PROJECT SCOUT

MIT Laboratory for Computer Science

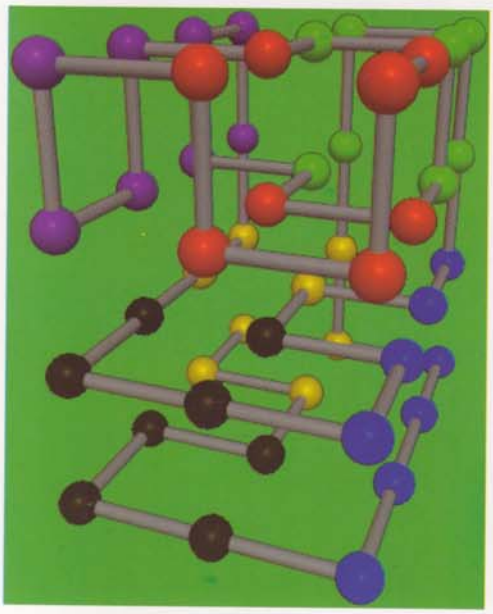
SCOUT is a unique collaboration at the MIT Laboratory for Computer Science tackling interesting, hard, computationally intensive, and heretofore intractable problems in science and engineering. Its purpose is to provide a symbiotic relationship among computer scientists and scientists from other disciplines, such as, aeronautics, astronautics, biology, mathematics, medicine, oceanography, physics, etc. By working together, each contributes to the other's success: the computer scientist testing the validity of new scalable computing ideas on real world problems, and the other scientist or engineer solving or gaining insight into the phenomenon being studied.

Due to its unique collaboration, Project SCOUT has pioneered some exciting advances in a variety of disciplines such as protein folding, plant and animal virus modeling, ocean modeling, modeling backflow contamination of ion-thrusters onto spacecraft, and planning radiosurgery of brain tumors, just to name a few.

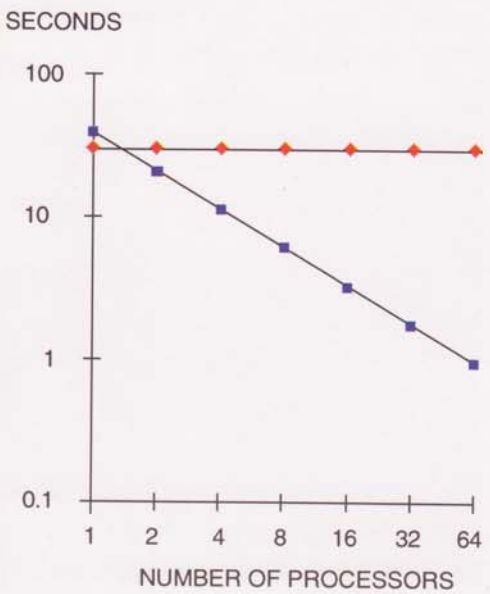
Protein Folding

Because a protein's structure defines its function, the understanding of how a protein folds into its unique 3-D structure is important to the manufacture of synthetic antibodies designed to kill specific viruses and to the creation of synthetic enzymes for people whose bodies are unable to manufacture them. The ***protein folding*** problem is being studied by building computer models that compute all possible legal configurations on a 3-D lattice (called conformations) and determining the energy states of each conformation.[1] The latter will determine how long it will take to make the protein, i.e., how long it will take to make the phase transition from the renatured to the denatured state. The biologist or pharmacologist would not be interested in a synthetic antibody that took 100 years to make.

