Efficient Belief Propagation for Utility Maximization and Repeated Inference

Aniruddh Nath and **Pedro Domingos**
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
{*nath, pedrod*}@*cs.washington.edu*

Abstract

Many problems require repeated inference on probabilistic graphical models, with different values for evidence variables or other changes. Examples of such problems include utility maximization, MAP inference, online and interactive inference, parameter and structure learning, and dynamic inference. Since small changes to the evidence typically only affect a small region of the network, repeatedly performing inference from scratch can be massively redundant. In this paper, we propose *expanding frontier belief propagation (EFBP)*, an efficient approximate algorithm for probabilistic inference with incremental changes to the evidence (or model). EFBP is an extension of loopy belief propagation (BP) where each run of inference reuses results from the previous ones, instead of starting from scratch with the new evidence; messages are only propagated in regions of the network affected by the changes. We provide theoretical guarantees bounding the difference in beliefs generated by EFBP and standard BP, and apply EFBP to the problem of expected utility maximization in influence diagrams. Experiments on viral marketing and combinatorial auction problems show that EFBP can converge much faster than BP without significantly affecting the quality of the solutions.

Introduction

Most work on approximate probabilistic inference in graphical models focuses on computing the marginal probabilities of a set of variables, or determining their most probable state, given some fixed evidence and a fixed model. However, many interesting problems require repeated inference, with changing evidence or a changing model:

- **Utility maximization** (Howard and Matheson 2005) in decision networks involves a search over possible choices of actions. We can treat each choice of actions as an assignment of truth values to evidence variables, and compute the expected utility of that choice by inferring the expected values of the utility variables.
- **Maximum a posteriori (MAP) inference** (Park 2002), i.e., computing the most likely state of a set of query variables $\mathbf{Q}$ given partial evidence $\mathbf{e}$ of the variables in the complement of $\mathbf{Q}$. We can treat the variables in $\mathbf{Q}$ as changing evidence, and compute the likelihood for each assignment of values to $\mathbf{Q}$.
- **Online inference**, where the values of evidence variables are repeatedly updated (possibly in real time).
- **Interactive inference**, where the choice of evidence variables, their values, and the choice of query can change arbitrarily as dictated by the user.
- **Parameter and structure learning** (Heckerman, Geiger, and Chickering 1995; Della Pietra, Della Pietra, and Lafferty 1997) involve repeatedly modifying the model and recomputing the likelihood.
- **Dynamic inference** (Murphy 2002) involves sequential problems, where the query at one time step depends on evidence and hidden variables at that step and previous ones.

The obvious way to solve these problems is simply to repeatedly perform inference from scratch each time the evidence (or model) changes. However, if the problem remains largely unchanged between two successive search iterations, it may be the case that most of the beliefs are not significantly altered by the changes. In these situations, it would be beneficial to reuse as much of the computation as possible between successive runs of inference.

Delcher et al. (1996) presented an exact online inference algorithm for tree-structured Bayesian networks. Acar et al. (2008) refer to the problem of repeated inference on variations of a model as *adaptive inference*. They described an exact algorithm that updates marginals as the dependencies in the model are updated. However, we are not aware of any general-purpose *approximate* inference algorithms that avoid redundant computation as the evidence changes. In this paper, we propose *expanding frontier belief propagation* (EFBP), an approximate inference algorithm that only updates the beliefs of variables significantly affected by the changed evidence. EFBP can be straightforwardly extended to handle changes to the model. We provide guarantees on the performance of EFBP relative to BP. These are based on the fact that, if the potentials in a model are bounded, the difference in marginal probabilities calculated by EFBP and traditional loopy BP can be bounded as well.

We apply EFBP to the problem of expected utility maximization in influence diagrams. Utility maximization can be cast as repeated calculation of the marginals in a graphical model with changing evidence. Under certain conditions, it can be shown that the same actions are chosen whether BP or EFBP is used to calculate the marginals. Experiments in viral marketing and combinatorial auction domains show that EFBP can be orders of magnitude faster than BP, without significantly affecting the quality of the solutions.

