SWINBURNE UNIVERSITY OF TECHNOLOGY

Improving energy efficiency in cluster computing

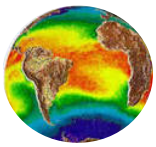
Tuan V. Dinh

Centre for Advanced Internet Architectures, Faculty of ICT

Swinburne University, Australia

Motivations

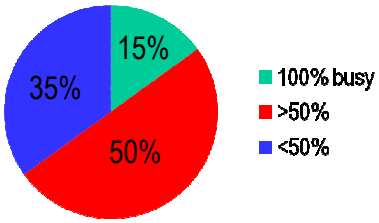
Data Centres and Servers
→ 1.5% global electricity
(all ICT → 3.0%)[1]



(1) S Ruth, "Green IT More Than Three Percent Solution?", Internet Computing, 2009

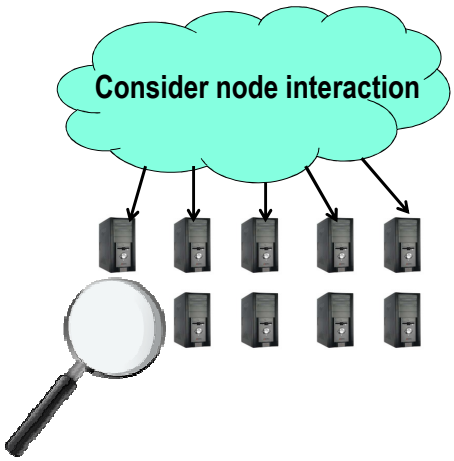
High-Performance Computing

- Not fully utilised most of the time
- Nodes on 24/7
  - Swinburne's Supercomputer (160 node cluster):

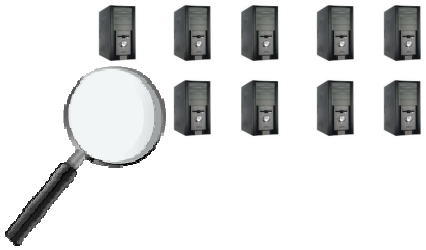


- Mechanisms to save power during idle time are either not present or inefficient

Approaches

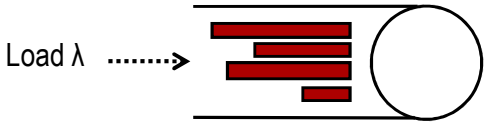


Approaches



SWINBURNE UNIVERSITY OF TECHNOLOGY

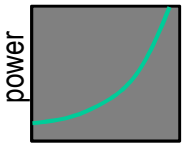
Speed scaling of a single server



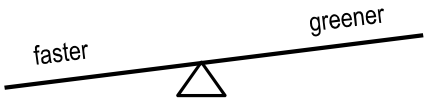
Server can run at multiple speeds

Facts:

- **Power is often a convex function of speed**
  - **Energy per job increases with speed**
- **There is always cost for delay**



CMOS ICs, disk drive motor



SWINBURNE UNIVERSITY OF TECHNOLOGY

"GOOD" DESIGNS "BALANCE" DELAY AND ENERGY

Design Goals

OPTIMALITY → Know workload → optimal setting?

ROBUSTNESS → λ unknown / wrong → how good is the "optimal" setting?

SIMPLICITY → Use few speeds

SWINBURNE UNIVERSITY OF TECHNOLOGY

Design Goals and Design Options

OPTIMALITY

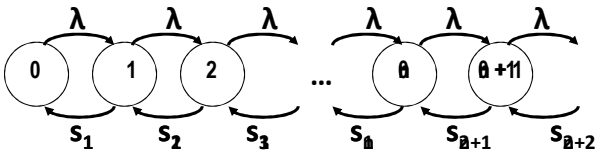
ROBUSTNESS

SIMPLICITY

HOW ? → Number of speeds; Which speeds; When to use each speed

SWINBURNE UNIVERSITY OF TECHNOLOGY

Model



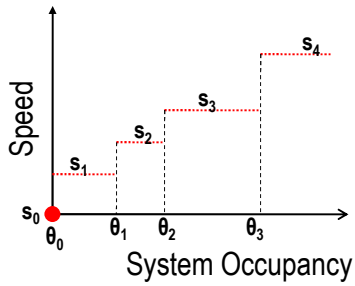
$$s_0 = 0, s_n \leq s_{max} \qquad \lambda < s_{max}$$

