Model Reductions for Inference: Generality of Pairwise, Binary, and Planar Factor Graphs

Frederik Eaton
frederik@ofb.net
Zoubin Ghahramani
zoubin@eng.cam.ac.uk
Department of Engineering, University of Cambridge, Cambridge CB2 1PZ, U.K.

We offer a solution to the problem of efficiently translating algorithms between different types of discrete statistical model. We investigate the expressive power of three classes of model—those with binary variables, with pairwise factors, and with planar topology—as well as their four intersections. We formalize a notion of "simple reduction" for the problem of inferring marginal probabilities and consider whether it is possible to "simply reduce" marginal inference from general discrete factor graphs to factor graphs in each of these seven subclasses. We characterize the reducibility of each class, showing in particular that the class of binary pairwise factor graphs is able to simply reduce only positive models. We also exhibit a continuous "spectral reduction" based on polynomial interpolation, which overcomes this limitation. Experiments assess the performance of standard approximate inference algorithms on the outputs of our reductions.

1 Introduction

Many researchers hold that intelligent systems, like humans, should be able to express and manipulate uncertain beliefs. Under this premise, the problem of formal reasoning becomes one of analyzing, or performing "inference" in, a statistical model. How should such a model be represented? We can try to inform the choice between alternative model structures by studying transformations, or "reductions," between them. In computer science, efficient reductions are a standard tool for measuring the difficulty and expressivity of reasoning frameworks. In this letter, we present a treatment of reductions that is oriented primarily toward the fields of machine learning and computational statistics.

We are interested in the problem of calculating the marginal probabilities of variables in a statistical model. Although this problem is perhaps the most basic and common form of statistical inference or probabilistic inference, for better clarity we shall refer to it here as marginal inference (MI). For the purpose of MI and other statistical inference problems, statistical models are often specified using a structure called a factor graph (see section 2.3), a simple yet flexible mechanism for defining probabilistic models. They generalize rich procedural representations such as causal networks and acyclic directed mixed graphs, as well as physical models such as the Ising model. We consider only factor graphs with variables having finite domain, also called "discrete," since our notion of reduction does not accommodate infinite models.

Sometimes results in marginal inference are formulated on restricted classes of discrete factor graphs (see section 2.2). These specialized factor graphs may be defined as subclasses of general factor graphs by constraining aspects of the model variables or connectivity. Model classes that can be defined in this way include models having only binary variables, or only pairwise factors, or whose graphical structure is topologically planar.

In this letter, we study the reduction properties of these classes of factor graph models with respect to the MI task. We say that MI on a particular class of discrete factor graphs can be reduced to MI on another class if a solution to the second problem can be easily used to solve the first, in a sense that is made precise in section 3.1. Although the feasibility of solving problems such as maximum a posteriori (MAP) and Boolean satisfiability (SAT) on general inputs by reduction to analogous members of these three classes is fairly well understood, apparently the corresponding results for marginal inference are not widely known.

We formalize a notion of reduction called simple reduction that is not only efficient but also able to express all reductions in common use in machine learning applications of inference. We show that all three of the above classes, and their four intersections, are able to "simply reduce" MI on general discrete factor graphs. Sometimes, however, this simple reduction is possible only in a circumscribed sense. More specifically, we show that binary pairwise factor graphs are not able to simply reduce some models containing states with zero probability. We also describe a more involved "spectral reduction," based on an idea from Valiant (1979b), which is continuous and polynomial time and is able to avoid this limitation. We hope that our results will help to clarify the relative usefulness of existing and future algorithms defined on these model classes.

The following section provides some necessary background. We open by reviewing the known facts on reductions in SAT, MAP, and MI in section 2.1. Then in section 2.2, we discuss some of the existing algorithms and decompositions for marginal inference, which have been defined on various subclasses of factor graphs. In section 2.3, we review the definition of factor graphs and marginal inference.

Our main theoretical contributions appear in section 3. This opens with our definition of MI simple reducibility in section 3.1, which is motivated by the traditional notion of polynomial-time reducibility and forms the basis for the rest of the section. The results are presented in sections 3.2 to 3.6 and summarized in section 3.7.

Finally, in section 4, we present numerical experiments that are intended to give a sense of the performance of existing marginal inference algorithms on models produced by our reductions.

2 Fundamentals

Readers who are unfamiliar with basic statistical inference terminology may wish to review section 2.3 before continuing.

2.1 Theory of Reductions. An important body of computer science research investigates equivalences between problems in various complexity classes. This is done by establishing, between pairs of problems, transformations, or reductions, which must be in some sense more efficient than the problems themselves. In this section, we give a broad overview of the theory of reductions, contrasting reductions in three computational frameworks. The first consists of NP decision problems like SAT, which we shall call the satisfaction framework. Next we consider the optimization framework, through the maximum a posteriori (MAP) problem that arises in computational statistics and has applications in computer vision. Finally we discuss existing concepts of reduction in the inference framework, whose canonical task we have referred to as marginal inference (MI).

2.1.1 Satisfaction Framework. We can define the satisfaction framework as consisting of problems in NP, the class of decision problems whose positive inputs have "correctness proofs" the size of which is bounded by a polynomial in the size of the input.[1] A standard definition of reducibility exists for problems in NP: we say that a problem A is polynomial-time Turing reducible (or just "reducible") to a problem B if, given an oracle (an idealized subroutine) that solves problem B in constant time, we can solve problem A in a time that is bounded by a polynomial in the size of A.[2]

A canonical problem in NP is the Boolean satisfiability problem, or SAT, which is the problem of determining whether a given Boolean formula can be satisfied by assigning the right values to each of the variables. Given a way of solving this decision problem, we can straightforwardly obtain a full satisfying assignment of any satisfiable formula. We do this by iteratively testing the satisfiability of a sequence of modified formulas, in which additional clauses have been conjoined to force each variable in turn to take one

[1]"Decision problem" refers to a problem formulation with a yes or no answer.

[2]Under the alternate polynomial-time many-one reducibility, the program must transform an instance of problem A into a single instance of problem B, again in polynomial time, and the result of solving the B instance must be correct for A. It is an open question whether the two definitions yield different notions of NP-completeness.

or the other of its values. So it is also natural to think of SAT as the problem of finding such an assignment.[3]

According to the Cook-Levin theorem, any problem in NP can be transformed, or reduced, to SAT (Cook, 1971). SAT and other problems that share this property are said to be NP-complete. There is a significant amount of research whose object is to identify and classify the many important NP-complete problems that arise in computer science.

The term *universal*, although not standard in this context, informally describes problems that are able to reduce more general classes in their frameworks, by analogy to the universality of Turing machines. In this sense, NP-complete problems like SAT are universal for the satisfaction framework. The term *universal* could also be applied to NP-hard problems—those problems that are able to reduce NP but are not themselves in NP (either because they are more difficult or because they are not phrased as decision problems).

A Boolean formula is constructed by applying the connectives $\wedge$ (*and*, or conjunction) and $\vee$ (*or*, or disjunction) as well as the unary operator $\neg$ (*not*, or negation) to an arbitrary set of binary-valued variables in any order. Boolean formulas can be transformed into a number of "normal forms," possibly by adding extra variables, which preserve the satisfying assignments of the original variables and thus can be used to solve the original problem. For example, any satisfying assignment of

$$a \vee b \vee c \vee d \tag{2.1}$$

can be extended to a satisfying assignment of

$$(a \vee b \vee x) \wedge (\neg x \vee c \vee d) \tag{2.2}$$

by choosing an appropriate value for x, and any satisfying assignment of the second formula also satisfies the first. This idea can be generalized to show that any Boolean satisfiability problem may be reduced in polynomial time to the problem of solving a formula in k-CNF (conjunctive normal form). These formulas look like a conjunction of disjunctive clauses,

$$\bigwedge_{c} \left(\Big(\bigvee_{i \in c^+} v_i \Big) \vee \Big(\bigvee_{i \in c^-} \neg v_i \Big) \right), \tag{2.3}$$

where c ranges over a set of two-part "clauses" $c \equiv c^+ \cup c^-$, each of size $|c| \leq k$. This is a standard result that holds when $k \geq 3$ (otherwise, we cannot

[3]Note that SAT is distinct from UNSAT, the problem of proving a formula unsatisfiable, which is co-NP-complete, so the two ("yes" or "no") outcomes of a decision problem are not symmetrical in the satisfaction framework.

fit enough auxiliary variables in our clauses). The problem of finding a satisfying assignment for a formula in k-CNF is called k-SAT. In other words SAT is reducible to k-SAT (making the latter NP-complete) for $k \geq 3$. On the other hand, 2-SAT is in P, a fact that we will use in section 3.4.

There is a straightforward analogy between k-CNF formulas, and factor graphs with binary variables and k-ary factors. By introducing a factor for each k-CNF clause and specifying that it take the value 1 when that clause is satisfied and 0 otherwise, we can get a distribution that partitions all of its probability equally among each of the formula's satisfying assignments. This transformation can be used to reduce SAT to the problem of marginal inference (MI), implying that MI is NP-hard (provided that some weak guarantees are made about the accuracy of the output marginals; Cooper, 1990). The counterpart of binary-pairwise factor graphs under this correspondence is formulas in the class 2-CNF, which are not able to reduce general SAT instances.

Relaxing the pairwise ($k = 2$) condition, we can alternatively consider Boolean formulas that are in some sense planar. Planarity in this context is defined as the property of being able to embed in a plane the bipartite graph relating clauses and variables. It turns out that SAT can be reduced to planar 3-SAT or, in other words, the latter is NP-complete (Lichtenstein, 1982).

Although SAT is defined in terms of binary variables, we can also imagine generalized n-ary constraint problems involving larger variable domains. These are instances of constraint satisfaction problems (CSP; Schaefer, 1978), which are reducible to SAT by introducing a Boolean indicator variable for each variable-value pair in the input. In CSP with arbitrary constraints, the corresponding pairwise and planar pairwise classes become universal (by analogy to theorems 4 and 16). Thus we can say that in the satisfaction framework, only the binary pairwise class and its subclasses are not fully general.

2.1.2 Optimization Framework. This framework consists of problems where variables must be adjusted not to satisfy a set of constraints as in SAT but to maximize or minimize an objective function. The maximum a posteriori problem (MAP) is a standard optimization problem with roots in statistics, whose goal can be stated as finding the state with maximum probability in a statistical model (which can be defined by a factor graph). MAP with binary variables is equivalent to weighted k-SAT, in which weights are assigned to each clause and the goal is to find a variable assignment where the sum of weights of satisfied clauses is maximized. Thus, it is NP-hard.

For our purposes, we can use a notion of reduction for optimization problems, based on oracles, that is a straightforward adaptation of reduction from the satisfaction framework. It is known that MAP may be reduced to the maximum weight independent set problem (MWIS) (Sanghavi, Shah, & Willsky, 2009), in which weights are assigned to the vertices of a graph and the goal is to find a set of vertices of maximum weight subject to the constraint that no two of these vertices are connected by an edge. MWIS in turn can be easily reduced to MAP on a binary-pairwise graph. Thus, binary pairwise graphs are universal in MAP. This is also known as the statement that optimization of pseudo-Boolean functions can be reduced to optimization of quadratic pseudo-Boolean functions (Rosenberg, 1975; Boros & Hammer, 2002). The outputs of some of these reductions are apparently difficult to optimize using standard algorithms, and additional work examines the problem of finding reductions to objective functions with properties such as submodularity (Ishikawa, 2009) that facilitate optimization.

Barahona (1982) shows that MAP on planar binary pairwise graphs is NP-hard by reducing to it the NP-complete planar maximum independent set problem (Garey & Johnson, 1979).[4] This does not in itself imply the existence of a reduction to binary pairwise planar MAP within the optimization framework, but it is easy to construct one, as we outline after the proof of theorem 22.

It is possible to use inference algorithms to solve MAP problems, just as with SAT problems. However, we are only aware of a "limiting" reduction in this case. It works by introducing a temperature variable T and creating a distribution that puts all of its mass on the most likely state in the limit $T \to 0$. In other words, if $f(x)$ is to be maximized, we can make guesses about the optimal value of x by calculating the marginals of the "Boltzmann distribution" (or "Gibbs measure") $P(x) \equiv \frac{1}{Z(T)} \exp \frac{1}{T} f(x)$, where $Z(T)$ is a normalizing constant, for smaller and smaller values of T.

A SAT instance can likewise be solved using a MAP solver if it is first expressed as an optimization problem, in which (for instance) the goal may be to find a variable assignment maximizing the number of satisfied clauses, as in weighted k-SAT.

2.1.3 Inference Framework. The computational problem of inferring marginal probabilities or, equivalently, calculating the partition function of a statistical model (see section 2.3) has its origin in two distinct fields of research. In the first, statistical physics, the inference problem arises in the study of the equilibrium states of physical systems, such as the Ising spin glass model of a ferromagnetic solid. The models arising in physical application domains are typically characterized by a repetitive, symmetrical structure. Although such symmetry is absent from many of the models arising in machine learning applications, most of the inference algorithms that are applied to machine learning today have their origin in statistical physics. This includes Gibbs and other sampling methods, as well as

[4]Here the maximum independent set optimization problem is considered as a decision problem, where the "decision" is whether the objective function can be bounded by a given value.

variational approximations such as mean field and the Bethe-Peierls approximation, which is related to belief propagation.

The second field is theoretical computer science, which is largely concerned with establishing equivalences between classes of computational problems using tractable reductions. In the same way that reductions within the class NP are used to relate problems in the satisfiability framework, theoretical computer science approaches the inference framework through counting problems that are represented by the class #P (pronounced "sharp-P"). The class #P is defined as the set of functions whose integer-valued outputs can be expressed as the number of accepting branches of a nondeterministic Turing machine running in polynomial time. This class was defined in its present form by Valiant (1979a), who showed that the problem of computing the permanent of an integer matrix is #P-complete by reducing it to #SAT.[5] Using reductions to PERMANENT he also proved the #P-completeness of a number of other problems, including #2-SAT (Valiant, 1979b). (This is considered a surprising result given that the corresponding decision problem 2-SAT is in P.)

