MAP estimation in MRFs via rank aggregation

Rahul Gupta[1] RAHULGUPTA@IN.IBM.COM
IBM Research Lab, New Delhi, India

Sunita Sarawagi SUNITA@IITB.AC.IN
IIT Bombay, India

Abstract

Efficient estimation of the maximum a priori (MAP) assignment in large statistical relational networks still remains an open issue in spite of the extensive research in this area. We propose a novel method of exploiting top-K MAP estimates from simpler subgraphs to find an assignment that is either MAP optimal, or has an associated bound on how far it is from the optimal. Our method extends the well-known tree reweighted max-product algorithm (TRW) and is guaranteed to always provide tighter upper bounds. Experiments on synthetic and real data show that we are able to the find the optimal in many more cases than TRW, at significantly fewer iterations and our bounds are much tighter than those provided by TRW.

1. Introduction

Many applications of statistical relational learning give rise to large and complex graphical models where collective inference is useful (Jensen et al., 2004) yet finding the optimum MAP labeling is challenging. Examples include, classification of hyperlinked documents (Chakrabarti et al., 1998; Taskar, 2004; Lu & Getoor, 2003), collective information extraction (Bunescu & Mooney, 2004; Finkel et al., 2005; Wellner et al., 2004; Mansuri & Sarawagi, 2006), and record linkage (Parag & Domingos, 2004; McCallum & Wellner, 2003; Bilenko, 2004). The complexity of estimating the MAP assignment on a graph with arbitrary dependency potentials is exponential in the size of the largest clique of the graph. In some cases it is possible to exploit the special form of potentials to design optimal MAP algorithms like the mincut algorithm (Kolmogorov & Zabih, 2004) for associative potentials and binary labels. In most other cases, various forms of the max-product belief propagation algorithms is used to get an approximate assignment. The belief propagation (BP) algorithm converges to the optimal MAP assignment on trees and on graphs with at most a single loop and a unique MAP (Wainwright et al., 2004; Weiss & Freeman, 2001). However, in the general case the basic BP algorithm may not converge, and when it does, the maximum score from the pseudo max-marginals might be unrelated to the optimal. The tree reweighted max-product algorithm (TRW) (Wainwright et al., 2005) is a variant of the BP algorithm that conceptually decomposes the original potential as a convex combination of tree-structured potentials. It then uses slightly modified message passing steps to reach consensus amongst the max-marginals of the tree edges. The algorithm can provide an upper bound on the optimal score and this bound is tight if and only if all trees agree on a MAP assignment. However, there is no guarantee that such an agreement will always be achieved. On termination we are left with the difficult problem of choosing a good assignment. If the algorithm converges so that the max-marginal beliefs are calibrated or a local optimum is detected via a weak tree agreement, then the beliefs can be used to construct a good and sometimes optimal solution (Meltzer et al., 2005; Kolmogorov & Wainwright, 2005).

Our method is an enhancement of the TRW method where we decompose the set of potentials into an additive combination of potentials over edge sets. We find the top K highest scoring assignments in each set using a straightforward extension of the max-product algorithm (Nilsson, 1998; Yanover & Weiss, 2003). We then perform a rank aggregation operation (Fagin et al., 2001) on the output assignments from each set

[1]Work done as a student at IIT Bombay

Presented at the ICML Workshop on Open Problems in Statistical Relational Learning, Pittsburgh, PA, 2006.

to obtain an optimal MAP or improved bounds than possible by TRW. The TRW method provides tight bounds only when the trees agree on a MAP assignment. In contrast, we dynamically compute bounds from the rank aggregation step to detect an optimal solution from the top K MAP assignments of trees. This enables us to find tight bounds in many more cases and in fewer iterations as we show in our experimental results over synthetic and real graphs.

We also discuss how the ranked aggregation framework can enable us to find the optimal MAP in some applications by combining the benefits of reparameterization offered by belief propagation algorithms with the guarantee of optimality offered by mincut algorithms on associative potentials. We conclude with a listing of open problems in this area.

