



In this issue...

[Software Helps Chemists See Atoms in a Different Light](#)

[New Network Design Proves Faster, Cheaper](#)

[NAS Condor Population Expands to Hundreds](#)

[NAS Systems Prepared for New Millennium](#)

[New Directives-based Parallel Benchmarks in Beta-test](#)

[High-speed Processor Techniques](#)

[Recent Technical Seminars Available on Video](#)

[Credits](#)

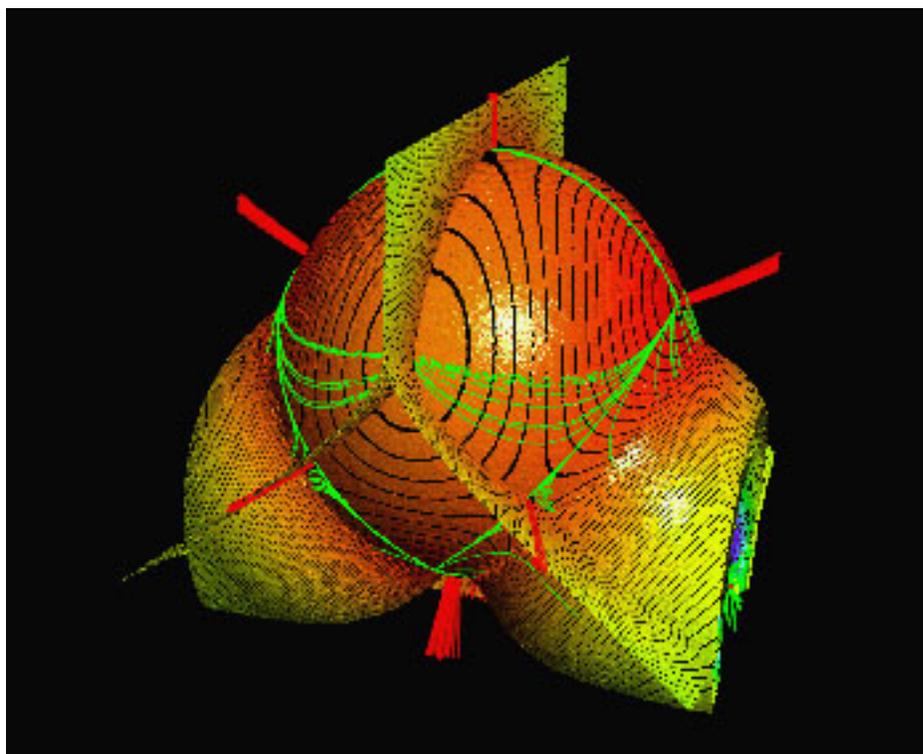
[Subscribe](#)

[Download the print version of NAS News Summer 1999 as a PDF file \(387K\)](#)

Software Helps Chemists See Atoms in a Different Light

In an unexpected crossover of the kind that energizes interdisciplinary researchers, software developed at NAS to analyze fluid flow around moving aircraft is giving chemists and computer scientists compelling new ways to visualize atoms, molecules, and their interactions.

[To The Article...](#)



In this unconventional view of a water molecule, the textured surfaces show where the electron charge density is neither increasing nor decreasing. Chemists studying "quantum topology" interpret these surfaces as natural partitions between atoms in a molecule. NAS researchers Chris Henze and

Creon Levit isolated and depicted the surfaces using mathematical techniques originally employed at NAS to highlight critical features in aerodynamic simulations, such as the "separation lines" where vortices strike an aircraft's wings. Dataset by Chris Henze

[To The Article...](#)

**In this issue...**

[Software Helps Chemists See Atoms in a Different Light](#)

[New Network Design Proves Faster, Cheaper](#)

[NAS Condor Population Expands to Hundreds](#)

[NAS Systems Prepared for New Millennium](#)

[New Directives-based Parallel Benchmarks in Beta-test](#)

[High-speed Processor Techniques](#)

[Recent Technical Seminars Available on Video](#)

Software Helps Chemists See Atoms in a Different Light

by [Wade Roush](#)

In an unexpected crossover of the kind that energizes interdisciplinary researchers, software developed at NAS to analyze fluid flow around moving aircraft is giving chemists and computer scientists compelling new ways to visualize atoms, molecules, and their interactions.

Researchers call the new approach "quantum topology," in reference to the previously unseen landscapes it generates based on the probable locations of a molecule's electrons. They have used it to produce stunning three-dimensional pictures showing curvy, globular surfaces and other strange geometrical features in molecules as seemingly simple as water. "No one before has had software to visualize these kinds of molecular surfaces in three dimensions," says Chris Henze, a researcher in the NAS [data analysis group](#). "We wanted to see molecules in their full 3-D glory."

The images help to explain why molecules take their distinctive shapes, how they orient themselves for chemical reactions, and even where new atoms may bind if a molecule is bent out of shape—all without recourse to the complex formulas of traditional chemical physics, in which atoms are pictured as nuclei buried inside successive shells of orbiting electrons. Such images, say Henze and colleague Creon Levit of the NAS [science and technology group](#), could one day help chemists design artificial molecules with useful pharmaceutical properties. They could even be used to improve simulations of biochemical processes like those that gave rise to life on Earth.

Within This Article...

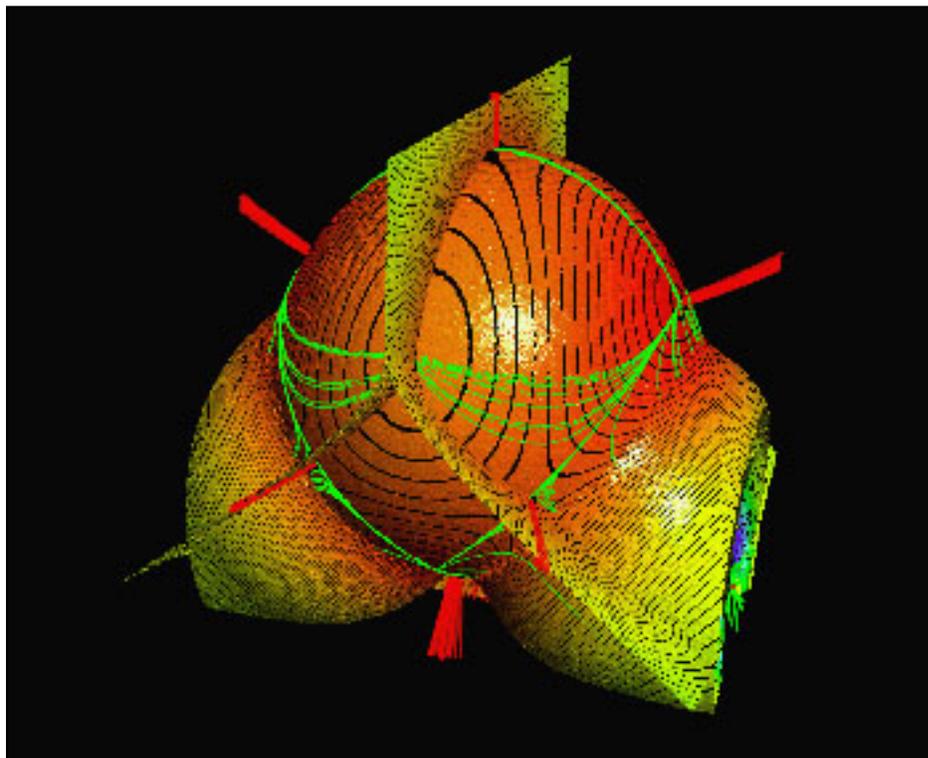
[From Aerodynamics to Atoms](#)

[Surprising Complexity](#)

[Predicting Chemical Reactivity](#)

[Sidebar: Quantum Topology Chronology](#)

[Sidebar: A Graphic Example](#)



In this unconventional view of a water molecule, the textured surfaces show where the electron charge density is neither increasing nor decreasing. Chemists studying "quantum topology" interpret these surfaces as natural partitions between atoms in a molecule. NAS researchers Chris Henze and Creon Levit isolated and depicted the surfaces using mathematical techniques originally employed at NAS to highlight critical features in aerodynamic simulations, such as the "separation lines" where vortices strike an aircraft's wings. Dataset by Chris Henze

From Aerodynamics to Atoms

The mathematical technique behind quantum topology, which involves finding the rate at which negative charge increases or decreases in the space between the atomic nuclei in a molecule, has been understood for years. But only recently did Henze and Levit adapt the technique and write new software-modeled on tools developed at NAS to analyze aerodynamic simulations-to create their colorful images.

"Creon and Chris can do in almost a blink of an eye" what used to take days of computer processing, says collaborator Preston MacDougall, a Middle Tennessee State University chemist who was among the first to use a property called "electron charge density" to predict a molecule's chemical reactivity. "In terms of visualizing these topological properties of electron density in three dimensions, they are alone in the field."

