Sparse Message Passing Algorithms for Weighted Maximum Satisfiability

Aron Culotta CULOTTA@CS.UMASS.EDU
Andrew McCallum MCCALLUM@CS.UMASS.EDU
Department of Computer Science, University of Massachusetts, Amherst, MA 01003

Bart Selman SELMAN@CS.CORNELL.EDU
Ashish Sabharwal SABHAR@CS.CORNELL.EDU
Department of Computer Science, Cornell University, Ithaca, NY 14853

Abstract

Weighted maximum satisfiability is a well-studied problem that has important applicability to artificial intelligence (for instance, MAP inference in Bayesian networks). General-purpose stochastic search algorithms have proven to be accurate and efficient for large problem instances; however, these algorithms largely ignore structural properties of the input. For example, many problems are highly *clustered*, in that they contain a collection of loosely coupled subproblems (e.g. pipelines of NLP tasks). In this paper, we propose a message passing algorithm to solve weighted maximum satisfiability problems that exhibit this clustering property. Our algorithm fuses local solutions to each subproblem into a global solution by iteratively passing summary information between clusters and recomputing local solutions. Because the size of these messages can become unwieldy for large problems, we explore several message compression techniques to transmit the most valuable information as compactly as possible. We empirically compare our algorithm against a state-of-the-art stochastic solver and show that for certain classes of problems our message passing algorithm finds significantly better solutions.

Presented at *North East Student Colloquium on Artificial Intelligence (NESCAI)*, 2007. Copyright the authors.

1. Introduction

A weighted maximum satisfiability problem (WMAX-SAT) has as input a Boolean formula in conjunctive normal form, where each clause has a corresponding weight. A solution is a truth assignment that maximizes the sum of the weights of the satisfied clauses.

Many important problems in artificial intelligence can be reduced to instances of WMAX-SAT. For example, Park (2002) has shown that finding the most probable variable assignment in a Bayesian network can be solved efficiently through a reduction to WMAX-SAT. More recently, Richardson and Domingos (2006) have used WMAX-SAT solvers to find the most probable variable assignment in weighted logics.

Stochastic WMAX-SAT solvers such as Walksat (Selman et al., 1993) are efficient, scalable, and domain-independent. However, because of their generality, they ignore structural properties of the problem. For instance, many problems in artificial intelligence decompose into loosely-coupled subproblems, in which densely connected clusters of variables are connected to adjacent clusters by a relatively small number of clauses. In many cases, each of these subproblems can be solved with an efficient specialized algorithm (e.g., a dynamic program); the difficulty is in communicating solutions among subproblems.

In this paper, we propose a search algorithm for WMAX-SAT that is designed specifically for problems that exhibit this clustering of variables. Inspired by belief propagation algorithms that have been effective for probabilistic inference (Yedidia et al., 2000), our algorithm computes solutions to overlapping subproblems of the input, then iteratively passes messages between subproblems to converge

upon a global solution. When the subproblems are solved using a stochastic solver such as Walksat, our algorithm can be understood as a way to augment stochastic search with belief propagation. The resulting algorithm provides a general framework for solving many related maximization problems jointly.

As we will show, the size of these messages grows quickly with the number of variables shared among clusters. We therefore introduce and compare several *sparse* representations of messages that allow us to compactly transmit summary information from one cluster to the next.

We empirically validate our approach on synthetic and real-world WMAX-SAT problems and demonstrate significant improvements in solution quality. We hypothesize two reasons for this improvement: First, by computing solutions to subproblems, we effectively "cache" partial solutions to the original problem. Thus, when the assignment to a variable in one subproblem is changed, we can quickly look-up the best compatible solution to a related subproblem. Second, the subproblems are often easier to solve than the original problem because they are less constrained (i.e. have a lower clause-to-variable ratio). We perform several experiments to support these claims.

In the following sections, we first formally define WMAX-SAT, then describe solutions based on max-product and generalized max-product algorithms. We then propose an algorithm we call *sparse generalized max-product*, present several experiments validating our approach, and conclude with a discussion of related work.

2. Weighted Maximum Satisfiability

Let $\mathbf{X}$ be a set of n boolean variables $\{X_1 \dots X_n\}$, and let $\mathbf{c}$ be a set of m *clause functions* $\{c_1 \dots c_m\}$, where $c_i(\mathbf{X}_s \subseteq \mathbf{X})$ maps a set of variables to a real number. Let $\delta(\mathbf{X}_s) \in \{0, 1\}$ be an indicator function that is 1 if and only if there exists $X_i \in \mathbf{X}_s$ such that $X_i = 1$; that is, $\delta(\mathbf{X}_s)$ indicates if the disjunction of the variables in $\mathbf{X}_s$ evaluates to true. Clause $c_i(\mathbf{X}_s)$ is said to be *satisfied* if $\delta(\mathbf{X}_s) = 1$. The value of a clause is defined as $c_i(\mathbf{X}_s) = w_i\delta(\mathbf{X}_s)$, where w_i is the *weight* for clause c_i.

Let $\mathbf{x} \in \{0, 1\}^n$ be an assignment to $\mathbf{X}$. The *score* of an assignment is the sum of the weights of the satisfied clauses: $s(\mathbf{X} = \mathbf{x}) = \sum_{c_i \in \mathbf{c}} c_i(\mathbf{X}_s = \mathbf{x}_s)$. *Weighted maximum satisfiability (WMAX-SAT)* is the problem of finding the assignment with the highest score:

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmax}}\ s(\mathbf{X} = \mathbf{x}) \tag{1}$$

Many efficient algorithms have been proposed to solve WMAX-SAT. (Stützle et al. (2002) provide an overview.) Given the efficiency and generality of these techniques, it is reasonable to encode many difficult optimization problems as WMAX-SAT instances and solve them with an off-the-shelf solver. Indeed, this approach has been taken in probabilistic reasoning to find the most likely assignment to variables in a graphical model in cases where the exact dynamic programming solution is intractable (Park, 2002; Richardson & Domingos, 2006).

However, this "reduce to SAT" approach ignores structural properties of the original problem. In this paper, we consider an important class of problems that can be viewed as a combination of subproblems. Each subproblem is represented by a set of clauses over a subset of $\mathbf{X}$, and two subproblems interact via the variables they share. We refer to this class of problems as *clustered* WMAX-SAT instances.

Solving clustered WMAX-SAT instances with a standard solver fails to take advantage of this structure. This is especially problematic if there exist efficient, exact solutions to subproblems, as these exact solutions will only be approximated with a general WMAX-SAT solver. For example, if parsing is one subproblem of a larger NLP task, it would be preferable to use the exact dynamic programming solution to perform parsing, rather than approximate search. In the following section, we describe message passing algorithms for WMAX-SAT, and then propose an algorithm that is particularly suited to clustered WMAX-SAT problems.

3. A Graphical Model for WMAX-SAT

We will use the language of graphical models to represent WMAX-SAT problems. A graphical model represents a probability distribution over random variables $\mathbf{X}$ with a set of functions called *factors* (or *compatibility functions*) over subsets of $\mathbf{X}$. Let $\mathbf{f} = \{f_1 \dots f_m\}$ be the set of factors, where $f_i : \mathbf{X} \rightarrow \mathcal{R}^+$. A graphical model defines a distribution over $\mathbf{X}$ as $p(\mathbf{X}) \propto \prod_i f_i(\mathbf{X}_s)$. A common inference task is to find the most probable assignment to $\mathbf{X}$:

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmax}}\ p(\mathbf{X} = \mathbf{x}) \tag{2}$$

$$= \underset{\mathbf{x}}{\operatorname{argmax}} \sum_i \log f_i(\mathbf{X}_s = \mathbf{x}_s) \tag{3}$$

Substituting $\log f_i(\mathbf{X}_s = \mathbf{x}_s) = c_i(\mathbf{X}_s = \mathbf{x}_x)$ in Equation 3 results in an optimization problem equivalent to the WMAX-SAT problem given in Equation 1, where factors represent clauses, and the probability of an assignment is proportional to its score.

A distribution can be represented by a *factor graph* $\mathcal{G} = (\mathbf{X}, \mathbf{f}, \mathbf{E})$, a bipartite graph where vertex set X is a set of random variables, f is the set of factors, and edges E connect factors to their arguments (see Figure 1(a)). If $\mathcal{G}$ is acyclic, then the well-known *max-product* message passing algorithm can exactly compute the most probable assignment to $\mathbf{X}$ in polynomial time (Weiss & Freeman,

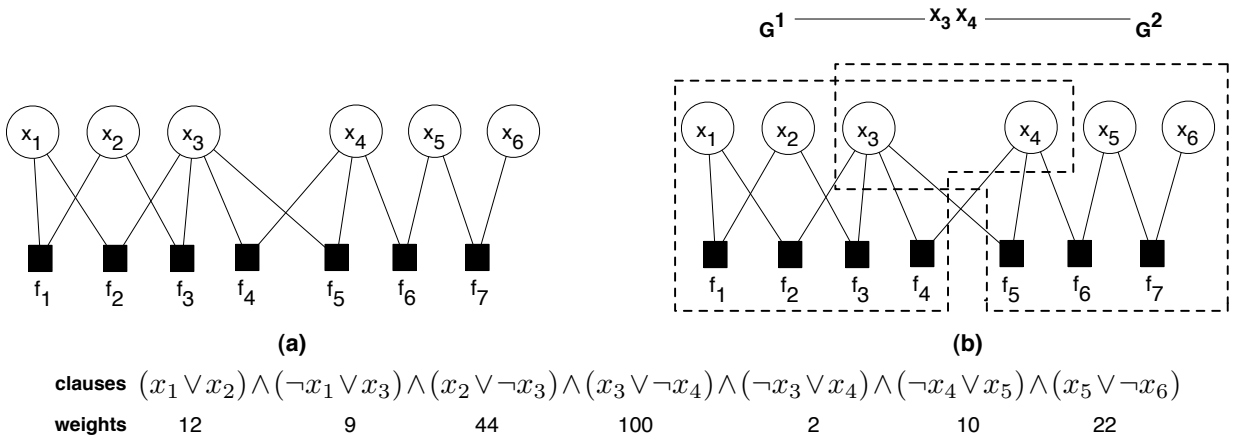

Figure 1. (a) A factor graph for a WMAX-SAT instance with 7 clauses and 6 variables. (b) A cluster graph for the same instance containing two clusters, G^1 and G^2.

2001). When $\mathcal{G}$ contains cycles, max-product provides an approximate solution; however, not only is there no guarantee that this approximation will return the optimal solution, but there is also no guarantee that the algorithm will converge. There are two additional difficulties of using max-product to solve WMAX-SAT. First, it is non-trivial to convert the output of max-product into an optimal assignment to $\mathbf{X}$ because max-product provides a distribution over assignments to single variable, not a joint distribution over all variables. Second, much like approximate solvers for WMAX-SAT, max-product ignores any structure in the connectivity of the variables. Thus, max-product is not well-suited to clustered WMAX-SAT instances.

4. Sparse Generalized Max-Product for WMAX-SAT

Yedidia et al. (2000) introduced *generalized belief propagation* to perform inference in graphical models that contain clusters of variables. We refer to the maximization version of generalized belief propagation as *generalized max-product*. In this section, we first describe how to apply generalized max-product to WMAX-SAT and discuss the difficulties involved. We then propose ways to overcome these difficulties by extending generalized max-product to operate with sparse messages and stochastic WMAX-SAT solvers.

Let the original factor graph $\mathcal{G} = (\mathbf{X}, \mathbf{E}, \mathbf{f})$ be partitioned into a *cluster graph* $\mathcal{G}_c$ containing t overlapping subgraphs $\{G^1 \dots G^t\}$ called *clusters*, where $G^k = (\mathbf{X}^k, \mathbf{E}^k, \mathbf{f}^k)$, $\mathbf{X}^k \subseteq \mathbf{X}$ and $\mathbf{f}^k \subseteq \mathbf{f}$ (Figure 1(b)). Note that each factor appears in exactly one cluster, but variables may appear in more than one cluster.

$\mathbf{X}^k$ contains all variables that are arguments to factors in $\mathbf{f}^k$. Let $\mathbf{X}^{ij}$ be the variables shared between cluster G^i and G^j. $\mathcal{G}_c$ contains an edge between two clusters G^i and G^j if they share variables, i.e., $\mathbf{X}^{ij} \neq \emptyset$. Each edge in $\mathcal{G}_c$ is labeled with $\mathbf{X}^{ij}$.

