Focused Inference

Rómer E. Rosales†$^+$ and Tommi S. Jaakkola†
†Computer Science and Artificial Intelligence Lab., MIT. Cambridge, MA 02139
$^+$Computer-Aided Diagnosis and Therapy, Siemens Medical Solutions. Malvern, PA 19355 [*]

Abstract

We develop a method similar to variable elimination for computing approximate marginals in graphical models. An underlying notion in this method is that it is not always necessary to compute marginals over all the variables in the graph, but focus on a few variables of interest. The Focused Inference (FI) algorithm introduced reduces the original distribution to a simpler one over the variables of interest. This is done in an iterative manner where in each step the operations are guided by (local) optimality properties. We exemplify various properties of the focused inference algorithm and compare it with other methods. Numerical simulation indicates that FI outperform competing methods.

1 INTRODUCTION AND RELATED WORK

Probabilistic models are useful in a wide variety of fields. An effective way to represent the structure of a probability distribution is by means of a graph; *e.g.,* a graphical model, where variables and their dependencies are associated to nodes and edges in the graph. A crucial task in using such models is to compute marginal distributions over single or groups of random variables. This is referred to here as probabilistic inference. However, the complexity of exact inference scales exponentially with the tree-width of the associated graphical model, and even finding ϵ-approximations (*i.e.,* within given error bounds) is NP-hard [2]. Approximate methods can nevertheless be indispensable in practice.

Approximate inference methods have relied on several key ideas. For example, we can try to simplify the original model to the extent that it becomes tractable. In some cases it is feasible to identify groups of nodes that are nearly conditionally independent or configurations that are highly improbable, and then modify the original graph appropriately to represent this finding before running an exact algorithm (*e.g.,* [9]). Variational methods, on the other hand, typically look for the best approximation within a restricted class of distributions, for example, by minimizing the KL-divergence $D(q||p)$ between the approximation q and the original distribution p[7]. The quality of this approximation is tied to how expressive the restricted class is. Other methods, such as Assumed Density Filtering (ADF)[13](see also [14]), Expectation Propagation (EP)[14] and sequential fitting[5] define the quality of approximation in terms of $D(p||q)$, preserving select statistics in the course of incorporating evidence. Similarly, belief Propagation (BP)[16, 12] and its generalizations[20, 19] seek locally (not globally) consistent beliefs about the values of variables and have been useful in various contexts.

Variational methods can generally provide a bound on the likelihood but are typically symmetry breaking in the sense that the optimized approximate marginals are asymmetric in the absence of any evidence to this effect (cf. mode selection). Propagation algorithms such as BP or EP avoid symmetry breaking due to the different optimization criterion. They are exact for trees, or hyper-trees in the case of generalized BP, but, with the singular exception of [18], do not provide bounds, nor are necessarily guaranteed to converge without additional assumptions.

The structure of the approximating distribution (*e.g.,* [6, 14, 15]), the message propagation scheme (*e.g.,* [19]), or the clusters in generalized BP can lead to important variability in accuracy; finding a *good* structure or clusters is an essential and still unresolved problem.

In this paper we pay closer attention to the essential operation for computing a subset of desired marginals,

[*]Current address. This work was done while the first author was with MIT CSAIL.

i.e., marginalizing out each of the remaining hidden variables. The plain focused inference (FI) algorithm is a simple iterative process that eliminates variables step by step (or in parallel whenever possible) and obtains an approximation of the select marginal distribution. We extend the the basic FI idea to a distributed algorithm operating in a tree-like structure. Our method provides a formalism for performing the necessary graph/distribution transformation operations to be exact on restricted graphs, and approximate for others. While FI can be seen to generate approximating distributions at each step, these distributions do not have to be tractable.

2 DEFINITIONS - BACKGROUND

Let $X = (X_1, ..., X_N)$ be a random vector with X_i taking values denoted $\mathbf{x}_i$, where $\mathbf{x}_i \in \mathcal{X}$. We let $\mathcal{X}$ be the discrete space $\mathcal{X} = \{0, 1, ..., M-1\}$; thus, X takes values in the Cartesian product space $\mathcal{X}^N$. In this paper we consider probability distributions $p(\mathbf{x})$ whose structure is represented by the undirected bipartite graph $\mathcal{G} = (\{V, F\}; E)$, with variable nodes V such that $X = \{X_i | i \in V\}$, factor nodes F, and edges $E \subseteq \{(i, \alpha) | i \in V, \alpha \in F\}$. The factor graph [10] $\mathcal{G}$ corresponds to the following family of distributions:

$$p(X = \mathbf{x}) = \frac{1}{Z_p} \prod_{\alpha \in F} \psi_\alpha(\mathbf{x}_\alpha) \prod_{i \in V} \phi_i(\mathbf{x}_i), \qquad (1)$$

where ψ and ϕ are positive functions (potentials or factors), and X_α are all the random variables connected to the factor node α; *i.e.,* $X_\alpha = \{X_i | (i, \alpha) \in E\}$. For later convenience, we denote single node factors by ϕ. This graph representation is more explicit than the standard undirected graphical model representation regarding the factorization of the probability distribution. In this paper we concentrate primarily on the cases when p can be defined by factor nodes with degree at most two[1]. The neighborhood set of the variable node i is defined $\nu(i) = \{j | (i, \alpha) \in E, (j, \alpha) \in E\}$ (this includes the node i itself), while the neighbors of the variable node i is the set $\nu(i)^- = \nu(i) - \{i\}$. The factors associated to a variable node i are denoted $F(i)$, with $F(i) = \{\alpha \in F | (i, \alpha) \in E\}$. Throughout this paper, the short-hand $p(\mathbf{x})$ will denote $p(X = \mathbf{x})$.