2. Background: Markov Random Fields

We assume a graph $G = (V, E)$ where nodes correspond to output variables $\mathcal{X}_1 \ldots \mathcal{X}_n$ and edges correspond to pairs of interacting variables. Any MRF can be expressed using pairwise potentials alone (Weiss & Freeman, 2001). We use the notation of (Wainwright et al., 2005) where ϕ is a vector of potentials defined over elements of $\mathcal{X}^n = \mathcal{X}_1 \times \mathcal{X}_2 \ldots \mathcal{X}_n$ where each ϕ_α is defined either over a single variable $s \in V$ with a label $x_s \in \mathcal{X}_s$ or a variable pair $(s,t) \in E$ with a label pair $(x_s, x_t) \in \mathcal{X}_s \times \mathcal{X}_t$. Thus, $\phi : \mathcal{X}^n \rightarrow \mathbf{R}^d$, where d denotes the dimensionality of ϕ. Let θ denote the vector of parameters corresponding to the potential $\phi()$. The score of an assignment $\mathbf{x} \in \mathcal{X}^n$ is

$$\text{score}(\mathbf{x}) = \langle \phi(\mathbf{x}), \theta \rangle = \sum_\alpha \theta_\alpha \phi_\alpha(\mathbf{x})$$

The MAP estimation problem in a MRF is to find a highest scoring assignment $\mathbf{x}^{\text{map}} \in \mathcal{X}^n$

$$\mathbf{x}^{\text{map}} = \text{argmax}_{\mathbf{x} \in \mathcal{X}^n} \langle \phi(\mathbf{x}), \theta \rangle$$

When G does not have any cycles, $\mathbf{x}^{\text{map}}$ can be found in linear time using the well-known max-product belief propagation algorithm (Pearl, 1988). For general graphs with cycles and arbitrary potentials, the MAP can be found optimally using the max-product algorithm on a clique tree created from the triangulated graph. The complexity of this algorithm is exponential in the size of the largest clique in the graph. Therefore, a number of approximate methods have been proposed, most of which are variants of the belief propagation algorithm but applied to general graphs with cycles (Wainwright et al., 2005; Kolmogorov, 2004; Pearl, 1988). Of these the various tree-reweighted belief propagation algorithms (Wainwright et al., 2005; Kolmogorov, 2004) provide upper bounds on the value of MAP when they fail to find the optimal. Our goal is to improve on these bounds.

The output of our algorithm is either a MAP assignment $\mathbf{x}^{\text{map}}$ or an approximate MAP assignment $\mathbf{x}^*$ along with a **gap** G such that $\langle \phi(\mathbf{x}^{\text{map}}), \theta \rangle - \langle \phi(\mathbf{x}^*), \theta \rangle \leq G$.

3. MAP estimation via rank aggregation

Let $S_1 \ldots S_L$ denote sets of collection of edges over the MRF graph G such that each edge is included in at least one set and all nodes are included in all sets. In this section we will assume that each of the sets S_i defines a spanning tree over G. In Section 5 we discuss generalizations to other cases.

We choose parameters $\theta(S_1) \ldots \theta(S_L)$ over the sets such that the sum of these parameters are *equivalent* to the given parameters $\bar{\theta}$ of G. Thus,

$$\sum_i \theta(S_i) \equiv \bar{\theta}; \quad \theta_\alpha(S_i) = 0 \ \ \forall \alpha \notin S_i \tag{1}$$

The equivalence of two sets of parameters θ^1 and θ^2 means that $\forall \mathbf{x}$, $\langle \phi(\mathbf{x}), \theta^1 \rangle = \langle \phi(\mathbf{x}), \theta^2 \rangle$. A simple mechanism to ensure such equivalence is to set $\theta_\alpha(S_i) = \frac{\bar{\theta}_\alpha}{n_\alpha} \ \ \forall \alpha \in S_i$ where n_α is the number of sets that contain α. The optimal choice of sets S_i and parameters $\theta(S_i)$ is an open problem (Wainwright et al., 2005) [2]

On each of the subsets S_i we find a ranked list of assignments $R_i = \mathbf{x}_1^i, \ldots \mathbf{x}_{k_i}^i$ and an upper bound $\mathbf{ub}_i$ on the maximum value of $\langle \phi(\mathbf{x}), \theta(S_i) \rangle$ over assignments not in the list R_i. That is,

$$\mathbf{ub}_i \geq \max_{\mathbf{x} \notin R_i} \langle \phi(\mathbf{x}), \theta(S_i) \rangle \tag{2}$$

Since each set S_i is a spanning subset of the graph, an assignment $\mathbf{x}_j^i$ in each R_i is also a complete assignment of labels to all nodes in graph G. For trees, it is easy to extend the max-product algorithm to find the top K highest scoring assignments in $O(nK \log(K+1))$ time (Nilsson, 1998). The value of $\langle \phi(\mathbf{x}_K^i), \theta(S_i) \rangle$ provides the upper bound $\mathbf{ub}_i$. Also, for trees these bounds are *tight* in the sense that $\mathbf{ub}_i \leq \min_{\mathbf{x} \in R_i} \langle \phi(\mathbf{x}), \theta(S_i) \rangle$.

