Asynchronous Markov Chain Monte Carlo and Gibbs Sampling

Alexander Terenin (Petuum, Inc. and Imperial College London) · Joint work with Dan Simpson (University of Toronto), David Draper (University of California, Santa Cruz), and Eric P. Xing (Petuum, Inc. and Carnegie Mellon University)

Abstract

Markov Chain Monte Carlo (MCMC) methods such as Gibbs sampling are widely used for training Bayesian models. Parallelizing these methods for faster computation and distributed deployment is an area of active research. We present a theoretical framework for convergence analysis of MCMC methods executed asynchronously.

Markov Chain Monte Carlo

$X_0 \xrightarrow{P} X_1 \xrightarrow{P} X_2 \xrightarrow{P} \ldots \xrightarrow{P} X_{n+1} \xrightarrow{P} X_{n+2}$

Convergence Theory

Algorithm 1. For a set of $n$ workers, do the following in parallel.
1. Read a value $x_k$ from shared memory.
2. Update the value by computing $x_{k+1}$ using any MCMC method.
3. Write $x_{k+1}$ to shared memory.

Algorithm 1 is not a Markov chain.

Asynchronous MCMC with shared memory

Algorithm 2. For a set of $m$ workers, repeat without synchronization.
1. Update the local value by computing $x_{k+1}$ using an MCMC method.
2. Transmit $x_{k+1}$ to all other workers, process any received messages.

Algorithm 2 does not converge in general.

Distributed Convergence Theory

To satisfy this condition, we use the Metropolis-Hastings method.

Asynchronous Gibbs Sampling on a computer

Related Work

Asynchronous optimization.

Recent analysis: [5], industrial deployment: [2].

Asynchronous Monte Carlo.

Recent analysis: [1, 3, 7], industrial deployment: [4].

References


Acknowledgments

We are grateful to Petuum, Inc. for funding this research and providing computational resources. We are grateful to eBay, Inc. for funding previous research, and providing computational resources used in that work. Membership on this list does not imply agreement with the ideas given here, nor responsibility for any errors may be present.