The Optimal CPU and Interconnect for an HPC Cluster

Andreas Koch

Transtec AG, Tübingen, Deutschland

1 HPC cluster in general

The classic mainframe solutions in the HPC area consist of either SMP (symmetric multi-processor) or as MPP (massively parallel processing) systems. These are made up of incompatible limited-lot-production hardware devices that have been optimised to fulfil a special purpose by a single vendor. Those big-iron systems make use of shared memory architectures in which all processors share common main memory.

In contrast to that, HPC (high performance computing) cluster solutions are made up of a number of commercially available computer systems using off-the-shelf hardware that is not restricted to a particular application.

HPC clusters typically use distributed memory structures, that is a decentralised form of main memory scattered over individual nodes which is addressed by the CPU. The cluster is controlled by at least one administrative computer whose task is to make the system available to users with the help of a auxiliary program.

From the viewpoint of the user a cluster consists of a software interface whose purpose is making resources available to user applications. This interface, also referred to as the middleware layer, runs on top of the operating system. Jobs are assigned to processors over a dedicated network.

In what follows we will take up aspects of High Performance Compute Cluster that are relevant for procurement planning.

Fig. 1: transtec HPC cluster system (University of Leuven, Belgium)

2 Structure of an HPC cluster

In its ideal form an HPC cluster consists of a number of computation nodes with more or less identical construction and one or more access computers. As a rule, economical IA32/64 hardware is employed, the similar type of hardware used in personal computers. Typically, the number of nodes lies between 8 and 256; however it may run into the thousands. The access computer is referred to as the front end, server node or head node, while the computation nodes are simply called compute nodes, or compute for short.
The management network distributes the jobs to the compute nodes and passes the results back to the front end. Additionally, there can be a second network for the communication only between the compute nodes.

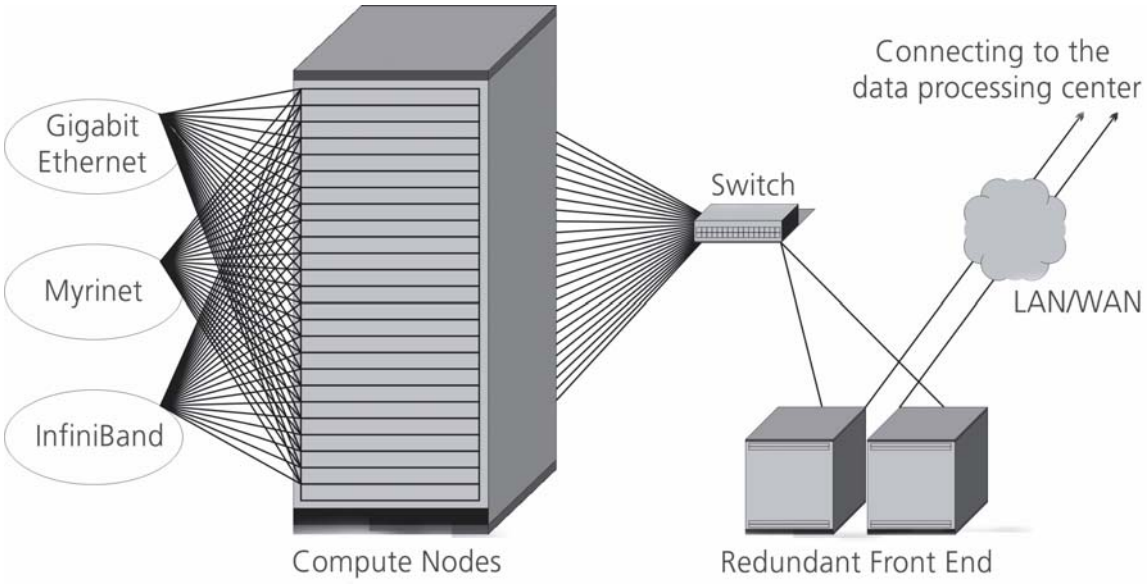

Fig. 2: Architecture of an HPC cluster in principle

3 Compute node optimisation

The optimisation of the compute nodes is one of the most important aspects regarding the investment.

