A Primal-Dual Solver for Large-Scale Tracking-by-Assignment

Stefan Haller[1], **Mangal Prakash**[2], **Lisa Hutschenreiter**[1], **Tobias Pietzsch**[2], **Carsten Rother**[1], **Florian Jug**[2], **Paul Swoboda**[3], **Bogdan Savchynskyy**[1]
[1]Visual Learning Lab, Heidelberg University, [2]Center for Systems Biology, Dresden,
[3]Max-Planck-Institute for Informatics, Saarbrücken

Abstract

We propose a fast approximate solver for the combinatorial problem known as *tracking-by-assignment*, which we apply to cell tracking. The latter plays a key role in discovery in many life sciences, especially in cell and developmental biology. So far, in the most general setting this problem was addressed by off-the-shelf solvers like Gurobi, whose run time and memory requirements rapidly grow with the size of the input. In contrast, for our method this growth is nearly linear.

Our contribution consists of a new (1) decomposable compact representation of the problem; (2) dual block-coordinate ascent method for optimizing the decomposition-based dual; and (3) primal heuristics that reconstructs a feasible integer solution based on the dual information. Compared to solving the problem with Gurobi, we observe an up to 60 times speed-up, while reducing the memory footprint significantly. We demonstrate the efficacy of our method on real-world tracking problems.

1 Introduction

The tracking problem consists of segmenting images obtained over T time steps and matching the segments in consecutive images to each other. In the case of cell tracking each cell in the image t must be matched to the corresponding cell (or a pair of cells in case of a division) in the image $t+1$. Tracking problems are important not only for bioimaging [4, 18, 36] but also for general computer vision [23, 38]. Cell tracking represents one of the hardest types of this problem, due to the existence of cell divisions and the indistinguishability of individual cells.

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

The most successfully deployed tracking models are typically formulated as integer linear programs (ILPs) and are in general NP-hard. This makes the commonly used ILP-based optimizations only amenable to moderately sized tracking instances. With the advent of modern microscopy techniques, this bottleneck became a limiting factor for many real-world applications, leading either to faulty tracking results, or impractical optimization problems. In order to address run time and scalability issues, we propose a new method to solve tracking-by-assignment problems, whose iterations as well as memory footprint scale nearly linearly with the problem size. Its convergence speed enables us to obtain high-quality approximate solutions in a fraction of the time required by the best off-the-shelf solvers.

While this paper focuses on biologically motivated object tracking, similar problems also arise in other vision domains [11, 34, 37]. We believe that with appropriate modifications our ideas can be applied there as well.

Related work. Any visual tracking contains two key interrelated operations: segmentation and matching. On one side, matching requires segmentation, on the other side, the segmentation quality can often be significantly improved by the matching results. There are approaches addressing these two problems jointly [16], however, the resulting algorithms are quite time consuming. Less expensive modeling techniques can be categorized into tracking-by-model-evolution and tracking-by-assignment [15].

In *tracking-by-model-evolution*, objects are detected in the first frame and a model of their properties, e. g. shape and position, is obtained. This model is then updated greedily for pairs of neighboring frames, thereby tracking all detected objects. For sensible results these methods typically require a high temporal resolution [15, 20]. Recently, neural networks for tracking-by-model-evolution have been proposed that jointly tackle segmentation and tracking and have the ability to handle object divisions [1, 27]. They incorporate temporal information, e. g. by using LSTMs [1]. However, such network based approaches require vast amounts of annotated training data, which is typically

not available for biomedical tracking problems.

By contrast, *tracking-by-assignment* first segments potential object candidates in all time frames [13, 17, 20, 25, 29, 30]. There are two important cases: a single segmentation hypothesis per object [20, 29] and multiple ones [17, 30]. Multiple segmentation hypotheses may correspond to different (typically overlapping) positions of the same object, to parts of a single object looking like separate ones, or to several objects close to each other looking like a single one. Apart from better tracking quality, multiple segmentation approaches facilitate user-driven proofreading of automated results [14, 17].

Existing optimization methods for tracking-by-assignment fall into two categories: (*i*) local approaches that attempt to overcome scalability issues by decomposing the overall tracking problem into smaller sub-problems [3, 12, 39], and (*ii*) global approaches, treating the whole spatiotemporal problem jointly [2, 10, 25, 26]. While the first type of method scales better with the problem size, the second one leads to better solutions. Among approaches of the second type we distinguish the work [26], that couples multiple min-cost-flow networks to handle divisions and finds an approximate solution to the overall problem with an off-the-shelf LP solver. Another notable contribution is the primal heuristics in [10], based on sequentially computing shortest paths in an augmented flow graph which accounts for object divisions. This work generalizes the approach of [25] which utilizes the Viterbi algorithm. However, these works do not handle overlapping segmentation hypotheses. Additionally, [10] and [25] do not provide any bounds on the quality of the proposed solutions. The work [2] employs stochastic gradient descent to maximize a Lagrange dual based on multiple min-cost-flow subproblems, and obtains primal solutions by rounding. However, the method does not allow for cell divisions.

After all, the most general models including both multiple hypothesis and object divisions have been addressed with off-the-shelf ILP solvers only [13, 14, 17, 30]. Building a scalable solver able to compete with, for example, Gurobi in this case seems to be a non-trivial task, which has not been addressed in the literature yet.

Contributions. We propose a new approximate optimization method for cell-tracking problems, which favorably compares with Gurobi in terms of run time and memory footprint, while delivering solutions of a comparable quality. Our method is able to handle cell divisions and multiple segmentation hypotheses. Together with an approximate primal solution it provides a lower bound on the optimum. This is achieved by optimizing the Lagrange dual problem constructed from a new compact decomposition. To optimize the dual we propose a specialized fast converging algorithm based on the block-coordinate ascent principle. The approximate primal solution is obtained with a novel primal heuristics based on conflict resolution and greedy elongation of trajectories. While the dual solver simplifies the objective function by reweighting its costs, the primal one reconstructs an integer solution based on these costs. We empirically show advantages of our framework on publicly available instances of the cell-tracking challenge [36], on instances of developing flywing tissue, and on an instance of nuclei tracking in developing drosophila embryos. These datasets represent different biological applications and exhibit diverse characteristics. Therefore, we believe our method to be applicable to a wide spectrum of cell-tracking problems.

The focus of our work is to improve the optimization stage of a typical tracking-by-assignment pipeline, therefore, we do not address modeling and segmentation aspects of the problem here. We assume that for each time step $t \in \{1, \dots, T\}$ a set of *segmentation hypotheses* is available along with a set of possible transitions and the corresponding costs.

Mathematical proofs and information about our code and models can be found in the supplement.

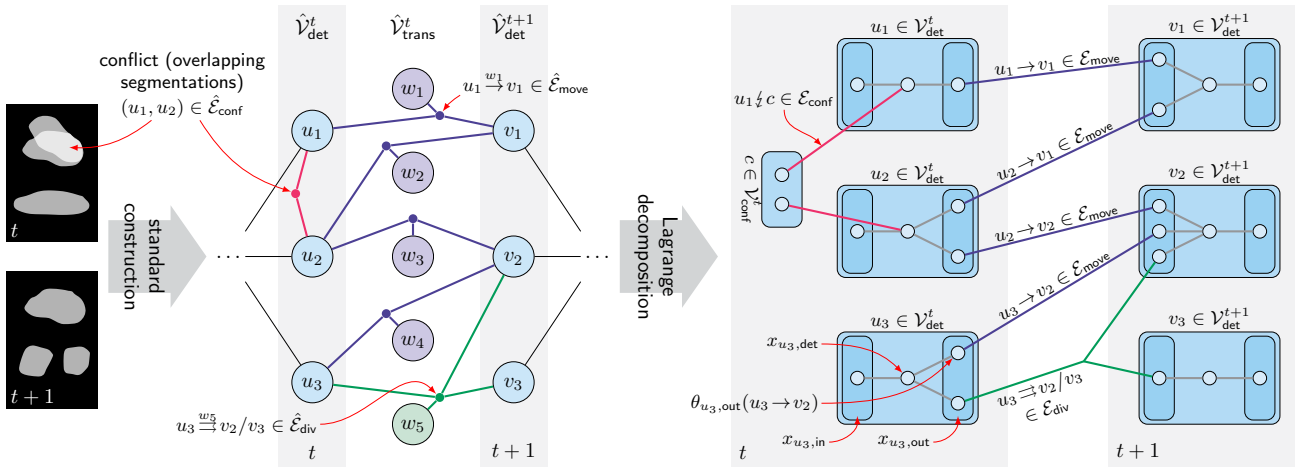
2 Standard tracking as ILP

The standard modeling approach for tracking-by-assignment is based on its *problem (hyper-)graph* representation [13, 14, 20, 24, 29, 30], see Figure 1 (b). In the following we omit the prefix *hyper-* and use the *hat* superscript (as in $\hat{\mathcal{V}}$ or $\hat{\mathcal{E}}$) for the standard problem graph to distinguish it from the graph we propose later. Nodes $\hat{\mathcal{V}}$ of a problem graph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}})$ are associated with *finite-valued variables*, and edges $\hat{\mathcal{E}} \subseteq 2^{\hat{\mathcal{V}}}$ correspond to the *coupling constraints* between the respective nodes. Here, $2^{\hat{\mathcal{V}}}$ denotes the power set of $\hat{\mathcal{V}}$.

