Published in final edited form as:
Concurr Comput. 2014 September 10; 26(13): 2112–2121. doi:10.1002/cpe.3264.

Optimizing high performance computing workflow for protein functional annotation

Larissa Stanberry[1,*,†], **Bhanu Rekepalli**[2], **Yuan Liu**[2], **Paul Giblock**[3], **Roger Higdon**[1], **Elizabeth Montague**[1], **William Broomall**[1], **Natali Kolker**[1], and **Eugene Kolker**[4]

[1]Bioinformatics & High-Throughput Analysis Laboratory and High-Throughput Analysis Core, Seattle Children's Research Institute (SCRI), DELSA Global, Seattle, WA 98101, USA

[2]Joint Institute for Computational Sciences, University of Tennessee – Oak Ridge National Laboratory (JICS UT – ORNL), DELSA Global, Oak Ridge, TN, USA

[3]Cisco Systems, San Jose, CA 95134, USA

[4]Bioinformatics & High-throughput Analysis Laboratory, SCRI, High-throughput Analysis Core, SCRI, Predicitive Analytics, Seattle Children's Hospital, Departments of Pediatrics and Biomedical Informatics & Medical Education, University of Washington, DELSA Global

Abstract

Functional annotation of newly sequenced genomes is one of the major challenges in modern biology. With modern sequencing technologies, the protein sequence universe is rapidly expanding. Newly sequenced bacterial genomes alone contain over 7.5 million proteins. The rate of data generation has far surpassed that of protein annotation. The volume of protein data makes manual curation infeasible, whereas a high compute cost limits the utility of existing automated approaches. In this work, we present an improved and optmized automated workflow to enable large-scale protein annotation. The workflow uses high performance computing architectures and a low complexity classification algorithm to assign proteins into existing clusters of orthologous groups of proteins. On the basis of the Position-Specific Iterative Basic Local Alignment Search Tool the algorithm ensures at least 80% specificity and sensitivity of the resulting classifications. The workflow utilizes highly scalable parallel applications for classification and sequence alignment. Using Extreme Science and Engineering Discovery Environment supercomputers, the workflow processed 1,200,000 newly sequenced bacterial proteins. With the rapid expansion of the protein sequence universe, the proposed workflow will enable scientists to annotate big genome data.

Keywords

science gateways; petascale; data-enabled life sciences; sequence similarity; computational bioinformatics; protein annotation; protein sequence universe; COG; BLAST; PSI-BLAST; HSPp-BLAST; XSEDE; PS

[*]Correspondence to: Larissa Stanberry, 1900 9th Ave, C9S-9, Seattle, WA 98101, USA. [†]larissa.stanberry@seattlechildrens.org.

1. INTRODUCTION

Advances in sequencing technologies enabled rapid generation of data rapidly approaching exa-byte scale [1–3]. The rapid expansion of the protein sequence universe makes protein annotation one of the principal challenges in life sciences [4]. Functional annotation of big protein data is an immense computational challenge that requires advanced analytical tools and next-generation compute capabilities [5–10]. Approximately one third of proteins in any newly sequenced genome have unknown function [11–15]. This barrier remains relatively constant as new organisms are being sequenced. The rapid expansion of protein sequence universe makes manual curation infeasible.

Clustering methods have gained popularity as automated approaches that can facilitate the annotation of large number of proteins [16–23]. Clustering approaches typically assume sequence similarity of orthologous genes. Thus, functionally similar proteins are expected to cluster together.

Existing clustering approaches are not easily scalable to handle large data volumes [4, 24]. Consequently, public annotation resources are struggling to cope with the influx of data and a growing number of them can no longer sustain the expansion [19, 21, 25]. The Clusters of Orthologous Groups of proteins (COG) database is one of the primary tools in functional annotation and comparative genomics [21]. Last updated in 2006, the database contains 70% of protein-coding genes from 69 prokaryotic and eukaryotic genomes. COG clusters are based on a top-hit approach using Basic Local Alignment Search Tool (BLAST) sequence similarity scores [21, 26]. The resulting clusters were manually curated to verify the grouping and functional annotation.

The COG clusters were used as seeds in the eggNOG database to construct groups of orthologous genes [17, 27]. The eggNOG method uses precomputed all-versus-all alignment scores from FASTA algorithm. The FASTA algorithm is more sensitive than BLAST but has higher compute cost. The size of user queries in eggNOG is limited to 30 sequences and can take up to 10 min according to the manual. Both the query size and the processing speed limit the utility of eggNOG as an annotation resource for big data. With the proliferation of new organisms and environments being sequenced, even small laboratories would require efficient annotation pipelines capable of handling large volumes of sequencing data.

In [28], we proposed a new workflow to enable large-scale functional annotation using high performance computing (HPC) architectures. The workflow uses a Position-Specific Iterative (PSI)-BLAST approach and low-complexity classification method to assign newly sequenced bacterial genomes to existing COGs [21, 26]. To reduce error propagation, the workflow is iterative. With each iteration, the expansion of COG clusters leads to increase in compute time. Here, we optimize the workflow to reduce compute time and memory demands while retaining the accuracy. The optimized workflow relies on highly scalable parallel implementations of the analysis tools to enable rapid analysis on the National Science Foundation (NSF)-funded Kraken supercomputer and the Newton HPC cluster at the University of Tennessee [29, 30]. Kraken is a Cray XT5 machine consisting of 112,896 AMD Opteron compute cores at 2.6 GHz in 9408 nodes with 147 TB of total memory [29].

Newton consists of 250 compute nodes with a total of over 2400 processor cores. The processors are Intel Xeon model E5-2670 clocked at 2.6 GHz. Each node has 32 GB of system memory. The Kraken development will be then used to adapt to other NSF resources and to develop a widely portable and parallel pipeline.

2. METHODS

2.1. Cluster of orthologous groups of proteins database

By a major principle of molecular evolution, functionally important proteins tend to be conserved across species. Developed by the National Center for Biotechnology Information (NCBI), the COG database is separated into prokaryotic (COG) and eukaryotic (KOG) orthologous groups [21]. The COG database contains 192,987 proteins from 66 bacterial proteomes that form 4873 clusters including 2824 well-annotated COGs with 107,711 proteins and 2049 poorly annotated COGs with 36,609. The KOG database contains 112,920 proteins from seven eukaryotic proteomes mapped into 4852 functional clusters [21, 31].

