Resource Management II

- Supercomputer community likes simple charging models, e.g.:
  - charge per CPU billing unit or by wall clock time (or both)
  - charge per CPU
  - charge for memory
  - charge per megabyte of online storage
  - premium rates for special handling
  - premium rates for higher job queue priority within a job class
  - charge for (pre-allocated and reserved) network bandwidth usage
  - special application charges
  - local consultant, programmer or administrator charges
  - transportable media charges
- Autonomous Suppliers:
  - control resources, quote prices, implement accounting system
- Independent Consumers:
  - select suppliers, consider value, implement resource requests

Basic Economics

- **cost** and **price** are not the same
  - as a producer, the cost is a measure of what it costs me to produce and sell something, which will be the sum of the prices I pay for components, plus my additional overheads etc
  - the price is what I charge a consumer for that good
- marginal cost is the cost of producing one more of the good
- true cost includes the amortised cost of the equipment, including the cost of capital
- cost will vary depending on the volume of goods I produce
- if demand rises, I may produce and sell more goods
  - cost may go down if I spread my equipment costs more thinly
  - cost may go up if I increase demand for a scarce input
- Lots of projects (e.g. lesc.ic.ac.uk/markets) aim to produce a basic market economy for the Grid, enabling resellers, futures etc

How does this apply to computer resources?

Externalities

- Simplistic economics does not capture the total costs
- **Externalities** are the costs other people have to pay because of my resource usage, but which I don't pay directly
  - e.g. cost of treating elderly miners was borne by the NHS, not the people who bought coal
- Health externalities are easy to find, e.g. impact of passive smoking
- Global warming is another major source of examples
  - Some countries will vanish (e.g. Maldives) because of emission of greenhouse gasses by developed and developing nations
- Many externalities arise in the use of Computing Resources
  - cost of sending one ethernet packet is essentially nil
  - all of the infrastructure is paid for
  - additional electricity consumption is undetectable
  - but if it collides with another packet, both suffer

Congestion Charging

- Idea of congestion charging is to allow for the externalities
- Don't normally pay, because cost is negligible
- If you contribute to congestion, then you pay
  - congestion or shadow prices
- Simple example:
  - Consider a network that can cope with 500 emails per second
  - if 499 emails are sent, they are all free
  - if 501 are sent, a cost is incurred for ALL of them
- Goal is to set the prices to ensure a social optimum
- Basically, try to raise just enough money to expand the resources to cope with the excess demand, or reduce demand to eliminate congestion
- Unfortunately, there are latencies in this :-(

Shadow Prices — outline theory I

- Consider a single finite resource
- Consumer i
  - attempts to maximise **utility** $u_i(x_i)$
    - this is the benefit gained from consuming x_i of resource
  - is charged some cost $C(\cdot)$ for the resource usage
  - maximise $U_i(x_i) = u_i(x_i) - C(\cdot)$
- Social Planner
  - wishes to achieve a *socially optimal* resource allocation
  - maximise sum of consumers' utilities, minus cost of overall system load (externalities)
    - let $y = \sum_i x_i$
    - maximise $\sum_i u_i(x_i) - C(y)$
  - solving this, for convex $C(y)$, gives:
    - $u_i'(x_i) = p(y)$ with $p(y) = C'(y)$
      - $p(y)$ is the shadow price of the load

Shadow Prices — outline theory II

- marginal benefit for consumer i matches the marginal cost to all consumers
- this is mathematically tractable, but:
  - requires knowledge of $u_i(x_i)$
  - which is usually not explicitly known
  - especially not to an outside agent
    - lying about $u_i(x_i)$ could give benefits!
- Decomposition:
  - consumer i is charged at a fixed rate t_i, in proportion to the amount of resource it receives
  - consumer maximises:
    - $U_i(x_i) = u_i(x_i) - t_i x_i$
  - solving this, for concave $u_i(x_i)$, gives:
    - $u_i'(x_i) = t_i$
  - if $t_i = p(y)$ this is the same as the social optimum
- When prices are correct: **User optimum** $\equiv$ **Social optimum**

Problems with shadow pricing

- Assumes consumers are rational and competent
- Assumes perfect information about prices is available
- Assumes utility functions are concave/convex
- Assumes elasticity
  - too little of a resource may be pointless
  - may have step-shaped utility function
- Favours richer consumers
- Allows rich malicious user to wreak havoc
  - submit large jobs to busy machines, ignoring idle ones
  - essentially a form of denial-of-service attack
  - trying to price other users out of the system
  - since users are not omniscient, could succeed
- Only find out about congestion afterwards

pricing Multiple Resources

- Multiple independent resources can be shadow priced separately
  - but how does consumer divide funds?
- If resources are substitutable it's more complex
  - Continuous substitution is OK
  - Discrete switching of pieces of load is harder to cope with
    - in general case, think about bin-packing problems
- Economic theoreticians can produce more complex models
- Systems builders want simple mechanisms
  e.g. for an indivisible resource:
  - *Vickrey Auction:*
    - highest bidder wins
    - price paid is second highest bid

Co-Reservation Problem

- Simple case:
  - CPU time (on a cluster)
  - Memory and Disk space
  - Network Bandwidth
- More complex case:
  - resources at multiple sites
  - network links between those sites
- Real Problem:
  - Choose several from a range of providers

Abstractly

- Consider a weighted graph
- Each edge represents a network connection
  - complete with a cost/performance profile
- Each node represents a potential provider
  - complete with a cost/performance profile
- Both profiles are multi-dimensional
- Choose a subgraph that meets your needs
- NB: the cost functions keep changing
- Load balancing/management should be a major problem in Grids

Simplistic Co-Reservation

- Actually entirely tractable
  - each community has its own view and needs
  - will determine reservation mechanisms that support co-reservation
  - low-level issues resolved by over-resourcing
  - coordination problems resolved by simplifying interactions
- the real problem is still lurking out there

- How do people orgnise holidays?
  - flight, hotel, restaurants, car hire

Research and (vs?) Practice

- Basic scheduling for the Grid is the standard issues, confounded by security and trust complications
- Basic accounting for the Grid is even more basic and dull

- Prospects of interesting approaches to both, but. . .
- only if the Grids get more interesting themselves

- Danger is that the communities just want something that will work
- Companies exist to create and service demand
- The real underlying problems are interestingly hard, but can be quite abstract and workarounds usually suffice

Example Academically-produced Grid Tool

- University of Wisconsin's Condor
  - http://www.cs.wisc.edu/condor/
- "A contributor to Condor's success is its ClassAd mechanism. Jobs want to find machines upon which they can execute. A job will require a specific platform on which to execute. Machines have specific resources available, such as the platform and the amount of available memory. A separate ClassAd is produced for each job and machine, listing all attributes. Condor acts as a matchmaker between the jobs and the machines by pairing the ClassAd of a job with the ClassAd of a machine."
- What's the comparison with a Beowulf cluster?
- How do these compare with the VMS queue support available for VaxClusters over 20 years ago?

Example Commercial Grid Tool

- Sun N1 Grid Engine 6
  - http://www.sun.com/software/gridware/
- "Computing tasks or jobs are distributed across the grid in accordance with resource requirements for the job, user requests for the job, and administrative/managerial policies. Usage accounting data is stored and made available so that it is possible to determine what resources were used in the execution of a job, and for whom the job was run."
- Sounds great, but it's for a single institution
- Scales superbly and is robustly engineered

Thought Experiment

- What is the right way to present Grid resource management and accounting to a user community?
- What issues do they really care about?
- What issues are annoying or irrelevant?
- Distinguish between sysadmins and true users
- Distinguish between grant-acquirers and tool users