Graphical Models and Decision Theory

Graphical models compactly represent the joint distribution of a set of variables $\mathbf{X} = (X_1, X_2, \ldots, X_n) \in \mathcal{X}$ as a product of factors (Pearl 1988): $P(\mathbf{X}=\mathbf{x}) = \frac{1}{Z}\prod_k \phi_k(\mathbf{x}_k)$, where each factor ϕ_k is a non-negative function of a subset of the variables $\mathbf{x}_k$, and Z is a normalization constant. Under appropriate restrictions, the model is a *Bayesian network* and $Z = 1$. A *Markov network* or *Markov random field* can have arbitrary factors. Graphical models can also be represented in *log-linear form*: $P(\mathbf{X}=\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_i w_i g_i(\mathbf{x})\right)$, where the *features* $g_i(\mathbf{x})$ are arbitrary functions of the state.

A key inference task in graphical models is computing the marginal probabilities of some variables (the query) given the values of some others (the evidence). This problem is #P-complete, but can be solved approximately using *loopy belief propagation* (BP) (Yedidia, Freeman, and Weiss 2003). In the simplest form, the Markov network is first converted to an equivalent pairwise network. Belief propagation then works by repeatedly passing messages between nodes in this network. The message from node t to node s is:

$$m_{ts}(x_s) = \sum_{x_t} \phi_{ts}(x_t, x_s)\phi_t(x_t) \prod_{u \in nb(t)\setminus\{s\}} m_{ut}(x_t)$$

where $nb(t)$ is the set of neighbors of t. The (unnormalized) belief at node t is given by: $M_t(x_t) = \phi_t(x_t)\prod_{u \in nb(t)} m_{ut}(x_t)$.

Many message-passing schedules are possible; the most widely used one is *flooding*, where all nodes send messages at each step. In general, belief propagation is not guaranteed to converge, and it may converge to an incorrect result, but in practice it often approximates the true probabilities well.

An *influence diagram* or *decision network* is a graphical representation of a decision problem (Howard and Matheson 2005). It consists of a Bayesian network augmented with two types of nodes: *decision* or *action* nodes and *utility* nodes. The action nodes represent the agent's choices; factors involving these nodes and *state* nodes in the Bayesian network represent the (probabilistic) effect of the actions on the world. *Utility* nodes represent the agent's utility function, and are connected to the state nodes that directly influence utility.

The fundamental inference problem in decision networks is finding the assignment of values to the action nodes that maximizes the agent's expected utility, possibly conditioned on some evidence. If $\mathbf{a}$ is a choice of actions, $\mathbf{e}$ is the evidence, $\mathbf{x}$ is a state, and $U(\mathbf{x}|\mathbf{a},\mathbf{e})$ is the utility of $\mathbf{x}$ given $\mathbf{a}$ and $\mathbf{e}$, then the *MEU problem* is to compute $\text{argmax}_{\mathbf{a}} E[U(\mathbf{x}|\mathbf{a},\mathbf{e})] = \text{argmax}_{\mathbf{a}} \sum_{\mathbf{x}} P(\mathbf{x}|\mathbf{a},\mathbf{e})U(\mathbf{x}|\mathbf{a},\mathbf{e})$.

Expanding Frontier Belief Propagation

The *repeated inference* problem deals with inference with changing evidence, changing model parameters or changing graph structure. For simplicity, we will focus on the case where the evidence changes and the choice of evidence variables is fixed, as are the parameters and structure; our formulation can be straightforwardly extended to incorporate these other kinds of variation.

Let $\mathbf{G}$ be a graphical model on the variables in $\mathbf{X}$. $\mathbf{E} \subseteq \mathbf{X}$ is the *evidence set*. We are given a sequence $\vec{\mathbf{e}} = (\mathbf{e}_1, \ldots, \mathbf{e}_{|\vec{\mathbf{e}}|})$, each element of which is an assignment of values to the variables in $\mathbf{E}$. For each element $\mathbf{e}_k$ of $\vec{\mathbf{e}}$, we wish to infer the marginal probabilities of the variables in $\mathbf{X}\backslash\mathbf{E}$, given that $\mathbf{E} = \mathbf{e}_k$.

