Marquette University

e-Publications@Marquette

Mathematics, Statistics and Computer Science Faculty Research and Publications

Mathematics, Statistics and Computer Science, Department of (- 2019)

7-15-2010

FastMEDUSA: A Parallelized Tool to Infer Gene Regulatory Networks

Serdar Bozdag
Marquette University, serdar.bozdag@marquette.edu

Aiguo Li
National Institutes of Health

Stefan Wuchty
National Institutes of Health

Howard A. Fine
National Institutes of Health

Follow this and additional works at: https://epublications.marquette.edu/mscs_fac

 Part of the Computer Sciences Commons, Mathematics Commons, and the Statistics and Probability Commons

Recommended Citation

Bozdag, Serdar; Li, Aiguo; Wuchty, Stefan; and Fine, Howard A., "FastMEDUSA: A Parallelized Tool to Infer Gene Regulatory Networks" (2010). *Mathematics, Statistics and Computer Science Faculty Research and Publications*. 72.
https://epublications.marquette.edu/mscs_fac/72

BIOINFORMATICS APPLICATIONS NOTE Vol. 26 no. 14 2010, pages 1792–1793
doi:10.1093/bioinformatics/btq275

Systems biology

Advance Access publication May 30, 2010

FastMEDUSA: a parallelized tool to infer gene regulatory networks

Serdar Bozdag[1], Aiguo Li[1], Stefan Wuchty[1,2] and Howard A. Fine[1,*]

[1]Neuro-Oncology Branch, National Cancer Institute, National Institute of Neurological Diseases and Stroke
[2]National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20892, USA

Associate Editor: Olga Troyanskaya

ABSTRACT

Motivation: In order to construct gene regulatory networks of higher organisms from gene expression and promoter sequence data efficiently, we developed FastMEDUSA. In this parallelized version of the regulatory network-modeling tool MEDUSA, expression and sequence data are shared among a user-defined number of processors on a single multi-core machine or cluster. Our results show that FastMEDUSA allows a more efficient utilization of computational resources. While the determination of a regulatory network of brain tumor in *Homo sapiens* takes 12 days with MEDUSA, FastMEDUSA obtained the same results in 6 h by utilizing 100 processors.

Availability: Source code and documentation of FastMEDUSA are available at https://wiki.nci.nih.gov/display/NOBbioinf/FastMEDUSA

Contact: hfine@mail.nih.gov

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on January 19, 2010; revised on April 21, 2010; accepted on May 21, 2010

1 INTRODUCTION

Among the numerous ways to determine gene regulatory networks of higher organism (Bussemaker *et al*., 2001; Elemento *et al*., 2007; Segal *et al*., 2003), MEDUSA is a well-known and powerful computational tool (Kundaje *et al*., 2008). Utilizing a boosting algorithm (Freund and Shapire, 1997), MEDUSA models promoter sequences and gene expression data from various experimental conditions. Providing a global predictive model of condition-specific expression states of target genes, MEDUSA has been successfully used to discover novel regulators in oxygen and heme regulatory networks in *Saccharomyces cerevisiae* (Kundaje *et al*., 2008).

Although MEDUSA is a powerful computational tool for inferring gene regulatory networks, it has certain limitations. MEDUSA requires the commercially available software package, MATLAB. In addition, the execution time of MEDUSA increases dramatically with larger sizes of input data sets. For example, MEDUSA needed >4 weeks for the modeling and analysis of about 7000 genes using 1000 iterations. Parallel computing has been used to solve the latter problem for several bioinformatics applications to date (Xiaohong *et al*., 2009). Here we introduce FastMEDUSA, a fast, parallelized, open source implementation of MEDUSA in C++ that uses freely available libraries. Our benchmark results showed that FastMEDUSA allows to model gene sets about 40 times faster than the original implementation of MEDUSA, utilizing 100 processors.

*To whom correspondence should be addressed.

2 METHODS

2.1 Algorithm

In MEDUSA, a training set consists of a matrix of discretized expression pairs (g, e) of a gene g in experiment e. MEDUSA applies boosting on the training set to iteratively build an alternating decision tree (ADT), which consists of weighted weak classifiers. A weak classifier, which consists of a motif m, regulator r and expression state s of r, classifies a set of examples (g, e) where motif m is present in the promoter of gene g and the expression state of regulator r is s in experiment e. A weak classifier with maximum classifier score is added to the ADT iteratively. Each element in the training set is reweighted for the next iteration where misclassified elements get a higher weight than accurately classified elements. In a final ADT, the total weight (i.e. prediction score) of the weak classifiers that satisfy an element in the test set represents the overall prediction for this element.

