Project Proposal, Redux

David Hall Alex Kantchelian

April 4, 2012

1 Introduction

Probabilistic graphical models (PGMs) are one of the cornerstones of modern AI, with applications ranging from medical diagnosis, to social network analysis, to robotics. PGMs provide the flexibility to model almost any real world phenomena in a coherent, simple way. However, exact computation in all but the most trivial PGMs is impossible. In the discrete case, inference in PGMs is #P-complete [Koller and Friedman, 2009], and for continuous variables it is possible to come up with incomputable models, even for "well behaved" distributions. [Ackerman et al., 2011]

Therefore, approximate inference regimes are one of the main cottage industries within the AI community, with several variants proposed each year. Broadly, they fall into three categories. The first is based on random sampling, especially Markov Chain Monte Carlo. MCMC is a randomized algorithm that is perhaps the easiest to implement, though it almost always the slowest in practice. Second are relaxation techniques based on linear or convex programming, but these are complex to implement and require significantly different implementations for each new PGM considered. Finally, there are message passing algorithms like mean field inference, belief propagation, and expectation propagation. We focus on message passing algorithms.

Unfortunately, all of these algorithms are tricky to parallelize, especially on arbitrarily structured graphs. Most of these algorithms can be parallelized to a certain extent given a graph coloring, but there is a limit to how much parallelism can be extracted from graph partitions in arbitrary graphs.

Message-passing algorithms typically work by decomposing the graph structure into subgraphs, and then (potentially) further approximating those parts with simpler distributions. Then, one proceeds one component at a time, optimizing it locally using the other components as a guide. (That is, these algorithms are basically a kind of coordinate ascent.) Of course, these updates are inherently non-sequential. Therefore, one possibility is to simply do these updates in parallel, using "old information" when making decisions [Cseke and Heskes, 2010]. However, this update schedule can lead to non-convergence, because oscillations between different local minima are possible.

We propose a new message passing algorithm called Concurrent Expectation Propagation (CEP) that allows for non-serialized concurrent updates in exactly this fashion. Our algorithm

should be able to work with arbitrary graph structures as well as both discrete and non-discrete distributions.

Our algorithm works by two closely related mechanisms. First, message passing algorithms optimize an approximation to the true PGM that is "as close as possible" to the original graph while remaining within a tractable set of decomposed distributions. One class of approximations—convex approximations—ensures that there is only one optimum. Thus, different pieces of the structure of the graph (which are operating largely independently on different processors) cannot update towards different local optima, which is one of the primary sources of oscillation.

Second, we introduce hysteresis into the updates. That is, when updating a component, one retains some fraction of the old approximation instead of completely replacing it. This lagging is actually one of the easiest ways to achieve convexity, meaning that we surprisingly only have to make one change to the algorithm.

We will implement CEP on hopefully three architectures: a traditional multi-core CPU, a high-end consumer graphics card, and—if possible—a specialized message passing architecture. We will compare our algorithm to a serialized version of the algorithm, as well as a non-parallelized traditional version of EP.

Finally, because of the hysteresis in our updates and the convexity of our approximation, it should be possible to run our algorithm where some updates are lost or corrupted. Thus, this scheme might be able to function in scenarios with improper synchronization or other flaky hardware. We therefore also propose to study the effect of how different (incorrect) locking schemes might lead to faster implementation with a minimal loss in accuracy.

2 Previous work

2.1 Distributed Message Passing

We are not the first to propose to parallelize a message passing algorithm. Cseke and Heskes [2010] proposed parallel EP, wherein they performed the unmodified Expectation Propagation updates in parallel. In their application, their scheme worked fine, but it is simple to come up with examples where naively doing parallel updates would immediately fail.

More closely related is the distributed belief propagation of Schwing et al. [2011]. They also relied on a convexification scheme, but their approach differs in two ways from ours. First, expectation propagation is a more powerful algorithm than belief propagation, being able to handle more complex decompositions as well as non-discrete distributions. Second, the decompositions they did use are geared towards a MapReduce architecture, with updates happening only occasionally.