In a parsimonious reduction (Papadimitriou, 1994), only a single oracle call is needed, and the output of the reduction is just the output of the oracle. This is the #P analog of polynomial-time many-one reductions. Unlike the situation for NP-complete, it is not the case that parsimonious reductions might yield an equivalent definition of #P-complete. Some #P reductions are parsimonious, but many are not. For example, the reduction of #SAT to #2-SAT is not parsimonious.

The #SAT class and its subclasses #*k*-SAT, which count the number of satisfying assignments of Boolean formulas in conjunctive normal form, are equivalent to computing the partition function of binary factor graphs with all potential entries equal to 0 or 1. Factor graphs with larger integer entries can be easily represented as counting the number of solutions of a constraint problem with, for example, extra variables whose range of allowed values is constructed to have a size equal to each of the desired potential entries. Factor graphs with rational entries can be modeled by dividing an integer-valued factor graph by a polynomial-time computable denominator, $Z = \tilde{Z}/K$. The class of such rational-valued computational problems is named $\#P_{\mathbb{Q}}$ by Goldberg and Jerrum (2008). The focus of recent work in this area is on the problem of classifying the complexity of #CSP classes, which correspond to the partition function problem for factor graphs whose potentials are not freely chosen but come from a predefined constraint language characterizing the problem class. Restricting potentials

[5]The class #P-complete is defined to include problems $y \in$#P such that any problem in #P can be solved in polynomial time using an integer-output TM with an oracle for y. Or, equivalently, #P$\subseteq$FP^y (FP being the class of functions computed by a TM in polynomial time).

to a constraint language usually results in “dichotomy theorems” as first obtained by Schaefer, 1978, see Dyer, Goldberg, & Jerrum, 2009; Bulatov & Grohe, 2005). The theoretical computer science approach to inference also includes more complex results such as the “holographic reduction” of Valiant (2008), which, for example, allows us to prove that the problem of counting the number of solutions of monotone planar 3CNF formula in which each variable occurs twice is easy if the answer need only be known modulo 7, although the same problem is hard if it must be calculated modulo 2 (Valiant, 2006).

These results make use of chains of reductions, often obtained through a creative use of integer arithmetic. Many of them are not obviously relevant to machine learning applications of inference. As an example, a number of #CSP results must employ reductions that implement variable conditioning in #CSP problems whose constraint language might lack delta functions. Such a reduction appears in Dyer et al. (2009): when generalized to real numbers, it is not continuous, and it multiplies the complexity of inference by at least the factorial of the size of the variable domain. It is hard to see how this could be applicable to practical machine learning algorithms, for which variable conditioning is generally implemented as a trivial operation.

In spite of these differences of approach, we were able to fruitfully adapt what has become a standard technique from theoretical computer science, based on polynomial interpolation, which first appeared in Valiant (1979b). We use it to construct a reduction that is both continuous and not overly expensive (see section 3.6), for a case where our more intuitive simple reduction fails to apply.

We do not dismiss the possibility that the many other ideas from theoretical computer science research on counting problems may one day have important applications to statistical inference problems in machine learning or statistical physics. The focus of this letter, however, is on reductions with straightforward, potentially immediate applications to these domains. We consider the task of marginal inference (MI), which we define as the problem of computing probabilities in a statistical model. This is roughly interchangeable with computing the partition function, but slightly more natural. We are interested in reductions that are continuous and, perhaps, statistically meaningful. Although we demand polynomial time complexity, we are actually interested in other measures of efficiency as well. We formalize a notion of reduction that has appeared in the machine learning and statistical physics literature in various specialized incarnations. These include the pairwise reduction and variable clustering reductions, such as used in the cluster variational method and the junction tree algorithm. The details of our reduction concept, which we call “simple reduction,” are presented in section 3.

2.1.4 Discussion. Let us contrast the three problems, SAT, MAP, and MI. One point of difference is in the role of auxiliary variables in reductions.

In MI, an auxiliary variable is introduced in such a way that the desired distribution is obtained from marginalizing out the new variable. Thus, the values of the distribution at each setting of the auxiliary variable must add up to exactly the correct quantity. In MAP, only one value of an auxiliary variable typically plays a role in the solution state, although the reduction must presumably be constructed so that the variable can switch values as appropriate to preserve the original optimization landscape. For example, given binary variables taking values in {0, 1}, we can use an auxiliary variable to turn a degree 3 term into four terms of degree 1 or 2:

$$\max_{y}(f(y) + y_1y_2y_3) = \max_{y,z}(f(y) - 2z + y_1z + y_2z + y_3z). \tag{2.4}$$

In the new maximization problem, the auxiliary variable z is usually forced to take a single value, although if exactly two of the y_i's are 1, then z can take either value. In SAT, similar situations occur: many satisfying assignments in the input formula appear in the transformed formula with auxiliary variables forced to take only one value, but situations where both values are allowed may also occur. Consider the previous example, equation 2.2: assignments in which both $a \vee b$ and $c \vee d$ are true are "duplicated" in the new model, with one copy for each possible value of x.

Another way of viewing the distinction among the satisfaction, optimization, and inference frameworks is in terms of solution verification. For an SAT instance, it is simple to check that an assignment is indeed satisfying, and given such an assignment, we can immediately conclude that a formula is satisfiable (although proving that a formula is not satisfiable can be more difficult). For MAP, given a state of the model, there is no easy way in general to tell whether that state is an optimum. However, given two states, we can say which one is "better" by calculating and comparing the objective function, the unnormalized joint probability, at each state. Thus, in MAP, there is an easily obtained total ordering of possible solutions in which the "true" solution is maximal. For MI, given two sets of univariate marginals, there is apparently no easy way to tell which one is better. But if we are given two approximate MI algorithms, each of which can be queried for conditioned marginals, then we may create a score as described in Eaton (2011) that provides an indicator of the better approximation. Although such a score is deterministic and is guaranteed to favor exact inference, the ordering it induces on approximations may contain cycles. This complication is absent from the simpler MAP setting. Thus, we see that solution verification becomes progressively more difficult in the satisfaction, optimization, and inference frameworks, respectively.

We have described how each of the three computational frameworks is able, to a certain extent, to express problems in the preceding frameworks. We might imagine that successively more powerful algorithms can be built on each of the frameworks in turn. This hypothesis has some rough empirical support. For instance, state-of-the-art algorithms for satisfaction are based on techniques from optimization (such as GSAT or WalkSAT; see Selman, Levesque, & Mitchell, 1992) and, more recently, from inference (such as survey propagation; see Braunstein, Mezard, & Zecchina, 2005).

2.2 Marginal Inference on Specialized Graphs. When defining new marginal inference algorithms, it is sometimes useful or necessary to impose restrictions such as "binary" or "pairwise" or "planar" on aspects of the input models. Recall that our goal is to simplify the choice and interpretation of such restrictions by characterizing the ways in which general models may be transformed into each of these classes. Here we give a review of some of the more prominent occurrences of restricted model classes in the literature on marginal inference.

The first published example of what is now called belief propagation (BP), presented by Gallager (1962), was defined on binary factor graphs (i.e., graphs where all variables are binary) with parity-check factors. The BP formulation of Pearl (1982) originally used tree-structured causal (Bayesian) networks, and later loopy causal networks (Pearl, 1988). However, BP is easily generalized to arbitrary factor graphs (Kschischang, Frey, & Loeliger, 2001).

The BP algorithm is sometimes specified on pairwise factor graphs, for pedagogical reasons (Yedidia, Freeman, & Weiss, 2001b) or for suitability to a parent algorithm (Wainwright, Jaakkola, & Willsky, 2002). It is straightforward to reduce general factor graphs to pairwise form (see theorem 4) and BP is actually invariant under such reductions.

Algorithms and results for binary graphs often assume pairwise connectivity as well. An exception is the loop decomposition of Chertkov and Chernyak (2006), which is defined on binary n-wise factor graphs.[6] This decomposition has been used to lower-bound the partition function of a binary pairwise factor graph (BPFG) with "attractive" (i.e., ferromagnetic) potentials (Sudderth, Wainwright, & Willsky, 2008).[7] Related theoretical results are often defined on BPFGs (Watanabe & Fukumizu, 2011). The algorithm of Montanari and Rizzo (2005) was defined on BPFGs but is easily generalized (Mooij, Wemmenhove, Kappen, & Rizzo, 2007).

MI algorithms specific to BPFGs include belief optimization (Welling & Teh, 2001) and the self-avoiding-walk (SAW) tree expansion of Weitz (2006, applied to inference by Jung & Shah, 2006). The SAW-tree expansion has been the subject of interest. Although the expansion has been applied to

[6]That is, containing factors of arbitrary size.

[7]Sudderth, Wainwright, and Willsky define a binary pairwise model to be attractive if for every edge potential, the relation $\psi_{ij}(0,0)\psi_{ij}(1,1) \geq \psi_{ij}(0,1)\psi_{ij}(1,0)$ holds. This is equivalent to ferromagnetic interactions $J_{ij} \geq 0$ in the traditional Ising model. In the optimization framework, such models are referred to as having a submodular energy function.

graphs with n-ary variables or n-wise factors (Ihler, 2007; Nair & Tetali, 2007), no one has been able to generalize the original construction to provide exact marginals in non-BPFGs while preserving the tree structure. Producing such a generalization or proving its impossibility is an open problem.

As for planar BPFGs, we know of two important results for MI on this class. The first is Globerson and Jaakkola's (2007) algorithm for upper-bounding Z and calculating marginals, based on the Fisher-Kasteleyn-Temperley (FKT) algorithm of statistical physics (Fisher, 1966; Kasteleyn, 1963). The second result (Chertkov, Chernyak, & Teodorescu, 2008) shows how to perform approximate inference by summing a truncated loop series on planar graphs, using the related Pfaffian formula of Kasteleyn (1961). The papers of Fisher and Kastelyn treat models satisfying the additional constraint of pure interactions. Such models assign equal probability to a state and its complement. This property is equivalent to containing only "soft-XOR" (equation 3.14) pairwise factors and no unary factors. The class of models with this property is quite restrictive and is not even closed under variable conditioning. Barahona (1982) also showed that inference is tractable in this special case, traditionally called "spin glasses." Here we should also mention Valiant's recent work on holographic reductions (Valiant, 2008), which although oriented toward integer-valued counting problems, has a special focus on planar graphs and makes multiple uses of the FKT result.

2.3 Definitions. We define a statistical model to be a probability distribution over some set of (discrete) random variables: $x \in \prod_{i\in\mathcal{V}} \mathcal{X}_i$, where $\mathcal{V}$ is a set of variable indices and the $\mathcal{X}_i$ are finite sets. This distribution should also be associated with a structure encoding it in one of various possible ways. Such a structure is often given as a factor graph, and we will assume this representation in all of the material that follows. A *factor graph* is a collection $\mathcal{F}$ of *factors*, each associated with a set α of variables and a function (its potential or local function) from the domains of such variables to the nonnegative real numbers,

$$\psi_\alpha : \mathcal{X}_\alpha \to \mathbb{R}_+, \tag{2.5}$$

where $\mathcal{X}_\alpha \equiv \prod_{i\in\alpha} \mathcal{X}_i$. These functions are multiplied together and normalized to induce a distribution over the variables:

$$P(x) = \frac{1}{Z} \prod_{\alpha\in\mathcal{F}} \psi_\alpha(x_\alpha). \tag{2.6}$$

The normalization constant Z is also known as the partition function. Factor potentials may also just be called factors.[8] We refer to the class of general discrete factor graphs as "DFGs."

The structure of a factor graph is often illustrated by a diagram with a circular vertex for every variable and a square vertex for every factor and edges connecting variables with the factors that contain them (see, e.g., equation 3.7). Binary factors may simply be represented as edges (e.g., equation 3.15).

The problem of *marginal inference* (MI) (also called probabilistic inference or Bayesian statistical inference) is to calculate marginals,

$$P(x_i) \equiv \sum_{x_{\setminus i}} P(x). \tag{2.7}$$

Here $x_{\setminus i}$ represents the set of all x variables excluding x_i, that is, $x_{\mathcal{V}\setminus i}$. When such calculation is only approximate, then we sometimes say that "approximate marginal inference" (or when there is no room for confusion, "approximate inference") is being done. When the calculation is exact (to machine precision), the problem is often called exact inference. In this case, or when the resulting approximation is required to be accurate to within some bound, then for general factor graphs, MI is known to be NP-hard (Cooper, 1990; Barahona, 1982). In this letter, we are not too concerned with the accuracy guarantees, if any, of MI algorithms, which might be applied to a reduction or transformation of a factor graph, since in each case the reduction itself is either exact or can be made arbitrarily precise. We refer to both exact and approximate forms of MI as simply "MI."

By introducing a factor that is a delta function, we can constrain a variable to take a given value. The resulting distribution is equal to a conditioned version of the original distribution:

$$P(x|x_i = x_i^*) = \frac{1}{Z'}\delta(x_i, x_i^*)\prod_{\alpha}\psi_\alpha(x_\alpha). \tag{2.8}$$

MI in the conditioned model gives conditioned marginals, and these can be combined with unconditioned marginals to compute the probability of arbitrarily many variables:

$$P(x_1, x_2, x_3) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2). \tag{2.9}$$

[8]Many authors prefer the terminology of an older and less flexible, but essentially identical, representation, called a Markov random field (MRF). In an MRF, the product in equation 2.6 multiplies potential functions whose domains correspond to cliques of a graph, whereas our function domains are arbitrary sets of variables.

In fact, many inference algorithms, for example, BP, produce estimates of the partition function, allowing such "multivariable marginal" probabilities to be computed in one step. If r is a set of variables, then we have[9]

$$P(x_r) = \frac{Z'}{Z} \equiv \frac{\sum_{x'} \prod_{i \in r} \delta(x'_i, x_i) \prod_\alpha \psi_\alpha(x'_\alpha)}{\sum_x \prod_\alpha \psi_\alpha(x_\alpha)}. \tag{2.10}$$

We shall call an object such as x_r, representing an assignment of values to one or more variables, a *partial assignment* (PA) and consider the problem of "weighing," or calculating the probability mass of, multivariable PAs as equivalent to MI. When more clarity is needed, we shall write the variables and the assigned values of the PA separately, for example, $x_r = x_r^*$, as in $P(x_r = x_r^*)$. Where we have a superset $r' \supseteq r$ and $(x_{r'}^*)_r = x_r^*$, then we say the PA $\{x_{r'} = x_{r'}^*\}$ is an extension of the PA $\{x_r = x_r^*\}$. Since PAs are a special kind of event in σ-algebra, we also use terminology from sets, and speak accordingly of the "union" or "intersection" of PAs. One may check that PAs are closed under intersection but not union.