3 FOCUSED INFERENCE APPROXIMATION

Consider the basic marginalization operation. When marginalizing the joint distribution $p(\mathbf{x})$ with respect to a single variable X_e, the fundamental computational issues, for discrete representations, are the time complexity of combining the relevant random variable configurations and the space complexity of representing the result. In general we have that with $\bar{e} = V - \{e\}$,

$$\sum_{\mathbf{x}_e} p(\mathbf{x}) \propto h_2(\mathbf{x}_{\bar{e}}) \sum_{\mathbf{x}_e} h_1(\mathbf{x}_{\nu(e)}) = h_2(\mathbf{x}_{\bar{e}}) f(\mathbf{x}_{\bar{e}}). \qquad (2)$$

Even if the graph corresponding to p is a tree, representing $f(\mathbf{x}_{\bar{e}})$ without resorting to its functional form may require $\mathcal{O}(M^{|\nu(e)|-1})$ space. Further computations (like marginalizing with respect to another variable), referring to this result may also have exponential time complexity.

This exact operation can been seen as a step in a bucket elimination algorithm [3]. This is also the basic operation that data structures like the clique-tree or junction tree are designed to handle in exact inference methods like variable elimination [17], and illustrates why some elimination and induced triangulations are much more efficient than others, even though all perform exact calculations.

There are instances when f turns out to accept simple (*e.g.,* product) decompositions. In this case it is possible to improve upon the above complexity bounds on inference. However, it is not clear how to induce such decompositions given a distribution p. The essence of the focused inference algorithm explained in this section lies in variable elimination and in generating a succession of non-exact decompositions to compute optimal marginal approximations in the context of Eq. 2. We will discuss an extension of the basic algorithm later in the paper.

3.1 BASIC FI ALGORITHM

Let $p(\mathbf{x})$ be the distribution of interest, with associated factor graph $\mathcal{G} = (\{V, F\}, E)$, focused inference (FI) is based on a new graph decomposition together with an approximation that reduces the original distribution p (and graph $\mathcal{G}$) to a *simpler* one in an iterative manner, with certain optimality properties at each step. We can think of the essential process as *focusing* only on a few target node(s) at a time, whose marginal distributions (*e.g.,* pairwise) are to be approximated. Each iteration eliminates variable and factors, includes new factors, and modifies the distribution appropriately to keep the focused approximation accurate.

We start by formalizing FI for a single iteration and when the target variables consist of a specific pair of nodes $\mathcal{T}$, $\mathcal{T} \subset V$, and later generalize it to multiple pairwise marginals.

The first step of the iteration consists on choosing a non-target node $e \in V - \mathcal{T}$ and rewriting the corre-

[1]Note this need not be the case for joint marginals of p

sponding probability distribution as:

$$p(\mathbf{x}) = \tilde{p}_1(\mathbf{x}_{\nu(e)})\tilde{p}_2(\mathbf{x}_{\bar{e}}), \tag{3}$$

where the two new distributions are defined according to the following decomposition:

$$\tilde{p}_1(\mathbf{x}_{\nu(e)}) = \frac{1}{Z_{\tilde{p}_1}} \prod_{\alpha \in F(e)} \psi(\mathbf{x}_\alpha) \prod_{i \in \nu(e)} \phi_i(\mathbf{x}_i) \tag{4}$$

$$\tilde{p}_2(\mathbf{x}_{\bar{e}}) \propto \prod_{\alpha \notin F(e)} \psi(\mathbf{x}_\alpha) \prod_{i \notin \nu(e)} \phi_i(\mathbf{x}_i). \tag{5}$$

Assuming each factor involves at most two variables, $\tilde{p}_1(\mathbf{x}_{\nu(e)})$ is always a tree-structured distribution and thus computing exact marginals from $\tilde{p}_1$ can be done efficiently. $\tilde{p}_2(\mathbf{x}_{\bar{e}})$ remains generally intractable. The decomposition is not unique (even up to constant) since we are free to trade single node marginals between the components. Note that the decomposition itself involves no approximations, only rewriting of the original distribution.

The exact node/edge removal operation (marginalizing) consist on finding $f(\mathbf{x}_{\nu(e)^-}) = \sum_{\mathbf{x}_e} \tilde{p}_1(\mathbf{x}_{\nu(e)})$; see Fig. 1(b). The above decomposition is helpful since sensible approximations for the first portion of the full distribution (Eq. 4) are readily available. In particular, in this paper we employ the specific class of approximations that optimize the KL-divergence $D(f(\mathbf{x}_{\nu(e)^-})||q(\mathbf{x}_{\nu(e)^-}))$ between $f(\mathbf{x}_{\nu(e)^-})$ and the approximating distribution $q(\mathbf{x}_{\nu(e)^-})$ constrained to be a tree-distribution. We denote this class of approximating distributions by Q.