[2] (Wainwright et al., 2005) defines a similar decomposition over trees but they also have a parameter ρ to express $\bar{\theta}$ as a convex combination over tree distribution. Given the linear form of the MAP function, such a convex combination is not strictly needed for any of our bounds, we therefore do not use them to keep our notation simple.

We evaluate the value of $\langle \phi(\mathbf{x}), \theta \rangle$ for each assignment $\mathbf{x}$ in the lists $R = R_1 \cup \ldots \cup R_L$ in a suitable order that makes it more likely to find optimal assignments earlier on. Some of the assignments could be common across multiple lists. Let $\mathbf{x}^*$ be the assignment with the highest score. Thus,

$$\mathbf{x}^* = \text{argmax}_{\mathbf{x} \in R_1 \cup \ldots \cup R_L} \langle \phi(\mathbf{x}), \theta \rangle \quad (3)$$

Define the upper bound $\mathbf{ub}$ as $\sum_i \mathbf{ub}_i$.

Theorem 3.1. *If $\langle \phi(\mathbf{x}^*), \theta \rangle \geq \mathbf{ub}$, then $\mathbf{x}^*$ is MAP optimal, otherwise $\langle \phi(\mathbf{x}^{map}), \theta \rangle \leq \mathbf{ub}$.*

Proof.

$$\begin{aligned} \langle \phi(\mathbf{x}^{\text{map}}), \theta \rangle &= \max(\max_{\mathbf{x} \notin R} \langle \phi(\mathbf{x}), \theta \rangle, \max_{\mathbf{x} \in R} \langle \phi(\mathbf{x}), \theta \rangle) \\ &= \max(\max_{\mathbf{x} \notin R} \langle \phi(\mathbf{x}), \theta \rangle, \langle \phi(\mathbf{x}^*), \theta \rangle) \end{aligned}$$

We next prove that $\max_{\mathbf{x} \notin R} \langle \phi(\mathbf{x}), \theta \rangle \leq \mathbf{ub}$

$$\begin{aligned} \max_{\mathbf{x} \notin R} \langle \phi(\mathbf{x}), \theta \rangle &= \max_{\mathbf{x} \notin R} \sum_i \langle \phi(\mathbf{x}), \theta(S_i) \rangle \quad \text{(from 1)} \\ &\leq \sum_i \max_{\mathbf{x} \notin R_i} \langle \phi(\mathbf{x}), \theta(S_i) \rangle \quad (R_i \subseteq R) \\ &\leq \sum_i \mathbf{ub}_i = \mathbf{ub} \quad \text{(from 2)} \end{aligned}$$

Thus, $\langle \phi(\mathbf{x}^{\text{map}}), \theta \rangle \leq \max(\mathbf{ub}, \langle \phi(\mathbf{x}^*), \theta \rangle)$ By definition of MAP $\langle \phi(\mathbf{x}^*), \theta \rangle \leq \langle \phi(\mathbf{x}^{\text{map}}), \theta \rangle$. If $\langle \phi(\mathbf{x}^*), \theta \rangle \geq \mathbf{ub}$, then $\langle \phi(\mathbf{x}^*), \theta \rangle = \langle \phi(\mathbf{x}^{\text{map}}), \theta \rangle$, otherwise $\langle \phi(\mathbf{x}^{\text{map}}), \theta \rangle \leq \max(\mathbf{ub}, \langle \phi(\mathbf{x}^*), \theta \rangle) = \mathbf{ub}$ □

Algorithm 1 shows our ranked aggregation algorithm that returns an assignment $\mathbf{x}^*$ and an upper bound on gap between the scores of $\mathbf{x}^{\text{map}}$ and $\mathbf{x}^*$. In this algorithm, we allow the ranked assignments in R_i and the bounds $\mathbf{ub}_i$ to be refined incrementally rather than instantiate all of R_i in advance. This form of lazy evaluation allows for the possibility of generating fewer than K assignments if an optimal is found before it. We implemented an online version of the $Top-K$ max-product algorithm that does not require a fixed K to be specified a priori and can incrementally find the next highest scoring assignment. Another method of reducing the number of assignments evaluated is to order the lists such that lists that are likely to contain the MAP are placed earlier in the ordering.