Changing the values of a small number of evidence nodes is unlikely to significantly change the probabilities of most state nodes in a large network. EFBP takes advantage of this by only updating regions of the network affected by the new evidence. The algorithm starts by computing the marginal probabilities of non-evidence nodes given the initial evidence values using standard BP (e.g., by flooding). Then, each time the evidence is changed, it maintains a set Δ of nodes affected by the changed evidence variables, initialized to contain only those changed nodes. In each iteration of BP, only the nodes in Δ send messages. Neighbors of nodes in Δ are added to Δ if the messages they receive differ by more than a threshold γ from the final messages they received when they last participated in BP. (The neighbors of a node are the nodes that appear in some factor with it, i.e., its Markov blanket.) In the worst case, the whole network may be added to Δ, and we revert to BP. In most domains, however, the effect of a change usually dies down quickly, and only influences a small region of the network. In such situations, EFBP can converge much faster than BP.

Pseudocode for EFBP is shown in Algorithm 1. $\hat{m}_{ts}^{k,i}$ is the message from node t to node s in iteration i of inference run k. $\hat{m}_{ts}^{k,c_k}$ is the final message sent from t to s in inference run k. (If s was not in Δ in inference run k, then $\hat{m}_{ts}^{k,c_k} = \hat{m}_{ts}^{k-1,c_{k-1}}$.) To handle changes to the model from one iteration to the next, we simply add to Δ variables directly affected by that change (e.g., the child variable of a new edge when learning a Bayesian network).

EFBP is based on a similar principle to residual belief propagation (RBP) (Elidan, McGraw, and Koller 2006): focus effort on the regions of the graph that are furthest from convergence. However, while RBP is used to schedule message updates in a single run of belief propagation, the purpose of EFBP is to minimize redundant computation when repeatedly running BP on the same network, with varying settings of some of the variables. One could run EFBP using RBP to schedule messages, to speed up convergence.

If the potentials are bounded, EFBP's marginal probability estimates provide bounds on BP's. We can show this by viewing the difference between the beliefs generated by EFBP and those generated by BP as multiplicative error in the messages passed by EFBP:

$$\hat{m}_{ts}^{k,i}(x_s) = \tilde{m}_{ts}^{k,i}(x_s)\tilde{e}_{ts}^{i}(x_s)$$

Algorithm 1 EFBP(variables $\mathbf{X}$, graphical model $\mathbf{G}$, evidence $\vec{\mathbf{e}}$, threshold γ)

```
for all $t, s$: $\hat{m}_{ts}^{1,1} \leftarrow 1$
Perform standard BP on $\mathbf{X}$ and $\mathbf{G}$ with evidence $\mathbf{e}_1$.
for $k \leftarrow 2$ to $|\vec{\mathbf{e}}|$ do
  for all $t, s$: $\hat{m}_{ts}^{k,1} \leftarrow \hat{m}_{ts}^{k-1,c_{k-1}}$
  $\Delta \leftarrow$ set of evidence nodes that change
      between $\mathbf{e}_k$ and $\mathbf{e}_{k-1}$
  $converged \leftarrow$ False
  $i \leftarrow 1$
  while $converged =$ False do
    Send messages from all nodes in $\Delta$,
        given evidence $\mathbf{e}_k$
    $converged \leftarrow$ True
    for all nodes $s$ that receive messages do
      if $|\hat{m}_{ts}^{k,i} - \hat{m}_{ts}^{k-1,c_{k-1}}| > \gamma$ for any $t \in nb(s)$ then
        Insert $s$ into $\Delta$
      end if
      if $s \in \Delta$ and $|\hat{m}_{ts}^{k,i} - \hat{m}_{ts}^{k,i-1}| > \gamma$
      for any $t \in nb(s)$ then
        $converged \leftarrow$ False
      end if
    end for
    $i \leftarrow i + 1$
  end while
end for
```

where $\hat{m}_{ts}^{k,i}(x_s)$ is the message sent by EFBP in iteration i of inference run k (i.e., nodes not in Δ effectively resend the same messages as in $i-1$); $\tilde{m}_{ts}^{k,i}(x_s)$ is the message sent if all nodes recalculate their messages in iteration i, as in BP; and $\tilde{e}_{ts}^{i}(x_s)$ is the multiplicative error introduced in iteration i of EFBP.