Both COG and KOG are used to identify paralog and ortholog proteins. According to Google Scholar, the COG project is one of the most popular protein resources with approximately 4.5K citations. However, because of computational costs, the database was last updated in 2008 and is not currently maintained. Here, we are using the COG database of prokaryotic genomes that we will refer to as COGs.

2.2. Classification

The existing clustering algorithms are based on pairwise sequence similarity. However, computing pairwise scores is computationally intensive. To overcome this limitation, we developed a simple classification method to assign proteins to existing COGs. The method is based on PSI-BLAST where the score is computed using a position-specific scoring matrix, or a profile [32]. The profile allows detecting distant relationships between proteins.

Here, for each COG cluster, we compute a profile and a consensus sequence. The consensus is constructed by aligning sequences in a given cluster. The resulting consensus reflects an agreement of residues in the alignment columns. For each newly sequenced protein, we then compute a similarity score with a given consensus and a corresponding profile. Hence, the use of PSI-BLAST allows comparing a protein with the entire cluster simultaneously, thus avoiding numerous pairwise alignments. For each protein, we identify the top-scoring COG cluster. The protein is assigned into the top-scoring cluster if the corresponding score exceeds the threshold.

The optimal threshold selection was determined using test and train sets built from COG database. The validation was perfomed on archaeal data. For more details, the reader can refer to [28].

2.3. Implementation

The proposed pipeline includes BLAST functions, MUSCLE [33], Perl scripts, and statistical analysis performed in R [34]. BLAST and MUSCLE were designed to run on a typical workstation. Thus, many parallel implementations of these tools focused on clusters

exist, such as mpi-BLAST [35], ScalaBLAST [36], and ClustalW-MPI [37]. These tools achieve scalability through database fragmentation, parallel I/O, load balancing, and query prefetching. However, only a few highly scalable implementations capable of using tens of thousands of cores on supercomputers are available including BLAST on Blue Gene/L [38], pioBLAST [39], and HSPp-BLAST [15]. We chose to use the HSPp-BLAST implementation because of familiarity and proven results on the Kraken supercomputer.

The scalable wrapper, introduced with HSPp-BLAST, can also be used to scale other components of the pipeline, such as MUSCLE. This wrapper is designed to provide enhanced scalability on HPC systems with tens of thousands of compute cores. It does this through two main components: `mcw` and `stdiowrap`. The `mcw` program is aMessage Passing Interface (MPI) application written in the C programming language [40] and is named for its hierarchical design: master, controller, worker. Figure 1 shows the system architecture originally presented in [41] as applied to PSI-BLAST in this project. The master is responsible for distributing the sequence database as well as input sequences to the controller positioned on each of the other nodes. The workers represent each individual instance of the scaled tool, for example, BLAST. The BLAST and MUSCLE programs are linked with the `stdiowrap` library, which provides input and output redirections with the controller through interprocess communication implemented by means of shared memory segments.

For data storage, supercomputers largely use a distributed file system, such as Lustre [42]. In a distributed file system, each instance of the tool needs to load a copy of the database. The file system does not perform well when thousands of processes are trying to read the same files simultaneously. The wrapper provides input enhancements by reading the database with the master, then broadcasting the data with MPI's `MPI_Bcast` function. The `MPI_Bcast` scales logarithmically with the number of nodes. In addition, in order to reduce input latency, input sequences are read by the master and prefetched by the controllers. The performance of output operations was also improved by implementing a two-stage buffering technique to provide asynchronous writes. The tools write to an in-memory buffer instead of directly to disk. The data are flushed (and optionally compressed) from in-memory buffers to disk by a background process when the buffers are nearly full, rather than on demand. This increases the output bandwidth, results in more uniform output time, and almost eliminates blocking in the tool itself.

To reduce error propagation, the COG database is expanded gradually. After each expansion, we recompute the profiles and the consensus sequences. We used the first four iterations to estimate the complexity and compute demand time of the algorithm. For iterations 1–4, we sampled at random $n = 200; 000$ new proteins. The sampled sequences are formatted into a database using `formatdb`. For each COG cluster, we use HSPp-BLAST to compute the alignment scores for the COG clusters' consensus sequences against the newly sequenced proteins. We use the respective profile to jump start the search. The proteins were assigned to the top hit COG given the alignment score exceeded the threshold.

After each iteration, the expanded COGs were used to recompute the consensus and profiles and update the classification threshold. The consensus sequence and a profile for each COG

are built with `set_extractCore`, a custom Perl script developed by NCBI (Y. Wolf). Given the list of protein IDs in a COG, this script identifies a nonredundant subset of sequences using `fastacmd` and `blastclust`. The common domain for nonredundant sequences is then built with PSI-BLAST, and the sequences are aligned with MUSCLE. The algorithm used in the first four iterations is given in Figure 1 of [28].

The proposed workflow is currently implemented as a series of Perl, BASH, and R scripts. For PSI-BLAST computations, we bundled the results into a single file and transferred them to the Kraken supercomputer [29]. We ran HSPp-BLAST on Kraken using 480 compute cores. For the first four iterations, the mean (± standard deviation) result file size was 31 ± 2.8 GB. The COG profiles are computed on the Newton HPC cluster [30].

2.4. Algorithm optimization

In the first four iterations of the original implementation [28], we have analyzed 800,000 proteins from the newly sequences genomes [43]. Consequently, 513,923 proteins were assigned to 4525 COG clusters. The results were comparable across the four iterations in terms of the alignment score distribution and the sensitivity and specificity of the classification approach [28]. The rate of cluster expansion varied across COGs and so did the profile computation. The run time for profile computation depended on the size and heterogeneity of the COG cluster. Figure 2 shows the run time as a function of COG size. By our estimates, for COGs with 500 sequences or less, the profile computing run time is sublinear, $\mathcal{O}(\sqrt{n})$, and it is $\mathcal{O}(n^2)$. otherwise.

After iteration 4, a number of COG clusters were expanded to include over 2000 sequences. This in turn increased the compute time. To make the algorithm more efficient, we developed a new processing scheme (Figure 3). Under the new scheme, after each iteration, we identify sufficiently expanded COGs that have at least 5% cumulative expansion in size. For any sufficiently expanded COG that contains 2000 or more proteins, the sequences are first pre-clustered with `usearch`. The resulting subcluster cores are then used to compute COG profiles and consensus. Profiles for COGs with less than 5% cumulative expansion in size remain unchanged.