Note that we can easily apply optimization problems such as MAP to a conditioned model using the constraint technique of equation 2.8. But applying MAP to a model in which some variables have been summed over or "marginalized out" is not straightforward. Given an algorithm that computes the most probable assignment of all the variables in the model, there is no general way to adapt it to compute the most probable assignment of a subset of the variables when summing over the others.

The terms *marginal inference* and *partial assignment* are not common. The rest of our terminology is fairly standard. Factor graphs were defined in Kschischang et al. (2001). Potential functions (for an MRF) are called ψ in Castillo, Gutiérrez, and Hadi (1997). Wiegerinck (2000) first indexed clusters of variables with Greek letters α, β, γ.

3 Theory

3.1 Definition of Reduction.

It is customary in computer science to define equivalence classes of problems using polynomial-time reductions. These are programs whose running time is bounded by a polynomial function in the size of the input, which solve one class of problems using calls to a constant-time oracle solving another class of problems. Polynomials are

[9]We can also go in the opposite direction and compute the partition function from conditioned marginals. This can be done by making use of the unnormalized joint at an arbitrary state x, which is easily evaluated: $ZP(x) = \prod_\alpha \psi_\alpha(x_\alpha)$. Then

$$Z = \frac{ZP(x)}{P(x)} = \frac{\prod_\alpha \psi_\alpha(x_\alpha)}{\prod_i P(x_i | x_{1\ldots i-1})}.$$

used because they are closed under multiplication, addition, and composition, and any of a number of natural models of computation such as the Turing machine (TM) can simulate each other with only a polynomial-time overhead (Bernstein & Vazirani, 1997). Thus it is not necessary to be too specific about the machine on which a program is said to run in polynomial time. We discussed polynomial-time reductions for inference in section 2.1.3, where we defined $\#\mathrm{P}_{\mathbb{Q}}$ and #P-complete.

From the standpoint of machine learning applications, it makes sense to demand that reductions should preserve polynomial time complexity. However, polynomial-time programs could still be very slow, and researchers in applied fields may understand the idea of transforming one inference problem into another to imply something more stringent than is understood by theoreticians. Although inference is NP-hard, or rather #P-hard, or $\#\mathrm{P}_{\mathbb{Q}}$-complete, the difference between a linear-time and quadratic-time reduction may be important because inference algorithms often depend for their efficiency on specific inputs having a special structure, such as low connection strength (as explored by Mooij & Kappen, 2005a, in the case of BP) or low connection density (a measure used to characterize random k-SAT problems; Braunstein et al., 2005), so that their running time on inputs of interest is subexponential. In other words, the size of the input problem alone is not a good measure of difficulty. Furthermore, the question of simulating one kind of hardware using another has diminishing relevance in practice, particularly in the case of minimalist hardware models like the TM, where working memory is not even random access. Sometimes hardware for inference is imagined to be highly parallel, as by analogy to biological systems, and even model dependent. In any case, the hardware invariance of polynomial-time complexity classes is not as relevant in applied fields as it is in theoretical ones.[10]

We propose a reduction concept called "simple reduction" which is related to the "parsimonious reduction" from complexity theory. In simple reductions, one model is first transformed into another, and then any number of "queries" can be made on the transformed model. Each query poses the question, "What is the probability (in the original model) of the partial assignment (PA) x_r?" This is answered by transforming x_r into a single PA in the output model, say, y_s, such that $P(x_r) = Q(y_s)$. Our reduction concept has perhaps been assumed implicitly by previous authors and can accommodate existing reductions such as the pairwise reduction given in Yedidia

[10]The complexity class NC has been used to describe parallel computations and consists of problems that can be solved in polylogarithmic time on a computer with a polynomial number of processors. It is possible that NC could be used to provide a (polylogarithmic time) reduction, which is more appropriate to statistical inference, but we do not attempt this here. Additional complexity classes for randomized and approximate polynomial-time computations have also been formally studied, and reductions based on these classes are relevant to marginal inference but have the same drawbacks as other polynomial-time reduction concepts. See Arora and Barak (2009).

et al. (2001b), the d-regular to three-regular reduction of Fisher (1966) and, with some modification, variable clustering transformations such as junction tree (Jensen, Olesen, & Andersen, 1990).

Although we stipulate, for the sake of tradition, that the model conversion phase of the reduction should be polynomial time in the size of the model, in the reductions presented here, the relation between input and output model size, as well as the time complexity of the conversion, is at most quadratic in the case of planar outputs, and linear otherwise.[11] Even with such a minimal notion of reduction, we are able to reduce positive DFGs to planar binary pairwise factor graphs.

In terms of existing counting reductions for #P or #$\text{P}_{\mathbb{Q}}$, our simple reduction can be seen as separating the input to a counting problem into two parts, comprising the model P and the query x_r. The weighted counting reduction concept is based on algorithms that calculate just the partition function of a model. Such algorithms can then be called a second time after conditioning some model variables to obtain probabilities. In Figures 1a, 1b, and 1c we contrast these notions of reduction.

In Figure 1d we depict the "spectral reduction," which is presented in section 3.6. It requires inference to be done in multiple output models, combining the results in a nontrivial way, and so it is not a simple reduction. However, it is polynomial time, in fact linear time, and preserves the possibly desirable property of continuity in the relation between input and output model parameters. Using this reduction, it is possible to remedy a shortcoming of the simple reduction, which is its inability to reduce models with zeros to BPFGs.

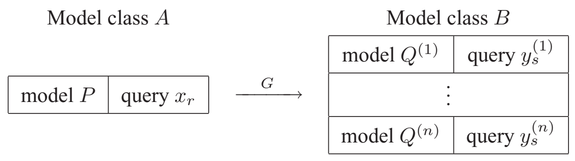
We now present the formal definition of simple reduction. We divide the reduction into two phases. First, the model is transformed ("model conversion"). The model needs to come from an infinite class of models so that we can talk about demanding a polynomial relationship between input and output model sizes. Next, one or more queries can be answered by the transformed model ("query conversion"). Only a fixed model is needed into define the query-conversion step.

Definition 1.

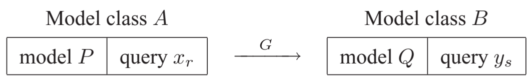
A. Query conversion. *We say that marginal inference (MI) in a model $P(x)$ is "simply reducible" to MI in a model $Q(y)$ if there exists a function F from PAs in P to PAs in Q such that for any PA x_r in P, letting $y_s = F(x_r)$ we have*

1. *Conservation of probability mass: $P(x_r) = Q(y_s)$*
2. *Preservation of containment: Given $r' \supset r$ and $x_{r'}$ such that $(x_{r'})_r = x_r$, we have $F(x_{r'}) = y_{s'}$ where $s' \supset s$ and $(y_{s'})_s = y_s$*

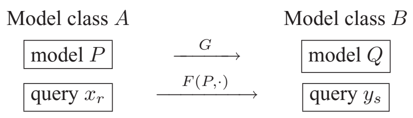
[11]For complexity bounds, we assume input models with bounded variable arity and bounded factor size.



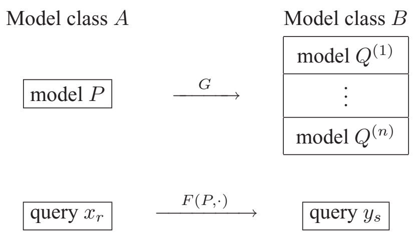
(a) A generic polynomial-time reduction. Multiple oracle calls can be made, and outputs are combined arbitrarily. "Queries" are not intrinsic to this reduction concept, but may represent conditioned variables in a partition function calculation.



(b) A parsimonious reduction. As in (a), but the output of the reduction is the unmodified output of the single oracle call.



(c) Our two-phase "simple reduction". As in (b), but Q depends only on P, and the function $F(P,\cdot)$ which relates the queries is a valid "PA-map".



(d) A special case of (a), the "spectral reduction" (section 3.6). Multiple models must be evaluated, as in (a), but queries are still related by a "PA-map" as in (c).

Figure 1: Possible notions of reduction in the inference framework.

3. *Efficiency: We should be able to compute $F(x_r)$ in time bounded by a polynomial function of the size of r*

We call F the "PA-map" for the reduction.

B. Model conversion. *We say that MI in a model class A is "simply reducible" to MI in a model class B if there exists a function G mapping models $P \in A$ to models $Q \in B$ and a function $F(P, x_r)$ mapping models P to PA-maps $F(P, \cdot)$*

such that for any $P \in A$:

1. *$F(P, \cdot)$ satisfies the above requirements for a PA-map and can be computed in time bounded by a polynomial function of the size of P. In other words, P is simply reducible to $G(P)$, and the PA-map for the reduction can be efficiently computed.*
2. *The time required to compute $G(P)$ is bounded by a polynomial function of the size of P*

Here we define the "size" of a model P as the number of entries in the potential functions for the factor graph which specifies P.

Part B of definition 1 is a straightforward extension of the model reduction concept to model classes, preserving the polynomial time constraint. Below we discuss some consequences of part A, which describes reductions between models using query conversions defined by valid PA-maps.

Note that the definition of simple reduction between two models P and Q encompasses the common situation where the variables of P are "included" in Q but Q also has extra "latent" variables that need to be "marginalized out." In this scenario, we can see F as a kind of embedding. All of our reductions can be viewed in this way, except for those that convert from n-ary variables to binary variables.

Condition A2 that a PA-map F should preserve containment implies that we can write the value of F at a multivariable PA in terms of its values at single variables: $F(x_r) = \bigcap_{i \in r} F(x_i)$. Thus, it is enough to define F on all (variable, value) pairs.

Note that the PA-map F cannot be multivalued. Reductions are forbidden in which a probability $P(x_r)$ is calculated from the union of multiple (disjoint) PAs in Q, that is, $P(x_r) = Q(\bigcup_i y_{s_i}) = \sum_i Q(y_{s_i})$. The reason for this is to preserve the polynomial time complexity of the reduction. Due to the containment-preserving property of F, if we apply F to the intersection of n PAs in P, each of which maps to a union of two PAs in Q, then the result will be a union of 2^n PAs in Q. But computing this mass will take exponential time in n.

Generic transformations from DFGs to tree-structured models, like junction tree, cannot be simple reductions because they are exponential in the model conversion phase. However, the variable clustering idea used in these reductions may be represented as a simple reduction if additional variables are introduced in the output model corresponding to the variables of the input model. These should be attached to the "clustered" variables, with 0-1 factors constraining the latter to take the appropriate range of values. This lets us avoid the need for a multivalued PA-map.

It may happen that the PA-map F is not invertible: it is not the case, under our definition, that P is reducible to $Q \Leftrightarrow Q$ is reducible to P. For an example, suppose $P(x_i) = Q(F(x_i))$ where F encodes a four-valued x_1 in binary:

$$F(x_1 = 0) = \{y_{(1,2)} = (0, 0)\}, \tag{3.1}$$

$$F(x_1 = 1) = \{y_{(1,2)} = (0, 1)\}, \tag{3.2}$$

$$F(x_1 = 2) = \{y_{(1,2)} = (1, 0)\}, \tag{3.3}$$

$$F(x_1 = 3) = \{y_{(1,2)} = (1, 1)\}. \tag{3.4}$$

Then $Q(y_2 = 0) = P(x_1 = 0) + P(x_1 = 2)$, which is a relationship we cannot express through a simple reduction since our PA-maps must, as above, be single-valued.

It is easy to see that our notion of reduction is transitive: if P simple-reduces to Q with PA-map F and Q to R with PA-map G, then P simple-reduces to R with PA-map $G \circ F$.

We finish this section with two definitions that we use in the rest of the letter:

Definition 2. *A class of factor graphs is "universal" if the class of general discrete factor graphs (DFGs) can be simply reduced to it.*

Definition 3. *A class of factor graphs is "positive universal" if the class of DFGs with strictly positive parameters (positive DFGs) can be simply reduced to it.*

3.2 Pairwise Factor Graphs.

A common restriction imposed on factor graphs is to require all factors to have size 1 or 2. (Note that size 1, or singleton, factors can be seen as degenerate factors of size 2.) Such graphs are called "pairwise" factor graphs. It is easy to show that arbitrary (n-wise) factor graphs can be reduced into pairwise form. A version of the following theorem was outlined in Yedidia et al. (2001b):

Theorem 4. *Pairwise factor graphs are universal.*

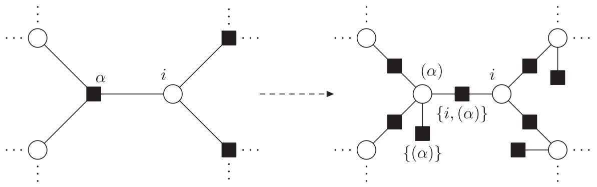
Proof. One way to effect the reduction is to create a variable—i or (α)—for each variable i and factor α in the old graph, introduce singleton factors $\{(\alpha)\}$ for each α and pairwise factors $\{i, (\alpha)\}$ for each $i \in \alpha$, and assign to these factors the following potentials:

$$\hat{\psi}_{\{i,(\alpha)\}}(\hat{x}_i, \hat{x}_{(\alpha)}) = \begin{cases} 1 & \text{if } \hat{x}_i = [\hat{x}_{(\alpha)}]_i, \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

$$\hat{\psi}_{\{(\alpha)\}}(\hat{x}_{(\alpha)}) = \psi_\alpha([\hat{x}_{(\alpha)}]), \tag{3.6}$$

where the new variable domains are $\hat{\mathcal{X}}_i = \mathcal{X}_i$ and $\hat{\mathcal{X}}_{(\alpha)} = (\mathcal{X}_\alpha)$. In this notation, α and x_α are seen as sets or vectors of values, while α and x_α are "scalar" encodings of the same quantities. Here [ ] is used as a kind of inverse of ( ), so $x_\alpha = [\hat{x}_{(\alpha)}]$ indicates the vector of variable assignments x_α (in the old graph) corresponding to the single variable assignment $\hat{x}_{(\alpha)} = (x_\alpha)$ (in the new graph).