Node set. For tracking-by-assignment the set of nodes $\hat{\mathcal{V}}$ is divided into disjoint subsets $\hat{\mathcal{V}}^t$ corresponding to each time step $t \in \{1, \dots, T\}$, i. e. $\hat{\mathcal{V}} = \bigcup_{t=1}^{T} \hat{\mathcal{V}}^t$. In its turn, each subset $\hat{\mathcal{V}}^t$ is subdivided into a set $\hat{\mathcal{V}}^t_{\mathsf{det}}$ representing the *segmentation hypothesis (detections)* at time step t, and a set $\hat{\mathcal{V}}^t_{\mathsf{trans}}$ representing the possible *transitions (moves, divisions)* from time step t to $t+1$. We will write $\hat{\mathcal{V}}_{\mathsf{det}} = \bigcup_{t=1}^{T} \hat{\mathcal{V}}^t_{\mathsf{det}}$ and $\hat{\mathcal{V}}_{\mathsf{trans}} = \bigcup_{t=1}^{T} \hat{\mathcal{V}}^t_{\mathsf{trans}}$ for the sets of all detection and transition nodes.

Each segmentation hypothesis as well as each transition is associated with a *binary variable*, i. e. its value is in the set $\{0, 1\}$. We will refer to the variable corresponding to node $v \in \hat{\mathcal{V}}$ as x_v, where $x_v \in \{0, 1\}$. A variable is said to be *active* if it assumes value 1.

Edge set. The set of edges $\hat{\mathcal{E}}$ coupling the nodes is divided into subsets $\hat{\mathcal{E}}^t$ corresponding to each time step $t \in \{1, \dots, T\}$, i. e. $\hat{\mathcal{E}} = \bigcup_{t=1}^{T} \hat{\mathcal{E}}^t$. In turn, $\hat{\mathcal{E}}^t = \hat{\mathcal{E}}^t_{\mathsf{move}} \cup \hat{\mathcal{E}}^t_{\mathsf{div}} \cup \hat{\mathcal{E}}^t_{\mathsf{conf}}$, where $\hat{\mathcal{E}}^t_{\mathsf{move}} \subseteq \hat{\mathcal{V}}^t_{\mathsf{det}} \times \hat{\mathcal{V}}^{t+1}_{\mathsf{det}} \times \hat{\mathcal{V}}^t_{\mathsf{trans}}$ and



(a) oversegmentation **(b)** standard problem graph $\hat{\mathcal{G}}$ (Sec. 2) **(c)** our compact decomposition graph $\mathcal{G}$ (Sec. 3)

Figure 1: Illustration of the standard model and our proposed problem decomposition. Each detection (segmentation hypothesis) in **(a)** is represented by a node in $\hat{\mathcal{V}}_{\text{det}}$ in the standard model **(b)**. Possible transitions between time steps are represented by a node in $\hat{\mathcal{V}}_{\text{trans}}$. Hyper-edges $\hat{\mathcal{E}}$ handle detection and transition coupling constraints. Our Lagrange decomposition **(c)** represents each detection together with its incoming and outgoing transitions by a single node in $\mathcal{V}_{\text{det}}$. Conflicting detections are represented by separate nodes in $\mathcal{V}_{\text{conf}}$. Edges $\mathcal{E}$ correspond to coupling constraints between nodes and refer to transitions (blue), divisions (green), and conflicts (red).

$\hat{\mathcal{E}}^t_{\text{div}} \subseteq \hat{\mathcal{V}}^t_{\text{det}} \times (\hat{\mathcal{V}}^{t+1}_{\text{det}})^2 \times \hat{\mathcal{V}}^t_{\text{trans}}$ are edges corresponding to possible moves and divisions of the cells between time steps t and $t+1$, and $\hat{\mathcal{E}}^t_{\text{conf}} \subseteq 2^{\hat{\mathcal{V}}^t_{\text{det}}}$ are the edges prohibiting the activation of conflicting (intersecting) segmentation hypothesis at time step t. Note that for each node in $\hat{\mathcal{V}}^t_{\text{trans}}$ there is exactly one incident edge.

Coupling constraints. Let $(u, v, w) \in \hat{\mathcal{E}}^t_{\text{move}}$ be a possible move from time step t to $t+1$ connecting nodes $u \in \hat{\mathcal{V}}^t_{\text{det}}$ and $v \in \hat{\mathcal{V}}^t_{\text{det}}$ via transition node $w \in \hat{\mathcal{V}}^t_{\text{trans}}$. We write $u \overset{w}{\to} v$ for such edges. Then the set of corresponding coupling constraints is defined as

$$\forall\, u \overset{w}{\to} v \in \hat{\mathcal{E}}_{\text{move}} : \quad x_w \le x_u \;\wedge\; x_w \le x_v\,, \tag{1}$$

ensuring that if either of the hypothesis is deactivated ($x_u = 0$ or $x_v = 0$), the move is deactivated as well ($x_w = 0$). Analogously, we denote possible divisions $(u, v, v', w) \in \hat{\mathcal{E}}^t_{\text{div}}$ by $u \overset{w}{\rightrightarrows} v/v'$, and obtain the following coupling constraints for divisions:

$$\forall\, u \overset{w}{\rightrightarrows} v/v' \in \hat{\mathcal{E}}_{\text{div}} : x_w \le x_u \;\wedge\; x_w \le x_v \;\wedge\; x_w \le x_{v'}\,. \tag{2}$$

For any detection node $v \in \hat{\mathcal{V}}_{\text{det}}$ we denote by $\hat{\text{in}}(v)$ the set of *incoming* transitions, i. e. $\hat{\text{in}}(v) := \{w \in \hat{\mathcal{V}}_{\text{trans}} \mid \exists\, u\colon u \overset{w}{\to} v \in \hat{\mathcal{E}}_{\text{move}} \text{ or } \exists\, u, v'\colon u \overset{w}{\rightrightarrows} v/v' \in \hat{\mathcal{E}}_{\text{div}} \vee u \overset{w}{\rightrightarrows} v'/v \in \hat{\mathcal{E}}_{\text{div}}\}$. Likewise, for all $u \in \mathcal{V}_{\text{det}}$ we define $\hat{\text{out}}(u) := \{w \in \hat{\mathcal{V}}_{\text{trans}} \mid \exists\, v\colon u \overset{w}{\to} v \in \hat{\mathcal{E}}_{\text{move}} \text{ or } \exists\, v, v'\colon u \overset{w}{\rightrightarrows} v/v' \in \hat{\mathcal{E}}_{\text{div}}\}$ as the set of *outgoing* transitions.

