Supplementary Materials for "Naive Parallelization of Coordinate Descent Methods and an Application on Multi-core L1-regularized Classification"

Yong Zhuang*
Carnegie Mellon University
yong.zhuang22@gmail.com

Yuchin Juan†
Criteo Research
yc.juan@criteo.com

Guo-Xun Yuan
Facebook, Inc.
gxyzuan@gmail.com

Chih-Jen Lin
National Taiwan University
cjlin@csie.ntu.edu.tw

ACM Reference Format:
Yong Zhuang, Yuchin Juan, Guo-Xun Yuan, and Chih-Jen Lin. 2018. Supplementary Materials for "Naive Parallelization of Coordinate Descent Methods and an Application on Multi-core L1-regularized Classification". In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3269206.3271687

A LINE SEARCH IN BUNDLE CDN

In this section, we explain a line search trick that can be applied in Naive CDN but cannot be applied in Bundle CDN. In addition, we demonstrate the impact of the line search trick.

In (13), the computational cost mainly comes from evaluating

$$\Delta_j(\beta^t d) \equiv L(\boldsymbol{w} + \beta^t d \boldsymbol{e}_j) - L(\boldsymbol{w}),$$

which costs $O(\text{nnz}_j)$ and nnz_j is the number of non-zeros in feature j. In CDN, for both SVM and LR, an upper bound of $\Delta_j(\beta^t d)$ is calculated during the training process. The cost of computing this upper bound $\bar{\Delta}_j(\beta^t d)$ is just $O(1)$. The details can be found in Eq. (40) for SVM and Eq. (49) - (54) for LR in [2]. During the line search process, before computing $\Delta_j(\beta^t d)$, CDN first checks if $\bar{\Delta}_j(\beta^t d)$ satisfies (13). If satisfied, then we can skip the entire line search process, making the cost of line search drop from $O(\text{nnz}_j)$ to $O(1)$. (If not satisfied, then we still need to run the standard line search.)

This trick, designed for one-variable problems, may not be applicable for multi-variable problems like (16). Therefore, if we check the code published by [1],[1] the implementation of this trick was commented out. They also remove the trick in the single-thread CDN to measure the speedup, meaning that they are comparing with a slower version of CDN. However, we believe in practice people would be more interested in the speedup against the original version of CDN. In Figure I, we compare the following setting on kdd2010-a.

- Naive CDN
- Naive CDN without the trick
- Bundle CDN with the best bundle size

The experiment shows that the trick indeed plays an important role for this data set.

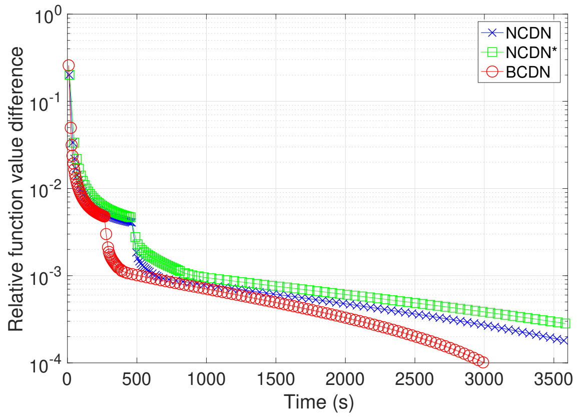

Figure I: Comparison between Naive CDN (NCDN), NCDN without the line-search trick (denoted as NCDN* in the figure), and Bundle CDN (BCDN) with the bundle size 29,500. 16 threads are used for all methods. The data set used is kdd2010-a.

*Most of the work was done during the internship in Criteo Research.
†This author contributes equally with Yong Zhuang.
[1]https://github.com/bianan/ParallelCDN

REFERENCES

[1] Yatao Bian, Xiong Li, Mingqi Cao, and Yuncai Liu. 2013. Bundle CDN: a highly parallelized approach for large-scale l1-regularized logistic regression. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/ PKDD).*
[2] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research* 9 (2008), 1871–1874. http://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf