Asynchronous Gibbs Sampling

Alexander Terenin (eBay, Inc. and UC Santa Cruz)
Dan Simpson (University of Bath, UK) • David Draper (UC Santa Cruz)

Abstract
Bayesian statistical models are attractive in the Big Data setting, but Markov Chain Monte Carlo methods – the cornerstone of modern Bayesian computation – do not extend easily to the cluster setting, where the data is too big to fit on one machine. We present a novel scheme to parallelize MCMC with no synchronization or locking, avoiding typical performance bottlenecks, particularly in settings where the number of model parameters grows with data size.

Big Data: the problem
Data: too big to fit on one machine
• Lives on a Hadoop cluster: fault-tolerant distributed storage
  – Big Data ⇔ Big Hardware
• Computational cost of 1 data point ≈ cost of all N points
  – Data must be operated on in parallel
• Need parallel algorithms that scale superlinearly
• Model dimensionality often grows with N

Previous Work on Scalable Bayesian Inference
• Data that fits on one machine
  – Approximate Methods: Variational Bayes [2], ABC [6]
  – Exact Methods: Langevin Diffusion based methods [4]
• Data too big for one machine
  – Consensus Monte Carlo [5]: for models where dimensionality doesn’t grow with N (exact for some models)
  – Hogwild [3]: asynchronous stochastic gradient descent, point estimation only, widely used in machine learning

The Algorithm
1. Start with a latent variable model, partition data among workers
2. Run Gibbs steps on each worker
  • Condition on most recent local variables
  • Condition on most recent known non-local variables, which may be out of date in the cluster as a whole
  • Draw samples, transmit to other workers
  • Never stop, never synchronize, never wait

Implementation
• Highly non-trivial: not expressible in the MapReduce paradigm (typical high-level parallel computation framework used in Hadoop)
• Expressible in shared memory and actor models of parallelism
• Written in Scala, a modern language well-suited to parallel use cases
• Code available via eBay Software Foundation on GitHub

Convergence
Two kinds of algorithms: different communication between workers
• Exact Asynchronous Gibbs: apply Metropolis-Hastings correction
• Approximate Asynchronous Gibbs: accept all updates

Theorem: Exact Asynchronous Gibbs converges
• Intuitive sketch of proof
  1. Instant communication ⇒ exact algorithm converges
     – Define a Markov chain that (1) selects a worker, (2) selects a full conditional, (3) proposes a new state from that full conditional at every worker, (4) performs a Metropolis step on each worker
     – Proof via detailed balance, workers accept/reject independently
  2. Exact algorithm converges ⇒ asynchronous convergence
     – Proof via result on convergence of asynchronous algorithms

Approximate Asynchronous Gibbs: Noisy Monte Carlo [1]
• Replace Metropolis-Hastings ratio with biased estimator – 1
• Perform diagnostic check at runtime to ensure bias is small

Future Work
• Better understanding of Noisy Monte Carlo [1] approximations
• Implement algorithm in MPI for use on traditional supercomputers

Results
Gaussian Process Regression: 71,500 latent variables (toy problem)
• Correct answer in 20 minutes on 143-core cluster
Large hierarchical model: 1,000,000 latent variables (real-world)
• Same answer as sequential-scan Gibbs (up to Monte Carlo error)

<table>
<thead>
<tr>
<th>Standard Gibbs Sampler</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-core machine, high memory, problem-specific parallelization</td>
</tr>
<tr>
<td>Asynchronous Gibbs</td>
</tr>
<tr>
<td>160-core cluster</td>
</tr>
</tbody>
</table>

References

Acknowledgments
We are grateful to UC Santa Cruz and the Baskin School of Engineering for funding this research, and to eBay, Inc. for funding and providing computational resources.