In the second step of the FI iteration we solve:

$$q(\mathbf{x}_{\nu(e)^-}) = \arg\min_{q \in Q} D(\sum_{\mathbf{x}_e} \tilde{p}_1(\mathbf{x}_{\nu(e)})||q(\mathbf{x}_{\nu(e)^-}). \tag{6}$$

This projection can be solved efficiently whenever Q is restricted to trees.

This optimization is related to that used by ADF (and thus EP); however in FI no fixed, predetermined reference(*e.g.,* tree-structured) distribution is set, at each step the structure of the best local approximating distribution can be obtained dynamically. The projection operation may (automatically) introduce factors that were not previously present. We clarify here that we are not assuming a specific choice of ADF *terms*. Also, recall than in ADF the structure of the approximating distribution is predetermined (not found by ADF itself).

One way to represent the solution to Eq. 6 is by the following tree-structured factorization:

$$q(\mathbf{x}_{\nu(e)^-}) = \prod_{(i,j) \in E_T} q(\mathbf{x}_i, \mathbf{x}_j) / \prod_{i \in V_T} q(\mathbf{x}_i)^{d_i - 1}, \tag{7}$$

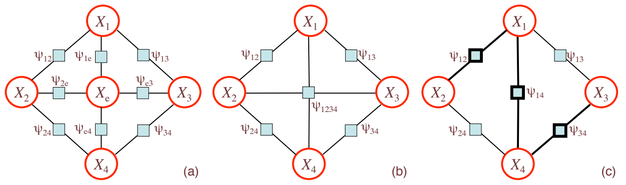

Figure 1: One-step approximation by removal of X_e: (a) original factor graph, (b) exact (marginalized) graph, and (c) example FI approximation where appropriate factors (in bold) have been created (ψ_{14}) and updated (ψ_{12}, ψ_{34})

where d_i denotes the degree of node i and $\mathcal{G}_T = (V_T, E_T)$ is a tree (q is in the family of distributions Q). The optimal tree $\mathcal{G}_T$ can be found efficiently.

In the third step of the algorithm we combine the projected approximation with the remaining variables to get the approximation to marginalized $p(\mathbf{x})$:

$$\hat{p}(\mathbf{x}_{\bar{e}}) = q(\mathbf{x}_{\nu(e)^-})\tilde{p}_2(\mathbf{x}_{\bar{e}}). \tag{8}$$

This node/edge elimination and approximation iteration is repeated until all but the focus set $\mathcal{T}$ is left in the graph [2]. The new distribution $\hat{p}(\mathbf{x}_{\bar{e}})$ is again defined in the form of Eq. 1. This involves redefining the previous potentials and incorporating new ones. The following provides the resulting factor/potential update equations [3]. For each pair $(i, j) \in E_T$, potentials can be modified or created:

$$\psi_\alpha(\mathbf{x}_\alpha) \leftarrow \psi_\alpha(\mathbf{x}_\alpha) \frac{q(\mathbf{x}_i, \mathbf{x}_j) q(\mathbf{x}_i)^{-\rho_i} q(\mathbf{x}_j)^{-\rho_j}}{\phi_i(\mathbf{x}_i)\phi_j(\mathbf{x}_j)} \text{ (modify)} \tag{9}$$

$$\psi_\alpha(\mathbf{x}_\alpha) \leftarrow \frac{q(\mathbf{x}_i, \mathbf{x}_j) q(\mathbf{x}_i)^{-\rho_i} q(\mathbf{x}_j)^{-\rho_j}}{\phi_i(\mathbf{x}_i)\phi_j(\mathbf{x}_j)} \text{ (create),} \tag{10}$$

where $\rho_i = (d_i - 1)/d_i$ and the potential is modified whenever both (i, α) and (j, α) are in E (created otherwise). The graphical operations for factor graph $\mathcal{G}$ are variable node, factor, edge removal, and edge addition, respectively:

$$\begin{array}{ll} (i) \quad V \leftarrow V - \{e\}, & (ii) \quad F \leftarrow F - F(e), \\ (iii) \quad E \leftarrow E - \{(e, \alpha) | \alpha \in F(e)\}, & \\ (iv) \quad E \leftarrow E \cup \{(i, \alpha), (j, \alpha) | (i, j) \in E_T\} & \end{array}$$

One iteration applied to a simple five-variable factor graph is shown in Fig. 1. The repetition of these steps defines an elimination ordering $\mathcal{E} = (e_1, ..., e_K)$ and a series of approximating distributions $\{\mathcal{Q}_k\}_{k=1,...,K}$ which characterize one focusing operation. While these basic steps are fixed, the global algorithm is more flexible; for example, in the choice of approximating distributions, in the elimination ordering, etc. These and other aspects will be further discussed in the next section.

[2] Alternatively until a tractable substructure containing the set $\mathcal{T}$ has been reached

[3] This is one succinct way to state the update equations for multinomials, other equivalent forms can be also used. This form does not require updating the potentials ϕ_i

4 ALGORITHM ANALYSIS

Here we illustrate key properties of the algorithm and provide additional details.

4.1 ALGORITHM COMPLEXITY AND OPTIMALITY

Under the decomposition defined in Eqs. 4-5, the minimization problem in Eq. 6 for a fixed tree structure has a known solution in $\mathcal{O}(M^3)$. The problem reduces to finding the pairwise marginals of $\tilde{p}_1$ along the tree edges E_T. Since $\tilde{p}_1$ is always a tree (a star graph centered at X_e) and each potential ψ is a function of at most two random variables, any of these marginals can be found in $\mathcal{O}(M^3)$. As for finding the best tree-structured distribution family in Q, this result follows directly from [1] applied to our decomposition.