To assess the effect of pre-clustering on profile compute time and oucomes, we performed a small study. For that, we have selected all clusters that had at least 2000 sequences after the third iteration of the original algorithm [28]. The four selected COGs were pre-clustered using `usearch`, and the identified subcluster cores were used to compute the profiles. Table I shows the reduction in sequence set size (relative) and in compute time when using subcluster cores as compare with the entire COG clusters. The pre-clustering effectively reduces the size of the sequence set and the time it takes to compute the profiles.

Further, we considered the 200,000 query proteins from iteration 4. At the fourth iteration, 4411 proteins were assigned to one of the COG0477, 0583, 0745, and 1028 using profiles computed from the entire COGs. We then clustered the four largest COGs with `usearch`, extracted the subcluster cores, and computed the consensus and profiles with `set_extractCore` routine. For each of the query proteins, we then computed the PSI-BLAST alignment score against the four largest COGs based on subcluster profiles and

consensus. Figure 4 shows the correspondence between the PSIBLAST scores computed using profiles of the entire COG cluster and those of subcluster cores only. Overall, the two alignments scores show a high degree of correlation. While the difference in scores is statistically significant, the magnitude is relatively low. The subcluster-based scores tend to be higher than those based on the complete cluster with the exception of COG0477 that shows negative bias (Figure 5, Table I). Note that we limited the display to alignment scores that exceed the classification threshold, that is, those that have an alignment score big enough to be classified.

From 4411 proteins assigned to the four COGs at the fourth iteration, 4238 proteins (96%) would be classified into COGs. From these, 4131 (94%) would be assigned to the same COG as in iteration 4, and 107 (2%) would be assigned to COGs different from those in iteration 4. Also with preclustering, 203 (4.5%) proteins would not be assigned to any COGs because of the low alignment scores.

These results show that the preclustering step can sufficiently reduce compute time without affecting the quality of the output.

2.5. Algorithmic improvements

After the four iterations, the COG clusters in most cases were significantly expanded. This expansion led to `blastpgp` program failing. The program did not crash, but rather produced numerous warning messages of the form *"ObjMgrNextAvailEntityID failed with idx 2048"*. Consequently, the program produced incorrect alignments.

Because the failure comes as a warning, it is rather easy to overlook. In our case, `blastpgp` is first used to build COG cluster profiles and consensus, and then used to compute alignment scores for protein sequences against given COG consensus and profiles. The resulting profiles looked as expected and were successfully processed by the subsequent run of `blastpgp` as an input. In fact, we only discovered this issue when optimizing the algorithm for large COGs as described in the next section.

Later, we found numerous mentions of this warning by `blastpgp` users, most did not dwell on it and mediated the problem by either shortening the size of the query or reducing the number of passes. Although not incorporated into the `blastpgp`, the patch to fix this bug has been successfully developed and implemented [44]. The reason for the warning and the subsequent fail was due to `assignedIDsArray` variable not being reset after each iteration of `blastpgp` or after each query sequence. The provided fix resets the array thus ensuring sufficient number of IDs for the next round of computations. The solution has been rigorously tested and implemented at the Finnish IT Center for Science [44]. In addition, the implementation provides anywhere from 0% to 40% improvement in the compute speed depending on the query [45].

3. RESULTS

Currently, we have tuned and deployed the new classification pipeline on Kraken supercomputer. We have run the fifth iteration processing 400,000 proteins under the new

classification scheme (Figure 3). As a result 201,837 proteins were assigned to 3862 clusters with the median log alignment score of 5.5.

To monitor the accuracy of the classification approach, we used the results from the first five iterations to compare the classification of the sequences in the original database with that of their analogues. We performed BLAST to match sequences in the original COG database with their newly sequenced analogues. After iteration 5, there were 10,894 high certainty alignments (bit score > 300) out of which 10,142 (93%) had a matching COG assignment.

We matched 190,194 sequences in the original COG with their newly sequenced analogues. From these, 29,676 (15%) were selected for classifcation in iterations 1–5. From 29,676 selected for classification, 18,373 proteins were classified by both methods including 16,683 that were assigned to the same cluster by the two methods (Table II). The two methods gave identical classification for23,276 (16,683C6,593 or 78%) of proteins. The classification using new method differed from the original for 6400 (= 676 + 4034 + 18, 373 − 16; 683 or 22%) of proteins. These results confirm the quality of the classification to be consistent with that expected from the design.

4. CONCLUSION

Functional protein annotation is one of the most important challenges in biology [4]. The number of sequenced genomes increases rapidly. The Earth Microbiome Project alone is expected to sequence 500,000 microbial genomes [3]. This is well over a 100-fold increase in the number of sequenced microbial genomes and proteins currently contained in GenBank [46]. The i5K Insect and other Arthropod Genome Sequencing Initiative are expected to yield nearly 100 million new protein sequences [2]. Assigning functions to this glut of newly sequenced proteins is an immense computational and scientific challenge.

In this paper, we propose improvements to the automated workflow to enable large-scale functional annotation using HPC architectures. We use the workflow to annotate newly sequenced bacterial genomes. Currently, we have processed 1,200,000 newly sequenced proteins. The improvements include a revised scheme to compute profiles for large COGs. The new scheme allows reduction in compute time while maintaining the accuracy of classifications. The improved workflow relies on highly scalable parallel applications for the alignment and classification analysis.

The workflow was developed for Extreme Science and Engineering Discovery Environment supercomputers. Currently, we are working on enhancing and extending the wrapper. We also are working on enabling streaming when the output from one tool serves as an input to the next tool through the shared memory segments. Streaming allows running all tools to process a single COG on the same compute node, thus effectively reducing compute time and memory load. Streaming differs from other workflow automation systems such as Airavata [47] that uses a course-grained approach where each step is performed as a separate job.

The scope of functional annotation task far exceeds resources and capabilities of the life sciences community. The rate of increase in sequencing data requires new, trans-disciplinary approaches to efficiently translate the influx of new data into ground-breaking discoveries. Communities like Data-Enabled Life Sciences Alliance (DELSA Global) stand to promote a collective innovation through interdisciplinary research and transdisciplinary engagement [48, 49]. Grand challenges in modern life sciences can be only solved through concerted effort of scientific collectives like DELSA Global.