The new pairwise potentials are constructed to enforce consistency between the representatives of the old variables and copies of them appearing in representatives of the old factors by assigning zero weight to illegal states, and the new singleton potentials incorporate the values of the old factors, with the result that the legal states have the same weight as in the original graph. The transformation is illustrated in the following diagram:

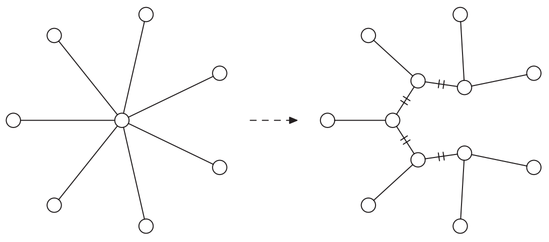


(3.7)

The complexity of the reduction is technically $O(n^{4/3})$, because a 3-ary factor with variables of large arity d will have size d^3 but yields consistency factors $\{i, (\alpha)\}$ of size d^4. These consistency factors could be encoded more efficiently due to their regular structure. Furthermore, if the variable arity is bounded above, then the reduction is $O(n)$ (linear).

It is straightforward to check that for the above construction, belief propagation (BP) on the reduced pairwise graph is equivalent to BP on the original graph. By contrast, mean field does not commute with the transformation.

3.2.1 Pairwise Graphs of Bounded Degree. As an important special case, we also address the question of the universality of pairwise graphs with nodes of bounded degree. The "degree" of a node is defined as the number of factors containing it. Clearly, the class of pairwise models with all nodes having degree ≤ 2 cannot be universal, since in such models, all variables must be connected as a single path and will be constrained to satisfy the conditional independence relationships of such a topology (Pearl, 2000). But it is straightforward to reduce a pairwise factor graph to an equivalent graph with all nodes of degree ≤ 3, as also observed previously (Fisher, 1966; Chertkov et al., 2008). The transformation to be applied to each node of degree $d > 3$ is depicted below, for $d = 7$:



(3.8)

Edges with a double tic indicate binary factors with "identity" potentials $\begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix}$. These edges force the auxiliary variables to all share the same value. Each node of degree $d > 3$ is replaced by $d - 2$ new nodes in this transformation, and $d - 3$ new edges are added to the graph. Note that the arity of the auxiliary variables is the same as those of the input model. Also, for planar graphs (defined in section 3.5), planarity and even the number of faces is preserved. This means that in all of our universality results, the "pairwise" class can also be understood as the class "pairwise, with nodes of degree ≤ 3". We will leave this implicit in what follows. To summarize:

Theorem 5. *Any pairwise factor graph can be reduced to a pairwise factor graph with nodes of degree ≤ 3. This reduction can be done in such a way that maximum variable arity and graph planarity is preserved.*

3.3 Binary Factor Graphs. We next consider factor graphs with all variables having domain size 2 (but factors of arbitrary size). Such graphs are called binary. We can easily reduce general factor graphs to binary form by introducing an arbitrary binary encoding for each input variable and then adjoining factors that reproduce the values of the original factors on the encoded states. In its simplest incarnation, such a reduction would assign zero weight to states in the new graph that do not correspond to any states in the original (i.e., they are outside the image of the encoding function).

However, if we construct our reduction a bit more carefully, we can see that it is not necessary for the output graph to have any potentials with entries of zero weight if such zeroes are not present in the input graph. Since zeroes in potentials can pose problems for some inference algorithms and because we will need to make use of the existence of "positive-preserving" reductions in theorem 15, we will now exhibit a binary reduction with this stronger property.

Definition 6. *We will call a reduction "positive preserving" if, given any input graph with strictly positive potentials, the output graph will also have strictly positive potentials.*

Theorem 7. *There is a positive-preserving reduction from general factor graphs to binary factor graphs.*

Proof. This construction is more intuitive than it may appear. The idea is to choose a minimal binary encoding for the values of each variable in the input model. Each state in the output model must map back to a unique input state. A construction that simply excludes certain output states by assigning them probability zero is not possible because of the positivity requirement. Instead, some input states are associated with multiple output states, and extra scaling factors are introduced to compensate for the duplicated probability mass. The details follow.

Choose a minimal binary "prefix-free" encoding for the values in each variable's domain $\mathcal{X}_i$.[12] The encoding will correspond to a binary tree with $|\mathcal{X}_i|$ leaves, where each node has either zero or two children (whence "minimal"). The encoded values may contain different numbers of bits, since $|\mathcal{X}_i|$ might not be a power of 2. It is easy to check that such an encoding exists for any $|\mathcal{X}_i| \geq 1$. Additionally, to ensure that the reduction is polynomial, in particular that the size of the output factors is linear in the size of the variable domains and input factors, let the encoding correspond to a ("balanced") binary tree of maximum depth $\lceil \log_2 |\mathcal{X}_i| \rceil$; call this number k_i. In the new graph, for each variable i, introduce k_i binary variables. For each factor α in the original graph, create a factor in the new graph containing variables $\bigcup_{i \in \alpha} \beta_i$, whose entry at a given (binary) assignment y_{β_i} corresponds to the entry of $\psi_\alpha(x_\alpha)$ in the original graph, where $(x_\alpha)_i$ is the unique decoding of the y_{β_i} for each i. Note that this factor has $2^{\sum_{i \in \alpha} \lceil \log_2 |\mathcal{X}_i| \rceil} \leq 2^{|\alpha|} |\mathcal{X}_\alpha| \leq |\mathcal{X}_\alpha|^2$ entries. Finally, we need to compensate for the fact that a single variable assignment x_i in the original graph may correspond to multiple assignments y_{β_i} in the new graph due to the presence of extra unused variables when a particular x_i is encoded with fewer than k_i bits. To this end, attach a factor to y_{β_i} with entries equal to $2^{l_i(y_{\beta_i}) - k_i}$, where $l_i(y_{\beta_i})$ is the actual length of the encoding of the value x_i corresponding to y_{β_i}. This ensures that summing over the unused variables in each encoding gives the correct probability of an assignment in the original graph.

The complexity of this reduction is $O(n^2)$, according to the bound in the proof. This is a tight bound because a variable taking $2^n + 1$ values must be encoded with $n + 1$ variables taking 2^{n+1} values, and as $n \to \infty$ the ratio becomes two. But if we upper-bound the variable arity and factor size in the input model, then the complexity is $O(n)$ (linear).

3.4 Binary Pairwise Factor Graphs. The binary restriction is usually combined with the pairwise restriction of section 3.2, resulting in "binary pairwise" factor graphs (BPFGs). Several algorithms and decompositions have been proposed that apply only to BPFGs, so it is interesting to ask if it is possible to reduce more general factor graphs to the binary pairwise form. Such a reduction might be imagined as first converting the input to binary form by choosing an encoding of the input variables and then adding latent variables to implement the correct distribution over the new graph. We show that for general input graphs—in particular, for those that may

[12] A prefix-free encoding is a variable-length encoding in which no codeword forms a prefix for a longer codeword.

contain states having zero probability—a valid simple reduction to BPFGs does not exist.

Our proof depends on a fact about k-SAT. Recall that k-SAT is the problem of finding satisfying assignments to Boolean formulas written in the format of equation 2.3, namely as a conjunction of disjunctive clauses of size k. For such a formula to be satisfied, every clause must be true, which means that at least one of its positive variables must be true, or at least one of its negative variables must be false. For $k \geq 3$, k-SAT is NP-complete, and in fact we can create a k-SAT instance where a given set of assignments to some variables, and no other assignments, satisfies the formula (possibly by introducing extra auxiliary variables). This is not possible with 2-SAT, however, whose satisfying assignments always form a structure called a "median graph" and can easily be shown to have the following property (Knuth, 2008):

Lemma 1 *(median property). Given a set of three satisfying assignments to a 2-SAT formula, if we construct a new assignment (the "median" of the three) in which each of the variables takes the value it took in the majority of the other assignments, then the new assignment is also satisfying.*

Our theorem follows directly from the observation that the positive states of a binary pairwise factor graph correspond to solutions of 2-SAT:

Theorem 4. *Binary pairwise factor graphs are not universal. In particular, there exist factor graphs that cannot be reduced to binary pairwise form.*

Proof. We will assume that the input graph is binary. Call this graph P and let it as usual be given by

$$P(x) = \frac{1}{Z} \prod_{\alpha \in \mathcal{F}} \psi_\alpha(x_\alpha). \tag{3.9}$$

We see that a state x^* has positive probability if and only if the following Boolean expression is true:

$$\bigwedge_{\alpha \in \mathcal{F}} \bigwedge_{\substack{x_\alpha \in \mathcal{X}_\alpha \\ \psi_\alpha(x_\alpha)=0}} \bigvee_{i \in \alpha} x_i \neq x_i^*. \tag{3.10}$$

Introduce a Boolean variable v_i which is true if $x_i^* = 1$ and false otherwise; the expression becomes:

$$\bigwedge_{\alpha \in \mathcal{F}} \bigwedge_{\substack{x_\alpha \in \mathcal{X}_\alpha \\ \psi_\alpha(x_\alpha)=0}} \left(\Big(\bigvee_{\substack{i \in \alpha \\ x_i=0}} v_i \Big) \vee \Big(\bigvee_{\substack{i \in \alpha \\ x_i=1}} \neg v_i \Big) \right). \tag{3.11}$$

The positive states of P are thus exactly the solutions of a k-SAT instance, where k is the number of variables in the largest factor in $\mathcal{F}$. Any set of states can be realized as a solution set of k-SAT when $k \geq 3$, but when $k = 2$, such sets must obey the median rule defined above. If we can show that our definition of simple reduction preserves lack of median structure, then we are done: an arbitrary model P (without median structure) cannot then be reduced to a binary pairwise model Q (with median structure).

Let F be the PA-map of a representation of $P(x)$ by $Q(y)$, where Q has median structure. Consider a triple of states $x^{(1)}, x^{(2)}, x^{(3)}$ in P (i.e., these are full, not partial, assignments), each with positive probability, and let x^* be their median. These map under F to a triple of PAs $y^{(1)}_{r_1}$, $y^{(2)}_{r_2}$, $y^{(3)}_{r_3}$ in Q. Since each PA $y^{(i)}_{r_i}$ has positive probability $Q(y^{(i)}_{r_i}) = P(x^{(i)})$, it can be extended to a full state $y^{(i)}$ with positive probability. The median of these three states we call y^*. Since we assumed the median property for Q, we have $Q(y^*) > 0$. Now we would like to show that the full state y^* is an extension of the PA $F(x^*)$. This follows from the variable intersection rule for PA maps: $F(x) = \bigcap_i F(x_i)$. More specifically, let i be a variable in P. Since x^*_i is a median of $(x^{(1)}_i, x^{(2)}_i, x^{(3)}_i)$, it must have the same value of two of these—say, without loss of generality, $x^{(1)}_i$ and $x^{(2)}_i$. But $y^{(1)}_{r_1}$ and $y^{(2)}_{r_2}$ will then both be consistent with $F(x^*_i) = F(x^{(1)}_i) = F(x^{(2)}_i)$. As a consequence, y^* will share this consistency: any variable that is fixed in $F(x^*_i)$ will appear in both $y^{(1)}$ and $y^{(2)}$ and hence y^*. Since we have shown y^* is consistent with $F(x^*_i)$ for all i, it follows that y^* must be an extension of $F(x^*)$.

Now, $Q(y^*) > 0$ since we assumed Q to have median structure. But $y^* \in F(x^*)$ so $P(x^*) = Q(F(x^*)) \geq Q(y^*) > 0$. Thus, x^* has positive probability in P. Hence, P has median structure.

We have proven that our reductions preserve lack of median structure, from which it follows that MI in a model whose positive states lack median structure cannot be reduced to MI in a binary pairwise factor graph. We have indicated that general factor graphs do not have median structure, but it may help to give a concrete counterexample. The following distribution, which we call the XOR distribution, lacks the median structure and so is not representable by a binary pairwise graph:

$$P(s_1, s_2, s_3) = \begin{cases} \frac{1}{4} & \prod_i s_i = -1 \\ 0 & \text{otherwise} \end{cases} . \tag{3.12}$$

where $s_i \in \pm 1$. The median structure demands that $s = (1, 1, 1)$ has a positive probability, since the following three positive configurations each have a majority assignment of 1 for each variable:

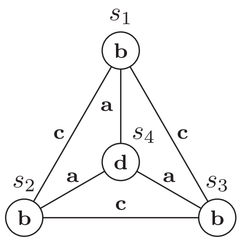
$$\begin{array}{lrrr}
 & s_1 & s_2 & s_3 \\
\hline
 & -1 & 1 & 1 \\
 & 1 & -1 & 1 \\
 & 1 & 1 & -1 \\
\hline
\text{median:} & 1 & 1 & 1
\end{array} \tag{3.13}$$

But the distribution assigns it a zero probability.

We saw that the XOR distribution of equation 3.12 cannot be represented by a binary pairwise factor graph. It is, however, possible to construct a sequence of binary pairwise graphs that approaches the XOR distribution with arbitrary precision. This is because it is possible to implement the following distribution, whose general n-wise form we call "soft-XOR," as a binary-pairwise factor graph. For finite k:

$$P(s_1, s_2, s_3) \propto \exp\left(k \prod_{i=1}^{3} s_i\right). \tag{3.14}$$

The following explicit construction is due to Martijn Leisink (personal communication, 2010). Introduce an auxiliary variable s_4 and create a network:



, (3.15)

with weights shown (corresponding to pairwise and singleton factors $\exp(as_1s_4)$, $\exp(bs_1)$, and so on), having values:

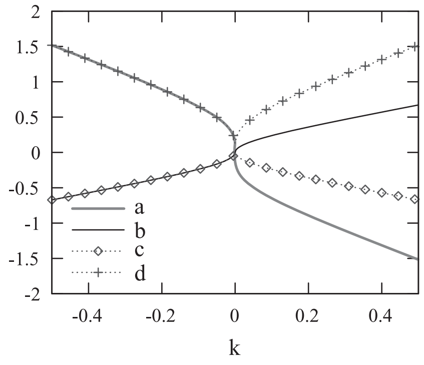
$$b = \frac{k}{4|k|}\,\text{acosh}\left(e^{4|k|}\right), \tag{3.16}$$

$$c = -|b|, \tag{3.17}$$

$$a = \frac{-k}{4|k|}\,\text{acosh}\left(e^{8|b|}\right), \tag{3.18}$$

$$d = |a|. \tag{3.19}$$

The relationships of the weight parameters to k are plotted here:



(3.20)

This set of weights is not unique, since although there are four unknown weights and four unique (up to permutation) values for the state $s_{1:3}$, the partition function of the new model is an extra degree of freedom that has been constrained by the simplifying choice, $d = |a|$, from which follows $c = -|b|$ and the other two equations. It is straightforward to verify that the network induces the distribution P of equation 3.14 on $s_{1:3}$ when marginalizing out s_4.

