Tracking Objects with Higher Order Interactions via Delayed Column Generation

Shaofei Wang
Beijing A&E Technologies

Steffen Wolf
University of Heidelberg

Charless C. Fowlkes
University of California, Irvine

Julian Yarkony
Experian Data Lab

Abstract

We study the problem of multi-target tracking and data association in video. We formulate this in terms of selecting a subset of high-quality tracks subject to the constraint that no pair of selected tracks is associated with a common detection (of an object). This objective is equivalent to the classic NP-hard problem of finding a maximum-weight set packing (MWSP) where tracks correspond to sets and is made further difficult since the number of candidate tracks grows exponentially in the number of detections. We present a relaxation of this combinatorial problem that uses a column generation formulation where the pricing problem is solved via dynamic programming to efficiently explore the space of tracks. We employ row generation to tighten the bound in such a way as to preserve efficient inference in the pricing problem. We show the practical utility of this algorithm for pedestrian and particle tracking.

1 Introduction

Multi-target tracking in video is often formulated from the perspective of grouping disjoint sets of candidate detections into "tracks" whose underlying trajectories can be estimated using traditional single-target methods such as Kalman filtering. There is a well developed literature on methods for exploring this combinatorial space of possible data associations in order to find collections of low-cost, disjoint tracks.

We first highlight three common approaches that are closely related to our method. The first approach is based on reduction to minimum-cost network flow [19] which maps tracks to unit flows in a network whose edge costs encode track quality. This elegant construction utilizes edge capacity constraints to enforce disjoint tracks and allows for exact, polynomial-time inference. However, this formulation is quite limited in integrating joint statistics over multiple detections assigned to a track. In particular, it is constrained to first-order dynamics in which the cost of a detection being associated with a given track depends only on the immediately neighboring detections.

A second approach is the Multiple Hypothesis Tracking [6, 14, 12](MHT) which attempts to model higher order dynamics by grouping short sequences of detections into a set of hypothesis tracks that can be evaluated and pruned in an online manner. This trades efficiency and global exactness of min-cost flow trackers for additional modeling power. For example, the cost of a track may be computed using, *e.g.*spline-based fitting or Bayesian estimation/Kalman filters and instance specific appearance models. However, such methods face a combinatorial problem of assembling compatible sets of tracklets which is usually tackled using greedy approximations.

Our method is most closely related to the third approach: the Lagrangian relaxation method of [7], which attempts to preserve the speed and guarantees of min-cost flow tracking while also capturing higher order dynamics of the objects. A large number of short sequences of detections (subtracks) are generated, each of which is associated with a cost. The set of subtracks form the basis from which tracks are constructed. The corresponding optimization is attacked via sub-gradient optimization of the Lagrangian relaxation corresponding to the constrained objective.

Inspired by [16], we attack the problem of reasoning over subtrack assembly using column/row generation [9, 4, 18] to allow higher-order interactions within a track while achieving faster inference with tighter bounds than [7]. This paper is organized as follows. In Section 2 we formulate tracking as optimization of an Integer Linear Program (ILP). In Section 2.2 we relax integrality constraints and demonstrate a simple case

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

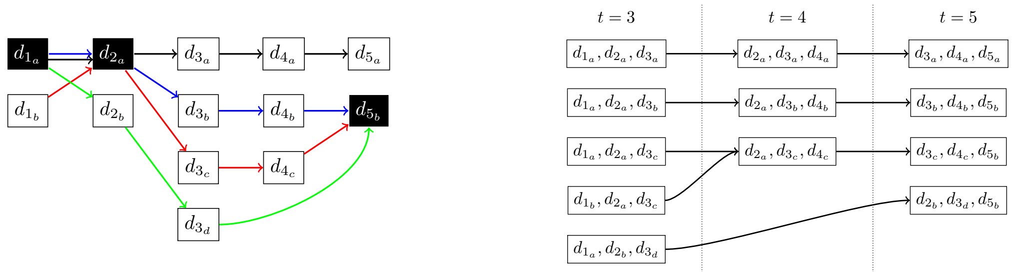

Figure 1: **Left**: A tracking graph with four tracks that conflict over a triplet of detections. Here we associate two indices with a candidate detection so d_{1_a} indicates the a'th detection at time 1. The triplet $d_{1_a}d_{2_a}d_{5_b}$ involved in multiple colored tracks corresponds to the example described in Section 2.2. **Right**: The score for a track is a sum of subtrack scores. Here we illustrate some possible tracks and subtracks (boxes) where directed arrows indicate the valid successors of a given subtrack. We order the subtracks by the time of their final detection. Note that a subtrack may skip some time steps, *e.g.*, $[d_{2_b}d_{3_d}d_{5_b}]$ might correspond to occlusion where no detection is observed at time 4. An optimal valid track can be found by dynamic programming over this graph of subtracks.

in which the LP relaxation is loose and demonstrate how to tighten the bound. In Section 3 we formulate optimization over the tighter bound and discuss inference using column and row generation. In Section 4 we demonstrate the effectiveness of our approach on pedestrian tracking and particle tracking benchmarks. In Section 5 we study the mathematical properties of our relaxation and bounds.

2 Multi-target Tracking Objective

We now describe our ILP formulation of multi-target tracking problem. Given a set of candidate detections $\mathcal{D}$, each with a specified space-time location, our goal is to identify a collection of tracks that describe the trajectories of objects through a scene and the subset of detections associated with each such trajectory.

