A. Lagrangean decomposition

A.1. Derivation of the dual

We first introduce vectors $y \in \{0,1\}^n$ as well as $x^j \in \mathcal{X}_j$ for $j \in [m]$ and rewrite (BP) redundantly as

$$\min_{y, x^1, \ldots, x^m} \quad c^\top y \quad \text{s.t.} \quad y_{\mathcal{I}_j} = x^j, \; x^j \in \mathcal{X}_j \quad \forall j \in [m]. \tag{13}$$

Now, let $\mathcal{J}_i = \{j \in [m] \mid i \in \mathcal{I}_j\}$ denote the set of variable indices constrained by $\mathcal{X}_j$. For $i \in [n]$ and $j \in \mathcal{J}_i$ we introduce dual variables λ_i^j associated with the equality constraint $y_i = x_i^j$. For each set of Lagrange variables λ we obtain a lower bound to the original problem (13) given by

$$\begin{aligned} \min_{y, x^1, \ldots, x^m} & \sum_{i \in [n]} \Big(c_i - \sum_{j \in \mathcal{J}_i} \lambda_i^j\Big) y_i + \sum_j {\lambda^j}^\top x^j \\ \text{s.t.} & \quad y \in \{0,1\}^n, \quad x^j \in \mathcal{X}_j \quad \forall j \in [m] \end{aligned} \tag{14}$$

Optimization above can be decoupled for each x^j, $j \in [m]$. and maximizing over λ gives the Lagrangean dual

$$\max_\lambda \min_{y \in \{0,1\}^n} \sum_{i \in [n]} \Big(c_i - \sum_{j \in \mathcal{J}_i} \lambda_i^j\Big) y_i + \sum_j \min_{x \in \mathcal{X}_j} x^\top \lambda^j. \tag{15}$$

For simplification we can eliminate y from the formulation by observing that (w.l.o.g.) the maximum is attained for some λ that satisfies $\sum_{j \in \mathcal{J}_i} \lambda_i^j = c_i$ for all $i \in [n]$. Hence, the simplified dual problem reads

$$\max_\lambda \quad \sum_j \min_{x \in \mathcal{X}_j} x^\top \lambda^j \quad \text{s.t.} \quad \sum_{j \in \mathcal{J}_i} \lambda_i^j = c_i \quad \forall i \in [n]. \tag{D}$$

A.2. Min-marginal averaging

Proof of Proposition 1

Proof. Let $\bar{\lambda}_i^j = \lambda_i^j - (m_{ij}^1 - m_{ij}^0)$. Then

$$E^j(\bar{\lambda}^j) = \begin{cases} E^j(\lambda^j) - (m_{ij}^1 - m_{ij}^0) & \text{if } m_{ij}^1 - m_{ij}^0 < 0 \\ E^j(\lambda^j) & \text{else.} \end{cases} \tag{16}$$

Moreover, the min-marginal differences w.r.t. $\bar{\lambda}$ vanish. Now, let $\bar{\bar{\lambda}}_i^j = \bar{\lambda}_i^j + \frac{1}{|\mathcal{J}_i|} \sum_{k \in \mathcal{J}_i} m_{ik}^1 - m_{ik}^0$. Then

$$E^j(\bar{\bar{\lambda}}^j) = E^j(\bar{\lambda}^j) + \frac{1}{|\mathcal{J}_i|} \sum_{k \in \mathcal{J}_i} m_{ik}^1 - m_{ik}^0 \tag{17}$$

if $\frac{1}{|\mathcal{J}_i|} \sum_{k \in \mathcal{J}_i} m_{ik}^1 - m_{ik}^0 < 0$ and

$$E^j(\bar{\bar{\lambda}}^j) = E^j(\bar{\lambda}^j) \tag{18}$$

otherwise. Hence, the dual bound increases in total by

$$\begin{aligned} & \sum_{j \in \mathcal{J}_i} E^j(\bar{\bar{\lambda}}^j) - E^j(\lambda^j) \\ &= - \sum_{\{k \in \mathcal{J}_i \mid m_{ik}^1 - m_{ik}^0 < 0\}} (m_{ik}^1 - m_{ik}^0) \\ &\quad + \min\Big\{0, \sum_{k \in \mathcal{J}_i} m_{ik}^1 - m_{ik}^0\Big\}. \end{aligned} \tag{19}$$

□

A.3. Variable order

The order in which we process the variable indices $i \in [n]$ in Algorithm 1 should facilitate the increase of the dual bound in each iteration. Therefore, we prefer to process indices $i, i' \in [n]$ consecutively if their updates influence the min-marginals of the associated primal variables x_i and $x_{i'}$, which is the case if there is a subproblem that contains both variables. A suitable variable order can be obtained by searching for a permutation of the constraint matrix with lowest bandwidth. The bandwidth of a matrix is the width of the smallest band around the diagonal such that all non-zero entries are contained in it. Bandwidth-minimization is NP-hard (Papadimitriou, 1976), but fast heuristics such as the Cuthill-McKee algorithm (Cuthill & McKee, 1969) are available. We run the algorithm on the bipartite variable-constraint adjacency matrix and extract the variable order from the result.

