Accelerating Dynamic Programs via Nested Benders Decomposition with Application to Multi-Person Pose Estimation

Shaofei Wang[1⋆], Alexander Ihler[2], Konrad Kording[3], and Julian Yarkony[4]

[1] Baidu Inc.
[2] UC Irvine
[3] University of Pennsylvania
[4] Experian Data Lab

Abstract. We present a novel approach to solve dynamic programs (DP), which are frequent in computer vision, on tree-structured graphs with exponential node state space. Typical DP approaches have to enumerate the joint state space of two adjacent nodes on every edge of the tree to compute the optimal messages. Here we propose an algorithm based on Nested Benders Decomposition (NBD) that iteratively lower-bounds the message on every edge and promises to be far more efficient. We apply our NBD algorithm along with a novel Minimum Weight Set Packing (MWSP) formulation to a multi-person pose estimation problem. While our algorithm is provably optimal at termination it operates in linear time for practical DP problems, gaining up to 500x speed up over traditional DP algorithm which have polynomial complexity.

Keywords: Nested Benders Decomposition, Column Generation, Multi-Person Pose Estimation

1 Introduction

Many vision tasks involve optimizing over large, combinatorial spaces, arising for example from low-level detectors generating large numbers of competing hypotheses which must be compared and combined to produce an overall prediction of the scene. A concrete example is multi-person pose estimation (MPPE), which is a foundational image processing task that can feed into many downstream vision-based applications, such as movement science, security, and rehabilitation. MPPE can be approached in a bottom-up manner, by generating candidate detections of body parts using, *e.g.*, a convolutional neural network (CNN), and subsequently grouping them into people.

The ensuing optimization problems, however, can be difficult for non-specialized approaches to solve efficiently. Relatively simple (tree-structured or nearly tree-structured) parts-based models can use dynamic programming (DP) to solve

⋆ work was done as an independent researcher before joining Baidu

object detection [6], pose estimation [17] and tracking [14] tasks. However, typical dynamic programming is quadratic in the number of states that the variables take on; when this is large, it can quickly become intractable. In certain special cases, such as costs based on Euclidean distance, tricks like the generalized distance transform [7] can be used to compute solutions more efficiently, for example in deformable parts models [6, 17], but are not applicable to more general cost functions.

In this paper we examine a model for MPPE that is formulated as a minimum-weight set packing problem, in which each set corresponds to an individual person in the image, and consists of the collection of all detections associated with that person (which may include multiple detections of the same part, due to noise in the low-level detectors). We solve the set packing problem as an integer linear program using implicit column generation, where each column corresponds to a pose, or collection of part detections potentially associated with a single person.

Unfortunately, while this means that the *structure* of the cost function remains tree-like, similar to single-pose parts models [6, 17], the number of *states* that the variables take on in this model are extremely large – each part (head, neck, etc.) can be associated with any number of detections in the image, meaning that the variables take on values in the power set of all detections of that part. This property renders a standard dynamic program on the tree intractable.

To address this issue, we apply a *nested Benders decomposition* (NBD) [13, 5] approach, that iteratively lower bounds the desired dynamic programming messages between nodes. The process terminates with the exact messages for optimal states of each node, while typically being vastly more efficient than direct enumeration over all combinations of the two power sets.

We demonstrate the effectiveness of our approach on the MPII-Multiperson validation set [2]. Contrary to existing primal heuristic solvers (*e.g.* [10]) for the MPPE model, our formulation is provably optimal when the LP relaxation is tight, which is true for over 99% of the cases in our experiments.

Our paper is structured as follows. We review related DP algorithms and MPPE systems in Section 2. In Section 3 we formulate MPPE as a min-weight set packing problem, which we solve via Implicit Column Generation (ICG) with dynamic programming as the pricing method. In Section 4 we show how the pricing step of ICG can be stated as a dynamic program. In Section 5 we introduce our NBD message passing, which replaces traditional message passing in the DP. Finally, in Section 6 we conduct experiments on the MPII-Multi-Person validation set, showing that our NBD based DP achieves up to 500x speed up over dynamic programming on real MPPE problems, while achieving comparable average precision results to a state-of-the-art solver based on a primal heuristic approach.

2 Related Work

In this section, we describe some of the relevant existing methodologies and applications of work which relate to our approach. Specifically, we discuss fast ex-

act dynamic programming methodologies and combinatorial optimization based models for MPPE.

2.1 Fast Dynamic Programming

The time complexity of dynamic programming (DP) grows linearly in the number of variables in the tree and quadratically in the state space of the variables. For applications in which the quadratic growth is a key bottleneck two relevant papers should be considered. In [12] the pairwise terms between variables in the tree are known in advance of the optimization and are identical across each edge in the tree. Hence they can be pre-sorted before inference, so that for each state of a variable the pairwise terms for the remaining variable are ordered. By exploiting these sorted lists, one can compute messages by processing only the lowest cost portion of the list and still guarantee optimality.

In a separate line of work [4], a column generation approach is introduced which attacks the dual LP relaxation of the DP. Applying duality, pairwise terms in the primal become constraints in the dual. Although finding violated constraints exhaustively would require the exact same time complexity as solving the DP with a more standard approach, by lower bounding the reduced costs the exhaustive enumeration can be avoided. Similarly, the LP does not need to be solved explicitly and instead can be solved as a DP.

In contrast to these lines of work our DP has significant structure in its pairwise interactions, corresponding to a high tree width binary Ising model, which we exploit. The previously cited work was not designed with domains containing these types of structures in mind.

