Whole-genome assays have now become pervasive, and the computational challenge of managing such large data collections is substantial. In order to mine these data efficiently, it is necessary to develop methods that use storage, memory and processing resources carefully.

Results: The Sleipnir C++ library implements a variety of machine learning and data manipulation algorithms with a focus on heterogeneous data integration and efficiency for very large biological data collections. Sleipnir allows microarray processing, functional ontology mining, clustering, Bayesian learning and inference and support vector machine tasks to be performed for heterogeneous data on scales not previously practical. In addition to the library, which can easily be integrated into new computational systems, prebuilt tools are provided to perform a variety of common tasks. Many tools are multithreaded for parallelization in desktop or high-throughput computing environments, and most tasks can be performed in minutes for hundreds of datasets using a standard personal computer.

Availability: Source code (C++) and documentation are available at http://function.princeton.edu/sleipnir and compiled binaries are available from the authors on request.

Contact: ogt@princeton.edu

1 INTRODUCTION

Whole-genome assays have now become pervasive, and the resulting wealth of data represents a new opportunity for biological discovery. A single genome-scale dataset can capture a snapshot of cellular function; integrative analysis of hundreds or thousands of genome-scale datasets can provide even more extensive systems-level insights regarding gene interactions under diverse conditions (Troyanskaya, 2005). Integrated approaches have already resulted in important biological discoveries (Hong et al., 2008; Myers and Troyanskaya, 2007), and the breadth and depth of possible analyses will only increase as additional experimental data is collected.

As the amount of data to be analyzed continues to increase, computational efficiency becomes a greater concern. Specialized resources exist to enable very high-throughput computing for specific applications (Pekurovsky et al., 2004; Swindells et al., 2002), but few computational options exist allowing researchers to quickly mine large collections of genome-scale datasets.

To address this need, we have created the Sleipnir library for computational functional genomics. The library contains algorithms and data types for efficiently manipulating and mining very large biological data collections. The core C++ library can be integrated into computational systems to provide rapid analysis of functional genomic data. Additionally, a variety of tools are provided that use the library to perform common tasks: microarray processing, Bayesian and support vector machine (SVM) learning and so forth. Even when analyzing individual datasets, Sleipnir often outperforms existing utilities in processing time, memory usage or both (Table 1).

Tools provided with Sleipnir address common data manipulation requirements, in many cases processing hundreds of datasets on a standard desktop computer. Additionally, the core Sleipnir library can be easily employed to efficiently apply new algorithms to complex biological data.

2 METHODS

The Sleipnir library contains a wide variety of tools for consuming standard biological data formats, manipulating and normalizing data and performing machine learning and prediction. These are discussed extensively in the user and developer documentation included with the library (http://function.princeton.edu/sleipnir) and are presented here in summary.

Sleipnir provides C++ classes to parse pairwise interaction data and standard microarray file formats. Microarray data can be converted into pairwise similarity/distance scores using a variety of measures, discretized, normalized, randomized for bootstrapping or synthetic data production, split or merged, imputed or clustered.

To facilitate functional enrichment analysis, gene function prediction and gold standard generation from known gene functions and relationships, Sleipnir provides a uniform interface to several organism-independent function annotation catalogs. Information from organism-specific annotations can be merged with these functional annotations. Sleipnir also includes collections of data structures for dealing with common biological entities: gene identifiers, sets of genes, groups of related files, etc. Other utility classes include resources for multithreading, a ready-made network client/server class and a variety of mathematical and statistical tools.

Sleipnir provides several tools for rapid machine learning and data mining. The SMILIE Bayesian network library (Druzdzel, 1999) and the SVM Light (Joachims, 1999) library are used to learn and evaluate Bayesian or SVM models from very large collections of biological data. Arbitrary Bayesian structures are allowed, with parameters learned either discriminatively or generatively (Greiner and Zhou, 2005) from data in a context-specific
Table 1. Sleipnir efficiency on integration and single dataset tasks

| Implementation | Peak RAM (KB) | Time (s) |
|----------------|---------------|----------|
| Bayesian learning (500 genes, 15 datasets) | Sleipnir 1376 | <1 |
| | GeNe 6832 | 4 |
| | BNT 593180 | 15 |
| Bayesian inference (500 genes, 15 datasets) | Sleipnir 1216 | 1 |
| | BNT 273992 | >600 |

Memory usage and runtimes for Sleipnir and a number of other common tools for Bayesian analysis and biological data manipulation (de Hoon et al., 2004; Druzdzel, 1999; Murphy, 2001; Saté et al., 2003; Troyanskaya et al., 2001). All microarray operations were performed on the 300 conditions and 6153 genes of (Hughes et al., 2000) using Euclidean distance. Bayesian operations were performed on simulated data using a binary gold standard and five randomly distributed values per dataset. Tests were run in a single thread on a 2 GHz Intel Core 2 Duo. In every case, Sleipnir demonstrates a substantial advantage in speed, memory usage or both.