A.4. Averaging strategy

The min-marginal averaging update w.r.t. $i \in [n]$ defined in (2) subtracts the min-marginal difference from each corresponding dual variable and distributes the sum of min-marginal differences evenly to all dual variables associated with i. In the case of higher-order graphical models an alternative averaging strategy called Sequential Reweighted Message Passing (SRMP) (Kolmogorov, 2014) was shown to improve the convergence behavior of the associated DBCA algorithm. In close analogy to SRMP we suggest the following averaging scheme as an alternative to the default update. For $i \in [n]$ let

$$\mathcal{J}_i^> = \{j \in \mathcal{J}_i \mid \exists i' > i \text{ such that } i' \in \mathcal{I}_j\} \tag{20}$$

denote the subproblem indices that contain any variable with index greater than i, and define $\mathcal{J}_i^<$ similarly. The sets $\mathcal{J}_i^>$ and $\mathcal{J}_i^>$ are defined here w.r.t. the default order on $[n]$ for the sake of simplicity, but can be defined for any other variable order in an analogous way. The SRMP averaging update distributes the sum of min-marginal differences evenly to all λ_i^j for $j \in \mathcal{J}_i^>$ in the forward pass. If $\mathcal{J}_i^> = \emptyset$, we fall back to the default averaging scheme by setting $\mathcal{J}_i^> = \mathcal{J}_i$. More precisely, if $\mathcal{J}_i^> \neq \emptyset$ during the forward pass, the update (2) is replaced by

$$\lambda_i^j \leftarrow \lambda_i^j - (m_{ij}^1 - m_{ij}^0) + \frac{1}{|\mathcal{J}_i^>|} \sum_{k \in \mathcal{J}_i} m_{ik}^1 - m_{ik}^0 \quad (21)$$

for $i \in \mathcal{J}_i^>$ and

$$\lambda_i^j \leftarrow \lambda_i^j - (m_{ij}^1 - m_{ij}^0) \quad (22)$$

for $i \in \mathcal{J}_i \setminus \mathcal{J}_i^>$. The backward pass is performed similarly with $\mathcal{J}_i^<$ instead of $\mathcal{J}_i^>$.

A.5. Smoothing

It is well-known that, except in special cases (Dlask & Werner, 2020), DBCA can fail to reach the optimum of the relaxation. Suboptimal stationary points of DBCA algorithms are analyzed in (Werner et al., 2020). One way to attain optima of Lagrangean relaxations with DBCA algorithms is to replace the original non-smooth dual objective with a smooth approximation on which DBCA is guaranteed to find the optimum. We propose such a smooth approximation below for our Lagrangean decomposition (D) and detail according update rules. Analogous techniques were used in (Meltzer et al., 2012) to smooth the TRWS algorithm (Kolmogorov, 2006).

First, we replace the original energy $E^j(\lambda^j)$ through a log-sum-exp based approximation. For any smoothing parameter $\alpha > 0$ we define

$$E_\alpha^j(\lambda^j) = -\alpha \cdot \log\Big(\sum_{x \in \mathcal{X}_j} \exp\Big(\frac{-x^\top \lambda^j}{\alpha}\Big)\Big). \quad (23)$$

This results in the smooth Lagrangean dual $\max_\lambda \sum_{j \in [m]} E_\alpha^j(\lambda^j)$. Further, instead of min-marginals m_{ij}^β we define marginal log-sum-exp values as

$$m_{ij}^{\alpha,\beta} = -\alpha \cdot \log\Big(\sum_{x \in \mathcal{X}_j : x_i = \beta} \exp\Big(\frac{-x^\top \lambda^j}{\alpha}\Big)\Big). \quad (24)$$

Finally, the min-marginal averaging operation (2) is replaced by

$$\lambda_i^j \leftarrow \lambda_i^j - (m_{ij}^{\alpha,1} - m_{ij}^{\alpha,0}) + \frac{1}{|\mathcal{J}_i|} \sum_{k \in \mathcal{J}_i} m_{ik}^{\alpha,1} - m_{ik}^{\alpha,0}. \quad (25)$$

With this change of operations, Algorithm 1 becomes a DBCA method for the smooth approximation.

