Manchester Computing
Supercomputing, Visualization & e-Science

Service Level Agreement Based Scheduling

the Inter-disciplinary Scheduling Network

asap research
automated scheduling optimisation & planning

http://www.sve.man.ac.uk/Research/AtoZ/SLABS

Jon MacLaren

mailto:jon.maclaren@man.ac.uk

Open Issues in Scheduling Workshop
NeSC, Edinburgh. 22nd October 2003

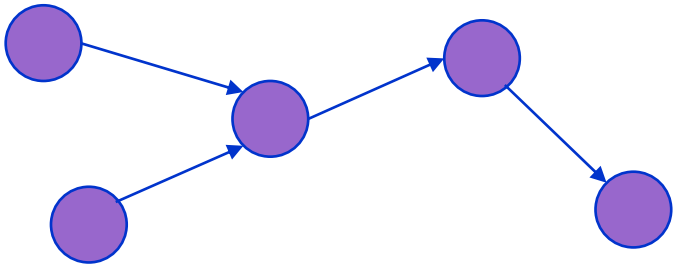
THE UNIVERSITY of MANCHESTER

Who's Involved in the Project

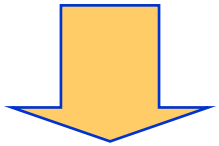
- Rizos Sakellariou
- Jon MacLaren
- Terry Hewitt
- Edmund Burke
- Jon Garibaldi
- Djamila Ouelhadj
- RAs to be determined

- Project commences January 2004
- Runs for three years

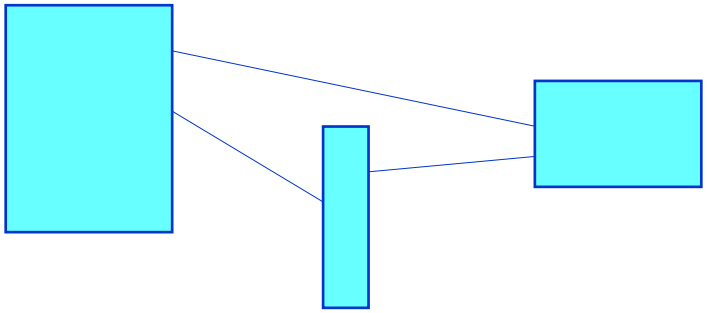
Scheduling Workflows



Want to take a workflow...



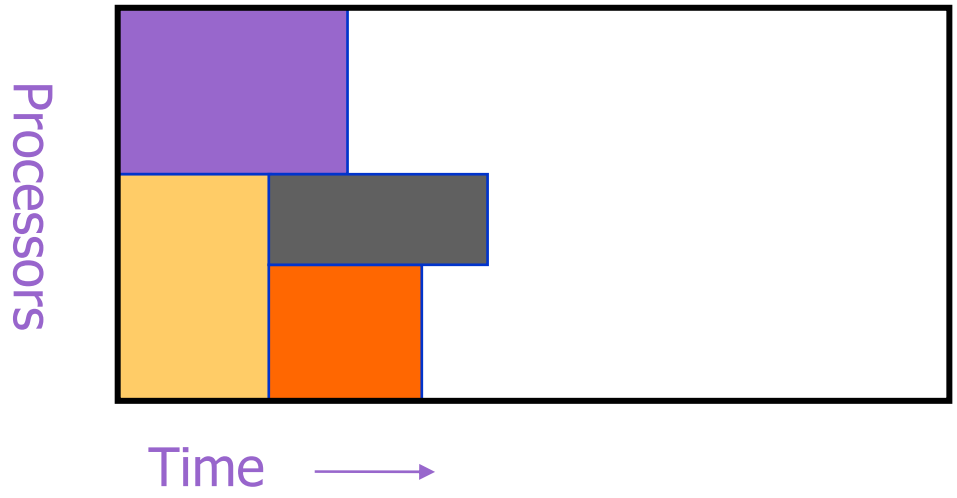
...and schedule it...



...onto the Grid – a Heterogenous Computing Environment

Compute Resource Scheduling

- Each computing resource has its own local scheduler, which has control over that resource
- Brokers/superschedulers DO NOT have control over the resources – they have a similar role to that of a user
- Grid is heterogenous
- Individual compute resources are often supercomputers, with many processors (homogenous)
- Local schedulers are “batch” schedulers



Goal

We want to be able to schedule complex workflows onto compute resources with their own schedulers

Have to honour dependences between parts of the workflow

Want the user to have some assurances about time to completion

Can't just wait until one component comes back to start thinking about scheduling the next component – want to hide batch queue latencies somehow

Need to control when the components execute

How can we do this now?

Only alternative is to use Advance Reservation (AR)

Often, AR is a sledgehammer being used to crack a nut

Often we're only interested in the bounds (soonest start, latest end time)

Also, the client may not want to set the time precisely

You force the client (user or broker) to artificially narrow the bounds in an arbitrary way

Bad for the Resource Owner

The other problem is that advance reservation doesn't fit very well into the batch processing model

It causes expensive gaps in the schedule that can't be effectively plugged.

- Checkpointing (especially by the OS) takes (a lot of) time.
- Suspend/Resume quicker, but impacts available resources/performance of the AR job (cost of swapping out)

Utilisation decreases rapidly (worse than linear) as the percentage of AR jobs in the job mix increases

Affects the income of the site, or results in additional (or even unpredictable) costs for anybody doing Grid things...(bad for the client again)

What's causing the problems?

Current schedulers offer two basic levels of service:

Run this when it reaches the head of the queue

?

Run this at a precise time (advance reservation)

What's the solution?

Clients (users, brokers, superschedulers) often know or could easily define constraints for soonest start, latest end time – which could form part of an SLA

Additional constraints might be based on performance when running (e.g. time per iteration) or cost.

But all of this information is being ignored, i.e. it is available, but not captured!!!

So the solution is to capture and use this information!!!