2.2 Multi-person Pose Estimation in Combinatorial Context

Our experimental work is closely related to the sub-graph multi-cut integer linear programming formulation of [15, 8, 10], which we refer to as MC for shorthand. MC models the problem of MPPE as partitioning detections into body parts (or false positives) and clustering those detections into poses. The clustering process is done according to the correlation clustering [3, 19, 1] criteria, with costs parameterized by the part associated with the detection. This formulation is notable as it performs a type of non-maximum-suppression (NMS) by allowing poses to be associated with multiple detections of a given body part. However, the optimization problem of MC is often too hard to solve exactly and is thus attacked with heuristic methods. Additionally, MC has no easy way of incorporating a prior model on the number of poses in the image.

In contrast to MC, our model permits efficient inference with provable guarantees while modeling a prior using the cost of associating candidate detections with parts in advance of optimization. Optimization need not associate each such detection with a person, and can instead label it as a false positive. Associating detections with parts in advance of optimization is not problematic in practice, since the deep neural network nearly always produces highly unimodal probability distributions on the label of a given detection.

3 Multi-Person Pose Estimation as Minimum Weight Set Packing

In this section we formulate the bottom-up MPPE task as a Minimum Weight Set Packing (MWSP) problem and attack it with Implicit Column Generation. We use the body part detector of [8], which, after post-processing (thresholding, non max suppression (NMS), etc.), outputs a set of body part detections with costs that we interpret as terms in a subsequent cost function. We use the terms 'detection' and 'part detection' interchangeably in the remainder of this paper.

Each detection is associated with exactly one body part. We use fourteen body parts, consisting of the head and neck, along with right and left variants of the ankle, knee, hip, wrist and shoulder. We use the post-processing system of [8] which outputs pairwise costs that either encourage or discourage the joint assignment of two part detections to a common pose. Each pose thus consists of a selection of part detections; a pose can contain no detection of a body part (corresponding to an occlusion), or multiple detections (NMS) of that part. Each pose is associated with a cost that is a quadratic function of its members.

Given the set of poses and their associated costs we model the MPPE problem as a MWSP problem, which selects a set of poses that are pairwise disjoint (meaning that no two selected poses share a common detection) of minimum total cost.

3.1 Problem Formulation

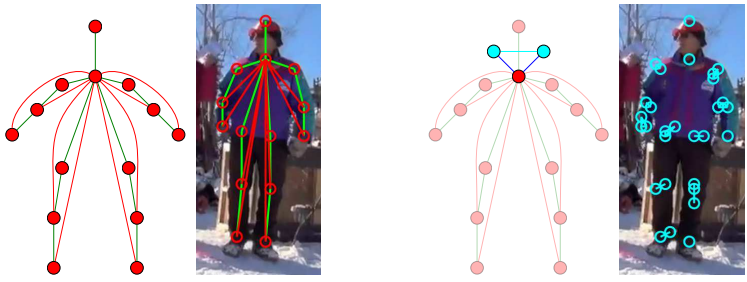
Detections and Parts Formally, we denote the set of part detections as $\mathcal{D}$ and index it with d. Similarly, we use $\mathcal{R}$ to denote the set of body parts and index it with r. We use $\mathcal{D}^r$ to denote the set of part detections of part r.

We use $\mathcal{S}^r$ to denote the power set of detections of part r, and index it with s. We describe mappings of detections to power set members using matrix $S^r \in \{0,1\}^{|\mathcal{D}|\times|\mathcal{S}^r|}$ where $S^r_{ds} = 1$ if and only if detection d is associated with configuration s. For convenience we explicitly define neck as part 0 and thus its power set is $\mathcal{S}^0$.

Poses: We denote the set of all possible poses over $\mathcal{D}$, *i.e.* the power set of $\mathcal{D}$, as $\mathcal{P}$ and index it with p. We describe mappings of detections to poses using a matrix $P \in \{0,1\}^{|\mathcal{D}|\times|\mathcal{P}|}$, and set $P_{dp} = 1$ if and only if detection d is associated with pose p. Since $\mathcal{P}$ is the power set of $\mathcal{D}$, it is too large to be considered explicitly. Thus, our algorithm works by building a subset $\hat{\mathcal{P}} \subseteq \mathcal{P}$ that captures the relevant poses to the optimization (see Section 3.2).

Pairwise Disjoint Constraints: We describe a selection of poses using indicator vector $\gamma \in \{0,1\}^{|\mathcal{P}|}$ where $\gamma_p = 1$ indicates that pose $p \in \mathcal{P}$ is selected, and $\gamma_p = 0$ otherwise.

A solution γ is valid if and only if the selected poses are pairwise disjoint, which is written formally as $P\gamma \leq 1$. The non-matrix version of the inequality $P\gamma \leq 1$ is $\sum_{p\in\mathcal{P}} P_{dp}\gamma_p \leq 1$ for each $d \in \mathcal{D}$.



(a) augmented-tree graph (b) detections of a pose

Fig. 1: Graphical representation of our pose model. (a) We model a pose in the image as an augmented-tree, in which each red node represents a body part, green edges are connections of traditional pictorial structure, while red edges are augmented connections from neck to all non-adjacent parts of neck. (b) Each body part can be associated with multiple part detections, a red node represents a body part while cyan nodes represent part detections of that part, blue edges indicate assignment of part detections to certain part of a person while cyan edges indicate pairwise costs among detections of the same part. The possible states of a body part thus consists of the power set of part detections of that part.

