Winning the PASCAL 2011 MAP Challenge with Enhanced AND/OR Branch-and-Bound

Lars Otten, Alexander Ihler, Kalev Kask, Rina Dechter
Department of Computer Science
University of California, Irvine, U.S.A.
{lotten,ihler,dechter}@ics.uci.edu, kalev@otc.net

Abstract

This paper describes our entry for the MAP/MPE track of the PASCAL 2011 Probabilistic Inference Challenge, which placed first in all three time limit categories, 20 seconds, 20 minutes, and 1 hour. Our baseline is a branch-and-bound algorithm that explores the AND/OR context-minimal search graph of a graphical model guided by a mini-bucket heuristic. Augmented with recent advances that convert the algorithm into an anytime scheme, that improve the heuristic power via cost-shifting schemes, and using enhanced variable ordering schemes, it constitutes one of the most powerful MAP/MPE inference methods to date.

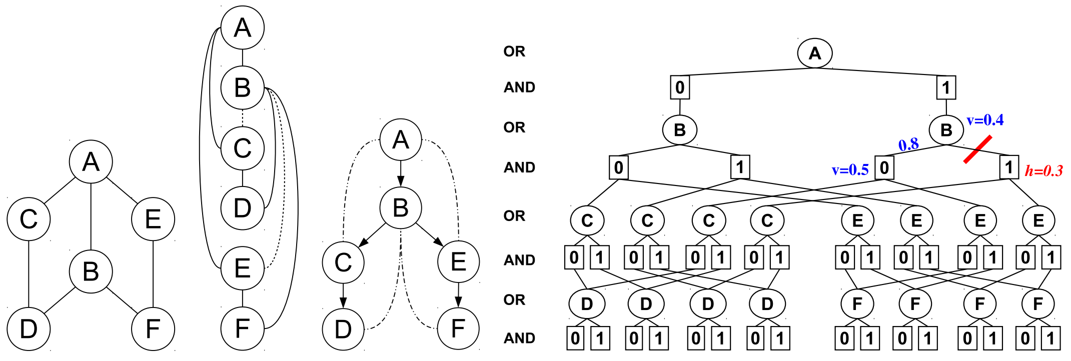
1 Introduction

There has been a recent tendency in the graphical models community to dismiss traditional search algorithms as unsuitable for combinatorial optimization challenges such as MAP/MPE problems, due to the huge search spaces inherent to these problems in most applications. Moreover, search algorithms often advertise themselves as exact, the very idea of which often seems hopeless for real applications. Belief propagation and sampling schemes on the other hand seem appealing due to their modest attitude. They do not insist on an (eventual) guarantee of optimality and therefore can sidestep the issue of combinatorially large search spaces, making them a practical alternative for approximate inference.

However, this perception is challenged by the success of search-based solvers in competitions for approximate reasoning [5, 6]. Success in real-world applications such as vision has also started to emerge [18, 19]. A more nuanced view, in which both “exact” search and approximate components such as message passing or sampling are used to complement one another is critical to achieving the best performance.

In this paper we describe our search-based algorithm, called DAOOPT, that won first place in all categories of the PASCAL 2011 Probabilistic Inference Challenge [6]. Through this algorithmic example we demonstrate the relevance of complete anytime search to approximation algorithms. The guarantee of an eventual proof of optimality is clearly another virtue of such schemes.

Our baseline algorithm is an AND/OR Branch and Bound scheme that explores the AND/OR context minimal search space of the graphical model with the aid of a mini-bucket heuristic. This class of algorithms has been developed over the last decade as summarized in a sequence of papers [12, 4, 3, 16, 17] and an implementation of it won 3rd place in the 2010 UAI competition [5]. The most recent push in performance can be attributed to three central advances: (1) work boosting the anytime capability of AND/OR depth-first search schemes [20], combined with stochastic local search [10]; (2) to significant improvement of the mini-bucket heuristic using belief propagation and dual decomposition views of cost-shifting [11]; and (3) enhanced, highly efficient variable ordering schemes [14].



(a) Primal graph. (b) Induced graph. (c) Pseudotree. (d) AND/OR context-minimal search graph with AOBB pruning example.

Figure 1: Example problem with six variables, induced graph along ordering A, B, C, D, E, F, corresponding pseudo tree, and resulting AND/OR search graph with AOBB pruning example.

In the sequel we describe our different algorithmic components and the empirical evaluation in the PASCAL Challenge [6].