We denote the (exponentially large) set of all possible tracks by $\mathcal{P}$ and use X to denote the detection-track incidence matrix $X \in \{0,1\}^{|\mathcal{D}|\times|\mathcal{P}|}$ where $X_{dp} = 1$ if and only if track p visits detection d. A solution to the multi-target tracking problem is denoted by the indicator vector $\boldsymbol{\gamma} \in \{0,1\}^{|\mathcal{P}|}$ where $\boldsymbol{\gamma}_p = 1$ indicates that track p is included in the solution and $\boldsymbol{\gamma}_p = 0$ otherwise. A collection of tracks specified by $\boldsymbol{\gamma}$ is a valid solution if and only if each detection is associated with at most one active track. We use $\Theta \in \mathbb{R}^{|\mathcal{P}|}$ to denote the costs associated with tracks (full description of Θ is in Section 2.1). Here Θ_p describes the cost of track p. We now express our tracking problem as an ILP.

$$\min_{\boldsymbol{\gamma}\in\bar{\boldsymbol{\Gamma}}} \Theta^t\boldsymbol{\gamma} \quad \text{with} \quad \bar{\boldsymbol{\Gamma}} = \{\boldsymbol{\gamma} \in \{0,1\}^{|\mathcal{P}|} : \; X\boldsymbol{\gamma} \leq 1\} \tag{1}$$

We note that this is equivalent to finding a maximum-weight set packing which is NP-hard [11].

2.1 Decomposable Track Scores

A classic approach to scoring an individual track is to use a Markov model that incorporates unary scores associated with individual detections along with pairwise compatibilities between subsequent detections assigned to a track. We consider a more general scoring function corresponding to a model in which a track is defined by an ordered sequence of subtracks whose scores may depend arbitrarily on detections across several frames. Let $\mathcal{S}$ denote a set of subtracks, each of which contains K detections where K is a user defined modeling parameter that trades off inference complexity and modeling power. For a given subtrack $s \in \mathcal{S}$, let s_k indicate the k'th detection in the sequence $s = \{s_1, \ldots, s_K\}$ ordered by time from earliest to latest. We describe the mapping of subtracks to tracks using $T \in \{0,1\}^{|\mathcal{S}|\times|\mathcal{P}|}$ where $T_{sp} = 1$ indicates that track p contains subtrack s as a subsequence.

We decompose track costs Θ in terms of the subtrack costs $\theta \in \mathbb{R}^{|\mathcal{S}|}$ where each subtrack s is associated with cost θ_s and use θ_0 to denote a constant birth cost associated with instancing a track. We define the cost of a track p denoted Θ_p as $\Theta_p = \theta_0 + \sum_{s\in\mathcal{S}} T_{sp}\theta_s$. We illustrate the notion of subtracks in Fig 1(right).

2.2 LP Relaxation and Triplets

We now attack optimization in Eq 1 using the well studied tools of LP relaxations. We use $\boldsymbol{\Gamma} = \{\boldsymbol{\gamma} \in [0,1]^{|\mathcal{P}|} : \; X\boldsymbol{\gamma} \leq 1\}$ to denote a convex relaxation of the constraint set $\bar{\boldsymbol{\Gamma}}$.

$$\min_{\boldsymbol{\gamma}\in\bar{\boldsymbol{\Gamma}}} \Theta^t\boldsymbol{\gamma} \quad \geq \quad \min_{\boldsymbol{\gamma}\in\boldsymbol{\Gamma}} \Theta^t\boldsymbol{\gamma} \tag{2}$$

The LP relaxation in Eq 2 only contains constraints for collections of tracks that share a common detection. From the view point of maximum-weight set packing, this includes some cliques of conflicting sets but misses many others. As a concrete example, visualized in Fig 1(left), consider four tracks $\mathcal{P} = \{p_1, p_2, p_3, p_4\}$ over three detections $\mathcal{D} = \{d_1, d_2, d_3\}$, where the first three tracks each contain two of the three detections $\{d_1, d_2\}, \{d_1, d_3\}, \{d_2, d_3\}$, and the fourth track contains all three $\{d_1, d_2, d_3\}$. Suppose the track costs are given by $\Theta_{p_1} = \Theta_{p_2} = \Theta_{p_3} = -4$ and $\Theta_{p_4} = -5$. The optimal integer solution sets $\boldsymbol{\gamma}_{p_4} = 1$, and has a cost of -5. However a lower cost fractional solution sets $\boldsymbol{\gamma}_{p_1} = \boldsymbol{\gamma}_{p_2} = \boldsymbol{\gamma}_{p_3} = 0.5$ and $\boldsymbol{\gamma}_{p_4} = 0$ which has cost -6. Hence the LP relaxation is loose in this case.

A tighter bound can be motivated by the following observation. *For any set of three unique detections the number of tracks that pass through two or more members can be no larger than one.* We utilize this tighter relaxation of $\bar{\boldsymbol{\Gamma}}$. We denote the set of groups of three unique detections (which we refer to as triplets) by $\mathcal{C}$ and index it with c. We use [...] to denote the indicator function.

$$\boldsymbol{\Gamma}^C : \{\boldsymbol{\gamma} \in \mathbb{R}^{|\mathcal{P}|} : \boldsymbol{\gamma} \geq 0, \quad X\boldsymbol{\gamma} \leq 1, \quad C\boldsymbol{\gamma} \leq 1\}$$
$$C_{cp} = [\sum_{d \in c} X_{dp} \geq 2] \quad \forall c \in \mathcal{C}, p \in \mathcal{P} \quad C \in \{0,1\}^{|\mathcal{C}| \times |\mathcal{P}|}$$

3 Dual Optimization over $\boldsymbol{\Gamma}^C$

We write tracking as optimization in the primal and dual form below.

$$\min_{\boldsymbol{\gamma} \in \boldsymbol{\Gamma}^C} \Theta^t \boldsymbol{\gamma} = \max_{\substack{\boldsymbol{\lambda} \geq 0 \\ \boldsymbol{\lambda}^{\mathcal{C}} \geq 0 \\ \Theta + X^t \boldsymbol{\lambda} + C^t \boldsymbol{\lambda}^{\mathcal{C}} \geq 0}} -1^t \boldsymbol{\lambda} - 1^t \boldsymbol{\lambda}^{\mathcal{C}} \tag{3}$$

Given that $\mathcal{P}$ and $\mathcal{C}$ are exponential in the number of detections, we work with a small subset active of columns (tracks) and rows (triplets). The nascent subsets of $\mathcal{P}, \mathcal{C}$ are denoted $\hat{\mathcal{P}}, \hat{\mathcal{C}}$ respectively. In Alg 1 we write column/row generation optimization given subroutines COLUMN($\boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}$), ROW($\boldsymbol{\gamma}$) that identify a group of violated constraints in primal and dual including the most violated in each.