This allows us to bound the difference in marginal probabilities calculated by BP and EFBP. For simplicitly, we prove this for the special case of binary nodes. The proof uses a measure called the *dynamic range* of a function, defined as follows (Ihler, Fisher, and Willsky 2005):

$$d(f) = \sup_{x,y} \sqrt{f(x)/f(y)}$$

Theorem 1. *For binary node x_t, the probability estimated by BP at convergence (p_t) can be bounded as follows in terms of the probability estimated by EFBP ($\hat{p}_t$) after n iterations:*

$$\begin{aligned} p_t &\geq \frac{1}{(\zeta_t^n)^2[(1/\hat{p}_t)-1]+1} = lb(p_t) \\ p_t &\leq \frac{1}{(1/\zeta_t^n)^2[(1/\hat{p}_t)-1]+1} = ub(p_t) \end{aligned}$$

where $\log \zeta_t^n = \sum_{u \in nb(t)} \log \nu_{ut}^n$, $\nu_{ts}^1 = \delta(\tilde{e}_{ts}^1) d(\phi_{ts})^2$, *and* ν_{ut}^i *is given by:*

$$\begin{aligned} \log \nu_{ts}^{i+1} &= \log \frac{d(\phi_{ts})^2 \varepsilon_{ts}^i + 1}{d(\phi_{ts})^2 + \varepsilon_{ts}^i} + \log \delta(\tilde{e}_{ts}^i) \\ \log \varepsilon_{ts}^i &= \sum_{u \in nb(t) \setminus s} \log \nu_{ut}^i \end{aligned}$$

$$\delta(\tilde{e}_{ts}^i) = \sup_{x_s, y_s} \sqrt{\frac{1+\gamma/\left(\tilde{m}_{ts}^{k,i}(x_s)\right)}{1-\gamma/\left(\tilde{m}_{ts}^{k,i}(y_s)\right)}}$$

Proof.
Since EFBP recalculates messages when $|\hat{m}_{ts}^{k,i} - \tilde{m}_{ts}^{k,i}| > \gamma$,

$$\begin{aligned} \tilde{m}_{ts}^{k,i}(x_s) - \gamma \;\leq\; & \tilde{m}_{ts}^{k,i}(x_s)\tilde{e}_{ts}^i(x_s) \;\leq \tilde{m}_{ts}^{k,i}(x_s) + \gamma \\ 1 - \gamma/\tilde{m}_{ts}^{k,i}(x_s) \;\leq\; & \tilde{e}_{ts}^i(x_s) \;\leq 1 + \gamma/\tilde{m}_{ts}^{k,i}(x_s) \end{aligned}$$

This allows us to bound the dynamic range of $\tilde{e}_{ts}^i$:

$$d(\tilde{e}_{ts}^i) \leq \sup_{x_s, y_s} \sqrt{\frac{1+\gamma/\left(\tilde{m}_{ts}^{k,i}(x_s)\right)}{1-\gamma/\left(\tilde{m}_{ts}^{k,i}(y_s)\right)}} = \delta(\tilde{e}_{ts}^i)$$

Theorem 15 of Ihler et al. (2005) implies that, for any fixed point beliefs $\{M_t\}$ found by belief propagation, after $n \geq 1$ iterations of EFBP resulting in beliefs $\{\hat{M}_t^n\}$ we have

$$\log d(M_t/\hat{M}_t^n) \leq \sum_{u \in nb(t)} \log \nu_{ut}^n = \log \zeta_t^n$$

It follows that $d(M_t/\hat{M}_t^n) \leq \zeta_t^n$, and therefore $\frac{M_t(1)/\hat{M}_t^n(1)}{M_t(0)/\hat{M}_t^n(0)} \leq (\zeta_t^n)^2$ and $(1-\hat{p}_t)/\hat{p}_t \leq (\zeta_t^n)^2(1-p_t)/p_t$, where p_t and $\hat{p}_t$ are obtained by normalizing M_t and $\hat{M}_t$. The upper bound follows, and the lower bound can be obtained similarly. □

Intuitively, ν_{ut}^i can be thought of as a measure of the accumulated error in the incoming message from u to t in iteration i. It can be computed iteratively using a message-passing algorithm similar to BP.

