This handout is meant to serve as a “getting started” document for parallel programming on the SGI/Cray Origin 2000 at NCSA. It briefly describes the architecture and programming environment on the Origin, focusing on support for message-passing programming (using MPI), and shared-memory programming (using ANL macros).

I encourage you to use the local Linux Beowulf cluster for all of your development and as many timing runs as possible. Clearly, you will be limited by the size of the cluster (14 nodes). For Homeworks 3 and 4, this is not a limitation and the Origin will only be used for comparative performance measurements (e.g., how does the lower communication overheads on the Origin affect your performance). The Origin accounts were primarily set up to ensure that the class could proceed without disruption even if we ran into some unanticipated systems problems with the cluster machines. They also provide you access to a much more scalable and high-performance machine (128 or 256 processors) in case you need one for your course project.

Accessing the Origin

Secure shell (ssh) to modi4.ncsa.uiuc.edu. Those of you who have registered for the course should have an account on the machine. Please let me know at the earliest if this is not so. You should change your default password the first time you log in.

Architecture

The SGI Origin 2000 is a cache-coherent non-uniform memory access (ccNUMA) multiprocessor. The Origin system can consist of up to 512 nodes interconnected by a scalable Craylink network. Each node consists of one or two R10000 processors, up to 4 GB of coherent memory, and a connection to a portion of the XIO IO subsystem. Look at http://www.ncsa.uiuc.edu/SCD/Hardware/Origin2000/ for additional details.

Programming and Execution Environment

The Origin runs a variant of the Cellular IRIX (a Unix variant) operating system. Standard UNIX programs as well as all the GNU Utilities are available in /usr/ncsa/bin.

All your programs will be written in the C language. To compile these programs for parallel execution you should use the Origin C compiler available as /opt/MIPSpro/bin/cc. A sample makefile is provided in my home directory (~vijayk/Homework3/) on modi4.ncsa.uiuc.edu which contains the compiler options you will need for Homework 3.

In Homework 3, you will use MPI to create a message-passing executable. These executables need to be run using the mpirun command: the number of processors is specified using the -np option. Look at http://www.ncsa.uiuc.edu/SCD/Hardware/CommonDoc/MessPass/ for further details about MPI and mpirun.

In Homework 4, you will use the ANL macros to create a shared-memory executable. Additional details will be provided along with the Homework 4 description.
For performance measurements, you will need to use the batch submission system instead of running the program interactively. The NCSA Origin runs a job manager called lsbatch. Look at http://www.ncsa.uiuc.edu/SCD/Hardware/Origin2000/Doc/Jobs.html for example job submission scripts and additional details about running jobs in both interactive and batch modes on the NCSA Origin. The documentation will tell you that to achieve reasonable confidence in your speedup measurements, you need to use the dedicated batch queues. **However, please use the dedicated queues very sparingly: you are charged for all of the processors making up the queue, even if your job runs on a significantly smaller subset.** In particular, on the Origin, the smallest-sized dedicated queue has 128 processors. For every hour of CPU usage on this queue, you will be charged 128 SUs. To put this in perspective, the combined allocation for everyone in the course for the entire semester is 5000 SUs. Needless to say, your runs for Homeworks 3 and 4 should be a few minutes (at most) each, so there should be plenty of units to go around. This however assumes that you have debugged your program runs before submitting something to the dedicated queue.

Finally, to debug your parallel program and profile its performance, use the cvd utility (the X-windows based Workshop debugger). Look at the man pages for additional details. Of course, you should make every effort to ensure that your program is correct (that it runs on one node of the Origin) before you try and run it with multiple processors. The Origin also has hardware performance counters that let you track things like secondary cache misses. These performance counters are accessible from the perfex and SpeedShop programs. Look at the *SpeedShop User’s Guide* in insight (the X-windows based document browser) for additional details.


Let me note that I do not expect you to put in all the optimizations that this document describes.

**Information Resources**

In addition to the URLs provided above, the following are good sources of information about the SGI Origin.

- The NCSA resources on the SGI Cray Origin 2000:  

- The Origin itself has a very nice X-windows based document browser called insight which contains a searchable database of all documents relevant to the Origin.