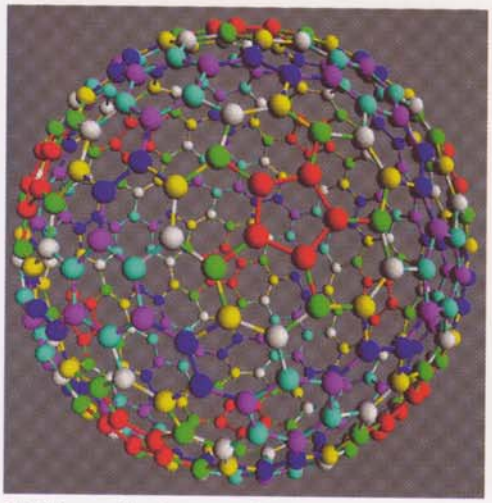
Two key contributions from computer science made it possible to compute for the first time 3x3x4 and 3x4x4 conformations as there are over a billion paths through the lattice for the former and a trillion for the latter making the problem heretofore intractable because of the computing time required. The first was an algorithm that aborted early on computations of illegal configurations.[2] A configuration is illegal if it crosses itself. The new algorithm is one to two orders of magnitude faster than the original. The second key contribution was the employment of a ***work stealing*** method to gain linear speedup with the number of processors used, yielding a speedup of over a 100 on a 128 node CM5.[3]



A 3x4x4 Folded Protein Cube



Runtime on a 3x3x4 Folded Protein Cube



Shell Assembly of a Normal Virus

Plant and Animal Virus Modeling

AIDS, herpes, and polio along with most plant and animal viruses have icosahedral-shaped shells just big enough to contain the genomic information, which is the RNA or DNA of their virus. Once a shell is complete with its genetic information, it is capable of infecting a cell. While nearly all previous research on interfering with the infection process has focused on how to prevent a fully-formed shell from binding to a cell, this work aims at modeling how these shells build in the hopes of eventually suggesting ways of interfering with their growth and causing deformity. If this can be achieved, the genomic information will not fit inside the shell, and the virus will not be viable.

One hypothesis as to how these structures form is based solely on simple local rules for how proteins interact. A software ***toolkit*** for modeling virus shells has been developed and used to show the hypothesis is viable.[4] The virus shells are modeled as an interconnection network of proteins, i.e., nodes, and their essential binding interactions, i.e., edges. The virus assembly *toolkit* has been used thus far to explore the tolerance margins of these shells and possible deformities such as a spiraling malformation observed in nature. Chemically the nodes or proteins are identical, however, they can assume different conformations which cause them to behave differently. By only communicating locally, each node in the network can be given enough information to uniquely form any size shell. This information consists of node conformation type, bond angles, bond lengths, and torsional angles. Our *toolkit* has indicated that bond lengths and bond angles can be varied up to 8 percent in every direction and the shell still closes. While a modern workstation is used to compute an undeviated shell, a scalable computer is needed to compute a perturbed one.

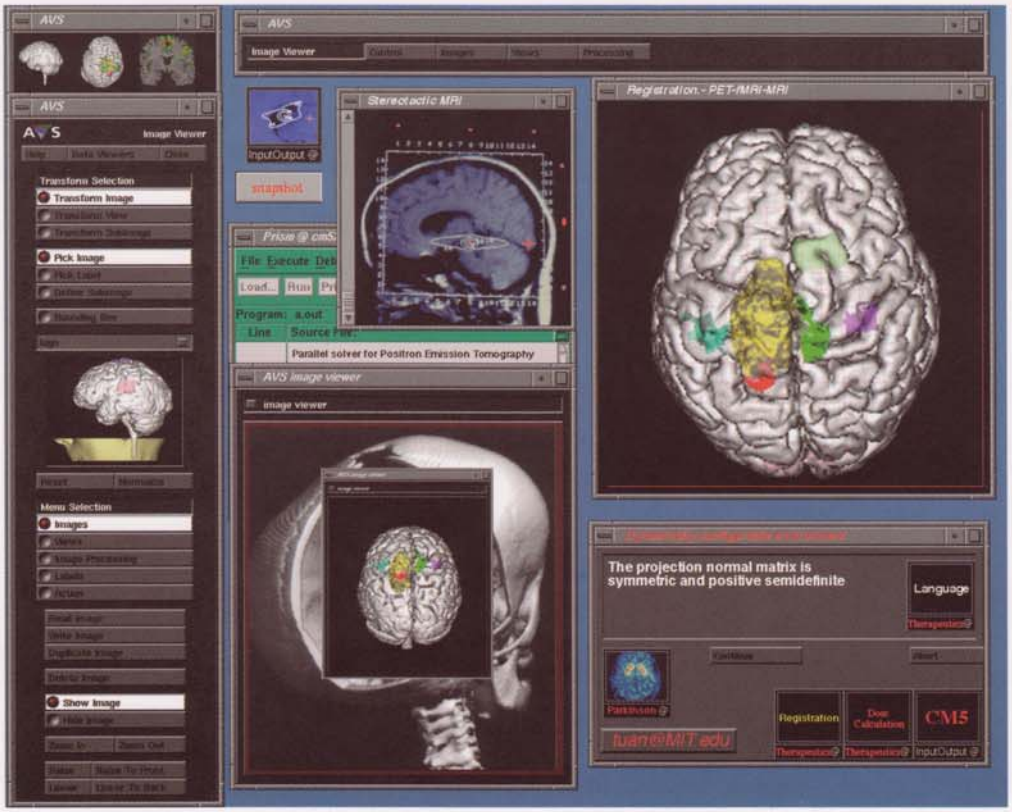
Neurosurgical Planning

Brain surgery for the removal of tumors often requires meticulous, time-consuming planning. Experiments with a set of **neurosurgical planning tools**[5], in a real world patient care environment, indicate that surgical planning time can be reduced by a factor of ten. The time it takes to plan a complex surgical procedure can be decreased from two days to fifteen minutes and for a radiation surgical procedure, electron or photon beam, or high energy charged particles, from two weeks to a day.

The neurosurgical tools allow for more precise therapy because: (i) a tumor can often change significantly in a couple of weeks and by shortening the time between detection and treatment, the uncertainty of its size and shape is diminished; and (ii) the tools allow employment of conformal beam shaping techniques for complex 3-D radiotherapy treatment.

These planning tools allow for more accurate tumor location because: (i) the registration function permits the matching of different frames of reference for mapping of brain function and target (tumor) locations by fusing data from several sources, i.e., CAT, MRI, functional MRI, and 3-D cylindrical PET scans; and (ii) several true 3-D reconstruction techniques, i.e., Fourier methods, back projection, and maximum likelihood estimation, can be used with our 3-D high resolution PET scan to more precisely measure, analyze, and register data for real-time interactive display.

These tools also provide information on the measurement instruments and measurement techniques, both experimental and those used in general practice, which will allow us to make suggestions for their improvement. A CM5 parallel processor is used for computation and a Silicon Graphics computer for visualization.

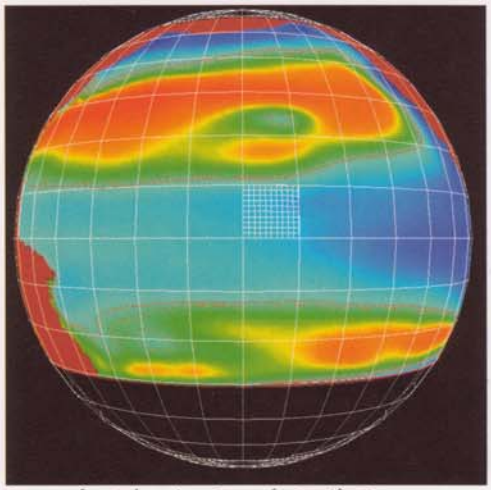


The figure above is a window snapshot showing measurement, analysis, and registration of multimodality data in individual subjects.

Ocean Modeling

Ocean models are used as tools to study, understand, and predict the state of the ocean and its role in the climate system. These models characterize the ocean by its distribution of currents, temperature, salinity, and pressure on a three-dimensional grid. The ocean's evolution over periods of days, months, and years is predicted by stepping forward in time the characterizing state variables in accord with the laws of classical mechanics and thermodynamics. To a large degree the realism that can be achieved is directly proportional to the resolution of the grid - which places requirements on memory size- and rapidity at which the model can be evolved - which is proportional computational speed of the computer.