Maximizing Expected Utility

The MEU problem in influence diagrams can be cast as a series of marginal probability computations with changing evidence. If $u_i(\mathbf{x}_i)$ and $p_i(\mathbf{x}_i|\mathbf{a},\mathbf{e})$ are respectively the utility and marginal probability of utility node i's parents $\mathbf{X}_i$ taking values $\mathbf{x}_i$, then the expected utility of an action choice is:

$$\begin{aligned} E[U(\mathbf{a}|\mathbf{e})] &= \sum_{\mathbf{x}} P(\mathbf{x}|\mathbf{a},\mathbf{e}) U(\mathbf{x}|\mathbf{a},\mathbf{e}) \\ &= \sum_{\mathbf{x}} P(\mathbf{x}|\mathbf{a},\mathbf{e}) \sum_i \sum_{\mathbf{x}_i} \mathbf{1}\{\mathbf{X}_i = \mathbf{x}_i\} u_i(\mathbf{x}_i) \\ &= \sum_i \sum_{\mathbf{x}_i} u_i(\mathbf{x}_i) \sum_{\mathbf{x}} \mathbf{1}\{\mathbf{X}_i = \mathbf{x}_i\} P(\mathbf{x}|\mathbf{a},\mathbf{e}) \\ &= \sum_i \sum_{\mathbf{x}_i} u_i(\mathbf{x}_i) p_i(\mathbf{x}_i|\mathbf{a},\mathbf{e}) \end{aligned}$$

Given a choice of actions, we can treat action nodes as evidence. To calculate the expected utility, we simply need to calculate the marginal probabilities of all values of all utility nodes. Different action choices can be thought of as different evidence.

In principle, the MEU action choice can be found by searching exhaustively over all action choices, computing the expected utility of each using any inference method. However, this will be infeasible in all but the smallest domains. An obvious alternative, particularly in very large domains, is greedy search: starting from a random action choice, consider flipping each action in turn, do so if it increases expected utility, and stop when a complete cycle produces no improvements. This is guaranteed to converge to a local optimum of the expected utility, and is the method we use in our experiments.

Lemma 2. *The expected utility $E_{bp}[U]$ of an action choice estimated by BP can be bounded as follows:*

$$E_{bp}[U] \geq \sum_i \sum_{\mathbf{x}_i} u_i(\mathbf{x}_i) \Big[\mathbf{1}\{u_i(\mathbf{x}_i) > 0\} lb(p_i(\mathbf{x}_i|\mathbf{a}, \mathbf{e})) + \mathbf{1}\{u_i(\mathbf{x}_i) < 0\} ub(p_i(\mathbf{x}_i|\mathbf{a}, \mathbf{e})) \Big]$$

$$E_{bp}[U] \leq \sum_i \sum_{\mathbf{x}_i} u_i(\mathbf{x}_i) \Big[\mathbf{1}\{u_i(\mathbf{x}_i) > 0\} ub(p_i(\mathbf{x}_i|\mathbf{a}, \mathbf{e})) + \mathbf{1}\{u_i(\mathbf{x}_i) < 0\} lb(p_i(\mathbf{x}_i|\mathbf{a}, \mathbf{e})) \Big]$$

Proof. The lemma follows immediately from Theorem 1 and the definition of expected utility. □

Based on this lemma, we can design a variant of EFBP that is guaranteed to produce the same action choice as BP when used with greedy search. Let $\mathbf{A}_i$ denote the set of action choices considered in the ith step of greedy search. If EFBP picks action choice $\mathbf{a} \in \mathbf{A}_i$, BP is guaranteed to make the same choice if the following condition holds:

$$lb(E[U(\mathbf{a})]) > \max_{\mathbf{a}' \in \mathbf{A}_i} ub(E[U(\mathbf{a}')]) \qquad (1)$$

When the condition does not hold, we revert the messages of all nodes to their values after the initial BP run, insert all changed actions into Δ, and rerun EFBP. If condition 1 still does not hold, we halve γ and repeat. (If γ becomes smaller than the BP convergence threshold, we run standard BP.) We call this algorithm EFBP$*$.

Note that using the above bound for search requires a slight modification to the calculation of ν^1_{ts}, since EFBP and BP result in different message initializations for the first BP iteration at every search step. If BP initializes all messages to 1 before every search step, the error function $\tilde{e}^1_{ts}(x_s) = \hat{m}^{k,1}_{ts}(x_s)$. If BP initializes its messages with their values from end of the previous search iteration, then ν^1_{ts} can also be similarly initialized. The calculations of ν^i_{ts} for $i \neq 1$ are not altered.