The product of the factors in each of the subgraphs is equal to the product of the factors in the original graph, that is $\prod_{f_i \in \mathbf{f}} f_i(\mathbf{x}_s) = \prod_{G^i \in \mathcal{G}_c} \prod_{f_i \in \mathbf{f}^i} f_i(\mathbf{x}_s)$. We define $s^i(\mathbf{x}^i) = \prod_{f_i \in \mathbf{f}^i} f_i(\mathbf{x}_s)$ as the local score of G^i.

Generalized max-product can be understood as the max-product algorithm applied to $\mathcal{G}_c$. Rather than computing distributions over single variables, the algorithm returns a distribution over the joint assignment to variables in each cluster. When clusters correspond to subproblems of the original input, this can be understood as computing a distribution over solutions to each subproblem, then iteratively modifying the solutions of each subproblem to reflect their viability as part of a global solution. The message transmitted from cluster i to cluster j is

$$m_{i \to j}(\mathbf{X}^{ij} = \mathbf{x}^{ij}) = \max_{\mathbf{x}^i : \mathbf{X}^{ij} = \mathbf{x}^{ij}} s^i(\mathbf{x}^i) \prod_{k \neq j} m_{k \to i}(\mathbf{x}^{ik}) \tag{4}$$

When $\mathcal{G}_c$ contains cycles, these messages are iteratively computed until convergence, at which point the *belief* of cluster G^i is defined as

$$b_i(\mathbf{X}^i = \mathbf{x}^i) = s^i(\mathbf{x}^i) \prod_j m_{j \to i}(\mathbf{x}^{ij}) \tag{5}$$

Thus, $m_{i \to j}(\mathbf{X}^{ij})$ is a vector representing a distribution over scores for assignments of $\mathbf{X}^{ij}$, and $b_i(\mathbf{X}^i)$ is a vector representing a distribution over scores for $\mathbf{X}^i$.

When $\mathcal{G}_c$ is acyclic, the resulting beliefs are exact; otherwise, generalized max-product offers no better theoretical

x_1	x_2	x_3	$m(\mathbf{x})$
0	0	0	9
0	0	1	10
0	1	0	8
0	1	1	4
1	0	0	7
1	0	1	9
1	1	0	15
1	1	1	6

(a)

x_1	$m(x_1)$	x_2	$m(x_2)$	x_3	$m(x_3)$
0	7	0	3	0	2
1	8	1	9	1	7

(b)

x_1	x_2	x_3	$m(\mathbf{x})$
1	1	0	15
0	0	1	10
0	0	0	9
1	0	1	9

(c)

Table 1. Three message representations: (a) complete table, (b) univariate table, (c) n-best list.

guarantees than traditional max-product. However, generalized belief propagation has been shown to often have better empirical convergence rates than traditional belief propagation (Yedidia et al., 2000). More importantly, generalized max-product is appealing for clustered WMAX-SAT instances because it directly accounts for problem structure. It also allows us to substitute specialized inference algorithms (when they exist) to efficiently compute solutions to subproblems.

There are two primary difficulties with applying generalized max-product to WMAX-SAT. First, the size of message $m_{i\to j}(\mathbf{X}^{ij})$ grows exponentially with $|\mathbf{X}^{ij}|$. Second, the maximization over $\mathbf{x}^i$ in Equation 4 naively requires an exponential search over assignments to $\mathbf{X}^i$. As it is not uncommon for $|\mathbf{X}^i|$ to be on the order of thousands of variables, we must consider alternatives to brute-force enumeration. The following sections address these issues in turn.

4.1. Representing sparse beliefs and messages

Ordinarily, $m_{i\to j}(\mathbf{X}^{ij})$ and $b_i(\mathbf{X}^i)$ are represented by a table with one entry per complete assignment to $\mathbf{X}^{ij}$ or $\mathbf{X}^i$. We refer to these as *complete messages* and *complete beliefs* (Table 1(a)).

When the complete messages grow too large, a simple approximation is to store a fully-factored message containing one score distribution per variable. Each table entry is the score of the best complete assignment containing the specified variable assignment. We refer to these as *univariate messages* and *univariate beliefs* (Table 1(b)). This is similar to the messages advocated recently in Duchi et al. (2007).

While univariate messages are compact, they do not represent interactions among variables, especially when they must be approximated with message passing. For example in Table 1(b), although each individual variable has a high score for assignment 1, the joint assignment 111 has one of the lowest scores in the complete message in 1(a).