The customer can choose between
- Several CPU types
- The right amount of memory
- SATA or SCSI drive (one, two - or even none)
- PCI-X or PCI express bus type
- High speed interconnect

The CPU decision and the choice of high speed interconnect will be discussed in detail in the next two chapters.
Regarding the memory, the answer is easy: Never let the system swap. And according the drive type, faster hard disks – may be two in RAID 0 – are only necessary, if the application is scratching intensively.
As the PCI-X bus is limited to below 1 GBit/s, it is recommended to look for PCI express in the nodes, if a high speed interconnect should be added.

4 CPU comparison

Experience gained from workstations can often prove valuable. Knowledge of the advantages and disadvantages of specific architecture can in principle be carried over to a cluster whose nodes use CPUs of the same type.
If such information is not available it should be acquired through pre-procurement tests; this is especially desirable in the case of large investments. Besides dual Intel Xeon and dual AMD Opteron configurations, logical candidates for testing are single P4 or Opteron systems and, of course, dual Intel Itanium (IA64) solutions.

Normally, the choice will be between Intel Xeon and AMD Opteron. The cost for a single CPU system in practice is slightly higher that for a dual CPU system, as several components like main board, power supply, interconnects, housing and rack can be share and therefore save money. To get the full speed out of an Itanium, the software has to be perfectly adapted to that architecture. And any 32 bit application will only run with a very poor performance.

To compare the Xeon with the Opteron, the user should run benchmarks with it own application. Those results should the basis for a decision. If there is a spread of applications planned to run on the cluster, the complete suite should be taken in account and the weighted results make up the individual benchmark.

If the usage of the cluster is not yet clear, benchmarks like the spec suite could be a helpful hint. Until summer 2006, the Opteron performed some 10% better than a classic Xeon. Since the new Woodcrest CPU is on the market, Intel is back and actually the better choice, unimportant which application will be chosen. Even according the power efficiency, Intel is going to beat AMD.

