D-Cube: Dense-Block Detection in Terabyte-Scale Tensors (Supplementary Document)

Kijung Shin, Bryan Hooi, Jisu Kim, Christos Faloutsos
School of Computer Science, Carnegie Mellon University
Pittsburgh, PA, USA
{kijungs, christos}@cs.cmu.edu, {bhooi, jisuk1}@andrew.cmu.edu

ABSTRACT

In this supplementary document, we provide implementation details, descriptions of datasets, and additional experimental results, all of which supplement the main paper [10].

1. MAPREDUCE IMPLEMENTATION

In this section, we present how D-CUBE can be implemented on MAPREDUCE, supplementing Section 3.3 of the main paper. Specifically, we describe two MAPREDUCE algorithms corresponding to steps of D-CUBE accessing tuples, which are stored in a distributed file system.

Computing Mass. Line 5 of Algorithm 1 can be performed by the following algorithm, where the amount of shuffled data can be reduced by combining the intermediate results within each mapper.

- Map-stage: Take a tuple t (i.e., $<t[A_1], ..., t[A_N], t[X]>$) and emit $<0, t[X]>$.
- Combine-stage/Reduce-stage: Take $<0, values>$ and emit $<0, sum(values)>$.

The value of the final output corresponds to the mass of the input relation.

Computing Attribute-value Sets. Line 2 of Algorithm 1 can be performed by the following algorithm, where the amount of shuffled data can be reduced by combining the intermediate results within each mapper.

- Map-stage: Take a tuple t (i.e., $<t[A_1], ..., t[A_N], t[X]>$) and emit N key/value pairs, $\{<(n, t[A_n]), 0>\}_{n=1}^{N}$.
- Combine-stage/Reduce-stage: Take $<(n, a), values>$ and emit $<(n, a), 0>$.

Each line $<(n, a), 0>$ of the final output indicates that a is a member of $\mathcal{R}_n$.

2. REAL-WORLD DATASETS

In this section, we describe the real-world tensor datasets used in the main paper. They are categorized into four groups: (1) Rating data (Yelp, Android, Netflix, and YahooM.), (2) Wikipedia revision history (KoWiki and EnWiki), (3) Temporal social networks (Youtube and SMS), and (4) TCP dumps (DARPA and AirForce). Some statistics of these datasets are listed in Table 3 in the main paper.

- Yelp [2]: In Yelp(user,business,date,rating,1), each tuple (u,b,d,r,1) indicates that user u gave business b rating r on date d on Yelp[1], a review site. This dataset was used for Yelp Dataset Challenge [2].
- Android [7]: In Android(user,app,timestamp,rating,1), each tuple (u,a,t,r,1) indicates that user u gave Android app a rating r at timestamp t (in hours) on Amazon, an online store.
- Netflix [3]: In Netflix(user,movie,date,rating,1), each tuple (u,m,d,r,1) indicates that user u gave movie m rating r on date d on Netflix, a movie rental and streaming service. This dataset was used for Netflix Prize[2].
- YahooM. [4]: In YahooM.(user,item,timestamp,rating,1), each tuple (u,i,t,r,1) indicates that user u gave musical item i rating r at timestamp t (in hours) on Yahoo! Music. This dataset was used for KDD Cup 2011[3].
- KoWiki [9]: In KoWiki(user,page,timestamp,#revisions), each tuple (u,p,t,r) indicates that user u revised page p r times at timestamp t (in hour) in Korean Wikipedia, a crowd-sourcing online encyclopedia. This dataset is the snapshot on February 4, 2016.
- EnWiki [9]: This dataset has the same schema with KoWiki Dataset, but data are from English Wikipedia. This dataset is the snapshot on February 4, 2016.
- Youtube [8]: In Youtube(source,destination,date,1), each tuple (s,d,t,1) indicates that a link from user s to user d was observed on date t on Youtube, a video-sharing website.
- SMS: In SMS(source,destination,timestamp,#messages), each tuple (s,d,t,m) indicates that user s sent m text messages to user d at timestamp t (in hour) in a large Asian city.
- DARPA [6] : In DARPA(source IP,destination IP, timestamp,#connections), each tuple (s,d,t,c) indicates that c connections were made from IP s to IP d at timestamp t (in minutes). This dataset was collected by the Cyber Systems and Technology Group in 1998.
- AirForce [1] : In AirForce(protocol,service,src bytes, dst bytes,flag,host count,host count,#connections), each

[1]http://www.yelp.com
[2]http://netflixprize.com/
[3]http://www.kdd.org/kdd2011/kddcup.shtml

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

tuple indicates that given number of connections have the the given dimension attribute values. This dataset was used for KDD Cup 1999 [1]. The description of each attribute is as follows:

- protocol: type of protocol (e.g. tcp and udp)
- service: network service on destination (e.g., http and telnet)
- src bytes: number of data bytes sent from source to destination
- dst bytes: number of data bytes sent from destination to source
- flag: normal or error status
- host count: number of connections made to the same host in the past two seconds
- srv count: number of connections made to the same service in the past two seconds
- #connections: number of connections with the given dimension attribute values.

