

## STATEMENT OF RESEARCH

By Jesús A. Izaguirre, Ph.D.

Donald E. Knuth, a famous Stanford computer scientist, says that “biology easily has 500 years of exciting [computer science] problems to work on.” Modern computational biology has the daunting task of interpreting enormous amounts of data. For example, it has been estimated that the prediction of the three dimensional structure and function of proteins discovered through the human genome will take at least ten times the computational and experimental efforts that generating the genome took in the first place. The scientific and societal impact of these projects is enormous, with the possibility of better understanding of diseases, computer-assisted design of medicinal drugs, bioengineering, and biologically-based material science.

My general research area is scientific computing, particularly multiscale modeling and simulation of biological materials, cells, and organisms. I develop algorithms and high performance software for three problems spanning the molecular, network, and cellular levels of biocomplexity: *simulation of proteins*; *analysis and prediction of protein interaction networks*; and *simulation of morphogenesis*, the structural development of an organism and its organs. These scientific problems serve as the source of important problems in computer and computational science. They also serve to validate the value of my work. My main contributions to computer and computational science are in the following areas: (i) design of hierarchical, multiscale, and *multilevel algorithms* to solve important computational problems such as the N-body problem, the numerical solution of stiff-oscillatory systems of ordinary differential equations (ODE), and the statistical sampling of complex probability density functions; (ii) development of *hybrid discrete-continuum simulation algorithms* and *software* to solve pattern formation and moving boundary problems; (iii) bioinformatics algorithms to predict protein interactions in genomes and coupling to the molecular scale; (iv) development of *recommender systems* that help users select algorithms or software with optimal configurations to solve their particular problem; (v) the implementation of these algorithms in *high-performance, flexible, open source software*, which uses modern software engineering techniques, domain-specific languages and compilers, and parallel and distributed algorithms; and (vi) the application of these algorithms and software, in multidisciplinary collaborations, to problems as varied as protein folding, the docking of medicinal drugs to flexible enzymes, the prediction of protein-protein interaction networks for genomes of medical importance such as *Plasmodium falciparum* (malaria), and the simulation of chicken limb development.

### Multiscale Algorithms

The research described in this section is funded primarily through an NSF Career award entitled “Scalable Mathematical and Computational Methods for Biomolecular Modeling,” for \$384,918 from Feb. 1, 2002 – Jan. 31, 2007. Results from this project have generated: 17 refereed publications (9 journal papers); an open source software framework called ProtoMol; two Ph.D. and two M.S. graduates; and one Ph.D. candidate and one Ph.D. student. Through NSF REU supplements 11 undergraduate students have participated in the research activities described here.

More recently, I obtained a grant from NSF Biological Databases and Informatics BDI-0450067 for \$809,367 for a three-year project entitled “Grid-enabled Integration of Experimental Data and Simulations for Flexible Protein Docking.” I am the PI, and Dr. Jeffrey Peng, an NMR experimentalist from Chemistry, and A. Striegel, an expert on grid computing from Computer Science and Engineering, are Co-PIs. This proposal contains preliminary results and new ideas to enhance multiscale sampling and integration in the context of docking to flexible proteins, which is an important biomedical application.

Concretely, molecular dynamics (MD) simulations of solvated proteins start from configurations describing the positions and velocities of each atom, typically involving tens or hundreds of thousands of particles. There are two main computational problems: one is to solve the *N-body problem*, that is, compute the forces experienced by each atom due to other atoms (for example, due to electrostatic Coulomb interactions), and the second is to advance the positions and velocities of the particles in simulation “time” using an “integrator” of Newton’s equations of motion (a large, stiff-oscillatory system of ordinary differential equations).

Typical MD simulations run for millions of steps and may require weeks to complete even in clusters or supercomputers. The bulk of this time is spent in solving the *N-body problem*. Direct computation of the forces leads to algorithms that require computational effort quadratic in the number of atoms in the system ( $N$ ), We have implemented a highly efficient parallel algorithm for a periodic system with linear complexity in  $N$ , cf. submitted manuscript Matthey *et al.* (2005). Periodicity is a way to mimic the much larger dimensions of protein systems within the cell. We have made this algorithm available to the research community, and it has enabled material science simulations of millions of particles not hereto practical; See Matthey *et al.*, *Physical Review Letters* **91**:165001 (2003). This algorithm is used by our collaborators; for example, E. Maginn in Chemical and Biomolecular Engineering is using it to simulate ionic liquids, and S. Meroueh in Chemistry is applying it to study the docking of medicinal drugs to proteins.