Let Greedy(I) denote greedy search using inference algorithm I to compute expected utilities.

Theorem 3. *Greedy(EFBP$*$) outputs the same action choice as Greedy(BP).*

Proof. Condition 1 guarantees that EFBP makes the same action choice as BP. Since EFBP$*$ guarantees that the condition holds in every iteration of the final run of EFBP, it guarantees that the action choice produced is the same. □

In practice, EFBP$*$ is unlikely to provide large speedups over BP, since the bound in Theorem 1 must be repeatedly recalculated even for nodes outside Δ. Empirically, running EFBP with a fixed threshold not much higher than BP's convergence threshold seems to yield an action choice of very similar utility to BP's in a fraction of the time.

Experiments

We evaluated the performance of EFBP on two domains: viral marketing and combinatorial auctions. We implemented EFBP as an extension of the Alchemy system (Kok et al. 2008). The experiments were run on 2.33 GHz processors with 2 GB of RAM. We used a convergence threshold of 10^{-4} for flooding, including the initial BP run for EFBP, and a threshold of $\gamma = 10^{-3}$ for the remaining iterations of EFBP. We evaluated the utility of the actions chosen by EFBP using a second run of full BP, with a threshold of 10^{-4}. As is usually done, both algorithms were run for a few (10) iterations after the convergence criterion was met.

Viral Marketing

Viral marketing is based on the premise that members of a social network influence each other's purchasing decisions. The goal is then to select the best set of people to market to, such that the overall profit is maximized by propagation of influence through the network. Originally formalized by Domingos and Richardson (2001), this problem has since received much attention, including both empirical and theoretical results.

A standard dataset in this area is the Epinions web of trust (Richardson and Domingos 2002). Epinions.com is a knowledge-sharing Web site that allows users to post and read reviews of products. The "web of trust" is formed by allowing users to maintain a list of peers whose opinions they trust. We used this network, containing 75,888 users and over 500,000 directed edges, in our experiments. With over 75,000 action nodes, this is a very large decision problem, and no general-purpose MEU algorithms have previously been applied to it (only domain-specific implementations).

We used a model very similar to that of Domingos and Richardson (2001). We represent this problem as a log-linear model with one binary state variable b_i representing the purchasing decision of each user i (i.e., whether or not they buy the product). The model also has one binary action variable m_i for each user, representing the choice of whether or not to market to that person. The prior belief about each user's purchasing decision is represented by introducing a singleton feature for that user. The value is 1 if the user buys the product, and 0 otherwise. In our experiments, this feature had a weight of -2. The effect of the marketing actions on the users' purchasing decisions is captured by a second set of features. For each user, we introduce a feature whose value is 1 when the formula $m_i \Rightarrow b_i$ is true, and 0 otherwise (i.e., when the user is marketed to but does not buy the product). All these features are given the same weight, w_1.

The web of trust is encoded as a set of pairwise features over the state variables. For each pair of users (i, j) such that

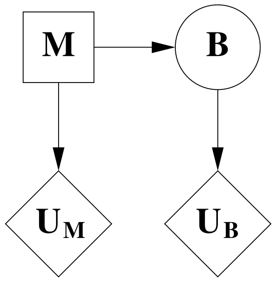


Figure 1: Influence diagram for viral marketing.

i trusts j, the value of the feature is 1 when the formula $b_j \Rightarrow b_i$ is true, and 0 otherwise (i.e., j buys the product but i does not.) These features represent the fact that if j purchases the product, i is more likely to do so as well. These features all have weight w_2.

For each user, we also introduce two utility nodes. The first represents the cost of marketing to the user, and the second represents the profit from each purchase. The first utility node is connected to the action node corresponding to the user, and the second is connected to the state node representing the user's purchasing decision. We set the cost of marketing to each user to -1, and the profit from each purchase to 20.