3.1 Row Generation

Finding the most violated row consists of the following optimization: $\max_{c \in \mathcal{C}} \sum_{p \in \mathcal{P}} C_{cp} \boldsymbol{\gamma}_p$.

We generate rows as needed by considering all triplets over detections associated with fractional valued tracks. We consider all triplets of detections, not only those that are nearby in space/time. Hence our work is analogous to the full cycle polytope studied in [18, 15, 1, 2, 3] as opposed to the local cycle polytope applied in [16].

3.2 Column generation without triplets constraints

We first discuss how COLUMN($\boldsymbol{\lambda}, \boldsymbol{\lambda}^c$) is computed efficiently for our track cost model using dynamic programming when $\boldsymbol{\lambda}^c$ is zero valued. We later show how to use this when $\boldsymbol{\lambda}^c$ is not zero valued. We specify that a subtrack s may be preceded by another subtrack $\hat{s}$ if and only if the least recent $K-1$ detections in s correspond to the most recent $K-1$ detections in $\hat{s}$. We denote the set of valid subtracks that may precede a subtrack s as $\{\Rightarrow s\}$.

We use ℓ_s to denote the cost of the cheapest track that terminates at subtrack s. Ordering subtracks by the time of last detection allows efficient computation of ℓ using the following dynamic programming update:

$$\ell_s \leftarrow \theta_s + \boldsymbol{\lambda}_{s_K} + \min\{\min_{\hat{s} \in \{\Rightarrow s\}} \ell_{\hat{s}}, \quad \theta_0 + \sum_{k=1}^{K-1} \boldsymbol{\lambda}_{s_k}\} \tag{4}$$

We find it is useful to add not only the minimum cost track (most violated constraint) to $\hat{\mathcal{P}}$ but also the (most violating) track terminating at each detection. This set of tracks is easy to extract from the dynamic program since it stores the minimum cost track terminating at each subtrack. While this over-generation of constraints substantially increases the number of constraints in the dual, we find that many of these constraints prove to be useful in the final optimization problem . Additionally, in our implementation dynamic programming consumes the overwhelming majority of computation time so adding more columns per iteration yielded faster overall run time.

3.3 Column Generation with triplet constraints

We denote the value of the slack corresponding to an arbitrary column/track p as $V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p)$ and the most violated column/track as $V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ which we define below.

$$V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p) = \Theta_p + \sum_{d \in \mathcal{D}} \boldsymbol{\lambda}_d X_{dp} + \sum_{c \in \hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp} \tag{5}$$
$$V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) = \min_{p \in \mathcal{P}} V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p)$$

Solving for $V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ can not be directly attacked using dynamic programming as in Section 3.2. However dynamic programming can be applied if we ignore the triplet term $\sum_{c \in \hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp}$, providing a lower bound.

This invites a branch and bound (B&B) approach. We find B&B is very practical because experimentally we observe that the number of non-zero values in $\boldsymbol{\lambda}^{\mathcal{C}}$ at any given iteration is small (< 5) for real problems.

Algorithm 1 Column/Row Generation

$\hat{\mathcal{P}} \leftarrow \{\}, \quad \hat{\mathcal{C}} \leftarrow \{\}$
repeat
  $\max_{\substack{\boldsymbol{\lambda}\geq 0 \\ \boldsymbol{\lambda}^{\mathcal{C}}\geq 0 \\ \Theta_{\hat{\mathcal{P}}}+X^t_{(:,\hat{\mathcal{P}})}\boldsymbol{\lambda}+C^t_{(\hat{\mathcal{C}},\hat{\mathcal{P}})}\boldsymbol{\lambda}^{\mathcal{C}}\geq 0}} -1^t\boldsymbol{\lambda} - 1^t\boldsymbol{\lambda}^{\mathcal{C}}$
  Recover $\boldsymbol{\gamma}$ from $\boldsymbol{\lambda}$
  $\dot{\mathcal{P}} \leftarrow \text{COLUMN}(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$
  $\dot{\mathcal{C}} \leftarrow \text{ROW}(\boldsymbol{\gamma})$
  $\hat{\mathcal{P}} \leftarrow [\hat{\mathcal{P}}, \dot{\mathcal{P}}] \qquad \hat{\mathcal{C}} \leftarrow [\hat{\mathcal{C}}, \dot{\mathcal{C}}]$
until $\dot{\mathcal{P}} = []$ and $\dot{\mathcal{C}} = []$

Algorithm 2 Upper Bound Rounding

while $\exists p \in \mathcal{P}$ s.t. $\boldsymbol{\gamma}_p \notin \{0,1\}$ **do**
  $p^* \leftarrow \arg\min_{\substack{p\in\mathcal{P} \\ \boldsymbol{\gamma}_p>0}} \Theta_p\boldsymbol{\gamma}_p - \sum_{\hat{p}\in\mathcal{P}_{\perp p}} \boldsymbol{\gamma}_{\hat{p}}\Theta_{\hat{p}}$
  $\boldsymbol{\gamma}_{\hat{p}} \leftarrow 0 \quad \forall \hat{p} \in \mathcal{P}_{\perp p^*}$
  $\boldsymbol{\gamma}_{p^*} \leftarrow 1$
end while
RETURN $\boldsymbol{\gamma}$

Figure 2: **Left**: Algorithm for dual-optimization of a lower bound on the optimal tracking by column generation where the notation $X_{(:,\hat{\mathcal{P}})}$ denotes selection of a subset of columns of X. We note that the primal solution $\boldsymbol{\gamma}$ is provided during the solution to $\boldsymbol{\lambda}$ by standard LP solvers like MATLAB or CPLEX. **Right**: We compute upper-bounds on the optimal tracking using a rounding procedure which greedily selects primal variables $\boldsymbol{\gamma}$ while removing intersecting tracks. We use $\mathcal{P}_{\perp p}$ to indicate the set of tracks in $\mathcal{P}$ that intersect track p (excluding p itself.