Proposition 1 *The focused inference algorithm of Sec. 3 is exact for any decimatable distribution p (tree-width two or, equivalently, when the maximal clique size of the triangulated graph is three). Not all elimination orderings yield the exact result.*

Proof The approximation is exact when all the steps are exact. The steps are exact if each variable has at most two neighbors when eliminated. Since the maximal clique size is three, this elimination constraint can always be satisfied through some elimination ordering.

An analogous result may not hold for ADF (or EP) with the same time complexity.

4.2 ELIMINATION ORDERING

For a graph $\mathcal{G} = (\{V, F\}, E)$ and for a set of nodes $\mathcal{F}$ of interest, an elimination ordering is a sequence of nodes $\mathcal{E} = (e_1, ..., e_K)$ with $e_i \in V - \mathcal{F}$. In case we measure the approximation accuracy in terms of the KL-divergence, the focused inference method suggests a seemingly natural elimination ordering $\mathcal{E}$: at each step, eliminate the (non-target) node that gives the lowest KL divergence $D(f(\mathbf{x}_{\nu(e)^-})||q(\mathbf{x}_{\nu(e)^-}))$ between marginalized and approximating distribution at each step. However, note that this gives a locally best and, in general, not a globally best ordering. Finding approximations to the best overall elimination ordering is a much harder problem due that the complexity of the problem representation increases rapidly with the number of elimination steps.

The focused inference algorithm is designed to concentrate on specific marginals and thus, a particular ordering is used to reduce the graph to those nodes of interest. In the most general setting, the algorithm has to be *run* again if other marginals are needed (or partially run since common calculations or partial results could be handily stored). A reasonable question is whether these marginals are consistent. The answer is negative, in general. Specifically, for a distribution $p(\mathbf{x})$ and two sets of focus (target) variables $\{X_a, X_b\}$ and $\{X_b, X_c\}$, the corresponding pairwise and single marginals produced by the focused inference algorithm under different elimination orderings $\mathcal{E}_1 = (e_{11}, ..., e_{1K_1})$ and $\mathcal{E}_2 = (e_{21}, ..., e_{2K_2})$ are consistent (1)if $p(\mathbf{x})$ is a decimatable distribution; since the algorithm is exact, and (2)if the elimination orderings are the same except for the final node: $K_1 = K_2 = N - 2$ and $e_{1i} = e_{2i}$ for $i < K$; since after eliminating $N - 3$ nodes, the remaining variables will be X_a, X_b, X_c, and their joint distribution is decimatable.

If we do not require that the approximation be optimal (at each step), the graph can be reduced to a tree very quickly. Moreover, this can be done so that the pairwise distributions are tree consistent. However, clearly this involves non-optimal (local) approximations.

4.3 CONSISTENCY OF SINGLE AND PAIRWISE MARGINALS

Let us instead consider how to start from the potentially inconsistent set of marginals found using the FI algorithm and reach a consistent set of marginals. We consider the following problem: given a set of pairwise marginals found under different elimination orderings, how can we obtain a set of consistent marginals with respect to a tree graph $\mathcal{G}_{T'} = (V_T, E_{T'})$.

Our approach consist on using a maximum likelihood criterion; specifically, let $\mathcal{M}$ be the set of ordered pairs (i, j) of marginals $\{q(\mathbf{x}_i, \mathbf{x}_j)\}$. Under this criterion, we wish to solve the following optimization problem:

$$\arg\max_{r \in R} - \sum_{(i,j) \in \mathcal{M}} q(\mathbf{x}_i, \mathbf{x}_j) \log r(\mathbf{x}_i, \mathbf{x}_j), \tag{11}$$

for the set of distributions R with model structure $\mathcal{G}_{T'}$. This problem is equivalent to the minimization of Eq. 6, and thus can be solved in closed-form.

4.4 INCLUDING SINGLE NODE POTENTIALS

The decomposition given by Eqs. 4- 5 is an instance of a more general decomposition which generalizes the way single node potentials are included as follows:

$$\tilde{p}_1(\mathbf{x}_{\nu(e)}) \quad \propto \quad \phi_e(\mathbf{x}_e) \prod_{\alpha \in F(e)} \psi(\mathbf{x}_\alpha) \prod_{i \in \nu(e)^-} \phi_i(\mathbf{x}_i)^{\eta_i} \tag{12}$$

$$\tilde{p}_2(\mathbf{x}_{\bar{e}}) \quad \propto \quad \prod_{\alpha \notin F(e)} \psi(\mathbf{x}_\alpha) \prod_{i \notin \nu(e)} \phi_i(\mathbf{x}_i) \prod_{i \in \nu(e)^-} \phi_i(\mathbf{x}_i)^{(1-\eta_i)}, \tag{13}$$

which subsumes the original one.