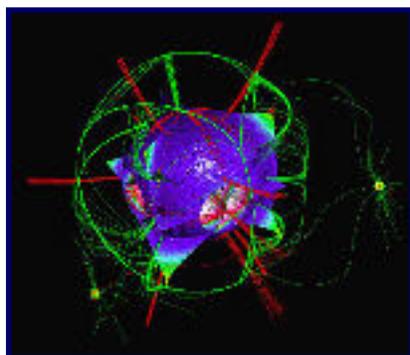
Levit and Henze made the advance when they recognized that they

could detect interatomic surfaces easily by writing feature detection algorithms, similar to NAS software that detects separation lines in aerodynamic simulations. Separation lines, which mark where air is moving abruptly away from an aircraft's wings, often signify the presence of drag-inducing vortices (see [NAS News, July-August 1998](#)). The flow vectors on either side of a separation line in an aerodynamic simulation closely resemble the field lines in the electron charge density of simple molecules. Indeed, the two systems can be described using the same mathematical techniques. "We gradually realized that the software we'd been writing to analyze the topology of fluid flow could also be used to analyze the topology of molecules," recounts Levit.

Surprising Complexity

Last year, MacDougall visited NAS for a 10-week fellowship jointly sponsored by the [American Society for Engineering Education](#) and the [Office of Human Resources and Education](#) at NASA headquarters. He, Levit, and Henze worked on software that converts 3-D data on a molecule's electron charge density (the probability of finding an electron at any given point around a nucleus) into a vector field representing the gradient of the charge density. The program then searches this field for separation surfaces, which can be thought of as boundaries between atoms.

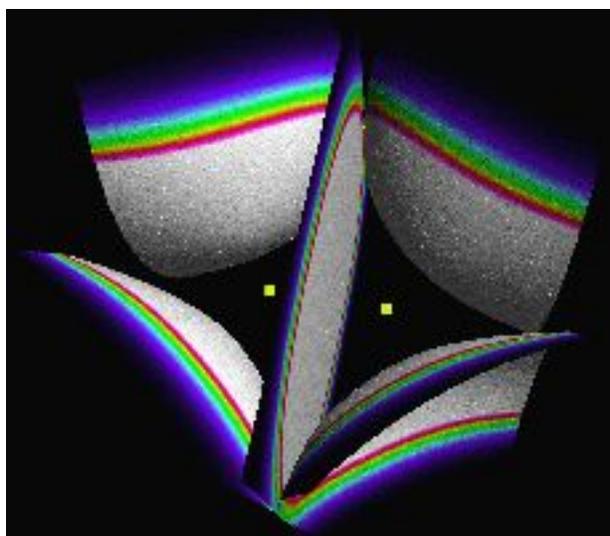
In the resulting images, "The surfaces are surprisingly complex, even for simple molecules," Henze comments. In a water molecule, for example, there is a bowl-shaped interatomic surface between each of the two hydrogen atoms and the central oxygen atom. In another picture of a more complex molecule, ethylene, the interatomic surfaces are clipped for visual clarity, but in actuality they extend to infinity, as if partitioning the entire universe into six huge atoms (directly below). "It's a very different picture from atoms as discrete building blocks," says Henze.



click the image above to see topological analysis of a water molecule. Dataset by Chris Henze

Predicting Chemical Reactivity

To study atomic interactions, the group turned to vector fields representing the gradient of the Laplacian of the charge density. "Zero-flux surfaces" in these fields, represented by rainbow colors in the illustrations [above](#) and [elsewhere](#), indicate where outside atoms have the best chance of attaching. Levit produced an animation showing how the gradient of the Laplacian changes in an ethylene molecule as the molecule is bent away from its natural planar configuration. "When you move ethylene from a planar to a bent position, reactive sites appear," says Levit. In other words, new topological sites open up where additional atoms can attach to the carbon backbone. This picture corresponds well with traditional chemical theory, which predicts that at high temperatures molecules will sometimes bend so far out of shape that they can form new chemical bonds.



In this visualization of ethylene, the yellow squares represent carbon nuclei. The colored surfaces, representing boundaries of zero flux in the gradient of the electron charge density, divide the space around the molecule into six huge atoms. Dataset by Chris Henze.

By highlighting the interatomic surfaces in the gradient of the Laplacian, explains Levit, chemists should soon be able to use computers to predict more precisely whether and how two molecules will fit together. And that's exactly what excites MacDougall, who will return to NAS this summer to complete his fellowship. "I'm interested because the shapes

of proteins and their interactions with substrates are really what control their biological function," he says. "One picture I would like to see is adenine, one of the bases in DNA, and what the other bases are seeing when they approach adenine. Cytosine and guanine don't like to bind to adenine, but thymine does. What's the pattern of bonding sites? Why is thymine drawn in? Traditional chemistry just draws the structure and says 'this is a bonding site because it satisfies some empirically derived rule.' I would like to see the bonding sites captured by topological analysis. To predict it from elementary physics-that would really be something," he says.

The traditional molecular orbital theory of chemistry cannot be thrown out, Levit notes. "It has pictures and models, and it makes predictions about which quantum topology says nothing. But it is rather ad hoc and difficult." Quantum topology, meanwhile, is a highly spatial and geometrical theory, and therefore easy to visualize. "The pictures are mental tools," says Levit, adding that knowing both molecular orbital theory and quantum topology is "like having a toolbox with twice as many tools."



**In this issue...**[Software Helps Chemists See Atoms in a Different Light](#)[New Network Design Proves Faster, Cheaper](#)[NAS Condor Population Expands to Hundreds](#)[NAS Systems Prepared for New Millennium](#)[New Directives-based Parallel Benchmarks in Beta-test](#)[High-speed Processor Techniques](#)[Recent Technical Seminars Available on Video](#)

New Network Design Proves Faster, Cheaper

by [Wade Roush](#)

Moore's Law predicts that computer chips will double in power every 18 to 24 months, guaranteeing constant performance gains for users willing to upgrade. But many of the biggest revolutions in computing, such as the growth of the Internet, have been rooted in new networking techniques, not improved circuitry. That's why researchers in the NAS Systems Division are excited about what they call the "P-mesh," an experimental network architecture designed to connect hundreds or thousands of off-the-shelf PCs into low-cost, high-end parallel computers or "commodity clusters." Recent tests have shown that the P-mesh is a viable alternative to traditional network architectures.

In past commodity clusters built at NAS and other institutions, processors have communicated through networks resembling trees or toruses. In these architectures, messages must stop at each hub to have their destination addresses checked. A P-mesh, by contrast, consists of a grid of switches, requiring at most one stop between any two processors. A large P-mesh can be just as fast as-but less failure-prone and less expensive than- a tree or torus, NAS researchers reported at a computing conference last January.

"We've got shelves full of Pentium Pro processors, operating systems to run them, and software to make them talk, but the piece that turns them into a supercomputer is the network," explains project lead Bill Nitzberg. Members of his commodity computing group experiment with different configurations of commercial hubs, routers, switches, network cards, and cables, trying to maximize the bandwidth of the network tying the cluster together, while minimizing the cluster's total cost. "We

Within This Article...[Getting in the Fast Lane](#)[Exploiting Low-cost Components](#)['Good Performance in Reality'](#)

want to choose the network that will give the best price/performance ratio," explains Nitzberg.

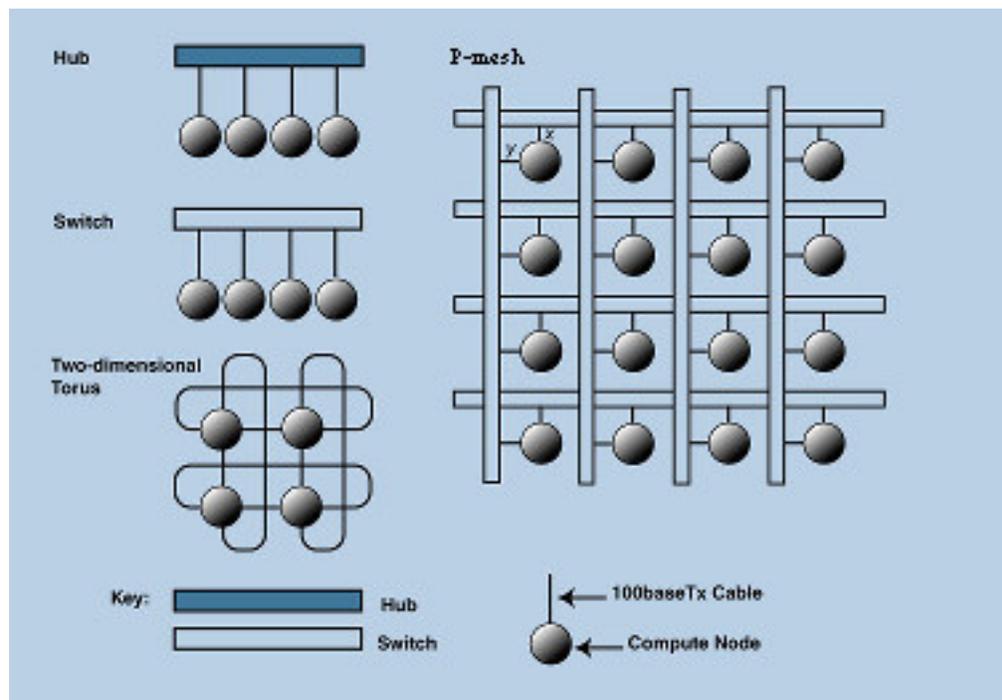
A prototype P-mesh has performed well in tests, and the project team has already applied for a patent on the architecture. "The practical contribution [of P-mesh] is that we have a new design point," says researcher Chris Kuszmaul, a member of the parallel systems group. "Depending on where we are in the cost fluctuations, that can help us to get closer to optimal."

