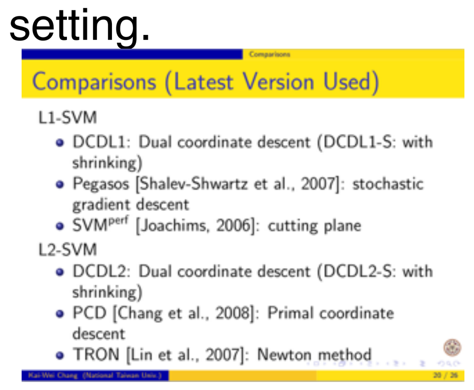
Parallel ASynchronous Greedy dual Coordinate Descent

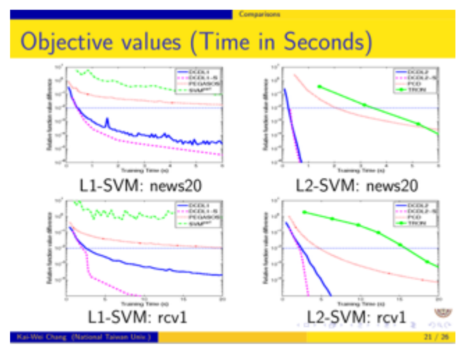
THE UNIVERSITY OF TEXAS AT AUSTIN
College of Natural Sciences

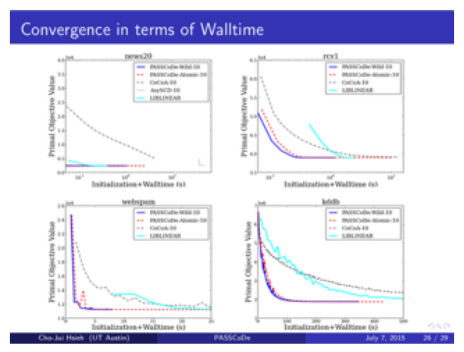
Anna Panyu Peng, Department of Mathematics and Computer Science /CNS, ICES panyuanna@gmail.com

Background

Dual coordinate descent method is a classic and effective optimization technique for solving empirical risk minimization problems such as Linear SVM where number of data and features are large i.e. text classification[1]. Several papers have studied specific implementations of the method i.e. stochastic DCD[2] or adaptive DCD where parallel techniques are exploited to reduce time complexity. Few have studied greedy DCD back then on a single machine. As I found out that when solving kernel SVM, computing kernel matrix is too slow so is convergence rate of objective function, I was motivated to parallelize greedy DCD in a multi-core shared memory setting.





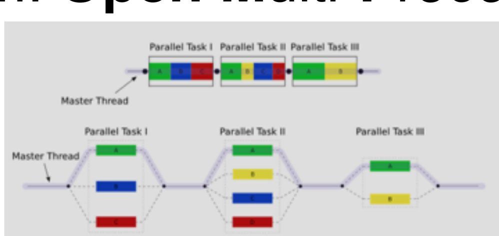


Research Questions

I am parallelizing greedy DCD for kernel SVM. Since DCD embeds the sequential idea in it, when deliberately paralleled, several obstacles confronted:

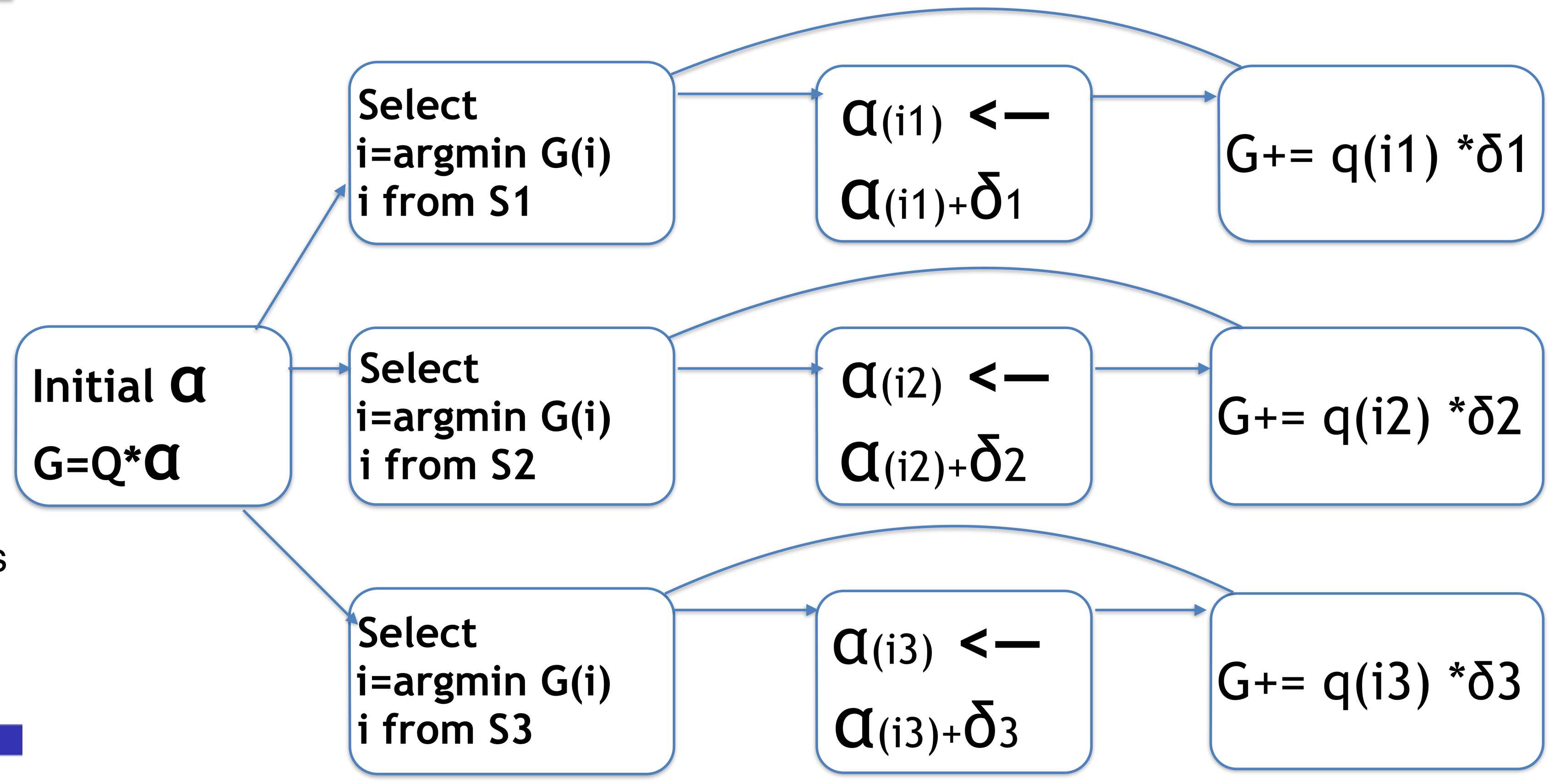
- How to partition gradient of objective function
- How to avoid conflict write
- How to load dense kernel matrix
- How to prove convergence of new algorithm