What will the SLAs Look Like

So whenever a job is submitted to our scheduler, there will be a negotiation, where constraints are agreed and an SLA is formed.

Natural protocol to achieve this is WS-Agreement

This uses WS-Policy to model the terms of the agreement

Some features of WSLA may be incorporated in the future

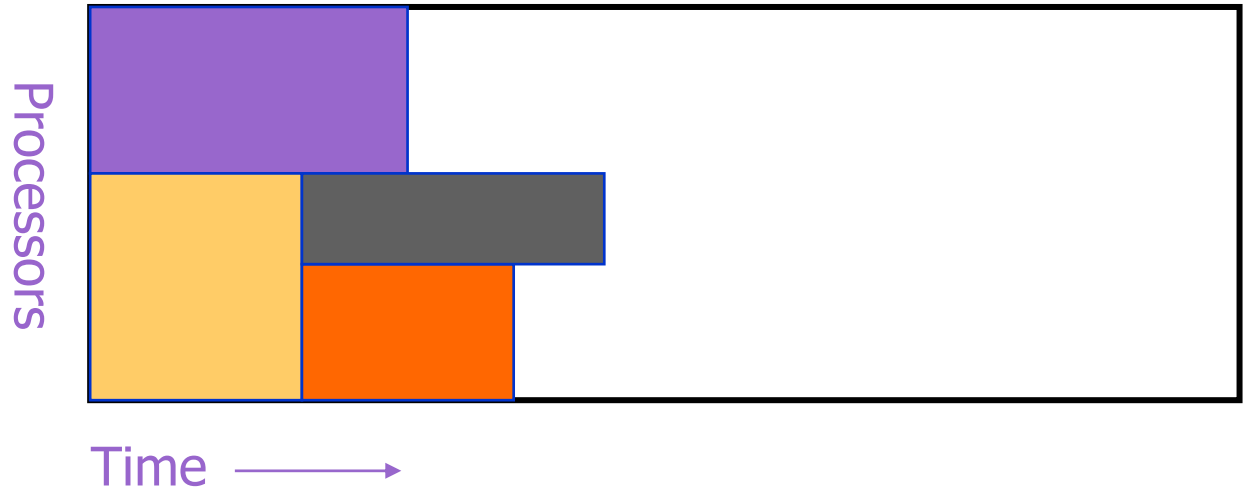
Need a set of WS-Policy terms representing the SLAs

Will include a compensation model (for violations)

Representation of SLAs is a key early goal of the project, and could be fed back into GRAAP-WG as an GGF Experimental Document

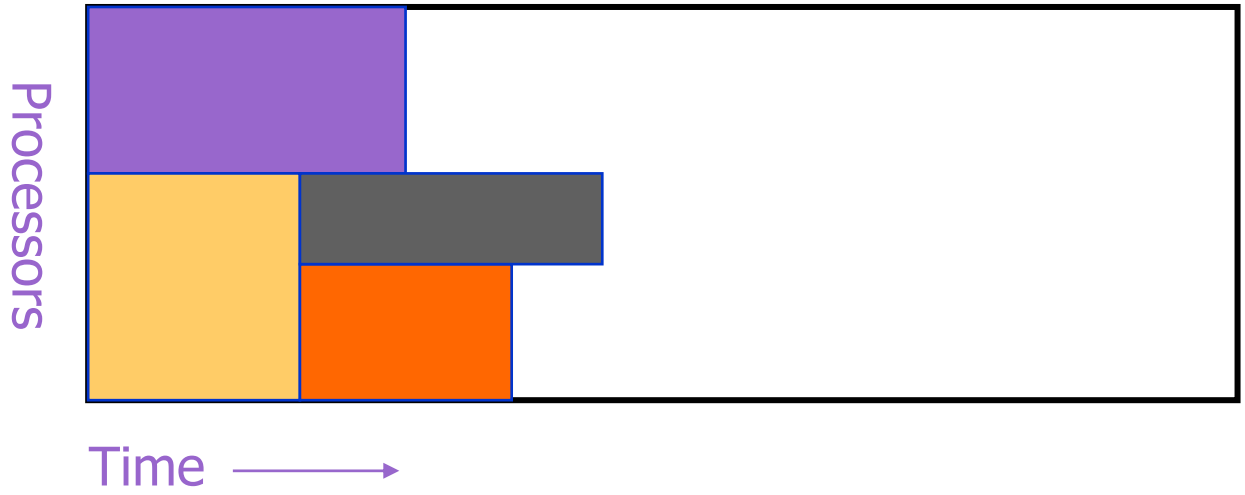
Renegotiation

With existing systems, jobs are like this:



Renegotiation

With existing systems, jobs are like this:



We want to **“break the box!”**

Change resources while the job runs!

The RealityGrid project

Mission: *"Using Grid technology to closely couple high performance computing, high throughput experiment and visualization, RealityGrid will move the bottleneck out of the hardware and back into the human mind."*

http://www.realitygrid.org/

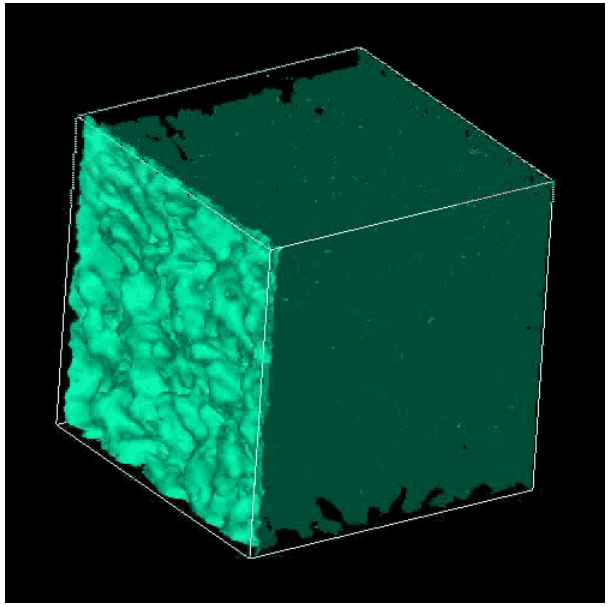
Aims:

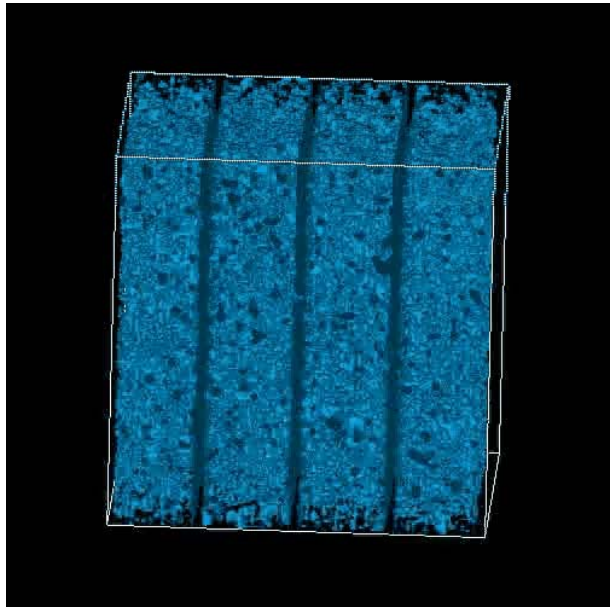
- to predict the realistic behavior of matter using diverse simulation methods (Lattice Boltzmann, Molecular Dynamics, Monte Carlo, …) spanning many time and length scales
- to discover new materials through integrated experiments.

Lattice gas methods

3D Lattice Gas method: Binary and ternary immiscible phase separation

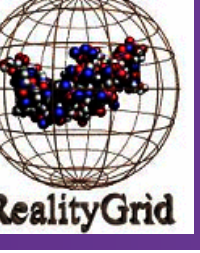
Invasion of a porous medium with residing fluid. Only oil and water [1]





Ternary system: two immiscible fluids plus surfactant. Only oil density shown. Shear Flow, lattice size=64^3, shear rate=0.25, reduced density=0.18 [2]

[1] Love P J, Maillet J-B, Coveney PV, Phys Rev E 64 61302 (2001);
[2] Love P J and Coveney P V, Phil Trans R Soc London A360, 357(2002)



What we want to do

Working to provide collaborative steering library for legacy codes

Includes application-level checkpointing for job migration

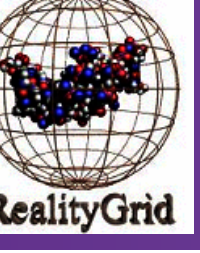
- Portable to different architectures
- Independent of processor counts

But also want to change resources without migration

Want to be able to extend experiment in time (at short notice)

- Either because the scientist finds something worthy of a nobel prize around the corner
- Or because the application is still writing out its checkpoint file, and knows it's running out of time

Want to be able to grab more resources to speed up either the simulation or the visualization



Why Renegotiation Helps

- Currently, we have to checkpoint, make a new reservation, then restart there
- Want to avoid checkpoint/restart when not relocating machine
- With very large simulations, state can approach 1TB!
- Also, the interruption is frustrating
- Doing this dynamically would avoid output and input of the state
- For distributed shared memory machines, redistribution of problem in memory would be required (non-trivial)
- For shared memory machines, only need to change the number of threads! (some support in OpenMP for this)

A Novel Approach to Scheduling?...

Each job has an associated SLA

The schedule is calculated based upon these SLAs

There is no queue, and jobs don't have a "priority"

Also, the current set of SLAs will determine what new SLAs the scheduler can commit to – saying no is an option

Introducing renegotiation makes the problem far more complex and dynamic...

Strategy and Policy

But of course, some jobs will fall over straight away, or be withdrawn, so some overbooking might come in handy

And of course, if someone came along with a huge amount of cash, we might want to break a couple of smaller agreements...

There are short term goals, e.g. maximise income

And long term goals, e.g. get a reputation for reliability

Also want to retain flexibility for fast (lucrative) increases in demand

But we will still want to enforce some policies, e.g.

- Favouring local users with preferential rates
- Adjusting rates to meet group targets