Cost Function: We express the total cost of a pose in terms of unary costs $\theta \in \mathbb{R}^{|\mathcal{D}|}$, where θ_d is the cost of assigning detection d to a pose, and pairwise costs $\phi \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{D}|}$, where $\phi_{d_1 d_2}$ is the cost of assigning detections d_1 and d_2 to a common pose. We use Ω to denote the cost of instancing a pose, which serves to regularize the number of people in an image. The cost of a pose is formally defined as :

$$\Theta_p = \Omega + \sum_{d\in\mathcal{D}} \theta_d P_{dp} + \sum_{\substack{d_1\in\mathcal{D}\\ d_2\in\mathcal{D}}} \phi_{d_1 d_2} P_{d_1 p} P_{d_2 p} \tag{1}$$

By enforcing some structure in the pairwise costs ϕ, we ensure that this optimization problem is tractable as a dynamic program. Consider a graph $G = (V, E)$, where $V = \mathcal{R}$, *i.e.* each node represents a body part, and $(\hat{r}, r) \in E$ if pairwise terms between part $\hat{r}$ and part r are non-zero. A common model in computer vision is to represent the location of parts in the body using a tree-structured model, for example in the deformable part model of [6, 17]; this forces the pairwise terms to be zero between non-adjacent parts in the tree [5].

In our application we augment this tree model with additional edges from the neck to all other non-adjacent body parts. This is illustrated in Fig 1. Then,

[5] WLOG: we assume that ϕ is upper triangular and that detections are ordered by part with the parent part being lower numbered than the child

conditioned on neck configuration s from $\mathcal{S}^0$, the conditional model is tree-structured and can be optimized using dynamic programming in $O(|\mathcal{R}|k^2)$ time, where k is the maximum number of detections per part.

Integer Linear Program: We now cast the problem of finding the lowest cost set of poses as an integer linear program (ILP) subject to pairwise disjoint constraints:

$$\min_{\gamma \in \{0,1\}^{|\mathcal{P}|}} \Theta^\top \gamma \quad \text{s.t.} \quad P\gamma \leq 1 \tag{2}$$

By relaxing the integrality constraints on γ, we obtain a linear program relaxation of the ILP, and convert the LP to its dual form using Lagrange multiplier set $\lambda \in \mathbb{R}_{0+}^{|\mathcal{D}|}$:

$$\min_{\substack{\gamma \geq 0 \\ P\gamma \leq 1}} \Theta^\top \gamma = \max_{\substack{\lambda \geq 0 \\ \Theta + P^\top \lambda \geq 0}} -1^\top \lambda \tag{3}$$

3.2 Implicit Column Generation

In this section we describe how to optimize the LP relaxation of Eq. (3). As discussed previously, the major difficulty to optimize Eq. (3) is the intractable size of $\mathcal{P}$. Instead, we incrementally construct a sufficient subset $\hat{\mathcal{P}} \subseteq \mathcal{P}$ so as to avoid enumerating $\mathcal{P}$ while still solving Eq. (3) exactly. This algorithm is called Implicit Column Generation(ICG) in the operations research literature, and is described formally in Alg. 1. Specifically, we alternate between finding poses with negative reduced costs (line 6) and re-optimizing Eq. (3) (line 3). Finding poses with negative reduced costs is achieved by conditioning on every neck configuration $s_0 \in \mathcal{S}^0$, and then identifying the lowest reduced cost pose among all the poses consistent with s_0 which we denote as $\mathcal{P}^{s_0}$.

Algorithm 1 Implicit Column Generation

1: $\hat{\mathcal{P}} \leftarrow \{\}$
2: **repeat**
3: $\quad \lambda \leftarrow$ Maximize dual in Eq. (3) over column set $\hat{\mathcal{P}}$
4: $\quad \dot{\mathcal{P}} \leftarrow \{\}$
5: $\quad$ **for** $s_0 \in \mathcal{S}^0$ **do**
6: $\quad\quad p_* \leftarrow \arg\min_{p \in \mathcal{P}^{s_0}} \Theta_p + \sum_{d \in \mathcal{D}} \lambda_d P_{dp}$
7: $\quad\quad$ **if** $\Theta_{p_*} + \sum_{d \in \mathcal{D}} \lambda_d P_{dp_*} < 0$ **then**
8: $\quad\quad\quad \dot{\mathcal{P}} \leftarrow [\dot{\mathcal{P}} \cup p_*]$
9: $\quad\quad$ **end if**
10: $\quad$ **end for**
11: $\quad \hat{\mathcal{P}} \leftarrow [\hat{\mathcal{P}}, \dot{\mathcal{P}}]$
12: **until** $|\dot{\mathcal{P}}| = 0$

4 Pricing via Dynamic Programming

A key step of Alg 1 is finding the pose with lowest reduced cost given dual variables λ (line 6):

$$\min_{p \in \mathcal{P}^{s_0}} \Theta_p + \sum_{d \in \mathcal{D}} \lambda_d P_{dp} \tag{4}$$

In the operations research literature, solving Eq. (4) is often called *pricing*.

