Research proposal

Better living through computing—efficient *in silico* simulations of biological phenomena

Institution:
Department of Computing, University of Copenhagen
Project duration:
1.9.2007-31.12.2009
Principal investigator, library development:
Jyrki Katajainen, Assoc. Prof. (University of Copenhagen)
Principal investigator, computational biology:
Lars Yde, Consultant (Indepenza)
Other investigators:
N. N., Ph. D. student
N. N., research assistant
Søren Gade, M. Sc. student
Claus Jensen, Ph. D. student
Anders Kristian Lyhne Thøgersen, M. Sc. student
Foreign partners:
Torben Hagerup, Prof. (University of Augsburg)
Peter Sanders, Prof. (University of Karlsruhe)
Sibylle Schupp, Assoc. Prof. (Chalmers University of Technology)
Jukka Teuhola, Prof. (University of Turku)
Jesper Larsson Träff, Senior Principal Researcher (NEC Europe Ltd.)

Contact details:
Jyrki Katajainen (team manager)

Department of Computing
University of Copenhagen

Worldwide Web: `http://www.diku.dk/~jyrki/`
Postal address: Universitetsparken 1, 2100 Copenhagen East, Denmark
Facsimile: (+45) 35 32 14 01
Telephone: (+45) 35 32 14 16
E-mail: `jyrki@diku.dk`

CPH STL Report 2007-2, April 2007.

Abstract. Since 2000, the CPH STL research team at the Performance Engineering Laboratory has been a nexus of program-library development in the C++ programming language as well as practical and theoretical algorithmics research, evident by the many published papers, reports and high-standard implementations available online at the CPH STL website.

Key areas of research have been runtime optimization, genericity in data structures and algorithms, and achieving optimality in implementation as well as proving theoretical bounds on optimality. The CPH STL has, however, also made strides in automated benchmarking and documentation, and is now mature for application to industrial-scale problems.

One specific target for application is the E-Cell system—a platform that seeks to achieve precise whole-cell simulation. The growth in life-science datasets and knowledge complexity, exemplified by the human genome and more recently the human metabolome, underscore the need for continuous performance improvement in biological simulation software, and the E-Cell system is no exception.

We have two goals in this project: to ship the industry-grade version of the CPH STL, and to use it in the E-Cell system as a runtime performance booster. We thus hope to use our expertise to aid in progress in drug discovery, metabolic pathway identification and *in silico* simulation of human physiology.

1. Background

The marriage between the life sciences and computing has grown ever more intimate since the publication of the human genome [5] in 2001, and the flurry of research that has followed in its wake [2, 4, 10]. So much so that in some respects, such as protein folding [7] and sequence alignment, one could not exist without the other. The mutually beneficial partnership is, however, not as recent as one might be led to believe by a simple perusal of research papers and projects over the last two decades.

As early as 1952, Alan Turing did pioneering work in mathematical modelling at Manchester University.His paper, *The chemical basis of morphogenesis* [9] published that same year, gave a mathematical footing to aspects of organism development, and Turing actually implemented his ideas on what was the world's first commercial computer, the Mark I at Manchester University. More recently, popular distributed computing projects such as the Stanford-based Folding@Home [7] have achieved microsecond, all-atom simulations of protein kinetics, i.e. the dynamics of how chains of amino acids fold into protein shapes—a computationally expensive and pharmacologically useful area of research.

The gap in years and scope between Turing's early work and massively parallel online systems such as Folding@Home belies the fact that they ultimately have similar goals: a faithful rendition of biologically relevant systems *in silico* and utilization of the predictive powers of true-to-life models. Another similarity between these two efforts, separated in time though they are, is the constraint placed on both by limitations in computing technology. Even in 2006, with a 200.000+ number of participating computers, the

$500\mu s$ simulation of a small protein was a major accomplishment for the Folding@Home project [6], and Moore's law not-withstanding, performance is still a limiting factor in computational biology.

Since autumn 2000, the Performance Engineering Laboratory (PE-lab) at the Department of Computing at the University of Copenhagen has been engaged in efforts to evolve the C++ Standard Template Library (the STL)[1]—a significant portion of the C++ standard library mandated by the official C++ standard [1]. The aim of this undertaking has been to provide an enhanced open-source implementation of the STL that is optimized for performance, along with accompanying benchmarking information and tool support, both of which have actively researched by PE-lab investigators during the course of the project. Also, the PE-lab investigators have made significant contributions in theoretical and practical computing as the project has unfolded, using the CPH STL project as a fulcrum of research in both areas.