2 Algorithm Details

We consider a MPE (Most Probable Explanation, sometimes also MAP, Maximum A Posteriori assignment) problem over a graphical model, $(X, F, D, \max, \prod)$. $F = \{f_1, \ldots, f_r\}$ is a set of functions over variables $X = \{X_1, \ldots, X_n\}$ with discrete domains $D = \{D_1, \ldots, D_n\}$, we aim to compute $\max_X \prod_i f_i$, the probability of the most likely assignment. The set of function scopes implies a *primal graph* and, given an ordering of the variables, an *induced graph* (where, from last to first, each node's earlier neighbors are connected) with a certain *induced width*. Another closely related combinatorial optimization problem is the *weighted constraint problem*, where we aim to minimize the sum of all costs, i.e. compute $\min_X \sum_i f_i$. These tasks have many practical applications but are known to be NP-hard in general [13].

2.1 AND/OR Search Spaces

The concept of **AND/OR search spaces** has recently been introduced to graphical models to better capture the structure of the underlying graph during search [3]. The search space is defined using a *pseudo tree* of the graph, which captures problem decomposition:

DEFINITION **1.** *A* pseudo tree *of an undirected graph $G = (X, E)$ is a directed, rooted tree $\mathcal{T} = (X, E')$, such that every arc of G not included in E' is a back-arc in $\mathcal{T}$, namely it connects a node in $\mathcal{T}$ to an ancestor in $\mathcal{T}$. The arcs in E' may not all be included in E.*

Given a graphical model instance with variables X and functions F its primal graph (X, E), and a pseudo tree $\mathcal{T}$, the associated *AND/OR search tree* consists of alternating levels of OR and AND nodes. Its structure is based on the underlying pseudo tree $\mathcal{T}$: the root of the AND/OR search tree is an *OR node* labeled with the root of $\mathcal{T}$. The children of an OR node $\langle X_i \rangle$ are *AND nodes* labeled with assignments $\langle X_i, x_j \rangle$ that are consistent with the assignments along the path from the root; the children of an AND node $\langle X_i, x_j \rangle$ are OR nodes labeled with the children of X_i in $\mathcal{T}$, representing conditionally independent subproblems.

Identical subproblems, identified by their context (the partial instantiation that separates the subproblem from the rest of the network), can be merged, yielding the *context-minimal AND/OR search graph* [3]. It was shown that, given a pseudo tree $\mathcal{T}$ of height h, the size of the AND/OR search tree based on $\mathcal{T}$ is $O(n \cdot k^h)$, where k bounds the variables' domain size. The context-minimal AND/OR search graph has size $O(n \cdot k^w)$, where w is the induced width of the problem graph along a depth-first traversal of $\mathcal{T}$ [3].

Example. Figure 1a shows an example problem graph with six variables. Figures 1b and 1c depict the induced graph and corresponding pseudo tree along ordering A, B, C, D, E, F, respectively. Figure 1d shows the Resulting context-minimal AND/OR search graph (induced width 2). Note that the AND nodes for B have two children each, representing independent subproblems and thus demonstrating problem decomposition. Furthermore, the OR nodes for D (with context $\{B, C\}$) and F (context $\{B, E\}$) have two edges converging from the AND level above them, signifying caching.

Given an AND/OR search space $S_\mathcal{T}$, a *solution subtree* $Sol_{S_\mathcal{T}}$ is a tree such that (1) it contains the root of $S_\mathcal{T}$; (2) if a nonterminal AND node $n \in S_\mathcal{T}$ is in $Sol_{S_\mathcal{T}}$ then all its children are in $Sol_{S_\mathcal{T}}$; (3) if a nonterminal OR node $n \in S_\mathcal{T}$ is in $Sol_{S_\mathcal{T}}$ then exactly one of its children is in $Sol_{S_\mathcal{T}}$.

2.2 AND/OR Branch-and-Bound

AND/OR Branch and Bound (AOBB) is a state-of-the-art algorithm for solving optimization problems such as max-product over graphical models. The edges of the AND/OR search graph can be annotated by weights derived from the set of cost functions F in the graphical model; finding the optimal-cost solution subtree solves the stated optimization task.