Check that when k is set to $\pm\infty$, the distribution becomes unnormalizable (even if each factor is normalized independently). Thus, the construction works only for finite k. However, in the limit as $k \to \pm\infty$, the distribution over $s_{1:3}$ approaches the XOR distribution (or its complement).

We will show that using the Leisink construction to implement binary pairwise soft-XORs of a given strength k, it is possible to construct a binary pairwise factor graph that approximately reduces any given input graph. The error of such an approximation can be made arbitrarily small by taking k to ∞. Thus, we can think of binary pairwise factor graphs as universal under an approximate form of reduction, based on a limit concept. We propose the term *"almost universal"* to describe factor graph classes with this property:

Definition 10. *A class C of factor graphs is "almost universal" if DFGs can be simply reduced to it in a limit. In other words, given an input graph with model $P(x)$, there is an infinite sequence of models $Q_1(y)$, $Q_2(y)$, ... with the following properties:*

1. *The Q_n are represented by factor graphs in C with the same connectivity (but presumably varying parameters).*
2. *Given an $\epsilon > 0$ we can find an N such that for all $n > N$, $Q_n(y)$ simply reduces some model $P_n(x)$ whose (multivariable) marginals are within ϵ of $P(x)$.*
3. *As in definition 1, the size of the graphs implementing the Q_n should be bounded by a polynomial in the size of P.*

Note that since we are dealing with finite graphs, the calculation of distance between marginals can be done according to any norm without affecting the definition. The limit is taken after fixing an input model. We make no guarantees about the rate of convergence as a function of input size. Finally, our notion of factor graph size is based on counting parameters, and a more refined definition that accounts for the cost of increasing numerical precision would require some adaptations to the above definition, since the size of the models Q_n would then no longer be constant. We leave this for future work.

We can now formulate the following theorem:

Theorem 11. *Binary pairwise factor graphs are almost universal.*

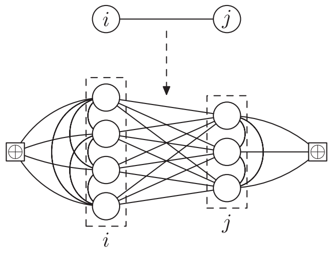
There is already theoretical support for a kind of universality in BPFGs. Valiant (1979b) proved the #P-completeness of #2-SAT, and even monotone #2-SAT, which corresponds to BPFGs with all potentials $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$. This implies that a polynomial-time reduction can be made from factor graphs with rational potentials to BPFGs with $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ potentials, but this reduction may be very slow and is not a "simple reduction." Here we give a limiting simple reduction.

We proceed to prove theorem 11.

Proof. Assume, without loss of generality, that the original graph is in pairwise form. Now create a new graph with a binary indicator variable $k = (i, x_i)$ for each of the (variable, value) pairs in the old graph, which will be by construction $y_k = 1$ if the variable i takes value x_i in the old graph and $y_k = 0$ otherwise. Introduce an edge $(k, l) = ((i, x_i), (j, x_j))$ for each edge (i, j) in the old graph and each pair of values (x_i, x_j), with factor potentials $\hat{\psi}_{kl}(y_k, y_l)$ equal to 1 if either $y_k = 0$ or $y_l = 0$ and equal to $\psi_{ij}(x_i, x_j)$ otherwise. One can see that this graph has an unnormalized joint that coincides with that of the original graph for each allowed state. We still need to exclude states where a variable i takes multiple values, that is, states y for which $y_{(i,x_i)} = y_{(i,x'_i)} = 1$ for some $x_i \neq x'_i$; and we need to ensure that at least one $y_{(i,x_i)}$ is 1 for each i.

We try to create a "1-of-n" gadget as follows. For each variable i and for each pair of values $x_i \neq x'_i$, introduce an edge $((i, x_i), (i, x'_i))$ with factor potential equal to zero if both $y_{(i,x_i)}$ and $y_{(i,x'_i)}$ are 1, and equal to 1 otherwise. This ensures that no more than one $y_{(i,x_i)}$ is 1 for each i. But the remaining case where $y_{(i,x_i)} = 0$ for all x_i is not yet excluded by the new graph. In fact, it is impossible to exclude it using only binary pairwise factors when $n \geq 3$, since it is a median of the other valid states. We can, however, exclude it by introducing a new XOR factor of size $|\mathcal{X}_i|$, which ensures that an odd (and therefore nonzero) number of the $y_{(i,x_i)}$ is equal to 1.

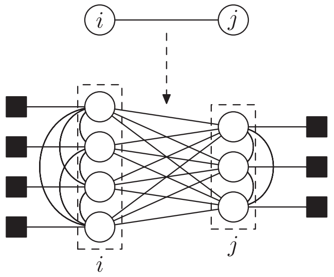
The following diagram describes the transformation for the case $|\mathcal{X}_i| = 4$ and $|\mathcal{X}_j| = 3$ (the two XOR factors are marked $\oplus$):



(3.21)

An XOR factor of size n can be constructed by combining $n - 2$ XOR factors of size 3 and introducing $n - 3$ auxiliary variables; a single edge can be used when $n = 2$. This resembles equation 3.24. XOR factors of size 3 can be achieved as a limit of binary pairwise graphs by letting $k \to \pm\infty$ in Leisink's construction of equation 3.14.

An alternative method, which creates a limiting reduction directly, is to give the model's valid states an extra weight t, which is allowed to vary. To do this, attach a singleton factor with potential $[\,1\ t\,]$ to each output variable:



(3.22)

The probability of the all-zero case for each variable then goes to zero as $t \to \infty$.

With either construction, the size of the output graph is at most quadratic in the size of the input. But if variable arities and factor sizes are bounded, the relationship is linear.

This also shows

Corollary 12. *Binary 3-wise factor graphs are universal.*

Since the 3-wise to pairwise reduction breaks down only in the presence of potential functions with some entries equal to zero, we ask whether it is possible to perform the binary pairwise reduction in a way that avoids resorting to a limit when graph potentials are all strictly positive.

We have already exhibited a positive-preserving reduction to binary factor graphs in theorem 7. If we can show that arbitrary n-wise binary factors with positive entries can be implemented using only pairwise factors, then we will have our positive BPFG reduction. We accomplish this in two steps, first by demonstrating that an n-wise soft-XOR factor of arbitrary finite strength can be constructed out of n 3-wise soft-XORs, and then by showing how to combine soft-XOR factors of sizes 1 through n to implement a factor with arbitrary positive potentials. As with the previous theorem (theorem 11), the construction is a proof of concept; it is not expected to be minimal, except that it satisfies the polynomial size constraints outlined earlier in definition 1.

Note that *positive universal* implies *almost universal*, for the finite and continuous reductions we consider in this letter.

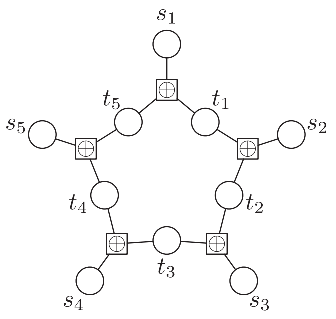
Proposition 13. *The n-wise soft-XOR factor*

$$P(s) \propto exp\left(k\prod_{i=1}^{n} s_i\right) \tag{3.23}$$

(with k finite) can be implemented in binary pairwise form.

Proof. An implementation of the above factor with $k \leq 0$ can be represented by a $k \geq 0$ factor by flipping the sign of one of the variables, so assume $k \geq 0$.

Consider connecting the n binary variables $s_{1:n}$ with 3-wise soft-XOR factors, each of strength k', and n auxiliary variables $t_{1:n}$ in a loop as shown for $n = 5$:



(3.24)

We will prove that for any k, we can always find a k' such that the graph in equation 3.24 implements the distribution of equation 3.23. The probability of a configuration of the s variables is

$$P'(s) \propto \sum_t \exp(k'(t_n s_1 t_1 + t_1 s_2 t_2 + \cdots + t_{n-1} s_n t_n)). \tag{3.25}$$

This summation has 2^n terms. Observe that when two states s and s' have the same parity, then the terms in the summation over t for $P'(s)$ are a permutation of those in the summation over t for $P'(s')$. To prove this, consider flipping a neighboring pair of s variables, say, s_i and s_{i+1}. The effect is the same as flipping t_i, which exchanges pairs of terms in the sum, leaving the total value invariant. But a sequence of such flips can be used to go between any s and s' if they have the same parity. In particular, flipping t_i for every i where $\prod_{j=1}^{i} s_j = -1$ rearranges the terms to correspond to $s = (1, 1, \ldots, 1)$ (if $\prod_{j=1}^{n} s_j = 1$) or to $s = (-1, 1, \ldots, 1)$ (if $\prod_{j=1}^{n} s_j = -1$). This shows that $P'(s)$ takes only one of two values:

$$P'(s) = \begin{cases} p_1 : \prod_{j=1}^{n} s_j = 1 \\ p_2 : \prod_{j=1}^{n} s_j = -1 \end{cases} \tag{3.26}$$

for some p_1 and p_2, which is the same as saying

$$P'(s) \propto \exp\left(k \prod_i s_i\right), \tag{3.27}$$

where $k = \frac{1}{2}\log\frac{p_1}{p_2}$, and hence $P'(s; k') = P(s; k)$ for this choice of k.

It remains to verify that the function mapping k' to k can be inverted. At least for small n, this function appears to be strictly monotonic, but it is not necessary to prove that fact in general. All that is needed is to observe that $\frac{p_1}{p_2}$ is a ratio of positive continuous functions of k' (each given respectively by the right-hand side of equation 3.25 for two different values of s). Thus, k is a continuous function of k'. Also note that $k' = 0 \Rightarrow p_1 = p_2 \Rightarrow k = 0$, and $k' \to \infty \Rightarrow k \to \infty$. The second implication can be reached by considering the values of the network when the 3-wise soft-XORs become "hard"-XORs. The intermediate value theorem then implies that for any positive k, we can find a k' such that the graph of equation 3.24 is equivalent to a n-wise soft-XOR of strength k.[13]

Corollary 14. *Any n-wise binary factor with strictly positive entries can be implemented in binary pairwise form.*

Proof. The 2^n functions $s \mapsto s_1^{e_1} s_2^{e_2} \ldots s_n^{e_n}$ parameterized by a vector $e \in \{0, 1\}^n$ form an independent basis for the space of real-valued functions of s, so we can write the factor's potential function as $\exp(\sum_e a_e s_1^{e_1} s_2^{e_2} \ldots s_n^{e_n})$ for some set of coefficients a_e. But such a potential can be implemented by superimposing 2^n soft-XOR factors of strength a_e, each covering subsets of the variables selected by the vector e. If the construction of proposition 13 is used for these factors, the output size will be $O(k \log k)$, where $k = 2^n$ is the number of potential function entries of the original factor.

Together with theorem 7, this proves:

Theorem 15. *Binary pairwise graphs are positive universal.*

Assuming bounded variable arity and factor size, the reduction from DFGs is again $O(n)$.

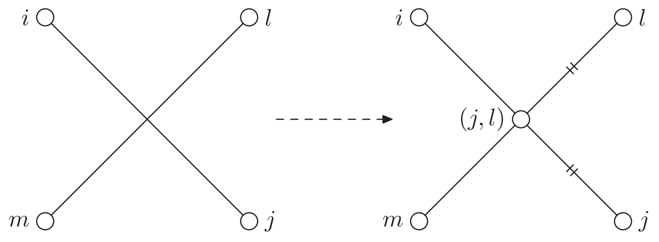
3.4.1 Implications of Nonuniversality. In a certain sense, because complexity classes are defined by reducibility relations, the nonuniversality of BPFGs implies that they occupy a different complexity class from general discrete factor graphs. It is not clear what, if any, implications our result has for realizable efficiency of inference algorithms on the various model classes. All eight of our classes are $\#\mathrm{P}_{\mathbb{Q}}$ -complete (implying NP-hardness). Although these complexity measures are well understood, if not rigorously proven, to imply exponential running time in the worst case, they bound computation time only as a function of problem size (in bits). But (as mentioned in section 3.1) it is common to quantify the difficulty of an inference

[13]The graph of equation 3.24 shows a particular implementation of an n-wise soft-XOR, with a cycle topology. Other topologies are possible, and in fact the proof to corollary 14 can be adapted to show that any network connecting n variables with soft-XORs and auxiliary nodes is equivalent to an n-wise soft-XOR at a given strength, provided that (1) the factor graph is connected, (2) each of the n observed variables belongs to a single soft-XOR factor, and (3) each auxiliary variable belongs to exactly two soft-XOR factors. For such networks, the mass assigned to a particular setting of observed and auxiliary variables is the same as that assigned to the result of flipping two observed variables, as well as every auxiliary variable along an arbitrary path between them. Connectedness ensures that such a path always exists. Any two observed-variable assignments with the same parity can be related by a sequence of such double-flips. As in the proof, this means that summing over the auxiliary variables gives a probability mass that depends on only the parity of the observed variables (the terms in each summation being a permutation of those in the other). The strength and size of the various soft-XOR factors need not even be the same for this equivalence to hold.

problem using other metrics, such as those based on parameter magnitude (Mooij & Kappen, 2005a; Ihler, 2007). Even in the satisfaction framework, difficulty metrics based on connection density play an important role in the classification of random k-SAT problems (as in Braunstein et al., 2005). Understanding how our reductions interact with these or other measures of inference difficulty might help shed some light on the possibly special status of BPFGs.