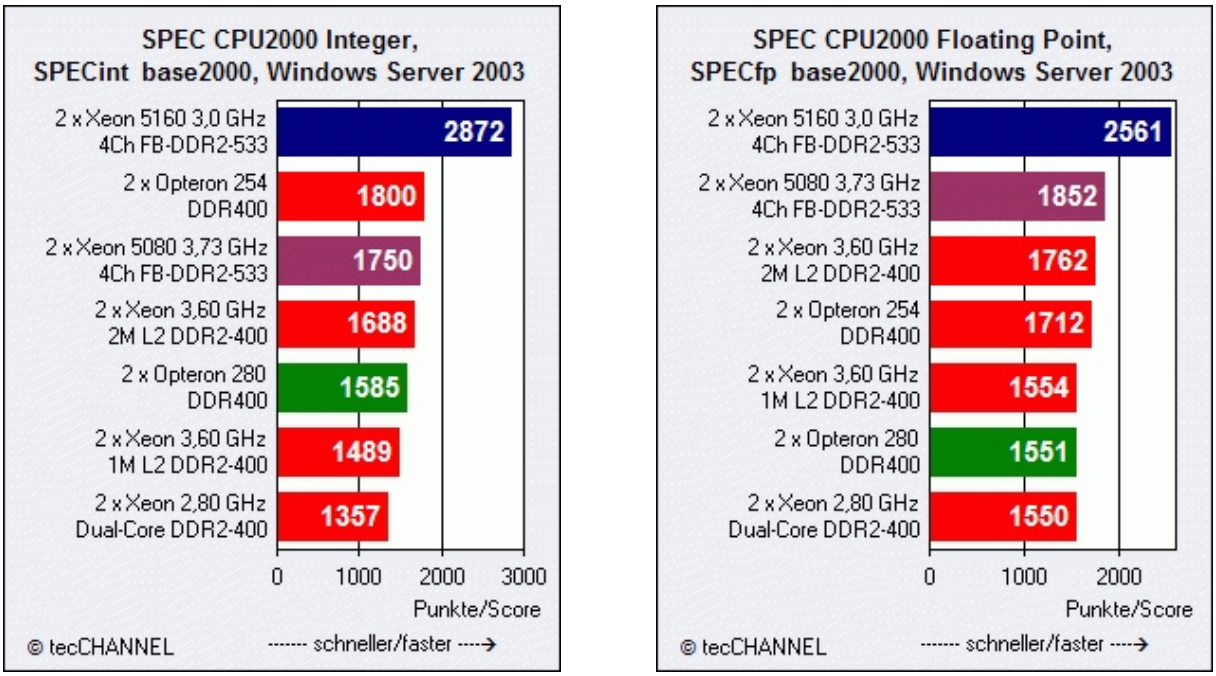

Fig. 3: Comparison of Intel Xeon and AMD Opteron based on Spec benchmark

For both processors, the Xeon and the Opteron, there are dual core versions on the market. Does it make sense to invest in that technology, and what would be the optimal frequency rate?
The speed up comparing a single core CPU with the identical dual core version would ideally be 200%. It's clear that for some overhead reasons, that can't be reached. For a first step in, the official information of each vendor can be a help.

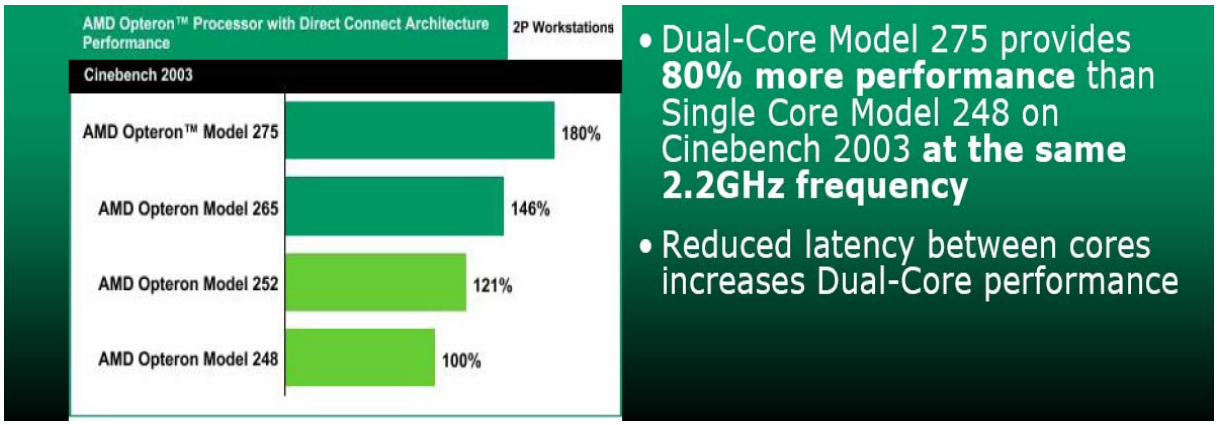

Fig. 4: Speed up of a dual core Opteron compared with a single core Opteron (Source AMD)

In Fig. 4, AMD compares the Opteron 248 (2.2GHz) with the Opteron 275 (2x 2.2GHz) and found out a speed up of 180%. But that not the case with every application. In a couple of benchmark, we found out values up to only 135%. The reason is the shared memory controller, which works for both cores (Fig. 3). If the application doesn't fit well in the 1 MB L2 cache that is dedicated to the individual core there is a lot of traffic to and from the memory. Every time, core #1 talks to the RAM, core #2 has to wait and is inefficient. – To find out the real life speed up, the users have to benchmark their own application with a dual core system comparing it with single core results.

A principal question is, which frequency would be the optimal to choose. Answering that, we also get a principal answer according the dual core CPUs.
If the application to be run on the cluster is very cost intensive, e.g. in the region of the hardware itself, then normally the usage of the fastest CPU available is recommended – as the price of the software licence is not bonded to the CPU frequency. If the software licence has to be paid on a core basis – and not for the physical processor – then the usage of single core CPUs is recommended, as the user has to pay 100% of the application licence, but gets out only between 80% and 35% of the CPU power.

5 Interconnect comparison

The properties of the network used to connect the compute nodes are determined by the types of applications to be run.