Acknowledgments

The authors thank Eugene Koonin and Yuri Wolf (NCBI) for expert advice and technical help in developing the analysis method. The authors are grateful to Elizabeth Stewart for critical reading of the paper. The feedback and comments from the anonymous reviewers are also much appreciated. This work was made possible by the support from the National Science Foundation under the Division of Biological Infrastructure award 0969929, National Institute of Diabetes and Digestive and Kidney Diseases of the National Institutes of Health under awards U01-DK-089571 and U01-DK-072473, Seattle Children's Research Institute award, The Robert B. McMillen Foundation award, and The Gordon and Betty Moore Foundation award to EK; XSEDE (Extreme Science and Engineering Discovery Environment) allocation of advanced computing resources (Kraken Supercomputer) was provided by the National Science Foundation and NSF grant EPS-0919436. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Science Foundation, National Institutes of Health, Seattle Children's Research Institute, Stanford University, The McMillen Foundation, or The Moore Foundation.

REFERENCES

1. Higdon R, Haynes W, Stanberry L, Stewart E, Yandl G, Howard C, Broomall W, Kolker N, Kolker E. Unraveling the complexities of life sciences data. Big Data. 2013; 1(1):42–50.
2. Robinson GE, Hackett KJ, Purcell-Miramontes M, Brown SJ, Evans JD, Goldsmith MR, Lawson D, Okamuro J, Robertson HM, Schneider DJ. Creating a buzz about insect genomes. Science. 2011; 331:1386. [PubMed: 21415334]
3. The Earth Microbiome Project.
4. Baumgartner WA, Cohen KB, Fox LM, Acquaah-Mensah G, Hunter L. Manual curation is not sufficient for annotation of genomic databases. Bioinformatics. 2007; 23:i41–i48. [PubMed: 17646325]
5. Kolker E, Higdon R, Haynes W, Welch D, Broomall W, Lancet D, Stanberry L, Kolker N. MOPED: model organism protein expression database. Nucleic Acids Research. 2012; 40 (Database issue):D1093–D1099. PMID: 22139914.
6. Kolker E, Higdon R, Welch D, Bauman A, Stewart E, Haynes W, Broomall W, Kolker N. SPIRE: systematic protein ivestigative research environment. Journal of Proteomics. 2011; 75(1):122–126. PMID: 21609792. [PubMed: 21609792]
7. Pennisi E. Human genome 10th anniversary. Will computers crash genomics? Science. 2011; 331:666–668. [PubMed: 21310981]
8. Schatz MC, Langmead B, Salzberg SL. Cloud computing and the DNA data race. Nature Biotechnology. 2010; 28:691–693.
9. Stanberry L, Higdon R, Haynes W, Kolker N, Broomall W, Ekanayake S, Ruan Y, Qiu J, Kolker E, Fox G. Visualizing the Protein Sequence Universe. Concurrency and Computation: Practice and Experience. 2013; 26(6):1313–1325.
10. Stein LD. The case for cloud computing in genome informatics. Genome Biology. 2010; 11:207. [PubMed: 20441614]
11. Bork P. Powers and pitfalls in sequence analysis: the 70% hurdle. Genome Research. 2000 Apr. 10:398–400. [PubMed: 10779480]
12. Galperin MY, Kolker E. New metrics for comparative genomics. Current Opinion in Biotechnology. 2006; 17(5):440–447. [PubMed: 16978854]

13. Kolker E, Makarova KS, Shabalina S, Picone AF, Purvine S, Holzman T, Cherny T, Armbruster D, Munson RS, Kolesov G, Frishman D, Galperin MY. Identification and functional analysis of 'hypothetical' genes expressed in Haemophilus influenzae. Nucleic Acids Research. 2004; 32:2353–2361. [PubMed: 15121896]
14. Koonin, E.; Galperin, M. Sequence - Evolution - Function: Computational Approaches in Comparative Genomics. Boston: Kluwer Academic; 2003. ISBN-10: 1-40207-274-0.
15. Rekepalli, B.; Vose, A.; Giblock, P. HSPp-BLAST: highly scalable parallel PSI-BLAST for very large-scale sequence searches. In: Saeed, F.; Khokhar, A.; Al-Mubaid, H., editors. Proceedings of the 3rd International Conference on Bioinformatics and Computational Biology (BICoB). Las Vegas, NV: BICoB; 2012. p. 37-42.
16. Enright AJ, Van Dongen S, Ouzounis CA. An efficient algorithm for large-scale detection of protein families. Nucleic Acids Research. 2002; 30(7):1575–1584. [PubMed: 11917018]
17. Jensen LJ, Julien P, Kuhn M, von Mering C, Muller J, Doerks T, Bork P. eggNOG: automated construction and annotation of orthologous groups of genes. Nucleic Acids Res. 2008 Jan. 36:D250–D254. [PubMed: 17942413]
18. Kolker N, Higdon R, Broomall W, Stanberry L, Welch D, Lu W, Haynes W, Barga R, Kolker E. Classifying proteins into functional groups based on all-versus-all BLAST of 10 million proteins. OMICS: A Journal of Integrative Biology. 2011; 15(7–8):513–521. [PubMed: 21809957]
19. Krause A, Stoye J, Vingron M. The SYSTERS protein sequence cluster set. Nucleic Acids Research. 2000; 28:270–272. [PubMed: 10592244]
20. Li W, Godzik A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics. 2006; 22(13):1658–1659. [PubMed: 16731699]
21. Tatusov RL, Fedorova ND, Jackson JD, Jacobs AR, Kiryutin B, Koonin EV, Krylov DM, Mazumder R, Mekhedov SL, Nikolskaya AN, Rao BS, Smirnov S, Sverdlov AV, Vasudevan S,Wolf YI, Yin JJ, Natale DA. The COG database: an updated version includes eukaryotes. BMC Bioinformatics. 2003; 4:41. [PubMed: 12969510]
22. Vlasblom J, Wodak S. Markov clustering versus affinity propagation for the partitioning of protein interaction graphs. BMC Bioinformatics. 2009; 10(1):99. [PubMed: 19331680]
23. Yona G, Linial N, Linial M. ProtoMap: automatic classification of protein sequences and hierarchy of protein families. Nucleic Acids Research. 2000; 28:49–55. [PubMed: 10592179]
24. Frishman D. Protein annotation at genomic scale: the current status. Chemical Reviews. 2007; 107:3448–3466. [PubMed: 17658902]
25. Kriventseva EV, Fleischmann W, Zdobnov EM, Apweiler R. CluSTr: a database of clusters of SWISS-PROT + TrEMBL proteins. Nucleic Acids Research. 2001; 29:33–36. [PubMed: 11125042]
26. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. Journal of Molecular Biology. 1990; 215:403–410. [PubMed: 2231712]
27. Powell S, Szklarczyk D, Trachana K, Roth A, Kuhn M, Muller J, Arnold R, Rattei T, Letunic I, Doerks T, Jensen LJ, von Mering C, Bork P. eggNOG v3.0: orthologous groups covering 1,133 organisms at 41 different taxonomic ranges. Nucleic Acids Research. 2012; 40(D1):D284–D289. [PubMed: 22096231]
28. Stanberry, L.; Liu, Y.; Rekepalli, B.; Giblock, P.; Higdon, R.; Broomall, W. High performance computing workflow for protein functional annotation. In: Wilkins-Diehr, N., editor. XSEDE. New York, NY, USA: ACM; 2013. p. 19
29. National Institute for Computational Sciences. http://www.nics.tennessee.edu/computing-resources/kraken. Kraken XT5.
30. The University of Tennessee. Newton HPC Program – High Performance Computing. http://newton.utk.edu.
31. Tatusov R, Koonin E, Lipman D. A genomic perspective on protein families. Science. 1997; 278:631–637. [PubMed: 9381173]
32. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research. 1997; 25:3389–3402. [PubMed: 9254694]

33. Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Research. 2004; 32(5):1792–1797. [PubMed: 15034147]
34. R Core Team. R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing; 2012. ISBN: 3-900051-07-0.
35. Darling, AE.; Carey, L.; chun Feng, W. Proceedings of ClusterWorld 2003. San Jose, CA: 2003. The design, implementation, and evaluation of mpiBLAST.
36. Oehmen CS, Baxter DJ. ScalaBLAST 2.0: rapid and robust blast calculations on multiprocessor systems. Bioinformatics. 2013; 29(6):797–798. [PubMed: 23361326]
37. Li K-B. ClustalW-MPI: ClustalW analysis using distributed and parallel computing. Bioinformatics. 2003; 19(12):1585–1586. [PubMed: 12912844]
38. Rangwala, H.; Lantz, E.; Musselman, R.; Pinnow, K.; Smith, B.; Wallenfelt, B. High Availability and Performance Workshop. Austin, TX: 2005. Massively parallel BLAST for the Blue Gene/L.
39. Lin, H.; Ma, X.; Ch, YP. International Parallel and Distributed Processing Symposium. Denver, CO: 2005. Efficient data access for parallel blast.
40. Kernighan BW, Ritchie DM. The C programming language. 1988
41. Rekapalli B, Giblock P, Reardon C. PoPLAR: portal for petascale lifescience applications and research. BMC Bioinformatics. 2013; 14(Suppl 9):S3. [PubMed: 23902523]
42. Cluster File Systems, Inc. Lustre: a scalable, high performance file system. 2002
43. NCBI. [Accessed on 14 January 2013] Genome assembly/annotation projects. ftp://ftp.ncbi.nih.gov/genomes/Bacteria
44. Osowski, K.; Westerholm, J.; Aspnäs, M. Technical Report. TURKU: Finland; 2007. Two cases of data overflow in the protein sequencing program `blastpgp`; p. 813Turku Centre for Computer Science, Joukahaisenkatu 3–5: 20520
45. Osowski K, Westerholm J, Aspnäs M. Optimized PSI-BLAST. http://www.csc.fi/english/research/sciences//bioscience/programs/blast/optimized_psiblast/?searchterm=blastpgp, http://www.csc.fi/.
46. Benson DA, Karsch-Mizrachi I, Clark K, Lipman DJ, Ostell J, Sayers EW. GenBank. Nucleic Acids Research. 2012; 40:D48–53. (Database issue): PMID: 22144687. [PubMed: 22144687]
47. Marru, S.; Gunathilake, L.; Herath, C.; Tangchaisin, P.; Pierce, M.; Mattmann, C.; Singh, R.; Gunarathne, T.; Chinthaka, E.; Gardler, R., et al. Proceedings of the 2011 ACM Workshop on Gateway Computing Environments. New York, NY, USA: ACM; 2011. Apache Airavata: a framework for distributed applications and computational workflows; p. 21-28.
48. Kolker E, Stewart E, Özdemir V. DELSA global for “Big Data” and the Bioeconomy: Catalyzing Collective Innovation. Industrial Biotechnology. 2012; 8(4):176–178.
49. Ozdemir V, Pang T, Knoppers BM, Avard D, Faraj SA, Zawati MH, Kolker E. Vaccines of the 21st century and vaccinomics: data-enabled science meets global health to spark collective action for vaccine innovation. OMICS: A Journal of Integrative Biology. 2011; 15(9):523–527. [PubMed: 21848418]