A compromise between these two extremes is a particle-based message. In this case, a particle is simply a row of the complete table, and a message is approximated by a collection of particles. A simple special case of a particle message is an n-best list (Table 1(c)). An n-best list stores scores for joint assignments as in complete messages, but the low scores are simply truncated from the table. This representation allows the messages to be sensitive to variable interactions while limiting memory requirements. We refer to these as *n-best messages* and *n-best beliefs*.

4.2. Computing sparse beliefs and messages

In this section we present an algorithm to pass sparse messages within generalized max-product. The first step is to initialize the sparse belief $b_i(\mathbf{X}^i)$ for each cluster. If a cluster has a specialized maximization algorithm such as a dynamic program, we can modify this algorithm to return the desired sparse representation. For example, to compute an n-best belief, we can simply compute the top n solutions using the maximization algorithm. If no specialized algorithm exists, we can treat this computation as a generic WMAX-SAT problem and use any off-the-shelf solver. In our experiments, we use Walksat (Jiang et al., 1995), a highly-scalable stochastic solver. We refer to the method that computes these initial beliefs as *ComputeUnconstrainedBeliefs*.

Given this initial assignment, the next step is to update each cluster's sparse beliefs given the sparse beliefs of its neighboring clusters. Let $\mathbf{X}^i \setminus \mathbf{X}^{ik}$ be the set of variables contained in cluster G^i but not in G^k. As shown in Equation 4, to compute message $m_{i\to j}(\mathbf{X}^{ij})$, we must find the maximum setting of $\mathbf{X}^i \setminus \mathbf{X}^{ik}$ *given* each assignment to $\mathbf{X}^{ik}$ stored in the incoming message from G^k.

We view this as a set of *constrained maximization problems*. For each incoming assignment to $\mathbf{X}^{ik}$, we must find the best setting(s) of $\mathbf{X}^i \setminus \mathbf{X}^{ik}$ *constrained* such that $\mathbf{X}^{ik} = \mathbf{x}^{ik}$. The local score of this best assignment is combined with the score of the incoming message to update $b_i(\mathbf{X}^i)$.

If a cluster has a specialized maximization algorithm, we can modify it to return constrained solutions, e.g. by constrained dynamic programming. Otherwise, we can treat this constrained maximization problem as a WMAX-SAT instance. Given an assignment $\mathbf{x}^{ik}$, we can simplify the factors in G^i (i.e. by removing clauses in G^i satisfied by $\mathbf{x}^{ik}$ and removing variables in $\mathbf{X}^{ik}$ from unsatisfied clauses).

Algorithm 1 Sparse Generalized Max-Product

// Initialize beliefs
for all $G^i \in \mathcal{G}_c$ **do**
  $b_i(\mathbf{X}^i) \Leftarrow$ *ComputeUnconstrainedBelief*(G^i)
end for
while Not Converged **do**
  for all $G^i \in \mathcal{G}_c$ **do**
    // for each neighboring cluster
    for all G^j s.t. $\mathbf{X}^{ij} \neq \emptyset$ **do**
      // for each incoming assignment
      for all $\mathbf{x}^{ij}$ s.t. $m_{j \to i}(\mathbf{X}^{ij} = \mathbf{x}^{ij}) \neq 0$ **do**
        ComputeConstrainedBelief$(G^i, \mathbf{x}^{ij})$
      end for
    end for
  end for
end while

We can now again use any off-the-shelf solver to compute constrained solutions (e.g. Walksat). We refer to the method that solves this constrained maximization problem as *ComputeConstrainedBelief*.

There are two additional details about *ComputeConstrainedBelief*. First, very often we can simply lookup the best assignment to $\mathbf{X}^i \setminus \mathbf{X}^{ik}$ given $\mathbf{x}^{ik}$ because a compatible solution may already exist in $b_i(\mathbf{X}^i)$. Second, because each round of message passing may increase the size of $b_i(\mathbf{X}^i)$, we may need to re-compress the belief, for example, by storing only the n-best assignments or recomputing the univariate values.

Message passing proceeds until either a specified number of iterations is reached or the difference in beliefs between iterations falls below a threshold. If the top beliefs of each cluster are incompatible at the end of message passing, we can use a decimation procedure similar to that used in Braunstein et al. (2005), although here we can decimate using clusters, rather than individual variables. Pseudo-code for the final algorithm is presented in Algorithm 1.