Proposition 5 (Approximation guarantees)**.** *For any $\alpha > 0$ and $j \in [m]$ it holds that*

$$E^j(\lambda^j) > E_\alpha^j(\lambda^j) \geq E^j(\lambda^j) - \alpha \log |\mathcal{X}_j|. \quad (26)$$

Proof. It holds that

$$\begin{aligned} E^j(\lambda^j) &= \min_{x \in \mathcal{X}_j} x^\top \lambda^j & (27)\\ &\geq -\alpha \log\Big(\exp\Big(-\min_{x \in \mathcal{X}_j} \frac{x^\top \lambda^j}{\alpha}\Big)\Big) & (28)\\ &\geq -\alpha \log\Big(\sum_{x \in \mathcal{X}_j} \exp\Big(-\frac{x^\top \lambda^j}{\alpha}\Big)\Big) = E_\alpha^j(\lambda^j) & (29)\\ &\geq -\alpha \log\Big(|\mathcal{X}_j| \cdot \exp\Big(-\min_{x \in \mathcal{X}_j} \frac{x^\top \lambda^j}{\alpha}\Big)\Big) & (30)\\ &= E^j(\lambda^j) - \alpha \log(|\mathcal{X}_j|). \end{aligned}$$

□

B. Primal heuristic

B.1. Search strategies

Another indicator of how suitable a variable/value pair is for fixation is the reduction of the number of feasible solutions when a given variable is fixed to some value.

Definition 10 (Search space reduction coefficient)**.** For $i \in [n]$ we define the *search space reduction coefficient* as

$$R_i = \sum_{j \in \mathcal{J}_i} |\{x \in \mathcal{X}_j \mid x_i = 1\}| - |\{x \in \mathcal{X}_j \mid x_i = 0\}|. \quad (31)$$

The quantity R_i indicates the difference in search space reduction between the fixation $x_i = 1$ and $x_i = 0$ across all associated subproblems. As an alternative choice for the variable scores S_i that determine the order of variable fixations, we propose $S_i = \text{sign}(R_i) M_i$. The resulting strategy prefers those variables for which the signs of R_i and M_i agree, thus aligning search space reduction in the individual subproblems with the min-marginal evidence.

C. Implementation with BDDs

Proof of Proposition 2

Proof. See (Bryant, 1986).The required changes due to insertion of BDD nodes with equal outgoing edges do not change the proofs. □

Proof of Proposition 3

Proof. First, we note that (6) returns correct marginals for variable i_ℓ if we have performed Algorithm 8 for variables $i_1, \ldots, i_{\ell-1}$ in that order and Algorithm 8 for variables $i_k, \ldots, i_{\ell+1}$ in that order. The reason is that Algorithms 7 and 8 are performing dynamic programming steps for the respective problem, i.e. for shortest path with the $(+, \min)$-algebra. When processing variable i in the forward pass of

Algorithm 5: Primal-Heuristic($\mathcal{I}, (\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}, (S_i)_{i\in[n]}$)

```
   Input: Open variable indices $\mathcal{I} \subset [n]$, restricted subproblems and indices $(\mathcal{X}_j, \mathcal{I}_j)$, $j \in [m]$, scores $(S_i)_{i\in\mathcal{I}}$
1  Find variable $i \in \mathcal{I}$ with maximum score $S_i$
2  (feasible, $\mathcal{I}'$, $(\mathcal{X}'_j, \mathcal{I}'_j)_{j\in[m]}$) = Restriction-Propagation($(\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}, (i, \beta)$)
3  if feasible and $\mathcal{I}' = \varnothing$ then
4      return solution
5  else if feasible then
6      feasible = Primal-Heuristic($\mathcal{I}', (\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}, (S_i)_{i\in\mathcal{I}'}$)
7  else if not feasible then
8      (feasible, $\mathcal{I}'$, $(\mathcal{X}'_j, \mathcal{I}'_j)_{j\in[m]}$) = Restriction-Propagation($(\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}, (i, 1-\beta)$)
9      if feasible then
10         return Primal-Heuristic($\mathcal{I}', (\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}, (S_i)_{i\in\mathcal{I}'}$)
11     else
12         return infeasible
13 output Partial solution to current subproblem or infeasible
```

Algorithm 6: Restriction-Propagation($(\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}, (i, \beta)$)

```
   Input: Subproblems $\mathcal{X}_j$, indices $\mathcal{I}_j \subset [n]$, $j \in [m]$,
          index/value pair to fix $(i, \beta)$.
1  for $j \in \mathcal{J}_i$ do
2      $\mathcal{X}_j^\beta = \{x \in \mathcal{X}_j \mid x_i = \beta\}$
3      $\mathcal{F} = \{(i', \beta') \in \mathcal{I}_j \times \{0,1\} \mid x \in \mathcal{X}_j^\beta \Rightarrow x_{i'} = \beta'\}$
4      $\mathcal{I}_j = \{i' \in \mathcal{I}_j \mid \nexists \beta' \text{ s.t. } (i', \beta') \in \mathcal{F}\}$
5      for $(i', \beta') \in \mathcal{F} \backslash \{(x, \beta)\}$ do
6          Restriction-Propagation($(\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}, (i', \beta')$)
7  Set feasible = true $\Leftrightarrow \forall j \in [m] : \mathcal{X}_j \neq \varnothing$
   Output: feasible, restricted subproblems/indices
           $(\mathcal{X}_j, \mathcal{I}_j)_{j\in[m]}$
```

