CENTER FOR HIGH PERFORMANCE COMPUTING
NEWS

## CROMDI'S INTERDISCIPLINARY TEAM IMPROVES DECISION MAKING AND CREATES NOVEL ART

Stefano Foresti, Director of CROMDI, Senior Scientist of CHPC; Julio Bermudez, Associate Professor of Architecture; Jim Agutter, Assistant Director of CROMDI, Research Asst. Prof. of Architecture; Frank Drews, Assistant Professor of Psychology; David Strayer, Associate Professor of Psychology; Elizabeth Tashjian, Associate Professor of Finance; Dwayne Westenskow, Professor of Bioengineering, Anesthesiology; Noah Syroid, Researcher of Biomedical and Computer Engineering

The Center for the Representation of Multi-Dimensional Information at the University of Utah has developed a new audio-visualization technology (IntuInfo) that facilitates monitoring and decision making with abundant, complex, and quickly changing data. This patent pending technology has been applied in medicine, finance, and entertainment. CROMDI is seeking partners and funding to prove its efficacy in information rich applications, such as: networking, security, process control, vehicle dashboard, traffic control and corporate management.
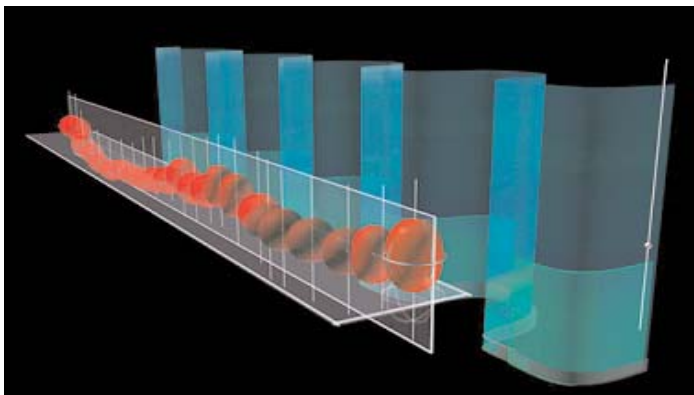
IntuInfo+Anesthesia shows 13 variables on a single screen in real time (Heart Rate: HR; Diastolic, Systolic and Mean Blood Pressures: DBP-SBP-MBP; Arterial Oxygen Saturation (Sa O2 Respiratory Rate (RR), Tidal Volume (VT) and gas inspired and expired content (O2 and CO2).

IntuInfo can maximize information per screen space, integrate many variables, enable comparison to normative values, simultaneously display present and historical data, interact to see global and local contexts. Because of its intuitive features, IntuInfo enables its users to recognize, understand and act on events faster, more accurately, with less mental effort, and with less training than is possible using existing visualization technologies.

CROMDI is an interdisciplinary team comprised of experts of Arts, Architecture, Communication, Computer Science, Mathematics, Music, Psychology, Medicine, Anesthesia,

BioEngineering, and Finance, that participate with their own unique perspectives through a unique methodology that has been refined over the years.

Anesthesia. Anesthesiologists face unexpected incidents during 20 percent of all anesthetics. One quarter of these incidents represent critical events posing significant danger to


Cardiovascular and Pulmonary Displays organized to show functional relationships between variables and to increase pattern recognition for disease states.

patients. Therefore, quick and accurate decisions are of major importance in anesthesia. The environment is stressful and the task is difficult because 30 or more variables need to be monitored and mentally correlated and integrated. The CROMDI team developed working prototypes that significantly reduce recognition times for detecting, diagnosing and treating anesthesia-related critical events. Testing showed a statistically significant decrease in detection time in several critical scenarios:

- Clinicians detected anesthesia-related critical events sooner (3.1 vs. 5.5 min)
- Abnormal events were diagnosed more accurately (error rate 1.1% vs. 4.1%)
- Problems were corrected in one-third the time (17 sec vs. 45 sec)
- Drug delivery was better controlled (EC95 error 21% vs. 44%)

The technology is being commercialized in partnership with manufacturers of medical displays.