Scientists in the MIT Department of Earth, Atmospheric and Planetary Sciences[6] are collaborating with computer scientists[7] in the Laboratory for Computer Science to develop a state-of-the-art ocean circulation model, based on the Navier Stokes equations, which exploits the new generation of massively parallel machines. Their ultimate goal is to develop a model with resolution sufficient to represent all the key aspects of the ocean's circulation. Monsoon, a dataflow computer designed at LCS, and Project SCOUT's CM5 have proven to be essential in this ocean modeling work.



Sea-surface Elevation Over the Pacific Ocean as Simulated on the CM5

Success has been achieved in articulating an ocean model in an implicitly parallel language called Id which runs on Monsoon and in a data parallel Fortran. The data parallel 'production' model has been implemented efficiently on the CM5. The algorithms are scalable and render a data parallel code that outperforms its serial counterparts. The model is fully operational and is already being used in high-resolution studies of the Pacific Ocean, Mediterranean Sea, and the convective process in the open ocean.

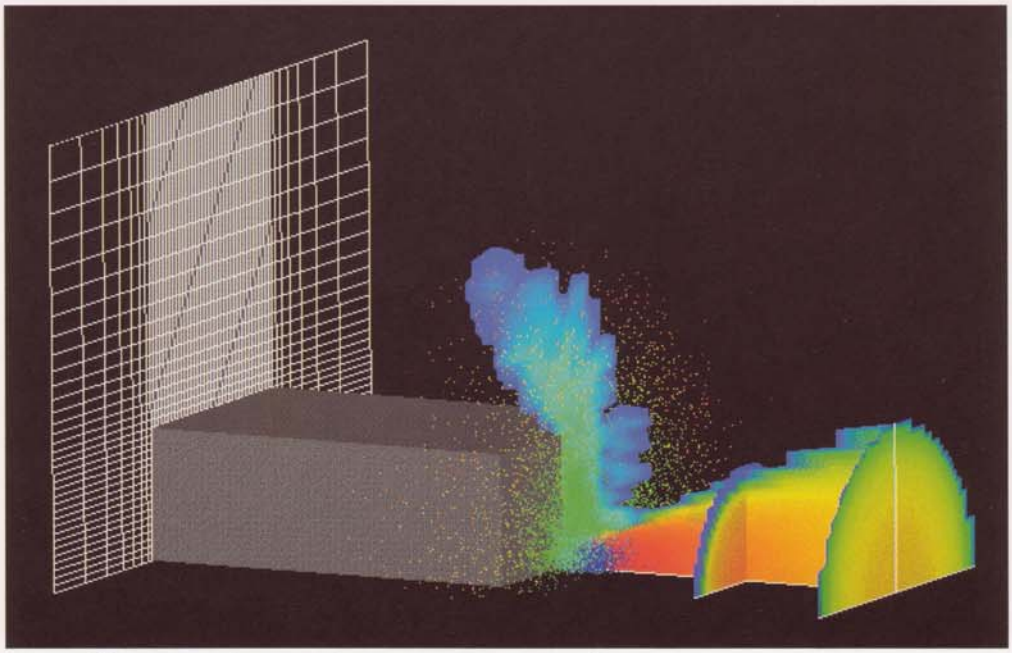
Backflow Contamination of Ion Thrusters Onto Spacecraft

Ion thrusters are propulsive devices with high fuel efficiency that are of current interest for spacecraft station-keeping on orbit. However, there is a component of the exhaust that can backflow onto the spacecraft. This backflow consists of charged and neutral species that constitute a potential contamination hazard to the spacecraft. It is of prime importance to estimate this hazard and both NASA and the DoD are interested in this problem. Large-scale computational simulations are the only method of attack since ground experiments are highly questionable due to the effects of finite-size facilities and their influence on a highly rarefied plasma.