Algorithm 1, forward messages $\overrightarrow{m}$ for variables i', $i' < i$ remain valid, and the same holds true for backward messages $\overleftarrow{m}$ for variables $i' > i$. Hence, we only have to update the forward messages for variable $i+1$ to obtain correct marginals for variable $i+1$ via (6). An analogous reasoning holds true for the backward pass. $\square$

C.1. Abstract BDD update steps

Algorithm 7: Abstract forward BDD update step

```
   Input: variable index $i \in \mathcal{I}$
1  for $v \in V$ with $\mathrm{idx}(v) = i$ do
2      $\overrightarrow{m}_v = \left(\bigotimes_{u:uv\in A^0} \overrightarrow{m}_u \oplus \theta_{uv}\right) \otimes \left(\bigotimes_{u:uv\in A^1} (\overrightarrow{m}_u \oplus \theta_{uv})\right)$
```

Algorithm 8: Abstract backward BDD update step

```
   Input: variable index $i \in \mathcal{I}$
1  for $v \in V$ with $\mathrm{idx}(v) = i$ do
2      $\overleftarrow{m}_v = \left(\overleftarrow{m}_{vv_0^+} \oplus \theta_{vv_0^+}\right) \otimes \left(\overleftarrow{m}_{vv_1^+} \oplus \theta_{vv_1^+}\right)$
```

Similar to the min-marginals (1), we can compute marginal log-sum-exp (24) and solution counts (and thus R_i in (31)) efficiently with incremental BDD update steps. To this end, one can employ Algorithms 7–8 that are completely analogous to Algorithms 2–3 by substitution of the symbols $(\oplus, \otimes, \mathbb{0}, \mathbb{1}, \theta^0, \theta^1)$ with those listed in Table 2.

C.2. Variable fixations

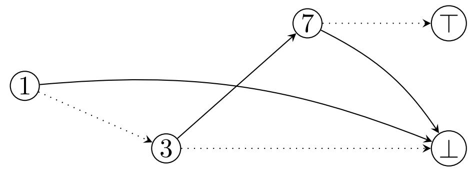

Figure 4. The BDD obtained from the one depicted in Figure 1 by fixing $x_3 = 1$. Nodes that are no longer reachable from the root or no longer lie on a path to $\top$ have been removed.

We implement the variable fixations $x_i = \beta$ in the primal heuristic as manipulations to all BDDs that involve variable x_i. The manipulations are specified in Algorithms 9 – 11. In order to fix $x_i = \beta$ in some BDD, Algorithm 9 redirects all outgoing $(1-\beta)$-arcs from the nodes $v \in V$ with $\mathrm{idx}(v) = i$ to $\bot$. If that leaves any node unreachable from the root, we recursively remove them with their outgoing arcs by Algorithm 10. If $\top$ becomes unreachable, the algorithm detects that the fixation renders the subproblem infeasible. Similarly, if some node no longer lies on any

Abstract	Min-marginal (1)	Marginal log-sum-exp (24)	Solution counts
$(\oplus, \otimes, \mathbb{0}, \mathbb{1}, \theta)$	$(+, \min, 0, \infty, [\lambda^j, 0])$	$(\cdot, +, 1, 0, [\exp(-\lambda_i^j/\alpha), 1])$	$(\cdot, +, 1, 0, [1, 1])$

Table 2. Symbols in Algorithms 7 and 8 to compute min-marginal (1), marginal log-sum-exp (24) and solution counts.

path towards $\top$, then we recursively remove it and redirect its incoming arcs by Algorithm 11. The variable fixation leads to a smaller BDD that represents the restricted feasible set. See Figure 4 for an example.

Remark. The complexity of any sequence of $\leq |\mathcal{I}|$ variable fixations for a BDD is bounded by the number of BDD nodes $|V|$.

Algorithm 9: Variable Fixation

```
   Input: variable index $i \in [n]$, $\beta \in \{0,1\}$
1  for $v \in V$ with idx$(v) = i$ do
2      Change $\beta$-arc $(v, v_\beta^+)$ to $(v, \bot)$
3      if $v_\beta^+$ has no incoming arcs left then
4          if Remove-Forward($v_\beta^+$) = false then
5              return false
6      if both outgoing arcs of $v$ point to $\bot$ then
7          Remove-Backward($v$)
8  return true
```

Algorithm 10: Remove-Forward

```
   Input: BDD node $v \in V$
1  if $v = \top$ then
2      return false
3  Remove $v$ and outgoing arcs $(v, v_0^+)$, $(v, v_1^+)$
4  if $v_0^+$ has no incoming arcs left then
5      Remove-Forward($v_0^+$)
6  if $v_1^+$ has no incoming arcs left then
7      Remove-Forward($v_1^+$)
```