Finance. Investment professionals and individuals have access to a wealth of information about companies, real-time,
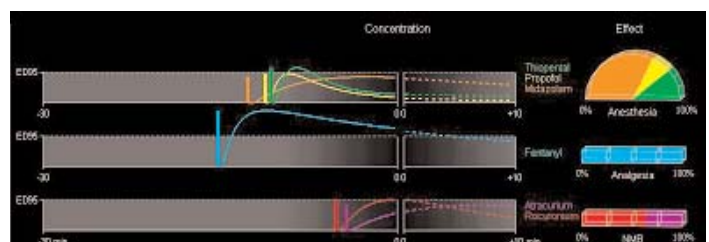

Drug Display designed to aid the clinician in monitoring and predicting the concentrations and effects of drugs administered during a surgery.

IntuInfo+Finance displays four zoomable views: Macro, Middle (details in the top and bottom right images), Micro (zoomed and described below), and a Custom Toolkit.
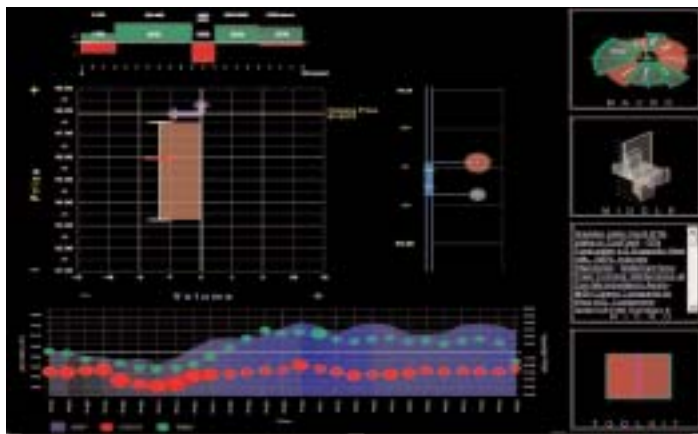
Micro-View displays three objects showing the current information on a single stock. Quote information: price, bid, ask, spread, bid size, ask size, and reference to typical values. Trade information: current price, prior close, day's range, standardized trading volume. "Who is in the market": volume segmented by trade size and by up and down volume. Trending Micro Display: daily return and volume information (filled circles) for two stocks.

historical and comparative market data. They want access to more data, but complain about too much data on their displays: this problem seems contradictory and it can not be solved with current display technology. CROMDI developed prototypes that integrate large quantities of data in a way that will lead the user to make rapid and accurate decisions. A macro and middle level displays, allow the trader to quickly view the performance of sectors and stocks in a portfolio. The micro level display allows the trader to analyze the current information on a single stock and quickly view its trend. A customer specified toolkit allows user to determine the form of objects and select financial variables to
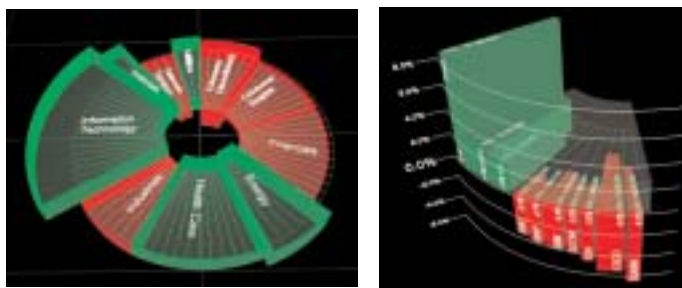
Macro-View shows day's price change, volume, and percentage holdings for the sectors in an index or portfolio. Middle-View shows the same variables for the stocks in a selected portfolio.

be displayed.

Entertainment. The CROMDI Arts Team has worked in collaboration with Yacov Sharir, an internationally renowned multimedia choreographer, to design, build and perform an interactive "artistic" audio-visual expression of a performer's movement and physiology. Monitoring equipment is mounted on the dancer: the data produced during the dancer's performance drives the artistic audio-visual display, based on Team's algorithms and design. This is the first technology-mediated performance of this kind. It has been performed nationally and internationally.

Dancer drives the visual expression.

NEW META-CLUSTER FOR BIOINFORMATICS AT CHPC

Julio C. Facelli, Center for High Performance Computing, University of Utah

The National Center for Research Resources (NCRR) from the National Institute of Health (NIH) has approved a grant to the Center of High Performance Computing to build a "meta-cluster" to enable the new generation of bioinformatics applications under development at the University of Utah. Meta-clustering moves beyond the concept of a single general purpose computational cluster towards a collection of specifically tailored computational elements optimized to solve certain classes of problems.

The CHPC meta-cluster will include a computational "cycle farm", a tightly coupled cluster with fast internode communication for optimitized parallel processing, a visualization cluster and a tightly coupled data intensive storage/processing unit for data mining applications. *Figure 1* provides a general overview of the design of the system. Preliminary discussions with several vendors indicate that the available funds will purchase a system with a total computer capacity exceeding the 2 TFlops mark.

Using highly advanced scheduling techniques (Maui/Silver) each of the separate elements will be seamlessly integrated (from a user's perspective) into what appears to be a single resource. CHPC will use GRID computing tools to facilitate the development of applications that may use several components of the meta-cluster or GRID resources available elsewhere. Based on our extensive experience in building and operating clusters of commodity hardware, we have designed the meta-cluster to not only provide the necessary resources for our emerging set of applications, but done so in the most cost effective manner. The applications planned for the meta-cluster include epidemiological studies of populations, investigations of the mechanics of knees and organ and atomistic level simulations of various biomolecular systems. The principal
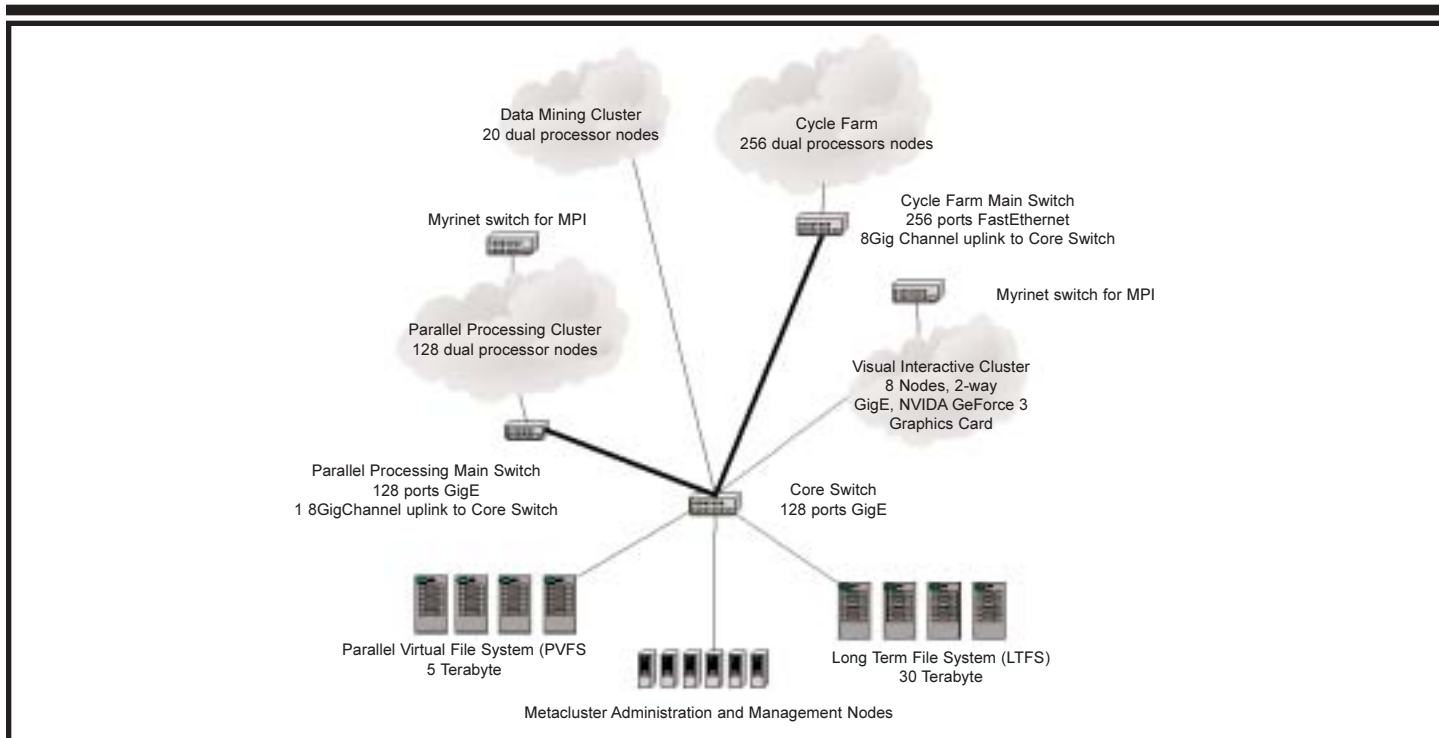
Figure 1: Bioinformatics Meta-Cluster Design

research projects that will make use of this new facility are those from:

- *Prof. Lisa Cannon-Albright, Medical Informatics: "Heritable Mechanisms Underlying Individual Variations in Disease and Treatment Response for Prostate Cancer."*
- *Prof. Thomas E. Cheatham, III, Medicinal Chemistry: "Simulation of the Structure and Dynamics of Nucleic Acids."*
- *Profs. David M. Grant, Ron Pugmire and Julio Facelli, Chemistry and CHPC: "Carbon-13 Magnetic Resonance: Methods and*
  Theory" and "Biological Applications of Parallel Genetic Algorithms."
- *Prof. Gregory A. Voth, Chemistry: "Simulations of Proton Translocation in Biomolecules" and "A New Method for Biomembrane Simulations."*
- *Prof. Jeffrey A. Weiss, Bioengineering: "Mechanics of the MCL in Normal and ACL-Deficient Knees."*
- *Prof. Robert B. Weiss, Human Genetics: "Mouse Genome Scaffolding" and "Full-Length Sequencing of Cancer Related cDNAs."*

The main componets of the meta-cluster are:

Parallel processing cluster:

The purpose of this subsystem is to provide resources for the highly parallel applications supported by this meta-cluster. Research programs needing this component make extensive use of parallel computing techniques and have demonstrated that their codes can scale well with the number of processors. This sub-cluster has been designed with parallel applications in mind. The processors will be connected via dual networks, GigE for data transfer to external storage and cluster management and a Myrinet low latency network for interprocessor communications (MPI). We have selected this due to the significant grain in

latency over standard protocols. It is expected that this subsystem will be used for highly parallel jobs, with a minimum of 32 processors allocated to each individual run. It is expected that a few highly parallel jobs a time will be scheduled in this partition.

Data mining cluster:

The purpose of this subsystem is to provide resources for jobs requiring relatively low parallelism and high data transfer rates. For this purpose these nodes are equipped with dual processor units that are able to overlap data processing and data movement. These nodes are directly connected, via GigE to the core switch which provides direct access to both the Parallel file system and the Long term storage facility.

Visualization cluster:

The bio-informatics applications supported by this metacluster will produce large amounts of data; the analysis of these data, in many cases will be well beyond the capabilities and resolution available in desktop systems. To enhance the analytic capabilities of the meta-cluster we propose to add a modest distributed visualization system. The design of the system follows closely the "Multigraphics" model developed by Pat Hanrahan at Stanford University. This model provides high-end visualization resources using commodity components.

Cycle farm cluster:

A number of the programs to be supported by this meta-cluster need large amounts of computer time, but the applications that they use exhibit relatively low levels of parallelism. In order to achieve this goal in the most cost effective way the processors of this sub-system will be connected to its main switch via 100FastE, instead of the most advanced and expensive connectivity available in the parallel processing cluster. It is expected that large numbers of jobs will be scheduled in the "cycle farm" using a few processors each. To meet the I/O demands for this large number of jobs the "cycle farm" switch will be connected to the core switch via one 8GigChannel uplink.

Long term storage subsystem:

A 30 Tb system for long term storage will be attached to the core switch to provide long term storage for biomedical and simulation data. The system will be designed following CHPC experience in storage devices for long term read only file systems. The user will have read access to this data, and special software will be provided to deposit files in the subsystem. By making this sub-system "read only" it is possible to eliminate the need of costly backup systems.

Parallel Virtual File System:

Many of the jobs that are expected to run in the meta-cluster will require extensive intermediate or scratch files. Typically because these files are large and allow for the efficient execution of programs, they should be located in a highly efficient file server. This system of 5 Tb will be directly connected to the core switch and accessible by all the nodes in the system. An administrative node in this subsystem will be used to run PVFS and other administrative tasks. This system will be dedicated to scratch files and will not be backed up.