The set of branches in our B&B tree is denoted $\mathcal{B}$. Each branch $b \in \mathcal{B}$ is defined by two sets $\mathcal{D}_{b+}$ and $\mathcal{D}_{b-}$, corresponding to detections that must be included in the track and those that must excluded from the track respectively. We write the set of all tracks that are consistent with a given $\mathcal{D}_{b-}$, $\mathcal{D}_{b+}$ or consistent with both $\mathcal{D}_{b-}$ and $\mathcal{D}_{b+}$ as $\mathcal{P}_{b-}, \mathcal{P}_{b+}$ and $\mathcal{P}_{b\pm}$ respectively. We specify the bounding, branching, and termination operators in Sections 3.3.1,3.3.2 and 3.3.3 respectively. The initial branch b is defined by $\mathcal{D}_{b+} = \mathcal{D}_{b-} = \{\}$.

3.3.1 Bounding Operation

Let $V^b(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ denote the value of the most violating slack over tracks in $\mathcal{P}_{b\pm}$. We can compute a lower-bound for this value, denoted V^b_{lb} by independently optimizing the dynamic program and the triplet penalty.

$$\begin{aligned}
&V^b(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) = \min_{p\in\mathcal{P}_{b\pm}} V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p) \\
&= \min_{p\in\mathcal{P}_{b\pm}} \Theta_p + \sum_{d\in\mathcal{D}} \boldsymbol{\lambda}_d X_{dp} + \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp} \\
&\geq \min_{p\in\mathcal{P}_{b-}} \Theta_p + \sum_{d\in\mathcal{D}} \boldsymbol{\lambda}_d X_{dp} + \min_{p\in\mathcal{P}_{b+}} \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp} \\
&\geq \min_{p\in\mathcal{P}_{b-}} \Theta_p + \sum_{d\in\mathcal{D}} \boldsymbol{\lambda}_d X_{dp} + \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c [\sum_{d\in c}[d \in \mathcal{D}_{b+}] \geq 2] \\
&= V^b_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})
\end{aligned}$$

Observe that dynamic programming can be used to efficiently search over $\mathcal{P}_{b-}$ to minimize the first term. For efficiency, subtracks whose inclusion conflicts with any detection in the required set $\mathcal{D}_{b+}$ can easily be removed before running the dynamic program.

3.3.2 Branch Operation

We now consider the branch operation. We describe an upper bound on $V^b(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ as $V^b_{ub}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$. This is constructed by adding in the active $\boldsymbol{\lambda}^{\mathcal{C}}$ terms ignored when constructing $V^b_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$. Let $p_b = \arg\ \min_{p\in\mathcal{P}_{b-}} \Theta_p + \sum_{d\in\mathcal{D}} \boldsymbol{\lambda}_d X_{dp}$. Then we have:

$$\begin{aligned}
&V^b_{ub}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \\
&= V^b_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) + \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp_b} [\sum_{d\in c}[d \in \mathcal{D}_{b+}] < 2] \qquad (6) \\
&= \Theta_{p_b} + \sum_{d\in\mathcal{D}} \boldsymbol{\lambda}_d X_{dp_b} + \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c [\sum_{d\in c}[d \in \mathcal{D}_{b+}] \geq 2] \\
&\quad + \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp_b} [\sum_{d\in c}[d \in \mathcal{D}_{b+}] < 2] \\
&= V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p_b) + \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c [\sum_{d\in c}[d \in \mathcal{D}_{b+}] \geq 2] \\
&\quad - \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp_b} [\sum_{d\in c}[d \in \mathcal{D}_{b+}] \geq 2] \\
&= V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p_b) + \sum_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c (1 - C_{cp_b}) [\sum_{d\in c}[d \in \mathcal{D}_{b+}] \geq 2] \\
&\geq V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p_b) \geq V^b(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})
\end{aligned}$$

Now consider the largest triplet constraint term $\boldsymbol{\lambda}^{\mathcal{C}}_c$ that is included in $V^b_{ub}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p_b)$ but not $V^b_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$.

$$c^* \leftarrow \arg\max_{c\in\hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp_b} [\sum_{d\in c}[d \in \mathcal{D}_{b+}] < 2] \qquad (7)$$

We create eight new branches for each of the eight different ways of allocating the three detections in the triplet corresponding to c^* to the include (+) and exclude (−) sets. We establish below that if b is the lowest cost branch and $\boldsymbol{\lambda}^C_{c^*} = 0$ then p_b is the track corresponding to the most violated column in $\mathcal{P}$. Hence a branch operator is not applied if $\boldsymbol{\lambda}^C_{c^*} = 0$. We refer to a branch b such that $\boldsymbol{\lambda}^C_{c^*} = 0$ as terminating.

3.3.3 Establishing Optimality at Termination

We now establish that B&B produces the most violated column at termination. We do this by proving that the cost of the track corresponding to the lowest cost branch is both an upper and lower bound on $V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ if that branch is terminating.

Consider branch b^* with corresponding track p_{b^*} such that the following conditions are true: (1) b^* is the lowest cost branch in the B&B tree; (2) b^* is terminating. We write criteria (1),(2) formally below.

$$(1) \quad V^{b^*}_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \leq V^{b}_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \quad \forall b \in \mathcal{B} \tag{8}$$

$$(2) \quad 0 = \max_{c \in \hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp_{b^*}} [\sum_{d \in c} [d \in \mathcal{D}_{b^*+}] < 2] \tag{9}$$

We now establish that $V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p_{b^*}) = V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$. Recall that by definition of B&B that the lowest value bound in any B&B tree is a lower bound on the true solution. Therefore Eq 8 implies the following.

$$V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \geq V^{b^*}_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \tag{10}$$

We now plug Eq 9 into Eq 6, and deduce the following:

$$V^{b^*}_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \geq V(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}, p_{b^*}) \geq V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \tag{11}$$

Observe that Eq 10 establishes that $V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \geq V^{b^*}_{lb}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$. Therefore all inequalities in Eq 11 are equalities and hence p_{b^*} is the lowest cost track.