2.2 Approximate and Concurrent Optimization

There has been a recent uptick of interest in the area of optimization under error-prone and other "incorrect" conditions, both from the machine learning community and the systems community.

On the systems side, Sartori et al. [2011] argued empirically for what they term "stochastic processors," which are classical processors that have been designed with the assumption that a certain amount of imprecision is acceptable in the functional units. In particular, they explain how classic algorithms, like sorting and bipartite matching, can be "robustified" when phrased in terms of optimization. They further provide experimental evidence that the resulting algorithms tolerate random bit flips in the floating point units. Similarly, Oboril et al. [2011] investigated variations on the conjugate gradient algorithm that were robust to random bit flips. All of these optimization algorithms they considered achieve their performance using convexity and hysteresis, the same two high level mechanisms we propose to use. Taking a more detailed view on voltage scaling, [Charkrapani et al., 2008] advocates for a non-uniform voltage scaling over the individual gates so that the most significant bits of the outputs incur the least error rates. [Palem et al., 2009] follow on this work and suggests using probabilistically correct circuits: designing inexact but highly energy-efficient logical circuits optimized for minimizing the expected error rate of the output given a known probability distribution on the inputs. While the authors show custom examples of such circuits, Lingamneni et al. [2011] introduce a general (approximation) algorithm for the so called "parsimonious" circuits.

On the machine learning side, the most relevant work is Hogwild! [Niu et al., 2011]. Essentially, the authors proposed and analyzed a variant of parallelized SGD where no locking was done in the updates to the weight vector x. This lock-free approach does not suffer much degradation over locking SGD, particularly for sparse problems where the majority of the components of the vector x are not in use at any particular time. And, of course, the resulting algorithm is much faster. Interestingly, they are still able to prove that their algorithm is guaranteed to converge at rates that are essentially no different from "correct" SGD algorithms.

References

Nathanael L. Ackerman, Cameron E. Freer, and Daniel M. Roy. Noncomputable conditional distributions. In *Proc. of the 26th Ann. Symp. on Logic in Comp. Sci.* IEEE Press, 2011.

LNB Charkrapani, KK Muntimadugu, A Lingamneni, J George, and KV Palem. Highly energy and performance efficient embedded computing through approximately correct arithmetic. In *CASES '08 Proceedings of the 2008 international conference on Compilers, architectures and synthesis for embedded systems*, pages 187–196, 2008. ISBN 9781605584690.

B. Cseke and T. Heskes. Improving posterior marginal approximations in latent gaussian models. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pages 121–128, 2010.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Avinash Lingamneni, Christian Enz, Krishna Palem, and Christian Piguet. Parsimonious Circuits for Error-Tolerant Minimization. In *PATMOS'11 Proceedings of the 21st international conference on Integrated circuit and system design*, pages 204–213, 2011.

Feng Niu, Benjamin Recht, Christopher Re, and Stephen J. Wright. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *CoRR*, 2011.

Fabian Oboril, Mehdi B. Tahoori, Vincent Heuveline, Dimitar Lukarski, and Jan-Philipp Weiss. Numerical defect correction as an algorithm-based fault tolerance technique for iterative solvers. In *Pacific Rim International Symposium on Dependable Computing*, pages 144–153, Los Alamitos, CA, USA, 2011. IEEE Computer Society.

Krishna V Palem, Zvi M Kedem, Lakshmi N.B. Chakrapani, Avinash Lingamneni, and Kirthi Krishna Muntimadugu. Sustaining Moore's Law in Embedded Computing through Probabilistic and Approximate Design: Retrospects and Prospects. In *Proceedings of the International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, pages 1–10, 2009. ISBN 9781605586267.

John Sartori, Joseph Sloan, and Rakesh Kumar. Stochastic computing: Embracing errors in architecture and design of processors and applications. In *International Conference on Compilers, Architecture, and Synthesis of Embedded System*, 2011.

Alexander G. Schwing, Tamir Hazan, Marc Pollefeys, and Raquel Urtasun. Distributed message passing for large scale graphical models. In *CVPR*, pages 1833–1840, 2011.