Getting in the Fast Lane

A parallel computer's speed, Kuszmaul explains, is always limited by the bandwidth of its communications network. Anyone who has been stuck in traffic can see why. If the skyscrapers of Manhattan were processors in a giant parallel computer, and messages had to travel between buildings by taxi, computations would be maddeningly slow. But if construction costs dropped and a mesh of superhighways were built above the city's streets and avenues, the same taxi could zoom to its destination with hardly a stop.

In search of the network equivalent of the superhighway, Nitzberg's team has put several common network architectures to the test. In the "hub" version, for example, they connected all processors, or nodes, to a single Ethernet hub. This 50-node PC cluster performed standard aerodynamic simulations five times more efficiently than Turing, a 64-processor SGI Origin-2000 supercomputer at NAS.

The cluster's edge, however, was all in its low price, not its speed. The researchers bought off-the-shelf processors, hubs, and cables relatively cheaply, so the cost per hour of computation was low. But because every processor connected to an Ethernet hub must stop each message briefly to check its address, the simulations required far more time on the 50-node cluster than on Turing, the group reported. [For background on the 50-node cluster and early software porting tests, see NAS News, [November-December '97](#) and [March-April '98](#)]



Chris Gong, adapted from Nitzberg et al.

Recently, the team also tested nodes connected by a single switch, by a ring or torus, and by a two-dimensional mesh of toruses (see illustration, above). These architectures are faster than a hub, but the group found that each has its own drawbacks. A network connected by a single switch, for example, is vulnerable to single-point failures: the whole network will crash if the switch fails. Even the biggest, most expensive switches, moreover, have no more than 100 ports for processors, making it impossible to scale up this architecture to thousands of nodes.

A torus or a two-dimensional mesh of toruses, meanwhile, is more reliable than a single switch and can be scaled up arbitrarily. As more and more nodes are added, however, this configuration becomes fiendishly difficult to wire. And because all messages in a torus are routed by the nodes, they make many stops on the way to their destinations, slowing the network's performance.

Exploiting Low-cost Components

The group needed to find a hybrid that would be easily scalable, as fast as a single switch, and as reliable as a mesh of toruses. And it would have to be inexpensive. As Nitzberg recounts, "We tried to think up a way of exploiting the fact that certain components are really cheap."

That was when the group hit on the P-mesh idea. The new architecture is like a mesh of toruses, where the cables have been replaced by switches. In effect, Nitzberg explains, these switches act as "smart

cables" that automatically connect pairs of nodes when they want to send messages. Packets of data are shuttled directly between sender and recipient along a dedicated, high-bandwidth link, with no more than a single intermediate stop to switch from one grid dimension to the other. The single-stop rule remains true no matter how many nodes and switches are added to the mesh, giving a P-mesh "the architecture scalability and hardware fault resilience of the torus, but without the loss of performance scalability," the team wrote in their [paper](#) for the annual Hawaii International Conference on Systems Science in January.

To test the design in practice, Nitzberg, Kuszmaul, Jeff Becker, Parkson Wong, and colleagues Ian Stockdale and John Jiang built a 16-node P-mesh and compared its performance to that of a hub architecture, a switch, and a 2-D mesh of toruses. As benchmarks they used three computational fluid dynamics problems from the NAS [Parallel Benchmarks](#) software suite. The P-mesh cluster outperformed the hub on all three benchmarks. It beat the switch in two out of the three cases, and bested the torus mesh on one benchmark.

'Good Performance in Reality'

"The results show that the P-mesh has good performance in reality," says Kuszmaul. The test also proved that P-Mesh is a fast, scalable, affordable alternative to other commodity cluster architectures—especially now that switches are low in price compared with high-end LAN components such as gigabit Ethernet hubs.

The P-mesh team's patent on the new architecture is pending. The team hopes to increase its test cluster to hundreds of nodes to see whether a larger P-mesh yields additional performance gains. "A P-mesh is not going to be the best decision every year, but some years it will be," Kuszmaul says. "It's during those times that the cost/performance benefits can be reaped."

For more information on the P-Mesh, contact [Bill Nitzberg](#) or [Chris Kuszmaul](#).



**In this issue...**[Software Helps Chemists See Atoms in a Different Light](#)[New Network Design Proves Faster, Cheaper](#)[NAS Condor Population Expands to Hundreds](#)[NAS Systems Prepared for New Millenium](#)[New Directives-based Parallel Benchmarks in Beta-test](#)[High-speed Processor Techniques](#)[Recent Technical Seminars Available on Video](#)

NAS Condor Population Expands to Hundreds

by [Wade Roush](#)

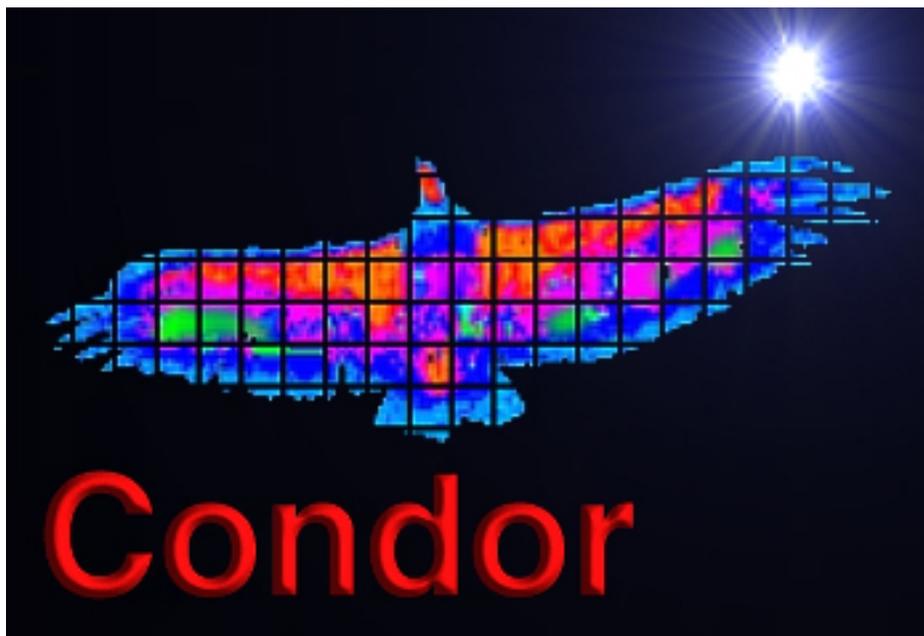
Few researchers get as much time off as their computer workstations, which sit unused through many a staff meeting, lunch hour, evening, or weekend. At the NAS Facility, however, such idleness is becoming a thing of the past. Workstations caught dozing for more than a few minutes are now being put to work by Condor, a software system that distributes "extra credit" jobs to machines that have finished their regular work.

More than a hundred Unix-based workstations have already been added to the NAS Condor pool. The pool will eventually grow to include most Unix workstations at NAS, according to a memorandum of understanding signed between the NAS Systems Division and the [University of Wisconsin](#), where Condor was developed. The move is designed to divert otherwise wasted computing resources for scientific work by users of the NAS-led [Information Power Grid](#), and to test Condor's ability to handle scientific applications written in new computer languages such as Java.

"At Wisconsin, their experience says that the average workstation is idle 14 hours a day," says Al Globus, a researcher in the NAS science and technology group. "NAS has 400-odd workstations. If you do the math, that adds up to 2 million CPU-hours per year, which is roughly equivalent to the CPU power of Steger," the 256-processor SGI Origin2000 computer at the NAS Facility. "It's a little silly to have these machines sitting around doing nothing if there is actual work they could be accomplishing."

Within This Article...[Perfect for Parallel Applications](#)[Making Condor Java-aware](#)[Passing the Checkpoint](#)

Boosting activity among NAS workstations should cause little trouble for their users. Condor recruits a workstation only after several minutes of dormancy. It returns complete control to the user at the first sign of activity such as a keystroke. The only change users might see once their workstations are added to the pool, Globus explains, is the disabling of screen savers such as Fireworks, which chew up valuable CPU time and can make a workstation appear busy to Condor.



NAS Condor logo by Steve Traugott.

Perfect for Parallel Applications

Pools of workstations naturally lend themselves to parallel applications, jobs that can be broken down into multiple parts and processed separately. That's why Globus, who helped pioneer the use of "genetic algorithms" to generate nanotechnology designs automatically, first became interested in Condor.

These algorithms, which apply "fitness functions" simulating the forces of natural selection to graph representations of molecules, are highly parallel by nature, according to Globus. "In natural evolution the fitness function is survival. In artificial evolution you write a function that combs through millions of different molecules to decide which shall 'die' and which shall 'reproduce.' That's very high-throughput and compute-intensive," he says.

Globus's first genetic algorithms, written in Java, quickly outran the limited processing power of his own workstation. "I knew that there were an awful lot of workstations sitting around doing nothing most of

the time, and I thought that would be a neat way to run my algorithms," he says.

In fact, University of Wisconsin researcher Miron Livny says the problem of load imbalances-"somebody waiting to do something while simultaneously a machine capable of doing it is sitting idle"-is exactly what motivated him to launch the Condor project in 1986. When Globus canvassed the field, he decided that Livny's system was the most promising of the various "cycle-stealing" systems available. The only problem: Condor doesn't provide support for Java applications.