Faster but more sophisticated algorithms are harder to use than slower but simpler methods. Usually, parameters interact in complex ways. We have created a recommender system called MDSimAid, a web portal (<http://mdsimaid.cse.nd.edu>) which optimizes the parameters for sequential and parallel implementations of fast N-body solvers. Resulting simulations are up to 50 times more accurate, 20 times faster, and twice as scalable in parallel as simulations using suggested parameters in the literature. This is documented in Crocker *et al.*, *J. Comp. Chem.*, 2005. Our technology can be used to rank any set of algorithms or software in test problems according to different performance criteria. We are planning to expand this to eventually include the setup, optimization, and analysis of the whole MD simulation protocol.

The presence of multiple time scales in protein simulations is a major difficulty in integrating the system in time. The fastest scales (e.g., atomic vibrations in femtoseconds) require time steps that are too small to reach the simulation times of processes of interest such as the folding of proteins (at least a microsecond). Ideally, a *multiscale integrator* should be able to integrate fast time scales with short time steps and slow scales with much longer time steps, independent of the fast scales present in the system. I have built multiscale integrators that use so-called geometric integrators to allow longer time steps and faster simulations. Depending on the intended application, these integrators achieve between 50% and 400% speedup with respect to state of the art multiscale integrators. The main techniques used for stabilizing the integrators are the building of *average* representations of interactions, the addition of *randomness* to the numerical integrators, and the minimization of numerical artifacts. These techniques can be used for the solution of other stiff-oscillatory systems of ODEs common in science and engineering. For example, astrophysics simulations and computer animation can benefit from this research.

My Ph.D. thesis and Izaguirre *et al.*, *J. of Chemical Physics* **110**:9853 (1999) implement an algorithm called MOLLY that uses averaging of the fast motions to reduce instabilities. Through MOLLY we achieved a speedup of 50% over r-RESPA, a standard multiscale alternative. In Izaguirre *et al.*, *J. of Chemical Physics* **114**:2090 (2001) we carefully add mild damping and stochastic elements (randomness) to make the algorithm stable. This method achieves 300% speedup over r-RESPA. The Ph.D. thesis of my student Q. Ma (2003) and Ma and Izaguirre, *SIAM Multiscale Modeling and Simulation* **2**:1 (2003) further reduced numerical artifacts using a symmetric damping technique that achieves a 400% speedup over r-RESPA. Skeel and Izaguirre, *Molecular Physics* **100**:3885 (2002), completed work started in my Ph.D. thesis to produce a more robust numerical integrator for stochastic systems of ODEs. Theoretical research on the numerical mechanisms that cause instability sheds light on possible new avenues for multiscale integrators. For example, in Ma *et al.*, *SIAM J. on Scientific Computing* **24**:1951 (2003), we showed that there are nonlinear effects that severely limit the applicability of purely geometric multiscale integrators.

Another important application of MD is to *sample* the geometrical conformations of a biological molecule from an ensemble describing a probability density function of interest. The M.S. thesis of Hampton (2004) and Izaguirre and Hampton, *J. of Computational Physics* **200**:581 (2004), describe the Shadow Hybrid Monte Carlo (SHMC) algorithm for sampling of conformational space of biological molecules. This method overcomes the exponential degradation of performance with system size of hybrid MD-Monte Carlo (HMC) methods for sampling. Our improvement uses the theory of geometric

integrators to sample more efficiently from the modified Hamiltonian that is almost exactly solved by the numerical integrator. SHMC has an asymptotic speedup of the fourth-root of the system size over standard HMC. One of the reviewers of this paper described it as a “significant achievement” and a “great manuscript”. This method can also be applied to sample other complex probability density functions, for example for training of neural networks and data mining.

These results have led my research in multiscale MD on two avenues, which are currently funded by my CAREER award and the NSF DBI grant. The general procedure is that of A. Brandt (Weizmann Institute) and collaborators: small scale statistics with time-accurate dynamics at large scales. The first step is to construct coarse variables to describe MD and sampling. This is done using equilibrium simulations at the fine level; deriving coarse variables and doing equilibrium simulations at the coarse level, and then going back and forth between levels until convergence. In the case that convergence is not obtained, a different set of coarse variables is proposed and the process is repeated. We are using information coming from NMR experiments to aid the selection of coarse variables. The second step is to build multiscale integrators or phase space propagators given these coarse and fine models. We are using our own results, and those of other scholars such as Gear and collaborators to accomplish this. Our intended contribution to the whole process will be twofold: First, the building of multiscale models through coarsening protocols for MD that use of experimental data and simulations. Second, we will construct more stable and faster propagators and integrators for these multiscale models.