Unknown Known

≈3 for CMOS weight parameter

$$\text{Cost per job} = \frac{E[(s_N)^{\alpha}] + \beta E[N]}{\lambda}$$

K – speed scheme



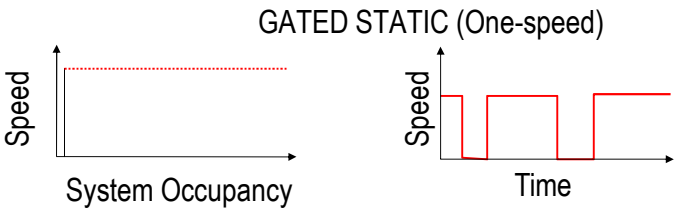
$$0 = s_0 \leq s_1 \leq \cdots \leq s_K \leq s_{K+1} = s_{max}$$

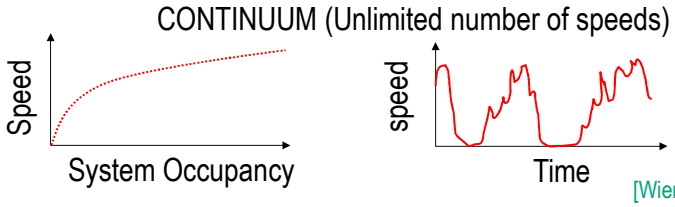
$$0 = \theta_0 \leq \theta_1 \leq \cdots \leq \theta_K \leq \theta_{K+1} = \infty$$

$$\text{Cost} = \pi_0 \left[\beta \sum_{i=1}^{K+1} \left(\prod_{j=1}^{i-1} \left(\frac{\lambda}{s_j}\right)^{\theta_j - \theta_{j-1}} \right) \left(\frac{\lambda}{s_i}\right)^{1-\theta_{i-1}} \frac{\left(\theta_{i-1}\left(\frac{\lambda}{s_i}\right)^{\theta_{i-1}-1} - \theta_i\left(\frac{\lambda}{s_i}\right)^{\theta_i - 1}\right)\left(1-\frac{\lambda}{s_i}\right) + \left(\left(\frac{\lambda}{s_i}\right)^{\theta_{i-1}} - \left(\frac{\lambda}{s_i}\right)^{\theta_i}\right)}{\left(1-\frac{\lambda}{s_i}\right)^2} \right.$$

$$\left. + \sum_{i=1}^{K+1} \left(\prod_{j=1}^{i-1} \left(\frac{\lambda}{s_j}\right)^{\theta_j - \theta_{j-1}} \right) \frac{\left(\frac{\lambda}{s_i}\right) - \left(\frac{\lambda}{s_i}\right)^{\theta_i - \theta_{i-1}+1}}{1-\frac{\lambda}{s_i}} P(s_i) \right] \qquad (\lambda \neq s_i) \;, P(s_i) = s_i^{\alpha}$$