Formally, let us assume the graph depicted in Fig. 1 (a) is conditioned on neck configuration, s_0, and thus becomes a tree graph. We define the the set of children of part r as $\{r \rightarrow\}$. We also define $\mu^r_{\hat{s}}$ as the cost-to-go, or the *message* of part r with its parent $\hat{r}$ associated with state $\hat{s}$:

$$\mu^r_{\hat{s}} = \min_{s \in \mathcal{S}^r} \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} S^{\hat{r}}_{\hat{d}\hat{s}} S^r_{ds} \phi_{\hat{d}d} + \nu^r_s \tag{5}$$

where the first term computes pairwise costs between part r and its parent $\hat{r}$. ν^r_s accounts for the cost of the sub-tree rooted at part r with state s, and is defined as:

$$\begin{aligned} \nu^r_s &= \psi^r_s + \sum_{\bar{r} \in \{r \rightarrow\}} \mu^{\bar{r}}_s \\ \psi^r_s &= \sum_{d \in \mathcal{D}^r} (\theta_d + \lambda_d) S^r_{ds} + \sum_{\substack{d_1 \in \mathcal{D}^r \\ d_2 \in \mathcal{D}^r}} \phi_{d_1 d_2} S^r_{d_1 s} S^r_{d_2 s} + \sum_{\substack{d_1 \in \mathcal{D}^0 \\ d_2 \in \mathcal{D}^r}} \phi_{d_1 d_2} S^0_{d_1 s_0} S^r_{d_2 s} \end{aligned} \tag{6}$$

Thus solving Eq. (4) involves computing and passing messages from leaf nodes (wrists and ankles) along the (conditional) tree graph $G = (V, E)$ to root node (head); Eq. (5) for root node equals to Eq (4) minus Ω. To compute $\mu^r_{\hat{s}}$ for every $\hat{s} \in \mathcal{S}^{\hat{r}}$, a node r need to pass through its states for each state of its parent node, thus resulting in polynomial time algorithm. If we have $|\mathcal{D}^r| = |\mathcal{D}^{\hat{r}}| = 15$, then we have roughly 30k states for r and $\hat{r}$, DP would then enumerate the joint space of 9×10^8 states, which becomes prohibitively expensive for practical applications.

5 Nested Benders Decomposition

In this section we present a near linear time algorithm (w.r.t $|\mathcal{S}^r|$) in practice that computes the message terms $\mu^{\bar{r}}_s$ in Eq. (6). The key idea of this algorithm is to apply Nested Benders Decomposition (NBD), so that for every parent-child edge $(r, \bar{r}), \forall \bar{r} \in \{r \rightarrow\}$, we iteratively construct a small sufficient set of affine functions of $\mathcal{D}^r$; the maximum of these functions lower bounds messages $\mu^{\bar{r}}_s$.

Essentially, each of these sets forms a lower envelope of messages, making them dependent on the maximum of the lower envelopes instead of child state $\bar{s}$; if the cardinality of the set is a small constant (relative to $|\mathcal{S}^{\bar{r}}|$), then we can compute the message on an edge for any parent state in $O(1)$ instead of $O(|\mathcal{S}^{\bar{r}}|)$, and thus computing messages for every state $s \in \mathcal{S}^r$ would take $O(|\mathcal{S}^r|)$ instead of $O(|\mathcal{S}^r| \times |\mathcal{S}^{\bar{r}}|)$.

5.1 Benders Decomposition Formulation

We now rigorously define our Benders Decomposition formulation for a specific parent-child edge pair $(r, \bar{r})$ which we denote as $e \in E$ for shorthand. We define the set of affine functions that lower bounds the message $\mu^{\bar{r}}_s$ as $\mathcal{Z}^e$ which we index by z, and parameterize the zth affine function as $(\omega^{ez}_0, \omega^{ez}_1, \ldots, \omega^{ez}_{|\mathcal{D}^r|})$. For simplicity of notation we drop the e superscript in the remaining of the paper. If $\mathcal{Z}^e$ indeed forms lower envelopes of $\mu^{\bar{r}}_s$ then we have:

$$\mu^{\bar{r}}_s = \max_{z \in \mathcal{Z}^e} \omega^z_0 + \sum_{d \in \mathcal{D}^r} \omega^z_d S^r_{sd}, \quad e = (r, \bar{r}) \in E \tag{7}$$

In the context of Benders Decomposition one affine function in $\mathcal{Z}^e$ is called a *Benders row*. For an edge e, we start with nascent set $\dot{\mathcal{Z}}^e$ with a single row in which $\omega^0_0 = -\infty, \omega^0_d = 0, d \in \mathcal{D}^r$ and iteratively add new Benders rows into $\dot{\mathcal{Z}}^e$. We define a lower bound on the message of edge $(r, \bar{r})$ as:

$$\mu^{\bar{r}-}_s = \max_{z \in \dot{\mathcal{Z}}^e} \omega^z_0 + \sum_{d \in \mathcal{D}^r} \omega^z_d S^r_{sd}, \quad e = (r, \bar{r}) \in E \tag{8}$$

which satisfies $\mu^{\bar{r}-}_s \leq \mu^{\bar{r}}_s$. The two terms become equal for $s^* = \arg\min_{s \in \mathcal{S}^r} \mu^{\bar{r}}_s$ if the lower bound is tight.

5.2 Producing new Benders rows

Until now we define parent-child pair as $(r, \bar{r})$ in the context of Eq. (6). In this section we describe how to generate new Benders rows in the context of Eq. (5), where parent-child pair is denoted as $(\hat{r}, r)$.

Given current set $\dot{\mathcal{Z}}^e$ of an edge $(\hat{r}, r) \in E$, with $\hat{r}$ associated with state $\hat{s}$, we check if there exist a new Benders row that can increase current lower bound $\mu^{r-}_{\hat{s}}$. This is computed by:

$$\min_{s \in \mathcal{S}^r} \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} S^{\hat{r}}_{\hat{d}\hat{s}} S^r_{ds} \phi_{\hat{d}d} + \nu^{r-}_s \tag{9}$$

where:

$$\nu_s^{r-} = \psi_s^r + \sum_{\bar{r} \in \{r \to\}} \mu_s^{\bar{r}-} \tag{10}$$

Integer Linear Program Here we reformulate Eq. (9) as an integer linear program. We use indicator vectors $x \in \{0,1\}^{|\mathcal{D}^r|}, y \in \{0,1\}^{|\mathcal{D}^{\hat{r}}| \times |\mathcal{D}^r|}$, where $x_s = 1$ if and only if $s \in \mathcal{S}^r$ is selected and $y_{\hat{d}d} = S^{\hat{r}}_{\hat{d}\hat{s}}(\sum_{s \in \mathcal{S}^r} x_s S^r_{ds})$:

$$\begin{aligned}
\min_{\substack{x \in \{0,1\}^{|\mathcal{S}^r|} \\ y \in \{0,1\}^{|\mathcal{D}^{\hat{r}}| \times |\mathcal{D}^r|}}} \quad & \sum_{s \in \mathcal{S}^r} \nu_s^{r-} x_s + \sum_{\substack{d \in \mathcal{D}^r \\ \hat{d} \in \mathcal{D}^{\hat{r}}}} \phi_{\hat{d}d} y_{\hat{d}d} \\
\text{s.t.} \quad & \sum_{s \in \mathcal{S}^r} x_s = 1 \\
& - y_{\hat{d}d} + S^{\hat{r}}_{\hat{d}\hat{s}} + \sum_{s \in \mathcal{S}^r} x_s S^r_{ds} \le 1, \quad \forall \hat{d} \in \mathcal{D}^{\hat{r}}, d \in \mathcal{D}^r \\
& y_{\hat{d}d} \le S^{\hat{r}}_{\hat{d}\hat{s}}, \quad \forall \hat{d} \in \mathcal{D}^{\hat{r}}, d \in \mathcal{D}^r \\
& y_{\hat{d}d} \le \sum_{s \in \mathcal{S}^r} x_s S^r_{ds}, \quad \forall \hat{d} \in \mathcal{D}^{\hat{r}}, d \in \mathcal{D}^r
\end{aligned} \tag{11}$$

We then relax x, y to be non-negative. In the supplement we provide proof that this relaxation is always tight. We express the dual of the relaxed LP below with dual variables $\delta^0 \in \mathbb{R}$, and $\delta^1, \delta^2, \delta^3$ each lie in $R_{0+}^{|\mathcal{D}^{\hat{r}}| \times |\mathcal{D}^r|}$ which is indexed by $\hat{d}, d$:

$$\begin{aligned}
\max_{\substack{\delta^0 \in \mathbb{R} \\ (\delta^1, \delta^2, \delta^3) \ge 0}} \quad & \delta^0 - \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} \delta^1_{\hat{d}d} + \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} (\delta^1_{\hat{d}d} - \delta^2_{\hat{d}d}) S^{\hat{r}}_{\hat{d}\hat{s}} \\
\text{s.t.} \quad & \nu_s^{r-} - \delta^0 + \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} (\delta^1_{\hat{d}d} - \delta^3_{\hat{d}d}) S^r_{ds} \ge 0, \quad \forall s \in \mathcal{S}^r \\
& \phi_{\hat{d}d} - \delta^1_{\hat{d}d} + \delta^2_{\hat{d}d} + \delta^3_{\hat{d}d} \ge 0, \quad \forall \hat{d} \in \mathcal{D}^{\hat{r}}, d \in \mathcal{D}^r
\end{aligned} \tag{12}$$

Observe Eq. (12) is an affine function of $\mathcal{D}^{\hat{r}}$, thus when dual variables are optimal Eq. (12) represents a new Benders row that we can add to $\dot{\mathcal{Z}}^e, e = (\hat{r}, r)$. Let us denote the new Benders row as z^*, then we construct this row from dual variables as:

$$\omega_0^{z^*} = \delta^0 - \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} \delta^1_{\hat{d}d} \tag{13}$$

$$\omega_{\hat{d}}^{z^*} = \sum_{d \in \mathcal{D}^r} \delta^1_{\hat{d}d} - \delta^2_{\hat{d}d}, \quad \forall \hat{d} \in \mathcal{D}^{\hat{r}} \tag{14}$$

Note that if all lower bounds on child messages $\mu^{\bar{r}-}_{s^*}, \forall \bar{r} \in \{r \rightarrow\}$ are tight for $s^* \in \mathcal{S}^r$ that minimizes Eq. (9) , then the new Benders row z^* forms a tight lower bound on message $\mu^r_{\hat{s}}$ for the specified parent state $\hat{s}$.

Solving Dual LP One could directly solve (12) in closed form, or via an off-the-shelf LP solver, both of which gives maximum lower bound for one parent state $\hat{s}$. However, ideally we want this new Benders row to also give a good lower bound to other parent states $\hat{s} \in \mathcal{S}^{\hat{r}}$, so that we can use as few rows as possible to form a tight lower bound on the messages.

We achieve this by adding an L1 regularization with tiny negative magnitude weight to prefer smaller values of δ^1, δ^2. This technique is referred to as a Pareto optimal cut or a Magnanti-Wong cut [11] in the operations research literature. We give detailed derivations as for why such regularization gives better overall lower bounds in the supplement.