At this point it is useful to compare the above results with the results in (Wainwright et al., 2005) which relies on a similar decomposition of potentials over trees. In (Wainwright et al., 2005) optimality is detected if and only if all trees agree on a MAP assignment. For the case of trees and with $K = 1$ we would detect optimality in exactly the same set of cases but our check for optimality is faster since it does not require enumeration of all MAP assignments of a tree which in degenerate cases can be quite large. However, as we show in the experiments there are many cases where we find the optimal solution even when the optimal is not in the topmost position of any of the lists. This sometimes enables us to find the optimal even when the belief propagation algorithm does not converge. Also, the bounds we return are guaranteed to be tighter. We can make the following additional claim about the rank-aggregation algorithm.

Corollary 3.1. *When an assignment appears in each list R_i and the bounds $\mathbf{ub}_i$ are tight, we are guaranteed to return a $\mathbf{x}^{map}$ from R. However, it is not necessary for the R_is to have a common assignment in order to find a $\mathbf{x}^{map}$.*

Algorithm 1 Rank-aggregate($S_1 \ldots S_L, \theta$)

Choose $\theta(S_i)$ $i = 1 \ldots L$: such that $\sum_i \theta(S_i) \equiv \theta$
$R_i, \mathbf{ub}_i$: Ranked assignment lists and bounds from each S_i
while lists not exhausted **do**
  i = index of first list with unseen assignments, advance to next $\mathbf{x}^i$ in R_i and get new $\mathbf{ub}_i$.
  $\mathbf{ub} = \sum_i \mathbf{ub}_i$
  $\mathbf{x}^* = \mathbf{x}^i$ if $\mathbf{x}^*$ uninitialized or $\langle \phi(\mathbf{x}^i), \theta \rangle > \langle \phi(\mathbf{x}^*), \theta \rangle$
  if $\langle \phi(\mathbf{x}^*), \theta \rangle \geq \mathbf{ub}$ **then**
    $\mathbf{x}^*$ is MAP optimal. **return** ($\mathbf{x}^*$,0).
  end if
end while
return ($\mathbf{x}^*$, $\mathbf{ub} - \langle \phi(\mathbf{x}^*), \theta \rangle$).

3.1. Reparameterization

The chances of reaching agreement amongst trees increases if we reparameterize the distribution so that the new parameters reflect the (max) marginal probability of the respective node and edge assignments. This makes it more likely that local modes from each set will agree with global modes. All variants of max-product belief propagation algorithms strive to achieve such reparameterization (Wainwright et al., 2005; Kolmogorov, 2004; Pearl, 1988). Of these the tree reweighted algorithms (Kolmogorov, 2004; Wainwright et al., 2005) are better suited to reaching agreement amongst a set of tree structured distributions.

Our final method is given in Algorithm 2. In each iteration, we first invoke the rank aggregation algorithm to find a MAP. If that fails, we apply one round of reparameterization to get a new value of θ for which

the value of $\langle \phi(\mathbf{x}), \theta \rangle$ is equal to the old value. We consider three different tree-based reparameterization techniques as discussed in the next section.

Algorithm 2 Find-MAP($S_1 \dots S_L, \bar{\theta}$)

$\theta^1 = \bar{\theta}$
for iterations $n = 1, 2 \dots$ **do**
  $(\mathbf{x}^*, \text{gap}) = \text{Rank-aggregate}(S_1 \dots S_L, \theta^n)$
  if gap is 0, i.e., $\mathbf{x}^*$ is optimal **then**
    return $(\mathbf{x}^*, 0)$
  end if
  $\theta^{n+1} = \text{reparameterize}(\theta^n, S_1 \dots S_L)$
  if θ^n same as θ^{n+1} **then**
    return ($\mathbf{x}^*$,gap)
  end if
end for

4. Experiments

We present empirical results of rank-aggregation over some synthetic and real-data. We ran three variants of the reparameterization algorithm — the edge-based (TRW-Edge) and tree-based (TRW-Tree) algorithms discussed in (Wainwright et al., 2005) and the sequential update algorithm (TRW-Sequential) described in (Kolmogorov, 2004). Of these, only TRW-Sequential is guaranteed to return monotonically decreasing bounds with every iteration and none of the algorithms is guaranteed to converge to a zero gap and give the global optimum. For TRW-Edge and TRW-Tree, at all iterations, we returned the best of the top-1 tree assignments as the MAP assignment. This assignment will be the global optimum if tree-agreement occurs. In the case of TRW-Sequential, we used the greedy scheme discussed in (Kolmogorov, 2004) to obtain the current estimate of the MAP assignment. In our experiments we observed that this scheme is generally superior to the "best of the best tree assignments" approach.