A full three-dimensional plasma particle-in-cell (PIC) code to simulate the backflow contamination from ion thrusters has been

developed.[8] The PIC method treats a plasma as a collection of computational particles that move under the influence of self-consistent electric and magnetic fields. To simulate a full-scale spacecraft of meter length scale dimensions poses enormous requirements on the number of particles necessary to give statistically reasonable results.

Currently being investigated is the backflow of a 30cm ion thruster using Xenon as a propellant on the ELITE spacecraft. The computational domain consists of 9.5 million grid points and up to 100 million particles. Massively parallel computers are the only available resources that can handle such a problem that was previously inconceivable. Such problems do not fit on workstations as two gigabytes of primary memory is required. This code was originally developed and tested using the SCOUT 128 node CM5 (on a smaller scale problem). It is now in production on the 512-node Intel Delta at Caltech, and is in the process of being ported onto the 256-node Cray T3D at the Jet Propulsion Laboratory (JPL).



3-D *Backflow Structure from Ion Thruster over* ESEX *Spacecraft*

Advances in Computer Science

Cilk

Cilk is a parallel C runtime system and multiscale programming language.[9] It executes dynamic multithreaded computations efficiently by using a work-stealing scheduler to load balance the computations. Cilk implementations currently run on the CM5, the MIT Phish network of workstations, and the SPARC 20 SMP. Applications that have been coded in Cilk include protein folding, graphics rendering, backtrack search, and the *Socrates chess program.

The Cilk language makes simple extensions to the C language to support multithreaded computations. A Cilk procedure is broken into a sequence of nonblocking C functions called threads. Threads can spawn other threads that can run in parallel. Consequently, procedures (and their threads) are organized into a tree hierarchy. Data dependencies impose additional ordering constraints on the threads.

The Cilk runtime system employs a work-stealing scheduler that schedules the threads efficiently with respect to both time and space. Most Cilk applications exhibit an overhead of less than 15 percent compared with the best serial C implementations of the same applications. The work-stealing scheduler also minimizes space required by the execution. Typically, the maximum number of closures (runtime data structures for threads) extant at any time is only a few hundred per processor over a period of time in which millions of threads are executed. Communication is also minimized by the scheduler.

*Socrates

*Socrates is a parallel chess-playing program developed jointly with Heuristic Software, Inc.[10] The Cilk development team chose chess as an application to demonstrate their dynamic, MIMD work-stealing scheduler and runtime system. *Socrates is based on Heuristic Software's popular Socrates program, a serial program written in C, which was ported to Cilk in a little over a month's time. *Socrates typically achieves over a 200-times speedup of the serial code when running on 512 processors. The overhead is mostly due to the speculative work performed by Jamboree search, the parallel search algorithm in *Socrates that replaces the serial search algorithm in Socrates. Scheduling overhead typically amounts to only a few percent.

The chess application is challenging from a computer science point of view because of the dynamic nature of the computation. The computation is a dynamically growing and shrinking tree, with data dependencies that preclude certain computations from proceeding until other computations have completed. Because the Jamboree search algorithm used in *Socrates speculatively evaluates portions of the game tree, it occasionally discovers that the work of a subtree must be aborted. The Cilk runtime system supports the selective aborting of computations at user level. *Socrates also uses the Strata active message library to implement a global "transposition table," which caches recently seen positions.

*Socrates was developed using the SCOUT CM5. Running on the NCSA 512-node CM5, it took 3rd prize in the 1994 ACM International Computer Chess Championship.

Strata

Strata[11] is a multi-function library that augments TMC's message passing library (CMMD 3.0) and offers:

- Split-phase control-network operations;
- Support for debugging and monitoring, including graphics and a version of print that can be used in handlers;
- Higher performance for CMMD control-network operations.