Assuming a maximization query, AOBB traverses the weighted context-minimal AND/OR graph in a depth-first manner while keeping track of the current lower bound on the maximal solution cost. A node n will be pruned if this lower bound exceeds a heuristic upper bound on the solution to the subproblem below n (cf. Section 2.2.1). The algorithm interleaves forward node expansion with a backward cost revision or propagation step that updates node values (capturing the current best solution to the subproblem rooted at each node), until search terminates and the optimal solution has been found [16, 17].

2.2.1 Mini-bucket Heuristics

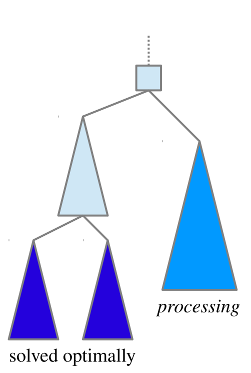
The heuristic $h(n)$ that we use in our experiments is the mini-bucket heuristic. It is based on mini-bucket elimination, which is an approximate variant of variable elimination and computes approximations to reasoning problems over graphical models [4]. A control parameter i allows a trade-off between accuracy of the heuristic and its time and space requirements – higher values of i yield a more accurate heuristic but take more time and space to compute. It was shown that the intermediate functions generated by the mini-bucket algorithm MBE(i) can be used to derive a heuristic function that is *admissible*, namely in a maximization context it overestimates the optimal cost solution to a subproblem in the AND/OR search graph [12].

Example. Figure 1d contains a simple pruning example: Assume that node $\langle B \rangle$ (with context $A = 1$) has a current value of 0.4, as a result of exploring its left child (AND node $\langle B, O \rangle$). Given a heuristic estimate of 0.3 for it's right child $\langle B, 1 \rangle$, the respectice subproblem can safely be pruned, since it can't possibly lead to a better solution (recall that an admissible heuristic value provides upper bounds in a maximization context).

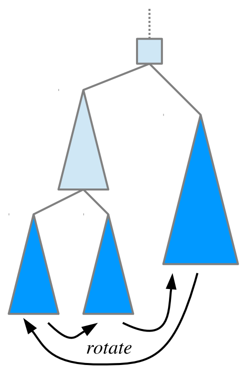
2.3 Breadth-Rotating AOBB

As a depth-first branch and bound scheme one would expect AOBB to quickly produce a non-optimal solution and then gradually improve upon it, maintaining the current best one throughout the search. However this ability is compromised in the context of AND/OR search.

Specifically, in AND/OR search spaces depth-first traversal of a set of independent subproblems will solve to completion all but one subproblem before the last one is even considered. As a consequence, the first generated overall non-optimal solution contains conditionally optimal solutions to all subproblems but the last one. Furthermore,



(a) Default depth-first subproblem processing.



(b) BRAOBB subproblem rotation.

Figure 2: Illustration of subproblem rotation in Breadth-Rotating AOBB.

depending on the problem structure and the complexity of the independent subproblems, the time to return even this first non-optimal overall solution can be significant, practically negating the anytime behavior of depth-first search (DFS).

Breadth-Rotating AND/OR Branch-and-Bound (BRAOBB) presents a principled extension of plain AOBB that addresses this issue and restores the anytime behavior over AND/OR search spaces [20]. It combines depth-rst exploration with the notion of "rotating" through different subproblems in a breadth-rst manner. Namely, node expansion still occurs depth-rst as in standard AOBB, but the algorithm takes turns in processing independent subproblems, each up to a given number of operations at a time, round-robin style.

Figure 2 illustrates this concept: In Figure 2a the two subproblems on the left need to be solved to completion before the third subproblem is considered at all. Using BRAOOBB, on the other hand, the three independent subproblems in Figure 2b contribute to the overall solution simultaneously.

Full algorithm details and analysis are provided in [20]. Among other things, we show that the desirable asymptotic memory complexity of depth-first search is maintained despite the breadth-first rotation over subproblems. BRAOBB is also demonstrated to provide drastically improved anytime performance for instances with several complex subproblem, yielding better solutions sooner – something critically important in the context of the time-limited PASCAL Inference Challenge.

2.4 Dual Decomposition and Message Passing

The mini-bucket bounding approach is equivalent to a particular *dual decomposition* bound, a class of methods that have gained considerable recent popularity in the machine learning community. Dual decomposition bounds for MAP/MPE are the dual formulation of a class of linear programming relaxations of the original graphical model [23, 1, 21]. They can be optimized in a number of ways, including several belief propagation-like message passing algorithms, most notably the "message passing linear programming" (MPLP) algorithm.