Making Condor Java-aware

In order to get his Java algorithms to run on the initial NAS Condor pool of 26 workstations, Globus says, he had to "play little games"-for example, explicitly starting the Java Virtual Machine, which must be operating on a workstation before Java applications can get down to work. "I would like Condor to know about Java so we don't have to play those games," he says.

The Wisconsin researchers have the same hopes. "When Al that said he wanted to use Condor at NAS and actually expand it to all workstations, we were very excited because of the new challenges this would bring our way," says Livny. Java's main strength-its ability to run on multiple platforms, from handheld information appliances to supercomputers-makes it well suited for clusters of workstations with heterogeneous architectures and operating systems, Livny continues. But because few Condor users had a need to run Java scientific applications on Condor, the Wisconsin group never got around to making the necessary modifications. "What's interesting and unique about Al is that he's using Java on Condor to do real science," Livny says. "It's a very attractive environment."

Under the memorandum of understanding, the Wisconsin team's main role will be to give Condor new Java-related capabilities, including the ability to checkpoint Java automatically. NAS's role, meanwhile, will be to exercise and test these improvements using actual scientific applications such as Globus's genetic algorithms.

Passing the Checkpoint

An important function of Condor, but one unsupported for Java applications before Globus's work, is checkpointing. When a machine belonging to a Condor pool has been idle for several minutes, Globus

explains, it sends a message advertising its availability and its characteristics to a central server running the Condor software. Condor looks through its list of pending jobs for any matches-what programmers call a "classified ad" procedure-then sends jobs to appropriate machines. If checkpointing is working and the application is using one of the standard programming languages (C, C++, or Fortran), Condor periodically takes a "snapshot" of the memory and CPU register state for a running job.

This way, the process can be migrated to another machine in the pool when the local user takes back control. "This ability to 'freeze' the memory state of a running process, kill it, and migrate it to another workstation is one of the most powerful features of Condor," says NAS infrastructure architect Steve Traugott, who is implementing the expanded Condor pool.

Without checkpointing, killed jobs would have to be restarted from the beginning-a tedious and inefficient process. Yet this was how Globus's genetic algorithms behaved, until he added special code that generates checkpoint files every ten minutes. Livny says he and the rest of the Condor team are now working on improvements that will enable Condor to checkpoint Java applications at any time users specify, without special code.

That will benefit Condor users like Globus. But the attraction to the Wisconsin team of working with NAS, Livny and Globus agree, is that NASA scientific applications will put Condor improvements to the test in grueling real-world situations. "I really want to know the answers to the scientific questions we're asking," Globus says. "Java is only a means to that end. That's the important thing for the Condor group."

Plans also call for the NAS Condor pool to become part of the Information Power Grid (IPG), an advanced network of computers, scientific instruments, and storage systems being built by NAS and its research partners. That way the Condor pool "can take a place next to the supercomputers as a partner in the Grid," says Traugott.

But the problem of representing the heterogeneous resources collected in any Condor pool to the rest of the Grid will need to be solved first, says Traugott. "In a Condor pool, each processor is as likely as not running a different operating system than most of the other processors in the pool, and also has a different hardware and memory architecture," he

says. "This means that the IPG infrastructure will need to be aware that a Condor 'node' has the ability to accept many different kinds of jobs."

Globus and Traugott are currently discussing with Livny and his team how this awareness can be imparted. "We have always looked at each Condor pool as an independent entity," says Livny. "But we definitely want them to talk to each other." This would enable NAS workstations to take over Condor jobs running at the University of Wisconsin or other institutions, or to send jobs to far-away Condor pools. "Basically, we are building a community of all these resources," Livny says.

More information about the project is available online at the [NAS](#) or [University of Wisconsin](#) Condor sites.



**In this issue...**[Software Helps Chemists See Atoms in a Different Light](#)[New Network Design Proves Faster, Cheaper](#)[NAS Condor Population Expands to Hundreds](#)[NAS Systems Prepared for New Millennium](#)[New Directives-based Parallel Benchmarks in Beta-test](#)[High-speed Processor Techniques](#)[Recent Technical Seminars Available on Video](#)

NAS Systems Prepared for New Millennium

by [Wade Roush](#)

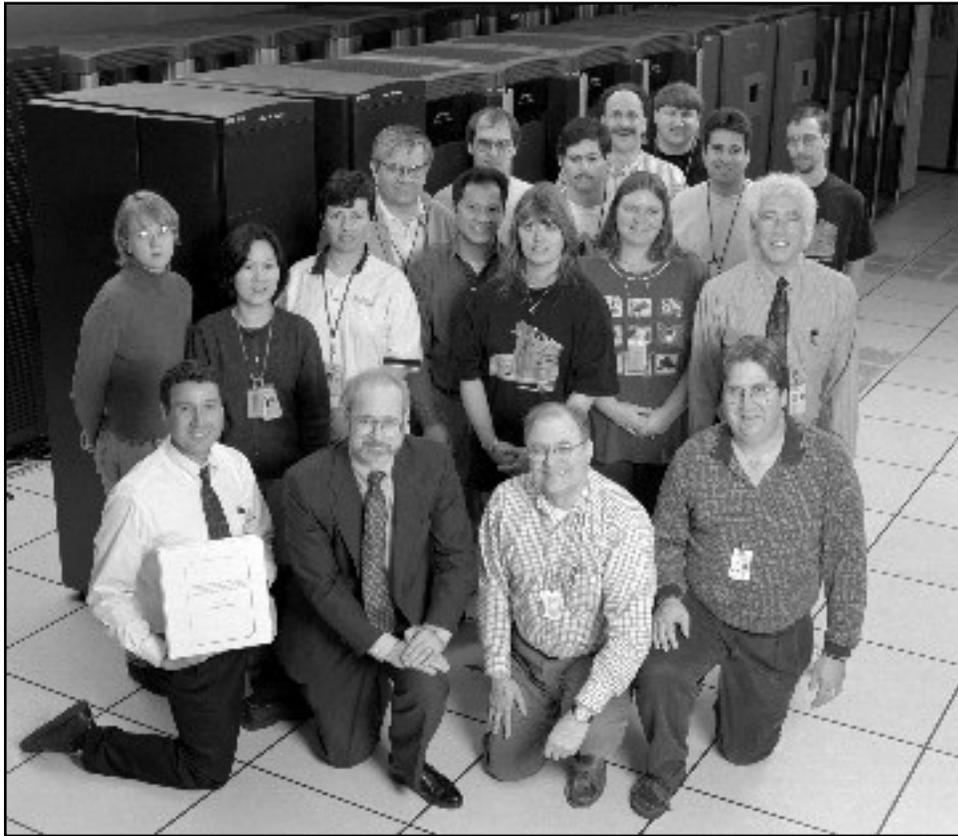
At the NAS Facility, the new millennium came more than 350 days early.

Beginning in early January, support staff reset a representative set of computers and storage systems in the division to roll over simultaneously from 11:59 p.m., December 31, 1999, to 12:00 a.m., January 1, 2000. The test was designed to demonstrate the facility's readiness for the year 2000 well in advance of a February 28, 1999, deadline mandated by NASA Administrator Daniel Goldin.

The NAS Systems Division maintains so many mission-critical computer systems that nothing less than a full-scale integrated systems test was required, says Bill Thigpen, Y2K coordinator for the Ames Information Sciences directorate, which includes NAS and several other divisions. The results were "very encouraging," Thigpen says. "NAS is not going to have a problem in the year 2000."

This puts the division in line with other Ames offices and directorates, many of which have reported full Y2K compliance for mission-critical systems. NASA-wide, 99 percent of such systems were Y2K-compliant by the end of March, 1999, according to the [President's Council on Year 2000 Conversion](#).

Within This Article...[Clockwork Coordination](#)[Non-compliant Systems To Be Scuttled](#)



Ames Y2K project manager Ray O'Brien and chief information officer Scott Santiago congratulate Bruce Blaylock, Bill Thigpen, and the NAS support team on Y2K certification. Front row, L to R: O'Brien, Santiago, Blaylock, Thigpen. Standing, L to R: Sabrina Farmer, Mi Young Cho, Mary Hultquist, Tom Lasinski, Archie De Guzman, Darrell Root, Larona Pfaller, Dan Sully, Dan DePauk, Chris Kleiber, Alan Powers, Frank Sabadin, Bob Hirsch, Tim Kirby. Photo by Dominic Hart.

Clockwork Coordination

The sheer number of systems to be tested at NAS guaranteed that the month of January would be demanding for the Y2K team and members of the NAS user support groups. The tested systems included all of the high-speed production computers at the facility (two CRAY C90s and four CRAY J90s), the SGI Origin2000 cluster (with a total of 384 processors), NASTore (the facility's mass storage system), the Portable Batch System, all administrative databases, local-area networks, control room systems, many workstations, and several web servers.

The test began at midnight on Monday, January 11. "The workstation group, the high-speed processor group, the LAN group, the parallel systems group, and the control room all worked together to take the systems down, bring them back up with the new date, and let them roll

through the period of the change," recounts Thigpen. In addition, the groups previewed several other rollover dates, and proved that data written in the 20th century could be accessed when the system date was in the 21st century. During the four-day test period, NAS users were given free computer time to test their own applications for Y2K problems under the different date scenarios.