The extra degrees of freedom are given by the variables $\eta = \{\eta_i\}, i \in \nu(e)^-$, $\eta_i \in \Re$ (note that $\phi_e(\mathbf{x}_e)$ must be fully included during e's elimination). This extra flexibility could be used to our advantage in finding a better distribution when minimizing Eq. 6. This is because single node potentials could be included such as to obtain an approximating q distribution giving smaller KL-divergence. However, one must be cautious, since we can almost always define η to obtain an approximating distribution for which the KL-divergence is almost zero. This can be done by allowing almost all of p's probability mass to fall in appropriate variable states easily represented by distributions in Q. Yet, despite this momentary success, the overall gain is not guaranteed to be larger since the resulting $\hat{p}$ distribution might be difficult to approximate in future steps. This may allow us to provide more accurate overall approximations by appropriately including the effect of single node potentials. Taking advantage of this generalization is an interesting problem that remain to be exploited.

4.5 FURTHER CONNECTIONS WITH OTHER METHODS

As a way to further understand FI, we now discuss other connections with related methods. Consider a single inclusion of a pairwise term in ADF. The marginals obtained by ADF for any (tractable) approximating distribution could also obtained by FI. This can be seen by noticing that FI can perform exact marginalization in a cycle (like ADF) and the inclusion of a pairwise term in ADF will at most generate a cycle. It is significant that for this equivalence to hold, FI needs to make locally suboptimal decisions and ignore those edges not in the cycle, even if they can be easily approximated during elimination. For simultaneous inclusion of multiple terms, ADF's complexity in general increases exponentially, FI's complexity remains as before, of course using an approximation.

Unlike ADF and EP, in FI there is no reference structure for the approximation made. Interestingly, intermediate (joint) approximations built by FI may not be tractable. Their structure can be chosen with locally optimal guarantees. FI builds these approximations dynamically, thus there are less choices to be made by hand regarding the structure of the approximating distribution. This is related to [5], in the sense that different approximating structures are found at each step; however, in [5] a (tractable) tree-structured joint distribution is maintained at all times.

There exist certain resemblance between the FI algorithm in Sec. 3.1 and the mini-buckets scheme [4] in the sense that both methods repeatedly approximate complex functions of multiple variables with products of simpler functions. In mini-buckets, the local approximations employ a non explicitly guided partitioning of variables to functions (a partitioning, to some extent corresponds to the structure of a local approximating distribution in the FI method). In FI, contrary to mini-buckets, the approximating distributions q, including their structure, can be solved for, and are optimal with respect to a definite criterion, the appropriate KL-divergence.

In mini-buckets the approximation relies on arithmetic bounds on product of positive functions. In a criterion derived from mini-buckets [11], this approximation is given by the solution of a linear optimization problem (thus more like FI). However, still the structure of the approximating distribution is not part of the formulation and also the optimization problem entails using an exponential number of constraints.

5 DISTRIBUTED FOCUSED INFERENCE

Let us say we are interested in the marginal distribution for all of the variables X_i. In the worst case, the basic FI algorithm needs to be re-run on the order of N times to obtain all marginals of interest. Is there a more efficient way to perform the necessary computations? In this section we address the question whether there exist a distributed (asynchronous) algorithm equivalent or based on the same fundamental ideas. There is a parallel extension for exact methods, where this is of course the case in inference (*e.g.,* [17, 16, 12]). Asynchronous algorithms also exist for fixed structure approximations (*e.g.,* [13, 7, 14]). However, note that in the FI approximation, the result of eliminating one variable is not propagated symmetrically through the graph (neighbors), it depends on the factorization chosen; the underlying factor graph is dynamically modified, based on previous approximations to other parts of the graph; and different elimination orderings (optimal in some sense for a particular focusing variable) are used for computing the different marginals. Thus, it is not clear for example, what data structure fits the underlying algorithm, what information or quantities a node should transfer to another, and if the overall algorithm allows for quantities to be stored efficiently, *e.g.,* locally.

It turns out that for a type of FI approximations, we can build a distributed algorithm and answer these questions. In order to define the algorithm, a tree structure similar to the clique-tree [12, 17] can be used for the underlying computations. However, unlike the above, in our case it is not necessary to find the maximal cliques or use the concept of triangulation explicitly. This is important because finding maximal cliques

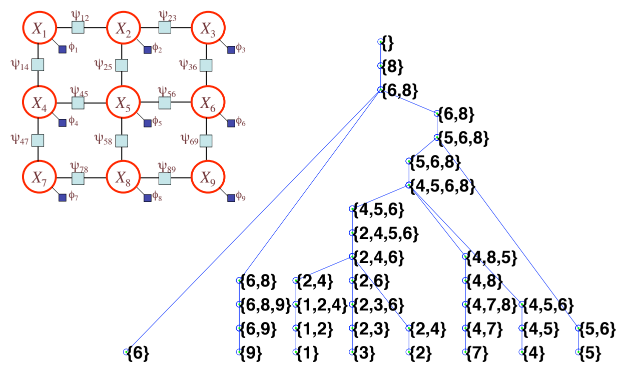

Figure 2: Example factor graph and the tree induced by the ordering $\mathcal{E} = (1, 3, 7, 9, 2, 4, 5, 6, 8)$

is in NP-complete [8]. The following algorithm defines the necessary tree structure whose nodes, denoted $\mathcal{A}_j$, are subsets of the random variables in p:

Algorithm for building Order-induced Tree

Denote elimination ordering $\mathcal{E} = (e_1, ..., e_n)$

1. Assign a single variable to the initial tree nodes: $\mathcal{A}_j = \{X_{e_j}\}$ $(j = 1, ..., N - 1)$
2. Iterate $i = 1...N$
   (a) Create new node $\mathcal{C} = \bigcup_j \mathcal{A}_j$ for j s.t. $\mathcal{A}_j$ does not have a parent **and** $X_{e_i} \in \mathcal{A}_j$
   (b) For each j found in (a) Let $\mathcal{B} = \mathcal{C} - \mathcal{A}_j = \{X_{b_l}\}$ and chain nodes: $(\mathcal{C}) \rightarrow (\mathcal{C} \backslash \{X_{b_1}\}) \rightarrow ... \rightarrow (\mathcal{C} \backslash \mathcal{B}) \rightarrow (\mathcal{A}_j)$
   (c) For all the (not eliminated) variables X_k sharing a factor with X_{e_i}
      i. Create new node $\mathcal{C}' = \mathcal{C} \cup X_k$ and make $\mathcal{C}'$ a parent of $\mathcal{C}$
      ii. Redefine $\mathcal{C} \leftarrow \mathcal{C}'$
   (d) Create new node $\mathcal{C}' = \mathcal{C} - \{X_{e_i}\}$ and make $\mathcal{C}'$ a parent of $\mathcal{C}$
   (e) Eliminate X_{e_i}

Since we do not need the concept of cliques, we simply call it an *Order-induced Tree* (OT). Note that when traversed bottom-up (to the root), this tree gives a marginalization ordering that properly tracks the resulting function arguments at the nodes in the order given. As in the basic FI, the above steps resemble bucket elimination [3]. In fact, at the graph level, the OT algorithm performs the variable inclusion and elimination operations in the same order as FI. An example OT tree for a simple 3×3 grid is shown in Fig. 2. The distributed algorithm (shown next) uses the OT as basic data structure for message passing.

Distributed Focused Inference Algorithm

1. Form OT and for each node $\mathcal{A}$ assign function:

$$\tilde{\psi}_{\mathcal{A}}(\mathbf{x}_{\mathcal{A}}) = \prod_{\alpha \in F(\mathcal{A})} \psi_\alpha(\mathbf{x}_\alpha) \prod_{i \in \mathcal{A}} \phi_i(\mathbf{x}_i), \quad (14)$$

   $F(\mathcal{A})$: set of factors whose variables are in $\mathcal{A}$

2. Pick any node in the tree as root node
3. Perform a bottom-up and top-down pass, sending the following message between neighbor nodes (random variable sets) $\mathcal{A}$ and $\mathcal{B}$:

$$m_{\mathcal{B} \rightarrow \mathcal{A}}(\mathbf{x}_{\mathcal{A}}) = \wp_{\mathbf{x}_{\mathcal{B} \backslash \mathcal{A}}}[\prod_{\mathcal{C} \in \nu(\mathcal{B})} \frac{m_{\mathcal{C} \rightarrow \mathcal{B}}(\mathbf{x}_{\mathcal{B}})}{\tilde{\psi}_{\mathcal{C} \cap \mathcal{B}}(\mathbf{x}_{\mathcal{C} \cap \mathcal{B}})} \tilde{\psi}_{\mathcal{B}}(\mathbf{x}_{\mathcal{B}})], \quad (15)$$

   where ν denotes neighborhood in the OT

4. Compute marginals for the nodes of interest:

$$p(\mathbf{x}_{\mathcal{A}}) \propto \prod_{\mathcal{B} \in \nu(\mathcal{A})} \frac{m_{\mathcal{B} \rightarrow \mathcal{A}}(\mathbf{x}_{\mathcal{A}})}{\tilde{\psi}_{\mathcal{B} \cap \mathcal{A}}(\mathbf{x}_{\mathcal{B} \cap \mathcal{A}})} \tilde{\psi}_{\mathcal{A}}(\mathbf{x}_{\mathcal{A}}) \quad (16)$$

   (*e.g.,* use the nodes $\mathcal{A}$ containing the single variables of interest; some joint marginals can be computed directly as well)

In the algorithm, the operator $\wp$, has a similar role than the minimization in Eq. 6. In the case of FI we have:

$$\wp_{\mathbf{x}_{\mathcal{B} \backslash \mathcal{A}}}[g] \stackrel{\triangle}{=} \arg\min_{q \in \mathcal{Q}} D(\frac{1}{Z_g} \sum_{\mathbf{x}_{\mathcal{B} \backslash \mathcal{A}}} g(\mathbf{x}_{\mathcal{B}}) || q(\mathbf{x}_{\mathcal{A}})), \quad (17)$$

which is the known projection operation in information geometry (as before $\mathcal{Q}$ is the set of tree structure-distributions). Since D is defined on distributions, Z_g is the necessary normalization constant. We defer a detailed analysis of the above algorithm and present the basic results in the remaining of this section.

Theorem 2 *The distributed algorithm computes the exact marginals when the minimization operation is replaced by exact summation, i.e., when* $\wp_{\mathbf{x}_{\mathcal{B} \backslash \mathcal{A}}}[g] \stackrel{\triangle}{=} \frac{1}{Z_g} \sum_{\mathbf{x}_{\mathcal{B} \backslash \mathcal{A}}} g(\mathbf{x}_{\mathcal{B}})$

Proof sketch It suffices to show that the new definition of $\wp$ is equivalent to solving for f in Eq. 2, and that sequential application of step 3 with any root node is equivalent to sequential application of Eq. 2 in a particular ordering.

