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
\[
\Delta_j(\beta^t d) \equiv L(w + \beta^t d e_j) - L(w),
\]
which costs \(O(mn_x j)\) and \(m n_x j\) 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 \(\Delta_j(\beta^t d)\) 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 \(\Delta_j(\beta^t d)\) satisfies (13). If satisfied, then we can skip the entire line search process, making the cost of line search drop from \(O(mn_x j)\) 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], 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 CDN 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.

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

REFERENCES


Nearly all the work was done during the internship in Criteo Research.

This author contributes equally with Yong Zhuang.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

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

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

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

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

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

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