In practice, TRW-Tree and TRW-Edge often suffer from oscillations, so a damping parameter of 0.5 was used in their update steps. For all the three variants, we fixed the set of trees apriori.

Synthetic graphs: We generated 50 instances each of 10×10 grids and 15-node cliques. The potentials of these graphs correspond to the Ising model. The cases where these potentials are attractive and mixed were considered separately. The node potentials in both the cases were chosen u.a.r. from $[-1, 1]$. For the attractive case, edge potentials were chosen u.a.r. from $[0, 1]$, while for the mixed case, they were chosen from $[-0.5, 0.5]$.

For the grid graphs, we used two spanning trees — one comprising of all rows and one column, and the other comprising of all columns and one row. For the 15-node cliques, we used 15 spanning trees, each one of which was a star rooted at a distinct node. Each algorithm was run under a limit of 70 iterations and for rank aggregation we chose $K = 10$.

Table 1 compares the proposed rank aggregation algorithm with the three tree agreement algorithms that detect optimality only when trees agree on a MAP. Our results are averaged over 50 graphs in each family. Some important observations:

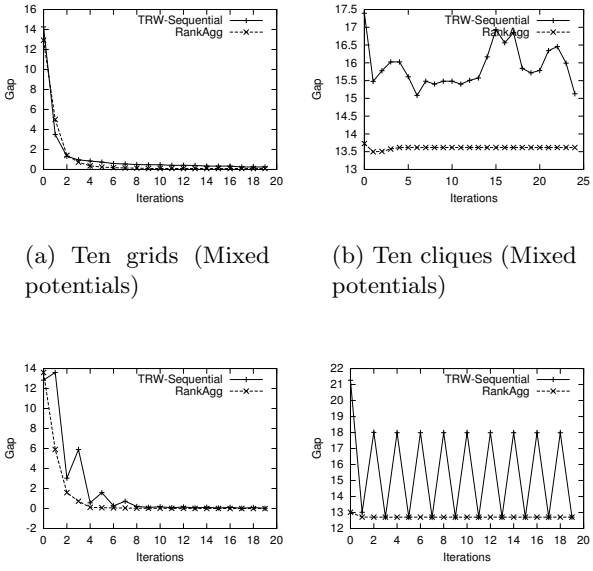
- Rank-aggregation finds the optimal in many cases where tree agreement cannot even after 70 iterations, the exception being cliques with mixed potentials, where no algorithm succeeds. The TRW-Tree algorithm is the biggest beneficiary of Rank-aggregation in almost all the cases.
- Rank-aggregation takes much fewer iterations to find the optimal. For cliques with attractive potentials, it always succeeds without any need for reparameterization. Even in the other cases, it takes just 50-70% of the iterations required by the tree-agreement algorithms. Although rank aggregation takes slightly longer per iteration in finding the top-K rather than just the top-1 assignment, the reduced number of iterations more than pays off for the extra time.
- Whenever rank-aggregation fails, the gap is significantly lower than that obtained by either of the tree-agreement algorithms.[3] For example, in cliques with mixed potentials the average gap of TRW-Sequential is reduced from 15.5 to 12.4. The results also confirm that in many cases, the MAP assignment may not be top-1 assignment for any tree. Hence, we observe higher average MAP scores returned by top-K in the case of failures. The difference is substantial in the case of cliques with mixed potentials (e.g. 9.7 vs 12.8 in TRW-Edge).
- The performance of TRW-Sequential is clearly superior to the other two tree-algorithms. However, here too, rank-aggregation further provides a significant improvement, mainly by greatly reducing the gap returned in the case of failures and cutting down the number of iterations required. In

[3] In some cases, like TRW-Edge over grids with mixed potentials, it may appear that the gap worsens with rank-aggregation. However, if we average only over the cases where top-K fails, then the gap reduces with rank-aggregation.