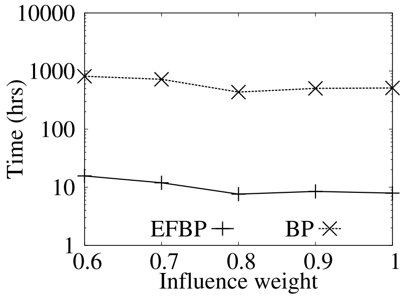
Note that since this model is not locally normalized, it is not a traditional influence diagram. However, it is equivalent to an influence diagram with four nodes (see Figure 1):

- An action node $\mathbf{M}$ whose state space is the set of possible marketing choices over all users.
- A corresponding utility node $\mathbf{U_M}$, representing the cost of each marketing choice.
- A state node $\mathbf{B}$, representing the set of possible purchasing decisions by the users.
- A corresponding utility node $\mathbf{U_B}$, representing the profit from each possible purchasing decision.

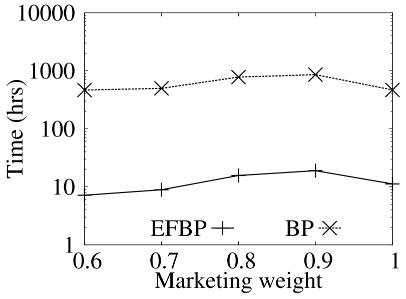
The log-linear model described above can be thought of as a compact representation of the conditional probability distributions and utility functions for this four-node influence diagram.

We ran greedy search with BP and EFBP directly on this model. The initial action choice was to market to no one. All results are averages of five runs. Fixing w_1 at 0.8 for all users, inference times varied as shown in Figure 2(a) as we changed w_2. Running utility maximization with BP until convergence was not feasible; instead, we extrapolated from the number of actions considered during the first 24 hours, assuming that search with BP would consider the same number of actions as search with EFBP. Figure 2(b) plots the result of a similar experiment, with w_2 fixed at 0.6 and varying w_1. In both cases, EFBP was consistently orders of magnitude faster than BP.

We can also compare the utility obtained by BP and EFBP given a fixed running time of 24 hours. EFBP consistently achieves about 40% higher utility than BP when varying the influence weight, with higher advantage for lower weights.



(a)



(b)

Figure 2: Convergence times for viral marketing.

For a marketing weight of 1.0, EFBP achieves about 80% higher utility than BP; this advantage decreases gradually to zero as the weight is reduced (reflecting the fact that fewer and fewer customers buy).

Richardson and Domingos (2002) tested various specially designed algorithms on the same network. They estimated that inference using the model and algorithm of Domingos and Richardson (2001), which are the most directly comparable to ours, would have taken hundreds of hours to converge (100 per pass). (This was reduced to 10-15 minutes using additional problem-specific approximations, and a much simpler linear model that did not require search was even faster.) Although these convergence times cannot be directly compared with our own (due to the different hardware, parameters, etc., used), it is noteworthy that EFBP converges much faster than the most general of their methods.

Probabilistic Combinatorial Auctions

Combinatorial auctions can be used to solve a wide variety of resource allocation problems (Cramton, Shoham, and Steinberg 2006). An auction consists of a set of bids, each of which is a set of requested products and an offered reward. The seller determines which bids to assign each product to. When all requested products are assigned to a bid, the bid is said to be satisfied, and the seller collects the reward. The seller's goal is to choose an assignment of products to bids that maximizes the sum of the rewards of satisfied bids; this is an NP-hard problem. While there has been much research on combinatorial auctions, it generally assumes that the world is deterministic (with a few exceptions, e.g., Golovin (2007)). In practice, however, both the supply and demand for products are subject to many sources

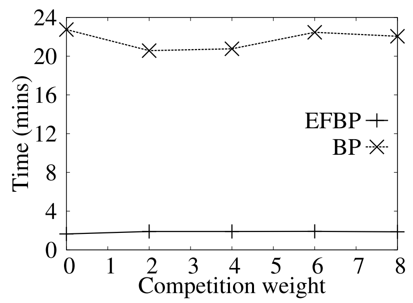


Figure 3: Convergence times for combinatorial auctions.

of uncertainty. (For example, supply of one product may make supply of another less likely because they compete for resources.)

We model uncertainty in combinatorial auctions by allowing product assignments to fail, resulting in the loss of reward from the corresponding bids. The model contains one multi-valued action for each product, with one possible value for each bid. Each product also has a binary state variable, to model the success or failure of the action corresponding to that product.

