A general framework for comparing (approximate) inference algorithms

Frank Hutter
Computer Science Department
The University Of British Columbia
201-2366 Main Mall
Vancouver, B.C., V6T 1Z4
hutter@cs.ubc.ca

Sohrab Shah
Computer Science Department
The University Of British Columbia
201-2366 Main Mall
Vancouver, B.C., V6T 1Z4
sshah@cs.ubc.ca

1 Introduction

Conclusive comparisons of approximate inference algorithms are difficult due a number of handicaps. The main problem is that most algorithms were developed independently for different domains and consequently have different input and output representations and formulations, for example, Bayesian networks vs Markov random fields (MRFs) [13]. Thus, researchers usually only compare novel approximate inference algorithms against a small subset of well-known algorithms instead of the current state-of-the-art. Analogously, practitioners usually employ well-known algorithms developed for their problem at hand, ignoring potentially superior algorithms that only require a slightly modified problem formulation.

We developed a simple framework that automates the task of converting between different problem formulations and offers a unified interface to a variety of approximate inference algorithms. This framework promises to facilitate straightforward comparisons of an arbitrary set of approximate inference algorithms which to date could only be done with considerable overhead. Consequently, a new algorithm's performance can now be compared against other published solutions in a standard way with minimal overhead. Furthermore, when assessing the best algorithm for a given problem, one can simply call the desired set of algorithms with a unified interface and compare the results.

The unified Matlab interface we implemented to date interfaces to implementations of a number of approximate inference algorithms for pairwise Markov random fields in the programming language C [1], as well as to our own Matlab implementation of the exact variable elimination algorithm in factor graphs and a Matlab implementation of loopy belief propagation.[2] For Bayesian networks, Markov random fields, factor graphs, and pairwise MRFs, we are mainly interested in two problems, namely the one of computing marginal probabilities for every random variable in the network and the one of computing the maximum a posteriori (MAP) instantiation of all variables. Both of these problems have applications in computer vision [2, 10, 12] and a great number of other areas including problems from so heterogneous areas as medical diagnosis, speech recognition, side-chain prediction in protein folding, and computer diagnosis [5].

For the task of computing marginals for single variables, our system currently interfaces to Gibbs sampling (GS) [9] which is prominent in the AI literature, as well as the cluster sampling algorithms Swendsen-Wang (SW) [11] and Wolff (WO) [14] with roots in statistical physics. Furthermore, it interfaces to the Mean-Field algorithm [13] and the sum-product versions of the iterative message passing algorithms loopy belief propagation (BP) [9], and generalized BP (GBP) [17].

For the MAP task, our system currently only supports the max-product versions of BP and GBP. In the future, we would like to compare these algorithms against prominent methods based on graph cuts (GC) [2, 12] as well as recent Stochastic Local Search algorithms [5]. Since adding new inference algorithms to our framework is very simple, we hope to be able to carry out this comparison based on the existing framework.

To both test the system and provide use cases of the system, we carried out a number of experiments on real-world BNs, randomly generated pairwise MRFs and on an image segmentation problem.

The rest of this report is structured as follows. Section 2 formal describes several compact representations for high-dimensional probability distributions, as well as conversions between them. In Section 3, we then sketch out a number of prominent approximate inference algorithms and their applicability. Section 4 gives an overview of the architecture of our system, followed by a description of our experiments and their results in sections 5 and 6, and conclude in Section 7.

[1] Thanks to Talya Meltzer for providing implementations of these algorithms.

[2] Thanks to Kevin Murphy for providing his code.

2 Representation

In this section, we introduce several compact representations for high-dimensional probability distributions, conversions between them and algorithms to compute marginal probabilities of single variables as well as the most likely instantiation of all variables.

A discrete *Bayesian network* $\mathcal{B}$ is a tuple $\langle \mathcal{G}, \Phi \rangle$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a directed acyclic graph (DAG) whose nodes represent discrete random variables, and Φ is an ordered set of conditional probability tables (CPTs) $\phi_V = P(V|pa(V))$, specifying the conditional probability distribution of each $V \in \mathcal{V}$ given its parents in $\mathcal{G}$. The set $fa(V) = \{V\} \cup pa(V)$ is called V's *family*. The semantics of a Bayesian Network is that it specifies a joint probability distribution ϕ over its variables $\mathcal{V}$ in factored form: $\phi = P(\mathcal{V}) = \prod_{V \in \mathcal{V}} \phi_V$.