Family	TRW-Tree		TRW-Edge		TRW-Sequential	
	TreeAgree	RankAgg	TreeAgree	RankAgg	TreeAgree	RankAgg
Grid (Attr.)	0 (-) 50 (25.8,93.9)	0 (-,-) 50 (23.8,95.5)	30 (16) 20 (1.8,96.2)	38 (15,7) 12 (1.6,96.7)	50 (9) 0 (-,-)	50 (7,6) 0 (-,-)
Grid (Mixed)	30 (17) 20 (0.6,61.8)	37 (12,8) 13 (0.5,61.5)	32 (13) 18 (0.7,61.8)	40 (10,6) 10 (0.7,61.4)	39 (8) 11 (0.8,60.8)	43 (6,6) 7 (0.7,61.2)
Clique (Attr.)	15 (2) 35 (8.6,54.2)	50 (1,2) 0 (-,-)	50 (13) 0 (-,-)	50 (1,2) 0 (-,-)	50 (2) 0 (-,-)	50 (1,2) 0 (-,-)
Clique (Mixed)	0 (-) 50 (22.7,8.9)	0 (-,-) 50 (17.2,11.5)	0 (-) 50 (16.7,9.7)	0 (-,-) 50 (12.3,12.8)	0 (-) 50 (15.5,10.9)	0 (-,-) 50 (12.4,13.0)

Table 1. : Comparing rank aggregation with the algorithms of (Wainwright et al., 2005) and (Kolmogorov, 2004) over synthetic data. In each cell, first row contains number of successes out of 50 (with average number of iterations (I) and K at which best found) and the second row contains number of failures out of 50 (along with the average gap G and average MAP score).

the specific case of TRW-Sequential, this improvement comes at an almost negligible price because the cost of finding the top-K and doing rank-aggregation is miniscule as compared to the much more expensive reparameterization step.



(a) Ten grids (Mixed potentials)

(b) Ten cliques (Mixed potentials)

(c) Grid (One instance)

(d) Clique (One instance)

Figure 1. Comparison of Rank Aggregation with TRW-Sequential on (a) aggregate of 10 grids (b) aggregate of 10 cliques with mixed potentials. The curves plot the average gap vs the number of iterations. Figures (c) and (d) plot the gap for one random instance of each family.

To compare the effectiveness of the best reparameterization algorithm, TRW-Sequential, vis-a-vis Rank Aggregation, we looked at the gap values obtained by the two approaches. In Figure 1, we plot the gap averaged over ten random instances each of a clique and a grid-graph with mixed potentials. We plot the evolution of the gap with the number of iterations. For grid-graphs, TRW-Sequential does not take too long to converge to the same gap values (in these cases, zero) as Rank Aggregation, but for cliques, the difference becomes significant. Here, it suffers from wide oscillations and non-monotonicity, as also depicted in the case of a random clique instance in Figure 1(d). Since the bounds returned by TRW-Sequential are always monotonically decreasing, such anomalous behavior can only be attributed to the MAP assignment selection scheme. Rank Aggregation, on the other hand, smoothes out this behavior by choosing its MAP from a pool of potentially very good solutions.

TRW-Edge		TRW-Sequential	
TreeAgree	RankAgg	TreeAgree	RankAgg
48 (18) 18 (0.5,-0.9) (1.0,-1.3)	61 (13,6) 5 (0.5,-1.1)	59 (13) 7 (2.9,-3.3) (1.6,-1.8)	61 (8,6) 5 (0.5,-1.1)

Table 2. : Rank aggregation over real-life data. The table should be interpreted like Table 1. The third row for top-1 contains failure statistics only for those cases where top-K also fails.

Real-life data: We looked at constrained inferencing in the context of segmenting and labeling bibliographic data. The dataset had 66 records and 24 labels. These labels were obtained via a Begin-Continue-End-Unique encoding of six labels — TITLE, AUTHOR, JOURNAL, YEAR, PAGE_NUMBER and OTHER. The record size varied between 4 and 28 words, with an average size of 11. Our aim was to compute the best segmentation and labeling of each record, while not repeating the TITLE, JOURNAL, YEAR and PAGE_NUMBER labels inside a record. We enforced this by adding edge potentials of $-\infty$ for every label pair begin-label and end-label for each la-

bel with uniqueness constraints. Consequently, our linear chain model became a clique in the presence of constraints and exact inference was thus intractable. Similar to the synthetic datasets, we report statistics for TRW-Edge and TRW-Sequential, averaged over all the records.