To guarantee that each hypothesis at time step t is matched to at most one at time steps $t+1$ and $t-1$, *uniqueness constraints* are introduced as follows:

$$\forall\, v \in \hat{\mathcal{V}}_{\text{det}} : \sum_{w \in \hat{\text{in}}(v)} x_w \le 1 \quad \text{and} \quad \sum_{w \in \hat{\text{out}}(v)} x_w \le 1\,. \tag{3}$$

Finally, conflicting segmentation hypothesis are connected via similar constraints:

$$\forall\, c \in \hat{\mathcal{E}}_{\text{conf}} : \sum_{v \in c} x_v \le 1\,. \tag{4}$$

Objective function. Let $\hat{\mathcal{X}} \subseteq \{0,1\}^{|\hat{\mathcal{V}}|}$ be the set of binary vectors x satisfying all coupling constraints defined by (1)-(4). Each coordinate x_v, $v \in \hat{\mathcal{V}}$, is associated with a *cost* $\theta_v \in \mathbb{R}$ based on image data (for segmentation hypothesis) and geometric priors (for transitions). The goal of tracking is to find an assignment $x \in \hat{\mathcal{X}}$ that minimizes the cost of the active binary variables, i. e. which solves

$$\min_{x \in \hat{\mathcal{X}}} \langle \theta, x \rangle\,, \tag{5}$$

where $\theta = (\theta_v)_{v \in \hat{\mathcal{V}}}$. Problem (5) is the standard ILP representation of the tracking problem. In this form it is usually addressed (see e. g. [13, 14, 20, 24, 29, 30]) by off-the-shelf solvers like Gurobi [9] or CPLEX [5]. However, the run time and memory requirements of these solvers rapidly grow with the size of the input. Moreover, even solving an LP relaxation of the above problem, i. e. considering a vector in $[0,1]^{|\hat{\mathcal{V}}|}$ satisfying (1)-(4), requires a significant time using standard solvers, as they are based on simplex or interior point methods with a super-linear iteration complexity. Note that first order subgradient-based methods perform even slower than the standard solvers [19].

3 Our decomposable representation

Efficiency of large-scale approximate optimization methods heavily depends on the problem decomposition used to build a dual problem. A good decomposition should contain small number of easily tractable subproblems. Consider a trivial decomposition of (5), when every binary variable corresponds to a separate subproblem. It would satisfy the tractability condition, but the large number of subproblems would significantly slow down the optimization. Therefore, below we give an alternative representation of the problem (5), which leads to a natural decomposition with a much smaller number of easily tractable subproblems.

Lagrange decomposition idea. Assume we want to minimize a function $F(x)$ representable as $F(x) = F_1(x) + F_2(x)$. The Lagrange decomposition [6, 7, 8, 32] duplicates the variable x and introduces the equality constraint $x_1 = x_2$, i. e. $\min_x F(x) = \min_{x_1,x_2\colon x_1=x_2} \big(F_1(x_1) + F_2(x_2)\big)$. Dualization of the constraint $x_1 = x_2$ leads to the *Lagrange dual problem*, which forms a lower bound for the original problem:

$$\min_x F(x) \geq \max_\lambda \min_{x_1,x_2} \big(F_1(x_1) + F_2(x_2) + \langle \lambda, x_1 - x_2\rangle\big)$$
$$= \max_\lambda \Big[\min_{x_1}\big(F_1(x_1) + \langle \lambda, x_1\rangle\big) + \min_{x_2}\big(F_2(x_2) - \langle \lambda, x_2\rangle\big)\Big]. \quad (6)$$

Tightness of the lower bound as well as the efficiency of its maximization depend on the decomposition of F into F_1 and F_2. Ideally, the minimization subproblems over x_1 and x_2 are solvable in closed form, and the coupling constraint $x_1 = x_2$ is only violated in a small subset of coordinates of the subproblem minima. The dual vector λ allows to reweight the functions associated with the duplicated variables during optimization in order to reduce violations of coupling constraints.

Decomposed graph. We will apply the Lagrange decomposition idea to problem (5). To this end, we first duplicate all binary variables and then regroup them. Each group corresponds to a new graph node. This leads to considerably less nodes. Although each node is associated with a non-binary variable, its minimal value can still be efficiently found. All coupling constraints turn into simple equalities as in the general scheme (6).

Graph structure. Our graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, see Figure 1 (c), contains only two types of nodes: detection and conflict nodes. The transition variables are duplicated (tripled for divisions) and their copies are assigned to the corresponding detection nodes. Since each detection corresponds to a large number of transitions, this significantly decreases the graph size. The detection variables are duplicated as well and their copies are assigned to the detection and conflict nodes. Below we give the formal definitions.

The set of graph nodes is defined as $\mathcal{V} = \mathcal{V}_{\mathsf{det}} \cup \mathcal{V}_{\mathsf{conf}}$, where $\mathcal{V}_{\mathsf{det}} := \hat{\mathcal{V}}_{\mathsf{det}}$ and $\mathcal{V}_{\mathsf{conf}} := \hat{\mathcal{E}}_{\mathsf{conf}}$, i. e. the detection nodes and conflict edges in the standard model correspond to detection and conflict nodes in our model. As in the standard model, $\mathcal{V}^t_{\mathsf{det}}$ and $\mathcal{V}^t_{\mathsf{conf}}$ denote the nodes at time step $t \in \{1, \dots, T\}$.

Each edge in the edge set $\mathcal{E} = \mathcal{E}_{\mathsf{move}} \cup \mathcal{E}_{\mathsf{div}} \cup \mathcal{E}_{\mathsf{conf}}$ corresponds to either a transition or conflict. The transition edges divide into $\mathcal{E}_{\mathsf{move}} := \hat{\mathcal{E}}_{\mathsf{move}}$, $\mathcal{E}_{\mathsf{move}} \subseteq (\mathcal{V}_{\mathsf{det}})^2$, corresponding to moves, and $\mathcal{E}_{\mathsf{div}} := \hat{\mathcal{E}}_{\mathsf{div}}$, $\mathcal{E}_{\mathsf{div}} \subseteq (\mathcal{V}_{\mathsf{det}})^3$, corresponding to divisions. As for the standard model, we will denote an edge $(u, v) \in \mathcal{E}_{\mathsf{move}}$ by $u \to v$, and an edge $(u, v, w) \in \mathcal{E}_{\mathsf{div}}$ by $u \rightrightarrows v/w$.

Conflict edges (u, c), denoted by $u \not\downarrow c$, are introduced between any detection node $u \in \mathcal{V}_{\mathsf{det}}$ and conflict node $c \in \mathcal{V}_{\mathsf{conf}}$ as soon as $\hat{u} \in \hat{c}$ for the corresponding $\hat{u} \in \hat{\mathcal{V}}_{\mathsf{det}}$ and $\hat{c} \in \hat{\mathcal{E}}_{\mathsf{conf}}$. Note, $\mathcal{E}_{\mathsf{conf}} \subseteq \mathcal{V}_{\mathsf{det}} \times \mathcal{V}_{\mathsf{conf}}$. When considering a conflict node $c \in \mathcal{V}_{\mathsf{conf}}$, we also use c to refer to all detection nodes that are part of the conflict, i. e. $u \in c$ if and only if $u \not\downarrow c \in \mathcal{E}_{\mathsf{conf}}$. Furthermore, for any $u \in \mathcal{V}_{\mathsf{det}}$ we define $\mathsf{conf}(u) := \{u' \not\downarrow c \in \mathcal{E}_{\mathsf{conf}} \mid u = u'\}$ as the set of all conflicts concerning u.

Detection variables. As noted above, nodes of the graph $\mathcal{G}$ correspond to variables having more than two states. These states are represented by binary vectors.

To define the state space of the detection variables we first introduce the sets $\mathsf{in}(u)$ and $\mathsf{out}(u)$, corresponding to $\hat{\mathsf{in}}(\hat{u})$ and $\hat{\mathsf{out}}(\hat{u})$ in $\hat{\mathcal{G}}$. For any detection node $u' \in \mathcal{V}_{\mathsf{det}}$ we denote by $\mathsf{in}(u')$ the set of all incoming transitions, i. e. $\mathsf{in}(u') := \{u \to v \in \mathcal{E}_{\mathsf{move}} \mid v = u'\} \cup \{u \rightrightarrows v/w \in \mathcal{E}_{\mathsf{div}} \mid v = u' \text{ or } w = u'\}$. Analogously, we define $\mathsf{out}(u') := \{u \to v \in \mathcal{E}_{\mathsf{move}} \mid u = u'\} \cup \{u \rightrightarrows v/w \in \mathcal{E}_{\mathsf{div}} \mid u = u'\}$ as the set of all outgoing transitions.