A discrete *Markov Random Field* (MRF) $\mathcal{B}$, also known as a *Markov network,* is a tuple $\langle \mathcal{G}, \Psi \rangle$, where $\mathcal{G}$ is an undirected graph whose nodes represent discrete random variables, and Ψ is a set of non-negative potential functions associated with cliques of $\mathcal{G}$. More specifically, each $\psi \in \Psi$ is associated with a set of variables $C \subseteq \mathcal{V}$ that form a not necessarily maximal clique in $\mathcal{G}$, and ψ assigns a non-negative value to every instantiation of C. Like a Bayesian network, an MRF encodes a joint probability distribibution ϕ over its variables $\mathcal{V}$, but since its potential functions are not required to define probabilities, a normalization constant Z is necessary to compute probabilities. The joint probability of an MRF is given by $\phi = 1/Z \times \prod_{\psi \in \Psi} \psi$. Based on its origins in statistical physics, the normalization constant is called the *partition function* and can be computed as $Z = \sum_{\mathcal{V}} \prod_{\psi \in \Psi} \psi$.

Both Bayesian networks and MRFs can be easily converted to a simple and highly general representation called factor graphs. A *factor graph* is a bipartite graph $\mathcal{FG} = (\mathcal{V} \cup \Psi, \mathcal{E})$ with one parition $\mathcal{V}$ representing variables, one partition representing potential functions, and with edges $\{v, \psi\} \in \mathcal{E}$ indicating that variable $v \in \mathcal{V}$ is in the scope of potential function $\psi \in \Psi$. A factor graph explicitly represents a non-negative function over $\mathcal{V}$ in factored form:

$$f(\mathcal{V}) = \prod_{\psi \in \Psi} \psi(\{V \in \mathcal{V} | \{\psi, v\} \in \mathcal{E}).$$

This function can be normalized to become a probability distribution by division by $Z = \sum_{\mathcal{V}} f(\mathcal{V})$. A Bayesian network can be converted to a factor graph by creating a node for every variable and every CPT of the Bayesian network and connecting the node for every CPT ϕ_V to each variable in V's family $fa(V)$. Similarly, the conversion of an MRF to a factor graph results in a node for every MRF node, as well as a node for every MRF potential function ψ that is connected to the nodes representing ψ's set of associated variables.

Of particular interest in many applications are so-called *pairwise MRFs* (MRF2s) whose potential functions are constrained to be associated with at most two variables. Since a number of algorithms can be easily formulated for pairwise MRFs, it is interesting that discrete networks in any of the above representations can be converted to pairwise MRFs. For such a conversion, Bayesian networks and MRFs can first be converted to factor graphs. The conversion of factor graphs to MRF2s introduces a new *mega-node* for every potential function of the factor graph. Such a mega-node has a domain size that equals the number of entries in the original potential function, has the original potential function as local evidence, and has deterministic pairwise potentials connecting it to its associated variables. For details and an example of this conversion, see [17].

Denoting the pairwise potentials between variables x_i and x_j in an MRF2 by ψ_{ij} and the local evidence potentials associated with every variable x_i by ϕ_i, the joint probability of an MRF2 under variable instantiation $\mathbf{x} = (\mathbf{x_1}, \dots, \mathbf{x_n})$ can be written as

$$p(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{Z}} \prod_{\mathbf{ij}} \psi_{\mathbf{ij}}(\mathbf{x_i}, \mathbf{x_j}) \prod_{\mathbf{i}} \phi_{\mathbf{i}}(\mathbf{x_i}).$$

Driven by results from statistical physics as well as practical concerns from applications such as computer vision, a number of specialized MRF2s have been proposed. The first such model was the Ising Model, proposed in 1925 by Ernst Ising to model a system of interacting parallel or antiparallel spins [15], possibly embedded in a magnetic field. Parallel adjacent spins are energetically desirable while antiparallel spins require more energy. The Ising model expresses this by means of interaction terms J_{ij} between pairs of adjacent nodes i and j which are 1 if their spins match and -1 otherwise. The magnetic field is modelled as the field strength $h_i(x_i)$ at every node i.

Any state of the system (a complete assignment **x** of up- or down-spins to each of the nodes) then exhibits the following free energy that is to be minimized across all complete assignments:

$$E(\mathbf{x}) = -\sum_{ij} J_{ij}(x_i, x_j) - \sum_i h_i(x_i).$$

With interaction terms $J_{ij}(x_i, x_j) = ln(\psi_{ij}(x_i, x_j))$, a magnetic field of $h_i(x_i) = ln(\phi(x_i, y_i))$, and temperature $T = 1$, this energy then corresponds to a probability in an MRF2 via Boltzmann's law from statistical mechanics [17]:

$$p(\mathbf{x}) = \frac{1}{Z} exp(-E(\mathbf{x})/T).$$

Since the form of pairwise potentials decides about the applicability of prominent algorithms like Graph-Cuts [1, 2, 6], we explicitly note that the Ising model exhibits pairwise potentials

$$\Psi_{ij} = \left(\begin{array}{cc} exp(J_{ij}) & exp(-J_{ij}) \\ exp(-J_{ij}) & exp(J_{ij}) \end{array} \right).$$

The original problem formulation required all interaction terms J_{ij} to be the same, but subsequently, the generalized Ising model allowed them to differ for every pair of nodes. One prominent example of Ising models are *spin glasses*, which exhibit a phase transition in problem hardness as the J_{ij} get negative, indicating a so-called *frustration* in systems where every node prefers to be in a different state than its neighbours.