PerfSim

Timing measurements of various basic CM-Fortran operations are used to formulate models that predict the running time of these operations using different language constructs. The models are used by PerfSim, an analysis tool that predicts the performance of CM-Fortran programs.[12] It works by executing the program without performing any operations on distributed arrays. Instead, it uses analytical models to predict the running time of runtime subroutines and analysis of the compiler generated code to predict the running time of nodal subroutines. PerfSim is useful for performance tuning of programs without expanding a lot of computing resources, and for monitoring the performance of the runtime system.

Phish

Phish[13] is a parallel job manager for a network of workstations that implements the Cilk dynamic thread scheduler and

incorporates several novel resource-sharing features. Cilk applications running under Phish are able to dynamically adjust their parallelism to exploit idle workstations through adaptive space sharing. In particular, when a workstation is left idle by its owner, it automatically consults the Phish job manager to find a running parallel job to work on. Later, when reclaimed by its owner, the workstation retreats from any Phish job. Thus, over the course of a job's execution, the set of machines working on the job expands and contracts based on the availability of idle workstations. Phish also provides transparent fault tolerance. It automatically detects crashed workstations and redoes work as necessary. With these features, Cilk applications running under Phish may run for long periods of time utilizing large amounts of compute resources.

Project SCOUT

SCOUT[14] is a consortium comprised of Boston University, Harvard University and the Massachusetts Institute of Technology. Its purpose is to foster an interdisciplinary collaboration among scientists to further the use of scalable computing technology. SCOUT is one of five founding members of the National Consortium for High Performance Computing (NCHPC). Other NCHPC organizations obtaining support for scalable machines include: three National Science Foundation (NSF) Supercomputer Centers, i.e., the National Center for Supercomputing Applications (NCSA), San Diego Supercomputer Center (SDSC), Pittsburgh Supercomputing Center (PSC); the Sandia National Laboratories; the Naval Research Laboratory (NRL); National Oceanic and Atmospheric Administration (NOAA); National Center for Atmospheric Research (NCAR); and the University of Colorado at Boulder.

To foster the collaboration among scientists of various disciplines the Laboratory for Computer Science hosts a SCOUT User Group which meets weekly to discuss timely issues concerning use of scalable computers. The Laboratory also runs a Supercomputing Technologies Seminar series with invited speakers discussing all aspects of supercomputing from parallel computer design to applications of scalable systems.

For more information about **Project SCOUT** please contact:

Thomas Greene, Manager, Project SCOUT
telephone: 617-253-5987
email: tjg@hq.lcs.mit.edu

Melissa LaBarge, Administrative Contact
telephone: 617-253-5852
email: melissa@hq.lcs.mit.edu

Acknowledgements and Collaborating Scientists

Work reported herein is supported by:
Project SCOUT - ARPA Grant MDA 972-92-J-1032
Protein Folding - NSF Grant DMR 90-22933
Plant & Animal Virus Modeling - ARPA/ONR Grant N00014-92-J-1799
Neurosurgical Planning - NIH Grant R01 CA32873-11 and DOE Grant DE-FG02-87ER60519
Ocean Modeling - ONR Grant N00014-92-J-1567 and ARPA/ONR Grant N00014-92-J-1310
Backflow Contamination of Ion Thrusters - JPL Grant 959948
Computer Science Advances - ARPA/ONR Grants N00014-91-J-1698, N00014-94-1-0985, and N00014-92-J-1310

[1]Toyoichi Tanaka, D.Sc., is a Professor of Physics, Alexander Yu Grosberg is a Visiting Scientist, and Vijay Pande is a graduate student, all in the MIT Department of Physics.

[2]Charles Leiserson is a Professor of Computer Science and Engineering in the MIT Department of Electrical Engineering and Computer Science (EECS) and a member of the Laboratory for Computer Science (LCS).

[3]Chris Joerg is a graduate student in EECS and a research assistant in LCS.

[4]Bonnie Berger is an Assistant Professor of Mathematics in the MIT Department of Mathematics and a member of LCS, Dr. Kenneth Jones is a member of LCS, Dr. Peter W. Shor is a member of the technical staff at AT&T Bell Labs, and Douglas Muir and Russell Schwartz are undergraduate students in EECS.