Non-compliant Systems To Be Scuttled

For the majority of systems, the Y2K team found no issues to report. Not every piece of equipment and software in the facility, however, proved Y2K-compliant. "There were two cases of moderate to major non-compliance: Sally and NASStore," explains Thigpen.

Sally was a stand-alone SGI file server that long functioned as the main gateway to the NAS workstation environment. It was unequipped to recognize the difference between the years 1900 and 2000, and was replaced this spring with several new gateway hosts. NASStore, a software system developed at NAS for the management of mass data storage, runs on Convex 3820 computers, which were known before the tests to be Y2K-noncompliant. However, NASStore was already scheduled for replacement this summer by a new group of SGI machines running the Data Migration Facility (DMF).

Aside from these anticipated Y2K problems, says Thigpen, the exercise exposed a few minor wrinkles in the division's workstations and accounting software that were ironed out even before the test ended.

Overall, "You guys did a really good job out there," says [Ames Y2K Project Office](#) director Ray O'Brien of the NAS testing effort. "Now we know we're compliant."



**In this issue...**

[Software Helps Chemists See Atoms in a Different Light](#)

[New Network Design Proves Faster, Cheaper](#)

[NAS Condor Population Expands to Hundreds](#)

[NAS Systems Prepared for New Millenium](#)

[New Directives-based Parallel Benchmarks in Beta-test](#)

[High-speed Processor Techniques](#)

[Recent Technical Seminars Available on Video](#)

New Directives-based Parallel Benchmarks in Beta-test

by [Wade Roush](#)

When the NAS name comes up among professionals in high-performance computing, it's often in association with the NAS Parallel Benchmarks (NPB), a kind of standardized exam for parallel computers. When computer scientists and manufacturers want to put new parallel architectures or parallelization tools to the test, they often turn to the benchmarks, software based on computational fluid dynamics problems that are stressful enough to give any parallel machine a run for its processors.

As parallel programming techniques have advanced, so have the benchmarks. NAS researchers released NPB 1.0 in 1991, and within a few years the benchmarks had already gone through several revisions. NPB 2.3, released in 1996, included changes that used the Message Passing Interface (MPI), a common standard for multicomputer message passing. A serial version premiered in 1997. Now a new variation on the benchmarks, tailored to work with shared-memory programming paradigms such as OpenMP and High Performance Fortran (HPF), is being readied for public release in September.

NAS researchers Michael Frumkin and Henry Jin, who developed the new PBN (Programming Baseline for NPB), emphasize that the new variation is not version 3.0 of the benchmarks. The benchmarks have yielded a large body of directly comparable results, the two say, precisely because they have stayed fixed since 1996. Instead, the researchers are offering PBN as an informal collection of NPB codes adapted to work more efficiently with parallel applications using OpenMP and HPF.

"We modified the benchmarks for our own use, as part of our research

Within This Article...

[Samples, Not Standards](#)

comparing parallel compilers and tools. Then people [outside NASA] became interested," says Michelle Hribar, a member of the NAS [legacy code modernization group](#). The group wanted to make the modifications available to the general research community, but without the work or disruption that goes with a new release, she adds.

"The question was, do we make this a full release, or make it something different from the benchmarks?" Hribar says. They chose something different and built PBN, which is now being evaluated by beta-testers at about a dozen academic and industry sites.

Samples, not Standards

In recent years, new parallel programming techniques have emerged based on "directives," annotations written into code on how a parallel computer should divide a computation into parts. Part of the legacy code modernization group's work, Hribar explains, is to develop and test compiler software that takes advantage of code including directives. To do this, the group needed sample parallel programs using directives.

NAS researchers designed NPB 2.3 as a true standard, usable on a range of parallel platforms without special modifications. However, directives differ depending on the architecture and manufacturer of the computer being used. So the group had no choice but to depart from the standard benchmarks. They developed PBN-OpenMP, a version of the benchmarks specialized for computers with shared-memory architectures such as the SGI Origin2000, and PBN-HPF, optimized for directives-based Fortran programs.

The group presented some of its results using the new programming baselines at the [SC98 High Performance Networking and Computing conference](#) in Orlando last November. The talk generated hubbub-and requests for the software. "There are a lot of people out there who are very interested in being able to compare their own results with what we've done," says Hribar.

At the same time, "We do not think that releasing a 'standard' directives-based parallel implementation makes sense," says Jerry Yan, who leads the legacy code modernization group. "Directives are architecture- and compiler-sensitive. An extensively annotated source code completely defeats our objective to compare a compiler's ability to discover and exploit parallelism inherent in the code. We have, nevertheless, decided to release the directive versions as 'samples,'" he says.

As a starting point for PBN-OpenMP and PBN-HPF, the group also produced PBN-S, a refined serial version of the benchmarks. In this version, all parallelism has been removed, and imperfections that limited the performance of previous serial versions of the benchmarks have been eliminated. Other researchers can now use PBN-S to create their own versions of the benchmarks tailored for other implementations of directives.

The new baselines have been distributed to beta testers at institutions such as SGI, Sun Microsystems, San Diego Supercomputer Center, Rice University, and Kuck & Associates. Full release is expected in September. To become a beta tester or to get more information about the Programming Baseline for NPB, [contact](#) Jerry Yan.



**In this issue...**[Software Helps Chemists See Atoms in a Different Light](#)[New Network Design Proves Faster, Cheaper](#)[NAS Condor Population Expands to Hundreds](#)[NAS Systems Prepared for New Millennium](#)[New Directives-based Parallel Benchmarks in Beta-test](#)[High-speed Processor Techniques](#)[Recent Technical Seminars Available on Video](#)

HSP Techniques: Using OpenMP on the SGI Origin2000s

by [Terry Nelson](#)

OpenMP, a new "vendor standard" in the area of parallel processing, is a portable, shared memory multiprocessing API (Application Programming Interface). It consists of a set of commands and directives that help to control the running of loops and code sections in Fortran on more than one processor at a time. An implementation for C and C++ is imminent.

OpenMP has been available on the SGI Origin2000s for a while, but became available on the Cray parallel vector processing (PVP) systems (the C90s and J90s) only with the latest operating system releases. The implementation details and history of parallel processing are somewhat different on the PVP and Origin systems, and merit a brief discussion.

Parallel Computing vs. Vectorization

The value of parallelizing jobs on the CPU level can be debated, since it expends some resources in handling the parallel regions. The result is increased CPU time, but reduced wall-clock time, clearly an advantage if the system as a whole has unused resources. In general, on PVP systems, programs should first be optimized for vectorization; on the Origins, code and data caching are the first steps toward optimization.

OpenMP

The quotes around "vendor standard" in the first sentence refer to the fact that this is not a standard in the sense of being sanctioned by a standards organization such as ANSI, but rather is a consensus among a group of software and hardware vendors and research organizations.

SGI has supported several earlier products, the latest called PCF, which allowed the parallelization of DO loops in a fashion similar to OpenMP. The best known command in this group may be the C\$DOACROSS directive. The PVP systems also have a long history of such products, the most recent being autotasking.

OpenMP Directives

Once it has been determined that parallelization will improve a specific loop or section of code, OpenMP directives are entered manually around these loops or sections. There are, however, a few tools to automate parts of this process, including 'auto_p' on the Origins, and ATExpert on the PVP systems. (In the code examples below, each directive line starts with a prefix like !\$OMP, C\$OMP, or *\$OMP which is meaningful to OpenMP, but will otherwise be treated as a comment.)

One advantage of OpenMP is that it not only functions on a DO loop level, but permits the programmer to

Within This Article...

[Parallel Computing vs. Vectorization](#)[OpenMP](#)[OpenMP Directives](#)[Data Decisions](#)[Special OpenMP Calls](#)[Using OpenMP](#)[Sample Program and Script](#)[Documentation](#)[Conclusion](#)[Update: Reading Cray Binary Files on the O2Ks](#)

define a 'parallel region' to be executed by all of the threads requested and available. The construct for a parallel region is:

```
!$OMP PARALLEL
```

block of code !\$OMP END PARALLEL Within a parallel region several constructs are possible. A do loop can be defined:

```
!$OMP DO
```

do loop !\$OMP END DO This construct illustrates the power of OpenMP. The passes inside the DO loop will be distributed for execution among the processors. The user can control the manner of this distribution with the SCHEDULE parameter after the DO. There is an implied barrier at the !\$OMP END DO. If that synchronization is not needed, a NOWAIT parameter can be added after the DO.

Non-iterative sections of code to be run in parallel can be defined:

```
!$OMP SECTIONS
!$OMP SECTION
    some code
!$OMP SECTION
    some other code
!$OMP SECTION
    and more code
!$OMP END SECTIONS
```

To the extent possible, the code following each SECTION line will be run on a different processor.

If an area of code must be executed only once by any processor, one can use the constructs:

```
!$OMP SINGLE
    code executed by only one processor
!$OMP END SINGLE
```

If an area of code must be executed by each processor, but only by one at a time, one can use the constructs:

```
!$OMP CRITICAL
    code executed by one processor at a time
!$OMP END CRITICAL
```

Several combined constructs exist, such as

```
!$OMP PARALLEL DO
    . . .
!$OMP END PARALLEL DO
```

or

```
!$OMP PARALLEL SECTIONS
    . . .
!$OMP END PARALLEL SECTIONS
```

Data Decisions

One of the trickier challenges in using OpenMP and similar systems is determining what data will be private

to a specific process, and what data will be shared. For example, loop induction variables should be private, but arrays which are only read should probably be shared. Most of the constructs have a similar set of clauses, which include:

PRIVATE(...) each process has its own copy

SHARED(...) all processes have access to these variables

DEFAULT(...) set default for variables of private, shared, or none

FIRSTPRIVATE(...) preset the variable

LASTPRIVATE(...) copy the final value

To avoid the extra overhead for very short loops, the IF(logical expression) tests a threshold to decide whether to proceed in parallel.

