y., Andres, B., Brox, T., and Schiele, B. A multi-cut formulation for joint segmentation and tracking of multiple objects. *arXiv preprint arXiv:1607.06317*, 2016.

Keuper, M., Tang, S., Andres, B., Brox, T., and Schiele, B. Motion segmentation & multiple object tracking by correlation co-clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(1):140–153, 2018.

Kirillov, A., Levinkov, E., Andres, B., Savchynskyy, B., and Rother, C. Instancecut: from edges to instances with multicut. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5008–5017, 2017.

Kovács, P. Minimum-cost flow algorithms: an experimental evaluation. *Optimization Methods and Software*, 30(1): 94–127, 2015.

Kumar, R., Charpiat, G., and Thonnat, M. Multiple object tracking by efficient graph partitioning. In *Asian Conference on Computer Vision*, pp. 445–460. Springer, 2014.

Leal-Taixé, L., Pons-Moll, G., and Rosenhahn, B. Branch-and-price global optimization for multi-view multi-target tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1987–1994. IEEE, 2012.

Leal-Taixé, L., Milan, A., Reid, I., Roth, S., and Schindler, K. MOTChallenge 2015: Towards a benchmark for multi-target tracking. *arXiv:1504.01942 [cs]*, April 2015. URL http://arxiv.org/abs/1504.01942. arXiv: 1504.01942.

Levinkov, E., Uhrig, J., Tang, S., Omran, M., Insafutdinov, E., Kirillov, A., Rother, C., Brox, T., Schiele, B., and Andres, B. Joint graph decomposition & node labeling: Problem, algorithms, applications. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6012–6020, 2017.

Li, Y., Huang, C., and Nevatia, R. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2953–2960. IEEE, 2009.

Ma, L., Tang, S., Black, M. J., and Van Gool, L. Customized multi-person tracker. In *Asian Conference on Computer Vision*, pp. 612–628. Springer, 2018.

Milan, A., Leal-Taixé, L., Reid, I., Roth, S., and Schindler, K. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]*, March 2016. URL http://arxiv.org/abs/1603.00831. arXiv: 1603.00831.

Nowozin, S. and Lampert, C. H. Global interactions in random field models: A potential function ensuring connectedness. *SIAM Journal on Imaging Sciences*, 3(4): 1048–1074, 2010. doi: 10.1137/090752614.

Rempfler, M., Lange, J.-H., Jug, F., Blasse, C., Myers, E. W., Menze, B. H., and Andres, B. Efficient algorithms for moral lineage tracing. In *IEEE International Conference on Computer Vision*, pp. 4695–4704, 2017.

Ristani, E. and Tomasi, C. Tracking multiple people online and in real time. In *Asian Conference on Computer Vision*, pp. 444–459. Springer, 2014.

Ristani, E. and Tomasi, C. Features for multi-target multi-camera tracking and re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6036–6046, 2018.

Ristani, E., Solera, F., Zou, R. S., Cucchiara, R., and Tomasi, C. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision Workshop on Benchmarking Multi-Target Tracking*, 2016.

Sadeghian, A., Alahi, A., and Savarese, S. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *IEEE International Conference on Computer Vision*, pp. 300–311, 2017.

Sheng, H., Zhang, Y., Chen, J., Xiong, Z., and Zhang, J. Heterogeneous association graph fusion for target association in multiple object tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(11): 3269–3280, 2018.

Suurballe, J. Disjoint paths in a network. *Networks*, 4(2): 125–145, 1974.

Tang, S., Andres, B., Andriluka, M., and Schiele, B. Subgraph decomposition for multi-target tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5033–5041, 2015.

Tang, S., Andres, B., Andriluka, M., and Schiele, B. Multi-person tracking by multicut and deep matching. In *European Conference on Computer Vision*, pp. 100–111. Springer, 2016.

Tang, S., Andriluka, M., Andres, B., and Schiele, B. Multiple people tracking by lifted multicut and person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Tholey, T. Linear time algorithms for two disjoint paths problems on directed acyclic graphs. *Theoretical Computer Science*, 465:35–48, 2012.

von Marcard, T., Henschel, R., Black, M. J., Rosenhahn, B., and Pons-Moll, G. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 601–617, 2018.

Wang, C., Wang, Y., Wang, Y., Wu, C.-T., and Yu, G. mussp: Efficient min-cost flow algorithm for multi-object tracking. In *Advances in Neural Information Processing Systems*, pp. 423–432, 2019a.

Wang, G., Wang, Y., Zhang, H., Gu, R., and Hwang, J.-N. Exploit the connectivity: Multi-object tracking with trackletnet. In *ACM International Conference on Multimedia*, pp. 482–490, 2019b.

Wei, L., Zhang, S., Gao, W., and Tian, Q. Person transfer gan to bridge domain gap for person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 79–88, 2018.

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. Deepflow: Large displacement optical flow with deep matching. In *IEEE Intenational Conference on Computer Vision*, Sydney, Australia, December 2013. URL http://hal.inria.fr/hal-00873592.

Xu, J., Cao, Y., Zhang, Z., and Hu, H. Spatial-temporal relation networks for multi-object tracking. In *IEEE International Conference on Computer Vision*, pp. 3988–3998, 2019.

Zamir, A. R., Dehghan, A., and Shah, M. GMCP-tracker: Global multi-object tracking using generalized minimum clique graphs. In *European Conference on Computer Vision*, pp. 343–356. Springer, 2012.

Zhang, L., Li, Y., and Nevatia, R. Global data association for multi-object tracking using network flows. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE, 2008.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., and Tian, Q. Scalable person re-identification: A benchmark. In *IEEE International Conference on Computer Vision*, pp. 1116–1124, 2015.

Zheng, Z., Yang, X., Yu, Z., Zheng, L., Yang, Y., and Kautz, J. Joint discriminative and generative learning for person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.