SIFTER-T: A scalable and optimized framework for the SIFTER phylogenomic method of probabilistic protein domain annotation

Danillo C. Almeida-e-Silva and Ricardo Z.N. Vêncio
Department of Computing and Mathematics FFCLRP-USP, University of Sao Paulo, Ribeirão Preto, Brazil

BioTechniques 58:140-142 (March 2015) doi 10.2144/000114266
Keywords: functional annotation; gene annotation; Sifter; bioinformatics; genomics software

Statistical Inference of Function Through Evolutionary Relationships (SIFTER) is a powerful computational platform for probabilistic protein domain annotation. Nevertheless, SIFTER is not widely used, likely due to usability and scalability issues. Here we present SIFTER-T (SIFTER Throughput-optimized), a substantial improvement over SIFTER's original proof-of-principle implementation. SIFTER-T is optimized for better performance, allowing it to be used at the genome-wide scale. Compared to SIFTER 2.0, SIFTER-T achieved an 87-fold performance improvement using published test data sets for the *known annotations recovering* module and a 72.3% speed increase for the *gene tree generation* module in quad-core machines, as well as a major decrease in memory usage during the realignment phase. Memory optimization allowed an expanded set of proteins to be handled by SIFTER's probabilistic method. The improvement in performance and automation that we achieved allowed us to build a web server to bring the power of Bayesian phylogenomic inference to the genomics community. SIFTER-T and its online interface are freely available under GNU license at http://labpib.fmrp.usp.br/methods/SIFTER-t/ and https://github.com/dcasbioinfo/SIFTER-t.

Many software tools are not widely adopted due to their complex interfaces and usability (1). Even tools known for the quality of their task execution are often abandoned in favor of those that are faster, simpler to use, or easier to install. Usability is so influential that even the most widely used tools (e.g., BLAST) are being adapted to user demands (2).

In the functional annotation field, SIFTER version 2.0 (3–5) is regarded as one of the best approaches when it comes to annotation quality (6). Recently, it was one of the top performing tools for functional annotation according to the international initiative Critical Assessment of Protein Function Annotation (CAFA), an open collaborative experiment designed to provide a large-scale assessment of software for predicting protein function using a time challenge (6). SIFTER combines two powerful concepts: phylogenomics (7) and Bayesian graphical models (8). Nevertheless, it is still not widely used. This paradox is probably due to framework usability and suitability issues when SIFTER is used at a high-throughput scale. To harness the power of this method for the wider bioinformatics community, several software engineering interventions need to be made to the original SIFTER proof-of-principle source code.

The original SIFTER workflow consists of two main steps: (*i*) annotation recovery for a list of genes and (*ii*) reconciled evolutionary gene tree generation for the same list. Next, SIFTER builds a Bayesian network structure in which the gene tree's leaves represent genes. Known functional annotations are associated to the aforementioned leaves and then probabilistically propagated along the Bayesian network to the leaves with no prior information. At the end of the process, a list of gene ontology functions and their probabilities of occurrence is generated for each gene with unknown function.

Although a powerful approach, this can be considered a prototype in terms of software. The current SIFTER version does not allow nucleotide or amino acid sequences to be inputted directly, nor does it accept current standards for gene annotation formats. Moreover, several necessary parameters are still hardcoded and difficult to be adjusted by the end user. Finally, its relationship to third-party dependent software is cumbersome, as is its visualization output.

The present work has two goals: (*i*) to enhance the tool's usability, through local implementations or a web-based front end and (*ii*) to optimize the original source code for better performance, allowing it to be used at a genome-wide scale. Studying the preprocessing workflow, we found opportunities for improvement and envisioned strategies to address them. As a consequence, the new implementation now also supports recent changes in the Gene Ontology (9), Gene Ontology Annotation (10), and Pfam (11) databases.

METHODS SUMMARY

Sifter-T is a framework that extends the functionality of Sifter, a prestigious protein functional annotation tool. With Sifter-T the annotations are performed in a more practical manner, and the result is generated much faster, allowing its use on a genome scale.

The strategies we have implemented include: the use of parallel threads; CPU load balancing, revised algorithms for best use of disk access, memory usage, and runtime; source-code adaptation to the currently used biological database formats; improved user accessibility; expansion of accepted input types; automation of the reconciliation process using gene trees and species trees; sequence filtering to reduce the analysis dimension; new output format; detailed documentation; and other minor implementations. The new framework, called SIFTER-T (SIFTER Throughput-optimized), is shown at Figure 1.