3.4 Rounding Fractional Solutions

We compute upper bounds using a fast principled method that avoids resolving the LP [7]. Observe that each solution of the LP during the column generation process (Alg 1) corresponds to a (fractional) primal solution in addition to the dual solution (computed "for free" by many LP solvers when solving the dual). We round a fractional $\boldsymbol{\gamma}$ via a greedy iterative approach that, at each iteration, selects the track p with minimum value $\Theta_p \boldsymbol{\gamma}_p$ discounted by the fractional cost of any tracks that share a detection with p (and hence can no longer be added to the tracking if p is added). We write the rounding procedure in Alg 2 using the notation $\mathcal{P}_{\perp p}$ to indicate the set of tracks in $\mathcal{P}$ that intersect track p (excluding p itself).

4 Experiments

4.1 Tracking Pedestrians in Video

We use a part of MOT 2015 training set [13] to train and evaluate real-world tracking models. MOT dataset consists of popular pedestrian benchmark datasets such as TUD, ETH and PETS. Specifically we use the learning framework of [17] with Kalman Filters to train models using ETH-Sunnyday and TUD-Stadtmitte, and test the models on TUD-Campus sequence. For detections we use the raw detector output provided by the MOT dataset. We train the models with varying subtrack length ($K = 2, 3, 4$) and allow for occlusion up to three frames. There are altogether 71 frames and 322 detections in the video, numbers of subtracks are 1,068, 3,633 and 13,090 for $K = 2, 3, 4$. For $K = 2$ we observe 48.5% Multiple Object Tracking Accuracy [5], 11 identity switches and 9 track fragments or for short hand (48.5,11,9). However when setting $K = 3, 4$ the performance is (49,10,7), and (49.9,9,7) which constitutes noticeable improvements over all three metrics. In Fig 3 we compared the timing/cost performance of our algorithm with the baseline algorithm of [7] on problem instances with a loose lower bound.

4.2 Relationship to Lagrangian Relaxation

We now review the work of Butt and Collins [7] which we compare against. [7] can be understood as optimizing over only the Lagrange multipliers $\boldsymbol{\lambda}$ leaving all terms $\boldsymbol{\lambda}^c$ zero valued. Thus [7] relaxes the constraint that no detection is associated with more than one track and enforces it softly with Lagrange multipliers $\boldsymbol{\lambda}$. However [7] adds in an extra constraint that no subtrack is used more than once. This is redundant, though its inclusion allows for solving the relaxed problem via network flow. Using our notation we write the optimization in [7] as:

$$\begin{aligned} \min_{\substack{\gamma \geq 0 \\ X\gamma \leq 1}} \Theta^t \boldsymbol{\gamma} &= \min_{\substack{\gamma \geq 0 \\ X\gamma \leq 1 \\ T\gamma \leq 1}} \Theta^t \boldsymbol{\gamma} \\ &= \min_{\substack{\gamma \geq 0 \\ X\gamma \leq 1 \\ T\gamma \leq 1}} \sum_{p \in \mathcal{P}} \boldsymbol{\gamma}_p (\theta_0 + \sum_{s \in \mathcal{S}} T_{sp}\theta_s) = \max_{\boldsymbol{\lambda} \geq 0} - \sum_{d \in \mathcal{D}} \lambda_d \\ &\quad + \min_{\substack{\gamma \geq 0 \\ T\gamma \leq 1}} \sum_{p \in \mathcal{P}} \boldsymbol{\gamma}_p (\theta_0 + \sum_{s \in \mathcal{S}} T_{sp}\theta_s + \sum_{d \in \mathcal{D}} \lambda_d X_{dp}) \end{aligned} \tag{12}$$

Solving Eq 12 is done using sub-gradient ascent in $\boldsymbol{\lambda}$ instead of linear programming. Given $\boldsymbol{\lambda}$, the minimization problem in Eq 12 corresponds exactly to a min-cost flow problem. In the min-cost flow problem,

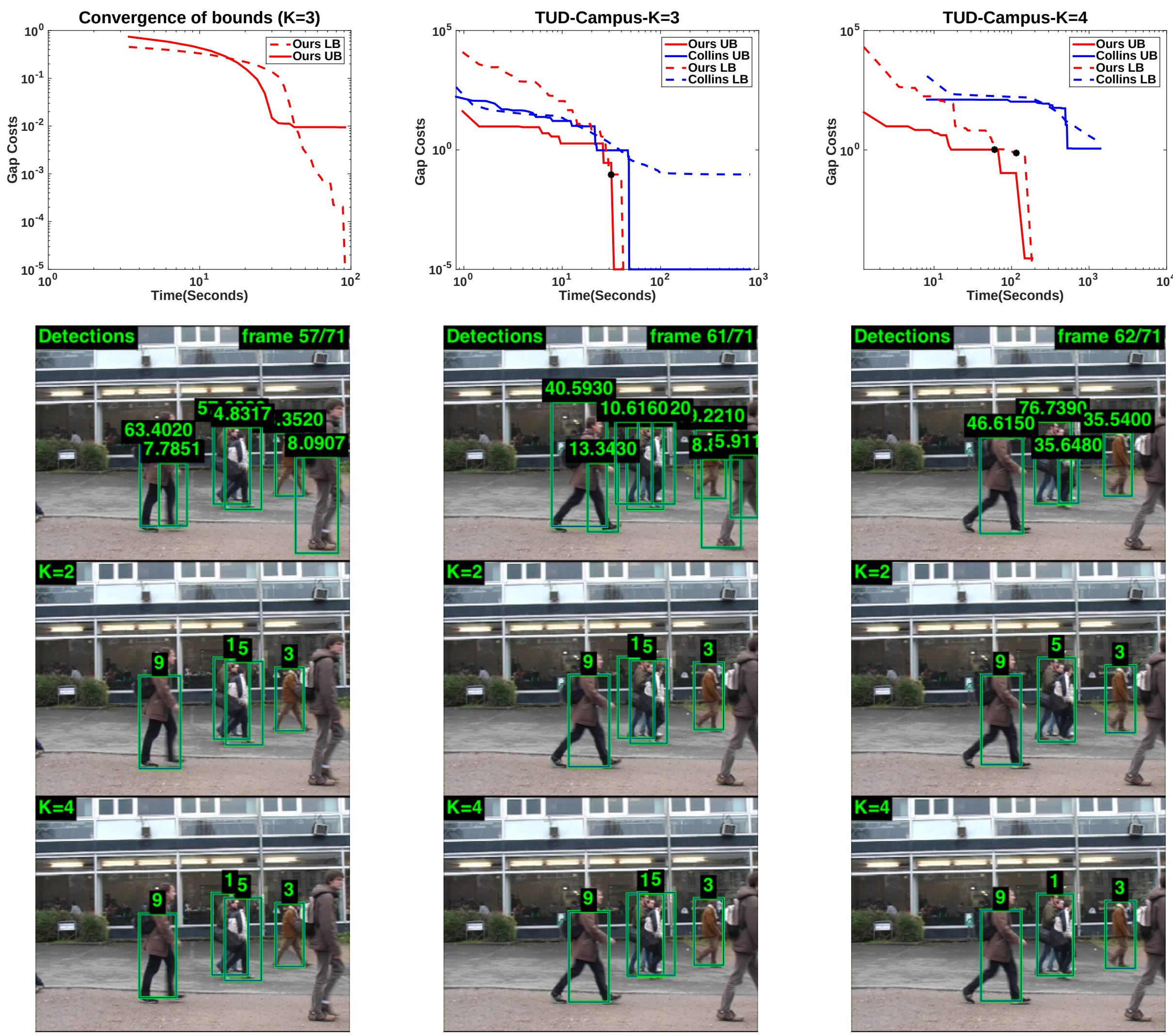

Figure 3: **Top**: left: For $K = 3$ on Particle Tracking Challenge data, we show the convergence of the bounds as a function of time in seconds on the test data set. We display the value of the upper, and lower bounds. middle and right: when training on a subset of motion features on MOT dataset we get instances with loose bound. For the two examples we plot the gap (absolute value of the difference) between the bounds and the final lower bound as a function of time. We indicate each time that a triplet is added with a black dot on the lower bound plot. In all examples the bound of [7] is loose and at least one triplet is needed to produce a tight bound which results in visually compelling tracks. In both comparisons against [7] our upper- and lower- bounds are tight at termination. **Bottom**: We illustrate a qualitative example of improvement as a result of increasing subtrack length. Top row is detector output and associated confidence provided by [13]. Second row and third row correspond to trackers of subtrack length $K = 2$ and $K = 4$ respectively. Notice that for $K = 2$ track 1 changes identity to 5, while with $K = 4$ the identity of track 1 does not change. Missing detections in tracking results are interpolated linearly and tracks.

nodes indicate subtracks and edges indicate valid transitions between subtracks. Each time the network flow is solved, a set of tracks is produced that may share detections. A greedy approach is used to select a good subset of the tracks produced. Hence our inference algorithm can be understood as optimizing a tighter relaxation that uses dynamic programming in place of network flow inference. The tighter bound is a consequence of the addition of the triplets.

4.3 Synthetic Data: Experiments on the Particle Tracking Challenge Data

We applied our algorithm to the data from the Particle Tracking Challenge (PTC) simulated data set for high density microtubules (SNR7),which consists of two videos (train,test) of 99 images of 512×512 pixels over time where the test data set contains 6733 tracks that cover 71035 detections. We take the set of ground truth detections as the set of detections and apply our tracking algorithm. From the set of detections we generate subtracks of K detections as follows. Consider a directed graph where the nodes are the set of all detections. For each detection d we draw a line from d to each of its three spatial nearest neighbors in the following frame. The set of all paths containing K nodes in this graph is the set of subtracks. In total we find 1,873,341 subtracks. We are provided with costs for each subtrack via logistic regression based on motion features consisting of autocorrelation, and autocovariance, and other distance features.

With $K = 3$ and optimized hyper-parameters we reach a Jaccard score of 0.924 as compared to a baseline of 0.754 . We identified 6329 tracks that are in the ground truth, missed 404 tracks that are in the ground truth and identified 118 tracks that are not in the ground truth. These results are produced via providing our output to the benchmarking code associated with [8]. In Fig 3, we apply Alg 1 and study tracking performance in terms of accuracy and cost.

The purpose of the synthetic experiments is to demonstrate that the generation of massive numbers of columns at once is an efficient mechanism to solve problems where the number of tracks is enormous and repeated calls to dynamic programming is computationally difficult. The value of the model and of the triplets is explored in the real data experiments on pedestrian tracking.

5 Study of the bounds

We now consider computing an anytime lower bound on the optimal tracking. We use $\mathcal{P}^d$ to refer to the set of tracks terminating at detection d. We rely on the redundant constraint that no two tracks terminate at the same detection.

$$\min_{\substack{\boldsymbol{\gamma} \in \{0,1\}^{|\mathcal{P}|} \\ \sum_{p \in \mathcal{P}^d} \boldsymbol{\gamma}_p \leq 1}} \max_{\substack{\boldsymbol{\lambda}^{\mathcal{C}} \geq 0 \\ \boldsymbol{\lambda} \geq 0}} \Theta^t \boldsymbol{\gamma} + \boldsymbol{\lambda}^t (X\boldsymbol{\gamma} - 1) + \boldsymbol{\lambda}^{\mathcal{C}t} (C\boldsymbol{\gamma} - 1) \tag{13}$$

We now relax the optimization and consider any non-negative $\boldsymbol{\lambda}, \boldsymbol{\lambda}^C$.