5.3 Nested Benders Decomposition for Exact Inference

Algorithm 2 Nested Benders Decomposition

$G = (\mathcal{R}, E)$, G is a tree-structured graph
$\dot{\mathcal{Z}}^e \leftarrow$ single row with $\omega_0 = -\infty, \omega_{\hat{d}} = 0, \forall d \in \mathcal{D}^{\hat{r}}, \forall e = (\hat{r}, r) \in E$
$s^*_r \leftarrow \emptyset, \Delta^r \leftarrow 0, \forall r \in \mathcal{R}$
repeat
  for $r \in \mathcal{R}$ proceeding from leaves to root **do**
    for $z \in \dot{\mathcal{Z}}^e, e = (\hat{r}, r), r \in \{\hat{r} \rightarrow\}$ **do**
      Update δ^0 via Eq. (15)
      Update ω^z_0 via Eq. (13)
    end for
  end for
  $s^*_r \leftarrow \arg\min_{s \in \mathcal{S}^r} \nu^{r-}_s$, where r is root
  for $r \in \mathcal{R}$ from children of root to leaves **do**
    $s^*_r \leftarrow \arg\min_{s \in \mathcal{S}^r} \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} S^{\hat{r}}_{\hat{d}\hat{s}} S^r_{ds} \phi_{\hat{d}d} + \nu^{r-}_s$, where $\hat{s} = s^*_{\hat{r}}$
    $\Delta^r \leftarrow \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} S^{\hat{r}}_{\hat{d}\hat{s}} S^r_{ds} \phi_{\hat{d}d} + \nu^{r-}_s - \max_{z \in \dot{\mathcal{Z}}^e} \omega^z_0 + \sum_{\hat{d} \in \mathcal{D}^{\hat{r}}} \omega^z_{\hat{d}} S^{\hat{r}}_{\hat{d}\hat{s}}$, where $s = s^*_r, \hat{s} = s^*_{\hat{r}}, e = (\hat{r}, r), r \in \{\hat{r} \rightarrow\}$
  end for
  $r^* \leftarrow \arg\max_{r \in \mathcal{R}} \Delta^r$
  $\dot{\mathcal{Z}}^e \leftarrow \dot{\mathcal{Z}}^e \cup z^*$ where z^* is the new Benders row for $e = (\hat{r}, r^*), r^* \in \{\hat{r} \rightarrow\}$
until $|\Delta^r| < \epsilon, \forall r \in \mathcal{R}$
RETURN pose p corresponding $\{s^*_r, \forall r \in \mathcal{R}\}$

Given the basic Benders Decomposition technique described in previous sections, we now introduce the Nested Benders Decomposition algorithm which is described as Alg 2. The algorithm can be summarized in four steps:

Update Old Benders Rows (line 5-10) The NBD algorithm repeatedly updates the lower bounds on the messages between nodes, which makes $\dot{\mathcal{Z}}^e$ become less tight when messages from child nodes change. Instead of constructing $\dot{\mathcal{Z}}^e$ from scratch every iteration, we re-use δ terms produced by previous iterations, fixing $\delta^1, \delta^2, \delta^3$ and only update δ^0 to produce valid Benders rows given new child messages in ν_s^{r-}:

$$\delta^0 \leftarrow \min_{s \in \mathcal{S}^r} \nu_s^{r-} + \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} (\delta^1_{\hat{d}d} - \delta^3_{\hat{d}d}) S^r_{ds} \tag{15}$$

Compute Optimal State and Gaps for Each Node (line 11-15) Next we proceed from root to leaves and compute optimal state of each node, given current lower bounds on messages. Given current state estimates of a node r and its parent $\hat{r}$, we measure the gap between the message estimated by itself and the message estimated by its parent, and denote this gap as Δ^r (line 14). Note Δ for root is always 0 since root does not have a parent.

Find the Node that Gives Maximum Gap (line 16) We find the node r on which the gap Δ^r is largest across all nodes, and denote this node as r^*.

Compute and Add New Benders Row (line 17) We produce a new Benders row z^* for r^*, by solving Eq. (12)-(14). This row z^* is then added to the corresponding set $\dot{\mathcal{Z}}^e$ where $e = (\hat{r}, r^*), r^* \in \{\hat{r} \rightarrow\}$.

We terminate when the gap Δ of every node in the graph is under a desired precision ϵ (0 in our implementation), and return the optimal state of every node. In the following we prove that Alg 2 terminates with optimal total cost at root part (which we denote here as part 1) as computed by DP.

Lemma 1. *At termination of Alg 2, $\nu^{1-}_{s^*_1}$ has cost equal to cost of the pose corresponding to configurations of nodes $\{s^*_r, \forall r \in \mathcal{R}\}$*

Proof. At termination of Alg 2 the following is established for each $r \in \mathcal{R}$ with states $s = s^*_r, \hat{s} = s^*_{\hat{r}}$:

$$\Delta^r = 0 = \sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} S^{\hat{r}}_{\hat{d}\hat{s}} S^r_{ds} \phi_{\hat{d}d} + \nu_s^{r-} - \max_{z \in \dot{\mathcal{Z}}^e} \omega_0^z + \sum_{\hat{d} \in \mathcal{D}^{\hat{r}}} \omega^z_{\hat{d}} S^{\hat{r}}_{\hat{d}\hat{s}} \tag{16}$$

By moving the $-\max_{z \in \dot{\mathcal{Z}}^e} \omega_0^z + \sum_{\hat{d} \in \mathcal{D}^{\hat{r}}} \omega^z_{\hat{d}} S^{\hat{r}}_{\hat{d}}$ to the other side we establish the following.

$$\sum_{\substack{\hat{d} \in \mathcal{D}^{\hat{r}} \\ d \in \mathcal{D}^r}} S^{\hat{r}}_{\hat{d}\hat{s}} S^r_{ds} \phi_{\hat{d}d} + \nu_s^{r-} = \max_{z \in \dot{\mathcal{Z}}^e} \omega_0^z + \sum_{\hat{d} \in \mathcal{D}^{\hat{r}}} \omega^z_{\hat{d}} S^{\hat{r}}_{\hat{d}\hat{s}} = \mu_{\hat{s}}^{r-} \tag{17}$$