When relatively little data is exchanged a Gigabit Ethernet network is adequate. This is the case, for example, for batch mode programs that return their results in a matter of minutes or hours with a data volume roughly equal to that of a several A4 pages. The large number of nodes rules out the danger that network bottlenecks could cause a compute node to be cut off from its job data for any length of time.
Applications that have such properties are called coarse grained.

If applications are not coarse grained or if the cluster is to be operated to any extent in parallel mode at least a high speed connection is required.

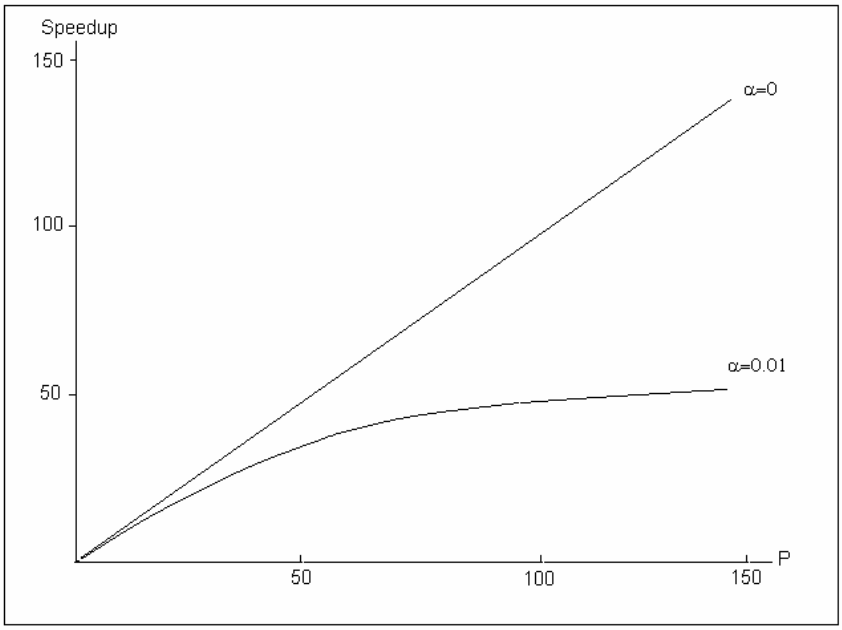

Fig. 5: Optimal speed up compared to real life speed up

In the case, the cluster works in parallel mode, the user wants have the result in a shorter time and is therefore interested in the speed up. But there are fundamental limitations the user has to fight with. It can be shown theoretically, that the speed up depends mainly on the amount of sequential code within the application. Fig. 5 shows the speed up for an application, which works 99% of the time in parallel and only 1% in sequential mode. The result is a basic limitation of the speed up to a factor slightly below 50, irrespective of the number of compute nodes.

In practice, the result is even worse. Because of increasing data traffic between the compute nodes, adding another node can be a disadvantage and will even slow down the application. As fig. 6 points out as an example, the runtime of the sample application didn't become shorter, but goes through a minimum and then increases when more nodes are influenced.
Therefore it is a good advice for the user to check in practice the speed up in reference to the number of nodes.