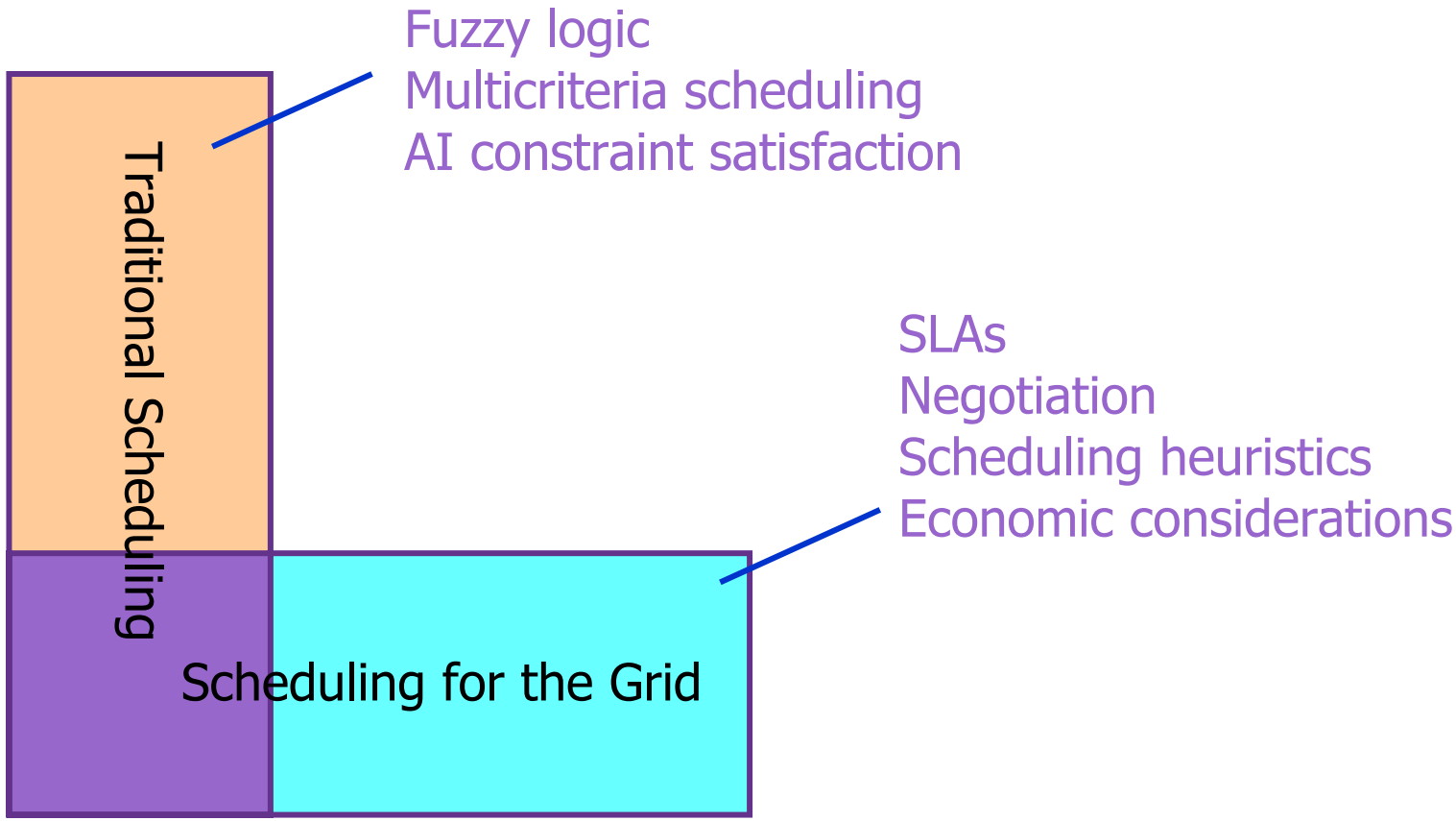
Thoughts on Scheduling Heuristics

There is a considerable body of literature on how to schedule in distributed computing environments

Most of this work doesn't feel suitable for our problem... Why not?

- Often the focus is on a central control point for many resources (not very Grid)
- Often trying to optimise for resource usage and throughput – but this is not appropriate for individual job requirements
- Jobs assumed to have static requirements

Our Research Challenges "L"



...Or a Traditional Approach to Scheduling?

The problem doesn't look like a batch scheduling problem any more.

It's shifted, and looks more like a "traditional scheduling" optimisation problem, e.g. optimising workflow in a factory.

We need the traditional scheduling community, and their techniques.

We've joined forces with the ASAP group from Nottingham, one of the best "traditional scheduling" groups in the world.

They bring their fuzzy logic and heuristic techniques to the table; we bring our Grid scheduling expertise.

Manchester Computing
Supercomputing, Visualization & e-Science

Flexible SLA Based Scheduling

Jon Garibaldi

Automated Scheduling, OtimisAtion and Planning (ASAP) Research Group

University of Nottingham

THE UNIVERSITY of MANCHESTER

Flexible SLA Based Scheduling

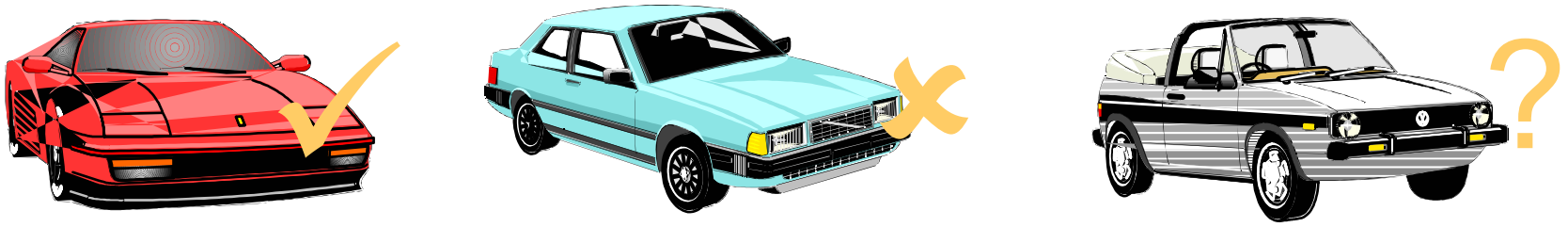
- Uncertain
  - fuzzy
  - multi-criteria
- Dynamic
  - scheduling and rescheduling
- Distributed
  - multi-agents

Classical/Crisp Logic

- Origins in Ancient Greece
  - Aristotle
  - Plato
- Two truth values: *true*, *false*
- Connectives: *not*, *and*, *or*
- Aristotle saw weaknesses
  - future events: ?
- Zeno's paradoxes