### **Multiscale, Multiphysics Modeling of Morphogenesis**

This research is funded by an NSF Biocomplexity grant entitled “Multiscale Simulation of Avian Limb Development” for \$3,000,000 from Sept. 2000 – Sept. 2005. I share 17% of the effort as a Co-PI/subcontractor. My portion of this research has generated: 8 refereed publications (5 journal papers); two open source software frameworks, CompuCell3D and Biologo; one postdoctoral research project; one M.S. graduate, one M.S. candidate, and two Ph.D. candidates. Through NSF REU supplements and other fellowships, 9 undergraduates have been involved in this research.

Morphogenesis involves cell differentiation, growth and migration, bulk changes in tissue shape (moving boundaries), and the secretion, resorption and diffusion of extracellular materials (e.g., proteins). We model morphogenesis at the cell level. This gives us a natural abstraction for computation and modeling that can be readily related to experiment. Cell-level modeling helps tackle complexity and makes modeling feasible. For example, a typical cell can contain roughly  $10^5$ - $10^6$  gene products. Estimating potential interactions between all of these products would be needlessly complicated and extremely expensive in time and space when run on a computer, even a highly parallel machine. By treating cells phenomenologically and ignoring intracellular behaviors we can reduce multiple complex interactions to a small set of behaviors such as movement, division, death, differentiation, shape change, force exertion, chemical and electrical charge secretion and absorption, and surface property distribution changes. On the other hand, macroscopic models like Physiome which operate at the level of tissues, while highly efficient, cannot reproduce some experimental observations which cell-centered models can.

Once the simulation results match experimental results, we study further test cases by, e.g. removing a parameter and viewing the changes in behavior. If we observe accurate behavior at the tissue level even though we have removed certain elements of the model, we have a smaller set of behaviors that can still accurately simulate experimental results. We can eventually obtain a minimal necessary set of single-cell behaviors through repeated testing. Then we can determine which gene networks drive this set of behaviors, and how they operate to more fully understand the role of gene networks in multicellular behavior. At this point, we can ask questions about the factors which result in abnormal growth - as we have done for a model of skeletal pattern formation which predicts how digit fusion happens (cf. Chaturvedi *et al.*, *LNCS* **3305:543**, 2004).

I have implemented a hybrid model where cell rearrangements are modeled using a lattice and the chemical signaling using PDEs. Our algorithms simulate and connect these processes. Chemical concentrations are projected onto the lattice of cells, which evolve according to a stochastic cellular automaton. On the other hand, cells secrete and absorb chemicals, hence affecting the continuum

treatment. This approach avoids instabilities of purely continuum models and has some of its speed advantages.

My group has implemented these algorithms efficiently in software for three dimensional simulations of morphogenesis, where efficient memory management is critical for performance. By using novel, specialized data structures and algorithms, we obtain a fourfold speedup and a tenfold memory reduction against standard alternatives. In collaboration with a biologist (S. Newman, New York Medical College), a mathematician (M. Alber, Notre Dame), and three physicists (J. A. Glazier, Indiana University; G. Hentschell, Emory University; and G. Forgacs, University of Missouri), we have validated the framework for various simulations of the formation of the skeleton of the chicken limb. These collaborators provided a realistic system of PDEs that model the biology, and experimental validation data. We are currently developing geometric algorithms to handle the growth of realistic shapes in morphogenesis in CompuCell3D. The goal is to validate our simulations against cell tracking experiments of real chicken embryo. This project has great potential for exciting collaborations in the burgeoning field of sensor networks.

### **Protein Interactions**

Proteins interact with each other in the cell. It is a central question in modern biology to discover not just one particular pathway, but to have an idea of the whole network of, for example, signaling pathways and metabolic networks. Ultimately, protein interaction networks provide a link between atomic and cellular processes and provide clues about function. In collaboration with D. Z. Chen from Computer Science and Engineering and S. Wuchty from Physics, we have designed an algorithm that uses pairwise protein interaction data and corresponding domain architectures of yeast proteins to identify a set of domain-pairs which explain observed interactions in a training set. Our algorithm predicts protein interactions in a testing set in yeast with a quality that outperforms existing algorithms. These results are documented in the submitted manuscript by Huang *et al.* (2005). We have also shown that by using clustering measures of the known protein interaction networks, we can dramatically improve the quality of our predictions, primarily by reducing the number of false positives. By using protein orthologs to yeast in higher order mammals and other organisms, we have been able to predict protein interaction network cores, from which further study can be performed. Our predicted networks show a strong tendency to co-expression and to evolutionary conservation. These results are being documented in Kanaan *et al.* (2005).