Consider $u \in \mathcal{V}_{\mathsf{det}}$. The set of states $\mathcal{X}_u$, which models whether the detection u is active, and if so, which incoming and outgoing edge is active, is defined as

$$\mathcal{X}_u = \left\{ \begin{pmatrix} x_{\mathsf{det}} \in \{0,1\} \\ x_{\mathsf{in}} \in \{0,1\}^{|\mathsf{in}(u)|} \\ x_{\mathsf{out}} \in \{0,1\}^{|\mathsf{out}(u)|} \end{pmatrix} \middle| \begin{array}{l} \langle \mathbb{1}, x_{\mathsf{in}} \rangle \leq x_{\mathsf{det}}, \\ \langle \mathbb{1}, x_{\mathsf{out}} \rangle \leq x_{\mathsf{det}} \end{array} \right\}. \quad (7)$$

The scalar products $\langle \mathbb{1}, x_{\mathsf{in}} \rangle$ and $\langle \mathbb{1}, x_{\mathsf{out}} \rangle$ express the number of activated incoming and outgoing transitions. Note that they can only be non-zero if the detection is active, i. e. if $x_{\mathsf{det}} = 1$. Below we use $x_{\mathsf{in}}(e)$ for any incoming edge $e \in \mathsf{in}(u)$ to refer to the value of this edge in the current state. Analogously, we use $x_{\mathsf{out}}(e)$ for edges $e \in \mathsf{out}(u)$. Recalling the standard model, for any $w \in \hat{\mathcal{V}}_{\mathsf{trans}}$ associated with $u \xrightarrow{w} v \in \hat{\mathcal{E}}_{\mathsf{move}}$ ($u \overset{w}{\rightrightarrows} v/v' \in \hat{\mathcal{E}}_{\mathsf{div}}$) the binary variable x_w is split into two (three) variables, one belonging to $\mathsf{out}(u)$ and another to $\mathsf{in}(v)$ (and $\mathsf{in}(v')$).

With each detection $u \in \mathcal{V}_{\mathsf{det}}$ we associate a cost vector $\theta_u = (\theta_{\mathsf{det}}, \theta_{\mathsf{in}}, \theta_{\mathsf{out}})$ consisting of the cost $\theta_{\mathsf{det}} \in \mathbb{R}$ for activating the detection, and costs $\theta_{\mathsf{in}} \in \mathbb{R}^{|\mathsf{in}(u)|}$

and $\theta_{\mathsf{out}} \in \mathbb{R}^{|\mathsf{out}(u)|}$ associated with the incoming and outgoing edges. $\theta_{\mathsf{det}} = \theta_{\hat{u}}$, where $\hat{u} \in \hat{\mathcal{V}}_{\mathsf{det}}$ is the corresponding detection node in the standard model. θ_{in} and θ_{out} are obtained by splitting the given transition costs between incoming and outgoing variables, i. e. $\theta_{\mathsf{in}}(e) = \frac{1}{|e|}\theta_{\hat{w}}$ for all $e \in \mathsf{in}(u)$, and, analogously, $\theta_{\mathsf{out}}(e) = \frac{1}{|e|}\theta_{\hat{w}}$ for all $e \in \mathsf{out}(u)$, where $\hat{w} \in \hat{\mathcal{V}}_{\mathsf{trans}}$ is the transition node corresponding to e in the standard model. So each admissible state $x = (x_{\mathsf{det}}, x_{\mathsf{in}}, x_{\mathsf{out}}) \in \mathcal{X}_u$ has a linear cost $\langle \theta_u, x\rangle = \langle \theta_{\mathsf{det}}, x_{\mathsf{det}}\rangle + \langle \theta_{\mathsf{in}}, x_{\mathsf{in}}\rangle + \langle \theta_{\mathsf{out}}, x_{\mathsf{out}}\rangle$.

Conflict variables. Let $c \in \mathcal{V}_{\mathsf{conf}}$. The associated set of states $\mathcal{X}_c$, that models which of the conflicting detections, if any, is active, can be written as

$$\mathcal{X}_c = \big\{x \in \{0,1\}^{|c|} \mid \langle \mathbb{1}, x\rangle \le 1\big\}. \tag{8}$$

For any detection node $u \in c$ we write $x(u)$ to refer to the value of this detection in the current state. With c we associate a cost vector $\theta_c \in \mathbb{R}^{|c|}$. These costs are initially zero, but may change during optimization. So each admissible state $x \in \mathcal{X}_c$ has a linear cost $\langle \theta_c, x\rangle$.

Coupling constraints. The semantics of each single move, division and detection activation is split between the states of multiple nodes in our problem graph. Consider a move $u \to v$ from detection node u to v. To obtain a consistent solution, we require moves to be consistent in u and v, i.e. $x_{u,\mathsf{out}}(u \to v) = x_{v,\mathsf{in}}(u \to v)$ for any feasible combination of states $x_u \in \mathcal{X}_u$, $x_v \in \mathcal{X}_v$. Analogous considerations for divisions and conflicts result in the following coupling constraints for $\mathcal{G}$:

$$\begin{aligned}
&\forall\, e = u \to v \in \mathcal{E}_{\mathsf{move}}: && x_{u,\mathsf{out}}(e) = x_{v,\mathsf{in}}(e),\\
&\forall\, e = u \rightrightarrows v/w \in \mathcal{E}_{\mathsf{div}}: && x_{u,\mathsf{out}}(e) = x_{v,\mathsf{in}}(e),\\
&\forall\, e = u \rightrightarrows v/w \in \mathcal{E}_{\mathsf{div}}: && x_{u,\mathsf{out}}(e) = x_{w,\mathsf{in}}(e),\\
&\forall\, u \not\downarrow c \in \mathcal{E}_{\mathsf{conf}}: && x_{u,\mathsf{det}} = x_c(u)\,.
\end{aligned} \tag{9}$$

The set of all state assignments satisfying all coupling constraints can then be written as

$$\mathcal{X} = \big\{\big(\{x_v \in \mathcal{X}_v\}_{v \in \mathcal{V}_{\mathsf{det}}}, \{x_c \in \mathcal{X}_c\}_{c \in \mathcal{V}_{\mathsf{conf}}}\big) \;\big|\; (9)\big\}.$$

Minimization problem. Our graph decomposition naturally gives rise to the minimization problem

$$\min_{x \in \mathcal{X}} \Big[E(\theta, x) := \sum_{u \in \mathcal{V}_{\mathsf{det}}} \langle \theta_u, x_u\rangle + \sum_{c \in \mathcal{V}_{\mathsf{conf}}} \langle \theta_c, x_c\rangle\Big]. \tag{10}$$

The goal is to find an optimal state assignment that satisfies all coupling constraints given costs θ.

Dualization of coupling constraints. Let us return to the general idea of the Lagrange decomposition (6). Assume $F_1(x) = \langle \theta_1, x\rangle$. Then $F_1(x) - \langle \lambda, x\rangle = \langle \theta_1 - \lambda, x\rangle$. Similarly, if $F_2(x) = \langle \theta_2, x\rangle$, then $F_2(x) + \langle \lambda, x\rangle = \langle \theta_2 + \lambda, x\rangle$. In other words, λ shifts the costs between parts of the decomposed problem. Since the value of the objective $F(x) = \langle \theta_1 - \lambda, x_1\rangle + \langle \theta_2 + \lambda, x_2\rangle$ remains the same for any value of λ if $x_1 = x_2$, this is also referred to as a *reparametrization* of the problem.

Dualizing all coupling constraints (9) in problem (10) results in the following reparametrized cost vectors:

Reparametrization. A *reparametrization* is a vector $\lambda \in \Lambda := \mathbb{R}^{|\mathcal{E}_{\mathsf{move}}| + 2|\mathcal{E}_{\mathsf{div}}| + |\mathcal{E}_{\mathsf{conf}}|}$. Its coordinates will be indexed with edges of the graph $\mathcal{G}$. That is, $\lambda(e) \in \mathbb{R}$ is the dual variable corresponding to the constraint $x_{u,\mathsf{in}}(e) = x_{v,\mathsf{out}}(e)$ if $e = u \to v \in \mathcal{E}_{\mathsf{move}}$, and $x_{u,\mathsf{det}} = x_c(u)$ if $e = u \not\downarrow c \in \mathcal{E}_{\mathsf{conf}}$. The only exception are divisions, since two constraints must be dualized for each $e = u \rightrightarrows v/w \in \mathcal{E}_{\mathsf{div}}$, namely $x_{u,\mathsf{out}}(e) = x_{v,\mathsf{in}}(e)$ and $x_{u,\mathsf{out}}(e) = x_{w,\mathsf{in}}(e)$, cf. (9). The corresponding dual variables are denoted as $\lambda_v(e)$ and $\lambda_w(e)$ respectively. The *reparametrized costs* θ^λ are defined as