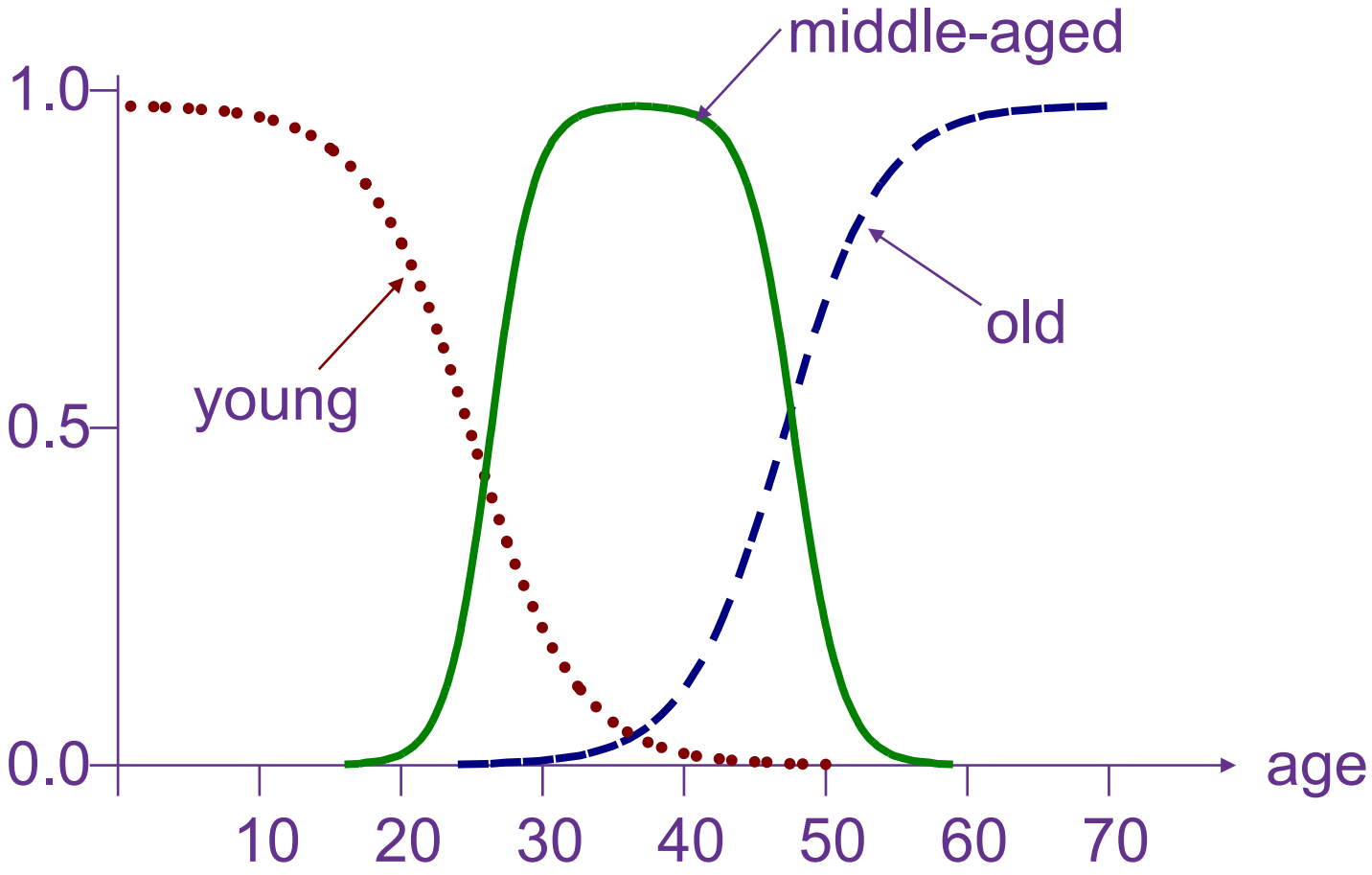
Fuzzy Logic

- Consider a real life question
  - would you describe the following as a ‘fast car’
  - i.e. is it a ‘fast car’: *true* or *false*?



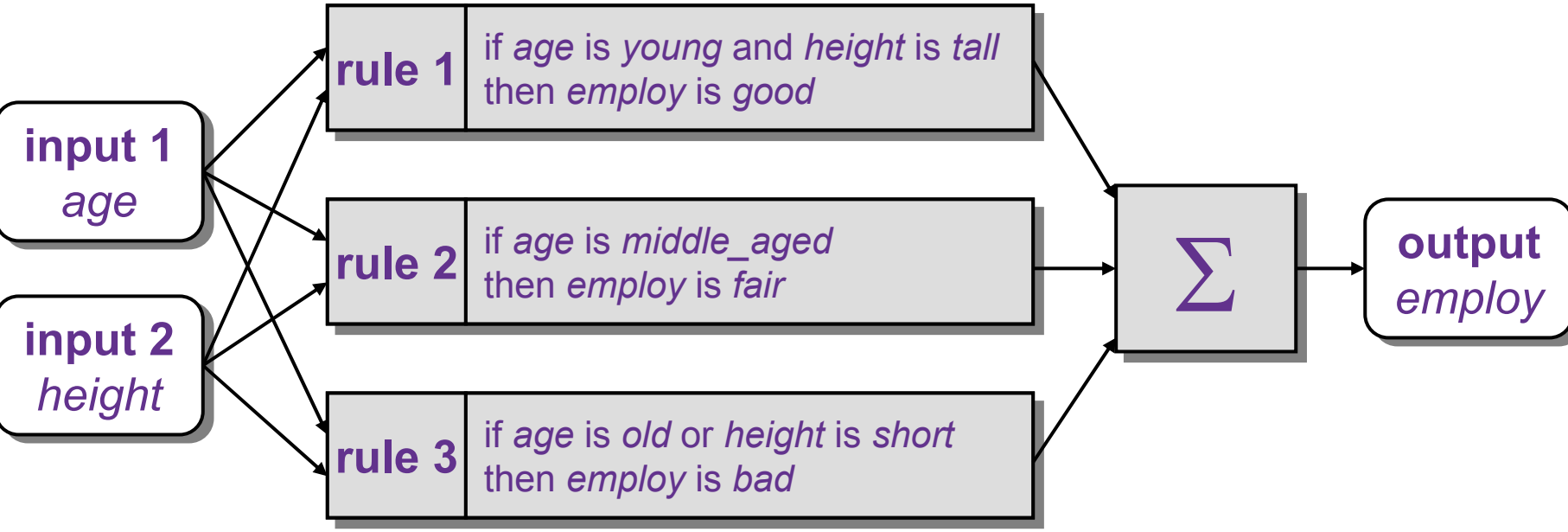
- In fuzzy logic, a real value between 0 (zero) and 1 (one) is used to represent the degree of truth
  - 0.0 (totally) false
  - 1.0 (totally) true
  - 0.5 half true / half false