3.5 Planar Binary Pairwise Graphs. Finally, we address the problem of reducing an arbitrary factor graph to planar form. Planar graphs are defined as graphs that can be drawn in a plane ($\mathbb{R}^2$) without any crossing edges. This condition is equivalent to forbidding K_5 and $K_{3,3}$ graph minors (see Wagner's theorem in Wagner, 1937, or Kuratowski's theorem of Kuratowski, 1930). Planar graphs have a number of special properties. For instance, a closed non-self-intersecting path splits a planar graph into two components, in analogy to the Jordan curve theorem. Also, a planar graph has a naturally defined dual, which is also planar. The planar separator theorem (Ungar, 1951) may be used to engineer efficient divide-and-conquer algorithms for planar graphs.

It seems useful to consider the possibility of reducing MI on general graphs to MI on planar graphs, partly because of the existence of a handful of results that apply to MI on planar graphs and also because, due to the special properties of planar graphs, one might anticipate that more of these results may be derived in the future. If we allow planar graphs to have variables with arbitrarily large domain, then the reduction task is straightforward. We just reduce the graph to pairwise form, draw the resulting graph in two dimensions (with an edge for each factor), and introduce a new variable wherever two edges cross. The new variable encodes the values at an arbitrarily chosen end point of each of the two original edges:



(3.28)

In the above notation, the pairwise factor in the right-hand diagram between the new (j,l) variable and j enforces consistency between x_j and $x_{(j,l)}$, and similarly for the factor between (j,l) and l (in both cases, this is indicated with a double tic). The domain of $x_{(j,l)}$ is just $\mathcal{X}_j \times \mathcal{X}_l$. The new factors between i and (j, l) and between k and (j, l) are filled with copies of the entries of ψ_{ij} and ψ_{kl}, respectively, similarly to the factors in the pairwise reduction theorem (theorem 4). It is possible to show that the new graph is at most quadratic in the size of the old. This proves:

Theorem 16. *Planar pairwise factor graphs are universal.*

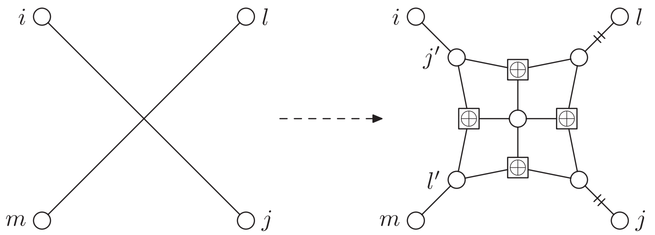
Since the model size is linear in the number of edges and four new edges may be introduced for each pair of crossing edges, the complexity of this reduction is $O(n^2)$.

Finding a reduction for the binary pairwise planar case is more difficult since only two values can be used to propagate data across an intersection. Inference in binary pairwise planar graphs was shown to be NP-hard by Barahona (1982), which suggests that there could be a way to reduce ordinary factor graphs to binary pairwise planar factor graphs. Such a reduction would be of interest because of the existence of a number of results that apply only to the planar binary pairwise case, in approximate inference and statistical physics (see section 2.2).

To start, it is not difficult to effect such a reduction in a limit:

Theorem 17. *Planar binary pairwise factor graphs are almost universal.*

Proof. Reduce the graph to binary pairwise form as described above and replace each pair of crossed edges with the following subgraph, using soft-XOR 3-wise factors of strength m, implemented in binary via the Leisink construction.



(3.29)

As previously, edges with a double tic enforce the constraint that their end point variables match (i.e., in this case, they have potentials $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$). The factor connecting i and j' in the new graph should be the same as ψ_{ij} in the old graph, and in the same way for k and l'. Crossed edges can be replaced iteratively, making sure to draw the subgraph of equation 3.29 at each step so that any other edges crossing either kl or ij in the original pass through one of its outer four edges.

In the limit as $m \to \infty$, the soft-XOR factors become XOR factors; note that $x_{l'}$ is then forced to take the value of x_l, and $x_{j'}$ to take the value of x_j. The central auxiliary variable simply reflects whether $x_l = x_j$.[14]

The reduction is again quadratic. For completeness, we also consider the universality of planar binary graphs (i.e., relaxing the pairwise constraint). If we transform the input first to a binary n-wise graph as in theorem 7, satisfying the polynomial complexity requirements, then it is easy to use the edge-uncrossing technique of equation 3.29 to make the result planar. In this case, the variables i and k in the diagram would represent factors. The XORs would be "hard" XORs, implemented using 3-wise factors. This proves that

Corollary 18. *Planar binary graphs are universal.*

Recall that although BPFGs could simply reduce arbitrary DFGs only in a limit, we were able to find a (nonlimiting) simple reduction from positive DFGs to BPFGs (see theorem 15). For the planar case, it was tempting to conjecture that only a limiting simple reduction exists from positive DFGs to planar BPFGs. Note that BPFGs with topologies like K_5 or $K_{3,3}$ induce distributions that, although general from the perspective of conditional independence, nevertheless enjoy a special structure that is absent from random distributions over five and six binary variables, respectively. This structure corresponds to a submanifold of the space of distributions. One might imagine that planar BPFGs would be able to approximate such structure only in a limit, given that no planar graph can contain K_5 or $K_{3,3}$ minors. However, this is not the case. It is possible to modify the previous theorem proof to obtain an *exact* crossover gadget for positive input potentials. Before describing how to do this, we first prove:

Lemma 19. *For $k \neq 0$, the equations*

$$tanh(a + b) tanh(a + c) = tanh\ k, \tag{3.30}$$

$$tanh(a - b) tanh(a - c) = tanh - k, \tag{3.31}$$

have a solution if and only if $|a| > |k|$.

Proof. The proposition holds for any odd sigmoid function with range $(-1, +1)$, not just tanh. If $k = 0$, then set all other variables to zero. Otherwise, if $a = 0$, then both of the left-hand sides are positive and the equation

[14]Observe that any one of the four soft-XOR factors can be removed as long as three remain, without changing the limiting behavior of the subgraph in equation 3.29.

is not solvable. For the remaining cases, note that we can choose both k and a positive by making appropriate sign inversions. Consider that as $a > 0$, at least one of $(a + b, a - b)$ must be positive, and similarly for $(a + c, a - c)$. Also, exactly one of these pairs must have opposite signs, since the signs on the right of equations 3.30 and 3.31 are opposite. So assume, without loss of generality, that $a + b > 0, a - b > 0, a + c > 0$, and $a - c < 0$. If $|a| \leq |k|$ then either $|a + b| \leq |k|$ or $|a - b| \leq |k|$, suppose the former. But since $|\tanh| < 1$ and tanh is monotonic, we have a contradiction:

$$|\tanh k| = |\tanh(a + b) \tanh(a + c)| < |\tanh(a + b)| \leq |\tanh k|. \quad (3.32)$$

Now assume $|a| > |k|$. From the foregoing considerations, we must have both $a + b > k$ and $a - b > k$. Taking the first equation, we solve for $c = \operatorname{atanh} \frac{\tanh k}{\tanh(a+b)} - a$. Substitute this c into the second equation and consider varying b continuously from 0 to k—a. Then $b = 0 \Rightarrow$

$$c < 0 \Rightarrow \tanh(a - b) \tanh(a - c) > 0 > \tanh -k, \quad (3.33)$$

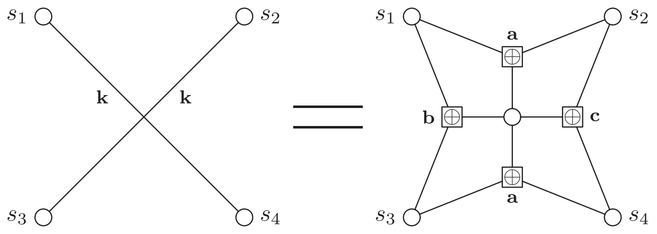
while $b = k - a \Rightarrow$

$$c = \infty \Rightarrow \tanh(a - b) \tanh(a - c) = -\tanh(2a - k) < \tanh -k. \quad (3.34)$$

By the intermediate value theorem and continuity, there must be a value of b where the second equation is also solved.

Theorem 20. *Planar BPFGs are positive universal.*

Proof. We first show that it is possible to implement the crossover network $\exp(ks_1s_4 + ks_2s_3)$ using four threewise soft-XORs, having strengths a, b, and c, as shown:



(3.35)

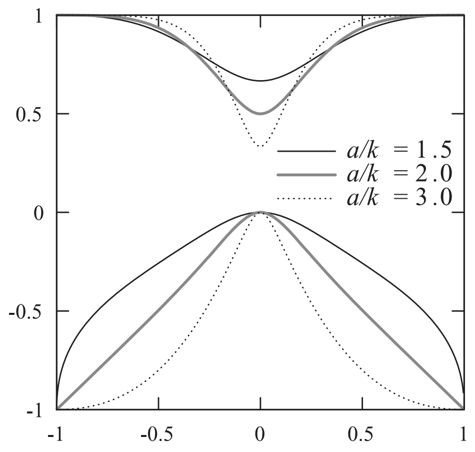
Both networks have the following inherent symmetries: (1) invert (s_1, s_4), (2) invert (s_2, s_3), (3) flip vertically: $s_1 \leftrightarrow s_3$, $s_2 \leftrightarrow s_4$. The network on the left, of course, has additional symmetries that are not shared by the soft-XOR gadget on the right for arbitrary values of a, b, and c. (We were not able to find a more symmetrical planar network implementing the crossover.) Although there are 16 possible assignments to $s_{1:4}$, these fall into three classes under the above symmetries. Call them $L = \{s_{1:4} = (1, 1, 1, 1)\}$, $M = \{s_{1:4} = (1, 1, 1, -1)\}$, and $R = \{s_{1:4} = (1, -1, 1, -1)\}$. Equating the ratios L/M and R/M for the left and right models and making use of the identity

$$\frac{1}{2} \log \frac{\cosh(x+y)}{\cosh(x-y)} = \operatorname{atanh}(\tanh x \tanh y) \tag{3.36}$$

gives the equations of lemma 19. These can also be solved analytically for given a and k. First switch to a tanh basis: $\kappa \equiv \tanh k$, $\alpha \equiv \tanh a$, $\beta \equiv \tanh b$ $\gamma \equiv \tanh c$. Then

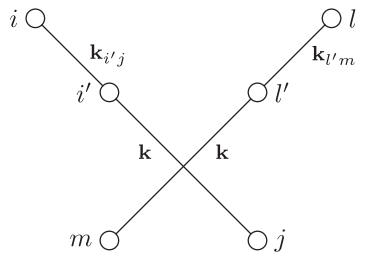
$$\beta, \gamma = \frac{\kappa(1-\alpha^4) \pm \sqrt{(\kappa(1-\alpha^4))^2 - 4\alpha^2(1-\kappa^2\alpha^2)(\kappa^2-\alpha^2)}}{2\alpha(1-\kappa^2\alpha^2)}. \tag{3.37}$$

The values of β and γ are plotted versus κ for three values of a/k:



(3.38)

Now, to implement the reduction, first convert the input model to positive binary pairwise form. Write the potentials of the new model in the spin (± 1) basis: $\psi_i(s_i) = \exp(h_i s_i)$, $\psi_{ij}(s_i, s_j) = \exp(k_{ij} s_i s_j)$. Whenever two pairs of edges, say (i, j), and (l, m), cross, choose $k > \max(|k_{ij}|, |k_{lm}|)$ and subdivide each edge as shown,



(3.39)

with new weights $k_{i'j} = \operatorname{atanh} \frac{\tanh \mathrm{k}_{\mathrm{ij}}}{\tanh \mathrm{k}}$, and $k_{l'm} = \operatorname{atanh} \frac{\tanh \mathrm{k}_{\mathrm{lm}}}{\tanh \mathrm{k}}$. Replace the new pair of crossed edges (i', j) and (l', m) with the soft-XOR gadget of equation 3.35, implemented in planar BPFG form using Leisink's construction.

We have exhibited one type of reduction to planar form as a proof of existence. It is interesting to speculate on the possibility, given a general factor graph, of finding a minimal representation—in either binary pairwise planar form or one of the other classes—that encodes the same or similar probabilities and also minimizes some complexity measure such as number or magnitude of parameters. It should be possible to experiment with finding minimal representations using entirely numerical methods, but we have not tried to do so.

3.6 Spectral Reductions. We showed that it is not possible to reduce general DFGs (with zeroes) to BPFGs using the simple reduction of definition 1. Here we show that a continuous polynomial-time reduction for this case is still possible, using a polynomial interpolation technique originally introduced by Valiant (1979b) for his reduction from perfect matchings to imperfect matchings:

Theorem 21. *DFGs are reducible to BFPGs.*

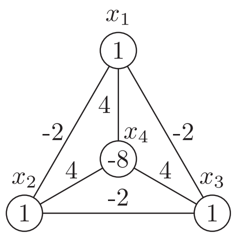
Proof. Consider the 1-of-n gadget of theorem 11. In that theorem, the all-zero case was excluded using two constructions. The first was a limit of soft-XOR factors. The second attached a $[\,1\ t\,]$ potential to each variable and considered the limit as $t \to \infty$. Taking a closer look at the second construction, we see that the partition function can be written as a polynomial in t; call it $Z(t)$. Each 1-of-n gadget, corresponding to a distinct variable in the original model, contributes a factor of either 1 or t to each nonzero term of $Z(t)$, according to whether all of its variables are zero (the illegal state) or whether one variable is one, respectively. For example, the constant term of $Z(t)$ is the partition function of a model where each 1-of-n gadget is in the illegal all-zero state; the linear term describes a model where exactly one gadget has a "one," and so on. The original partition function Z_0 appears as the leading coefficient of this polynomial, which has degree equal to the number of variables $|\mathcal{V}|$ in the original model. Z_0 can be recovered, along with all of the other $|\mathcal{V}|$ coefficients, by evaluating the polynomial at $|\mathcal{V}| + 1$ different values of t. The relationship between Z_0 and the vector of evaluations $(Z(t_i), i = 1 \ldots |\mathcal{V}| + 1)$ is of course continuous.

