Newton Changes

Kevin Roy
Senior HPC Consultant, Manchester Computing, University of Manchester

A history

Newton was brought into service on 1st October 2003, since then Newton has proved to be a welcome addition to the CSAR service. It has blended the high performance interconnect from SGI that had made the Origin series successful, with high performance processors from Intel. Many applications have seen great improvements over Turing and Green, but one of the down sides of the machine was the relative immaturity of the system.

The compilers have improved markedly since 2003; the compilers on the early system contained a number of internal compiler errors and many applications required specific compiler revisions. Intel's history can clearly explain this – Intel's long history in desktop computing had led to a reliable C/C++ compiler, but their exposure to large Fortran codes wasn't quite as comprehensive until the Itanium 2 processor which was targeting the HPC market. Intel Compiler version 8.1 has proved to be more optimal and reliable for nearly all applications. This will be discussed in a little more depth later on.

The Itanium 2 processor was a relatively new processor; this means a lack of robust community or commercial software particularly in the development tools area. Another area which has suffered is the reliability of the operating system (OS). SGI has a long history of producing scalable operating systems (Green has always had a single OS run across its 512 processors), which it has tried to bring to the Linux world. Many additions have made it into the Linux core with other features being added to a list of OS upgrades called Propack. It is the features in Propack that allow the operating system to scale beyond the sizes of machines that Linux can, and allow the interoperability with the cluster file system (CXFS) that is used on the SAN.

Propack Updates

Newton has recently (26th April) undergone a considerable update in Propack version. This new version (Propack 3.0 service pack 4) has been applied to alleviate some of the stability issues. Propack 3.0 has been available for some time and so has undergone considerable testing from other sites, but since CXFS is the file system employed on Newton we have been awaiting CXFS 3.2 which was released earlier this year.

Two of the major problems seen on Newton have been SAN reliability and I/O buffering in the operating system. SGI have identified the SAN software as being responsible for a number of the Newton crashes with fixes being available in CXFS 3.2. Linux has developed from a multipurpose desktop environment where manipulation of files is one of the key tasks so the I/O buffering (areas of memory used for keeping copies of files for quicker access) is set to a high level. In a HPC environment this batch area memory should be dedicated to the batch job. It is also useful in interactive areas of the machine but there have been occasions when this memory has not been released, leaving the lower numbered processors (reserved for the OS) to be short of memory. This in turn may eventually affect the performance of the rest of the machine. Both of these problems have been addressed in the recent updates.

Propack 3.0 is based on a newer version of glibc, so with it comes increased performance of some of these routines, in particular some of the maths library functions (from libm). It has also included many other bug fixes and security updates.

A few caveats should be noted.

- Though nearly every code that has been tested has provided exact numerical reproducibility; this cannot be guaranteed and so a recompile may be useful – we have also changed the default compiler so this might be an ideal time to recompile and gain the advantages of the 8.1 compilers.
- Any calls to private glibc functions may fail to link with the new system.

Compiler Changes

Intel Compilers v8.1 have been tested extensively on Newton by CSAR staff and others, and have proved to be far more efficient and reliable and so have recently become the default compiler. This new compiler has:

- Provided far less internal compiler errors on Fortran codes. These internal compiler errors often meant that only one specific release of the 7.1 compilers would work for a given application.
- Much quicker compile times. There were a few instances of files taking several minutes to compile, this has now been reduced.
- Provides faster codes. As expected with newer compilers they produce faster codes.