A multiobjective, constraint, global optimisation problem under uncertainties (or just: Resource allocation)

Tor Sørevik[1]
The Resource Allocation Committee(RAC)

[1]Dept. of Mathematics,
University of Bergen,
NORWAY

Tromsø
June 4 2008

Should HPC resources for science be taken for granted?

- The Research Council seems to think so
  Large NRC project, critically dependent on HPC resources, are awarded without being evaluated against availability of HPC-resources.
- Some scientists do too
  "It is fundamental infrastructure like a desktop, a phone, etc,..."
- Reality: The demand is higher than the supply, and the demand is increasing.

We need fair sharing of scarce resources!

What do we allocate?

1. Computing resources (=cycles)
2. Advanced user support
3. *Archive Storage*
4. *Grid resources*

What's the objectives?

The mandate says:

- Maximise scientific value
- Priority to those with most computational needs
- Priority for new user groups

In addition:

- Simplicity
- Flexibility

Potential conflicts

- Quality of research vs. New user groups.
  In the short term the expected payback is best for group familiar with HPC.
- Simplicity vs sufficient info for intelligent decisions
  Solution: Large/small applications
- Flexibility vs need for long term planning of research.
- ....

Local vs. Global optimisation

- When writing her application the user only knows about **her** needs and the available resources.
- RAC sees all applications and how they "map" onto the complete NOTUR-system.
- To reach "global optimum" we need to move applications from the preferred system to systems with available capacity.

What is the global optimum?

High utilisation vs. the right utilisation.

- We have to give way for massively parallel jobs
- We should use idle cycles

This may be regulated through:

- Setting the priority in the queueing system
- More active use of priority time vs. unprioritised time. (UiO's libero-que)
- Better reporting so we can analyse what really going on.

More on uncertainties

The users estimates are usually wrong! They overestimate.

- We overbook (with a factor 1.4)
- Users who notoriously ask for X-times what they usually consume are automatically slashed with a factor X.

Local quotas:

- Will all of the local quota be used?
- What happens when a local user gets her NOTUR-application reduced?

Our allocation algorithm

1. Reduce the *X-timers*
2. Move applicants to system with available capacity
3. Axing based on quality and needs. (Painfully and difficult)

2. Advanced user support

Optimising important codes by:

- Introducing new and faster algorithms
- Using optimised libraries for computational kernels
- Improving parallelism
- Improve IO, reduce memory use, ...

NB: The user must be involved in such projects.

Now may be the time for advanced user support!

- With the last upgrade the NOTUR-system got its performance boost through increased parallelism.
- To benefit from this one need scalable, parallel code.
- Many important codes do still have sequential bottlenecks.

3. Norstore: Archive storage

Hardware in place, pilot project about to start.
A completely different allocation process is needed:

- Long term. When first approved, it is hard to remove.
- Overbooking hazardous. Capacity must be available.
- Must be designed for growth.

4. Grid resources

- The hardware is distributed. The users don't want to know and be bother by this.
- Some grid software should make the distributed structure transparent.
- Caveat: In HPC speed is the ultimate goal. Applications are tuned for a specific hardware. Thus just distribute jobs to were idle cycles are available may not be possible.
- We need to start with user communities with well defined needs.
- It needs to be collaborative projects.

Thank you for your attention!