The *Potts model* [15] generalizes the Ising model to non-binary variables in a straight-forward fashion. In this model, the interaction term between adjacent nodes i and j in the graph is 0 for matching values and 1 otherwise. The Ising and Potts model are traditionally defined for grid-structured MRF2s, but to our best knowledge, almost no exact or approximate inference algorithm is bound to only work on grid-structured networks. The only exception we are aware of is an encode-and-solve approach where each layer of the MRF2 is viewed as a single variable in a Hidden Markov Model which can then be solved by the prominent forwards-backwards algorithm [8]. Other specialized algorithms for grid-structured networks, such as specialized versions of loopy belief propagation, only require this representation due to programming language specific features, such as efficient vectorization of parallel updates in Matlab.

Grid-structured pairwise MRFs are frequently employed for a variety of problems in computer vision where an image can be viewed as a grid-structured MRF2 with one variable per pixel or patch of pixels [2, 10, 12, 6]. The pairwise potentials in this domain enforce conformity constraints and smoothness between neighbouring variables, whereas potentials for single variables encode domain-dependent local evidence. Both the Ising and the Potts model are employed in computer vision, and for the latter one a variety of modifications have been proposed. For larger domain sizes, these generalizations assign a penalty to differing values of adjacent variables i and j that depends on the actual difference $d(x_i, x_j)$ between values x_i and x_j. Linear or quadratic interaction penalties are used, but a cutoff is essential in order to prevent oversmoothing at object boundaries.

3 Algorithms and their applicability

In this section, we sketch out some prominent inference algorithms and indicate which models they can be applied for.

Belief propagation (BP) has originally been introduced as an exact algorithm for tree-structured models [9], but is can also be applied for graphs with loops, in which case it becomes an approximate algorithm. For notational convenience, we explain BP for MRF2s; [17] states that this is mathematically equivalent to BP on other graphical models like factor graphs or Bayesian networks. For MRF2s, BP is an iterative message passing algorithm where the message send from node i to any of its adjacent nodes $j \in N(i)$ is

$$m_{ij}(x_j) = Z \sum_{x_i} \psi(x_i, x_j) \psi_(x_i) \prod_{k \in N(i) \setminus \{j\}} m_{ki}(x_i)$$

when computing marginal probabilities; for MAP estimation a maximization replcase the sum. The belief at every node upon termination of BP is then

$$bel_i(x_i) = Z \psi_i(x_i) \prod_{k \in N(i)} m_{ki}(x_i).$$

BP is not guaranteed to converge, but if it does so, then it converges to a local stationary point of the Bethe approximation to the free energy [17]. *Generalized BP (GBP)* builds on this fact and generalizes the energy function to be minimized upon convergence to the Kikuchi approximation to free energy [16, 17]. For every network to be applied for, GBP requires the specification of a so-called region graph, in which more powerful messages are passed between clusters of nodes. When bigger clusters are chosen in GBP, its messages become more powerful and its approximation improves, but unfortunately its complexity per iteration grows exponential in the cluster size. Since GBP usually needs to be adapted to the model at hand, we only report results for it on MRF2s. In this case, we choose the straightforward region graph consisting of quadruples of nodes (the graph's smallest loops) and their intersections.

Similar to BP and GBP, the *mean-field* (MF) algorithm iterates local updates of beliefs [13]. Its update equation

$$bel(x_i) = \phi(x_i, x_j) exp(\sum_{j \in N_i} \sum_{x_j} log \psi_{ij}(x_i, x_j)$$

for every node x_i is derived from minimizing the average Mean Field free energy.

Another widely used algorithm is *Gibbs sampling* (GS) whose popularity is mainly due to its generality. Since complete instantiations of a network usually cannot be sampled efficiently in the presence of evidence, this algorithm, starting by some random initiailization, iterates through the variables, sampling each variable x_i at a time, conditional on the current instantiation of all other network variables. Despite its usual generality, this algorithm does not apply to networks which have undergone the conversion from factor graphs to MRF2s. This is due to the deterministic potentials between all variables. After an initial hillclimbing phase, the algorithm would stay at the same spot of the search space forever, since, in that representation, each variable is constrained by its neighbours to keep its current value.