5. Experiments

The goal of these experiments is to understand under what conditions augmenting an off-the-shelf WMAX-SAT solver with a message passing protocol can improve performance. We vary both the characteristics of the problems as well as the type of message representation. In all experiments, we use the weighted version of Walksat (Jiang et al., 1995), which we ported to Java from the original C implementation[1]. Note that Walksat is used both for comparison and as a subroutine to compute messages.

For the first set of experiments, we randomly generate

[1] `www.cs.rochester.edu/u/kautz/walksat/`

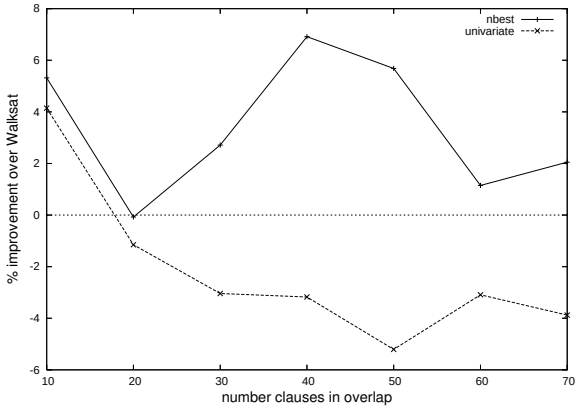

Figure 2. Comparison of n-best and marginal messages as the number of clauses containing shared variables increases.

WMAX-SAT instances that exhibit a clustering of variables. We implement a variant of the *Cluster3-SAT* problem generator given in Frank et al. (1997). We first generate k 3-SAT subproblems uniformly at random, then randomly link the problems together by generating clauses that contain variables from each subproblem. Weights for each clause are sampled from a Gaussian distribution. For these experiments, $n = 50$.

To compare our message passing algorithms with Walksat, we first run the message passing algorithm and record the score of the best assignment and the total number of Walksat iterations it required. We then run traditional Walksat on the original problem for the same number of iterations and record the score of the best assignment. This allows us to measure the efficiency of each search algorithm. For each setting of the problem generator, we generate 100 samples and average the results. In the following experiments, we compare performance across different message types, and across different numbers of variables, clauses, and clusters.

Figure 2 compares univariate and n-best messages on 700 instances with 100 variables, 400 clauses, and 2 clusters. The x-axis is the number of clauses containing variables that are shared between the two clusters. This figure supports our intuition that as the overlapping variables become more constrained, representing their distribution with a univariate message fails to capture the compatibilities among variables. We do not have a full explanation for the drop in performance for 20 clauses, although it may be related to the phase transition phenomenon that has been observed in unweighted satisfiability problems as the relative number of constraints increases.

In Figure 3, we test the hypothesis that message passing

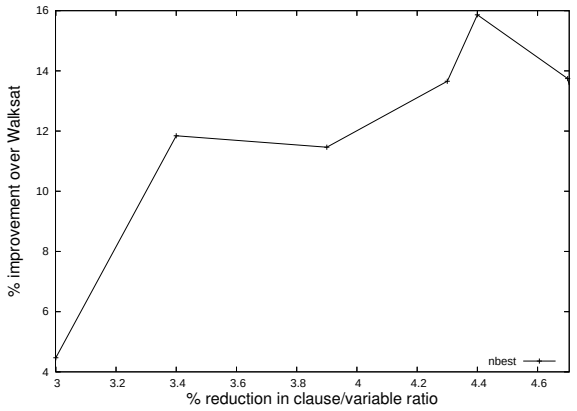

Figure 3. Correlation between reduction in clause-variable ratio and score improvement for n-best messages.

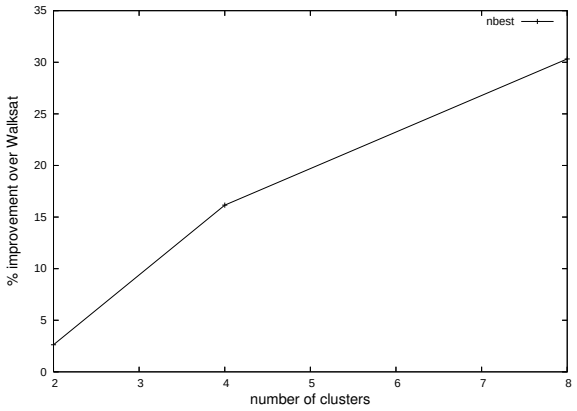

Figure 4. Comparison of improvement of n-best over Walksat as the number of clusters increases.