Service Nodes:

Several nodes directly connected to the Core switch have been added to the design to provide different services needed for the operation of the meta-cluster. In addition to two interactive nodes, for users to login to the system, nodes have been added for resource management, administration, network management and application serving.

PART 2: TIPS AND TRICKS FOR PROGRAMMING ON ICEBOX AND SIERRA TOOLS FOR TIMING USER PROGRAMS AND SMART WAYS TO CODE FOR SPEED

Martin Cuma, Center for High Performance Computing, University of Utah

In the Summer 2002 newsletter, we spent some time explaining ways to optimize application's performance using compiler flags and presented a simple user timing routine. In this article, we will follow up with list of tools that both Linux and Tru64 include for application profiling and suggest some relatively simple programming techniques that often result in a significant application speedup.

Simple timing

Timing of an application is beneficial for anyone who needs to run it repeatedly for many times. We have shown how to use a user-defined timing routine in the previous newsletter's article, however, this approach necessitates changes in the source code, which is often time consuming. There are several options that the system offers for timing. Runtime of the entire application can be counted by placing the UNIX command "time" before the executable name. This is often sufficient for a quick look on how changes in the code affect the performance and works with both in serial and in parallel.

Figure 1. Program profile display in pgprof.

Serial code profiling

For those who are unhappy with their application's speed, the first step should be to run it through a profiler to find out where the application spends most of its time, and concentrate programming efforts in this part of the code. In case the program is serial, there is a wealth of profiling tools available on almost any platform. Profilers are divided into several categories based on the method of the timing information collection.

Both Linux and Tru64 supply "gprof," which is based on Program Counter sampling at periodic intervals. The user must compile the executable with *-pg* flag (with GNU compilers on Icebox and Compaq compilers on Sierra) and run the executable. The data is collected in the file "gmon.out," and the results are displayed in a text format with the gprof command.

Sometime, it is worth it to find out how many times each line is executed. Coverage analysis tools can be used for such purpose, they annotate the source code with this information. Note, that these numbers are not based on time. In Linux, the coverage analysis tool is called "gcov," in the Tru64 "pixie." gcov works only with C code compiled with gcc (use flags *-fprofile-arcs -ftest-coverage* when compiling), while pixie works with both C and Fortran compilers.

The Portland Group compiler suite installed on Icebox also includes its own graphic profiler, "pgprof." Program sampling is invoked by compiling and linking the program with flag *-Mfunc*, coverage analysis with flag *-Mlines*. In both cases, after the program is executed, file "pgprof.out" is produced, that can be opened in pgprof and the profile data viewed in a simple and easy to use graphical format. The view can be further customized to view just the information required, and clicking on the profiled function displays the source code. *Figure 1* shows a sample profile window produced by pgprof.

Finally, some vendors supply advanced profiling tools that include profiling analysis to a programming environment allowing interactive source-based profiling information. One such tool for Linux, Intel (former KAI) VTune, seems to be quite promising and we are planning to evaluate it for a possible purchase.
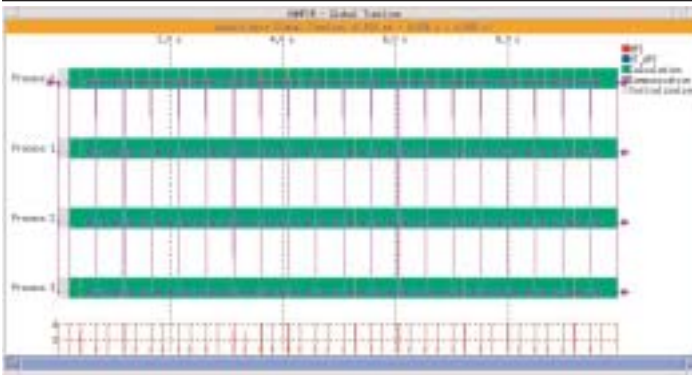
Figure 2. Vampir Global timeline view displaying user run time (green) and communication (red). Blue lines between processes represent communication calls.