We now substitute $\mu_s^{\bar{r}-}$ terms in Eq. (10) with Eq. (17)

$$\nu_{\hat{s}}^{\hat{r}-} = \psi_{\hat{s}}^{\hat{r}} + \sum_{\substack{\hat{d}\in\mathcal{D}^{\hat{r}} \\ d\in\mathcal{D}^r}} S_{\hat{d}\hat{s}}^{\hat{r}} S_{ds}^r \phi_{\hat{d}d} + \nu_s^{r-} \tag{18}$$

Note at the leaves $\nu_s^{r-} = \psi_s^r, \forall s \in \mathcal{S}^r$. From ν_s^{1-}, we recursively expand the ν^- terms and establish the following:

$$\nu_{s_1^*}^{1-} = \sum_{r\in\mathcal{R}} \psi_{s_r^*} + \sum_{(\hat{r},r)\in E} \sum_{\substack{\hat{d}\in\mathcal{D}^{\hat{r}} \\ d\in\mathcal{D}^r}} S_{\hat{d}s_{\hat{r}}^*}^{\hat{r}} S_{ds_r^*}^r \phi_{\hat{d}d} \tag{19}$$

Which is the summation of all unary and pairwise terms chosen by solution $\{s_r^*, \forall r \in \mathcal{R}\}$.

Lemma 2. *At termination of Alg 2,* $\nu_{s_1^*}^{1-}$ *has cost equal to* $\min_{s_1\in\mathcal{R}^1} \nu_{s_1}^1$

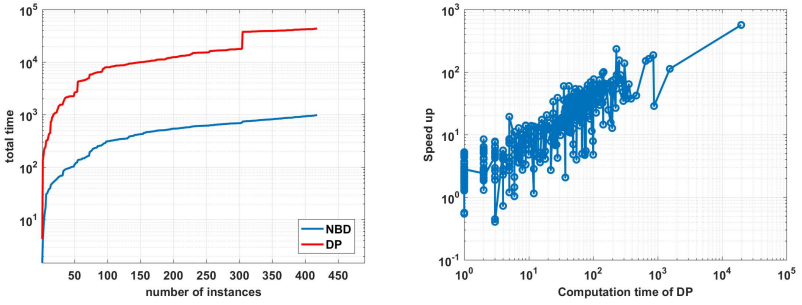
Proof. We prove this by contradiction. Suppose $\nu_{s_1^*}^{1-} \neq \min_{s_1\in\mathcal{R}^1} \nu_{s_1}^1$, according to Lemma 1 this must mean $\nu_{s_1^*}^{1-} > \min_{s_1\in\mathcal{R}^1} \nu_{s_1}^1$. If lower bounds on the messages from children of the root are tight, then it means $\nu_{s_1^*}^{1-}$ is not tight, Δ^1 would have been non-zero and Alg 2 would have not terminated, thus creating a contradiction. On the other hand, if lower bounds on certain message(s) from children is not tight, then the Δ value for that child node would have been non-zero and the algorithm would have continued running, still creating a contradiction.

Experimentally we observe that the total time consumed by steps in NBD is ordered from greatest to least as [1,2,4,3]. Note that the step solving the LP is the second least time consuming step of NBD.

6 Experiments

We evaluate our approach against a naive dynamic programming based formulation on MPII-Multi-person validation set [2], which consists of 418 images. The terms ϕ, θ are trained using the code of [8], with the following modifications:

1. We set $\phi_{d_1 d_2} = \infty$ for each pair of unique neck detections d_1, d_2; as a side effect this improves inference speed also since we need not explore the entire power set of neck detections.
2. We hand set Ω to a single value for the entire data set.
3. We limit the number of states of a given part/node to 50,000. We construct this set as follows: we begin with the state corresponding to zero detections included, then add in the group of states corresponding to one detection included; then add in the group of states corresponding to two detections included etc. If adding a group would have the state space exceed 50,000 states for the variable we don't add the group and terminate.



(a) (b)

Fig. 2: Timing comparison and speed-ups achieved by NBD. (a) Accumulated running time over problem instances for NBD and DP, respectively. (b) Factor of speed-up of NBD relative to DP, as a function of computation time spent for DP pricing. Note that in general the factor of speed-up grows as the problem gets harder for DP.

We compare solutions found by NBD and DP at each step of ICG; for all problem instances and all optimization steps, NBD obtains exactly the same solutions as DP (up to a tie in costs). Comparing total time spent doing NBD vs DP across problem instances we found that NBD is 44x faster than DP, and can be up to 500x faster on extreme problem instances. Comparison of accumulated running time used by NBD and DP over all 418 instances are shown in Fig. 2. We observe that the factor speed up provided by NBD increases as a function of the computation time of DP.

With regards to cost we observe that the integer solution produced over $\hat{\mathcal{P}}$ is identical to the LP value in over 99% of problem instances thus certifying that the optimal integer solution is produced. For those instances on which LP relaxation fails to produce integer results, the gaps between the LP objectives and the integer solutions are all within 1.5% of the LP objectives. This is achieved by solving the ILP in Eq 2 over $\hat{\mathcal{P}}$.

Part	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	mAP(UBody)	mAP	time (s/frame)
Ours	90.6	87.3	79.5	70.1	78.5	70.5	64.8	81.8	77.6	1.95
[10]	93.0	88.2	78.2	68.4	78.9	70.0	64.3	81.9	77.6	0.136

Table 1: We display average precision of our approach versus [10]. Running times are measured on an Intel i7-6700k quad-core CPU.