3 RESULTS

While Sleipnir’s efficiency in integrating and mining biological datasets is most critical for very large data collections, it is also practical for single dataset tasks and smaller analyses (Table 1). When compared to several common tools for microarray manipulation or Bayesian learning, Sleipnir consistently demonstrates a substantial advantage in runtime, memory usage or both. These improvements arise from a variety of optimizations but are broadly attributable to the flexibility allowed by C++ in manipulating large quantities of individual data (microarray values, interaction pairs, etc.) What Sleipnir trades off in generality (e.g. with respect to BNT) or robustness to malformed input (e.g. with respect to MeV), it gains in speed, memory management and overall scalability, allowing it to efficiently manipulate large data collections.

The Sleipnir library is particularly useful for large integration tasks involving hundreds of diverse biological datasets; example applications of Sleipnir in such settings include Huttenhower et al. (2006) and Myers and Troyanskaya (2007). A schematic of such a task is shown in Figure 1, where Sleipnir was used to learn 200 context-specific Bayesian classifiers each integrating 186 Saccharomyces cerevisiae datasets. Conditional probability tables were learned for each dataset within each context, entailing ∼75 000 probability distributions. The resulting Bayesian classifiers were used to infer context-specific functional relationship networks, each consuming 90 MB of disk space and calculated in 16.3 min. Sleipnir also supports an online mode for functional relationship inference in which no additional disk space is consumed and individual context-specific functional relationships can be produced in as little as 100 ns. Parallelization on four processor cores reduces the total learning and evaluation time by an optimal 4-fold speedup (∼13 h each for Bayesian learning and inference). Every stage of this complex data integration and machine learning task was performed using Sleipnir and its associated tools.

4 DISCUSSION

The Sleipnir library for computational functional genomics provides a wide range of data processing and machine learning algorithms optimized for integrating very large collections of heterogeneous biological data. These include algorithms for data integration, machine learning by Bayesian networks or SVMs, and data types for manipulating microarrays, gene identifiers, functional annotations and other common biological entities. Several tools are provided with the core library to perform common tasks, and most algorithms are multithreaded or parallelizable for distributed computing. The Sleipnir library enables computational biologists to efficiently integrate thousands of genomic datasets and to rapidly mine them for biological knowledge.

ACKNOWLEDGEMENTS

We thank Matthew Hibbs and Chad Myers for helpful discussions and code reviews.

Funding: This research is supported by NSF CAREER DBI-0546275, NIH R01 GM071966, NIH T32 HG003284 and NIGMS Center of Excellence P50 GM071508. O.G.T. is an Alfred P. Sloan Research Fellow.

Conflict of Interest: none declared.

REFERENCES

de Hoon, M.J. et al. (2004) Open source clustering software. Bioinformatics, 20, 1453–1454.
Druzdzel, M.J. (1999) SMILE: Structural Modeling, Inference, and Learning Engine and GeNe: a development environment for graphical decision-theoretic models.
The Sleipnir functional genomics library

In Proceeding of the Sixteenth National Conference on Artificial Intelligence. AAAI Press/The MIT Press, Menlo Park, CA, pp. 902–903.

Greiner,R. and Zhou,W. (2005) Structural extension to logistic regression: discriminative parameter learning of belief net classifiers. Mach. Learn., 99, 297–322.

Hong,E.L. et al. (2008) Gene Ontology annotations at SGD: new data sources and annotation methods. Nucleic Acids Res., 36, D577–D581.

Hughes,T.R. et al. (2000) Functional discovery via a compendium of expression profiles. Cell, 102, 109–126.

Huttenhower,C. et al. (2006) A scalable method for integration and functional analysis of multiple microarray datasets. Bioinformatics, 22, 2900–2907.

Joachims,T. (1999) Making large-scale SVM learning practical. In Schölkopf,B. et al. (eds) Advances in Kernel Methods – Support Vector Learning. MIT Press.

Murphy,K.P. (2001) The Bayes net toolbox for MATLAB. Comput. Sci. Stat., 33.

Myers,C.L. and Troyanskaya,O.G. (2007) Context-sensitive data integration and prediction of biological networks. Bioinformatics, 23, 2322–2330.

Pakanovskiy,D. et al. (2004) A case study of high-throughput biological data processing on parallel platforms. Bioinformatics, 20, 1940–1947.

Qureishi,A.I. et al. (2005) TM4: a free, open-source system for microarray data management and analysis. BioTechniques, 34, 374–378.

Swindells,M. et al. (2002) Application of high-throughput computing in bioinformatics. Philos. Trans., 360, 1179–1189.

Troyanskaya,O.G. (2005) Putting microarrays in a context: integrated analysis of diverse biological data. Brief. Bioinform., 6, 34–43.