Reduction DO loops can be simplified with a REDUCTION(operator: ...), which performs mathematical or logical reduction of specific variables.

Because controlling the order in which processors execute the parallel regions may be important, several synchronization constructs are provided.

!\$OMP MASTER / !\$OMP END MASTER executed only by the master thread

!\$OMP CRITICAL / !\$OMP END CRITICAL executed by one thread at a time

!\$OMP BARRIER all threads wait until each thread has reached this point

!\$OMP ATOMIC update a specific location atomically

!\$OMP FLUSH at this point write specified variables back to memory

!\$OMP ORDERED / !\$OMP END ORDERED sequential execution

Special OpenMP Calls

There are two calls that the program can use to help orient itself. This information is useful for domain decomposition models.

numthreads = omp_get_num_threads() returns the number of current threads

iam = omp_get_thread_num() returns this thread's number

Using OpenMP

OpenMP is available on all of the NAS Origins for f77 as well as f90. For both f77 and f90, compile using the -mp parameter. For the PVP systems, use f90 with *no* special parameter.

There are two environment variables which should be included in the run script. OMP_DYNAMIC should be set to TRUE to allow the system to dynamically adjust the number of CPUs assigned.

OMP_SET_NUM_THREADS requests the number of threads needed.

Sample Program and Script

As an example, the following is a very basic program and script using OpenMP.

program sample1

```

integer width, length, iam, numthreads, i, j
parameter (width=128, length=4096)
real a1(length, width)

c   This code is executed only by the master process.
    do i=1,width
      do j=1,length
        a(j,i) = 0.0
      enddo
    enddo
C   The next line defines a parallel region.

!$OMP PARALLEL PRIVATE(i,j,iam,numthreads) SHARED(a1)

C   The threads can orient themselves if they wish
    iam = omp_get_thread_num()
    numthreads = omp_get_num_threads()

C   Each process does the following print statement
    print *, 'iam is ', iam, ' of ', numthreads

!$OMP DO SCHEDULE (STATIC,64)
C   This gives 64 iterations at a time to      each thread in turn
    do i=1,width
      do j=1,length
        a1(j,i) = (float(i)/10) * (float(j)/10)
      enddo
    enddo

!$OMP END DO
C   Each process does the following print statement
    print *, ' After i-j do loops inside parallel'

!$OMP END PARALLEL
C   After end of parallel region. Only master does print statement.
    print *, ' After i-j do loops outside parallel'
end

```

If the file is called sample1.f, on the Origins one could compile the program with:

```
f77 -o sample1 -mp sample1.f
```

or

```
f90 -o sample1 -mp sample1.f
```

On the PVP machines I would omit the -mp, e.g.,

```
f90 -o sample1 sample1.f
```

For a run script, if one wants to run on 4 processors, one might use:

```

#PBS -S /bin/csh
#PBS -l cput=00:10:00
#PBS -l ncpus=4
setenv OMP_DYNAMIC TRUE

```

```
setenv OMP_SET_NUM_THREADS 4
cd $PBS_O_WORKDIR
./sample1
```

The output will vary somewhat from run to run because the order of executing processors will vary, but all of the print statements should be present.

Documentation

There are man pages for the thread aspects of running OpenMP. These can be seen by typing `man omp_threads`. The key manual with OpenMP information is the *CF90 Commands and Directives Reference Manual*.

For online Origin documentation, go to <http://techpubs.sgi.com/library/>. Select IRIX 6.5 on the Search menu, and type OpenMP in the Keyword Search box. For the PVP systems, go to <http://ncc.nas.nasa.gov:8080/> and search for OpenMP. For more general or background information about OpenMP, as well as the design document, see [here](#).

Conclusion

The NAS scientific consulting group encourages users to try OpenMP in their next code parallelization effort. The current/older versions, PCF and Autotasking, are officially considered 'outmoded', although there have not yet been indications of any change of support for these products. The increased power, flexibility, and portability provided by OpenMP may even induce some to translate their current algorithms.

Reading Cray Binary Files on the Origin2000

(An update to [HSP Techniques](#), March-April '99)

SGI has released a new version of the MIPSpro Fortran compiler (version 7.3.0.0.beta) which makes reading Cray binary files on the Origins much simpler than previously reported. The previous release of the Fortran compiler (version 7.2) enabled Cray-to-SGI (IEEE) binary data conversion to be done "on the fly" with use of the `assign -s cos -N cray ...` command, but it only supported conversion from 64-bit Cray format to 32-bit SGI format. To obtain 64-bit-to-64-bit conversion on the Origin2000, you had to modify a program to read Cray values into an input buffer, then call the CRY2MIPS conversion routine to do the numeric conversion. The CRY2MIPS function is no longer necessary with the version 7.3 compiler. The same `assign` command (with the `-s cos` and `-N cray` options) enables both 64-to-32 bit and 64-to-64 bit conversion as appropriate. It all depends on whether the target variable is 4-bytes or 8-bytes in length. Note that real and integer variables are 4 bytes by default on the Origins, and can be made 8-bytes long by explicit declaration, e.g., `REAL*8 A(10)`, or by compiler option, for example, `f90 -r8`. You can switch to this new compiler as long as your shell can use modules. Make sure that you have the following lines inserted in your `.cshrc` file after either the path definition or the sourcing of the `global.cshrc` file:

```
#Modules setup
if (-f /opt/modules/modules/init/csh ) then
  # Initialize modules
  source /opt/modules/modules/init/csh
  # allow module command/manpage
  module load modules
endif
```

Your shell will be set up to use modules as soon as you type `source .cshrc` or simply log back in.

The command `module avail` will show what modules are available on the system. The command `module load MIPSpro.7.3.0.0.beta` will load the new compiler for you. This beta version has problems converting complex numbers if they are mixed in the file with reals and integers. It is anticipated that the manufacturer release version will correct this flaw. You can check the version of the MIPSpro Fortran compiler that you are using by typing `f90 -version`.

For more help with OpenMP or data conversion, contact the NAS scientific consulting group at (650) 604-4444 or (800) 331-8737, or send email to nashelp@nas.nasa.gov.





In this issue...

[Software Helps Chemists See Atoms in a Different Light](#)

[New Network Design Proves Faster, Cheaper](#)

[NAS Condor Population Expands to Hundreds](#)

[NAS Systems Prepared for New Millenium](#)

[New Directives-based Parallel Benchmarks in Beta-test](#)

[High-speed Processor Techniques](#)

Recent Technical Seminars Available on Video

NAS Technical Seminars, October 1998 - February 1999

Videotapes of many research seminars presented at the NAS Facility are available for loan. Procedures for obtaining training materials and information on past training events can be found [online](#).

Black-box Multigrid Solvers for Elliptic Problems and their Singular Perturbations. At the February 24 NAS New Technology Seminar, Petr Vanek from the University of California, Los Angeles, gave an overview of smoother aggregation iterative methods for black-box multigrid solvers, emphasizing algorithms and the underlying mathematical principles. He also discussed a generalization suitable for treating the convection-diffusion problem, and demonstrated the efficiency of the method on industrial problems on unstructured meshes.

Irregular Subdivision and Signal Processing for Arbitrary Surface Triangulations. Recent progress in 3-D acquisition techniques and mesh simplification methods has made triangulated mesh hierarchies of arbitrary topology a basic geometric modeling primitive. At the February 17 NAS New Technology Seminar, Peter Schroeder from Caltech presented some of his recent work aimed at building signal processing type algorithms for unstructured surface triangulations.

An Infrastructure for Code Transformations and Analysis. Francois Bodin from IRISA/INRIA (France) presented a new program

Within This Article...

[Multigrid Solvers for Elliptic Problems](#)

[Processing for Arbitrary Surface Triangulations](#)

[Infrastructure for Code Transformations and Analysis](#)

[Modular Reconfigurable Robotics](#)

[VLSI for Signal Processing](#)

[ATR's Artificial Brain Project](#)

[Data Mining of Satellite Data](#)

[Legion: Applications Perspective](#)

[Protocol and Environment for Flow Solvers](#)

engineering infrastructure called TSF at the February 9 NAS New Technology Seminar. TSF is designed for the application developer rather than the compiler writer, and provides a customizable infrastructure for the traditionally time-consuming program transformations involved in code maintenance, tuning, or parallelization.

Modular Reconfigurable Robotics. At the February 4 NAS Nanotechnology Seminar, Mark Yim from Xerox Palo Alto Research Center discussed modular reconfigurable robots and presented some of the early work on modular robot locomotion done at Stanford in 1994. He covered more recent work on reconfigurable systems composed of 200 modules, and a new system in which reconfiguration is the main goal.