The improvements did not change the inference engine core made available by SIFTER. If the same input sequences, databases, and parameters are used, SIFTER-T returns exactly the same results as SIFTER. It is important to note that this is an artificial benchmark-only scenario because SIFTER-T's technical improvements allow it to, in fact, investigate more proteins and handle more a priori information, naturally leading to improved results even if user-input sequences are held constant. Using the benchmark data set proposed by the 2012 CAFA edition with default parameters, SIFTER-T achieved an F-measure score of 0.60, which is similar to the score of 0.50 obtained by the manually tuned SIFTER 2.0 (6). The bulk of structural changes around the inference core made it possible to interrogate some proteins that would be unfeasible in SIFTER 2.0. Of the 529 proteins evaluated in the present test using default parameters, SIFTER 2.0 was only able to deal with 214 proteins, but SIFTER-T was able to deal with 506 proteins.

With this implementation, we achieved enhanced performance speed-ups (one example is highlighted in Figure 2). As an illustration, using the original SIFTER 2.0 test data set "hundred families," we observed 87 times faster performance in the module responsible for known annotations recovery (from 37 min 48 s to 26 s). Similarly, we achieved a 72.3% speed increase in the module responsible for gene tree generation (from 3 h 57 min 14 sec to 1 h 26 min 22 sec) in quad-core machines. Regarding memory usage, we achieved a major decrease in usage, from nearly 201 GB (maximum) to nearly 8 GB (maximum), during the realignment phase for the complete Pfam families data set.

The increased performance allowed, for example, the reannotation of 419,029

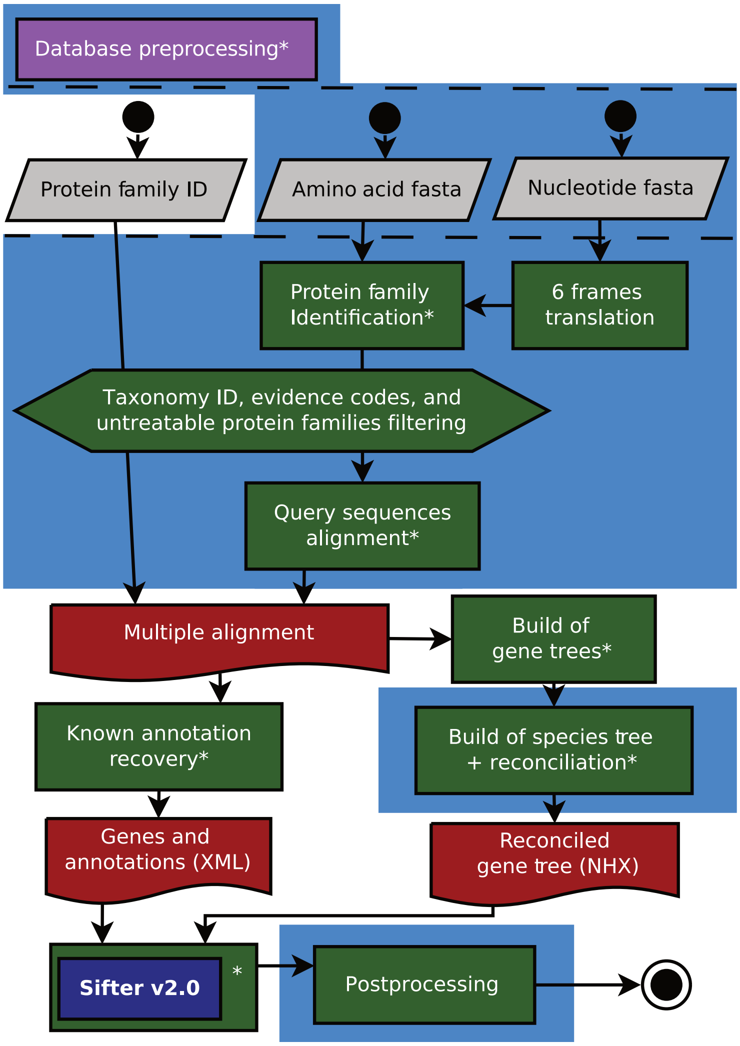
Database preprocessing*
Protein family ID
Amino acid fasta
Nucleotide fasta
Protein family Identification*
6 frames translation
Taxonomy ID, evidence codes, and untreatable protein families filtering
Query sequences alignment*
Multiple alignment
Build of gene trees*
Known annotation recovery*
Build of species tree + reconciliation*
Genes and annotations (XML)
Reconciled gene tree (NHX)
Sifter v2.0
*
Postprocessing

Figure 1. Overall view of the SIFTER-T (SIFTER Throughput-optimized) framework. Light blue: new features; red: intermediate files; green: SIFTER-T modules; purple: database pre-processing; grey: input data; asterisks: multithreaded modules.