3. ADDITIONAL EXPERIMENTS

We design additional experiments to answer the following questions:

- **Q5. Scalability with k**: How does D-Cube scale with the number of dense blocks it aims to find?
- **Q6. Speed and Accuracy**: How rapidly and accurately D-Cube detects dense blocks when ρ_{geo} or ρ_{ari} is used as the density measure?

Experimental settings were the same as in the main paper.

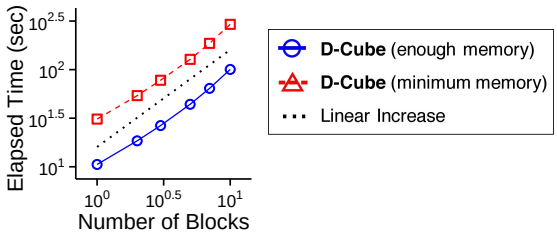

Figure 7: **D-Cube scales linearly with the number of blocks (i.e., k).**

3.1 Q5 Scalability with k

We show that D-Cube scales linearly with the number of blocks that it finds (i.e., k). We measure the running time of D-Cube on a synthetic tensor with 100 million tuples, and 3 dimension attributes each of whose cardinality is 100 thousands, as we increase k from 1 to 10. As seen in Figure 7, D-Cube scales linearly with k. This linear scalability of D-Cube held both with enough memory (blue solid line) to store all tuples and with minimum memory (red dashed line) to barely meet requirements. This is also consistent with our theoretical analysis on the time complexity of D-Cube (Theorem 1 in the main paper).

3.2 Q6: Speed and Accuracy

We compare how rapidly and accurately each method detects dense blocks in real-world tensors when ρ_{geo} or ρ_{ari} is used as the density measure. The same experiment with ρ_{susp} can be found in the main paper. Although CrossSpot [5] was originally designed to maximize ρ_{susp}, we used its variants that directly maximize the density metric compared in each experiment. We measured the wall-clock time (average over three runs) taken for detecting three blocks by each method, and measured the maximum density of the three blocks found by each method. Note that the serial version of D-Cube was used, and each dataset was cached in memory by D-Cube since they fit in memory (see Section 3.1.4 of the main paper).

Figure 8 shows the result with ρ_{geo}, where we used the maximum density policy for D-Cube (see Section 3.1.3 of the main paper), which consistently resulted in denser blocks than the maximum cardinality policy. D-Cube provided the best trade-off between speed and accuracy. D-Cube was up to **5× faster** than the second fastest method, and in terms of accuracy (i.e., density of the found blocks), D-Cube consistently spotted high-density blocks, while the accuracy of the other methods varied on data. Especially, D-Cube detected the densest blocks in SMS and DARPA Datasets. Compared with CPD, D-Cube spotted up to **6× denser** blocks.

Figure 9 shows the results with ρ_{ari}. Here we consider both the maximum cardinality policy and the maximum density policy, which gave comparable performances. D-Cube gave the best trade-off between speed and accuracy. Specifically, it was up to 5 **times faster** the second fastest method, while giving the similarly dense blocks with other methods except CPD. Compared with CPD, D-Cube spotted up to **48× denser** blocks.

Although MAF does not appear in Figures 8 and 9, it consistently provided sparser blocks than CPD with similar speed.

4. REFERENCES

[1] Kdd cup 1999 data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[2] Yelp dataset challenge. https://www.yelp.com/dataset_challenge.

[3] J. Bennett and S. Lanning. The netflix prize. In *KDD Cup*, 2007.

[4] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The yahoo! music dataset and kdd-cup'11. In *KDD Cup*, 2012.

[5] M. Jiang, A. Beutel, P. Cui, B. Hooi, S. Yang, and C. Faloutsos. A general suspiciousness metric for dense blocks in multimodal data. In *ICDM*, 2015.

[6] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, et al. Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. In *DISCEX*, 2000.

[7] J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *KDD*, 2015.

[8] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and Analysis of Online Social Networks. In *IMC*, 2007.

[9] K. Shin, B. Hooi, and C. Faloutsos. M-zoom: Fast dense-block detection in tensors with quality guarantees. In *ECML/PKDD*, 2016.

[10] K. Shin, B. Hooi, J. Kim, and C. Faloutsos. D-cube: Dense-block detection in terabyte-scale tensors. In *WSDM*. ACM, 2017.