To model competition between pairs of products, we introduce a set of pairwise features. The features have value 1 when one or both of the product assignments fail. In other words, the success of one increases the failure probability of the other. These features create a network of dependencies between product failures.

The model also contains one binary utility node for each bid. Note that, as in the viral marketing experiment, this model does not yield a traditional influence diagram. However, it can similarly be converted to an equivalent influence diagram with a small number of high-dimensional nodes.

We generated bids according to the decay distribution described by Sandholm (2002), with $\alpha = 0.75$, and randomly generated 1000 competition features from a uniform distribution over product pairs. Figure 3 plots the inference time for a 1000-product, 1000-bid auction, varying the weight of the competition features. The results are averages of ten runs. As in the viral marketing experiment, EFBP converges much faster than BP. Since in this case both algorithms can be run until convergence, the expected utilities of the chosen product assignments are extremely similar.

Conclusion

In this paper, we presented *expanding frontier belief propagation*, an efficient approximate algorithm for probabilistic inference with incremental changes to the evidence or model. EFBP only updates portions of the graph significantly affected by the changes, thus avoiding massive computational redundancy. We applied this algorithm to utility maximization problems, and provided theoretical bounds on the quality of the solutions.

Directions for future work include extending the "expanding frontier" idea to other kinds of inference algorithms (e.g., MCMC), applying it to other problems besides utility maximization (e.g., online inference, learning, etc.), and applying it to other domains.

Acknowledgements

This research was partly funded by ARO grant W911NF-08-1-0242, AFRL contract FA8750-09-C-0181, DARPA contracts FA8750-05-2-0283, FA8750-07-D-0185, HR0011-06-C-0025, HR0011-07-C-0060 and NBCH-D030010, NSF grants IIS-0534881 and IIS-0803481, and ONR grant N00014-08-1-0670. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, DARPA, NSF, ONR, or the United States Government.

References

Acar, U. A.; Ihler, A. T.; Mettu, R. R.; and Šumer, O. 2008. Adaptive inference on general graphical models. In *Proc. of UAI-08*.

Cramton, P.; Shoham, Y.; and Steinberg, R. 2006. *Combinatorial Auctions*. MIT Press.

Delcher, A. L.; Grove, A. J.; Kasif, S.; and Pearl, J. 1996. Logarithmic-time updates and queries in probabilistic networks. *Journal of Artificial Intelligence Research* 4:37–59.

Della Pietra, S.; Della Pietra, V.; and Lafferty, J. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19:380–392.

Domingos, P., and Richardson, M. 2001. Mining the network value of customers. In *Proc. of KDD-01*.

Elidan, G.; McGraw, I.; and Koller, D. 2006. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proc. of UAI-06*.

Golovin, D. 2007. Stochastic packing-market planning. In *Proc. of EC-07*.

Heckerman, D.; Geiger, D.; and Chickering, D. M. 1995. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning* 20:197–243.

Howard, R. A., and Matheson, J. E. 2005. Influence diagrams. *Decision Analysis* 2(3):127–143.

Ihler, A. T.; Fisher, J. W.; and Willsky, A. S. 2005. Loopy belief propagation: Convergence and effects of message errors. *Journal of Machine Learning Research* 6:905–936.

Kok, S.; Sumner, M.; Richardson, M.; Singla, P.; Poon, H.; Lowd, D.; Wang, J.; and Domingos, P. 2008. The Alchemy system for statistical relational AI. Technical report, University of Washington. http://alchemy.cs.washington.edu.

Murphy, K. P. 2002. *Dynamic Bayesian Networks: Representation, Inference and Learning*. Ph.D. Dissertation, University of California, Berkeley.

Park, J. 2002. MAP Complexity Results and Approximation Methods. In *Proc. of UAI-02*.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Richardson, M., and Domingos, P. 2002. Mining knowledge-sharing sites for viral marketing. In *Proc. of KDD-02*.

Sandholm, T. 2002. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence* 135:1–54.

Yedidia, J. S.; Freeman, W. T.; and Weiss, Y. 2003. Understanding belief propagation and its generalizations. In *Exploring Artificial Intelligence in the New Millenium*. Science and Technology Books.