Fuzzy Membership



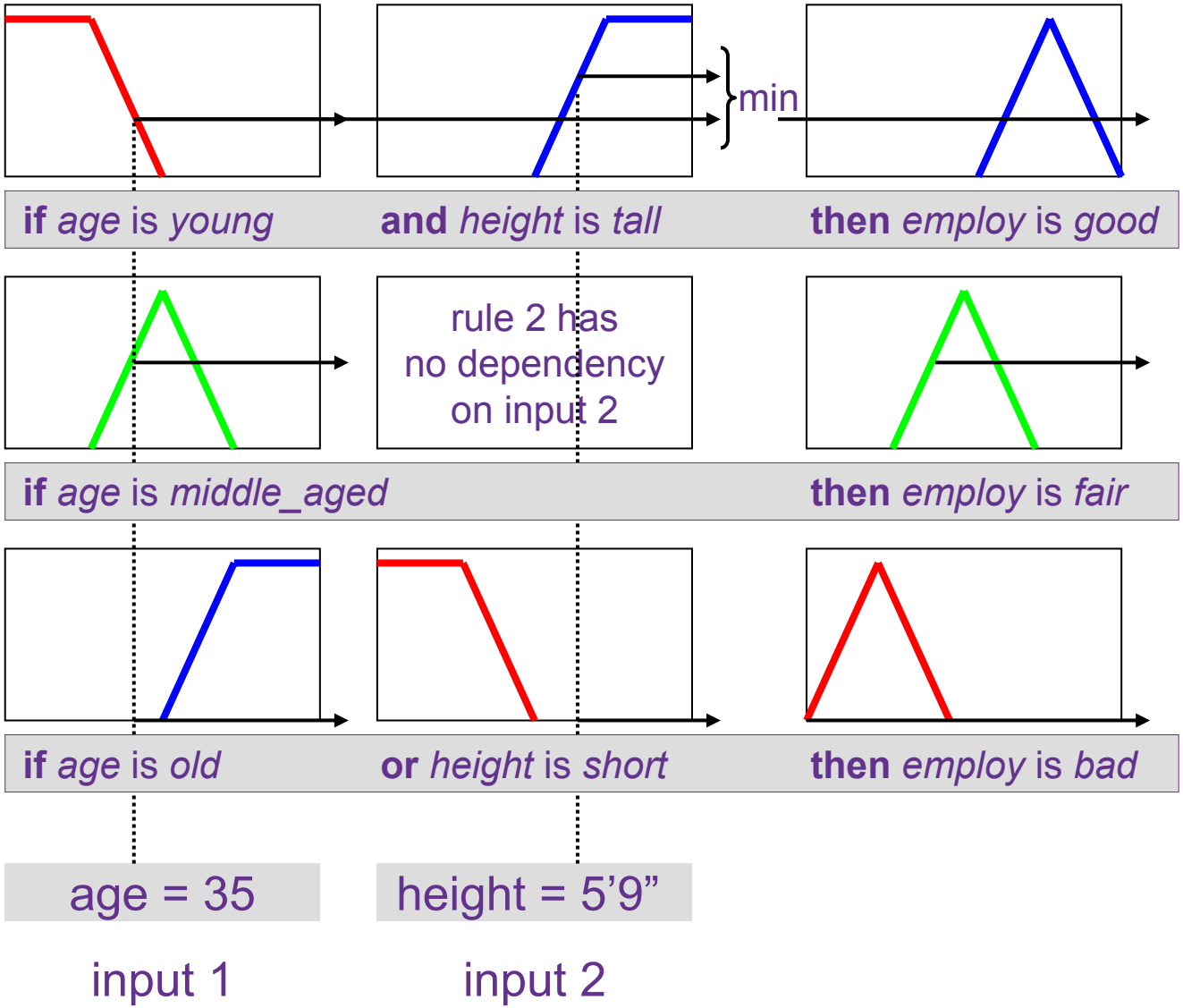
The numeric age is called the *base variable* of the linguistic term

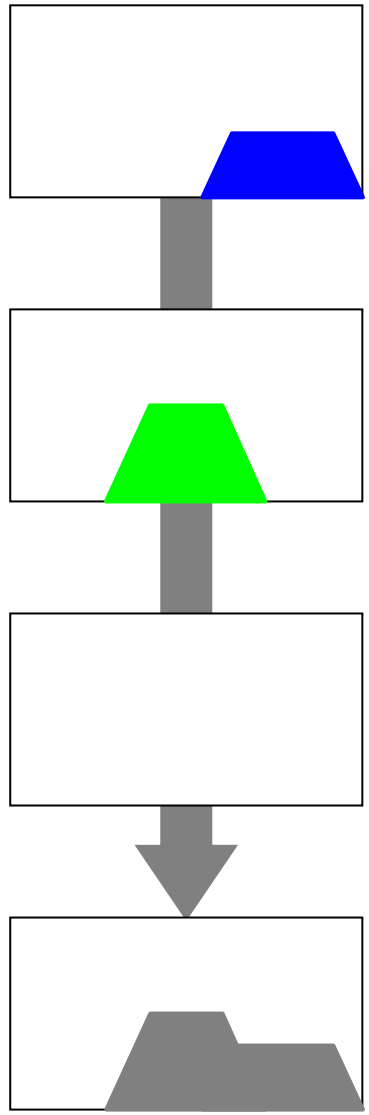
Inference Outline



- Fuzzify inputs
- Combine inputs
- Perform implication
- Aggregate output
- (Defuzzify)