Prior work – optimal costs

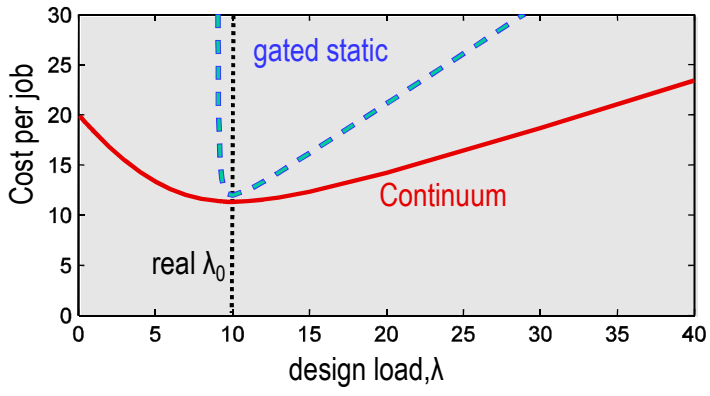




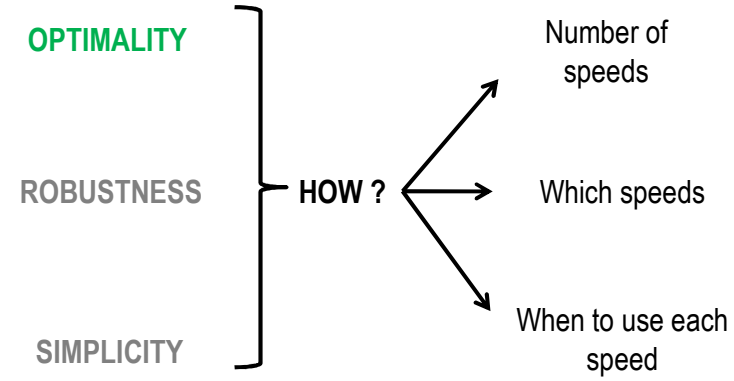
[Wierman, Andrew, Tang, INFOCOM 09]

$Cost_{gated\text{-}static} \leq 2xCost_{Continuum}$

Prior work - robustness



Design Goals and Design Options

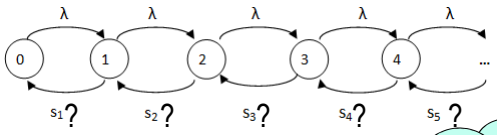


Optimal settings

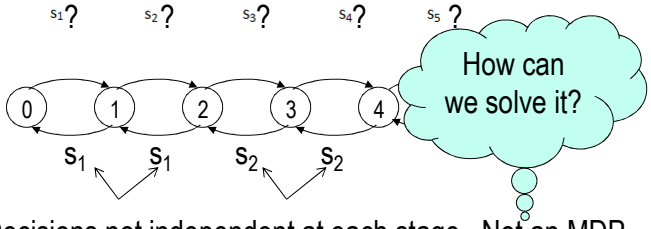
Gated-static: basic M/M/1 queue

$$0 = \frac{d}{d s_g}\left(E[s_g^{\alpha}] + \beta E[N]\right) = \frac{d}{d s_g}\left(\frac{\lambda}{s_g} s_g^{\alpha} + \beta \frac{\lambda}{s_g - \lambda}\right)$$

Continuum: Markov Decision Process



0 < K < ∞ :



Decisions not independent at each stage. Not an MDP

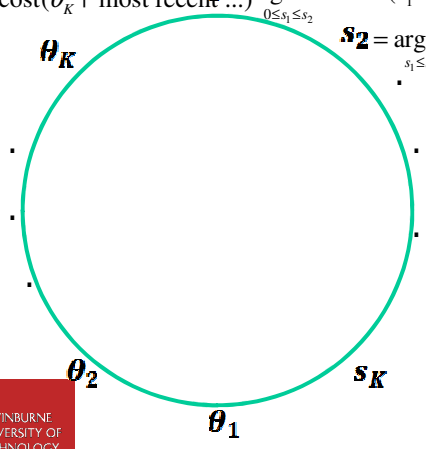
Gauss-Seidel iteration

Initial guess: $\pi_0 = (s_1, s_2, ..., s_K, \theta_1, \theta_2, ..., \theta_K)$

$\theta_K = \arg\min_{\theta_{K-1} \le \theta_K \le \infty} \text{cost}(\theta_K \mid \text{most recent ...})$

$s_1 = \arg\min_{0 \le s_1 \le s_2} \text{cost}(s_1 \mid \text{most recent } s_2, ..., s_K, \theta_1, ..., \theta_K)$