Initially, MEDUSA computes all motifs (i.e. k-mers and optionally dimers) from promoter sequences of genes (Fig. 1a). At each iteration, first, optimum weak classifier and its optimum position in the ADT are computed based on a classifier score. Subsequently, the algorithm generates probabilistic motifs by clustering the top n motifs that optimize classifier score, where n is a user-defined parameter. After computing pairwise distances, motifs are clustered iteratively. In each clustering step, classifier score of the clustered motif is computed by scanning the motif against the promoter sequences. If a clustered motif has a higher score than the motif in the weak classifier has, the motif is replaced by this clustered motif. The final weak classifier is added to the ADT, and each element in the training set is reweighted for the next iteration. For algorithmic details of MEDUSA, see Kundaje *et al*. (2007) and Middendorf *et al*. (2005).

FastMEDUSA, a parallelized implementation of the MEDUSA algorithm, designates one of the processors as the root processor to manage inter-processor communication and compute global results. The root processor assigns a unique subtree of the ADT to each remaining processor, which computes optimum weak classifier and its position for the assigned sub-ADT and sends results back to the root processor. In addition, the root processor partitions genes among remaining processors equally to parallelize gene-wise operations. For instance, each processor scans the clustered motif against promoter sequences of the assigned genes in each clustering step in parallel. Parallelizing reweighting elements, each processor reweights only the assigned genes. Summarizing all parallelized steps, we present a flowchart of FastMEDUSA in Figure 1a.

2.2 Implementation

FastMEDUSA was implemented in C++, utilizing the message passing interface (MPI) implementation MPICH2 for inter-process communication (http://www.mcs.anl.gov/research/projects/mpich2/) and GotoBLAS2

 © The Author 2010. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oxfordjournals.org

Downloaded from http://bioinformatics.oxfordjournals.org/ at marquette university Raynor Memorial Libraries on January 24, 2013

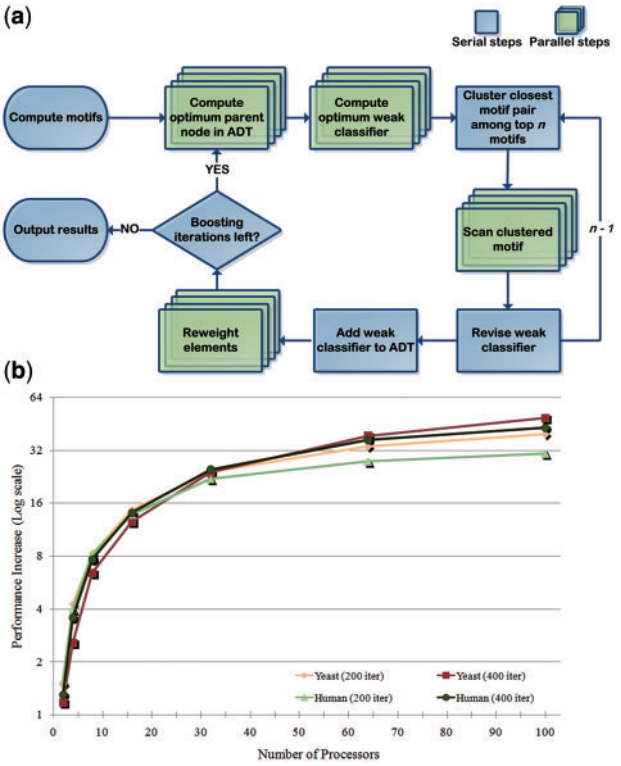

Fig. 1. In (**a**), we show the flowchart of FastMEDUSA, highlighting parallelized steps. (**b**) Testing the performance, we ran FastMEDUSA and MEDUSA on yeast and human data sets for 200 and 400 iterations. We observed that FastMEDUSA consistently outperforms MEDUSA. The graphs and raw data of output ADTs are reported in Supplementary Figures S1–S8 and Supplementary Tables S1–S8.

library (http://www.tacc.utexas.edu/tacc-projects/#blas) for performing matrix multiplication. FastMEDUSA runs on Mac OS X and Linux.