$$\begin{aligned} &\text{Eq } 13 \geq -\boldsymbol{\lambda}^t 1 - \boldsymbol{\lambda}^{\mathcal{C}t} 1 \\ &+ \min_{\substack{\boldsymbol{\gamma} \in \{0,1\}^{|\mathcal{P}|} \\ \sum_{p \in \mathcal{P}^d} \boldsymbol{\gamma}_p \leq 1}} \Theta^t \boldsymbol{\gamma} + \boldsymbol{\lambda}^t X \boldsymbol{\gamma} + \boldsymbol{\lambda}^{\mathcal{C}t} C \boldsymbol{\gamma} \geq -\boldsymbol{\lambda}^t 1 - \boldsymbol{\lambda}^{\mathcal{C}t} 1 \\ &+ \sum_{d \in \mathcal{D}} \min\{0, \min_{p \in \mathcal{P}^d} \Theta_p + \sum_{d \in \mathcal{D}} \boldsymbol{\lambda}_d X_{dp} + \sum_{c \in \hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp}\} \end{aligned}$$

For short hand we define the following quantity $V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ as the lowest cost over tracks terminating at detection d. $V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) = \min_{p \in \mathcal{P}^d} \Theta_p + \sum_{d \in \mathcal{D}} \boldsymbol{\lambda}_d X_{dp} + \sum_{c \in \hat{\mathcal{C}}} \boldsymbol{\lambda}^{\mathcal{C}}_c C_{cp}$

We bound $V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ from below in two different ways. First, we ignore the $\boldsymbol{\lambda}^{\mathcal{C}}$ terms and optimize via dynamic programming producing the following bound. $V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \geq \min_{\substack{s \in \mathcal{S} \\ s_K = d}} \ell_s$.

However we also bound $V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ by the minimizer over all d hence $V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \geq V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$. Combining the two bounds on $V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}})$ we produce the following bound.

$$V^{d*}(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) \geq \max\{V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}), \min_{\substack{s \in \mathcal{S} \\ s_K = d}} \ell_s\} \tag{14}$$

At termination of column/row generation no violated constraints exist so $V^*(\Theta, \boldsymbol{\lambda}, \boldsymbol{\lambda}^{\mathcal{C}}) = 0$ and thus the lower bound has value identical to the LP relaxation over $\boldsymbol{\Gamma}^C$ in Eq 3.

5.1 Further Tightening the Bound

Triplets are not the only mechanism by which the bound can be tightened. In practice we find that only these are needed for our application but we now explore related constraints which are known as generalized odd set inequalities [10]. Like the triplets these constraints can also be used inside of branch and bound dynamic programming. We consider inequalities specified by a pair of integers m_1, m_2 which are both greater than or equal to two. Observe that for any group of $(m_1 m_2 - 1)$ detections the number of disjoint tracks containing m_1 or more detections is no greater than m_2-1. We write this constraint as:

$$\mathcal{K}(m_1, m_1 m_2 - 1) \triangleq \sum_{p \in \mathcal{P}} [\sum_{d \in \hat{\mathcal{D}}} X_{dp} \geq m_1] \gamma_p \leq m_2 - 1$$

where $\hat{\mathcal{D}}$ is some set of cardinality $m_1 m_2 - 1$. This generalizes our triplet constraints which were of type $\mathcal{K}(2,3)$. Observe that any $m_1 > 2, m_2$ and integer k such that $2 < k < m_1$ the constraint $\mathcal{K}(m_1, m_1 m_2 - k)$ is dominated by $\mathcal{K}(m_1, m_1 m_2 - 1)$. For example, $\mathcal{K}(4,7)$ dominates $\mathcal{K}(4,6)$ and $\mathcal{K}(4,5)$.

To see that adding such inequalities tightens the bound, consider a set of $m_1 m_2 - 1$ detections $\hat{\mathcal{D}}$ and a fractional solution $\boldsymbol{\gamma}$ where the total weight of tracks passing through each group of m_1 detections is equal to $\frac{1}{\binom{m_1 m_2 - 2}{m_1 - 1}}$ so that $\sum_{p \in \mathcal{P}} \boldsymbol{\gamma}_p X_{dp} = 1$ for all $d \in \hat{\mathcal{D}}$. We now establish that this solution satisfies inequalities $\mathcal{K}(m_1, m_1 k - 1)$ for $2 \leq k < m_2$ but not for $k = m_2$, thus showing the theoretical need for constraints beyond triplets. Let $\hat{\mathcal{D}}^k$ be any subset of $m_1 k - 1$ members of $\hat{\mathcal{D}}$. Constraint $\mathcal{K}(m_1, m_1 k - 1)$ is satisfied by $\boldsymbol{\gamma}$ if

$$k - 1 \geq \sum_{p \in \mathcal{P}} [\sum_{d \in \mathcal{D}^k} X_{dp} \geq m_1] \boldsymbol{\gamma}_p \tag{15}$$

$$1 \geq \frac{\frac{1}{k-1}}{\binom{m_1 m_2 - 2}{m_1 - 1}} \sum_{p \in \mathcal{P}} [\sum_{d \in \mathcal{D}^k} X_{dp} = m_1] \tag{16}$$

$$= \frac{\frac{1}{k-1}}{\binom{m_1 m_2 - 2}{m_1 - 1}} \binom{k m_1 - 1}{m_1}$$

$$= \frac{k m_1 - 1}{(k-1) m_1} \frac{\prod_{i = k m_1 - m_1}^{k m_1 - 2} i}{\prod_{i = m_2 m_1 - m_1}^{m_2 m_1 - 2} i}$$

$$= \frac{k m_1 - 1}{(k-1) m_1} \prod_{i=0}^{m-2} \frac{k m_1 - m_1 + i}{m_2 m_1 - m_1 + i}$$