Practically speaking, the literature on mini-bucket and dual decomposition bounds are quite different. Standard mini-bucket gives a non-iterative bound; its parameter i is typically chosen to be high ("top-down", as close as possible to exact inference), resulting in a relaxation with large cliques. In contrast, most current dual decomposition approaches work on the original graph factors [8] or incrementally add slightly larger cliques such as cycles in a "bottom-up" fashion [22, 15].

In our implementation, we use cliques generated using standard mini-bucket approaches, but make use of message passing updates to improve the bound. In constraint satisfaction literature, these updates are often called *cost shifting* [1]. Since message passing is fastest on models with small cliques, like existing bottom-up approaches we first pass messages on the original graph, stopping after a time limit or convergence. We then find the largest i whose mini-bucket representation will fit in memory. We create an intermediate join-graph with clique size $i/2$ using mini-buckets, and again tighten the bound using message passing. Finally, we create a single-pass mini-bucket heuristic of size i, performing a "max-marginal matching" update that is equivalent to a single round of message passing, but requires no more memory than standard mini-bucket [11].

Our approach inherits the advantage both of large cliques (from mini-buckets) and of iterative improvements (from message passing). Many practical questions, however, such as the correct balance between quickly identifying large cliques (like mini-buckets) or using energy-based heuristics to identify the most important cliques (like cycle pursuit [22]) remain open.

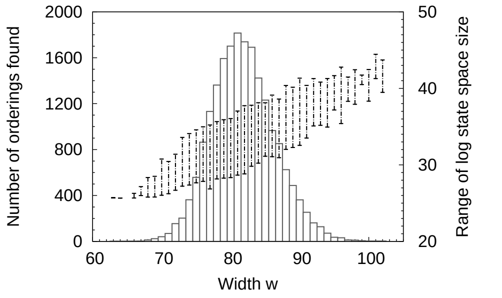


Figure 3: Example histogram of induced widths produced by 20,000 minfill iterations with random tie-breaking.

2.5 Enhanced Variable Ordering Scheme

As an algorithm exploring the AND/OR context-minimal search graph, the asympotitic time com-

	20 seconds	20 minutes	1 hour
1. MPLP on input graph	2 sec	30 sec / 500 iter	60 sec / 2000 iter
2. Stochastic Local search	2×2 sec	10×6 sec	20×10 sec
3. Iterative variable ordering	3 sec / 500 iter	60 sec / 10000 iter	180 sec / 30000 iter
4. MPLP on join graph	2 sec	30 sec / 250 iter	60 sec / 1000 iter
5. Mini-buckets + MM	$i = 15$, max 125MB	$i = 25$, max 4GB	$i = 35$, max 4GB
6. BRAOBB	*to completion or until timeout*		

Table 1: Algorithm parameters for different time limits. Steps are limited by execution time and max. number of iterations (whichever comes first), or in the case of mini-buckets with momemt matching, the largest i-bound that can fit within the given memory limit (4GB limit dictated by competition environment).

plexity of AOBB (and BRAOBB) is $O(n \cdot k^w)$,
i.e., exponential in the induced width w along a
given variable ordering (cf. Section 2.1 and [16, 20]). Hence, having orderings that yield small induced width is of central importance, yet for a given problem instance finding an ordering with the lowest possible width is known to be NP-hard. One therefore often relies on heuristics like *min-fill* and *min-degree*, which have proven to yield good results in practice.

These greedy schemes can have significant variance in terms of the orderings they produce, as Figure 3 illustrates by plotting a histogram of induced widths over 20,000 orderings produced by min-fill with simple random tie-breaking for a single problem instance (also shown is the variance in state space size, i.e., the product of variable domain sizes at each level in the resulting AND/OR search graph). We thus take the approach of iteratively running these ordering heuristics with the following key enhancements [14]:

- Highly efficient implementation with optimized data structures, to allow many times more iterations;
- Early detection of redundant iterations with unpromising results;
- Stochasticity in the greedy process, to encourage discovery of a more diverse set of orderings.

The last point goes beyond simple tie-breaking, but instead allows the algorithm to deviate from the heuristically best choice with a certain probability through random "pooling". The details of this scheme and its optimizations were developed in [14]. An extensive empirical evaluation showed impressive results, both in terms of speedup over previous implementations as well as the quality of the orderings returned.