Algorithm 11: Remove-Backward

```
   Input: BDD node $v \in V$
1  for $(u, v) \in A$ do
2      Replace $(u, v)$ by $(u, \bot)$
3      if both outgoing arcs of $u$ point to $\bot$ then
4          Remove-Backward($u$)
5  Remove $v$
```

D. Parallelization

Proof of Proposition 4

Proof. We need to show that the updates of the dual variable $\delta^{\rightarrow}$ and $\delta^{\leftarrow}$ in lines 9 and 16 of Algorithm 4 results in (i) feasible dual variables $\mu_a + \mu_{\bar{a}} \leq 0$ for every copy-arc pair $(\bar{a}, a)$ and (ii) is non-decreasing in the dual lower bound. We will show the statement for the update in line 9, the update in line 17 being analoguous.

(i) Note that $\delta^{\rightarrow}$ and $\delta^{\leftarrow}$ are non-negative. Hence, the first term $-\gamma \cdot \delta^{\rightarrow}_{(\bar{a},a)}$ in line 10 will decrease $\mu_{\bar{a}}$ (which does not affect feasibility), while the second term $\gamma \cdot \delta^{\leftarrow}_{(\bar{a},a)}$ offsets the changes made in line 17.

(ii) The subtraction $-\gamma \cdot \delta^{\rightarrow}_{(\bar{a},a)}$ will not decrease the cost of the optimal path in the BDD. First, since $\delta^{\rightarrow}$ is 0 for the arc the optimal solution takes, the cost of the optimal solution stays the same. For the other arcs $\bar{a}$ that are not in the optimal path, the value $\delta^{\rightarrow}_{(\bar{a},a)}$ is the difference of costs of the best path taking the arc $\bar{a}$ minus taking the optimal path. Hence, the cost update cannot result in previously non-optimal arcs to become optimal ones for $\gamma \leq 1$. Hence, the dual lower bound does not decrease after the subtraction.

Again note that $\delta^{\leftarrow}$ is non-negative. Hence, the second term $\gamma \cdot \delta^{\leftarrow}_{(\bar{a},a)}$ in line 10 will be non-decreasing in the dual lower bound, since it can only increase the costs.

□

E. ILP formulations

Below we detail the ILP formulations of the four problem types considered in the experiments.

E.1. Markov random fields

For MRFs, we formulate the associated optimization problem via the local polytope relaxation (Werner, 2007).

Definition 11 (MRF formulation)**.** Given a graph $G = (V, E)$ with label space $\mathcal{L}_i$ for all $i \in V$, unary potentials $\theta_i \in \mathbb{R}^{\mathcal{L}_i}$ for $i \in V$ and pairwise potentials $\theta_{ij} \in \mathbb{R}^{\mathcal{L}_i \times \mathcal{L}_j}$ for $ij \in E$ we define the feasible set Λ as those vectors

$$\mu \in \bigotimes_{i \in V} \{0,1\}^{\mathcal{L}_i} \otimes \bigotimes_{ij \in E} \{0,1\}^{\mathcal{L}_i \times \mathcal{L}_j} \tag{32}$$

that satisfy

$$\begin{array}{rl} \sum_{x_i \in \mathcal{L}_i} \mu_i(x_i) = 1 & \forall i \in V, \\ \sum_{x_i \in \mathcal{L}_i, x_j \in \mathcal{L}_j} \mu_{ij}(x_i, x_j) = 1 & \forall ij \in E, \\ \sum_{x_j \in \mathcal{L}_j} \mu_{ij}(x_i, x_j) = \mu_i(x_i) & \forall ij \in E, x_i \in \mathcal{L}_i, \\ \sum_{x_i \in \mathcal{L}_i} \mu_{ij}(x_i, x_j) = \mu_j(x_j) & \forall ij \in E, x_j \in \mathcal{L}_j. \end{array} \tag{33}$$

The overall 0–1-optimization problem reads

$$\min_{\mu \in \Lambda} \sum_{i \in V} \langle \theta_i, \mu_i \rangle + \sum_{ij \in E} \langle \theta_{ij}, \mu_{ij} \rangle . \tag{34}$$

E.2. Graph matching

The graph matching instances are about matching two sets of points L and R. There are both linear costs $c \in \mathbb{R}^{L \times R}$ as well as pairwise costs $d \in \mathbb{R}^{L \times L \times R \times R}$. We define the feasible set Γ as those vectors

$$\mu \in \{0,1\}^{L \times R}, \quad \nu \in \{0,1\}^{L \times L \times R \times R} \tag{35}$$

that satisfy

$$\begin{aligned} \mu \mathbb{1} &\leq \mathbb{1}, \\ \mu^\top \mathbb{1} &\leq \mathbb{1}, \\ \mu_{lr} &= \textstyle\sum_{l'=1}^{L} \sum_{r'=1}^{R} \nu_{ll'rr'} \quad \forall 1 \leq l \leq L,\ 1 \leq r \leq R, \\ \mu_{lr} &= \textstyle\sum_{l'=1}^{L} \sum_{r'=1}^{R} \nu_{l'lr'r} \quad \forall 1 \leq l \leq L,\ 1 \leq r \leq R. \end{aligned} \tag{36}$$