[Condor: Decade of High Throughput](#)

[Visible and UV Semiconductor Lasers](#)

[Surface Grids from CAD Data](#)

[MPWENO Schemes and Accuracy](#)

VLSI for Intelligent Signal Processing. Professor Dan Hammerstrom of the Oregon Graduate Institute of Science and Engineering discussed intelligent signal processing at the January 19 NAS New Technology Seminar. Massively parallel ISP models will enhance rather than replace existing computational paradigms, he predicts. Within 10 years, massively parallel, biologically inspired models for intelligent signal processing will become a significant component of general computing.

ATR's Artificial Brain (CAM-Brain) Project: Evolving Neural Network Circuit Modules Using FPGA-based Hardware in About One Second, and Assembling 10,000s of them into Humanly

Architected Artificial Brains. The artificial brain was the topic at the January 7 NAS Nanotechnology Seminar. Hugo de Garis from the Brain Builder Group at Japan's ATR Labs said that by 2001, ATR's CAM-Brain Project aims to build an artificial brain with a billion artificial neurons, using neural circuit modules based on evolved cellular automata (CA). These modules are downloaded into a large RAM space and updated fast enough for real-time control of a "kitten" robot called "Robokoneko" (in Japanese, ko means child and neko means cat).

Data Mining of Remotely Sensed Satellite Data. At the December 17, 1998, Information Power Grid Seminar, Thomas Hinke of the University of Alabama in Huntsville gave an overview of data mining,

the extraction of "hidden" information from the vast amount of data generated by many of today's scientific enterprises. He discussed some of the techniques that have been developed at UAH to mine data collected by Earth-observing satellites.

Legion: An Applications Perspective. On December 15, 1998, Andrew Grimshaw from the University of Virginia presented a "reflective metasystem" project called Legion. He explained that Legion is designed to provide users with a transparent interface to widely-scattered computing resources. Grimshaw presented the key capabilities of the Legion project, described the system architecture, and presented the system as viewed by users.

An Interface Protocol and Generic Remeshing Environment for Flow Solver Systems. Vincent Harrant from CFD Research Corp. presented a multidisciplinary software environment called MDICE at the December 8, 1998, New Technology Seminar. The environment can be used for several practical multidisciplinary applications that require grid remeshing. These include unsteady moving or deforming CFD calculations with prescribed motion of bodies, motion of bodies governed by 6-DOF analysis, fluid-structure interaction problems, store separation problems requiring integration with dynamics/kinematics equation solvers, flow induced feedback control simulations, and others.

The Flight of the Condor: A Decade of High Throughput Computing. At the November 19, 1998 Nanotechnology Seminar, Miron Livny from the University of Wisconsin discussed high-throughput computing (HTC) environments and said that the key to HTC is effective management and exploitation of distributively owned computing resources. Using the Condor resource management environment, scientists can harness resources previously unavailable resources (see article, page 3). Livny outlined the layered Resource Allocation and Management architecture of Condor and discussed the interactions between layers.

Visible and Ultraviolet Wavelength Semiconductor Lasers. Group-III nitride light-emitting diodes and lasers emit light in the visible to ultraviolet wavelength range, making them important for many optoelectronics applications. The gain mediums in these devices also exhibit interestingly strong many-body Coulomb effects. At the November 17, 1998, New Technology Seminar, Weng Chow from Sandia National Laboratory described a study treating the nitride active medium as a strongly interacting electron-hole plasma. He discussed the

interplay of Coulomb and bandstructure effects on laser properties, and compared theoretical and experimental results.

Surface Grid Generation from CAD Data. On October 15, 1998, Dr. Jin Chou from MCAT, Inc. at NASA Ames Research Center demonstrated a surface grid generator called Superg, which is based on the Boundary Representation (Brep) data format and allows users to create a surface grid across multiple surfaces. Chou described Superg's scripting ability, gave a retrospective of the project, and summarized lessons learned about automatic grid generation.

MPWENO Schemes with Increasingly High Order of Accuracy. At the October 13, 1998, New Technology Seminar, Dinshaw Balsara from the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign described monotonicity-preserving weighted essentially non-oscillatory (MPWENO) schemes for direct numerical simulations and large eddy simulations of compressible turbulence. He presented several test problems in one and two dimensions, and showed that for multidimensional problems where the flow is not aligned with any of the grid directions, MPWENO schemes have a substantial advantage over older total variation diminishing (TVD) schemes.





► [Main Menu](#)

In this issue...

[Software Helps Chemists See Atoms in a Different Light](#)

[New Network Design Proves Faster, Cheaper](#)

[NAS Condor Population Expands to Hundreds](#)

[NAS Systems Prepared for New Millenium](#)

[New Directives-based Parallel Benchmarks in Beta-test](#)

[High-speed Processor Techniques](#)

[Recent Technical Seminars Available on Video](#)

Credits

Executive Editor: Bill Feiereisen

Managing Editor: Wade Roush

Contributing Writer: Terry Nelson

Image Coordinator: Chris Gong

Web Work: Peter Adams

Special thanks to: Lynn Albaugh, Richard Anderson, Johnny Chang, Christine Cortez, James Donald, Jill Dunbar, Al Globus, Ana Grady, Dominic Hart, Chris Henze, Randy Kaemmerer, Chris Kuszmaul, Creon Levit, Miron Livny, Preston MacDougall, Bill Nitzberg, Ray O'Brien, Marcia Redmond, Scott Santiago, Bill Thigpen, Steve Traugott, Eugene Tu, Alex Woo, John Ziebarth, and all of the researchers, engineers, managers, technical staff, programmers, writers, editors, proofreaders, designers, illustrators, interns, and others who have contributed to the success of *NAS News* since its inaugural issue in June, 1986.

 **Subscribe**

In this issue...

[Software Helps Chemists See Atoms in a Different Light](#)

[New Network Design Proves Faster, Cheaper](#)

[NAS Condor Population Expands to Hundreds](#)

[NAS Systems Prepared for New Millennium](#)

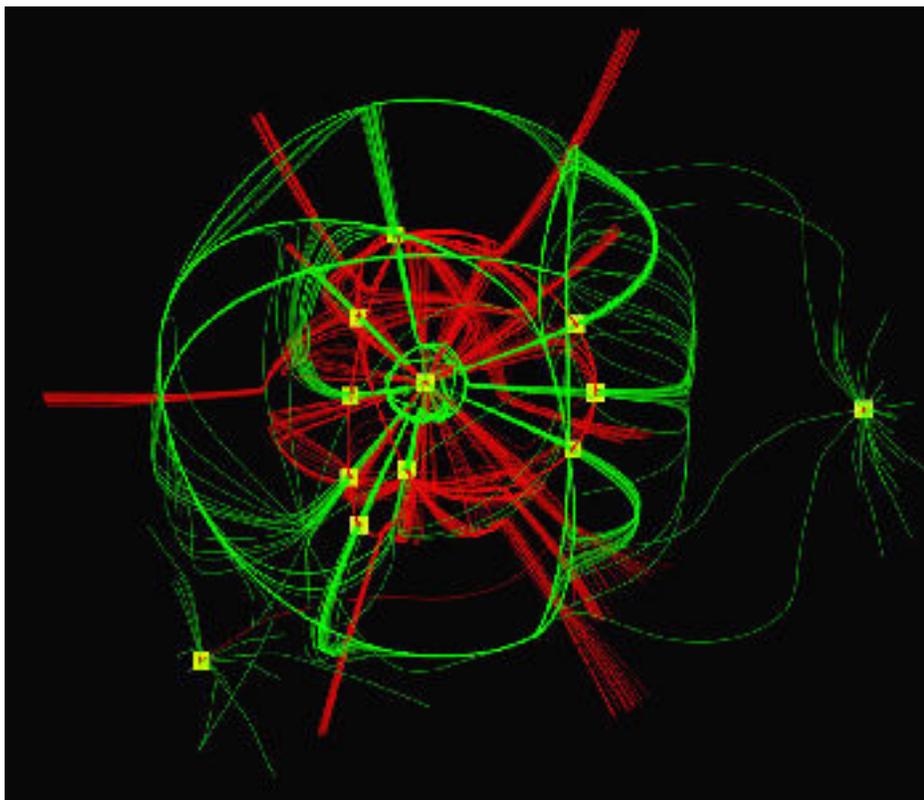
[New Directives-based Parallel Benchmarks in Beta-test](#)

