Advanced computing for systems biology

Kevin Burrage, Lindsay Hood and Mark A. Ragan
Submitted: 6th May 2006; received (in revised form): 29th August 2006

Abstract
Systems biology is based on computational modelling and simulation of large networks of interacting components. Models may be intended to capture processes, mechanisms, components and interactions at different levels of fidelity. Input data are often large and geographically disperse, and may require the computation to be moved to the data, not vice versa. In addition, complex system-level problems require collaboration across institutions and disciplines. Grid computing can offer robust, scalable solutions for distributed data, compute and expertise. We illustrate some of the range of computational and data requirements in systems biology with three case studies: one requiring large computation but small data (orthologue mapping in comparative genomics), a second involving complex terabyte data (the Visible Cell project) and a third that is both computationally and data-intensive (simulations at multiple temporal and spatial scales). Authentication, authorisation and audit systems are currently not well scalable and may present bottlenecks for distributed collaboration particularly where outcomes may be commercialised. Challenges remain in providing lightweight standards to facilitate the penetration of robust, scalable grid-type computing into diverse user communities to meet the evolving demands of systems biology.

Keywords: high-performance computing; computational modelling; multi-scale simulation; Visible Cell; orthologue mapping; systems biology

INTRODUCTION
Understanding biological systems is one of the grand challenges in science. There is clear evidence that computation is increasingly informing, and in turn being informed by, the biological sciences [1–4]. In this article, we outline some of the issues in modelling biological systems, provide observations on distributed computing and software development and present three case studies illuminating facets of advanced computing for systems biology.

LARGE-SCALE COMPUTING ISSUES IN SYSTEMS BIOLOGY
What is systems biology? Definitions differ, but most focus on how biological ‘parts’ (e.g. molecules, cells, organisms) interact to produce coherent, dynamic systems (cells, tissues and organs, ecosystems). It is rarely obvious how systemwide behaviour emerges from these networks of interacting parts, and some definitions focus on properties that we associate with living systems, including resilience, adaptation and evolution. In any case, we need to know the
component parts and at least the general nature of their interactions, and represent them symbolically. In this way, complex biological systems can be described in mathematical or statistical terms. It may additionally be possible to bring the perspectives and analytical methodologies of computer science, systems engineering, control theory and complexity theory to bear on these systems.

However, systems biology is defined and delineated, it is based on computational modelling and simulation [1, 2, 5]. In systems biology applications, computational modelling and simulation require access to large, diverse, semantically rich and often geographically disperse data sets. These data are integrated into computational models, which are then executed on computing hardware, often generating large volumes of output. This output must be managed, validated, mined and visualised, typically by researchers representing multiple disciplines (statistics, complexity theory, molecular or organismal biology). A range of issues must be addressed throughout this process:

- Data, both as input and output.
- Mapping the problem to a model.
- Embedding the model in a scalable computational framework.
- Implementing the model in real computer code of well-engineered software.
- Executing the code on hardware of appropriate architecture.

Embedding the model in a scalable computational framework is where the abstract view of modelling meets the reality of modern computing architectures. It is safe to assume that almost all computing for systems biology in the next few years will be done using variations of the current digital silicon-based technology: quantum and DNA computing, while of interest, are still very much experimental for real-world problems [4]. The classical, physics-dominated supercomputing of the 1980s has given way to a more diverse array of architectures. This shift has been driven by the economics of the semiconductor industry, where microprocessor design is expensive, but unit fabrication costs are low; hence, only high-volume components are economically feasible (bespoke processors, such as the Cray X1, tend to be for niche markets with wealthy customers). Biology has also contributed to this change because the majority of biological computation is about capacity (many small tasks to solve the problem) rather than capability (few large tasks to solve the problem). An example of this is genome assembly from shotgun sequencing: many small pairwise overlap calculations, followed by a single assembly of the overlaps typically requiring very large memory. Because different parts of a computational problem may be best suited to quite different detailed architectures, it may be attractive to assemble a heterogeneous collection of computers which can be coupled to solve a problem; this is the premise of the TeraGrid [6]. Space does not permit a detailed discussion on the advantages and disadvantages of such a system, but the reader is encouraged to explore documents on the TeraGrid Web site [6].

Here we discuss the philosophy of modelling as it applies in particular to systems biology, then follow with three case studies related to our own work. Case study 1 has large computational requirements but small data, and is in part (but not entirely) almost trivially amenable to a computing grid system. Case study 2 presents large, complex data requirements requiring the integration of disparate datasets, and thus has affinity for Web services and data grid techniques. It also offers some software development lessons. Case study 3 brings together the above two: it is both computationally and data-intensive, and moreover requires the implementation of difficult algorithms.

MODELS, COMPLEXITY AND REQUIREMENTS

Having stated that systems biology is about creating models that are concerned with both entities and interactions, one needs to ask what we want to achieve by modelling the system—is it to understand, to predict or to control? This matters because it will crucially determine what fidelity the model requires, and this in turn has consequences for the data and computational requirements for executing that model on real computer systems. The fidelity of a model is the degree to which it captures the processes or mechanisms, the entities and their relationships that we believe occur in the real system. Fidelity is relative to both our level of understanding of the system, and the end use we intend to make of the model.

The lowest-fidelity modelling is essentially the prediction and use of patterns. In this approach,
one simply takes an observed pattern and carries out some kind of statistical analysis to predict its future state. A classic example is trying to predict the future of the stock market using a time series analysis of historical data. In this, we completely ignore any external factors that have affected the historical record, and so we have, a priori, no reason to believe that the future state of the system is necessarily analogous or related to the historical state. This is not to suggest that pattern prediction has no value—if the future and historical states are analogous, then future states can be predicted quite accurately. Pattern prediction can also help us gain understanding and insight into the process that is producing the pattern.

The next step up in the fidelity ladder is what is usually called empirical modelling. With empirical models, the results or consequences of applying an underlying process are captured as a simple set of rules or observations that are then used in the modelling process. For example, consider a model of plant growth. A ‘first principles’ model might, for instance, consider factors such as physiology, water availability, solar radiation and photosynthesis, atmospheric phenomena and soil nutrients. Clearly, a large amount of biophysical process and data must be captured.

The highest level of fidelity is achieved with process-based models. In these, processes that we believe occur in the real system are captured or modelled ‘one to one’ in our model of that system at a level appropriate to the use intended for the model. As we move to finer spatial and temporal resolution, more processes may have to be included explicitly. As a consequence, all modelling contains some degree of empiricism. There is nothing inherently wrong with this, but one must be aware of the assumptions and limitations on which the model is predicated. The output from any model is not delivered on tablets of stone, but must be interpreted or analysed by the modeller, relative to its assumptions and limitations, including data required in the model. Visualisation of raw and modelled data is an important tool in this process.

The scale of modern ‘omic’ biological data makes providing a centralised data warehouse for all information relevant to a model and its analysis increasingly difficult. There are both technical and social reasons for this. Thus, advances in understanding biological systems will require us to learn how to work in distributed environments, and collaborate with coworkers who may be geographically disperse. Fortunately there is evidence of technological advances towards resolving this—the Grid.

The Grid—in the sense of an easy extension of the user’s desktop environment to utilise distributed data and computing services—has been publicised for over a decade [7], and yet to-date has made few inroads into the daily life of working scientists. We believe that will change over the next few years, largely due to related developments in the business world, with concepts such as Web services, software as a service and virtual organisations. Whether or not it is still called ‘grid computing’ and is built on technologies such as Globus, there is no doubt that scientists will need to learn to work in a world where resources are distributed [4].

In particular, we believe that data grid approaches will significantly benefit science, and that moving the computation to the data rather than the other way round (as is typically done today) will be required to make sense of all the biological data being generated and incorporate it into systems models.

CASE STUDY 1: RECOGNISING ORTHOLOGOUS NETWORKS VIA PHYLOGENOMICS

Systems biology is a study of ‘parts’ and their interactions. Even now, a decade or more into the genomic age, systems-biological investigations must often begin by making an inventory of relevant parts and interactions. These may be incompletely known, or indeed completely obscure, and must be discovered from primary data. For this we can seek help from comparative genomics. Features—e.g. sequences of genomic regions or the composition of regulatory networks—are similar in two or more organisms due to the interplay between genealogy and function. Function constrains diversification, so homologous features that remain more similar than expected under a non-selective model are likely to be functionally important, i.e. to be ‘parts’ that engage in relevant interactions. At finer scale, conserved sequences or structures may point to how these parts interact. Discovering conserved features is complicated, however, not only by diversification through time but also by genome-level events such as duplication, translocation and deletion of genomic regions. Fine-scale feature
mapping requires us to identify and compare only precise evolutionary counterparts (orthologues), and this turns out to be computationally intensive.

Orthology (common descent from a speciation event) can be unambiguously distinguished from paralogy (common descent from a gene duplication event) only on a phylogenetic tree [8]. The heuristic equating orthology with reciprocal best BLAST match [9] can be a useful first approximation, but is inadequate for fine-scale analysis especially with large gene families. Pathways and networks can be considered evolutionarily identical (and assumed, in the absence of evidence to the contrary, to be functionally identical) in different organisms if orthologues constitute their vertices. Finding maximal common subsets shared between networks, and the related problem of retrieving networks from databases, are known to be NP-complete. Here, we deal with the prior issue of establishing orthology among gene or protein sequences, based on the large-scale inference of trees [10].

Given the set of genes encoded by the genomes of interest (or alternatively, the proteins conceptually translated from these genes), the first step in large-scale orthologue mapping is to recognise putative gene (or protein) families. We begin with an all-versus-all BLAST, an embarrassingly parallel application that simply requires many processors; memory is generally not an issue (RAM requirement is a linear function of query length plus a linear function of maximum target length), and optimisation has been well explored. Sensitivity might be improved by use of more-rigorous statistical search operations, e.g. based on the Smith–Waterman algorithm with affine gap penalties, although at substantial CPU cost.

N × N BLAST yields N² pairwise vectors (edges), among which we next identify clusters of related edges. Even after removing statistically insignificant edges, clustering can pose a large-memory problem for which current Grid architectures are inadequate. Markov clustering [11] is the method of choice because it avoids false-positive matches caused by statistically over-represented substrings (‘promiscuous domains’), but its RAM requirement scales as number of matrices held in memory × cell size × graph cardinality × a large integer that, in turn, depends on resource settings [12]. A popular implementation [11] allows resource requirements to be tuned (at the cost of deviation from ‘pure’ clustering), but RAM requirements in the tens of gigabytes are not uncommon. For some applications, it is possible to post-process these clusters to extract putatively orthologous families, but the polynomial-time approach [10] may be too conservative for fine-scale orthologue analysis among complex genomes (e.g. those of vertebrates), and larger (paralogous) clusters might better be taken forward into subsequent steps.

Having delineated sets of homologous DNA or protein sequences at the granularity appropriate for the biological question, we next align each set to display homology position-by-position. Even pairwise alignment is NP-hard, but diverse algorithms are available that, in practice, yield high-quality multiple sequence alignment (MSA). However, it is rarely possible to know a priori which approach or parameterisation best suits the data, and a reasonable strategy is to apply a spectrum of alternative approaches, then select from among the resulting alignments the one that scores most highly according to an external criterion [13]. This strategy is well suited to distributed computing, although approaches that involve word (motif) discovery have memory requirements that scale as alphabet size to the power of motif length, so RAM can become limiting especially with protein or codon alphabets.

Given a set of aligned sequences, we next want to construct a hypothesis about their evolutionary history—i.e. infer a phylogenetic tree—on which orthologues can be identified. There is much to recommend a Bayesian approach [14, 15], in which the posterior probability of a tree is associated with its probability of being correct, given the prior probability, evolutionary model and data. Posterior probabilities usually cannot be computed analytically, but using Markov chain Monte Carlo (MCMC) methods, it is possible to examine equilibrium distributions of trees, and from these make probability statements about the true tree. MCMC, particularly with so-called heated chains (Metropolis-coupled MCMC), recovers from the posterior probability distribution a sample of topologies within which the empirical relative frequency of a given topology converges to its corresponding marginal posterior probability; the topology with highest relative frequency in this sample is selected, and posterior probabilities of its subtrees can be estimated by consensus from the topologies visited by the Markov chains. In current
implementations [16], memory requirement scales as Markov chains \times sequences \times site patterns \times character states \times rate categories \times 32, which for sets of 50–100 sequences translates to 0.5–1.0 GB. The Markov chains can be run in parallel, but as chain number is typically 4 and many data sets are often analysed simultaneously, the advantage of parallelisation may be modest except for very large data sets. Orthologous families should appear as coherent, statistically well-supported subtrees, and with appropriate indexing can be extracted in linear time. Comparing sets of orthologues can be more difficult, as many relevant operations are known or suspected to be NP-complete.

All the preceding steps, from delineation of gene or protein families through inference and analysis of trees, should ideally be refined by iteration. This is almost never done, both because the computational demands presented by real-life multi-genome problems are so substantial, and because workflow and data management can itself be complex. There is little reported experience in interfacing generic workflow and portal environments [17–20] with the distributed, multi-architecture high-performance computing resources required for multi-genome orthologue mapping as described here.

CASE STUDY 2: THE VISIBLE CELL

The Visible Cell project [21] is an ambitious undertaking of the Institute for Molecular Bioscience (IMB) and the ARC Centre in Bioinformatics. It is intended to provide a framework in which abstract mathematical and computational models can be embedded onto real three-dimensional cellular structural data obtained from high-resolution electron imaging. It is not intended to produce a general-purpose framework for systems biology modelling, but rather is predicated on the belief that, in order to unravel the complexities of the cell, biologists must be able to visualise the three-dimensional spatial arrangement of cellular components at high resolution, link gene expression and protein structures with cellular location and embed models of cell processes. Other cell representation and modelling environments [22–25] are similarly motivated, although without the same emphasis on real cell-image data.

This systems-oriented view, emphasising interactions as well as components, offers a good mapping to object-oriented analysis and design, which has emerged as a popular programming paradigm for designing complicated systems [26]. Although there have been increasing uses of more agile software development methodologies and languages (e.g. eXtreme Programming, Perl/PHP web scripts, Ruby) that emphasise rapid code development and could be considered somewhat counter to object-oriented methods, we believe there is a productive analogy between object-oriented software (and ultimately, software components) and biology modelled as components and interactions.

One of the first milestones of the Visible Cell is three-dimensional visualisation of a mammalian pancreatic beta .Cell at approximately 5 nm resolution, using data from tomographic reconstruction of electron microscope images. This presents the first challenge—data volume. A complete reconstruction will require between 5-10 TB of volume data, and we wish to visualise these data at multiple scales. Data of this volume cannot be rendered on anything other than extremely expensive visualisation clusters, and we desire to make these tools accessible to as wide a range of researchers as possible. This introduces the second challenge—we want the framework to be as platform-agnostic as possible.

We have addressed these issues in two ways: (1) a routine to tile and provide multiple levels of detail of raw microscope images stored in an Oracle database. Using Oracle Intermedia resulted in excessive image-loading times, so we wrote our own software in Java to run as a stored procedure in the Oracle database. Although Java can be slow, the relatively simple class structure required by our application imposes little need to traverse the class hierarchy with expensive function calls. (2) We implemented a Java viewer that interacts with the database to return the relevant tiles and view level of detail. Java implementation allows cross-platform viewing with no application recoding.

As part of the fundamental data management requirements for this project we are developing an object-oriented framework, called .Cell, that exposes various interfaces that data objects must implement if they are to be interoperable with components in the project. This framework provides a consistent interface to the different microscope sources at IMB, all of which have (different) proprietary data formats. The .Cell framework remains in active development, and is being documented so researchers can add their own tools (e.g. sophisticated visualisation environments) simply by conforming to the framework interfaces. This implies that we should keep
changes to interfaces to an absolute minimum as the project develops.

In addition to cell-image data, there will be other data types and sources:

- Segmented features extracted from the volumetric information and spatially registered to the volumes (they are represented as three-dimensional geometries) so they can be displayed simultaneously.
- Three-dimensional protein structures from structure databases.
- Protein localisation information, allowing us to embed protein structures in the volumes and segments.
- Models whose outputs can be spatially registered caricatures of cell structures, thus allowing for embedding in real spatial coordinates.
- Hyperlinked annotation and other metadata.

These disparate data sources are best accessed through a layer of abstraction, so the implementation specifics of each data source can be hidden from the application. We are applying technologies such as web services, data grid services and Storage Resource Broker [27] and find them to be important in this context. Ultimately, the models themselves should be available as grid services, allowing us to apply distributed computing resources—an approach entirely harmonious with the current trend towards ‘software as a service’ in business processing.

It is worth emphasising that software engineering issues must not be overlooked, as software systems required to make progress in even individual systems biology problems are becoming increasingly complex, and development of a generic systems biology framework is well beyond the capabilities of a single person ‘programming in the small’ [5]. This is a major reason for not constructing a general systems biology framework in this project—we simply do not have the resources to make significant headway with this approach.

CASE STUDY 3: MULTI-SCALE SIMULATIONS IN CELL BIOLOGY

It is increasingly appreciated that many cellular processes, including gene regulation and signalling cascades, are governed by effects associated with small numbers of certain key molecules [28–31]. In these situations the standard chemical kinetics framework described by systems of ordinary differential equations breaks down. The stochastic simulation algorithm (SSA) of Gillespie [52] represents a discrete modelling approach that attempts to model the uncertainty (intrinsic noise) in knowing when a reaction and what reaction takes place. Essentially, the SSA evolves a discrete non-linear Markov process whose elements are the numbers of molecules of all the chemical species in the reacting system as a function of time.

The SSA has been applied to good effect in a variety of settings [28, 33, 34] but its main drawback is that the time step can be very small if some molecular species are present in large numbers or if rate constants are large. A number of algorithmic approaches have been considered to overcome these computational bottlenecks, e.g. by allowing more than one reaction per time step [35, 36], using quasi-steady-state assumptions to reduce the size of the problem [37–39], or partitioning the problem into small, moderate and large numbers of molecules [40].

The need for advanced computing methodologies and implementations arises in this setting, as often thousands of independent simulations must be computed in order to collect appropriate statistics about moment behaviour. Because they are independent, these simulations can be distributed to different processing elements. Burrage et al. [41] give a grid implementation of Gillespie’s tau-leap method [32] in which MathML within a web browser is used to set up the chemical reaction system in an intuitive visual manner. The web browser then communicates with the networked computers running independent simulations. Numerical results, statistics and graphs are displayed back within the browser.

The SSA is a trajectorial approach, but since it describes the evolution of a stochastic process, this stochastic process has a probability density function described by the so-called Master Equation (ME). The ME is a system of linear ordinary differential equations in which there is one equation for each configuration of the state space. Consequently, the solution of the ME takes the form

\[ p(t) = e^{At}p(0) \]

where \( A \) is the state-space matrix. Even for relatively small systems, the dimension of \( A \) can be in the millions, so advanced computing techniques are necessary. Usually these techniques are based on domain decomposition ideas, but the art is to make sure that the computation-to-communication ratio is kept high. A proposed finite state projection
algorithm [42] reduces the size of the matrix $A$, and Krylov subspace techniques [43, 44] can be used to efficiently compute the exponential of a matrix times a vector.

The challenge is to apply these techniques in modelling important and complicated cellular processes. Particular examples include:

- Quorum sensing among proliferating cells, where a small change in protein expression levels in one cell can change the state of all other cells in the culture [45]. Here the computational challenge is to model genetic regulation within a cell, coupled to signalling among thousands of proliferating cells. Since cell number grows, adaptive techniques are needed to ensure suitable load balancing on the processing elements.

- Oscillations of calcium levels in cardiac cells play a major role in the stimulation of cardiac cells, resulting in contraction of the heart [46]. Calcium-induced calcium release occurs around the dyadic cleft and must be modelled at a fine scale by discrete models, which are then coupled to larger-scale dynamics.

- Epidermal growth factor (EGF) receptors play important roles in growth and differentiation in many mammalian cell types. Binding of EGF to the extracellular domain of the EGF receptor induces dimerisation, autophosphorylation and the activation of downstream proteins in signalling cascades [47]. Complex ordinary differential equation models have been developed involving 125 reactions and 94 species [48] but there is a need to model this from a discrete, stochastic perspective.

In addition, as soon as spatial effects are incorporated into the modelling process (there is a wide array of complicated ultrastructure within a cell), computational complexity increases dramatically. In these situations domain decomposition techniques are crucial, but again the art is in adaptively load-balancing the computational effort on the processing elements.

Other significant problems in systems biology have been identified not only at the cellular level, but also in the biology of tissues, organs and whole organisms. Progress will require linking experiments, observations, experimental data, models and simulation techniques across wide range of temporal and spatial scales. An especially challenging research project is the Physiome Project [49], in which the function of organs such as the heart is being modelled across a range of spatial scales of 15 orders of magnitude and temporal scales of nine orders of magnitude. Such integrative challenges will require not only grid computing [50] but also the use of markup language environments such as CellML [51] and additional computing methodologies such as computational steering.

**ACCESS AND INTEROPERABILITY**

As science becomes more global in its practice, with advances being made at the intersection of traditional discipline boundaries, the need to provide scalable authentication, authorisation and audit systems (AAA) to provide researchers access to resources becomes increasingly important. This is even more true with the life sciences, where there is an expectation that there will be real commercial gain through, e.g. improved therapeutic regimes. Systems such as traditional password files and LDAP will not scale, at least from a management perspective. We see AAA as a key problem to address, and note that there is work in progress on solutions such as Globus certificates [52] and shibboleth [53]. The (logical) separation of credential providers and credential consumers is a significant component in scaling these trust relationships to large-scale research communities.

We have touched on the value of published data standards and formats aiding data integration and presenting common interfaces for software applications. This, however, is only part of the story, as it addresses only syntactic data integration. The real goal is to have semantic interoperability and integration of disparate data sources and, more generally, of services. Technologies built around ontologies facilitating machine reasoning of data sets are the current solutions to this problem, with the concept being encapsulated with the vision of the Semantic Web [54] and current instantiations such as Web Ontology Language [55]. Other projects, such as BioMoby, have explored alternative solutions to this problem without the need for the construction of specific ontologies [56].

At the same time our collective challenge will be to make sure that such implementations are not overburdened with expensive, cumbersome frameworks and over-managed standards [57]. Occam’s principle (*pluralitas non est ponenda sine necessitate*)
should be a guiding light, with the caveat that these implementations should be sufficiently robust and portable to allow use by scientists with a wide variety of backgrounds and computing expertise.

Key Points
- In systems biology applications, computational modelling and simulation require access to large, diverse, semantically rich and often geographically dispersed data sets.
- These data are integrated into computational models, which are then executed on computing hardware, often generating large volumes of output.
- This output must be managed, validated, mined and visualised, typically by researchers representing multiple disciplines.
- With increasing data volumes, models and simulations must be made more sophisticated and computationally intensive.
- For these reasons, grid computing or its equivalent will continue to play an ever-increasing role in systems biology.
- Implementations for the grid must not be overburdened with expensive, cumbersome frameworks, and standards must be sufficiently portable to allow use by scientists with a wide variety of backgrounds and expertise.

Acknowledgement
We thank Prof. Werner Dubitzky for the invitation to discuss these issues and the Australian Research Council for support (ARC Centre in Bioinformatics CE0348221 and funding of K. B. through the Federation Fellowship program).

References