For the sake of completeness, we also report MPPE accuracy in terms of average precisions (APs) and compare it against a state-of-the-art primal heuristic solver [10] (Fig. 1). We note that compared to [10], we excel in hard-to-localize parts such as wrists and ankles, but fails at parts close to neck such as head and shoulder; this could be a side effect of the fact that costs from [8] are trained on power set of all detections including neck, thus pose associated with multiple neck detections could be a better choice for certain cases. In a more robust model, one could make a reliable head/neck detector, restricting each person to have only one head/neck.

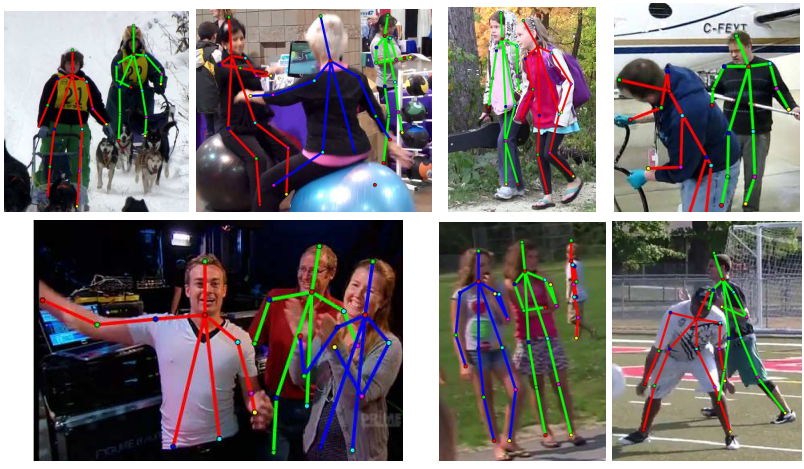


Fig. 3: Example output of our system.

7 Conclusion

We have described MPPE as MWSP problem which we address using ICG with corresponding pricing problem solved by NBD. For over 99% of cases we find provably optimal solutions, which is practically important in domains where knowledge of certainty matters, such as interventions in rehabilitation. Our procedure for solving the pricing problem vastly outperforms a baseline dynamic programming approach. We expect that NBD will find many applications in machine learning and computer vision, especially for solving dynamic programs with over high tree-width graphs. For example we could formulate sub-graph multi-cut tracking [16] as a MWSP problem solved with ICG with pricing solved via NBD. Moreover, for general graphs that main contain cycles, our NBD is directly applicable with dual decomposition algorithms [9, 18], which decompose the graph into a set of trees that are solvable by dynamic programs.

References

1. Andres, B., Kappes, J.H., Beier, T., Kothe, U., Hamprecht, F.A.: Probabilistic image segmentation with closedness constraints. In: Proc. of ICCV (2011)
2. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2d human pose estimation: New benchmark and state of the art analysis. In: Proc. of CVPR (2014)
3. Bansal, N., Blum, A., Chawla, S.: Correlation clustering. In: Journal of Machine Learning. pp. 238–247 (2002)
4. Belanger, D., Passos, A., Riedel, S., McCallum, A.: Map inference in chains using column generation. In: Proc. of NIPS (2012)
5. Birge, J.R.: Decomposition and partitioning methods for multistage stochastic linear programs. Operations research **33**(5), 989–1007 (1985)
6. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence **32**(9), 1627–1645 (2010)
7. Felzenszwalb, P.F., Huttenlocher, D.P.: Distance transforms of sampled functions. Tech. rep., Cornell Computing and Information Science (2004)
8. Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., Schiele, B.: Deepercut: A deeper, stronger, and faster multi-person pose estimation model. CoRR **abs/1605.03170** (2016), http://arxiv.org/abs/1605.03170
9. Komodakis, N., Paragios, N., Tziritas, G.: Mrf optimization via dual decomposition: Message-passing revisited. In: Proc. of ICCV (2007)
10. Levinkov, E., Uhrig, J., Tang, S., Omran, M., Insafutdinov, E., Kirillov, A., Rother, C., Brox, T., Schiele, B., Andres, B.: Joint graph decomposition and node labeling: Problem, algorithms, applications. In: Proc. of CVPR (2017)
11. Magnanti, T.L., Wong, R.T.: Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. Operations research **29**(3), 464–484 (1981)
12. McAuley, J.J., Caetano, T.S.: Exploiting data-independence for fast belief-propagation. In: Proc. of ICML (2010)
13. Murphy, J.: Benders, nested benders and stochastic programming: An intuitive introduction. arXiv preprint arXiv:1312.3158 (2013)
14. Pirsiavash, H., Ramanan, D., Fowlkes, C.C.: Globally-optimal greedy algorithms for tracking a variable number of objects. In: Proc. of CVPR (2011)
15. Pishchulin, L., Insafutdinov, E., Tang, S., Andres, B., Andriluka, M., Gehler, P., Schiele, B.: DeepCut: Joint subset partition and labeling for multi person pose estimation. In: Proc. of CVPR (2016)
16. Tang, S., Andres, B., Andriluka, M., Schiele, B.: Subgraph decomposition for multi-target tracking. In: Proc. of CVPR (2015)
17. Yang, Y., Ramanan, D.: Articulated pose estimation with flexible mixtures-of-parts. In: Proc. of CVPR (2011)
18. Yarkony, J., Fowlkes, C., Ihler, A.: Covering trees and lower-bounds on the quadratic assignment. In: Proc. of CVPR (2010)
19. Yarkony, J., Ihler, A., Fowlkes, C.: Fast planar correlation clustering for image segmentation. In: Proc. of ECCV (2012)