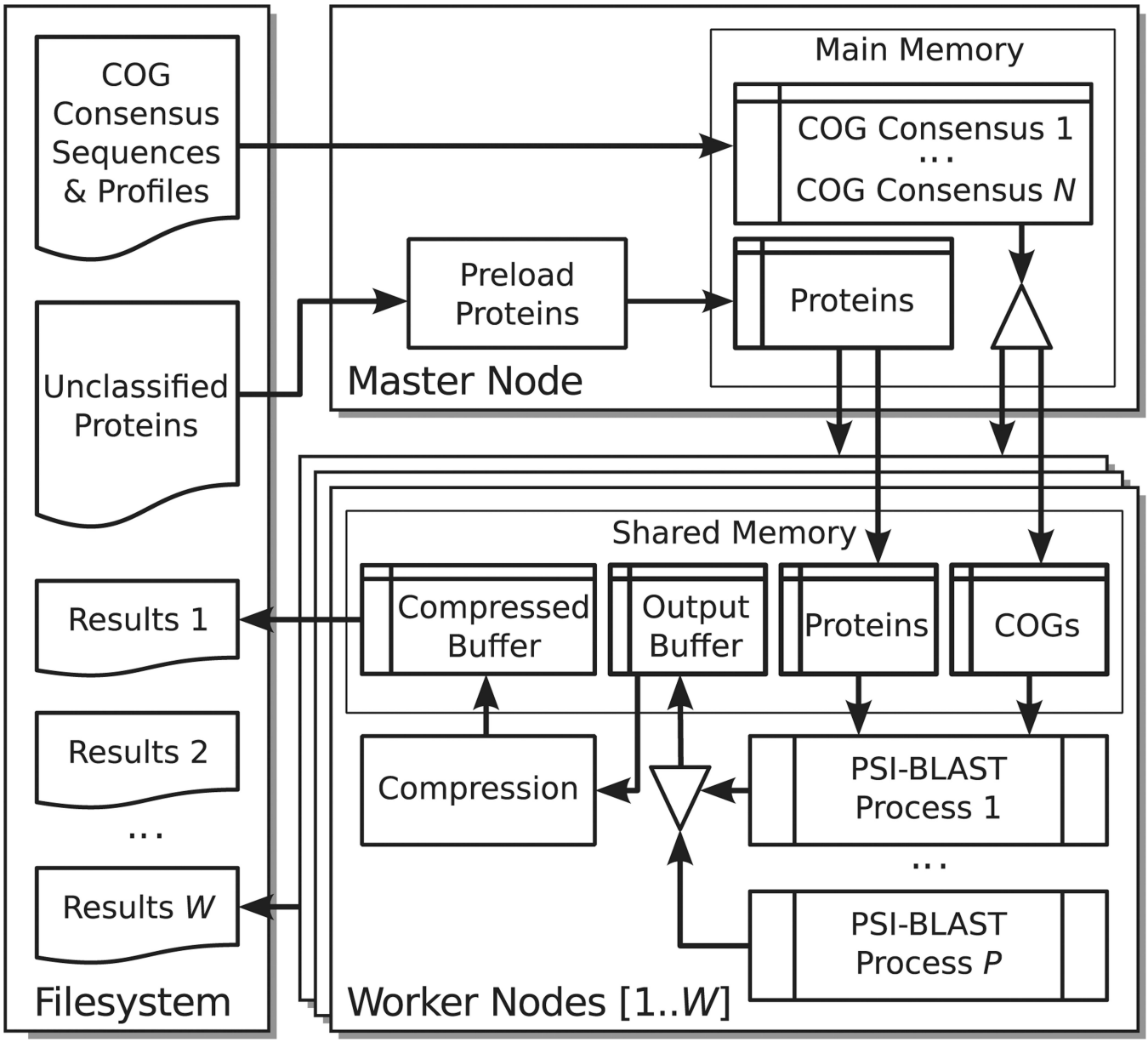

Figure 1.
The wrapper architecture. The high-level design of the parallel wrapper used to scale tools on HPC architectures.

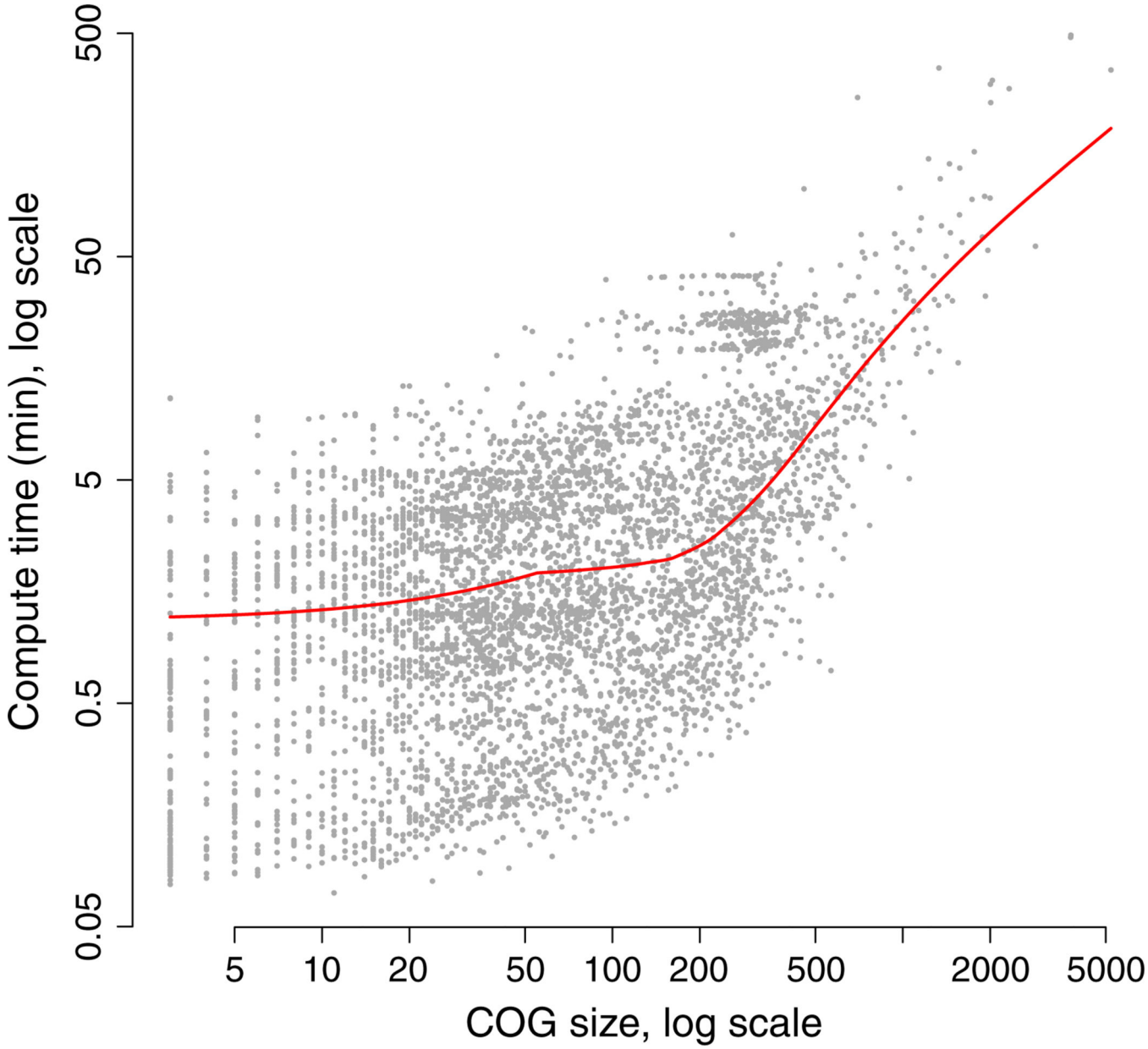

Figure 2.
Profile compute time as a function of COG size with LOWESS smoother overlayed (red). Both size and time are shown on natural log scale.