outperforms Walksat because of a reduction in the clause-variable ratio. Recall that the cluster graph is constructed such that two clusters may share variables but not factors. Thus, the average clause-variable ratio per cluster is less than that of the original problem. Figure 3 plots the percent improvement of n-best message passing over Walksat as the reduction in the clause-variable ratio increases. These experiments sample 175,000 different instances containing 80 to 180 variables, 300 to 600 clauses, and 2 clusters. The results provide strong correlational evidence in support of our hypothesis. Note, however, that the downward slope when $x > 4.4$ may indicate a ceiling to this trend.

Figure 4 plots performance as the number of clusters increases. We generate 300 instances with 400 variables, 1600 clauses, and the number of clusters varying from 2 to 8. In each case, the clusters are fully connected, with each pair of clusters sharing two variables. Figure 4 shows a significant increase in performance as the divisibility of the problem increases. Note that this improvement is amplified by the relatively small number of shared variables, which reduces the chances of a good solution being excluded by message compression.

Figure 5 plots performance as the number of variables shared between clusters increases. We generate 700 instances with 1000 variables, 4000 clauses, and 2 regions, and vary the number of shared variables from 5 to 100. This figure indicates that performance is best with few shared variables, but message passing still outperforms Walksat even as the potential message size increases to 2^{100}.

Finally, we test our algorithm on a real-world WMAX-SAT instance that is part of the benchmarks used in the MaxSAT-2006 competition.[2] In particular, we sample a problem from the *SPOT5* corpus, which was generated from a satellite scheduling domain (Bensana et al., 1999). To partition the problem into clusters, we use a well-know graph-theoretic technique based on betweenness-centrality (Tyler et al., 2003).

Figure 6 shows results averaged over 100 trials for an instance with 129 variables and 1037 clauses. We set $n = 100$. We can see that univariate message passing improves slightly over Walksat when the problem is split into 2 clusters, but performs worse with larger clusters. On the other hand, n-best message passing outperforms Walksat when there are 3 and 4 clusters. This is appealing, because an examination of the factor graph for this problem reveals 4 well-defined clusters.

6. Conclusions and Related Work

We have presented a new message passing algorithm to solve maximization problems that consist of multiple overlapping subproblems. We have empirically demonstrated the validity of this approach on synthetic and real-world data, and have evaluated hypotheses to explain its performance.

In the future, we plan to explore more sophisticated sparse message representations (e.g., Drechsler and Sieling (2001)) and to apply our algorithms to real-world NLP problems.

Related work on decomposing maximization problems include Dechter and Pearl (1989), who propose tree-based

[2] `www.iiia.csic.es/~maxsat06/ms06-bench.tgz`

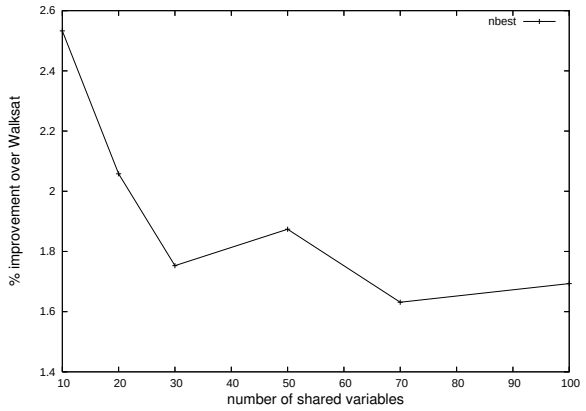

Figure 5. Comparison of improvement of n-best over Walksat as the number of variables shared between clusters increases.