Table 2 shows that Rank Aggregation leads to a marked improvement over using just top-1. Even in cases where it fails to find the optimum, Rank Aggregation reduces the gap by upto 70% and reports a much better MAP score than top-1 (ref. third row of Table 2). In successful cases too, Rank Aggregation takes much fewer iterations to report the MAP than top-1.

Tree Selection

As mentioned in (Kolmogorov, 2004) and (Wainwright et al., 2005), the twin issues of tree-selection and tree-parameter selection are important problems in themselves. Here, we illustrate the effect of tree-selection on the performance of TRW-Edge and TRW-Sequential over our synthetic dataset, with and without rank-aggregation. We tried two tree-selection approaches: (a) The default set of trees used in the first experiment and (b) Trees selected by keeping strongly correlated edges intact. For this, we ran ordinary belief propagation on the graph and computed the pseudo max-marginals. Using these, the approximate mutual information (MI) was computed for all pairs of adjacent vertices. MI was used as edge weights and we constructed maximum-weight spanning trees using these edge weights. The edges that were covered in a tree had their weights halved and we kept on choosing trees till all edges were covered by atleast one tree.

Table 3 shows the effect of tree selection. It is clear that no single tree-selection scheme can perform well for all kinds of graphs. For example, in the TRW-Edge scenario, the default tree selection scheme provides better gaps for cliques and better MAP-scores for grids. The MI-based scheme behaves in the complementary manner and also leads to more successes for grids.

Thus it remains an open issue to dynamically choose a set of relevant tree-selection criteria by looking at graph properties like sparsity, associativity, correlation between labels etc.

5. Discussion

The framework of the rank aggregation algorithm does not require individual sets S_i to be trees. We just require that each set of edges S_i return (1) a list R_i of high scoring assignments, and (2) an upper bound on the scores of assignments not in the list. For improving the final bounds our goal is to make each set the largest possible for which it is still possible to efficiently find the above quantities. We discuss one important case where this is possible.

When potentials on all edges are associative and labels are binary, MAP estimation corresponds to the optimization of a submodular function. Such functions can be optimized in polynomial time over arbitrary graphs with cycles, for example using the mincut algorithm (Kolmogorov & Zabih, 2004). In applications like collective inference for information extraction (Bunescu & Mooney, 2004; Finkel et al., 2005), image co-segmentation (Rother et al., 2006) and creation of digital tapestries (Rother et al., 2005) often there is a large collection of such associative edges mixed with other non-associative edges. We create a set S_i of the associative edges and use trees to cover the non-associative edges possibly mixed with associative edges. If the best mincut solution appears in the ranked lists from the trees then we are guaranteed to find an optimal solution based on Corollary 3.1, although not necessarily the one returned by mincut. Since it is not expensive to find top K assignments in trees, we can afford to choose a relatively large value of K.

We list other areas for further discussion and investigation

- Deciding how to distribute the graph potentials over the individual sets: Since we can detect optimality without requiring the trees to agree on an assignment it is unclear if tree-based reparameterization is the best method of distributing potentials of the graph over the trees.
- Choosing an order of exploring sets so as to reduce the number of assignments evaluated: This issue will be particularly important when finding the next highest scoring assignment is not cheap, for example when using mincuts of graphs.
- Extending the rank aggregation framework to constrained inference: In applications like information extraction, the basic MRF is a simple chain, but in the presence of constraints (for example, constraints on the cardinality of labels) the graph becomes complete. In such cases, we are investigating methods of dynamically choosing trees out of constrained edges only when violations are detected.

  Note that constraints can also be handled via the Integer Linear Programming (ILP) framework,

Graph	TreeSelection	TRW-Edge		TRW-Sequential	
		TreeAgree	RankAgg	TreeAgree	RankAgg
Grid (Mixed)	Default	32 (13) 18 (0.7,61.8)	40 (10,6) 10 (0.7,61.4)	39 (8) 11 (0.8,60.8)	43 (6,6) 7 (0.7,61.2)
	MI-based	37 (13) 13 (0.6,61.4)	44 (7,5) 6 (0.7,61.0)	39 (4) 11 (1.0,60.7)	44 (3,4) 6 (0.4,61.2)
Clique (Mixed)	Default	0 (-) 50 (16.7,9.7)	0 (-,-) 50 (12.3,12.8)	0 (-) 50 (15.5,10.9)	0 (-,-) 50 (12.4,13.0)
	MI-based	0 (-) 50 (18.3,10.5)	0 (-,-) 50 (12.2,13.3)	0 (-) 50 (15.2,11.2)	0 (-,-) 50 (10.3,14.1)