2.6 Stochastic Local Search

Stochastic local search (SLS) is a general, poweful paradigm, which in the context of optimization problems, can can often find good solutions quickly. It contract to exhaustive search like AOBB, however, SLS is incomplete in the sense that it can never prove optimality. In our competition entry, several rounds of SLS are thus used as a preprocessing step to provide a first solution quickly, which is then also provided as an initial bound to BRAOBB.

In particular, we adapted open-source code by Frank Hutter that implements *Guided Local Search+* (GLS+) [9]. This particular variant of dynamic local search combines greedy hill-climbing with a penalty mechanism for local optima, to make these configurations less desirable going forward. Its effectiveness for MPE problem has been demonstrated empirically [10].

2.7 Putting it All Together

Our competition *DAOOPT* entry was comprised of a combination of all of the above components. Table 1 outlines the order in which they were executed, as well as their runtime allocation for the three different time limits that were evaluated.

	20 seconds		*20 minutes*		*1 hour*	
Category	daoopt	ficolofo	daoopt	ficolofo	daoopt	ficolofo
CSP	**-0.9123**	-0.8669	**-0.8739**	-0.7862	**-0.8442**	-0.6958
Deep belief networks	-	-	-1.6286	**-1.6342**	-5.0470	**-5.1707**
Grids	**-0.3403**	-0.3210	**-0.2437**	-0.2241	**-0.1721**	-0.1590
Image alignment	0.0000	0.0000	-0.0006	-0.0006	-0.0006	-0.0006
Medical diagnosis	-0.0028	**-0.0046**	-0.0037	**-0.0043**	-0.0041	**-0.0043**
Object detection	-4.8201	**-4.8287**	-4.8237	**-4.8743**	-1.9368	**-1.9628**
Protein folding	-0.0308	-0.0308	-0.1135	**-0.1187**	-0.1146	**-0.1183**
Protein protein interaction	-	-	**-0.1341**	-0.1317	-0.1681	**-0.1744**
Relational	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Segmentation	-0.0300	-0.0300	-0.0300	-0.0300	-0.0338	-0.0338
Overall	**-6.1364**	-6.0819	**-7.8519**	-7.8041	**-8.3214**	-8.3196

Table 2: Final MPE results as reported on the competition website [6]. Shown are, for each of the three timelimits, per-domain scores (averaged across instances) and overall scores (sum over all domains). Lower scores are better.

3 Overview of Results

To compare performance of different solvers in the PASCAL Challenge, each solver was run on a variety of problem instances by the organizers, the output was recorded, and a scoring metric computed. For the MPE track, the score is defined as the relative improvement over a common asynchronous belief propagation baseline and a "default" solution (chosen to maximize only the unary functions). More specifically, denote with $x^s = x^s_1, \ldots, x^s_n$ the solution of the solver in question, $x^{bp} = x^{bp}_1, \ldots, x^{bp}_n$ and $x^{def} = x^{def}_1, \ldots, x^{def}_n$ the baseline belief propagation and default solution, respectively, and $E(x) = -\sum_j \log f_j(x)$ the *energy* of a given solution. The score of x^s is then defined as follows:

$$Score(x^s) = \frac{E(x^s) - \min\{E(x^{bp}), E(x^{def})\}}{|\min\{E(x^{bp}), E(x^{def})\}|}$$

Table 2 lists the final competition results as published on the official website [6]. We include our solver *daoopt* as well the runner-up *ficolofo*, which interleaves variable neighborhood search and exact search, limited to varying subspaces, with soft arc consistency / constraint propagation [2, 7].

The results in Table 2 show that the final overall results were quite close (in fact, in the final weeks of the competition the scoreboard leaders changed several times, as the solvers got tweaked and bugs got ironed out). In particular, for the 1 hour category the difference in overall score is only 0.0018. In general we note that the results are somewhat balanced, with both solvers having a slight edge in certain problem classes. Noteworthy, however, is the large advantage that daoopt seems to have in the CSP domain. Looking at the per-instance results (available online), this advantage seems to be due to only a handful of instances where daoopt returned a vastly better solution than ficolofo.

More detailed analysis of the results, regarding the notable CSP instances as well as other cases, necessitates access to the actual input files. This is not yet available but has been announced for the future by the competition organizers.

4 Summary

We have described DAOOPT, our submission to the 2011 PASCAL Probabilistic Inference Challenge that won all three categories of the MAP/MPE track. Its success challenges the notion that "traditional", complete search algorithms are of little practical relevance in today's applications, compared to dedicated approximate inference schemes. On the contrary, the outcome of the competition indicates that powerful techniques form both ends of the spectrum can be combined efficiently to yield unrivaled performance.