Mamdani Inference



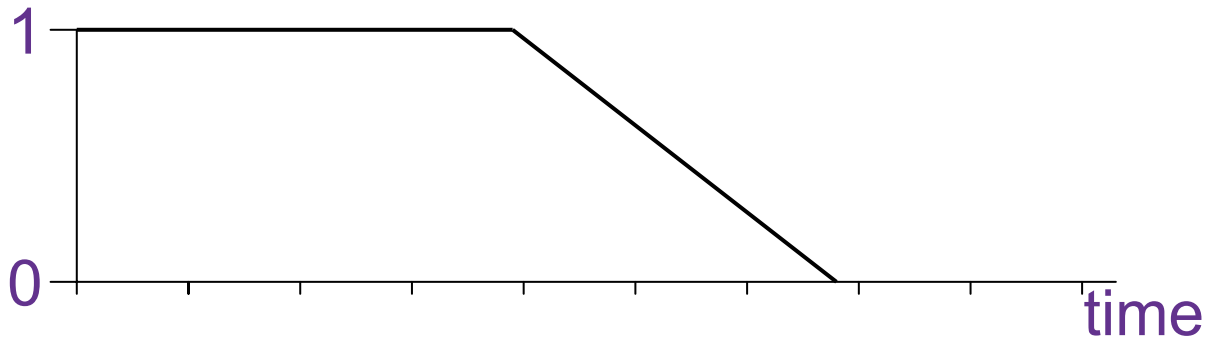


Fuzzy v. Probability

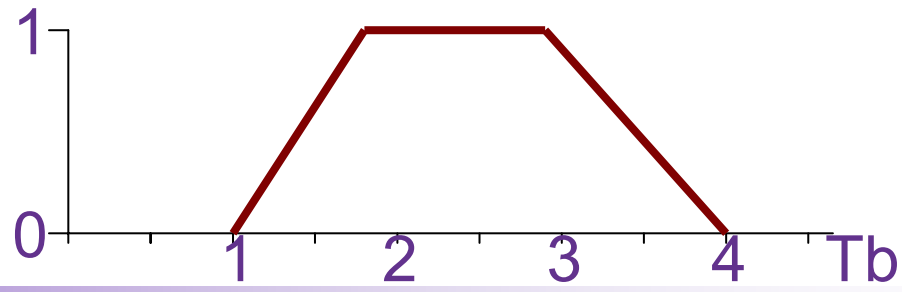
- Scenario
  - you are given two bottles of liquid, *A* and *B*
  - bottle *A* has 0.9 *probability* of containing drinkable liquid
  - bottle *B* has 0.9 *fuzzy membership* of set of drinkable liquids
- Which do you choose to drink?
- 'Answer'
  - bottle A has a 1 in 10 chance that it contains non-drinkable liquid
    - could be anything, e.g. poison?
  - bottle B is 9/10^{ths} along the scale of drinkable liquids
    - so must be reasonably drinkable

Fuzzy Scheduling

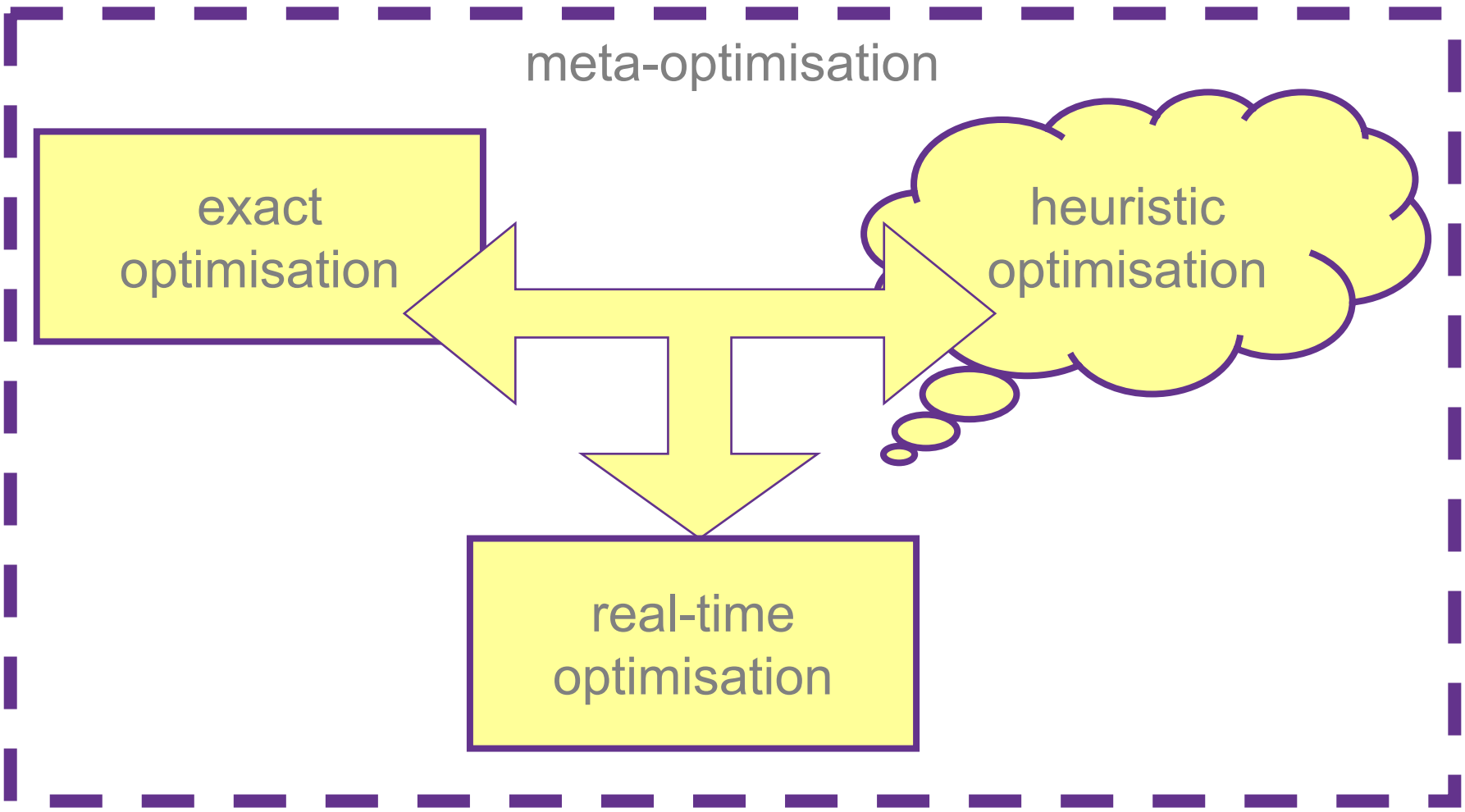
- Fuzzy job durations



- Fuzzy resources / constraints
  - crisp constraint Mem = 2 Tb
  - interval constraint Mem = [1 4] Tb
  - fuzzy constraint