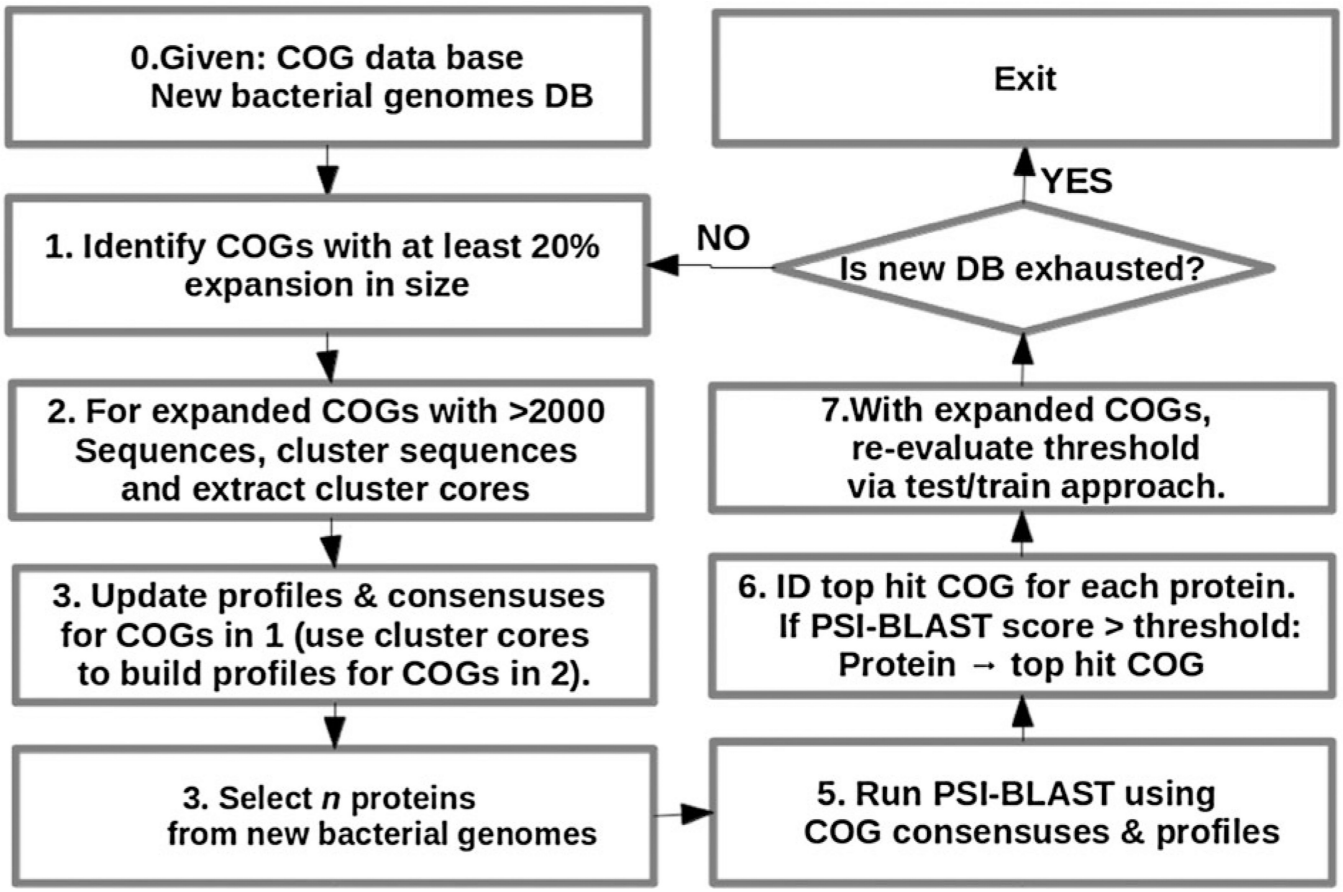

Figure 3.
The algorithm to classify newly sequenced genomes into existing clusters.