The reduction multiplies the complexity of inference by the number of model variables. This is unfortunate but not unheard of: algorithms such as linear response (Welling & Teh, 2004) introduce a similar added complexity over BP and MF.

This proof does not directly imply a universality for planar BPFGs via the reduction of theorem 20, which applied only to positive BPFG inputs. We now give a separate reduction from BPFGs with zeroes to planar BPFGs, along the same lines as the above:

Theorem 22. *DFGs can be reduced to planar BPFGs.*

Proof. We adapt the MAP 3-wise to pairwise transformation of equation 2.4 to create a gadget parameterized by an unknown t such that the leading coefficient of the resulting partition function is the same as that of a model in which the gadget is replaced by an XOR. Consider an optimization problem with binary (0,1) variables, and weights given by the following diagram:



(3.40)

In other words, for fixed $x_{1:3}$ the objective is:

$$\max_{x_4} - 8x_4 + 4x_4(x_1 + x_2 + x_3) + (x_1 + x_2 + x_3) \\ - 2(x_1x_2 + x_1x_3 + x_2x_3) \qquad (3.41)$$

Check that the value of this expression is 1 if an odd number of the $x_{1:3}$ is 1, and zero otherwise. Now create a factor graph with power-of-t potentials given by the above weights. For example, connecting x_4 and x_3 will be the factor $\begin{bmatrix} 1 & 1 \\ 1 & t^{-2} \end{bmatrix}$. The leading coefficient of the partition function of a model containing this gadget will correspond to the partition function of a model in which the gadget implements an XOR. If l is the number of XORs, the number of coefficients is $\leq 9l + 1$, corresponding to 1+ the range of the argument of equation 3.41. So this many evaluations of the partition function can be made to identify all the coefficients. The XORs can be used as in the construction of theorem 17 to uncross all of the crossed edges in a BPFG.

Note that the gadget of equation 3.40 can be used to show that binary pairwise planar MAP is universal in the optimization framework, with a construction similar to that of theorem 17. The weights must be scaled so that the range of values of the gadget is larger than the range of the original optimization problem. The spectral reduction method gives in general an interesting connection between inference and integer-valued optimization problems.

The theoretical computer science literature on counting problems also presents reductions with many other kinds of interpolation, not just of polynomial coefficients, which deserve further investigation. However, not all of these are obviously applicable to inference with real-valued parameters. Some, for instance, use modular arithmetic.

We have chosen the name *spectral reduction* because using multiple evaluations to compute the coefficients of a polynomial is reminiscent of the Fourier transform.

3.7 Summary. We have demonstrated a number of formal reductions between different types of (discrete) factor graphs (DFGs). Each of these reductions proves that inference in one particular class of graphs can be implemented using inference in a more restrictive class. To the best of our knowledge, of the theorems appearing in this section, only theorems 4 and 5 have been published before.

We summarize the results. All of the three classes (pairwise, binary, planar) by themselves can simply reduce DFGs, that is, they are "universal" (see definition 2). Planar binary factor graphs and planar pairwise factor graphs are also universal. Binary pairwise factor graphs (BPFGs) and planar BPFGs are universal only for models with strictly positive potential functions, but not for general models—what we have called "positive universal" (see definition 3). Simple reductions from DFGs with zeroes to each of these classes are also possible in a limit ("almost universal"; see definition 10). Finally, continuous polynomial-time reductions of the "spectral reduction" type exist from DFGs with zeroes to BPFGs and planar BPFGs. These results are depicted in Figure 2.

If we can bound the variable arity and factor size in the input DFG, the relationship between input and output model size is $O(n^2)$ for reductions to planar form, and $O(n)$ for all other reductions.

4 Experiments

The efficient mechanics of our simple reduction (definition 1), and its straightforward designation using one-to-one query transformations, set

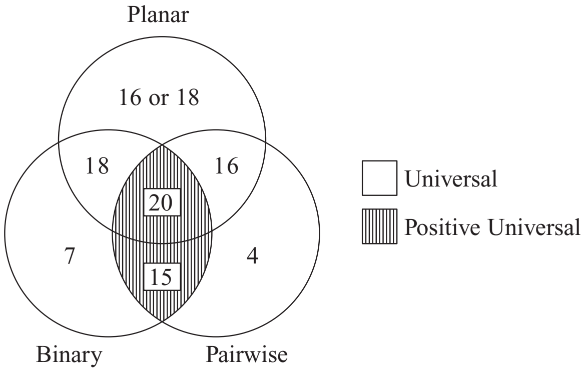


Figure 2: The three factor graph subclasses and their intersections. Regions are shaded according to the universality status of their subclass and numbered with the corresponding theorem or corollary. Recall that “positive universal” also implies “almost universal” according to our definitions.

it apart from existing polynomial-time reduction concepts. The choice of these properties was originally motivated by a need to better understand model transformations from a practical standpoint, with potential applications to modern inference algorithms. At the same time, because we lack an inference algorithm that can be considered optimal, we also lack a precise theoretical characterization of model difficulty. Consequently, it is hard for us to give a useful formal analysis of the ways in which our reductions might make inference on a given model more or less difficult. Instead, we present some computer experiments that are intended to provide a certain amount of insight into that question.

We do not consider reductions from models with multiple variables but focus on the case of a single n-ary variable, implemented in binary pairwise form using the 1-of-n gadget of theorem 11. We fix $n = 5$. We also skew the marginals with a singleton factor (translating under the reduction to five singleton factors) to avoid concerns about symmetry artifacts in the marginal error measurements.[15] But similar results are obtained for uniform scaling, as well as different values of n.

4.1 Simple Reductions. We consider two implementations of the 1-of-5 gadget, corresponding to the ones given in theorem 11. Both use a K_5 graph with $\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ potentials to constrain at most one (0,1) variable to be 1. The first excludes the all-zero case using a limit of soft-XORs (see Figure 3a),

[15]The factor has randomly chosen values (0.42, 0.031, 0.052, 0.43, 0.068), which are also the marginal probabilities for the variable.

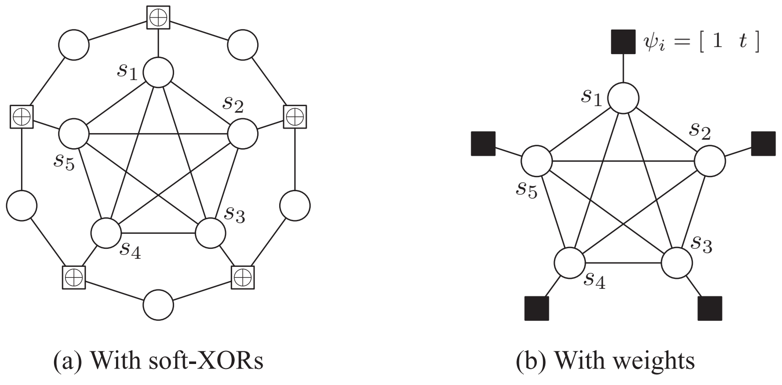


Figure 3: Different implementations of the 1-of-5 gadget.

and the second with a limit of singleton [1 t] weight factors (see Figure 3b). We also explore inference in the second model using the spectral reduction polynomial interpretation method of section 3.6.

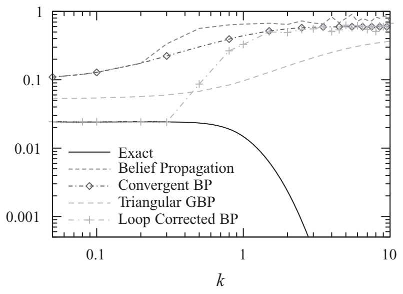
The following five approximate inference algorithms are applied to the "transformed" models:

- **Belief propagation:** Belief propagation (BP) with random order message updates
- **Convergent BP:** BP using the convergent double-loop algorithm of Heskes, Albers, and Kappen (HAK 2003), which is an instance of CCCP (Yuille, 2002)
- **Triangular GBP:** Generalized belief propagation (GBP), with triangular regions, using HAK updates (Yedidia, Freeman, & Weiss, 2001a; Heskes et al., 2003)[16]
- **Loop-corrected BP** (LCBP): Algorithm of Mooij et al. (2007), with full cavity updates
- **Gibbs sampling:** (Geman & Geman, 1984)

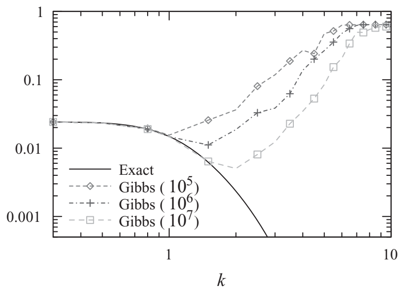
All but Gibbs sampling are based in some way on the BP message passing framework. Mean field message passing is not used because of the presence of zeroes in the K_5 edge factors, which forces the algorithm to put all of its weight on a single variable assignment (Minka, 2005).

None of the algorithms are specific to any of our restricted classes of factor graphs. This is because although these classes have been a useful structure for inference research, all of the currently competitive algorithms

[16]Note that GBP can use regions of arbitrary size. Larger regions are associated with better accuracy but slower convergence. We chose to restrict ourselves to triangular regions for simplicity.

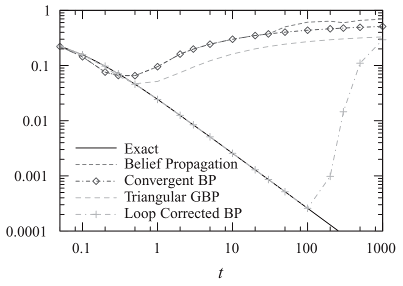


(a) Message-passing algorithms

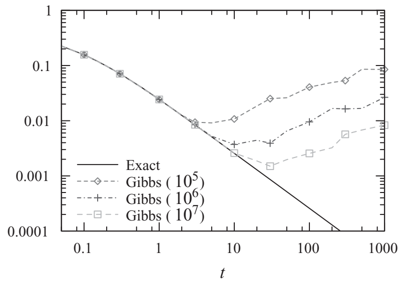


(b) Gibbs sampling

Figure 4: Performance of inference algorithms on 1-of-5 model using soft-XOR factors of strength k.



(a) Message-passing algorithms



(b) Gibbs sampling

Figure 5: Performance of inference algorithms on 1-of-5 model using t-weighting.

that might have been originally defined on them were apparently generalizable.

Errors are average L_1 errors of single variable marginals, computed with respect to exact marginals in the target model. Exact marginals in the target and transformed models are calculated using the junction tree algorithm (Jensen et al., 1990). All algorithms use the implementations of libDAI 0.2.4 (Mooij, 2010). Gibbs, BP, and LCBP results are averaged over 10 runs. The tolerance for convergence of message passing algorithms was set to 10^{-9}.

Figure 4 shows the results for the soft-XOR 1-of-5 as a function of soft-XOR strength k. Errors for message-passing algorithms are shown on the left and for Gibbs sampling on the right. Corresponding plots for the weighted transformed model, as a function of weight t, are shown in Figure 5.

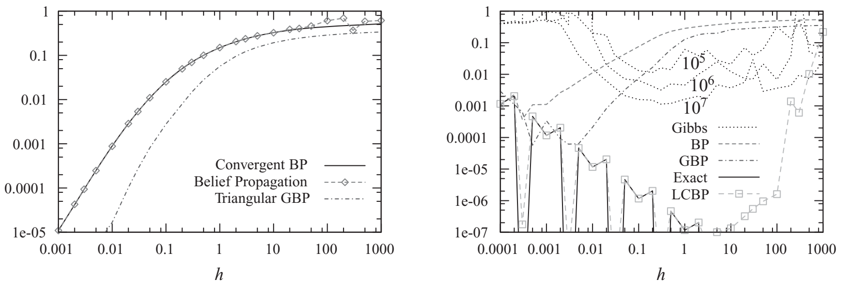
The results are straightforward to interpret. None of the message-passing algorithms was able to perform well on the soft-XOR model. Although LCBP seems to have been able to achieve near-exact marginals for small

k, it starts to fail for $k > 0.3$, well before the approximate soft-XOR model begins to converge to the target model at $k = 1$ (see the "Exact" error curve). Only Gibbs sampling was able to track exact marginals beyond $k = 1$. The point of best accuracy increases as more sampling is performed, occurring at $k = 1$ for 10^5 samples, $k = 1.5$ for 10^6, and $k = 2$ for 10^7. As k increases, the multimodal nature of the model becomes more pronounced, and the relative mass assigned to the all-zero state decreases. The message-passing algorithms have difficulty averaging over these multiple modes, although LCBP, which considers more distant correlations, does well initially. Gibbs sampling must transition through the all-zero state to go between modes, and so more samples are required as the probability of this state in the approximate model goes to zero.

Similar behavior is seen with the weighted transformed model, although it is interesting that the algorithms are almost uniformly more accurate in the weighted representation. Presumably this has something to do with its simpler structure, with only five variables and no soft-XORs. For instance, for Gibbs to transition between two modes in the soft-XOR model, not only a move to the all-zero state but also appropriate changes in the auxiliary variables may be required. The weighted model requires only one transition.

Errors of BP and GBP on the exact 1-of-5 model, implemented using hard-XORs with 3-wise factors, were 0.095 and 0.052, respectively. (Gibbs was unable to make any transitions in this model, and LCBP had convergence issues. Convergent BP had the same error as BP.) These errors are the same as the $k = 0$ soft-XOR errors. In both cases, the auxiliary variables had uniform marginals. These observations point to an interesting property of models that use the Leisink construction. The algorithms BP and GBP were not even able to make use of the 3-wise hard-XOR factor, as evidenced by the fact that the marginals remained the same when it was removed (i.e., replaced by a $k = 0$ soft-XOR, which has uniform potentials). Yet the performance of the algorithms was made even worse when k was increased, showing that the presence of the Leisink gadgets alone, even more than the XOR they are seeking to emulate, can create problems for message-passing algorithms based on BP. It is probable that the loops in the model cause the large factors of the Leisink gadget to be overcounted for large k, while for small k, nonlocal correlations die out and factor contributions are counted more accurately. Similar reasoning would apply to the weighted models, but to a lesser extent.