[5]Dr. Tuan Cao-Huu is a research associate in the Division of Applied Sciences at Harvard University and a research consultant in the Physics Research Group and Department of Radiology at the Massachusetts General Hospital (MGH), Dr. Gordon Lee Brownell, is the Director of the Physics Research Group at MGH and Professor Emeritus of the MIT Department of Nuclear Engineering, and Dr. David Chesler is a research consultant in the Physics Research Group at MGH.

[6]John Marshall is a Professor of Physical Oceanography in the MIT Department of Earth, Atmosphere, and Planetary Sciences (EAPS), Alistair Adcroft is a Visiting Scholar in EAPS, Curtis Heisey and Christopher Hill are Research Specialists in EAPS, and Lev T. Perelman is a Research Scientist with the MIT Spectroscopy Laboratory and with EAPS.

[7]Arvind is a Charles W. and Jennifer C. Johnson Professor of Computer Science and Engineering in EECS and a member of LCS, R. Paul Johnson is a research associate in LCS, and Kyoo-Chan Cho is a graduate student in EECS and a research assistant in LCS.

[8]Daniel Hastings is a Professor of Aeronautics and Astronautics and Associate Department Head for Research in the MIT Department of Aeronautics and Astronautics and Robie Samanta-Roy is a graduate student in the MIT Department of Aeronautics and Astronautics.

[9] Dr. Michael Halbherr was a visiting Ph.D. student at LCS from the Swiss Federal Institute of Technology, Dr. Yuli Zhou is a research associate in LCS, Dr. Bradley Kuszmaul is a postdoc in LCS, Robert Blumofe, Christopher Joerg, Keith Randall, Andy Shaw, and Sivan Toledo are graduate students in EECS and research assistants in LCS, and Professor Charles Leiserson.

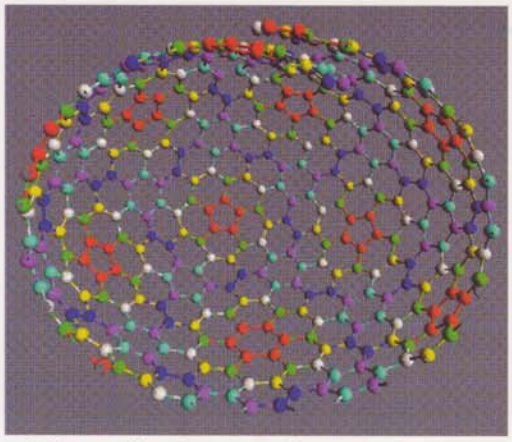
[10]Don Dailey and Larry Kaufman are from Heuristic Software, and Professor Charles Leiserson, Dr. Michael Halbherr, Dr. Bradley Kuszmaul, Robert Blumofe, and Christopher Joerg from LCS.

[11]Eric Brewer (Ph.D.) graduated in May 1994, and Robert Blumofe.

[12]Tzu-Yi Chen is an undergraduate student in EECS, and Sivan Toledo.

[13]Howard Lu is a graduate student in EECS, Philip Lisiecki is an undergraduate student in EECS, David Park (B.S.) graduated in May 1994, and Robert Blumofe.

[14]Albert Vezza is a Senior Research Scientist, Associate Director of the Laboratory for Computer Science and Principal Investigator, Project SCOUT.



Shell Assembly of a Non-viable Virus

This brochure was created to illustrate the types of research that is carried out in association with Project SCOUT. As such it highlights only a few of the activities conducted in association with it. The brochure was edited by Albert Vezza with help from Thomas Greene and Melissa LaBarge. Brochure layout and design and the project coordination was done by Melissa LaBarge. Our thanks go to contributing scientists and to Mary Ann Ladd of LCS and Jiri Schindler , a UROP student at LCS, who provided invaluable help with their technical assistance.