Methods and Materials

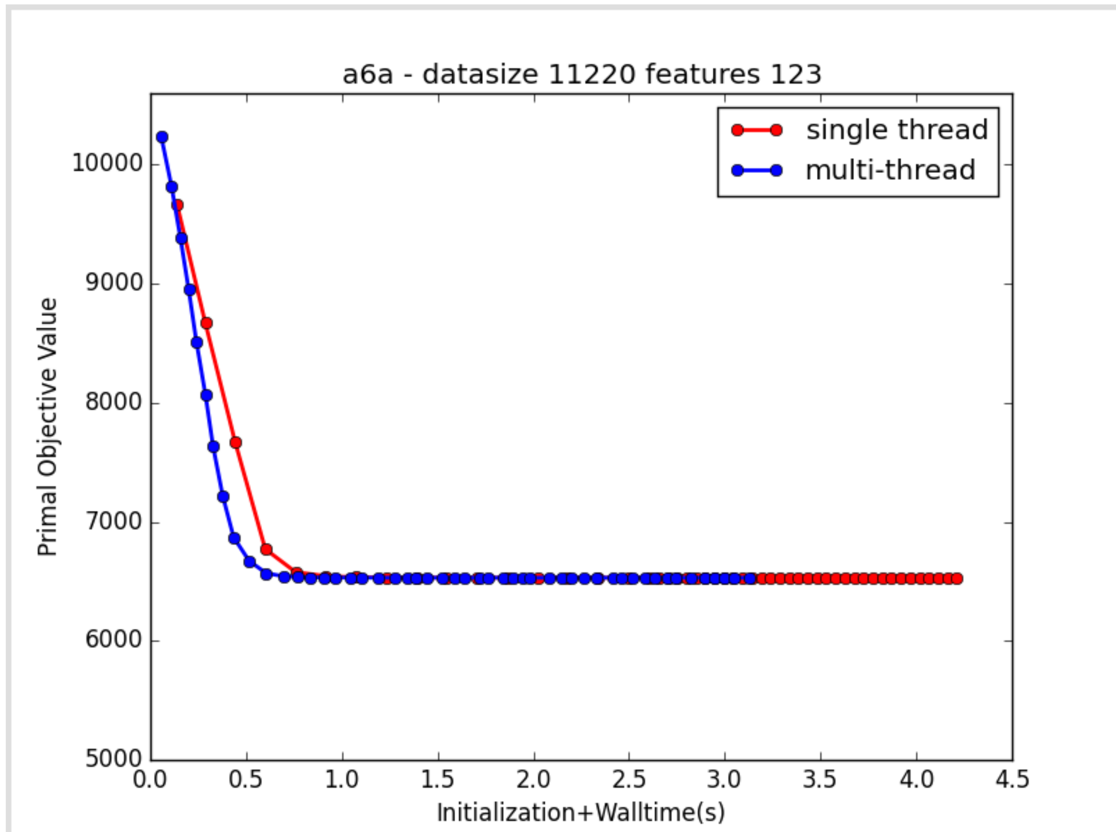
- Parallel Paradigm: **Open Multi-Processing** in C++
- 
- Partition method: Even partition of the vector of gradient of objective function into number of threads
- Kernel selection: Gaussian/RBF kernel: $K(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$
- Atomic: Apply atomic operation for gradient update:

```
                  // update G (gradient)
                  for(int k=0;k<active_size;k++)
                  {
                          double value = Q_i[k]*delta_alpha_i;
#pragma omp atomic
                          G[k] += value;
```