Parallel code profiling

Parallel profiling constitutes further challenge due to the inter-process interaction. MPI includes a function MPI_Wtime, which can be used in the source code as a user-defined timer. Most of the serial profiling tools don't work well in the parallel environment, especially if the processors are distributed. There are several specialized tools available though. Public domain MPICH supplies simple Java-based parallel profilers, Snapshot and Upshot, which profile the MPI calls. However, a de-facto standard in the parallel profiling is Vampir/Vampirtrace suite of programs, which runs on many platforms. Vampirtrace is a library that users must link with the MPI application, and then run it to produce a trace file. Again, tracing is based on the periodic sampling, however, Vampirtrace is aware of the communication calls and accounts for them as well. The trace file is then displayed using a graphic tool, Vampir. The developer of Vampir is in the final stage of testing a fully overhauled application that will, apart from MPI tracing, also profile OpenMP parallelism and include other useful performance information, such as processor cache, memory and I/O utilization. The CHPC has licenses for both Icebox and Sierra with instructions how to use them detailed on our webpage [1]. **Figure 2** shows a sample process view screen produced by Vampir.

Compiler optimizations and optimized mathematical libraries

In the remainder of this article, we will list some programming practices that may be beneficial for speeding up the code on modern computer architectures. The easiest and probably single most useful approach is to compile with optimization flags.

```
do j=1,m                              do j=1,m,2
  do i=1,n                              do i=1,n,2
    do k=1,p                              f11 = c(i,j)
      c(i,j) = c(i,j) +                   f21 = c(i+1,j)
               a(k,i)*b(k,j)              f12 = c(i,j+1)
    enddo                                 f22 = c(i+1,j+1)
  enddo                                   do k=1,p
enddo                                       f11 = f11 + a(k,i)*b(k,j)
                                            f21 = f21 + a(k,i+1)*b(k,j)
                                            f12 = f12 + a(k,i)*b(k,j+1)
                                            f22 = f22 + a(k,i+1)*b(k,j+1)
                                          enddo
                                          c(i,j) = f11
                                          c(i+1,j) = f21
                                          c(i,j+1) = f12
                                          c(i+1,j+1) = f22
                                        enddo
                                      enddo
```

Figure 3. Example of loop tiling. Untiled loop on the left, and 2x2 tiled loop on the right. The tiled loop keeps values f11, f12, f21, f22 in the processor registers.

These are usually designed to take advantage of the computer specifics. We have devoted most of the article in our last newsletter to this topic. Using optimized libraries for standard mathematical, and other, operations is probably the second most important step. Among these libraries are *libm* for common mathematical functions, BLAS, Lapack, etc. for linear algebra and other specialized libraries, such as FFT,… Most of the time, these functions perform operations on large data sets and the library functions include optimized algorithms for optimal data access for the specific platform (e.g. cache and register utilization). These libraries are available on both Icebox and Sierra and their usage is detailed on our website [2].

```
do i=1,nele                  do j=1,nele
  do j=1,nele                  do i=1,nele
    y(i,j) = 2.0*x(i,j)          y(i,j) = 2.0*x(i,j)
  enddo                        enddo
enddo                        enddo
```

Figure 4. Example of loop interchange. Loop on the left accesses the data in very slow strided fashion (Fortran stores arrays in column-major order, i.e. index i changes the fastest). Loop on the right accesses the data optimally.

Data alignment

When writing the code, it is useful to pay attention to the data alignment, which can bring improvements in the data access speeds. Global, static and structure variables in C and common blocks in Fortran should be ordered from largest to smallest data type and variable size. Also dynamically allocated data and those being referenced by a pointer can take advantage of the memory alignment. The C function *malloc* usually aligns on 8 byte boundaries, finer alignment can be achieved by functions *memalign* and valloc. Some compilers issue warnings when the alignment is not optimal.

Loop optimization techniques

Substantial performance gains can be occasionally achieved by loop rearrangement, as loops are one of the most commonly used constructs. Compilers can be told to perform various loop optimizations automatically but in some cases source code modifications can enhance compiler's analysis. Loop source code changes should be re-evaluated with the change of compiler version or change of architecture. Compilers usually perform efficient loop unrolling, reducing overhead for loop setup. Loop tiling, i.e. splitting a multiple loop problem such as a matrix multiply into smaller "tiled" loops, can speed the execution by reusing the data in the registers and in the cache. An example code in **Figure 3** illustrates this. While the original loop has a ratio of floating point to memory operations $2mnp/(2mnp+2mn)$, the tile loop has it as $2mnp/(mnp+2mn)$. In architectures with fast CPU and slow memory, which is the case for most modern computers, this can result in a noticeable performance gain.