The cluster sampling algorithms *Swendsen-Wang* [11] and *Wolff* [14] would not get stuck in a state where changing only one variable at a time yields probability zero. However, unfortunately, they have been formulated without reference to local evidence which cannot be added in a straightforward fashion.[3] Multiplying the evidence into the pairwise potentials is always possible, but would probably hurt algorithm performance considerably since they are optimized for Ising models.[4]

For solving the MAP problem in MRF2s, algorithms based on *graph cuts* (GC) have recently been employed with great success [1, 2, 12, 10]. For binary variables and pairwise potentials $J_{ij}(1,1) + J_{ij}(2,2) \leq J_{ij}(1,2) + J_{ij}(2,1)$, a single graph cut already yields an exact solution in low polynomial time by reducing the problem to a maximum flow problem (for details, see e.g., [6]). Even for non-binary settings with comparable domain sizes, binary graph cuts can be applied as powerful local search steps in a greedy hill-climbing procedure, either in so-called *α-expansions* (which cast the problem as deciding for each pixel whether to keep its current instantiation or to adopt instantiation α), or in so-called $\alpha - \beta$ swaps (which solve the subproblem of assigning optimal labels in $\{\alpha, \beta\}$ to the pixels currently labelled either α or β.) [6].

Last but not least, recently introduced *Stochastic Local Search (SLS)* algorithms for solving the MAP problem in arbitrary graphical models show much promise [5].

4 System architecture

In order to create a unified interface to the various algorithms above, we developed conversion routines that reformulated the various inputs (Bayesian networks (BNs) and Markov random fields (MRFs)) to conform to a consistent structure. Our structure of choice for new algorithms to be developed on was factor graphs (FGs). All of the dependence properties in Bayesian networks and Markov networks can be represented as factor graphs [4]. Many of the algorithms we interfaced to are defined on pairwise MRFs, but we simply converted FGs to pairwise MRFs [17] to run these algorithms. With these converters in hand, we were then able to convert any input into a FG and apply any algorithm, regardless of the structure of the original input.

While the main scientific contribution of this project is the framework itself, we applied the framework to an experimental comparison of some algorithms on a number of real-world BNs and MRFs for computing image segmentation as well as marginal beliefs on randomly generated spin glass models (see Section 5). Figure 1 shows the architecture of our framework that will take as input a BN or an MRF. For each algorithm in the system, we created a Matlab class that inherited from a generalised interface. We also implemented a variable elimination engine to compute exact marginal probabilities. Based on what representation the algorithm requires, we apply the appropriate conversion routines before running inference. Our simple interface allows the user to run inference and query beliefs with four function calls: a constructor to instantiate the inference engine, a method to enter local evidence, a method that runs inference and a method that queries the beliefs. This simple design is extensible and modular and will easily facilitate the incorporation of new algorithms into the system.

Algorithm	Marg.	MAP	Potentials
Loopy BP	+	+	any
GBP	+	+	any
Mean-Field	+	-	any?
Gibbs	+	(+)	deterministic constraints are a problem
Swendsen-Wang	+	(+)	Spin glasses w/o local evidence
Wolff	+	(+)	”
Graph cuts	-	+	see caption
SLS	-	+	any

Table 1: Applicability of different algorithms. + means applicable, - means not applicable and (+) in the case of sampling algorithms for MAP means that MAP can naturally be approximated by simply using the best sample seen so far; this approach, however, usually performs much worse than specialized MAP algorithms with strong bias like SLS. The applicability of Mean-Field is not clear to us; theoretically, we would expect it to work properly on all representations, but in our experiments it failed badly on arbitrary networks that were encoded as MRF2s: it did not even converge on an encoding of a simple tree-structured factor graph with three nodes and two factors. The application of Graph cuts to the inference problem or a subproblem requires a problem formulation with binary domains and potentials of the form $J_{ij}(1,1) + J_{ij}(2,2) \leq J_{ij}(1,2) + J_{ij}(2,1)$.

5 Experiments

We conducted four experiments to both test our system and compare performance of the algorithms under different types of input networks. Where applicable, ground truth results were obtained by running variable elimination [3]. This limited the size of networks we could test against ground truth, however reasonably sized networks were indeed tested. We measured accuracy of algorithm k by computing the sum of squares error e_k over the marginal beliefs of the nodes in the network:

$$e_k = \sum_i (p_{k_i} - q_i)^2 \qquad (1)$$

[3] Email communication with Talya Meltzer.