The synergistic effects of high-quality research and ambitious implementation by skilled developers has resulted in a solid base of code and associated tools for benchmarking and documenting code execution and performance. This code base requires integration and packaging work to reach the level of maturity needed for a 1.0 release. In essence, what is required is a consistently applied investment of time and effort to streamline the implementation artifacts already produced, and if necessary, complete what remains to achieve a stable release version. Doing this will allow the CPH STL to enter into the arena of real-life problems, both industrial and academic, and thus offer up the potential for significant performance gains wherever C++ is used for non-trivial applications. A fully-fledged implementation should not, however, stand alone. It should be supported by systematic benchmarking, published analyses and, perhaps most importantly, actual application studies of sizable projects where the CPH STL library has had a marked impact on performance. To the CPH STL investigators, the area of computational biology offers a well-suited candidate for such a case study, namely the E-Cell system [3] which is a product of international collaboration.

The E-Cell project began in 1996 at the Laboratory for Bioinformatics at Keio University SFC [3], and the system is at release version 3.1 at the time of this writing. The system provides a C++/Python platform for script-driven, user-defined simulation of complex systems based on an object-oriented, extendible architecture [8]. Although generic in scope, the platform is intended for, ultimately, whole-cell simulation, but is used for a variety of other computational biology applications [3]. Currently, a planned version 4.0 promises to support multi-spatial representation and industry-supported visualization technology[2] among other things, underscoring the continuing demand for space and time performance optimization to allow for more detail and scope in modelling.

[1] These efforts were quickly named *The Copenhagen Standard Template Library*, or CPH STL for short. Further information about the project is available online at the website `http://www.cphstl.dk`.

[2] `http://www.e-cell.org/ecell4workshop/workshop-results/schedule-minutes`

2. Goals and objectives

The overarching goal of this project can be summarized as follows: to deliver *measurable space/time performance gains* to the E-Cell C++ runtime system through a *release version of the CPH STL*, and to use the *adjunct benchmarking* results to *bolster the case* for adopting the CPH STL in present and future performance-sensitive C++ projects.

This does not mean that the choice of the E-Cell project was an afterthought, tangential to the project. On the contrary, the choice of case study specimen was based on three criteria: technical eligibility[3], social relevance[4], and scientific potential. Successful completion of the “Better living through computing” project will translate into direct, measurable contributions in the scientific, social and technical arenas, precisely because the object of study has a solid footprint in each of the aforementioned.

Timeline and deliverables

To illustrate the above point, consider that the most conspicuous deliverable, the CPH STL 1.0 release that is the prerequisite for an *in vivo* case study, will in no way be the only one. The timeline in Figure 1 shows both the proposed chronology of the project and the deliverables to be produced.

Two timelines are included, for clarity, to illustrate the parallelism inherent to the project. Each year will be punctuated by a comprehensive annual report, and regular workshops will be held at meaningful times throughout the project—the exact scheduling will depend on project findings and developments as well as fortuitous timing with international conferences and events, e.g. E-Cell project workshops which are held regularly.

The primary objective will thus be the fulfillment of the main deliverables:

– the CPH STL 1.0 beta and final releases,
– the CPH STL 1.0 report documenting the internals of the program library,
– the E-Cell/CPH STL integration and profiling release and report, and
– the annual and periodical project and findings reports.

In addition to these, further publications are planned throughout the course of the project, but their timing and volume will depend on the project findings, and the manpower allocated to project tasks. We have indicated in the timelines where we expect to have time and material for publications other than those indicated explicitly.

[3] The E-Cell runtime core is written in standard C++, and the system has obvious performance sensitivity since improvements translate directly into larger potential problem sets.
[4] Whole-cell simulation has direct application in pharmacology, drug kinetics analysis, and biology as a whole.