Observe that each term under the product is a positive real number less than or equal to one. Thus we have the following upper bound on the r.h.s. of Eqn.16

$$\frac{k m_1 - 1}{(k-1) m_1} \prod_{i=0}^{m-2} \frac{k m_1 - m_1 + i}{m_2 m_1 - m_1 + i} \tag{17}$$

$$\leq \frac{k m_1 - 1}{(k-1) m_1} \left(\frac{k m_1 - m_1}{m_2 m - m_1} \right) = \frac{k m_1 - 1}{m_2 m_1 - m_1}$$

When $k < m_2$, we observe that $\frac{k m_1 - 1}{m_2 m_1 - m_1} \leq 1$ so the constraints $\mathcal{K}(m_1, m_1 k - 1)$ with $k < m_2$ are satisfied. However, the constraint $\mathcal{K}(m_1, m_1 m_2 - 1)$ is violated since

$$\frac{m_2 m_1 - 1}{(m_2 - 1) m_1} \prod_{i=0}^{m-2} \frac{m_2 m_1 - m_1 + i}{m_2 m_1 - m_1 + i} \tag{18}$$

$$= \frac{m_2 m_1 - 1}{m_2 m_1 - m_1} > 1$$

5.2 Are higher-order inequalities needed?

Empirically we observe that triplets are sufficient to ensure integral solutions on the problems that we study. We now explain this by arguing that when the lower-order inequalities are satisfied it (generically) becomes increasingly difficult to conceive of violated higher order inequalities (over large sets of detections). For a fixed value of m_1 consider varying values of m_2 and denote a corresponding set of detections as $\mathcal{D}^{m_1 m_2 - 1}$. As suggested by Eq 18, finding violated constraints requires an increasingly large portion of the available tracks to be active. Recall that the active tracks participating in the constraint must have at least m_1 detections in $\mathcal{D}^{m_1 m_2 - 1}$. It might appear that activating tracks with more than m_1 detections in $\mathcal{D}^{m_1 m_2 - 1}$ would contribute to a violated constraint. However, a uniform weighting over tracks with exactly m_1 detections maximizes the value on the right side of the inequality in Eq 15 while still satisfying the basic constraint $X\gamma \leq 1$. The maximum total violation is thus bounded by $\frac{m_2 m_1 - 1}{m_2 m_1 - m_1} - 1$ which shrinks as m_2 grows.

In our application, having a great proportion of tracks active with exactly m_1 detections in some set $\mathcal{D}^{m_1 m_2 - 1}$ is rather unlikely (especially for large m_1). This is a consequence of utilizing sub-tracks containing multiple individual detection for which the scoring is reasonably discriminative. This severely restricts the space of low-cost tracks making it unlikely to find a large set of detections that also participate in many low-cost tracks.

6 Conclusions

We have introduced a new method for multi-target tracking built on an LP relaxation of the maximum-weight set packing problem. Our core contribution is a column generation approach that exploits dynamic programming to generate a large number of candidate tracks concurrently. This yields an efficient algorithm and provides rigorous bounds that can be tightened via row generation. We empirically observe that our algorithm rapidly produces compelling tracking results along with strong anytime performance relative the baseline Lagrangian relaxation[7].

Acknowledgements: CF was supported by NSF grants IIS-1618806 and IIS-1253538.

References

[1] F. Barahona. On the computational complexity of ising spin glass models. *Journal of Physics A: Mathematical, Nuclear and General*, 15(10):3241–3253, april 1982.

[2] F. Barahona. On cuts and matchings in planar graphs. *Mathematical Programming*, 36(2):53–68, november 1991.

[3] F. Barahona and A. Mahjoub. On the cut polytope. *Mathematical Programming*, 60(1-3):157–173, September 1986.

[4] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1996.

[5] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: The clear mot metrics. *Journal on Image Video Processing*, 2008.

[6] S. S. Blackman. Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):5–18, 2004.

[7] A. Butt and R. Collins. Multi-target tracking by lagrangian relaxation to min-cost network flow. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1846–1853, June 2013.

[8] N. Chenouard, I. Smal, F. de Chaumont, M. Maska, I. F. Sbalzarini, Y. Gong, J. Cardinale, C. Carthel, S. Coraluppi, M. Winter, A. R. Cohen, W. J. Godinez, K. Rohr, Y. Kalaidzidis, L. Liang, J. Duncan, H. Shen, Y. Xu, K. E. G. Magnusson, J. Jalden, H. M. Blau, P. Paul-Gilloteaux, P. Roudot, C. Kervrann, F. Waharte, J.-Y. Tinevez, S. L. Shorte, J. Willemse, K. Celler, G. P. van Wezel, H.-W. Dan, Y.-S. Tsai, C. O. de Solorzano, J.-C. Olivo-Marin, and E. Meijering. Objective comparison of particle tracking methods. *Nature Methods*, 11(3):281–289, Jan. 2014.

[9] P. Gilmore and R. Gomory. A linear programming approach to the cutting-stock problem. *Operations Research (volume 9)*, 1961.

[10] O. Heismann and R. Borndörfer. A generalization of odd set inequalities for the set packing problem. In *Operations Research Proceedings 2013*, pages 193–199. Springer, 2014.

[11] R. M. Karp. Reducibility among combinatorial problems. The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

[12] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4696–4704, 2015.

[13] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. MOTChallenge 2015: Towards a benchmark for multi-target tracking. *arXiv:1504.01942 [cs]*, Apr. 2015. arXiv: 1504.01942.

[14] A. P. Poore and N. Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems arising from multitarget tracking. *SIAM Journal on Optimization*, 3(3):544–563, 1993.

[15] D. Sontag, D. K. Choe, and Y. Li. Efficiently searching for frustrated cycles in MAP inference. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence (UAI-12)*, pages 795–804, Corvallis, Oregon, 2012. AUAI Press.

[16] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening lp relaxations for map using message passing. In *Proceedings of the Twenty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 503–510, july 2008.

[17] S. Wang and C. Fowlkes. Learning optimal parameters for multi-target tracking. In *British Machine Vision Conference*, 2015.

[18] J. Yarkony and C. Fowlkes. Planar ultrametrics for image segmentation. In *Neural Information Processing Systems*, 2015.

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