[4] Personal communication with Firas Hamze.

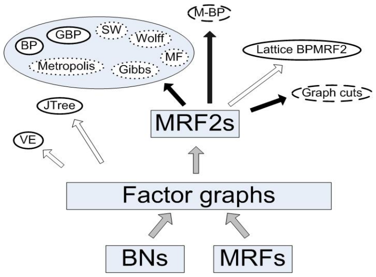


Figure 1: System architecture. Boxes are problem representations, ellipses are implementations of inference algorithms. Dotted ellipses are implementations that can only compute marginal probabilities, dashed ellipses are implementations that can only compute the most probable explanation (or the M best ones). Grey arrows are conversions, black arrows are calls to other algorithms, white arrows are calls to our algorithms and BNT code. Interfaces to Jtree, M-BP, and Graph cuts remain to be built.

where p_{k_i} are the beliefs of node i in the network found using algorithm k and q_i are the ground truth beliefs.[5] Unless explicitly stated otherwise, all runtimes reported are Wall clock runtimes on otherwise idle machines measured by Matlab' s built-in commands tic and toc.

5.1 Experiment 1 - Conversion effect

We measured the effect of the conversion routines from MRFs to FG to MRF2s to ensure they did not affect the inference results of the generally applicable algorithm BP. For this purpose, we generated MRF2s in the form of spin glasses conforming to the description in Section 2. The spin glass generation process had three parameters that could be set: psi, the variance of the zero-mean independently drawn random interaction terms J_{ij}; lsi, which governs the strength of local evidence h_i; and N, which gives the dimension of the spin glass. We created NxN size networks for N from 2 to 9 and ran inference using BP. Inference was run on both the uncoverted model and the converted model which first converted the MRF2 to a FG (using the fact that the MRF2 is just a special MRF) and then converted the FG to an MRF2 (employing the general procedure outlined in section 2). We compared the marginal probabilities of each node in the original graph under the two scenarios and in every single case found identical marginal probabilities. We also measured the running time to assess whether conversion was affecting execution time. Our results for this experiment show an increased running time for the converted networks by an approximate factor of 2.6 (see Figure 2); this is most likely due to the introduction of additional nodes in the graph in the conversion process.

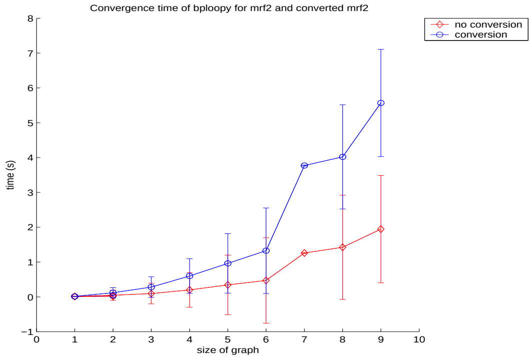


Figure 2: Wall clock time of BP for spin glasses with differing size, both for unconverted spin glasses (red) and spin glasses converted to FGs and then back to MRF2s. Plot shows mean running time over 10 runs for spin glass sizes N from 2 to 9 with $psi = 1.0$ and $lsi = 0.1$. Running time for converted networks was on average 2.6 longer over all runs.

5.2 Experiment 2 - Comparing algorithms on spin glass models

We compared the running time and accuracy of BP, GBP, GS and MF under the three different parameter regimes for generating spin glass models described above. First, we varied N from 3 to 9 while keeping psi and lsi fixed at 1.0 and 0.1 respectively. Second, we varied psi while keeping N and lsi fixed at 5 and 0.1. The range of psi is centered around the parameter reported in [16] hoping to reveal the levels of variance of the edge potentials at which point relative accuracy of algorithms diverge. Third, we varied lsi keeping N and lsi fixed at 5 and 0.1 respectively. Again, we looked to find divergence points of relative accuracy using this method. For all combinations of parameter settings, the inference algorithms were run 10 times and we report mean and standard deviations of the approximation error.

Figure 3, Figure 4 and Figure 5 show the accuracy and running time for the BP, GBP, GS and MF algorithms for varying N, psi, and lsi respectively. For larger problems, GBP performed best, whereas for smaller systems BP performed best. GBP showed the highest performance for problems with high variance in the potentials (and thus possibly tight interactions between variables), as well as for problems with weak local evidence (for which again interactions be-

[5] We are aware of the fact that KL divergence is often a preferable measure of error, but we experienced technical problems with some algorithms. These were due to those algorithms estimating small probabilities as zero, resulting in infinite KL divergence; this was especially often the case for the mean-field algorithm, a result that is interesting on its own.

tween variables dominate). With the other parameters being fixed, we were able to determine crossing over points of $N > 6$, $psi > 1.0$, and $lsi < 0.2$ where GBP was more accurate than BP (see Figures 3, 4, and 5). GBP always ran slower than BP under all parameter regimes although we caution the reader that GBP and BP did not always converge and running time was measured as time of execution of the algorithm.