$$\forall\, c \in \mathcal{V}_{\mathsf{conf}}\, \forall\, u \in c: \quad \theta^\lambda_c(u) := \theta_c(u) + \lambda(u \not\downarrow c),$$

$$\forall\, u \in \mathcal{V}_{\mathsf{det}}: \quad \theta^\lambda_{u,\mathsf{det}} := \theta_{u,\mathsf{det}} - \sum_{e \in \mathsf{conf}(u)} \lambda(e),$$

$$\forall\, u \in \mathcal{V}_{\mathsf{det}}\, \forall\, e \in \mathsf{in}(u): \quad \theta^\lambda_{u,\mathsf{in}}(e) := \begin{cases} \theta_{u,\mathsf{in}}(e) + \lambda(e), & e \in \mathcal{E}_{\mathsf{move}} \\ \theta_{u,\mathsf{in}}(e) + \lambda_u(e), & e \in \mathcal{E}_{\mathsf{div}} \end{cases},$$

$$\forall\, u \in \mathcal{V}_{\mathsf{det}}\, \forall\, e \in \mathsf{out}(u): \quad \theta^\lambda_{u,\mathsf{out}}(e) := \begin{cases} \theta_{u,\mathsf{out}}(e) - \lambda(e), & e \in \mathcal{E}_{\mathsf{move}} \\ \theta_{u,\mathsf{out}}(e) - \sum\limits_{v' \in \{v,w\}} \lambda_v(e), & e = u \rightrightarrows v/w \in \mathcal{E}_{\mathsf{div}} \end{cases}.$$

Each element of λ shifts the cost between two copies of variables in different subproblems coupled by an edge in $\mathcal{E}$. This shift does not influence the optimization objective as long as the coupling constraints (9) hold:

Proposition 1. $\forall x \in \mathcal{X},\ \lambda \in \Lambda\colon E(\theta, x) = E(\theta^\lambda, x)$.

Our dual is built similarly to the general scheme (6):

Proposition 2. *Dualizing all coupling constraints* (9) *in the objective* (10) *yields the* Lagrange dual problem $\max_{\lambda \in \Lambda} D(\lambda)$, *where*

$$D(\lambda) := \sum_{u \in \mathcal{V}_{\mathsf{det}}} \min_{x_u \in \mathcal{X}_u} \langle \theta^\lambda_u, x_u\rangle + \sum_{c \in \mathcal{V}_{\mathsf{conf}}} \min_{x_c \in \mathcal{X}_c} \langle \theta^\lambda_c, x_c\rangle\,. \tag{11}$$

Obviously, $D(\lambda)$ is concave and piecewise linear, i. e. non-smooth. By construction, $\max_{\lambda \in \Lambda} D(\lambda) \le \min_{x \in \mathcal{X}} E(\theta, x)$. The dual objective $D(\lambda)$ is a sum of small-sized minimization problems. Due to the structure of (7) and (8), each subproblem related to a graph node can be solved in linear time. As we show in the supplement, the maximization of the dual (11) yields the same value as the natural LP relaxation of (10).

4 Dual block-coordinate ascent (BCA)

In order to maximize (11), we developed an algorithm based on the BCA principle, as such methods perform competitively for similar relaxations of large-scale combinatorial problems. These techniques received a lot of attention in connection with the local polytope relax-

Algorithm 1: Dual optimization

$T' \leftarrow \{1, \dots, T\}$; $\lambda \leftarrow 0$
while *not converged* **do**
  for $t \in T'$ **do**
    for $v \in \mathcal{V}^t_{\mathsf{det}}$ **do**
      compute $\Delta^{\uparrow}_v$ on θ^λ ; $\lambda \leftarrow \lambda + \Delta^{\uparrow}_v$
    for $c \in \mathcal{V}^t_{\mathsf{conf}}$ **do**
      compute $\Delta^{\uparrow}_c$ on θ^λ; $\lambda \leftarrow \lambda + \Delta^{\uparrow}_c$
    estimate assignment (Alg. 2, optional)
    for $v \in \mathcal{V}^t_{\mathsf{det}}$ **do**
      compute $\Delta^{\rightarrow}_v$ on θ^λ ; $\lambda \leftarrow \lambda + \Delta^{\rightarrow}_v$
        (backward direction: use $\Delta^{\leftarrow}_v$)
  reverse the order of T'

ation of the discrete energy minimization problem [19]. BCA methods like TRW-S [21], SRMP [22] or the recently proposed DMM [31] and MPLP++ [35] notably outperform off-the-shelf solvers as well as dedicated subgradient-based methods. However, they are inapplicable in our case, due to a substantially different problem structure. The work [33] partially fills this gap by proposing a general framework for constructing dual BCA algorithms for a substantial subclass of combinatorial problems, which covers our problem as well. However, our experiments with the framework [33] did not lead to an improvement over Gurobi. Therefore, we constructed a new algorithm, which resembles the local polytope technique [21, 22, 28, 31, 35] and at the same time uses the results of [33] to guarantee monotonicity of the dual improvement.[1]

Dual BCA algorithm. Algorithm 1 is a realization of the BCA principle for the Lagrange dual (11) and guarantees its monotonous improvement. It contains four types of updates (also referred to as passing *messages*). The first two, $\Delta^{\uparrow}_u$ and $\Delta^{\uparrow}_c$, called *conflict updates*, reweight the detection variable costs $\theta_{u,\mathsf{det}}$ and conflict variable costs $\theta_c(u)$. The second two, $\Delta^{\rightarrow}_u$ and $\Delta^{\leftarrow}_u$, called *transition updates*, reweight costs $\theta_{u,\mathsf{in}}$ and $\theta_{v,\mathsf{out}}$ across consecutive time steps. All update vectors $\Delta^{\uparrow}_u$, $\Delta^{\uparrow}_c$, $\Delta^{\rightarrow}_u$ and $\Delta^{\leftarrow}_u$ are in Λ, with zero assigned to the unaffected coordinates. We process the time steps sequentially. We first perform all conflict updates within the current time step, then we propagate the costs to the next time step via transition updates.

Conflict updates. In the dual problem $D(\lambda)$ a detection will favor activation ($x_{\mathsf{det}}=1$) if its locally minimal state has negative cost. Ideally, only a single detection connected to a particular conflict node should be active. Otherwise, a coupling constraint on at least one of the edges in $\mathcal{E}_{\mathsf{conf}}$ or the constraint in $\mathcal{X}_c$ is violated, see (8), (9). The following updates encourage agreement of the local minimizers of the conflict nodes and the associated detection nodes. We define for all $u \in \mathcal{V}_{\mathsf{det}}$, $e \in \mathsf{conf}(u)$:

$$\Delta^{\uparrow}_u(e) := \min_{x \in \mathcal{X}_u \colon x_{\mathsf{det}}=1} \frac{\langle \theta_u, x \rangle}{|\mathsf{conf}(u)|}.$$

Intuitively, $\Delta^{\uparrow}_u$ redistributes as much of the cost as possible to the connected conflict nodes while preserving the locally optimal state in the detection node. Similarly, we define for all $c \in \mathcal{V}_{\mathsf{conf}}$, $u \in c$:

$$\Delta^{\uparrow}_c(e) := -\theta_c(u) + \frac{1}{2}\big[\langle \theta_c, z^\star_c \rangle + \langle \theta_c, z^{\star\star}_c \rangle\big],$$

where $e = u \not\downarrow c$, and $z^\star_c = \arg\min_{x \in \mathcal{X}_c} \langle \theta_c, x \rangle$ is the best and $z^{\star\star}_c = \arg\min_{x \in \mathcal{X}_c \setminus \{z^\star_c\}} \langle \theta_c, x \rangle$ the second-best state. $\Delta^{\uparrow}_c$ shifts the cost back such that only the most promising detection ends up with a negative activation cost.

Transition updates. To propagate information across time steps, we introduce the dual updates $\Delta^{\rightarrow}_u$ and $\Delta^{\leftarrow}_u$. We first define for all $u \in \mathcal{V}_{\mathsf{det}}$:

$$x^\star_u := \operatorname*{arg\,min}_{x \in \mathcal{X}_u \colon x_{\mathsf{det}}=1} \langle \theta_u, x \rangle, \qquad y^\star_u := \operatorname*{arg\,min}_{\substack{x \in \mathcal{X}_u \colon x_{\mathsf{det}}=1, \\ x_{\mathsf{in}} \neq x^\star_{u,\mathsf{in}},\ x_{\mathsf{out}} \neq x^\star_{u,\mathsf{out}}}} \langle \theta_u, x \rangle.$$

Intuitively, $x^\star_u$ is the best state of u under the assumption that u is active, while $y^\star_u$ is the best state differing from $x^\star_u$ in the incoming and outgoing edge.