The 0–1-optimization problem is

$$\min_{(\mu,\nu) \in \Gamma} \langle c, \mu \rangle + \langle d, \nu \rangle . \tag{37}$$

Whenever we have sparse costs d we sparsify our encoding accordingly by leaving out the corresponding variables ν.

E.3. Cell tracking

We use the formulation given in (Haller et al., 2020).

Definition 12 (Cell tracking)**.** Given a set of nodes V corresponding to possible cell detections, a set of cell transitions $E \in \binom{V}{2}$ and a set of cell divisions $E' \in \binom{V}{3}$, we define variables $x_i \in \{0,1\}$, $i \in V$ to correspond to cell detections, $y_{ij} \in \{0,1\}$, $ij \in E$ to cell transitions and $y'_{ijk} \in \{0,1\}$, $ijk \in E'$ to cell divisions. Additional conflict sets $C_l \subset V$, $l \in \{1,\ldots,L\}$ for excluding competing cell detection hypotheses are also given. The feasible set $\mathcal{C}$ is defined as those vectors

$$x \in \{0,1\}^V, \quad y \in \{0,1\}^E, \quad y' \in \{0,1\}^{E'} \tag{38}$$

that satisfy

$$\begin{aligned} x_i &= \sum_{j:ij \in E} y_{ij} + \sum_{jk:ijk \in E'} y'_{ijk} && \forall i \in V, \\ x_j &= \sum_{i:ij \in E} y_{ij} + \sum_{ik:ijk \in E'} y'_{ijk} + \sum_{ik:ikj \in E'} y'_{ikj} && \forall j \in V, \\ & \sum_{i \in C_l} x_i \leq 1 \quad \forall l \in \{1,\ldots,L\}. \end{aligned} \tag{39}$$

Given cell detection costs $\theta_i \in \mathbb{R}$, $i \in V$, cell transition costs $\theta_{ij} \in R$, $ij \in E$ and cell division costs $\theta_{ijk} \in R$, $ijk \in E'$, the 0–1-optimization problem is

$$\min_{(x,y,y') \in \mathcal{C}} \langle \theta, (x,y,y')^\top \rangle . \tag{40}$$

Dataset		N	Avg n	Avg m
Cell tracking – small		10	22k	44k
Cell tracking – large		5	6.0M	1.3M
GM	*Hotel*	105	379k	52k
	House	105	379k	52k
	Worms	30	1.5M	166k
MRF	*Color-seg*	3	2.1M	8.2M
	Color-seg-n4	9	948k	3.2M
	Color-seg-n8	9	1.1M	6.4M
	Object-seg	5	531k	1.6M
QAPLIB	*small*	105	399k	38k
	large	29	25.8M	1.1M
Discrete tomography		2700	15k	11k

Table 3. For each dataset the table shows the number of instances N, average number of variables n, average number of constraints m.

E.4. Discrete tomography

The discrete tomography problem is encoded as an MRF with additional tomographic projection constraints.

Definition 13 (Discrete tomography)**.** Given a graph $G = (V, E)$ with label space $\mathcal{L}_i = \{0, 1, \ldots, k_i\}$ for all $i \in V$, unary potentials $\theta_i \in \mathbb{R}^{\mathcal{L}_i}$ for $i \in V$, pairwise potentials $\theta_{ij} \in \mathbb{R}^{\mathcal{L}_i \times \mathcal{L}_j}$ for $ij \in E$ and tomographic projection constraints $\sum_{i \in P_l} x_i = b_l$ for $l \in \{1,\ldots,L\}$, $P_l \subset V$, $b_l \in \mathbb{N}$, the constraint can be summarized as

$$\left\{ \mu \in \Lambda : \sum_{i \in P_l} \sum_{x_i \in \mathcal{L}_i} x_i \cdot \mu_i(x_i) = b_l \quad \forall l \in \{1,\ldots,L\} \right\} . \tag{41}$$

F. Experiments

F.1. Additional plots

In Figure 5, we show lower bound convergence plots for `Gurobi` and `BDD-MP` on all datasets. In Figure 6, we plot the convergence time speedup due to parallelization for the remaining *MRF* instances. In Figure 7, we show the convergence behavior of lower bounds in relation to the number of threads.