From the above result, step 3 using Eq. 17 can be seen as passing (approximate) distributions over variables in the target nodes. Interestingly these distributions may be intractable themselves. The basic FI

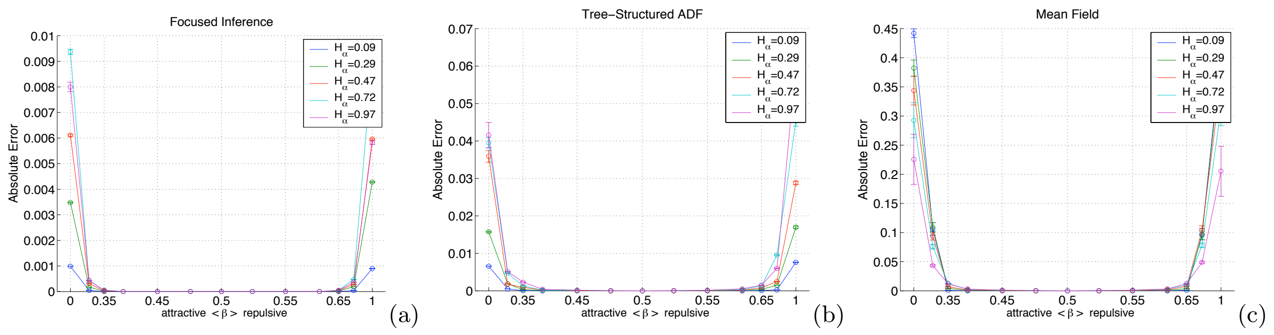

Figure 3: Numerical test results for grid networks with random single and pairwise potentials with various levels of entropy bounds (H_α,I_β). Note x-axis scale is given in terms of I_β and networks with maximally attractive and repulsive potentials are at $\beta = 0$ and $\beta = 1$ respectively. Performance of: (a) FI, (b) ADF, and (c) MF (y-axis scale varies)

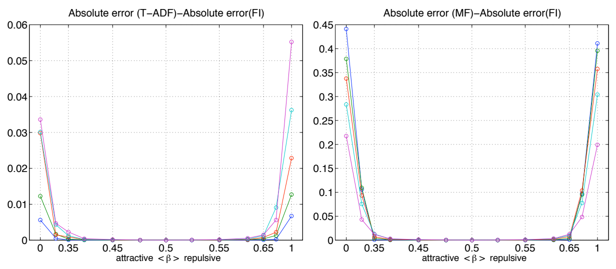

Figure 4: Absolute error differences (see Fig. 3 for legend)

algorithm, focusing on a single marginal, is equivalent to (1)choosing an OT with same ordering $\mathcal{E}$ and (2)performing just one pass (to the root). However, the marginals computed by running the basic FI algorithm for multiple focusing sets are not necessarily the same as those computed by the distributed algorithm. This can be easily seen by observing that different approximations are made in each case. The distributed algorithm is equivalent to multiple runs of FI where every run respects the approximations induced by the OT by means of its variable subsets and tree arrangement. The approximation over the target variables can vary in structure (*i.e.,* we can still choose optimal tree-structure distributions for message passing).

6 NUMERICAL EVALUATION

In order to test how FI performs for diverse types of distributions, we constructed a number of binary 9 × 9 grid networks by choosing its factors $\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \theta_{\mathbf{x}_i\mathbf{x}_j}$ according to different uniform priors. Specifically, we use a hyper parameter β to define the random variable $b \sim \mathcal{U}(\frac{1}{2}, 1-\beta)$ and set $\theta = (\theta_{00}, \theta_{01}; \theta_{10}, \theta_{11}) = (\frac{b}{2}, \frac{1-b}{2}; \frac{1-b}{2}, \frac{b}{2})$. When letting $0 < \beta < \frac{1}{2}$, attractive potentials with varying strength are constructed. Similarly, by letting $\frac{1}{2} < \beta < 1$ and $b \sim \mathcal{U}(1-\beta, \frac{1}{2})$ we define repulsive potentials. By varying β we control the (maximum allowed) mutual information I_β describing how dependent the states of node pairs are: maximum dependency is achieved at $\beta = 0$ (attractive) and $\beta = 1$ (repulsive), and full independency at $\beta = \frac{1}{2}$.

A second hyper parameter α controls the distribution of single node potentials. Let $\phi_i(\mathbf{x}_i) = \theta_{\mathbf{x}_i}$, we define $a \sim \mathcal{U}(\frac{1}{2} - \alpha, \frac{1}{2} + \alpha)$, for $0 < \alpha < \frac{1}{2}$ and let $(\theta_0, \theta_1) = (a, 1-a)$. Thus, α controls the entropy in the prior state of a single variable (associated to its single node potential); when $\alpha = \frac{1}{2}$ the minimum entropy allowed, denoted H_α, is 1 bit (full uncertainty), and when $\alpha = 0$ it is 0 bits. In our experiments we varied β and α to obtain probability distributions with different properties regarding strength of dependences and strength of value preference.

In all of the experiments, we used the basic FI algorithm (no consistency was enforced). We chose the node to be eliminated at each step simply by looking at the number of neighbors of each node in the intermediate graphs and picking those with less neighbors first, randomly when tied. For ADF, we chose the structure of the approximating distribution by keeping the most informative edges (maximizing the pairwise mutual information) forming a spanning tree. This criterion performed better than random edge selection.

Fig. 3(a) shows the accuracy of FI for probability distributions sampled under different settings of the hyper-parameters β and α. Performance is measured in terms of the average absolute difference between exact and approximate (single node) marginals. As expected the performance improves as the coupling between the nodes become weaker ($\beta \to \frac{1}{2}$) for all values of α. We performed the same tests using ADF and the variational Mean Field method. Fig. 3(b)(c) shows the performance results from these methods. Like FI, accuracy is higher at $\beta \approx \frac{1}{2}$ for any α.

FI clearly outperforms ADF under all conditions (Fig. 4-left). The ADF *terms* were the pairwise factors;

FI and ADF had equivalent computational complexity. Notably, the difference in performance increased with the strength of the dependences in the network and also with the strength of the *field* given by the single node potentials. Focused inference also outperformed Mean Field (MF) under all conditions (Fig. 4-right), even in the case when the basic Mean Field assumption is almost fully valid (when variables are almost independent). As variable dependencies grew stronger, the performance gap between FI and the competing methods became larger at increasing rates.

7 CONCLUSIONS

We introduced an approximate inference algorithm, similar to variable elimination, that is based on tailoring the approximation to the subset of variables of interest. We also developed a distributed message-passing version of the algorithm, constructed around a particular elimination ordering.

The basic decomposition step, followed by the projection, can be guaranteed to be optimal for decimatable graphs and properly chosen elimination ordering. We are not aware of similar results for ADF. In a more general context, the advantage of the focused inference algorithm lies primarily in the inclusion of dependencies induced by marginalization but not represented in the original graph. FI does not require setting a fixed reference distribution (*e.g.,* a class of tractable approximating distributions) for defining the approximation. The selection of dependencies to introduce is based on optimizing the projection of each local marginalization result down to a tree. The ability to maintain such dependencies through approximate marginalizations may underlie the superior empirical results.

Acknowledgements

The authors gratefully acknowledge support from DARPA IPTO program and British Aerospace Engineering.

References

[1] C. Chow and C. Liu, *Approximating discrete probability distributions with dependence trees*, Trans. Information Theory. **14** (1968).

[2] P. Dagum and M. Luby, *Approximating probabilistic inference in bayesian belief networks is NP-hard*, Artificial Intelligence **60** (1993), no. 1, 141–153.

[3] R. Dechter, *Bucket elimination: A unifying framework for reasoning*, Artificial Intelligence **113** (1999), no. 1-2, 41–85.

[4] R. Dechter and I. Rish, *Mini-buckets: A general scheme for approximating inference*, J. ACM **50** (2003), no. 2, 107–153.

[5] B. J. Frey, R. Patrascu, T. Jaakkola, and J. Moran, *Sequentially fitting inclusive trees for inference in noisy-or networks*, Neural Info. Proc. Systems, 2000.

[6] Z. Ghahramani and M. Jordan, *Factorial hidden markov models*, Neural Info. Proc. Systems, 1997.

[7] M. Jordan, Z. Ghahramani, T.Jaakkola, and L. Saul, *An introduction to variational methods for graphical models*, Learning in Graphical Models, M. Jordan (ed.) (1998).

[8] R. M. Karp, *Reducibility among combinatorial problems*, In R. E. Miller and J. W. Thatcher, eds., Complexity of Computer Computations (1972), 85–104.

[9] U. Kjærulff, *Reduction of computational complexity in Bayesian networks through removal of weak dependencies*, Proc. Uncert. in Artif. Intell., 1994.

[10] F. Kschischang and B. Frey, *Iterative decoding of compound codes by probability propagation in graphical models*, J. Sel. Areas in Comm. **16** (1998), 219–230.

[11] D. Larkin, *Approximate decomposition: A method for bounding and estimating probabilistic and deterministic queries*, Proc. Uncert. in Artif. Intell., 2003.

[12] S. L. Lauritzen and D. J. Spiegelhalter, *Local computations with probabilities on graphical structures and their application to expert systems*, J. Royal Stat. Society, B **50** (1988), no. 2, 157–223.

[13] P. Maybeck, *Stochastic models estimation and control*, Academic Press, 1982.

[14] T. Minka, *A family of algorithms for approximate Bayesian inference*, Ph.D. thesis, MIT, 2001.

[15] T. Minka and Y. Qi, *Tree-structured approximations by expectation propagation*, Neural Info. Proc. Systems, 2003.

[16] J. Pearl, *Probabilistic reasoning in intelligent systems*, Morgan-Kaufman, 1988.

[17] G. Shafer and P. Shenoy, *Probability propagation*, Ann. Math. Artificial Intel. **2** (1990), 327–351.

[18] M. Wainwright, T. Jaakkola, and A. Willsky, *A new class of upper bounds on the log partition function*, Proc. Uncert. in Artif. Intell., 2002.

[19] ———, *Tree-based reparameterization framework for analysis of sum-product and related algorithms*, Trans. Information Theory. **49-5** (2003).

[20] J. Yedidia, W. Freeman, and Y. Weiss, *Generalized belief propagation*, Neural Info. Proc. Systems, 2000, pp. 689–695.