Similar to the conflict updates, the transition updates propagate as much information as possible by setting all the outgoing respectively incoming costs to the same value. We define for all $u \in \mathcal{V}_{\mathsf{det}}$, $e \in \mathsf{out}(u)$:

$$\Delta^{\rightarrow}_u(e) := \min_{\substack{x \in \mathcal{X}_u \colon \\ x_{\mathsf{out}}(e)=1}} \langle \theta_u, x \rangle - \Theta_{u,\mathsf{out}}, \text{ if } e \in \mathcal{E}_{\mathsf{move}}$$

$$(\Delta^{\rightarrow}_u)_v(e) := \tfrac{1}{2}\Big[\min_{\substack{x \in \mathcal{X}_u \colon \\ x_{\mathsf{out}}(e)=1}} \langle \theta_u, x \rangle - \Theta_{u,\mathsf{out}}\Big], \text{ if } e = u \rightrightarrows v/w$$

where

$$\Theta_{u,\mathsf{out}} := \min\Big\{0, \tfrac{1}{2}\big[\langle \theta_u, x^\star_u \rangle + \langle \theta_u, (1, x^\star_{u,\mathsf{in}}, y^\star_{u,\mathsf{out}}) \rangle\big]\Big\},$$

is either 0 or the mean value between the cost of $x^\star_u$ and the next-best state with a different outgoing edge.

Similarly, for the updates in the opposite direction, we define for all $v \in \mathcal{V}_{\mathsf{det}}$, $e \in \mathsf{in}(v)$:

$$\Delta^{\leftarrow}_v(e) := -\min_{x \in \mathcal{X}_v, x_{\mathsf{in}}(e)=1} \langle \theta_v, x \rangle + \Theta_{v,\mathsf{in}}, \text{ if } e \in \mathcal{E}_{\mathsf{move}}. \quad (12)$$

Otherwise, if $e \in \mathcal{E}_{\mathsf{div}}$, $(\Delta^{\leftarrow}_v)_v(e)$ is assigned the right-hand-side of (12). Here

$$\Theta_{v,\mathsf{in}} := \min\Big\{0, \tfrac{1}{2}\big[\langle \theta_v, x^\star_v \rangle + \langle \theta_v, (1, y^\star_{v,\mathsf{in}}, x^\star_{v,\mathsf{out}}) \rangle\big]\Big\},$$

is either 0 or the mean value between the cost of $x^\star_v$ and the next-best state with a different incoming edge.

Proposition 3. *Dual updates* $\Delta \in \{\Delta^{\leftarrow}_u, \Delta^{\rightarrow}_u, \Delta^{\uparrow}_u \mid u \in \mathcal{V}_{\mathsf{det}}\} \cup \{\Delta^{\uparrow}_c \mid c \in \mathcal{V}_{\mathsf{conf}}\}$ *monotonically increase the dual function, i. e.* $\forall \lambda \in \Lambda \colon D(\lambda) \leq D(\lambda + \Delta)$.

Since the optimal dual value is bounded from above by the optimum of (5), the monotonicity implies con-

[1] In particular, our updates are *admissible* [33, Lem.1], but do not satisfy the maximality condition [33, eq.(15)].

Algorithm 2: Primal heuristics for time step t

$\mu^\star \leftarrow \underset{\mu\in\{0,1\}^{|\mathcal{V}^t_{\mathsf{det}}|}}{\arg\min} \langle s, \mu\rangle$ s.t. $\sum_{v\in c} \mu_v \le 1 \; \forall c \in \mathcal{V}^t_{\mathsf{conf}}$ (13)

for $v \in \mathcal{V}^t_{\mathsf{det}}$ **do**

  $x_v \leftarrow \mathsf{OFF}$ **if** $\mu^\star_v = 0$

order elements v of $\mathcal{V}^t_{\mathsf{det}}$ by their score $s(v)$

for $v \in \mathcal{V}^t_{\mathsf{det}}$ with $x_v \neq \mathsf{OFF}$ **do**

  $\mathcal{X}'_v \leftarrow \left\{ x \in \mathcal{X}_v \;\middle|\; \begin{array}{c} x \text{ does not violate} \\ \text{coupling constraints} \end{array} \right\}$

  assign $x_{v,\mathsf{in}}, x_{v,\mathsf{det}}$ according to $\arg\min_{x\in\mathcal{X}'_v} \langle \theta^\lambda_v, x\rangle$
  (backward direction: use $x_{v,\mathsf{out}}$ instead)

  propagate state x_v accross edges in $\mathcal{E}$

vergence of the sequence of dual values. The limit value of the sequence, though, need not be the dual optimum. This is a well-known property of block-coordinate-ascent methods, which may get stuck when applied to non-smooth functions.

5 Primal heuristics

For solving the tracking problem (10) computing a reparametrization λ is not enough. The main goal is to obtain a feasible primal assignment $x \in \mathcal{X}$ corresponding to a low objective value. Generally, this is non-trivial, since solutions to the node subproblems are usually inconsistent, even if an optimal dual solution $\lambda^\star = \arg\max_{\lambda'\in\Lambda} D(\lambda')$ is considered. As mentioned in Section 1, existing techniques either do not handle overlapping segmentation hypotheses or do not allow for cell divisions. Below, we propose a new primal heuristics that handles the considered general case and produces high-quality feasible primal assignments even for non-optimal dual vectors λ.

Temporal direction. Similar to the dual optimization algorithm our primal heuristics works in both temporal directions. For the sake of simplicity we restrict ourselves to the explanation of the forward direction. After obtaining the primal assignments for each direction, we keep the best of the two.

Incremental estimates. We estimate primal solution for each time frame sequentially one by one, starting from $t = 0$ and ending with $t = T$. Hence, we assume that primal assignments $x_u \in \mathcal{X}_u$ for $u \in \mathcal{V}^{t'}_{\mathsf{det}}$, $t' \in \{1, \dots, t-1\}$, are available when applying Algorithm 2 for time step t.

Conflict resolution. Before we look at edges that connect different time steps, we first resolve the conflicts within the current time step t. Overall, we want to activate the most promising detection hypotheses while still obeying all coupling constraints in $\mathcal{E}_{\mathsf{conf}}$. We score the individual detections $v \in \mathcal{V}_{\mathsf{det}}$ by their locally cost-optimal state conditioned on actually activating v, i.e. $s(v) := \min_{x\in\mathcal{X}_v \,:\, x_{\mathsf{det}}=1} \langle \theta^\lambda_v, x\rangle$. To resolve the conflicts of the current time step t, we solve with Gurobi the *weighted set packing problem* of all detections in t:

$$\min_{\mu\in\{0,1\}^{|\mathcal{V}^t_{\mathsf{det}}|}} \langle s, \mu\rangle \quad \text{s.t.} \quad \sum_{v\in c} \mu_v \le 1 \quad \forall c \in \mathcal{V}^t_{\mathsf{conf}} \,, \quad (13)$$

where $s = (s(v))_{v\in\mathcal{V}^t_{\mathsf{det}}}$. Note that even though the problem (13) is NP-hard, its encountered instances are small and can be solved almost instantly. In comparison to cheaper approaches, using (13) yields significantly better tracking solutions.

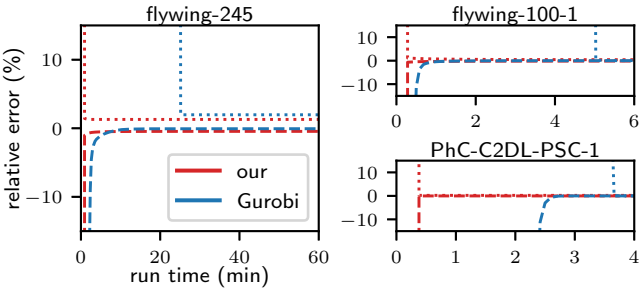


Figure 2: Lower- (==) and upper-bound (::::) for our solver and Gurobi on selected instances. We obtain high-quality solutions after only a few iterations.