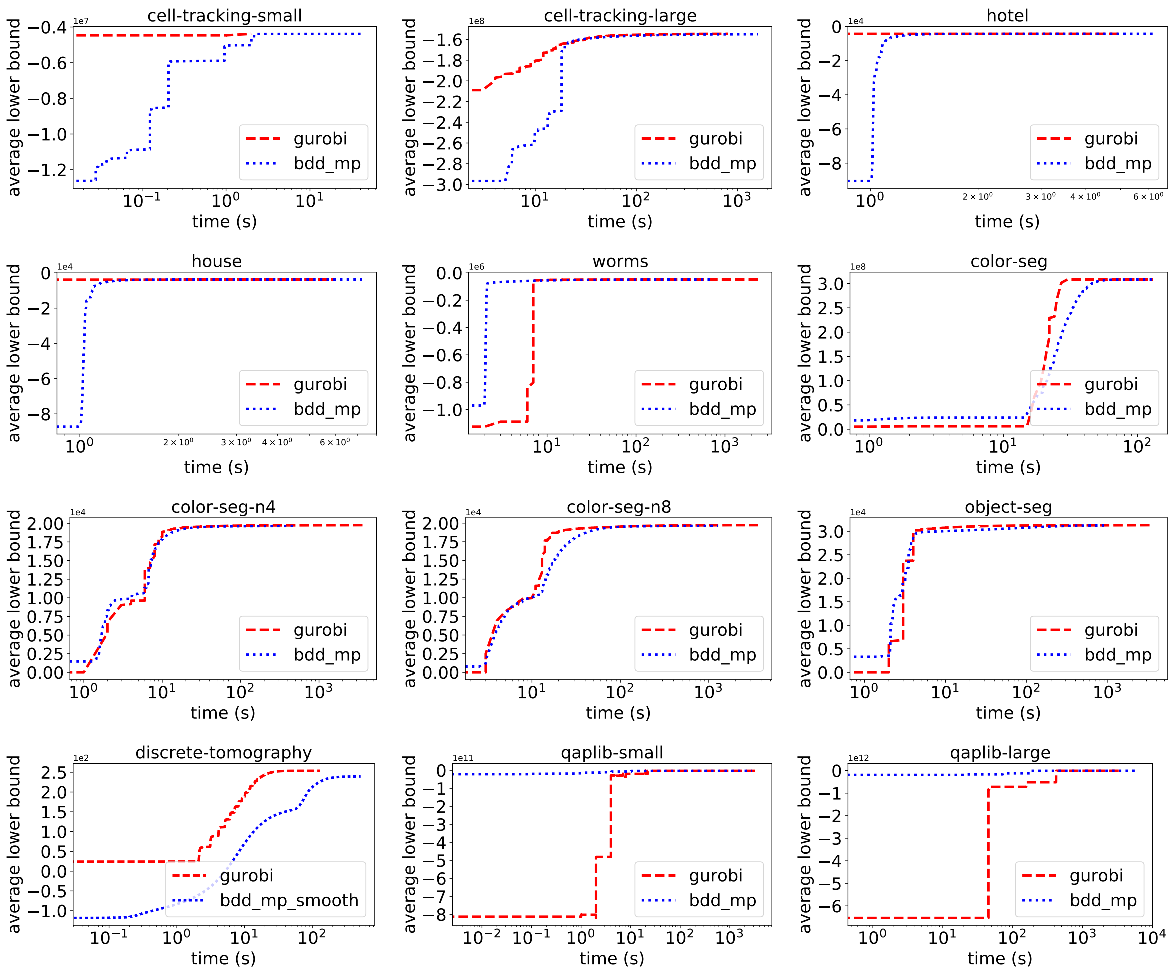

Figure 5. Averaged lower bound plots for `Gurobi` and the basic version of `BDD-MP` on all datasets.

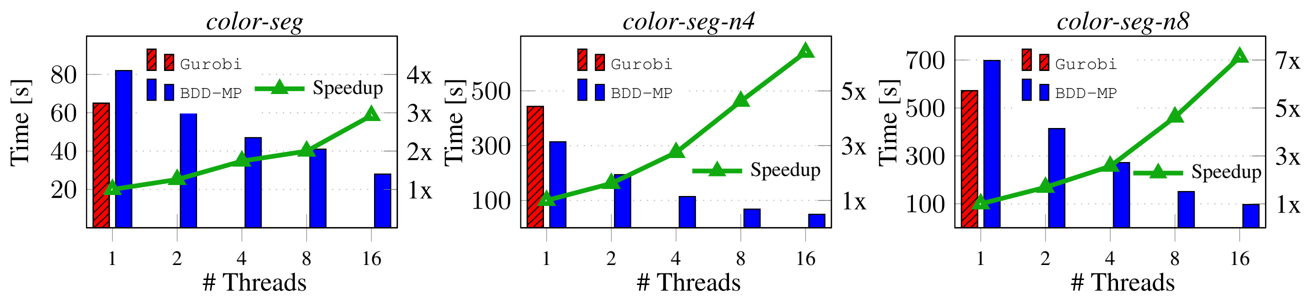

Figure 6. Running time until convergence (left axes) for Gurobi with dual simplex and our method with variable reordering and 1 to 16 threads. The right axes show the associated speedup factors of our method.