Table 3. : Comparing tree selection schemes and their effects on rank-aggregation. The cell entries should be interpreted in the same way as Table 1.

such as illustrated in (Roth & tau Yih, 2005). However, the quality of the ILP solution is heavily dependent on the LP relaxation and rounding mechanism, which in turn are sensitive to the nature of constraints. Also, the Rank-aggregation naturally generalizes to retrieving the top-K solutions rather than the top-1, which is not possible to do within the ILP framework. Lastly, the ILP machinery may be too expensive in the scenario where we are only interested in quickly computing a bound on the optimum and not the MAP itself.

Acknowledgments

We would like to thank the anonymous reviewers whose valuable comments helped improve the paper.

References

Bilenko, M. (2004). Learnable similarity functions and their applications to clustering and record linkage. *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA* (pp. 981–982). AAAI Press / The MIT Press.

Bunescu, R., & Mooney, R. J. (2004). Collective information extraction with relational markov networks. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics* (pp. 439–446).

Chakrabarti, S., Dom, B., & Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. *SIGMOD Rec.*, *27*, 307–318.

Fagin, R., Lotem, A., & Naor, M. (2001). Optimal aggregation algorithms for middleware. *Journal of Computer and System Sciences*, *66*, 614,656.

Finkel, J. R., Grenager, T., & Manning., C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*.

Jensen, D., Neville, J., & Gallagher, B. (2004). Why collective inference improves relational classification. *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Kolmogorov, V. (2004). *Convergent tree-reweighted message passing for energy minimization* (Technical Report MSR-TR-2004-90). Microsoft Research (MSR).

Kolmogorov, V., & Wainwright, M. J. (2005). On the optimality of tree-reweighted max-product message passing. *UAI*.

Kolmogorov, V., & Zabih, R. (2004). What energy functions can be minimized via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*.

Lu, Q., & Getoor, L. (2003). Link-based classification. *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA* (pp. 496–503).

Mansuri, I., & Sarawagi, S. (2006). A system for integrating unstructured data into relational databases. *Proc. of the 22nd IEEE Int'l Conference on Data Engineering (ICDE)*.

McCallum, A., & Wellner, B. (2003). Toward conditional models of identity uncertainty with application to proper noun coreference. *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web* (pp. 79–86). Acapulco, Mexico.

Meltzer, T., Yanover, C., & Weiss, Y. (2005). Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. *International Conference on Computer Vision* (pp. I: 428–435).

Nilsson, D. (1998). An efficient algorithm for finding the M most probable configurations in probabilistic expert systems. *Statistics and computing*, *8*, 159–173.

Parag, & Domingos, P. (2004). Multi-relational record linkage. *Proceedings of 3rd Workshop on Multi-Relational Data Mining at ACM SIGKDD*. Seattle, WA.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann.

Roth, D., & tau Yih, W. (2005). Integer linear programming inference for conditional random fields. *ICML '05: Proceedings of the 22nd international conference on Machine learning* (pp. 736–743). New York, NY, USA: ACM Press.

Rother, C., Kolmogorov, V., Minka, T., & Blake, A. (2006). *Cosegmentation of image pairs by histogram matching — incorporating a global constraint into MRFs* (Technical Report MSR-TR-2006-36). Microsoft Research (MSR).

Rother, C., Kumar, S., Kolmogorov, V., & Blake, A. (2005). Digital tapestry. *IEEE Computer Vision and Pattern Recognition or CVPR* (pp. I: 589–596).

Taskar, B. (2004). *Learning structured prediction models: A large margin approach*. Doctoral dissertation, Stanford University.

Wainwright, Jaakkola, & Willsky (2005). MAP estimation via agreement on trees: Message-passing and linear programming. *IEEETIT: IEEE Transactions on Information Theory*, *51*.

Wainwright, M., Jaakkola, T., & Willsky, A. (2004). Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and computing*, *14*, 143–166.

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

Wellner, B., McCallum, A., Peng, F., & Hay, M. (2004). An integrated, conditional model of information extraction and coreference with application to citation matching. *Conference on Uncertainty in Artificial Intelligence (UAI)*.

Yanover, C., & Weiss, Y. (2003). Finding the M most probable configurations in arbitrary graphical models. *NIPS*. MIT Press.