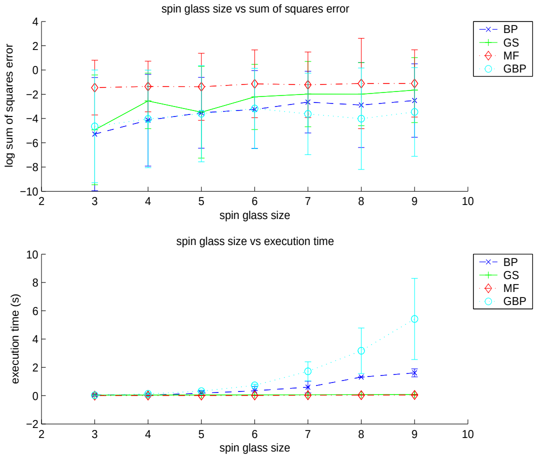


Figure 3: Error (top) and running time (bottom) of BP, GBP, GS and MF plotted against spin glass size N; estimates based on 10 runs for each value of N. BP was the most accurate for small networks (3x3), followed by GS, GBP and MF. For larger networks, (7x7 to 9x9), GBP was the most accurate, followed by BP, GS and MF. GBP however, was consistently the slowest in terms of running time, followed by BP, GS and MF.

5.3 Experiment 3 - Comparing BP and GBP for natural image processing

This experiment was done in collaboration with Tim Rees. The problem was to find the optimal segmentation of man-made and natural objects in images. For previous work on this problem, please refer to [7]. Using model parameters for local evidence and edge potentials of a MRF2 learned by feature-weighting (see report by Tim Rees), we compared the segmentation performance of MPE versions BP and GBP under an MRF model with the same edge potential over the entire image and the Discriminative Random Field (DRF) model [7] with different edge potentials for each edge. Accuracy was determined qualitatively by visually inspecting the output images which indicated man-made structures with white boxes.

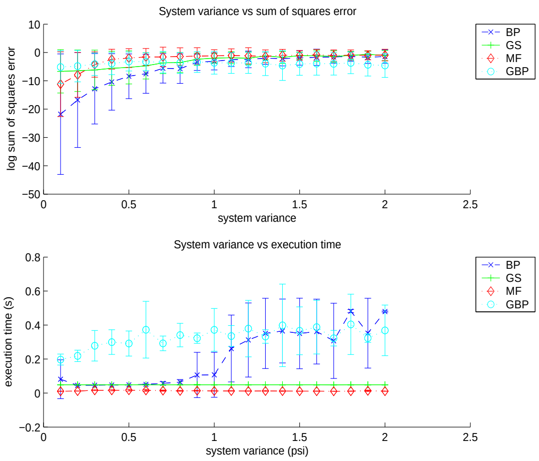
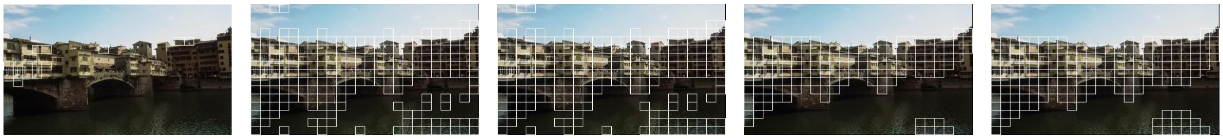


Figure 4: Error (top) and running time (bottom) of BP, GBP, GS and MF for varying variance psi in the interaction terms; estimates based on 10 runs for each value of psi. BP was the most accurate for lower variance, followed by GS, GBP and MF. For $psi > 1$, GBP performed better than BP, GS and MF. For $psi < 1$ running time followed the same trend as shown in Figure 3 with GBP slowest, followed by BP, GS and MF. Running time values for $psi > 1$ should be viewed with caution as the algorithms did not always converge to a solution in which and therefore the reported time is simply the time when the execution stopped. For further discussion on this point, please refer to the Discussion section.

5.4 Experiment 4 - Comparing MPE algorithms on Bayesian networks

For MAP solving in general networks, our framework currently only supports the BP algorithm, operating on encoded MRF2s.[6] Experiment 1 showed that the conversion into MRF2s does not affect BP's result, only increasing its runtime by a nearly constant factor; we concluded from this that we can use our available BP implementation to solve general graphical models, such as Bayesian networks. In Table 2, we compare this algorithm to published results for a number of Bayesian networks from the Bayesian network repository. The algorithms used for this comparison are the Branch-and-Bound algorithm BBMB which employs a Mini-Bucket heuristic with i-bound 10, as well as with the recent Stochastic Local Search algorithm GLS^+ (for detailed descriptions of these algorithms, the problem instances, and the results, see [5]). Identical ma-

[6] As mentioned in section 2, GBP requires the specification of a region graph which needs to be specified on an instance-by-instance basis; thus, we only report experiments with GBP for grid-structured MRF2s, where the region graph can be built easily.



(a) log class (b) mrf BP (c) mrf GBP (d) drf BP (e) drf GBP

Figure 6: Segmentation of image delineating man made structures with white boxes computed with a) log classifier (no inference), b) BP under MRF model c) GBP under MRF model, d) BP under DRF model, e) GBP under DRF model

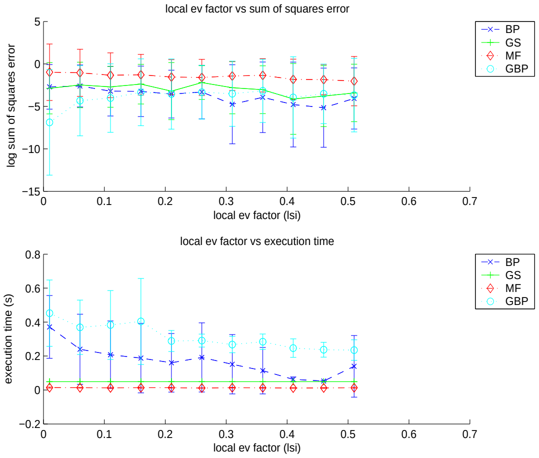


Figure 5: Error (top) and running time (bottom) of BP, GBP, GS and MF for varying strength lsi of the local evidence; estimates based on 10 runs for each value of lsi. GBP was the most accurate for lower lsi, followed by BP, GS and MF. For $lsi > 0.3$, BP performed better than GBP, GS and MF. Running time followed the same trend as shown in Figure 3 with GBP slowest, followed by BP, GS and MF.

Network	BP	BBMB	GLS^+
Alarm	9/0.05/0.006	0.00/+	0.00/+
Barley	-/880.14/0	36.76/+	19.22/+
Diabetes	-/1835.34/0	4.57/+	100/0.0099
Hailfinder	24/0.25/10^{-15}	0.00/+	0.00/+
Insurance	12/0.07/10^{-4}	0.00/+	0.00/+
Link	10/4.48/0	100/0	1.25/+
Mildew	-/834.2/0	1.25/+	0.26/+
Munin1	42/2.67/10^{-15}	30.14/+	0.34/+
Munin2	20/7.73/10^{-52}	3.98/+	0.96/+
Munin3	19/8.47/10^{-64}	4.55/+	0.87/+
Munin4	20/8.50/10^{-59}	31.72/+	100/0.035
Pigs	5/0.84/10^{-113}	0.08/+	0.14/+
Water	-/28.14/0	0.01/+	0.10/+

Table 2: Performance of BP, BBMB, and GLS^+ [5] on instances from the Bayesian network repository. For BP, we report number of iterations until convergence (- indicates no convergence within 2000 iterations), time until convergence, and approximation quality (found likelihood / optimal likelihood); and for BBMB and GLS^+, we report runtime and approximation quality (+ indicates optimal solution quality).

chines were used for the runtime analysis of all algorithms; for BBMB and GLS^+, CPU time is reported, and for BP wall clock time on an otherwise idle machine. From this experiment, we conclude that BP performs vastly inferior to current state-of-the-art MAP algorithms on real-world Bayesian network instances.

6 Discussion

6.1 Experimental Results

Experiment 1 showed that the conversion process had no effect on the marginal probabilities computed using BP. This was significant in that it allowed us to proceed with confidence that the conversion process would not affect any future results when comparing algorithms. The results obtained from Experiment 4, however, suggest that the conversion may indeed affect results. In retrospect, we realize that we should have also tested the effect of the conversion for the max-product version of BP instead for the sum-product version, for the max-product version should be subject to the same shortcomings as Gibbs sampling when handling deterministic potentials. The results for Experiment 2 suggest that GBP can handle highly correlated variables much better than BP, while the faster BP is superior for systems with comparably strong local evidence. Experiment 3 showed a progression of segmentations of images using a log classifier with no edge potentials, MPE of BP and GBP on an MRF model with consistent edge potentials and MPE of BP and GBP using a DRF model with variable edge potentials. It is clear from inspection that inference improved the results of the log classifier. Our qualitative results suggest that GBP had higher sensitivity when detecting regions with man-made objects (good coverage), but lower specificity than BP (more false positives). Future work on this project should apply Graph cuts and compare its performance with GLS$^+$. The particularly strong scaling behaviour of GLS$^+$ (see [5]) would be most useful in this setting, especially for larger images or for formulations of the problem that use regions smaller than 16x16 blocks inducing more nodes in the graph.