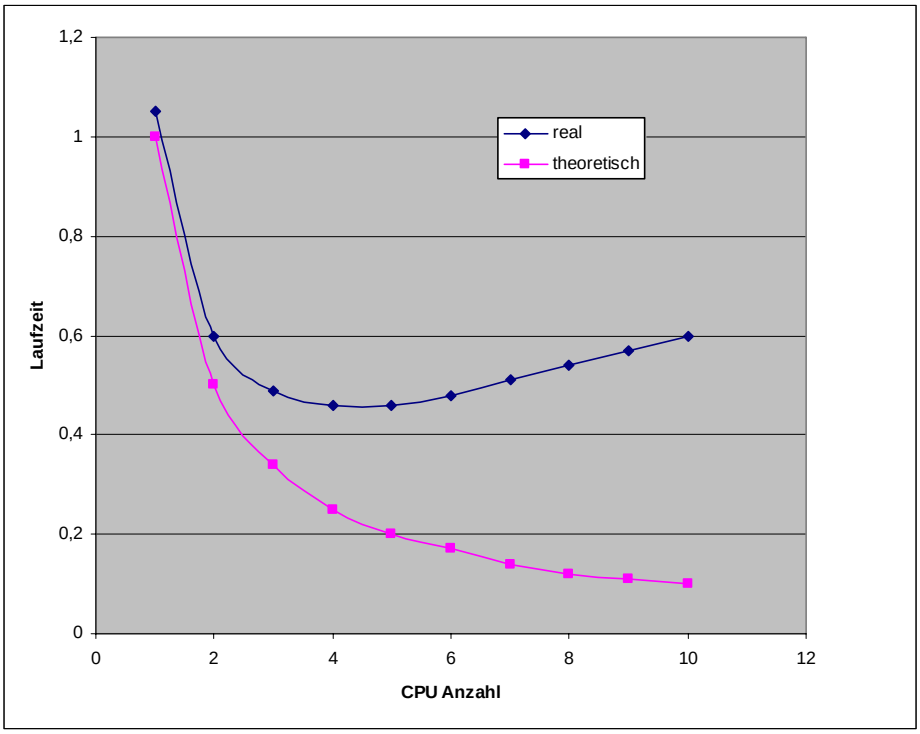

Fig 6: Real life speed up in comparison to theoretical speed up

Hardware is not the only factor influencing data transmission speed; the type of network protocol employed also has a significant influence on throughput rates. While the Ethernet standard TCP/IP is widely used and standardised to a tee, its normal latency is 30 µs for Gigabit Ethernet. The use of other protocols requires programs to be recompiled to comply with the correlating libraries.

The first high speed interconnect, the user should take into account is InfiniBand. At present 4x SDR (single data rate) InfiniBand offers a maximum data transfer rate of 10 Gbit/s. InfiniBand 4x DDR is actually coming up, offering 20 Gbit/s, and 12x SDR InfiniBand is providing up to 30 Gbit/s for inter switch connections. The actual latency that can be achieved with a Mellanox InfiniBand card is around 3 µs. The value will decrease with the next generation of chipsets, which are expected for end of summer.
InfiniBand is proven, several millions of ports have been sold through the last years, and as it's an open IEEE standard. For that reason we have competition on the market and also falling prices. The total price per compute node varies between 500 and 1000 US$.

Historical competitors to InfiniBand are Myrinet, SCI and Quadrics. All of them are based on a proprietary technology, it's a closed standard and the corresponding hardware is only sold by the respective manufacturer.

While a Myrinet (2 Gbit/s) connection fails to provide any real bandwidth improvement over Gigabit Ethernet, its latency is also several factors lower when the proprietary GM (Grand Message) protocol is used. This protocol is fine-tuned to Myrinet hardware and the effective data throughput is much higher than Gigabit Ethernet for small block sizes. The costs of a Myrinet connection are roughly around 1000 US$ per node.

Network performance comparable to Myrinet can be achieved with SCI (Scalable Coherent Interface), for example, using the Dolphin SCI adapter. The Quadrics QsNet network adapter is another high-speed technology; it has actually the lowest small-packet latency on the market. Because of a total price of up to 2000 US$ per node, that interconnect is rather uncommon.

To decide, which interconnect is right the real life speed up should be regarded. The benchmark should be run on a platform equipped with the desired processor and interconnect with a minimum of 4 nodes.

6 Further aspects

A lot of other aspects have to be considered, before buying an HPC cluster.

- How to optimise the front end node, to avoid a breakdown of the whole cluster?
- Does is make sense, to go for an HA cluster instead of a single front end node?
- Are there other aspects, regarding the compute nodes, we have to look at, like required service level or case type? What about a blade server solution?
- Is the total heat, produced by the cluster system, a critical aspect?
- According the installation: Which distribution will fit my needs? Which queuing system should be chosen? What is the actual choice of libraries and MPIs?
- How can the administrator control the nodes - via KVM, terminal server or something else?
- Is a maintenance contract to automatically update the cluster an aspect to look at?

The total discussion will take a minimum of two hours and doesn't fit in the actual timeframe. For those who are interested in, transtec offers a complete talk totally cost for free on site to help finding the right answers.

7 References

[1] Sterling, Thomas and Becker, Donald: „How to build a Beowulf", May 1999, 261 pages