Another good programming habit is to set up multiple loops to minimize data stride and align access pattern in the loop with how the data is stored in the memory. When this is not the case, loops should be interchanged, as illustrated in **Figure 4**.

Adjacent or closely spaced loops should be fused together. This decreases loop overhead and increases computational density, which enables improvement in software pipelining.

Most of the compilers perform fusion, but in some cases it is advantageous to do it manually.

Opposed to loop fusion is loop fission, splitting single loop into multiples. This is especially useful when the original loop contains conditional. One loop is then conditional free, while the other retains the condition. As most modern processors have deep pipelines that feed the instructions and data to the processing unit well ahead of time (to alleviate slow memory access), the code runs more efficiently in the conditional free loop. Naturally, if the conditional does not depend on the loop variables, it should always be outside the loop.

Conclusions

In this article we first try to familiarize the users with system tools that can be used to time their application's execution and find its bottlenecks. In the second half, we provide some more tips and programming techniques that can have a beneficial impact on the program performance. As always, if further questions related to these topics arise, we will be happy to answer them.

References

[1] CHPC's Vampir webpage http://www.chpc.utah.edu/index.php? currentNumber=3.2.37

[2] CHPC's Mathematical Software webpage http://www. chpc.utah.edu/index. php?currentNumber=3.2.18

POSTERS ON THE HILL 2003

Robert McDermott, Staff Scientist for Visualization, Center for High Performance Computing

Along with their faculty colleagues, undergraduate students participating in research projects, (who had graduated from a high school in the state of Utah) were asked to exhibit their posters in the rotunda of the Utah State Capitol on January 23, 2003. This year there were 30 students and their faculty associates participating. This event was organized at the request of the Office of Governmental Affairs and the Office of the Vice-President of Research. The students received assistance in the creation of the posters which were intended to communicate at a visual level with the legislators. This event was not intended to be overly academic, but rather to expose the legislators to the excitement and enthusiasm of the undergraduate students who have received this opportunity of working with University of Utah research faculty.

The Office of the Vice-President of Research, represented by Ron Pugmire, solicited student and research faculty to participate in the event. There was a balanced representation of academic disciplines. geographic representation across the state, and the gender of the students. A commitment from each faculty member working with their student was vital for a successful experience.

The University Office of Governmental Affairs (represented by Nancy Lyons and Kay Clark) sent hand written notes to legislators representing the district where participating students went to high school. These letters informed the legislator that a former student of their district had a poster at the rotunda. This follow-up communication facilitated a connection between the legislator and student.

The day of the event was loosely structured. The students

arrived at 8am, were provided a light breakfast, and remained until 1pm. The legislators had opportunities to see the posters and visit with the students on their way into the session, between committee meetings and on their way out for the day. Since many committee meetings were shortened during this first week of the legislative session, many legislators took advantage of the opportunity. It has been our experience over the past three years that most legislators visit their student constituents. They are very courteous and attentive to the student's which is very encouraging to both students and university organizers.

I worked with the students and faculty at the request of the Ron Pugmire. My duties included laying out the posters and handouts, following up on getting posters printed, laminated, mounted, and transported with easels to the rotunda. This year I took students choices for bagels, cream cheese, and beverages for their breakfast at the capitol.

Over the last three years I have prepared a somewhat common layout for all the posters in that year, the underlying premise being that they have a very visual presence. The common framework for the poster design is also a way of exhibiting the University's support for this effort in lobbying the legislators. This year I received a quality contribution from Nathan Galli of the Scientific Computing and Imaging Institute. He produced a stunning background for this year's posters from a photograph of campus and the mountains to the east. The strong visual component for the posters was chosen to help the students more easily communicate with the legislators in the very brief period of time that they are able to visit. Posters with fewer words relied on a handout where there was considerably more information to help the legislators get a more in-depth understanding of the student's work.

Each year a serious effort is expended to help with this lobbying effort including the Associate Vice President of Research, Ron Pugmire, attending the exhibit. This January Dr. Pugmire was pleased with the interaction between the students, their posters, and the legislators. In addition the Assistant Vice President of Governmental Affairs, Nancy Lyons, attended and she expressed a positive response to the appearance and the presence of the students with their posters. With these positive responses there may well be a Posters on the Hill Exhibit in January 2004.

■