4.2 Spectral Reductions. We used two interpolation methods to study the BPFG spectral reduction of theorem 21. The first applies to inference algorithms that provide an estimate of the partition function Z of a model, such as BP and GBP. It uses evaluations at multiple t's to estimate the leading coefficient of a polynomial corresponding to the partition function of a model conditioned on x_i. Call this $Z(x_i; t)$. This quantity is approximated by $b_i(x_i; t)Z(t)$ where $b_i(x_i; t)$ is the marginal estimate of variable x_i in the



(a) Using Z, at $t \in \{h, h/2\}$ (b) Without Z, at $t \in \{h, 2h/3, h/3\}$

Figure 6: Performance of "spectral" interpolation methods on 1-of-5 model, with and without estimates of Z.

t-model. Such approximations can be provided for every variable in the model using a single run of the algorithm. After running the algorithm for enough different values of t, approximate polynomials are interpolated for each variable-value pair, and approximate marginals are then obtained by normalizing the appropriate pairs of leading coefficients.

The second method interpolates a rational function corresponding to $Z(x_i; t)/Z(t)$, whose values are approximated by $b_i(x_i; t)$. The second method applies to any inference algorithm, including Gibbs and LCBP, because it requires only variable marginals and no partition function estimate. In this case, we are interested in the ratio of the leading coefficients of the numerator and denominator, which approximates $P(x_i)$.

For single-variable models, method 1 requires two evaluations and method 2 requires three. With method 1, we evaluate at $t_1 = h$ and $t_2 = h/2$, for different values of h. The formula is $Z_0 = \frac{Z(t_1)-Z(t_2)}{t_1-t_2}$. With method 2, we evaluate at $t_{1:3} = (h, 2h/3, h/3)$. The formula is

$$b = \frac{\dfrac{b_1t_1 - b_2t_2}{b_1 - b_2} - \dfrac{b_2t_2 - b_3t_3}{b_2 - b_3}}{\dfrac{t_1 - t_2}{b_1 - b_2} - \dfrac{t_2 - t_3}{b_2 - b_3}}. \tag{4.1}$$

Results for the two methods are shown in Figure 6. Note that the algorithms with Z estimates are able to give an almost arbitrarily small error under method 1 by choosing h small. This is in surprising contrast to the approximate simple reductions presented earlier, where the same algorithms were largely unable to provide useful results.

Similar behavior appears for BP and GBP in method 2, although floating-point precision issues are encountered in the lower left corner of the plot (delimited by the "Exact" line). These precision issues can be seen to arise from the fact that each new coefficient to be estimated requires an additional constant number of digits of accuracy, which is partly dependent on the model and the choice of evaluation points. To put it another way, the three-evaluation method is trying to measure a nonlinearity in $b_i(x_i; t)$, which becomes more linear as $t \to 0$. This is a big concern for spectral reduction methods and should also affect method 1 on multivariate models. However, there may be ways around this problem, such as using complex roots of unity, modifying the algorithm to propagate polynomials in t directly (with truncation), or simply using higher-precision arithmetic. The second option would be close to the linear response algorithm of Welling and Teh (2004). How these ideas could be made to apply to Gibbs sampling, as shown in Figure 6b, is not so clear. Method 2 is able to achieve reasonable results with Gibbs on our 1-of-5 target model, which, due to complete isolation of the modes, is unamenable to direct Gibbs sampling or even other mode-smoothing techniques like tempered sampling (Kimura, Taki, & Kikō, 1991). This is rather encouraging for us, but note that the minimum errors are the same as with the weighted model, shown in Figure 5b, which required only one evaluation, and both the model and our analysis of it are rather specialized.

We would not expect the spectral reduction method to allow us to boost the precision of BP on multivariate input models, because the transformed models would contain loops even if the target (input) model is a tree.

We are interested to know whether it is possible to apply the spectral reduction idea to general inference problems, for instance to disentangle inference in a model by adding free variables that function to smooth over modes, and then, using some kind of polynomial interpolation arithmetic, to recover approximate marginals for the original model.

5 Conclusion

We examined the problem of model reductions from a statistical perspective. We formalized a useful reduction concept called simple reduction based on a single-valued mapping of queries between models. We were able to completely characterize the simple reducibility of discrete factor graphs to three classes—binary, pairwise, and planar—and their four intersections (see Figure 2). Simple reductions from discrete factor graphs with zeroes to binary pairwise factor graphs were shown to be impossible in general, but we exhibited a continuous spectral reduction requiring a linear number of evaluations of the reduced model (quadratic for the planar case), which overcomes this difficulty. Experiments then showed that models produced by the spectral reduction were more amenable to inference than those produced by the simple reduction. An open problem, suggested by our results, is to find a complexity measure with respect to which inference in binary-pairwise factor graphs can be proven to be more tractable than inference in general discrete factor graphs.

Acknowledgments

We are indebted to Martijn Leisink for his 3-wise soft-XOR construction and to Joris Mooij for pointing us to it. We also thank Tom Minka, Yee Whye Teh, Jonathan Yedidia, Tom Heskes, Wim Wiegerinck, Alex Ihler, and Dror Weitz for replying to queries, and three anonymous reviewers for thoughtful and useful feedback.

References

Arora, S., & Barak, B. (2009). *Computational complexity: A modern approach*. Cambridge: Cambridge University Press.

Barahona, F. (1982). On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, *15*, 3241–3253.

Bernstein, E., & Vazirani, U. (1997). Quantum complexity theory. *SIAM Journal on Computing*, *26*(5), 1411–1473.

Boros, E., & Hammer, P. (2002). Pseudo-Boolean optimization. *Discrete Applied Mathematics*, *123*(1–3), 155–225.

Braunstein, A., Mezard, M., & Zecchina, R. (2005). Survey propagation: An algorithm for satisfiability. *Random Structures and Algorithms*, *27*(2), 201–226.

Bulatov, A., & Grohe, M. (2005). The complexity of partition functions. *Theoretical Computer Science*, *348*(2), 148–186.

Castillo, E., Gutiérrez, J., & Hadi, A. (1997). *Expert systems and probabilistic network models*. New York: Springer-Verlag.

Chertkov, M., & Chernyak, V. (2006). Loop series for discrete statistical models on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, *2006*. p. 6009.

Chertkov, M., Chernyak, V., & Teodorescu, R. (2008). Belief propagation and loop series on planar graphs. *Journal of Statistical Mechanics: Theory and Experiment, 2008*. p. 05003.

Cook, S. (1971). The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing* (pp. 151–158). New York: ACM.

Cooper, G. (1990). The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, *42*(2–3), 393–405.

Dyer, M., Goldberg, L., & Jerrum, M. (2009). The complexity of weighted Boolean CSP. *SIAM Journal on Computing*, *38*(5), 1970–1986.

Eaton, F. (2011). A conditional game for comparing approximations. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. San Francisco: Morgan Kaufmann.

Fisher, M. (1966). On the dimer solution of planar Ising models. *Journal of Mathematical Physics*, *7*, 1776–1781.

Gallager, R. (1962). Low-density parity-check codes. *IRE Transactions on Information Theory*, *8*(1), 21–28.

Garey, M., & Johnson, D. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. New York: Freeman.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*(6), 721–741.

Globerson, A., & Jaakkola, T. (2007). Approximate inference using planar graph decomposition. In B. Schölkopf, J. Platt, & T. Hofmann (Eds.), *Advances in Neural Information Processing Systems, 19*. Cambridge, MA: MIT Press.

Goldberg, L., & Jerrum, M. (2008). Inapproximability of the Tutte polynomial. *Information and Computation*, *206*(7), 908–929.

Heskes, T., Albers, K., & Kappen, B. (2003). Approximate inference and constrained optimization. In *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence* (pp. 313–320). San Francisco: Morgan Kaufmann.

Ihler, A. (2007). Accuracy bounds for belief propagation. In *Proceedings of the 23rd Conference in Uncertainty in Artificial Intelligence* (pp. 183–190). Corvallis, OR: AOAI Press.

Ishikawa, H. (2009). Higher-order clique reduction in binary graph cut. In *Computer Vision and Pattern Recognition* (pp. 2993–3000). Piscataway, NJ: IEEE.

Jensen, F., Olesen, K., & Andersen, S. (1990). An algebra of Bayesian belief universes for knowledge-based systems. *Networks*, *20*(5), 637–659.

Jung, K., & Shah, D. (2006). Inference in binary pair-wise Markov random fields through self-avoiding walks. arXiv preprint cs/0610111.

Kasteleyn, P. (1961). The statistics of dimers on a lattice. *Physica*, *27*(12), 1209–1225.

Kasteleyn, P. (1963). Dimer statistics and phase transitions. *Journal of Mathematical Physics*, *4*, 287–293.

Kimura, K., Taki, K., & Kikō, S. (1991). *Time-homogeneous parallel annealing algorithm*. N. P. Institute for New Generation Computer Technology.

Knuth, D. (2008). *The art of computer programming, IV, Fascicle 0: Introduction to combinatorial algorithms and Boolean functions*. Reading, MA: Addison-Wesley.

Kschischang, F., Frey, B., & Loeliger, H. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, *47*(2), 498–519.

Kuratowski, K. (1930). Sur le problème des courbes gauches en topologie. *Fund. Math.*, *15*, 271–283.

Lichtenstein, D. (1982). Planar formulae & their uses. *SIAM Journal on Computing*, *11*, 329–343.

Minka, T. (2005). *Divergence measures and message passing* (Tech. Rep. MSR-TR-2005-173). Cambridge, UK: Microsoft Research.

Montanari, A., & Rizzo, T. (2005). How to compute loop corrections to the Bethe approximation. *Journal of Statistical Mechanics: Theory and Experiment*, *10*, P10011.

Mooij, J. (2010). libDAI 0.2.5: A free/open source C++ library for discrete approximate inference. *Journal of Machine Learning Research*, *11*, 2169–2173.

Mooij, J., & Kappen, H. (2005a). On the properties of the Bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2005*(11), P11012.

Mooij, J., & Kappen, H. (2005b). Validity estimates for loopy belief propagation on binary real-world networks. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances* *in neural information processing systems, 17* (pp. 945–952). Cambridge, MA: MIT Press.

Mooij, J., Wemmenhove, B., Kappen, H., & Rizzo, T. (2007). Loop corrected belief propagation. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*. San Francisco: Morgan Kaufmann.

Nair, C., & Tetali, P. (2007). The correlation decay (CD) tree and strong spatial mixing in multi-spin systems. arXiv preprint math/0701494.

Papadimitriou, C. (1994). *Computational complexity*. Reading, MA: Addison-Wesley.

Pearl, J. (1982). Reverend Bayes on inference engines: A distributed hierarchical approach. In *Proceedings of the AAAI National Conference on AI* (pp. 133–136). Menlo Park, CA: AAAI Press.

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

Pearl, J. (2000). *Causality: Models, reasoning, and inference*. Cambridge: Cambridge University Press.

Rosenberg, I. (1975). Reduction of bivalent maximization to the quadratic case. *Cahiers du Centre d'Etudes de Recherche Operationnelle*, *17*, 71–74.

Sanghavi, S., Shah, D., & Willsky, A. (2009). Message passing for maximum weight independent set. *IEEE Transactions on Information Theory*, *55*(11), 4822–4834.

Schaefer, T. (1978). The complexity of satisfiability problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing* (pp. 216–226). New York: ACM.

Selman, B., Levesque, H., & Mitchell, D. (1992). A new method for solving hard satisfiability problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 440–446). Menlo Park, CA: AAAI Press.

Sudderth, E., Wainwright, M., & Willsky, A. (2008). Loop series and Bethe variational bounds in attractive graphical models. In J. C. Platt, D. Koller, Y. Singer, & S. T. Roweis (Eds.), *Advances in neural information processing systems, 20* (pp. 1425–1432). Cambridge, MA: MIT Press.

Ungar, P. (1951). A theorem on planar graphs. *Journal of the London Mathematical Society*, *1*(4), 256–262.

Valiant, L. (1979a). The complexity of computing the permanent. *Theoretical Computer Science*, *8*(2), 189–201.

Valiant, L. (1979b). The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, *8*, 410–421.

Valiant, L. (2006). Accidental algorthims. In *47th Annual IEEE Symposium on Foundations of Computer Science* (pp. 509–517). Piscataway, NJ: IEEE.

Valiant, L. (2008). Holographic algorithms. *SIAM Journal on Computing*, *37*(5), 1565–1594.

Wagner, K. (1937). Über eine eigenschaft der ebenen komplexe. *Math. Ann.*, *144*, 570–590.

Wainwright, M., Jaakkola, T., & Willsky, A. (2002). A new class of upper bounds on the log partition function. In *Proceedings of the Eighteenth Conference Annual Conference on Uncertainty in Artificial Intelligence* (PP. 536–554). San Francisco: Morgan Kaufmann.

Watanabe, Y., & Fukumizu, K. (2011). New graph polynomials from the Bethe approximation of the Ising partition function. *Combinatorics, Probability and Computing*, *20*, 299–320.

Weitz, D. (2006). Counting independent sets up to the tree threshold. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing* (pp. 140–149). New York: ACM.

Welling, M., & Teh, Y. (2001). Belief optimization for binary networks: A stable alternative to loopy belief propagation. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence* (pp. 554–561). San Francisco: Morgan Kaufmann.

Welling, M., & Teh, Y. (2004). Linear response algorithms for approximate inference in graphical models. *Neural Computation*, *16*(1), 197–221.

Wiegerinck, W. (2000). Variational approximations between mean field theory and the Junction Tree Algorithm. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence* (pp. 626–633). San Francisco: Morgan Kaufmann.

Yedidia, J., Freeman, W., & Weiss, Y. (2001a). Generalized belief propagation. In T. K. Leen, T. G. Dietterich, & V. Tresp (Eds.), *Advances in neural information processing systems*, *13* (pp. 689–695). Cambridge, MA: MIT Press.

Yedidia, J., Freeman, W., & Weiss, Y. (2001b). Understanding belief propagation and its generalizations. *International Joint Conference on Artificial Intelligence*. San Francisco: Morgan Kaufmann.

Yuille, A. (2002). CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, *14*(7), 1691–1722.

Received September 14, 2011; accepted January 4, 2013.