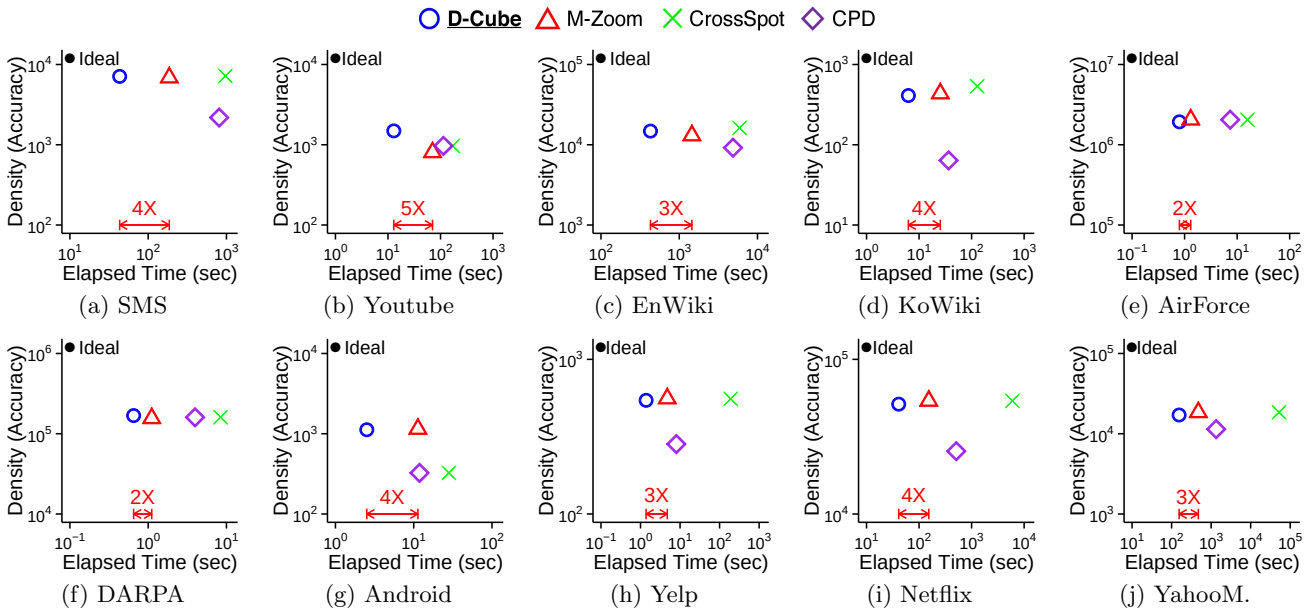

Figure 8: **D-Cube achieves both speed and accuracy in terms of ρ_{geo}.** In each plot, points represent the speed of different methods and the highest density (ρ_{geo}) of three blocks found by the methods. Upper-left region indicates better performance. D-Cube gave the best trade-off between speed and density. Specifically, D-Cube was up to **5× faster** than M-Zoom, which was the second best method, with similarly dense blocks.

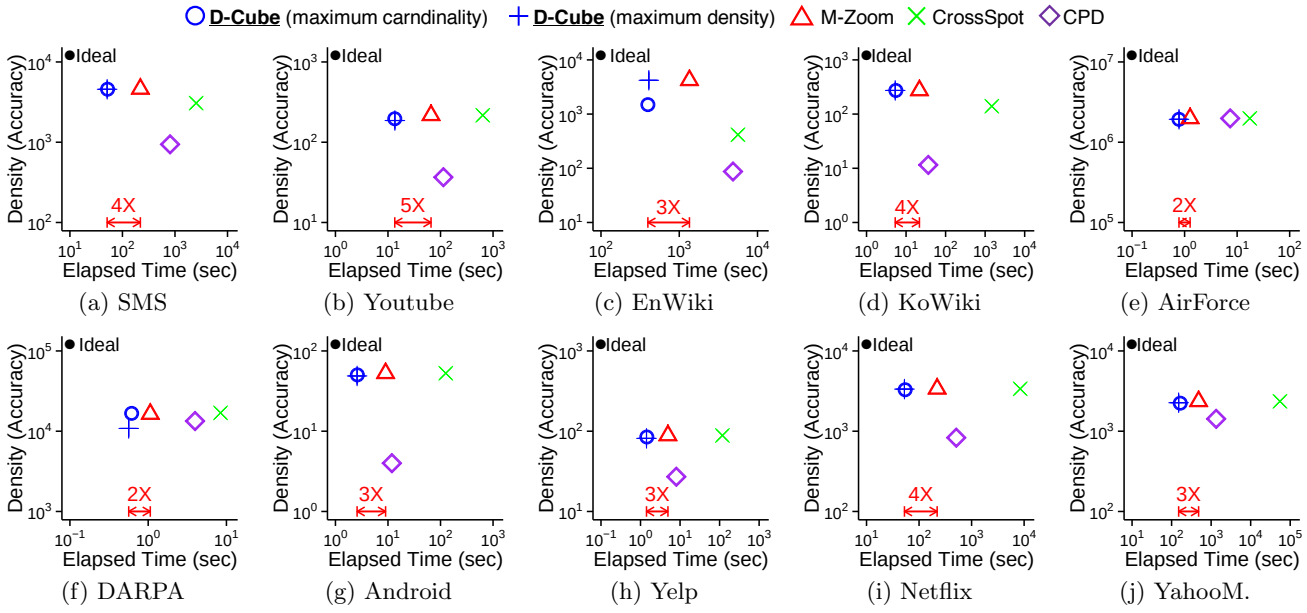

Figure 9: **D-Cube achieves both speed and accuracy in terms of ρ_{ari}.** In each plot, points represent the speed of different methods and the highest density (ρ_{ari}) of three blocks found by the methods. Upper-left region indicates better performance. D-Cube gave the best trade-off between speed and density. Specifically, D-Cube was up to **5× faster** than M-Zoom, which was the second best method, with similarly dense blocks.