I am currently working on a proposal for NIH trying to link these bioinformatics predictions with validation using protein-protein docking (enhanced using our multiscale sampling algorithms), which we hope can be also validated through X-ray crystallography in a facility such as the NIH Structural Genomics at the University of Wisconsin-Madison. We have had talks about this validation with Dr. George N. Phillips, Professor of Biochemistry and Computer Sciences. They are interested in knowing which proteins need to be co-crystallized, which we can suggest based on the strongest predictions of our methods. Eventually, we want to perform genome-wide validation using the *computational, storage, and experimental* grid.

We are trying to make these methods applicable to the genome of malaria, in collaboration with Dr. Michael Ferdig from Biological Sciences at Notre Dame. Because experiments are very difficult on this parasite, the inclusion of simulation methods to discover potential protein domains, protein interactions, and pathways, will be extremely important for advancing knowledge about possible drugs, therapies, and how malaria develops drug resistance. All of these might be helpful in treating this disease, which cause 300 million acute illnesses and at least one million deaths annually (<http://rbm.who.int>).

### **Flexible and High-Performance Software Implementations**

I have disseminated our algorithms in well designed open source software that is easy to extend and also efficient. I believe that this practice facilitates reproducibility of results and validation of algorithms and simulations, besides making the algorithms more generally available to the potential user community. Flexibility and efficiency are two requirements that are in apparent contradiction. By a careful design that uses generic and object-oriented programming to hide optimizations such as parallel programming, and by the use of interpreted and compiled domain specific languages, we have been able to meet both

requirements simultaneously, cf. Cickovski *et al.* (2004b). Open source software packages enable users to validate their simulations by having access to the algorithms and source code.

ProtoMol contains support for developing fast parallel N-body solvers and multiscale integrators. It is described in Matthey *et al.*, *ACM Trans. on Mathematical Software* **20**:1 (2004). ProtoMol is one of the few *open source* MD simulation packages available. It has been downloaded 868 times from the Source Forge repository and is within the top quartile of open source software activity in the categories of bioinformatics and computational chemistry. Collaborators in Notre Dame and in Bergen, Norway are using it to implement new algorithms, as a tool in courses on computational science, and to produce new scientific results.

CompuCell3D (Cickovski *et al.* (2005); Izaguirre *et al.*, *Bioinformatics* **20**:1129 (2004)) implements the 3-D hybrid discrete-continuum morphogenesis model described above. This is the first open source code to simulate morphogenesis. Since its release in Source Forge it has been downloaded 350 times. Collaborators at Indiana University and New York Medical College are using CompuCell3D to simulate biological and physical processes in a variety of fields. Biologo (Cickovski and Izaguirre (2004)) is a domain specific language for morphogenesis. It enables scientists to extend CompuCell3D by writing an XML file describing their model, which gets compiled into C++ code. Users can describe multiple energy functions at a high level, cell types, state automata transitions, chemical fields, and arbitrary interactions among all these components. We have validated it through four vastly different simulations of morphogenesis.

Through our experience developing these frameworks we are producing a catalog of design patterns for scientific software, for which we are evaluating the performance and maintainability implications. This manuscript will be submitted to OOPSLA 2005 in mid March.

As part of the NSF BDI grant, I am collaborating with A. Striegel and D. Thain from CSE at Notre Dame to develop a grid-enabled database of molecular simulations. We are working to simplify parameter sweeps, the management of simulations, and improve the reliability of storing these very large data sets on a volatile environment such as the grid. This work is documented in Wozniak *et al.*, (2005a,b).

#### **SUMMARY OF TEACHING AND ADVISING**

While at Notre Dame, I have advised 2 postdoctoral research associates (and looking for a new one), and **graduated** 1 Ph.D. student (currently assistant professor of computer science at NJIT), 1 Ph.D. student in Norway (as his de-facto co-advisor), and 3 M.S. students (2 are continuing for the Ph.D.). Currently I am **advising** 2 Ph.D. candidates (expected to graduate in May, 2005), 3 Ph.D. students, and 3 M.S. candidates. I have served in the committee of students in other departments, such as chemical engineering. I am particularly committed to undergraduate research mentoring. I have directed 22 undergraduate research projects; 11 of these undergraduates have been co-authors on peer-reviewed conferences or journals, and in a few instances even the leading authors; 11 of these undergraduates are currently in graduate school.