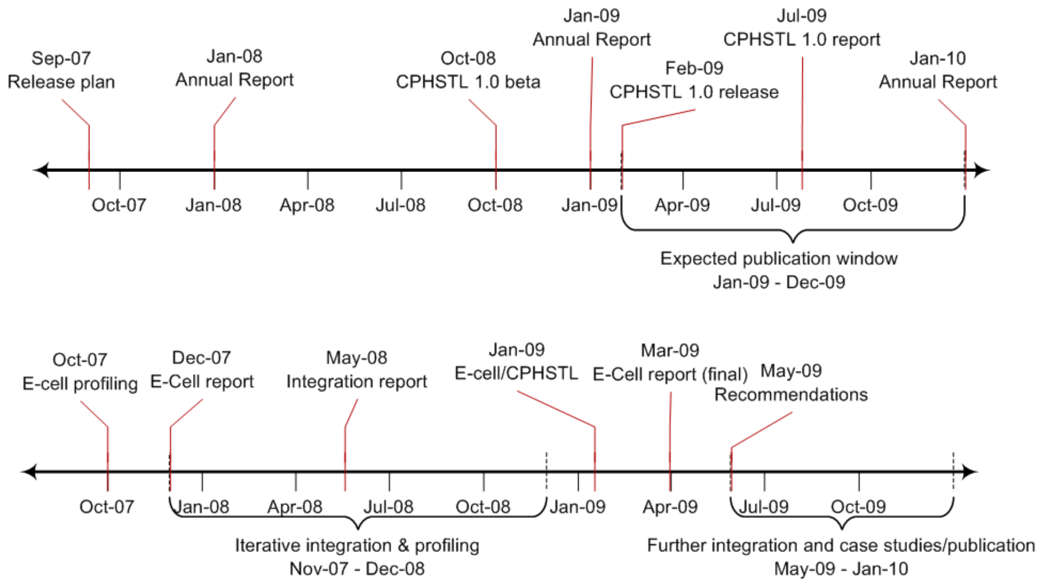

Figure 1. Project timeline. Top: CPH STL release, Bottom: E-Cell integration and further case study.

Funding

Achieving the goals described requires a substantial investment of time and effort from the investigators, but obviously also an infusion of funding. To be specific, this project requires funding for:

1. A full-time programmer for one year to completely achieve the CPH STL 1.0 release version.
2. A scholarship for a Ph. D. student who helps to prepare the sizable CPH STL 1.0 report in collaboration with the team manager and other investigators. Also, he or she will participate in implementation and packaging of the CPH STL 1.0 deliverable, and the theoretical work that must be anticipated in conjunction with the project.
3. Travelling expenses for project investigators, and for relevant visitors to the project.
4. Expenses for hosting workshops, purchasing literature and hardware/ computer access for benchmarking.

The budget shown in Appendix I gives the estimated figures for each of the above items.

Appendices

A: Sibylle Schupp, Evaluation of the CPH STL project
B: Jukka Teuhola, Brief evaluation of CPH STL project

C: Jyrki Katajainen, Curriculum vitae
D: Jyrki Katajainen, List of recent publications (2000-2007)
E: Lars Yde, Curriculum vitae
F: Søren Gade, Curriculum vitae
G: Claus Jensen, Curriculum vitae
H: Anders Kristian Lyhne Thøgersen, Curriculum vitae
I: Detailed project budget

References

[1] British Standards Institute, *The C++ Standard: Incorporating Technical Corrigendum 1*, 2nd Edition, John Wiley and Sons, Ltd. (2003).

[2] Cytomics.info, Human cytome project, cytomics & systems biology, Website accessible at `http://www.cytomics.info` (2007).

[3] E-Cell Project, Welcome to E-Cell project, Website accessible at `http://www.e-cell.org` (2003–2007).

[4] Genomic Disorders Research Centre, The human variome project: International collection of human gene variation, Website accessible at `http://www.humanvariomeproject.org` (2007).

[5] International Human Genome Sequencing Consortium, Initial sequencing and analysis of the human genome, *Nature* **409** (2001), 860–921.

[6] G. Jayachandran, V. Vishal, and V. S. Pande, Using massively parallel simulation and Markovian models to study protein folding: Examining the dynamics of the villin headpiece, *The Journal of Chemical Physics* **124** (2006), Article 164902.

[7] S. M. Larson, C. D. Snow, M. R. Shirts, and V. S. Pande, Folding@Home and Genome@Home: Using distributed computing to tackle previously intractable problems in computational biology, *Computational Genomics: Theory and Application*, Horizon Press (2004).

[8] K. Takahashi et al., E-cell 2: multi-platform E-Cell simulation system, *Bioinformatics* **19** (2003), 1727–1729.

[9] A. M. Turing, The chemical basis of morphogenesis, *Royal Society of London Philosophical Transactions, Series B* **237** (1952), 37–72.

[10] D. Wishart et al., HMDB: the human metabolome database, *Nucleic Acids Research* **35** (2007), D521–D526.