Table 4. The table reports lower bounds and running times for *QAPLIB small*[1] obtained with Gurobi's dual simplex method (`Simplex`), its barrier method with 16 threads (`Barrier`) and our method `BDD-MP`. The running times for `Barrier` do not include any crossover step. [1]We removed 6 instances that the barrier method could not solve within the time limit in order to enable a comparison.

	Lower Bound (LB)			LB Time [s]		
Dataset	`Simplex`	`Barrier`	`BDD-MP`	`Simplex`	`Barrier`	`BDD-MP`
QAPLIB small[1]	3.086e06	**8.499e06**	3.554e06	1670.2	374.6	**68.6**

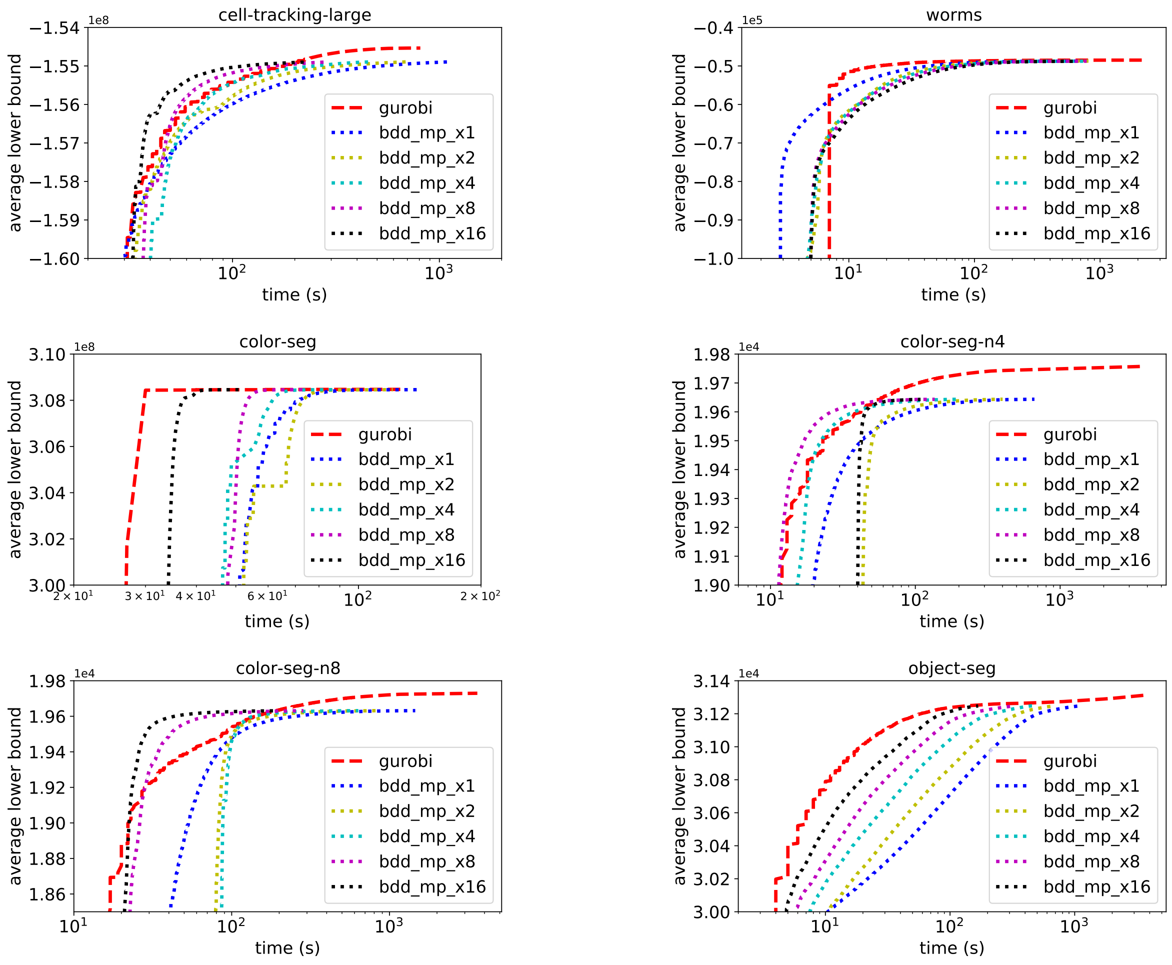

Figure 7. Zoomed-in lower bound convergence plots for Gurobi (dual simplex) and our method with variable reordering and 1 thread (`x1`) up to 16 threads (`x16`). The parallelized versions of our method converge faster after catching up with the initialization overhead.