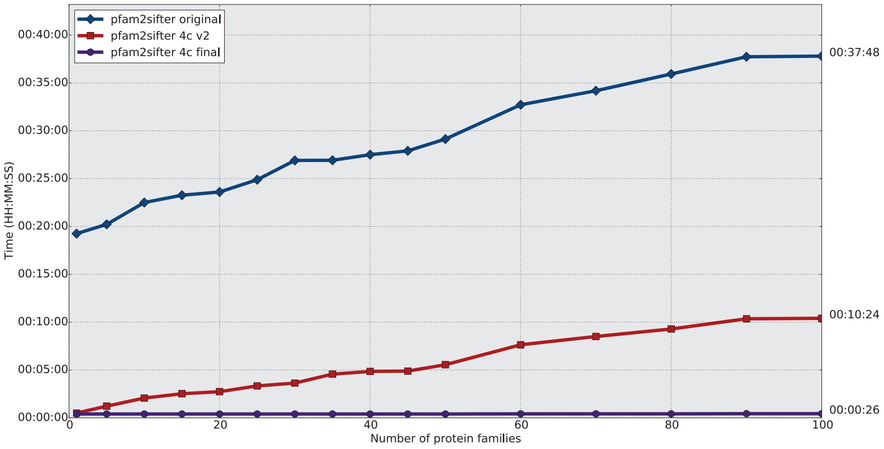

Figure 2. SIFTER-T (SIFTER Throughput-optimized) performance benchmark. Light blue: original SIFTER script (*pfam2sifter.py*) to recover gene annotations for known genes. Red: SIFTER-T implementation using improved multithreading processes and disk access optimization. Purple: SIFTER-T implementation using improved multithreading processes, disk access optimization, and high-performance internal BioPython data structures.

Saccharum officinarum (sugarcane) ESTs (http://compbio.dfci.harvard.edu/tgi/) (12) to be performed by SIFTER-T in 5 days, whereas BLAST (v2.2.26) (13) took 49 days in a standard bioinformatics laboratory computer (Intel 2x 6 core machine with 48 GB of memory and RAID SCSI disks).

SIFTER-T also presents completely new features relative to SIFTER v2.0: nucleotide and amino acid sequences as input data, annotation filtering by GO evidence codes or taxonomic group of origin, accessible and manageable output, specification of evidence codes prior probabilities, automated dependency and input data checking, annotation of huge protein families in modest machines (8 GB of memory), and multiprocessing capabilities.

The level of automation achieved allowed us to build a web-based tool to perform functional annotations from community requests, a resource that would be extremely difficult to implement using the original SIFTER v2.0 framework.

In conclusion, SIFTER-T is an open source tool with better usability and performance compared with the original Berkeley SIFTER 2.0 implementation. The new SIFTER-T features allow researchers to have easy and quick access to SIFTER's powerful annotation mathematical method, now with enhanced experimental customization. The online SIFTER-T interface can be found at: http://labpib.fmrp.usp.br/methods/SIFTER-t/.

Author contributions

D.C.A.S. devised, developed, and implemented software and methods. R.Z.N.V. proposed and supervised the project. D.C.A.S. and R.Z.N.V. wrote the manuscript.

Acknowledgments

We thank R.C. Waldemarin, M.A. Fenerich, A.M. Sasso, and other Laboratory for Biological Information Processing (LabPIB) colleagues for critical discussions. We thank the anonymous reviewers whose valuable contributions improved our work. We thank Prof. Steven E. Brenner at the University of California, Berkeley for allowing SIFTER-T to join the CAFA2 challenge. We thank the support of Microsoft Research/FAPESP grant 2009/53161-6. D.C.A.S. was supported by FAPESP fellowship 2010/50491-1.