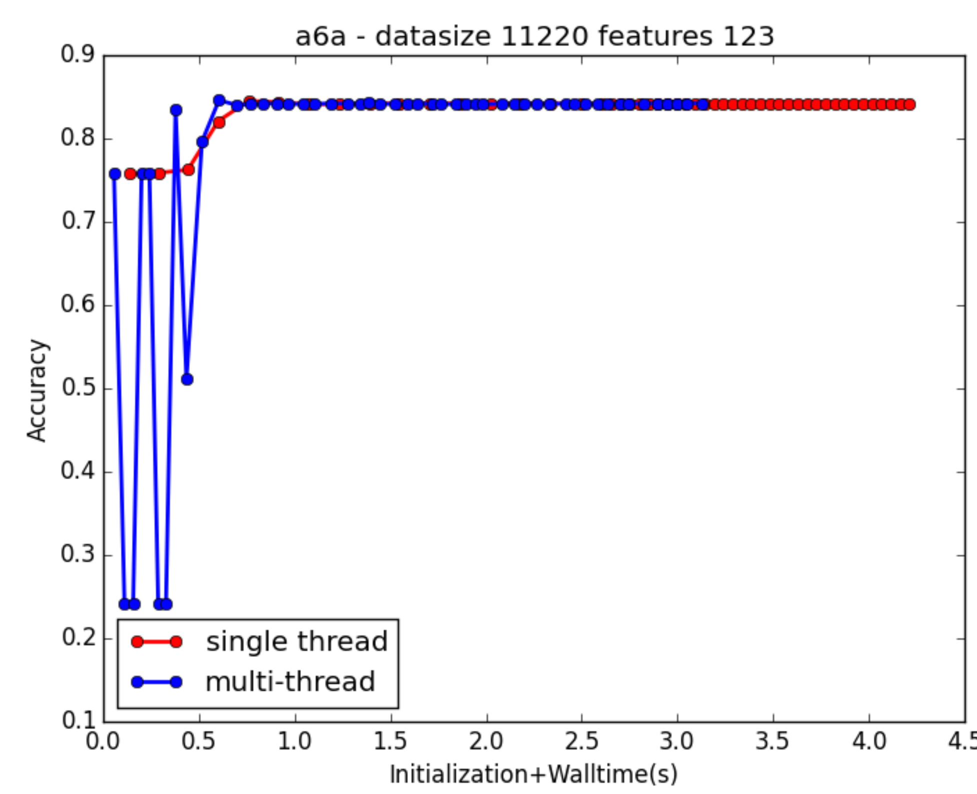
Algorithm



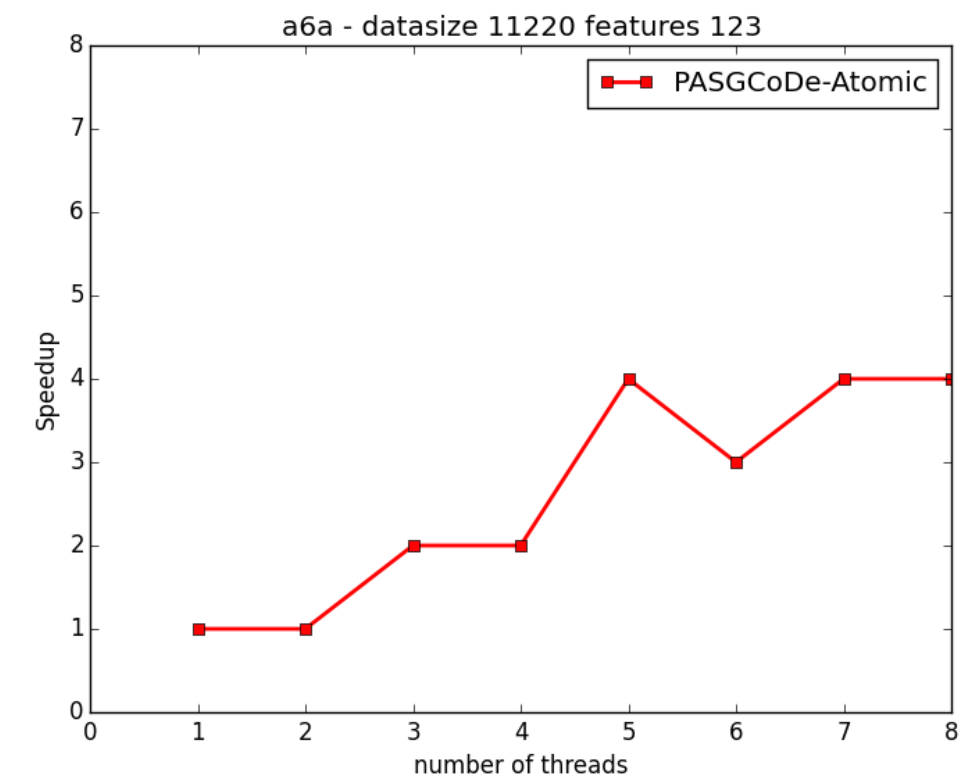
Time vs. Objective Value



Time vs. Accuracy



of Threads vs. Speedup



Future Directions

Preliminary outcomes point out further work:

- As observed from the results, four-thread convergence rate is only approximately 10% faster than single thread, which is incoherent with speedup. So I need to accelerate the convergence rate of the parallel method
- To provide theoretical guarantee and analysis of our fast algorithm

Acknowledgments

Inderjit S. Dhillon, **Director**, Center for Big Data Analytics, Department of Computer Science, University of Texas at Austin

***Cho-Jui Hsieh*, Professor,** UC Davis; Ph.D, University of Texas at Austin

*The **Institute for Computational Engineering and Sciences** (ICES),* University of Texas at Austin

Department of Mathematics and Computer Science/ CNS, *University of Texas at Austin*

References

1. Kai-Wei Chang, C.-J. Hsieh, C.-J. Lin, S. S. Keerthi, and S. Sundararajan *A Dual Coordinate Descent Method for Large-scale Linear SVM.* ICML, 2008
2. Cho-Jui Hsieh, Hsiang-Fu Yu, Inderjit S. Dhillon *PASSCoDe: Parallel ASynchronous Stochastic dual Co-ordinate Descent*. ICML, 2015