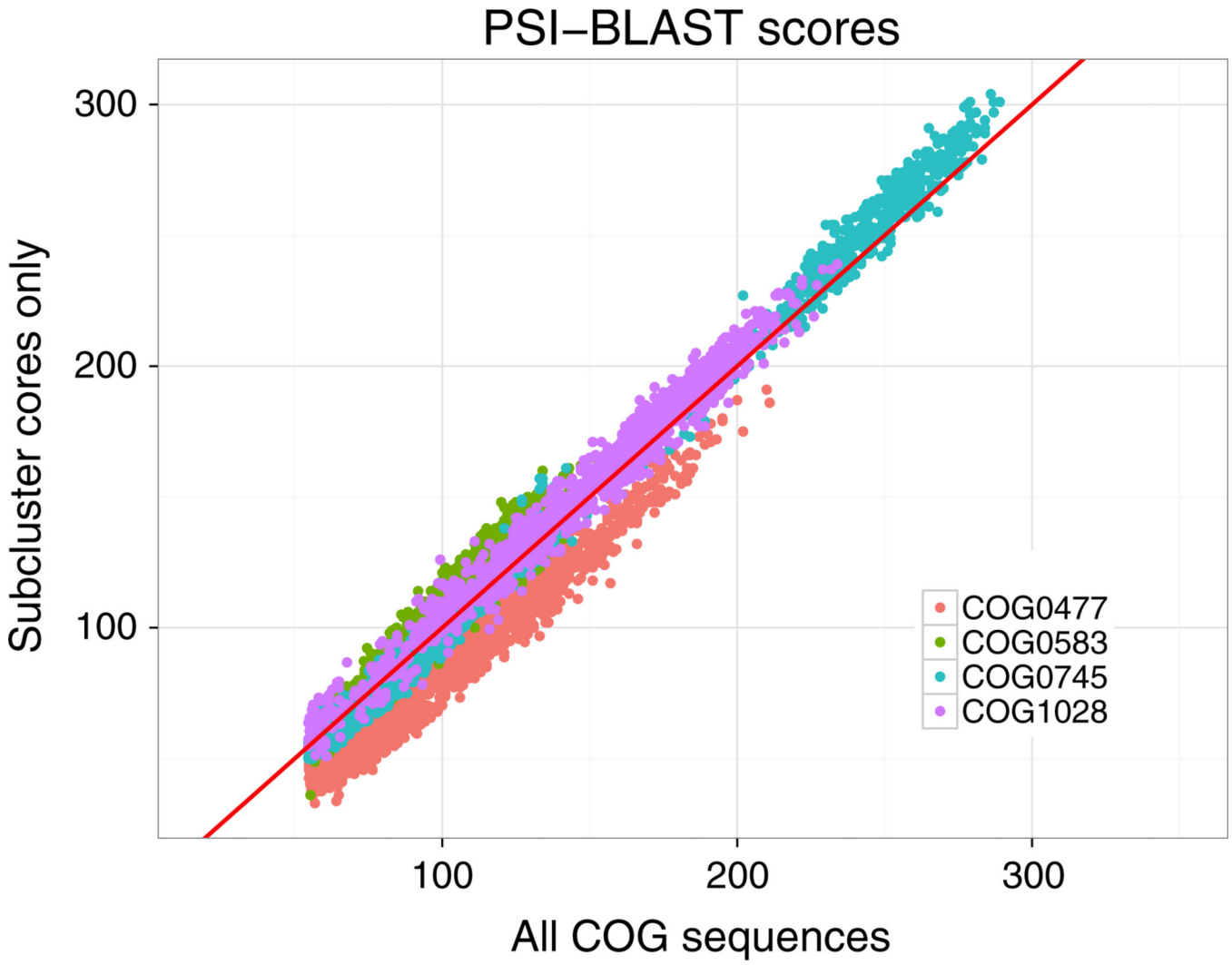

Figure 4.

PSI-BLAST scores based on profiles computed using the entire COG cluster (horizontal axis) versus profiles based on subcluster cores only. Shown are the subcluster-based score above the classification threshold of 4.49 (on log scale) for iteration 4. The *y* D *x* line is overlayed in red.

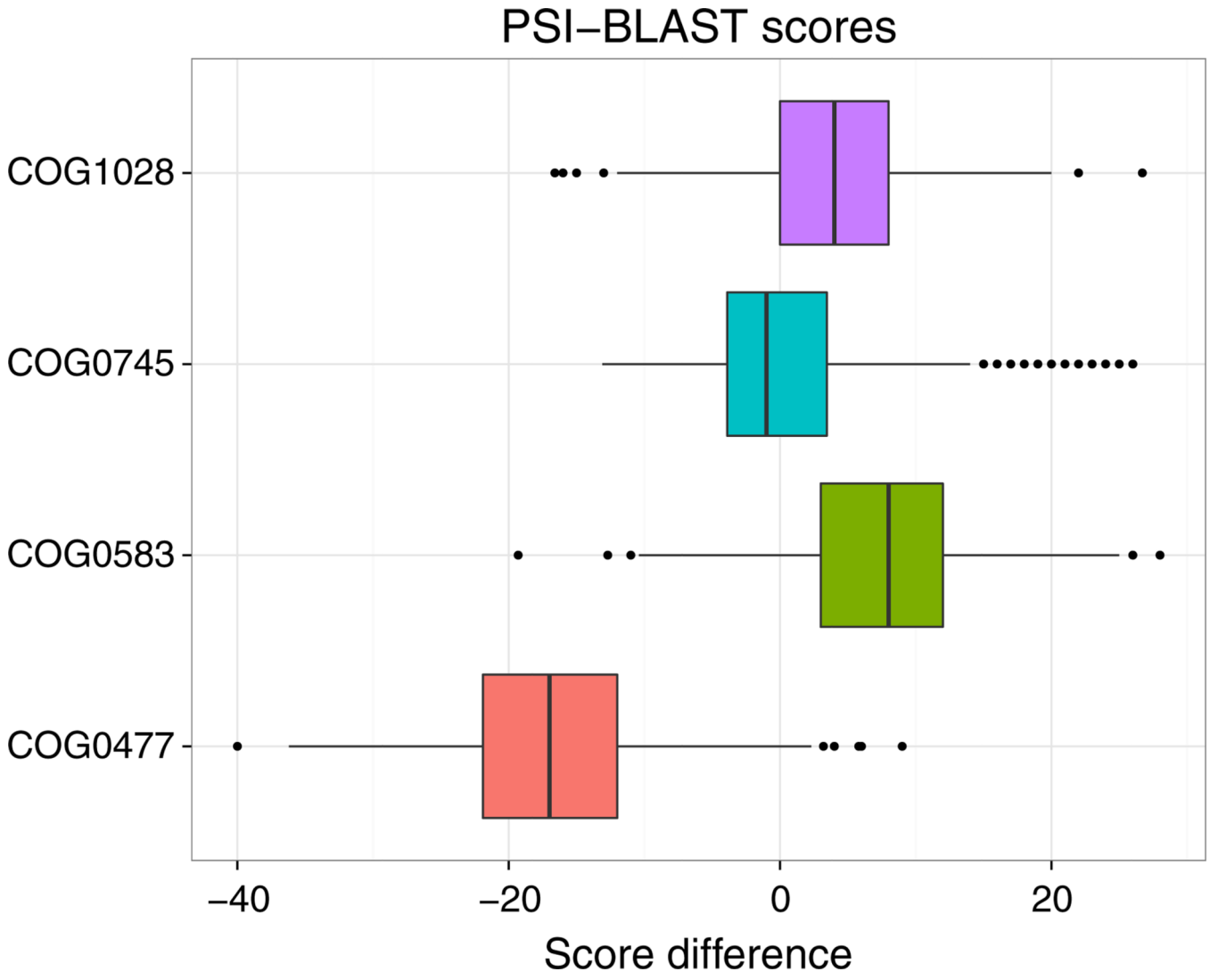

Figure 5.
Distribution of PSI-BLAST score differences for using the subcluster cores versus the entire COG.

Table I

Evaluation of the profile compute time with COG preclustering.

				Change in	
COG	**Size**	**Time**	**Subcluster no.**	**Time**	**Score***
0477	3234	417	1569 (45%)	−65	−16.7
0583	4208	253	1910 (51%)	−72	7.8
0745	2205	48	643 (71%)	−31	0.5
1028	3794	453	1490 (61%)	−84	4.0

Table 1 legend (left to right): (1)COG number; (2)Number of sequences in COG; (3) Time (min) to compute profiles using all cluster sequences; (4) Number of subclusters identified in each COG (% of the total cluster size); (6)Change in time (min) when computing profiles using subclusters versus the whole COG cluster; (7)Change in the mean alignment score for calculations based on subcluster profiles versus the whole cluster;

* all changes were statistically significant with p-value $\ll$ 0:001.

Table II

Iterations 1–5: number (%) of proteins analyzed classified by the new method and the original COG database.

		By new method	
		Yes	**No**
Original COG	Yes	18,373 (62%)	4034 (14%)
	No	676 (2%)	6593 (22%)