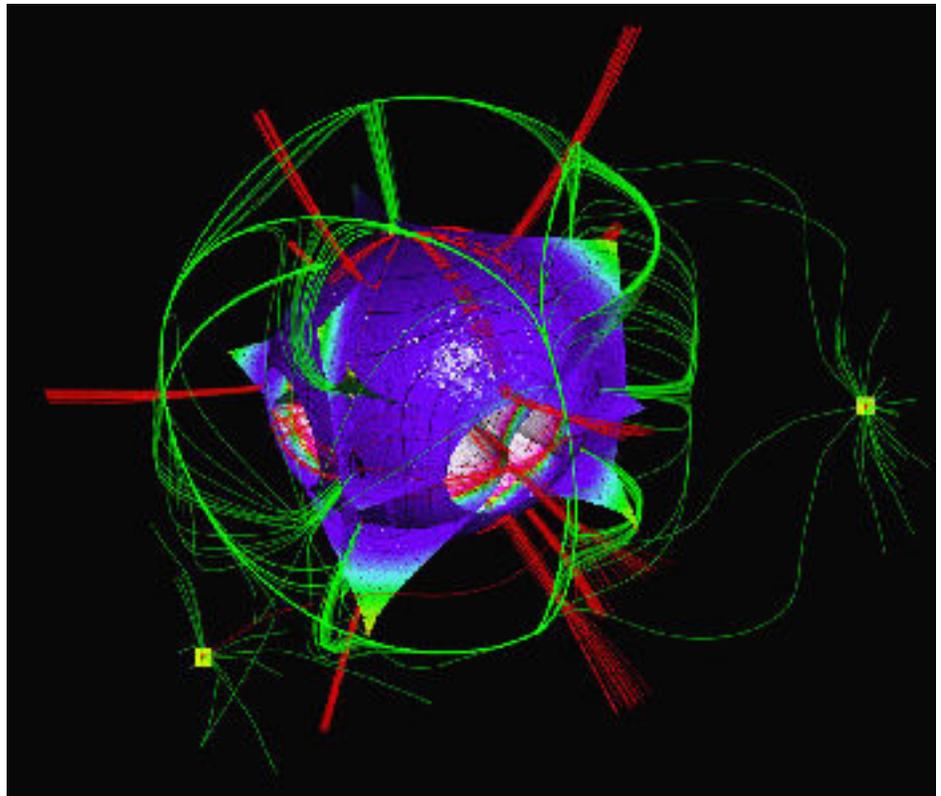
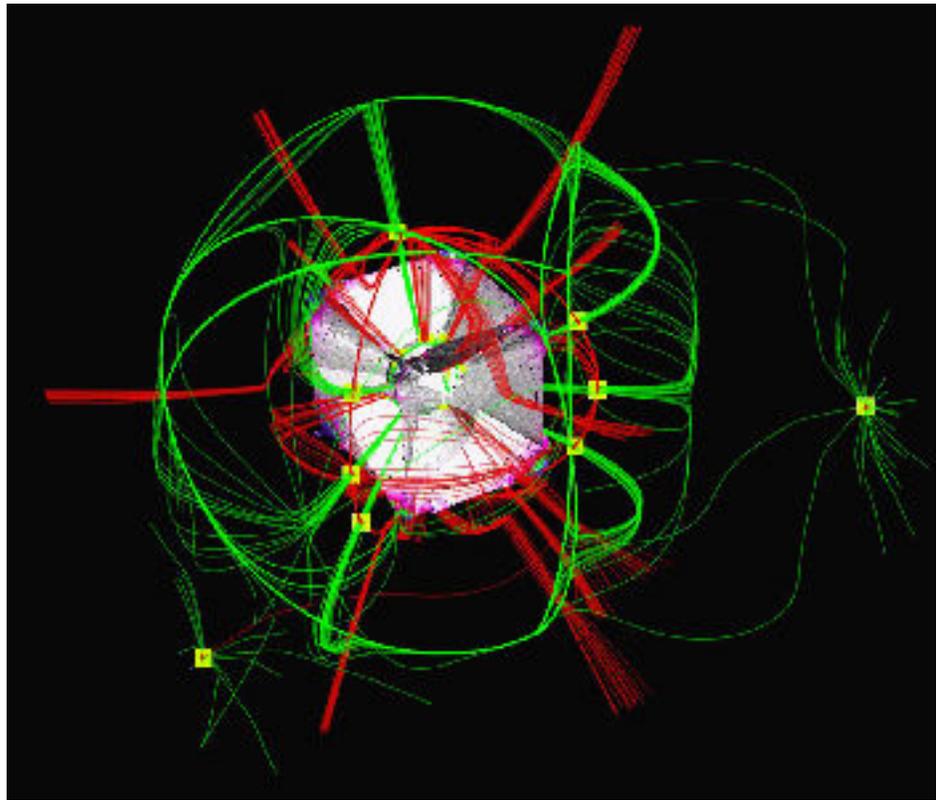
[High-speed Processor Techniques](#)

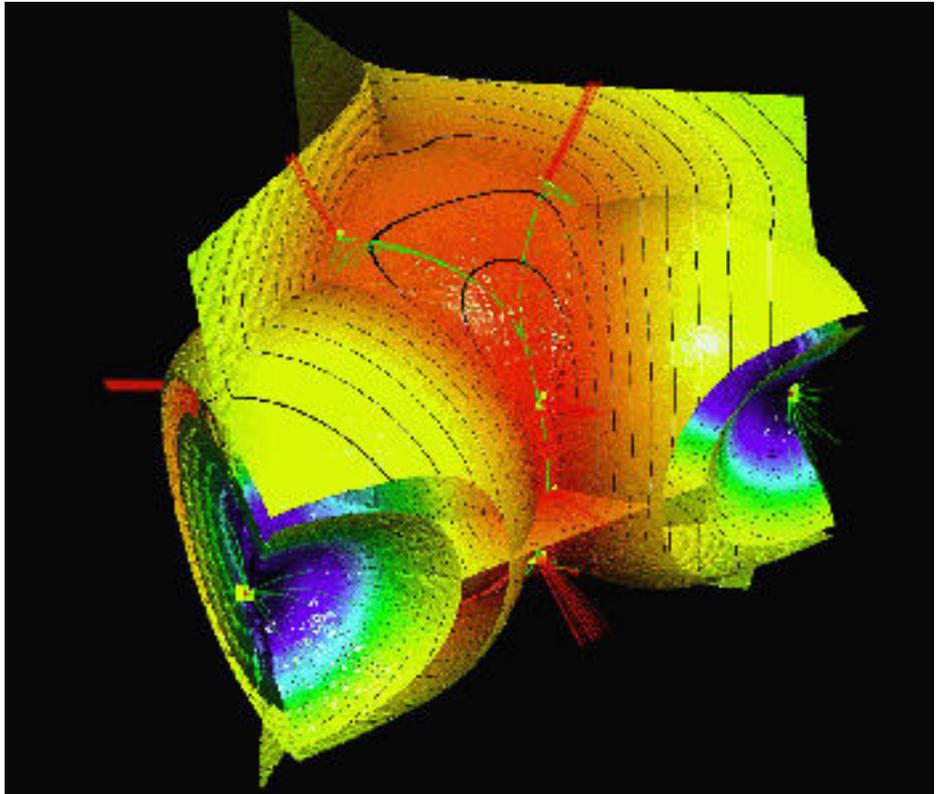
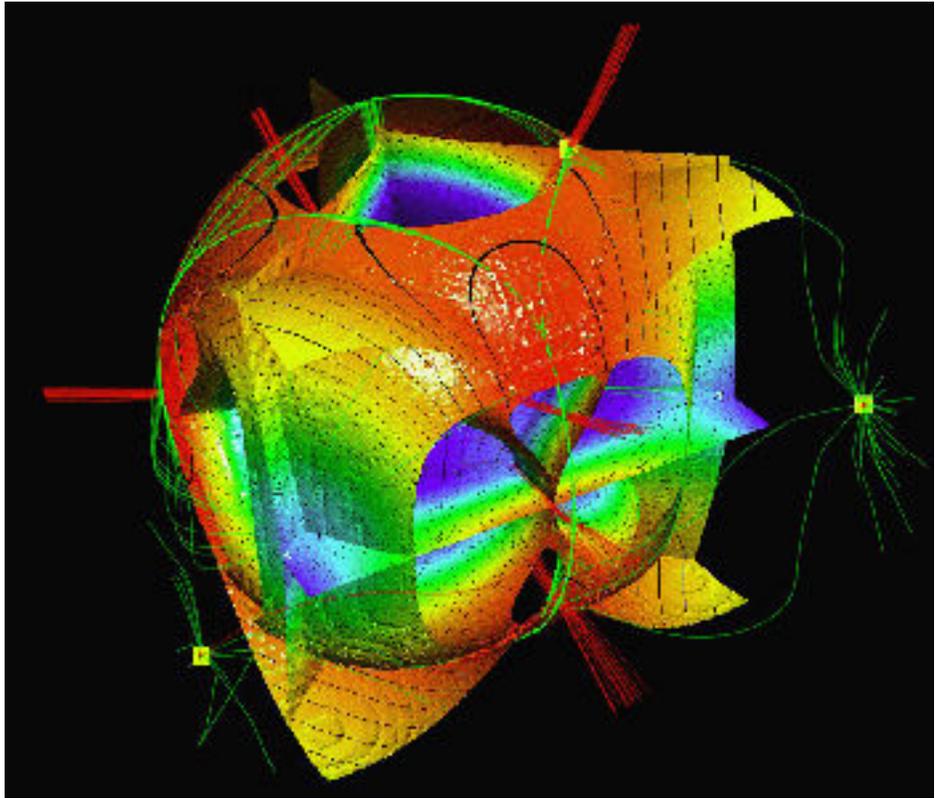
[Recent Technical Seminars Available on Video](#)

An Example of Quantum Topology

Quantum topology can reveal the complex inner landscape of a water molecule. The yellow squares in this series of portraits, variously clipped to reveal internal structure, are "critical points" where the Laplacian of the electronic charge density reaches a local maximum. The green and red lines are "trajectories" showing the paths massless particles would follow if they were swept along between these points in a vector field defined by the gradient of the Laplacian. "Zero-flux surfaces" or surfaces crossed by no trajectories, shown here in rainbow colors, are thought to predict how a molecule will fit together with others. All datasets by Chris Henze.







[Return to main article](#)