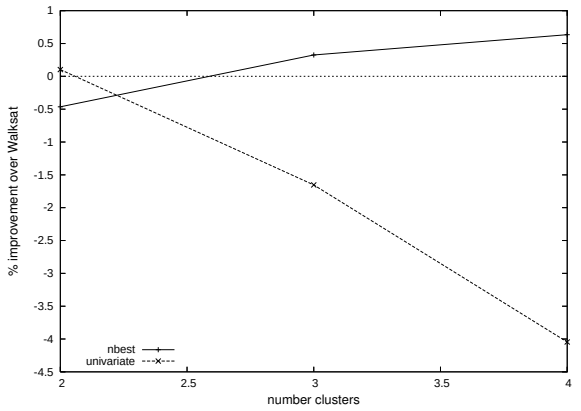

Figure 6. Comparison of Walksat and message passing algorithms on *SPOT5* data.

decompositions for constraint networks; Bjesse et al. (2003), who propose tree-based decompositions for (unweighted) SAT problems; and Amir and McIlraith (2005), who propose message passing algorithms for first-order logic. These approaches consider neither weighted maximization problems nor sparse message computation. Additionally, Braunstein et al. (2005) recently proposed an algorithm for unweighted SAT called *survey propagation*. Survey propagation can be understood as a variant of traditional (non-generalized) belief propagation, and therefore may not be well-suited for clustered SAT instances.

Sparse message passing has been studied previously, though mainly for real-valued domains (Koller et al., 1999; Sudderth et al., 2003). Related work in discrete domains include Pal et al. (2006), who introduce a learning algorithm for linear-chain models with large state spaces based on a variant of beam-search; and Duchi et al. (2007), who propose combining dynamic programming with message passing using univariate messages.

7. Acknowledgments

The authors thank Carla Gomes, Chris Pal, and Charles Sutton for helpful discussions. This work was supported in part by the Center for Intelligent Information Retrieval, in part by U.S. Government contract #NBCH040171 through a subcontract with BBNT Solutions LLC, in part by Microsoft Live Labs, and in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

References

Amir, E., & McIlraith, S. (2005). Partition-based logical reasoning for first-order and propositional theories. *Artif. Intell.*, *162*, 49–88.

Bensana, E., Lemaitre, M., & Verfaillie, G. (1999). Earth observation satellite management. *Constraints*, *4*, 293–299.

Bjesse, P., Kukula, J., Damiano, R., Stanion, T., & Zhu, Y. (2003). Guiding SAT diagnosis with tree decompositions. *SAT 2003*.

Braunstein, A., Mézard, M., & Zecchina, R. (2005). Survey propagation: an algorithm for satisfiability. *Random Structures and Algorithms*, *27*, 201–226.

Dechter, R., & Pearl, J. (1989). Tree clustering for constraint networks. *Artificial Intelligence*, *38*, 353–266.

Drechsler, R., & Sieling, D. (2001). Binary decision diagrams in theory and practice. *Int J STTT*, *3*, 112–136.

Duchi, J., Tarlow, D., Elidan, G., & Koller, D. (2007). Using combinatorial optimization within max-product belief propagation. *Advances in Neural Information Processing Systems (NIPS 2006)*.

Frank, J., Cheeseman, P., & Stutz, J. (1997). When gravity fails: Local search topology. *Journal of Artificial Intelligence Research*, *7*, 249–281.

Jiang, Y., Kautz, H., & Selman, B. (1995). Solving problems with hard and soft constraints using a stochastic al-

gorithm for max-sat. *1st Workshop on Artificial Intellignce and Operations Research*.

Koller, D., Lerner, U., & Angelov, D. (1999). A general algorithm for approximate inference and its application to hybrid bayes nets. *UAI*.

Pal, C., Sutton, C., & McCallum, A. (2006). Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Park, J. D. (2002). Using weighted max-sat engines to solve mpe. *AAAI/IAAI* (pp. 682–687).

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, *62*, 107–136.

Selman, B., Kautz, H. A., & Cohen, B. (1993). Local search strategies for satisfiability testing. *Proceedings 1993 DIMACS Workshop on Maximum Clique, Graph Coloring, and Satisfiability*.

Stützle, T., Hoos, H., & Roli, A. (2002). *A review of the literature on local search algorithms for max-sat* (Technical Report AIDA-01-02). Technische Universität Darmstadt.

Sudderth, E., Ihler, A., Freeman, W., & Willsky, A. (2003). Nonparametric belief propagation. *CVPR*.

Tyler, J. R., Wilkinson, D. M., & Huberman, B. A. (2003). *Email as spectroscopy: Automated discovery of community structure within organizations* (Technical Report). Hewlett-Packard Labs.

Weiss, Y., & Freeman, W. (2001). On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, *47*.

Yedidia, J. S., Freeman, W. T., & Weiss, Y. (2000). Generalized belief propagation. *NIPS* (pp. 689–695).