Competing interests

The authors declare no competing interests.

References

1. **Veretnik, S., J.L. Fink, and P.E. Bourne.** 2008. Computational biology resources lack persistence and usability. PLOS Comput. Biol. *4*:e1000136.
2. **Boratyn, G.M., C. Camacho, P.S. Cooper, G. Coulouris, A. Fong, N. Ma, T.L. Madden, W.T. Matten, et al.** 2013. BLAST: a more efficient report with usability improvements. Nucleic Acids Res. *41*:W29-W33.
3. **Engelhardt, B.E., M.I. Jordan, K.E. Muratore, and S.E. Brenner.** 2005. Protein molecular function prediction by Bayesian phylogenomics. PLOS Comput. Biol. *1*:e45.
4. **Engelhardt, B.E., M.I. Jordan, and S.E. Brenner.** 2006. A graphical model for predicting protein molecular function. p. 297-304. In Proceedings of the 23rd International Conference on Machine Learning. New York, NY. ACM Press.
5. **Engelhardt, B.E., M.I. Jordan, J.R. Srouji, and S.E. Brenner.** 2011. Genome-scale phylogenetic function annotation of large and diverse protein families. Genome Res. *21*:1969-1980.
6. **Radivojac, P., W.T. Clark, T.R. Oron, A.M. Schnoes, T. Wittkop, A. Sokolov, K. Graim, C. Funk, et al.** 2013. A large-scale evaluation of computational protein function prediction. Nat. Methods *10*:221-227.
7. **Eisen, J.A.** 1998. Phylogenomics: improving functional predictions for uncharacterized genes by evolutionary analysis. Genome Res. *8*:163-167.
8. **Jordan, M.I.** 2004. Graphical Models. Statist. Sci. *19*:140-155.
9. **Gene Ontology Consortium.** 2013. Gene Ontology annotations and resources. Nucleic Acids Res. *41(Database issue)*:D530-D535.
10. **Dimmer, E.C., R.P. Huntley, Y. Alam-Faruque, T. Sawford, C. O'Donovan, M.J. Martin, B. Bely, P. Browne, et al.** 2012. The UniProt-GO Annotation database in 2011. Nucleic Acids Res. *40(Database issue)*:D565-D570.
11. **Punta, M., P.C. Coggill, R.Y. Eberhardt, J. Mistry, J. Tate, C. Boursnell, N. Pang, K. Forslund, et al.** 2012. The Pfam protein families database. Nucleic Acids Res. *40(D1)*:D290-D301.
12. **Arruda, P.** 2001. Sugarcane transcriptome. A landmark in plant genomics in the tropics. Genet. Mol. Biol. *24*:0-0.
13. **Altschul, S.F., W. Gish, W. Miller, E.W. Myers, and D.J. Lipman.** 1990. Basic local alignment search tool. J. Mol. Biol. *215*:403-410.

Received 08 September 2014; accepted 17 December 2014.

Address correspondence to Dr. Ricardo Vêncio, Department of Computing and Mathematics, FFCLRP-USP, University of Sao Paulo, Brazil. E-mail: rvencio@usp.br

Copyright © 2015 Informa Healthcare USA, Inc. *BioTechniques* (ISSN 0736-6205) is a registered trademark of Informa Healthcare USA, Inc., and is published monthly at 52 Vanderbilt Avenue, 16th Floor, New York, NY 10017, USA, Phone: (212) 520-2777, Fax: (212) 520-2705, Internet: www. BioTechniques.com. Articles are listed in all major informational databases. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by Informa Healthcare USA, Inc., provided that the appropriate fee is paid directly to Copyright Clearance Center, 222 Rosewood Dr., Danvers, MA 01923, USA. Prior to photocopying items for educational classroom use, please contact the Copyright Clearance Center, Customer Service, 222 Rosewood Dr., Danvers, MA 01923, USA, (267) 557-3999. Periodical postage is paid at New York, NY, and additional mailing offices. Subscription Rates: Single copy price is $203. Subscriptions for institutes and other nonqualified recipients are $1215 annually. **POSTMASTER:** Send Address Changes to: *BioTechniques* 7300 Linder Ave Skokie, IL 60077, USA. Internet: www.BioTechniques.com. Canadian publication agreement No. 40612608. Canadian return address, P.O. Box 25542, London, ON N6C 6B2, Canada. Printed in USA.