6.2 Experimental design

In some cases, error bars in the plots of Figures 3, 4, and 5 were highly overlapping, which leads to difficulty in interpreting relative performance. More runs would likely reduce the error and create more interpretable results. Due to long running times of VE to obtain ground truth, this was not feasible in the scope of this project. Future studies however should keep this in mind.

All our experiments suffered from a non-optimal timing mechanism. Sometimes algorithms did not converge and running time was simply taken as execution time of the algorithm with some default stopping criterion, such as a predefined maximal number of iterations. This could be significantly improved where, for example, the time for each iteration could first be calibrated in order to determine the numer of iterations an algorithm would take for a fixed amount of time. Then, all algorithms could be run with comparable runtimes, which results in a much more useful comparison. Also, in order to be of practical use, our framework needs to implement a better timing mechanism than Matlab's built-in measure of Wall clock time. Especially for large experiments, it can in general not be guaranteed that the used machine is otherwise idle, such that Wall clock measurements would often be uninformative.

6.3 Improvements to the system

There are many possible useful additions and improvements that can be made to our system. We would like to include GC (see Introduction) into the system. This would allow us to compare state of the art algorithms for stereo data to other algorithms presently in our system. We could also then use GC to run inference on the natural images presented in Experiment 3. As shown in Table 1, some of the algorithms are limited on the types of models on which they can run. Implementations for GS and MF that can run on FGs would add to the system and allow us to compare more algorithms on general networks. We would also like to include generalised versions of SW and WO that can operate on MRF2s with local evidence.

7 Conclusions

We developed a general framework to interface to various approximate inference algorithms. The framework allows users to input networks in various representations (namely BNs, MRFs, MRF2s, and FGs), and run their choice of inference algorithms. Although we did not include all algorithms originally outlined in our proposal, we nevertheless succeeded in creating a functional interface and general framework for approximate inference that could be easily extended in the future. Using our system, we were able to design experiments comparing performance of different combinations of algorithms under different input data scenarios. Although slower, GBP generally outperformed BP, MF, and GS for marginal beliefs on lattice-type input data. In a use case of the system a colleague was successfully able to compare results of max-product BP and GBP and choose the better of the two to yield promising results in identifying man-made structures in landscape-type images (see Figure 6 and report by Tim Rees). We were also able to add results of max-product BP to previously published results for standard BNs. In retrospect, BP's disastrous results in this domain may be due to the conversion to MRF2s which may affect the performance of max-product BP but not sum-product BP. Future experiments will investigate this further.

References

[1] Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. In *IEEE Computer Vision and Pattern Recognition Conference*, 1998.

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. 23(11), 2001.

[3] R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1-2):41–85, 1999.

[4] B. Frey and N. Jojic. A comparison of algorithms for inference and learning in probabilistic graphical models. 2004. To appear.

[5] F. Hutter. Stochastic local search for solving the most probable explanation problem in bayesian networks, MSc. thesis, Darmstadt University of Technology, Darmstadt, Germany, September 2004.

[6] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[7] S. Kumar and M. Hebert. Discriminative fields for modeling spatial dependencies in natural images.

[8] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, Computer Science Division, July 2002.

[9] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan-Kaufmann Series In Representation And Reasoning. Morgan Kaufmann Publishers, San Francisco, CA, USA, 1988.

[10] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 2002.

[11] R. H. Swendsen and J.-S. Wang. Non-universal critical dynamics in monte carlo simulations. *Physical Review Letters*, 1987.

[12] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV'03)*, volume 2, pages 900 – 906. IEEE Computer Society Press, 2003.

[13] Y. Weiss. *Comparing the mean field method and belief propagation for approximate inference in MRFs*. MIT press. To appear.

[14] U. Wolff. Collective monte carlo updating for spin systems. *Physical Review Letters*, 62:361, 1989.

[15] F. Y. Wu. The potts model. *Reviews of Modern Physics*, 1982.

[16] J. Yedidia, W. Freeman, and Y. Weiss. Characterization of belief propagation and its generalizations. Technical report, Mitsubishi Electrical Research Laboratories Inc., TR-2001-15, 2001.

[17] J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. Technical report, Mitsubishi Electrical Research Laboratories Inc., TR-2001-22, 2002.