Source Code. The C++ source code of BRAOBB is available under GPL license at `http://github.com/lotten/daoopt`.

Acknowledments. This work was supported in part by NSF grants IIS-0713118, IIS-1065618 and NIH grant 5R01HG004175-03.

References

[1] Martin Cooper and Thomas Schiex. Arc consistency for soft constraints. *Artificial Intelligence*, 154(1):199–227, 2004.

[2] Martin C. Cooper, Simon de Givry, Marti Sanchez, Thomas Schiex, Matthias Zytnicki, and Tomas Werner. Soft arc consistency revisited. *Artif. Intell.*, 174(7-8):449–478, 2010.

[3] Rina Dechter and Robert Mateescu. AND/OR search spaces for graphical models. *Artif. Intell.*, 171(2-3):73–106, 2007.

[4] Rina Dechter and Irina Rish. Mini-buckets: A general scheme for bounded inference. *J. ACM*, 50(2):107–153, 2003.

[5] Gal Elidan, Amir Globerson, and Uri Heinemann. 2010 UAI Approximate Inference Challenge. `http://www.cs.huji.ac.il/project/UAI10/`, 2010.

[6] Gal Elidan, Amir Globerson, and Uri Heinemann. PASCAL 2011 Probabilistic Inference Challenge. `http://www.cs.huji.ac.il/project/PASCAL/`, 2012.

[7] Mathieu Fontaine, Samir Loudni, and Patrice Boizumault. Guiding vns with tree decomposition. In *IEEE 23rd International Conference on Tools with Artificial Intelligence, ICTAI 2011*, pages 505–512, 2011.

[8] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Advances in Neural Information Processing Systems 21*, 2007.

[9] Frank Hutter. SLS4MPE source code. `http://www.cs.ubc.ca/labs/beta/Projects/SLS4MPE/`, 2005.

[10] Frank Hutter, Holger H. Hoos, and Thomas Stützle. Efficient stochastic local search for MPE solving. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence, IJCAI 2005*, pages 169–174, 2005.

[11] Alexander Ihler, Natalia Flevora, Lars Otten, and Rina Dechter. Join-graph based cost-shifting schemes. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence, UAI 2012*, 2012.

[12] Kalev Kask and Rina Dechter. A general scheme for automatic generation of search heuristics from specification dependencies. *Artif. Intell.*, 129(1-2):91–131, 2001.

[13] Kalev Kask, Rina Dechter, Javier Larrosa, and Avi Dechter. Unifying tree decompositions for reasoning in graphical models. *Artif. Intell.*, 166(1-2):165–193, 2005.

[14] Kalev Kask, Andrew Gelfand, Lars Otten, and Rina Dechter. Pushing the power of stochastic greedy ordering schemes for inference in graphical models. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence, AAAI 2011*, 2011.

[15] Nikos Komodakis and Nikos Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. In *European Conference on Computer Vision*, pages 806–820, 2008.

[16] Radu Marinescu and Rina Dechter. AND/OR Branch-and-Bound search for combinatorial optimization in graphical models. *Artif. Intell.*, 173(16-17):1457–1491, 2009.

[17] Radu Marinescu and Rina Dechter. Memory intensive AND/OR search for combinatorial optimization in graphical models. *Artif. Intell.*, 173(16-17):1492–1524, 2009.

[18] Vlad I. Morariu and Larry S. Davis. Multi-agent event recognition in structured scenarios. In *24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011*, pages 3289–3296, 2011.

[19] Vlad I. Morariu, David Harwood, and Larry S. Davis. Tracking People's Hands and Feet using Mixed Network AND/OR Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99, 2012.

[20] Lars Otten and Rina Dechter. Anytime AND/OR depth-first search for combinatorial optimization. *AI Commun.*, 25(3):211–227, 2012.

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

[22] David Sontag, Talya Meltzer, Amir Globerson, Tommi Jaakkola, and Yair Weiss. Tightening LP relaxations for MAP using message passing. In *Uncertainty in Artificial Intelligence*, pages 503–510, 2008.

[23] Martin Wainwright, Tommi Jaakkola, and Alan Willsky. MAP estimation via agreement on (hyper)trees: message-passing and linear programming approaches. *IEEE Trans. Inform. Theory*, 51(11):3697–3717, 2005.