2.3 Benchmarking

We tested the performance of FastMEDUSA and MEDUSA utilizing discretized expression and promoter sequences of 6291 genes, 501 regulators in 18 samples of yeast (Kundaje *et al*., 2008) and 14 819 genes, 1268 regulators in 159 samples of human brain tumor (glioma) data set (Li *et al*., 2009). Human regulators and promoter sequences were obtained from MatBase (Cartharius *et al*., 2005) and Gene2Promoter (http://www.genomatix.de), respectively.

We ran both tools on the Biowulf Linux cluster (1 Gb/s Ethernet network, 4-core 2.8 GHz AMD Opteron(tm) Processor 290 and 8 GB memory) at the National Institutes of Health. To obtain comparable ADTs, we used the same default parameters, disabled all random choices and reduced the precision of some *double* variables in both tools (see Supplementary Material for modified M files of MEDUSA).

3 RESULTS

We tested the performance of FastMEDUSA and MEDUSA by controlling for the number of iterations and the number of processors used. Specifically, we defined the performance increase of FastMEDUSA utilizing i iterations and n processors as $\mathrm{PI}_{i,n} = t_i^{\mathrm{MEDUSA}} / t_{i,n}^{\mathrm{FastMEDUSA}}$, where t is the execution time. The results on the yeast and human data sets show that FastMEDUSA consistently outperforms MEDUSA (Fig. 1b). For instance, MEDUSA takes about 12 days to analyze the human glioma data set utilizing 400 iterations, whereas FastMEDUSA analyzes the same data set and obtains the same ADTs in about 6 h utilizing 100 processors. FastMEDUSA is also about 30% faster than MEDUSA in serial mode. The slope of the performance increase reduces for increasing computing resources because of the increase in the cost of inter-process communications.

4 CONCLUSIONS

Through the implementation of FastMEDUSA, a parallelized version of the gene regulatory-network modeling tool MEDUSA, we solve two problems inherent in the original implementation. First, we provide an open source implementation in C++ that uses freely available libraries. More importantly, we show that FastMEDUSA allows a dramatically more efficient utilization of computational resources. We expect that FastMEDUSA will be used on single multi-core machines, distributed systems or cloud computing facilities to model large data sets of complex organisms that would have been highly cumbersome using the serial implementation of the original algorithm.

ACKNOWLEDGEMENTS

We would like to thank Dr Christina Leslie and Steve Lianoglou for their useful discussion about MEDUSA.

Funding: Intramural Research Program of the National Institutes of Health; National Cancer Institute.

Conflict of Interest: none declared.

REFERENCES

Bussemaker,H.J. *et al*. (2001) Regulatory element detection using correlation with expression. *Nat. Genet.*, **27**, 167–171.

Cartharius,K. *et al*. (2005) MatInspector and beyond: promoter analysis based on transcription factor binding sites. *Bioinformatics*, **21**, 2933–2942.

Elemento,O. *et al*. (2007) A universal framework for regulatory element discovery across all genomes and data types. *Mol. Cell*, **28**, 337–350.

Freund,Y. and Shapire,R. (1997) A decision-theoretic generalization of on-line and an application of boosting. *J. Comp. Syst. Sci.*, **55**, 119–139.

Kundaje,A. *et al*. (2007) Learning regulatory programs that accurately predict differential expression with MEDUSA. *Ann. N Y Acad. Sci.*, **1115**, 178–202.

Kundaje,A. *et al*. (2008) A predictive model of the oxygen and heme regulatory network in yeast. *PLoS Comput. Biol.*, **4**, e1000224.

Li,A. *et al*. (2009) Unsupervised analysis of transcriptomic profiles reveals six glioma subtypes. *Cancer Res.*, **69**, 2091–2099.

Middendorf,M. *et al*. (2005) *Motif Discovery Through Predictive Modeling of Gene Regulation*. Miyano,S. *et al*. (eds), *RECOMB 2005*. Springer, Cambridge, MA, pp. 358–552.

Segal,E. *et al*. (2003) Genome-wide discovery of transcriptional modules from DNA sequence and gene expression. *Bioinformatics*, **19** (Suppl. 1), i273–i282.

Xiaohong,Q. *et al*. (2009) Cloud technologies for bioinformatics applications. *Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*. ACM, Portland, Oregon.

Downloaded from http://bioinformatics.oxfordjournals.org/ at marquette university Raynor Memorial Libraries on January 24, 2013