Transition assignments. After resolving all conflicts for the current time step t, we look for a consistent assignment for $x_{v,\mathsf{in}}$ for all $v \in \mathcal{V}^t_{\mathsf{det}}$ with $\mu^\star_v = 1$, where $\mu^\star$ is a solution of (13). Ideally, we would like to set them to a locally optimal state given their current cost θ^λ_v. Unfortunately, this will in general violate coupling constraints in $\mathcal{E}^{t-1}_{\mathsf{move}}$ and $\mathcal{E}^{t-1}_{\mathsf{div}}$. Therefore, we use a greedy approach and process all possible nodes, i. e. all $v \in \mathcal{V}^t_{\mathsf{det}}$ with $\mu^\star_v = 1$, sequentially. While optimizing $x_{v,\mathsf{in}}$ for a given node v we ignore all options that would violate coupling constraints with already fixed nodes. Obviously, such a procedure heavily depends on the node ordering. As we want to address the most promising states first, we order all nodes by their score $s(v)$ starting with the state with the best, i. e. lowest, score.

Propagation. When $x_{v,\mathsf{in}}$ is assigned we propagate this across incident coupling constraints in $\mathcal{E}^{t-1}_{\mathsf{move}} \cup \mathcal{E}^{t-1}_{\mathsf{div}}$. Eventually, this results in all variables $x_{u,\mathsf{out}}$ being set for all nodes $u \in \mathcal{V}^{t-1}_{\mathsf{det}}$ of the previous time step. This procedure leads to a consistent variable assignment up to and including time step t.

Continuing the procedure for all time steps leads to a consistent variable assignment for the whole problem. As the quality of the assignment is expected to improve with improvement of the dual, we run primal heuristics repeatedly each 25 dual iterations.

6 Experimental evaluation

We evaluate the performance of our solver on cell and nucleus tracking datasets from different biomedical re-

Instance	Our Solver				Gurobi				Improvement	
	time (s)	mem. (MiB)	err. (%)	TRA	time relax. (s)	/ equ. (s)	/ opt. (s)	mem. (MiB)	time	mem.
drosophila	**2.7**	**157**	0.00	0.9999	0.9	9.2	9.3	1154	**3.4**×	**7.3**×
flywing-100-1	**82.4**	**474**	0.65	0.9867	270.6	301.8	554.9	5522	**3.7**×	**11.6**×
flywing-100-2	**78.5**	**490**	0.98	0.9826	243.0	275.2	2311.5	8527	**3.5**×	**17.4**×
flywing-245	**159.8**	**1192**	1.29[1]	—[2]	1159.0	9540.6	—[3]	20756	**59.7**×	**17.4**×
Fluo-C2DL-MSC-1	1.9	**53**	0.38	0.9922	0.0	**0.5**	**0.6**	90	0.3×	**1.7**×
Fluo-C2DL-MSC-2	2.0	**50**	0.08	0.9863	0.0	**0.1**	**0.1**	61	0.1×	**1.2**×
Fluo-N2DH-GOWT1-1	**0.5**	**58**	0.00	1.0000	0.0	0.6	0.6	153	**1.2**×	**2.6**×
Fluo-N2DH-GOWT1-2	**0.3**	**65**	0.00	1.0000	0.0	1.1	1.1	196	**3.3**×	**3.0**×
PhC-C2DL-PSC-1	**22.7**	**930**	0.23	0.9952	40.9	219.7	267.6	13199	**9.7**×	**14.2**×
PhC-C2DL-PSC-2	**17.9**	**708**	0.14	0.9969	22.2	127.5	156.9	9870	**7.1**×	**13.9**×

[1]opt. unknown, shows rel. primal/dual-diff. instead [2]opt. unknown, no reference available [3]did not terminate within 8 h

Table 1: Quantitative comparison of our solver and the ILP solver Gurobi. We display run time, maximal memory consumption (mem.), relative error of (5) compared to optimum (err.) and TRA score. For Gurobi time of root relaxation (relax.), finding a comparable solution (equ.) and finding an optimal solution (opt.) is reported.

search projects. As our contributions are exclusively concerned with the optimization of the problem instances, we are not discussing any aspects of potential model mismatch. We used reasonable segmentations obtained by different segmentation routines and selected appropriate costs for the different tracking events without excessive fine-tuning (see supplement). Existing methods that allow for overlapping segmentation hypotheses and cell divisions [13, 14, 17, 30] offload the optimization to off-the-shelf ILP solvers like Gurobi [9], which we use as baseline for our comparison.

Datasets and tracking instances. In total, we use 10 problem instances from 3 biomedical data domains, see supplement for additional information.

Drosophila embryo data: We have one problem instance for tracking nuclei in a developing *Drosophila* embryo. The tracking model consists of 252 frames, each containing about 320 detection hypotheses. With about 160 actual objects per time step, we typically observe two conflicting hypotheses per real object in the data.

Flywing data: We use 3 problem instances for tracking membrane-labelled cells in developing *Drosophila* flywing tissue. Two of these consist of 100 frames, each containing about 2 100 detection hypotheses. The third instance is larger, consisting of 245 frames with more than 3 300 detection hypotheses each. In contrast to the embryo data, the segmentation hypotheses in these problem instances are very dense, leading to considerably larger sets of (transitively) conflicting detections.

Cell Tracking Challenge (CTC) data: Finally, we use the publicly available cell tracking datasets *Fluo-C2DL-MSC*, *Fluo-N2DH-GOWT1*, and *PhC-C2DL-PSC* [36] to evaluate our solver. The CTC data consists of 48, 92, and 426 frames, where each frame contains on average 88, 186, and 1 400 detection hypotheses, respectively, with conflict set sizes of about 10, 7, and 3. Each dataset consists of two time-lapse movies, allowing us to generate six tracking instances.

Evaluation criteria. We evaluate the performance using two metrics, namely, total *run time* and overall *memory consumption*. To ensure a fair run time comparison between our solver and Gurobi, we measure not only the time it takes Gurobi to compute the optimal solution, *time (opt.)*, but also the time Gurobi needs to surpass the quality of our best primal solution, *time (equ.)*. Additionally, we compute the *relative error* $\frac{|E(\theta,x)-E(\theta,x^\star)|}{|E(\theta,x^\star)|}$ of our final assignment x with respect to the optimal solution $x^\star$. We finally also compute the *TRA* [36] score, a commonly used tracking scoring function. TRA values are in $[0, 1]$, where 1 means that the final assignment is identical to the reference solution, while 0 occurs if the compared solutions have nothing in common. We use TRA to compare our tracking results to the optimal solution obtained with Gurobi. Times show the median results of 5 single-threaded runs on an Intel i7-4770 3.40GHz CPU.

Results and conclusions. In Table 1 we show the results for our solver and Gurobi on all instances. Especially for larger problems, i. e. the *PhC-C2DL-PSC* and *flywing* instances, our solver obtains near-optimal solutions in a fraction of the time (sped-up by factor between 3 and 60). Due to the compact decomposition, our solver consistently requires considerably less memory than Gurobi (up to 17x). Figure 2 shows that our solver converges to small relative errors after only a few iterations. The high quality of our solutions is confirmed by the TRA scores. Compared to existing techniques our solver is scalable and quickly provides high quality solutions even for large-scale real-world problems, so far beeing practically intractable. Therefore it is even applicable in low-latency settings like user-driven proofreading of automated tracking results.

Acknowledgements. We thank the Centre for Information Services and High Performance Computing (ZIH) at TU Dresden for generous allocation of HPC resources (project HPDLF). This work was

supported partly by the German Reserach Foundation (DFG SA 2640/1-1, "ERBI"), the European Reserach Council (ERC European Unions Horizon 2020 research and innovation program, grant 647769) and the German Federal Ministry of Research and Education (BMBF, code 01IS18026C, "ScaDS2").

References

[1] Assaf Arbelle and Tammy Riklin Raviv. Microscopy cell segmentation via convolutional LSTM networks. *arXiv:1805.11247*, 2018.

[2] Asad A. Butt and Robert T. Collins. Multi-target tracking by Lagrangian relaxation to min-cost network flow. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[3] Gregory Castanon and Lucas Finn. Multi-target tracklet stitching through network flows. In *IEEE Aerospace Conference*, 2011.

[4] Nicolas Chenouard, Ihor Smal, Fabrice de Chaumont, Martin Maška, Ivo F. Sbalzarini, Yuanhao Gong, Janick Cardinale, Craig Carthel, Stefano Coraluppi, Mark Winter, Andrew R. Cohen, William J. Godinez, Karl Rohr, Yannis Kalaidzidis, Liang Liang, James Duncan, Hongying Shen, Yingke Xu, Klas E.G. Magnusson, Joakim Jaldén, Helen M. Blau, Perrine Paul-Gilloteaux, Philippe Roudot, Charles Kervrann, François Waharte, Jean-Yves Tinevez, Spencer L. Shorte, Joost Willemse, Katherine Celler, Gilles P. van Wezel, Han-Wei Dan, Yuh-Show Tsai, Carlos Ortiz-de Solorzano, Jean-Christophe Olivo-Marin, and Erik Meijering. Objective comparison of particle tracking methods. *Nature Methods*, 11(3), 2014.

[5] IBM ILOG CPLEX. V12.8: User's manual for CPLEX. 2017.

[6] Monique Guignard. Lagrangean relaxation. *Top*, 11(2), 2003.

[7] Monique Guignard and Siwhan Kim. Lagrangean decomposition: A model yielding stronger lagrangean bounds. *Mathematical programming*, 39 (2), 1987.

[8] Monique Guignard and Siwhan Kim. Lagrangean decomposition for integer programming: theory and applications. *RAIRO-Operations Research*, 21 (4), 1987.

[9] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018. URL http://www.gurobi.com.

[10] Carsten Haubold, Janez Aleš, Steffen Wolf, and Fred A. Hamprecht. A generalized successive shortest paths solver for tracking dividing targets. In *European Conference on Computer Vision*. Springer, 2016.

[11] Umar Iqbal, Anton Milan, and Juergen Gall. Posetrack: Joint multi-person pose estimation and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[12] Khuloud Jaqaman, Dinah Loerke, Marcel Mettlen, Hirotaka Kuwata, Sergio Grinstein, Sandra L. Schmid, and Gaudenz Danuser. Robust single-particle tracking in live-cell time-lapse sequences. *Nature Methods*, 5(8), 2008.

[13] Florian Jug, Tobias Pietzsch, Dagmar Kainmüller, Jan Funke, Matthias Kaiser, Erik van Nimwegen, Carsten Rother, and Gene Myers. Optimal joint segmentation and tracking of Escherichia coli in the mother machine. In *Bayesian and Graphical Models for Biomedical Imaging*. Springer, 2014.

[14] Florian Jug, Tobias Pietzsch, Dagmar Kainmüller, and Gene Myers. Tracking by assignment facilitates data curation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention, IMIC Workshop*, 2014.

[15] Florian Jug, Tobias Pietzsch, Stephan Preibisch, and Pavel Tomancak. Bioimage informatics in the context of drosophila research. *Methods*, 68(1), 2014.

[16] Florian Jug, Evgeny Levinkov, Corinna Blasse, Eugene W. Myers, and Bjoern Andres. Moral lineage tracing. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[17] Matthias Kaiser, Florian Jug, Thomas Julou, Siddharth Deshpande, Thomas Pfohl, Olin K. Silander, Gene Myers, and Erik van Nimwegen. Monitoring single-cell gene regulation under dynamically controllable conditions with integrated microfluidics and software. *Nature Communications*, 9(1), 2018.

[18] Lee Kamentsky, Thouis R. Jones, Adam Fraser, Mark-Anthony Bray, David J. Logan, Katherine L. Madden, Vebjorn Ljosa, Curtis Rueden, Kevin W. Eliceiri, and Anne E. Carpenter. Improved structure, function and compatibility for CellProfiler: modular high-throughput image analysis software. *Bioinformatics*, 27(8), 2011.

[19] Jörg H. Kappes, Bjoern Andres, Fred A. Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X. Kausler, Thorben Kröger, Jan Lellmann, Nikos Komodakis, Bogdan Savchynskyy, and Carsten Rother. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision*, 2015.

[20] Bernhard X. Kausler, Martin Schiegg, Bjoern Andres, Martin Lindner, Ullrich Koethe, Heike Leitte, Jochen Wittbrodt, Lars Hufnagel, and Fred A.

Hamprecht. A discrete chain graph model for 3d+ t cell tracking with high misdetection robustness. In *European Conference on Computer Vision*. Springer, 2012.

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

[22] Vladimir Kolmogorov. A new look at reweighted message passing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(5), 2015.

[23] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Cehovin Zajc, Tomas Vojir, Gustav Hager, Alan Lukezic, Abdelrahman Eldesokey, et al. The visual object tracking vot2017 challenge results. In *IEEE International Conference on Computer Vision*, 2017.

[24] Xinghua Lou, Frederik O. Kaster, Martin S. Lindner, Bernhard X. Kausler, Ullrich Köthe, Burkhard Höckendorf, Jochen Wittbrodt, Heike Jänicke, and Fred A. Hamprecht. Deltr: Digital embryo lineage tree reconstructor. In *IEEE International Symposium on Biomedical Imaging*, 2011.

[25] Klas E.G. Magnusson, Joakim Jaldén, Penney M. Gilbert, and Helen M. Blau. Global linking of cell tracks using the viterbi algorithm. *IEEE Transactions on Medical Imaging*, 34(4), 2015.

[26] Dirk Padfield, Jens Rittscher, and Badrinath Roysam. Coupled minimum-cost flow cell tracking. In *International Conference on Information Processing in Medical Imaging*. Springer, 2009.

[27] Christian Payer, Darko Štern, Thomas Neff, Horst Bischof, and Martin Urschler. Instance segmentation and tracking with cosine embeddings and recurrent hourglass networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018.

[28] Bogdan Savchynskyy. Discrete graphical models — an optimization perspective. *Foundations and Trends in Computer Graphics and Vision*, 11(3-4), 2019.

[29] Martin Schiegg, Philipp Hanslovsky, Bernhard X. Kausler, Lars Hufnagel, and Fred A. Hamprecht. Conservation tracking. In *IEEE International Conference on Computer Vision*, 2013.

[30] Martin Schiegg, Philipp Hanslovsky, Carsten Haubold, Ullrich Köthe, Lars Hufnagel, and Fred A. Hamprecht. Graphical model for joint segmentation and tracking of multiple dividing cells. *Bioinformatics*, 31(6), 2014.

[31] Alexander Shekhovtsov, Christian Reinbacher, Gottfried Graber, and Thomas Pock. Solving dense image matching in real-time using discrete-continuous optimization. In *21st Computer Vision Winter Workshop*, 2016.

[32] David Sontag, Amir Globerson, and Tommi Jaakkola. Introduction to dual composition for inference. In *Optimization for Machine Learning*. MIT Press, 2011.

[33] Paul Swoboda, Jan Kuske, and Bogdan Savchynskyy. A dual ascent framework for lagrangean decomposition of combinatorial problems. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[34] Siyu Tang, Bjoern Andres, Miykhaylo Andriluka, and Bernt Schiele. Subgraph decomposition for multi-target tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[35] Siddharth Tourani, Alexander Shekhovtsov, Carsten Rother, and Bogdan Savchynskyy. MPLP++: Fast, parallel dual block-coordinate ascent for dense graphical models. In *European Conference on Computer Vision*, 2018.

[36] Vladimír Ulman, Martin Maška, Klas E.G. Magnusson, Olaf Ronneberger, Carsten Haubold, Nathalie Harder, Pavel Matula, Petr Matula, David Svoboda, Miroslav Radojevic, Ihor Smal, Karl Rohr, Joakim Jaldén, Helen M. Blau, Oleh Dzyubachyk, Boudewijn Lelieveldt, Pengdong Xiao, Yuexiang Li, Siu-Yeung Cho, Alexandre C. Dufour, Jean-Christophe Olivo-Marin, Constantino C. Reyes-Aldasoro, Jose A. Solis-Lemus, Robert Bensch, Thomas Brox, Johannes Stegmaier, Ralf Mikut, Steffen Wolf, Fred A. Hamprecht, Tiago Esteves, Pedro Quelhas, Ömer Demirel, Lars Malmström, Florian Jug, Pavel Tomancak, Erik Meijering, Arrate Muñoz-Barrutia, Michal Kozubek, and Carlos Ortiz-de Solorzano. An objective comparison of cell-tracking algorithms. *Nature Methods*, 14(12), 2017.

[37] Xinchao Wang, Engin Türetken, Francois Fleuret, and Pascal Fua. Tracking interacting objects optimally using integer programming. In *European Conference on Computer Vision*. Springer, 2014.

[38] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9), 2015.

[39] Junliang Xing, Haizhou Ai, and Shihong Lao. Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.