I have also trained 20 undergraduates as teaching assistants for my courses. At least 13 of them have discovered or strengthened a teaching vocation and are in graduate school. This initiative benefits also the students taking the class, because they find it easy to communicate with their peers, who act as mediators between the more senior teaching staff and the students.

I am also concerned about encouraging minorities to pursue computer science, engineering, and scientific careers. I have been the faculty advisor of the Mexican American Engineers and Scientists (MAES) and the Society of Hispanic Professional Engineers (SHPE) since I began at Notre Dame in 1999. The students are organizing lecture series for the engineering and science students, organizing social events, and charitable events for the local community.

My philosophy of teaching is to give individual attention to students by incorporating different teaching styles and learning activities (e.g., group and individual assignments). In lectures I seek to give the context in which particular course contents are placed. For example, when teaching computer programming I spend two lectures giving them a history of how we have arrived at the concepts that are now commonplace in programming, and the last two lectures talking about how the content will apply in the rest of the curriculum and in their professional life. My methodology of teaching is to progressively

open black boxes. I like to show them a problem, an overall solution, and then descend to the details of each aspect that makes this solution successful. I also round up my presentations discussing alternative solutions.

I have tried to enable a collaborative learning environment through supervised final group projects. The students of CSE 212, Fundamentals of Computing II, and CSE 331, Data Structures, have had to give a presentation and write a paper describing their projects. Project papers and software deliverables for 7 offerings of these courses in the last 5 years, encompassing 98 final projects under my supervision, are compiled in <http://www.nd.edu/~cseprog>. This repository is a valuable resource for instructors and students of these courses at Notre Dame and elsewhere. *I plan to use this wealth of examples for an application driven textbook on computer programming.* I have used a similar approach in my graduate courses; where some final projects have turned into conference proceedings.

I received a faculty undergraduate teaching award in the spring of 2002, elected by the graduating class of Computer Science and Engineering students. I also strongly believe in assessment of course materials and delivery, a practice that is now institutionalized in my department. I have changed several aspects of my teaching using this feedback. Some of these convictions have come through hard lessons. For example, in fall 2002 I introduced a new programming language called Eiffel, which was a good vehicle for conveying important principles of software engineering. In retrospect, it was a premature decision, and caused a negative (and perhaps understandable) reaction from the students. However, as the final projects of that semester show, the students learned a great deal. Since that experience, my evaluation has recovered to its usual.

#### **SUMMARY OF SERVICE**

I designed a new undergraduate course in our curriculum, CSE 212, Fundamentals of Computing II. This course is an introduction object-oriented analysis, design, and implementation. It presents methods for building and testing reliable software, such as design by contract, and gives them a practical introduction to design. I also developed the accompanying lab sessions for this course. I have designed and taught several graduate courses in our department, including computational methods in molecular biology, computational biology, and numerical analysis. I was part of a committee to introduce a new biology course for all undergraduates in the college of engineering. I am part of a team of faculty from multiple departments teaching a new Seminar on Biocomplexity for freshmen this spring.

I served in the graduate committee from January 2000 – December 2004, with duties regarding the admission of graduate students and revision of academic policies. During 2001, I collaborated extensively with my colleagues in preparing the department for transition to a new Chair. I have served and I am currently serving in the undergraduate curriculum committee. We are preparing for an ABET accreditation.

I am founding member and in the executive committee of the Interdisciplinary Center for the Study of Biocomplexity. This center promotes scholarly activities such as Biocomplexity Workshops (the VII such workshop will occur this fall), seminar series, and the awarding of fellowships to graduate and undergraduate students. The center recently acquired a high performance cluster for biocomputing, funded by an NSF MRI grant.

I have served as reviewer for several panels for proposals to NSF and NIH and for top conferences and journals in scientific computing and software engineering, such as the J. on Computational Physics, the SIAM J. Scientific Computing, SIAM Multiscale Modeling and Simulation, and the IEEE Trans. on Software Engineering.

I was invited to deliver a plenary lecture to an NSF workshop on Cyber-Enabled Chemistry in October 2004. This workshop resulted in a new program solicitation in which I am planning to participate. I am a member of SIAM, IEEE, ACM, ASEE, and the New York Academy of Sciences, and look forward to expanded collaboration and service in these and other professional societies.