Schedule Optimisation



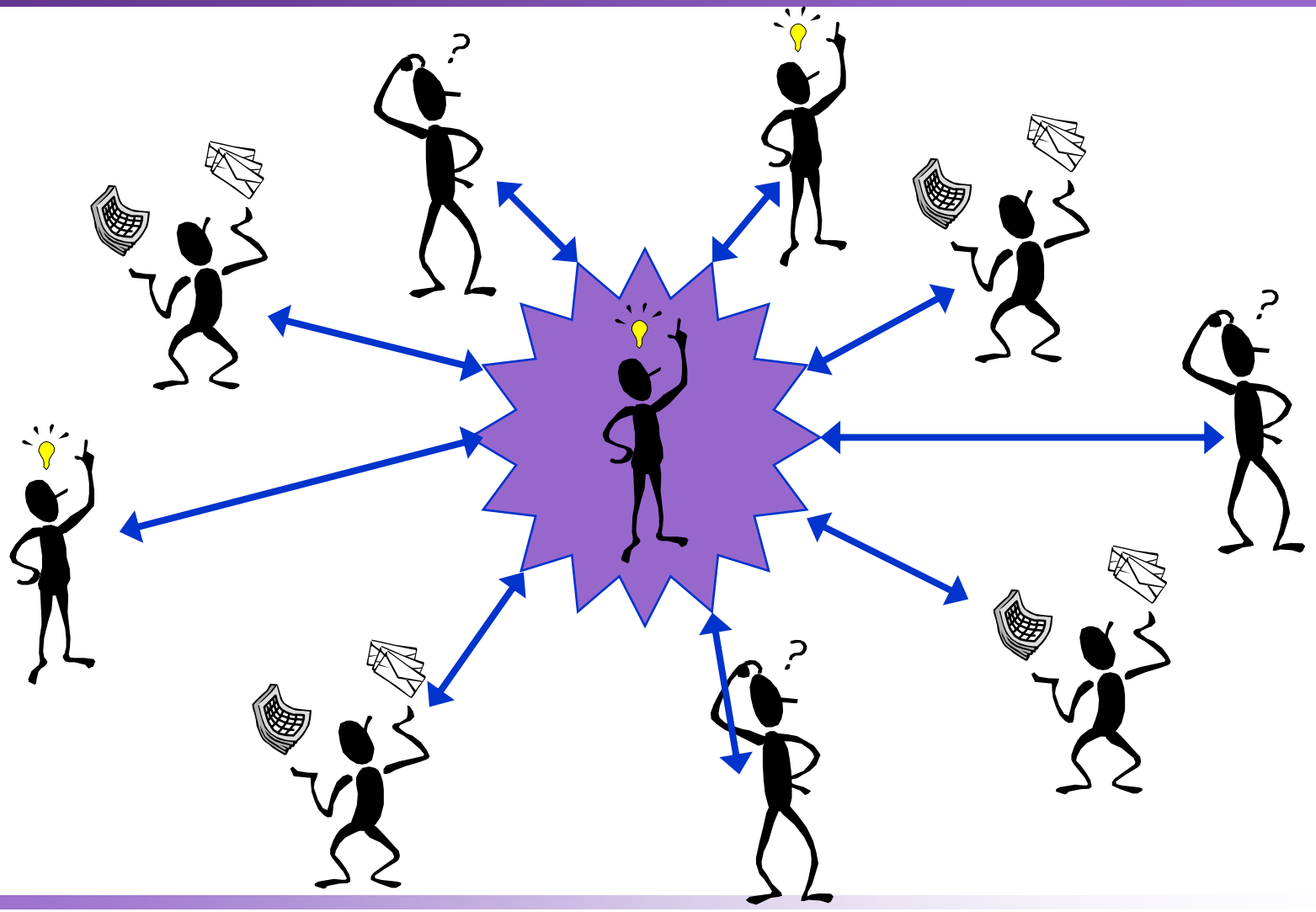
Multi-criteria Optimisation

- We need to optimise something
  - the objective function
  - minimise lateness?
  - minimise cost (to the user)?
  - maximise revenue (to the provider)?
- Of course, these can be combined (weighted sum) into a single objective function
- Or, for example
  - the user is provided with two criteria ‘estimate of lateness’ and ‘cost’
  - increase cost -> decrease lateness
- Optimise towards the pareto optimal surface
  - how to achieve good coverage? research issue

Dynamic Rescheduling

- Real-time / real-world events
  - new jobs submitted / jobs deleted
  - new resources come online
  - resource breakdown / failure
  - job crash (surely not!?)
- The optimisation algorithm needs to continually monitor the schedule and perform dynamic rescheduling
- Obviously if a resource is unallocated and jobs are waiting
  - jobs need to be allocated to resources quickly
- But
  - there may be advantage in spending time optimising a reschedule
  - research issue

Multi-agent Based Optimisation



‘Traditional’ State of the Art

- Fuzzy Multicriteria Approaches to Scheduling and Rescheduling Problems in Uncertain Environments
  - ASAP, University of Nottingham
  - CTAC, Coventry University
- Hybrid Metaheuristic Approaches for Air Traffic Control Scheduling
  - University of Nottingham, National Air Traffic Services
- Using Real Time Information for Effective Dynamic Scheduling
  - University of Nottingham, University of Bradford
- Case Based Reasoning in Personnel Rostering
  - University of Nottingham, Queen’s Medical Centre

Fostering Collaboration between Grid and Traditional Scheduling Communities