$s_2 = \arg\min_{s_1 \le s_2 \le s_3} \text{cost}(s_2 \mid \text{most recent } s_1, s_3, ...)$



Design Goals and Design Options

OPTIMALITY

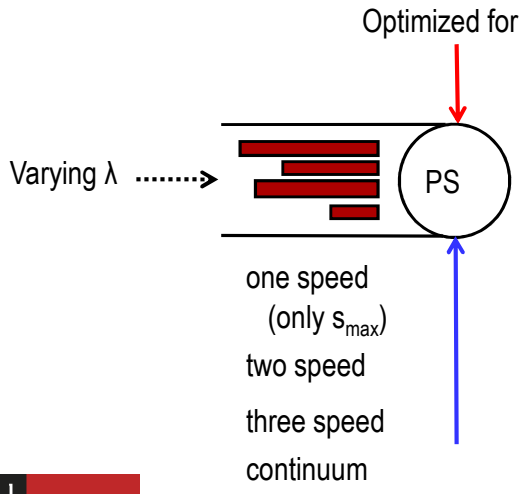
ROBUSTNESS

SIMPLICITY

HOW ?

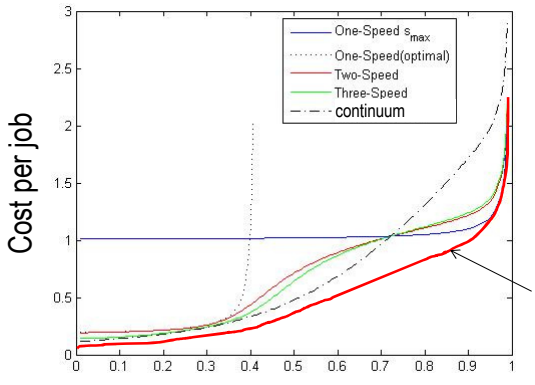
- Number of speeds
- Which speeds
- When to use each speed

Numerical Robustness Experiment



Numerical Robustness Experiment

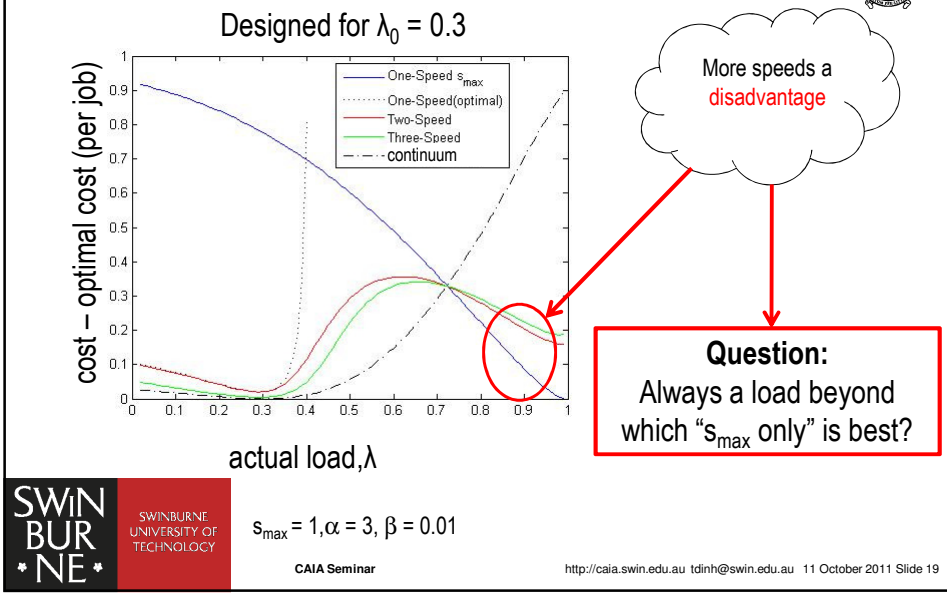
Designed for $\lambda_0 = 0.3$



Minimum possible cost:
Cost of Continuum,
designed for actual λ

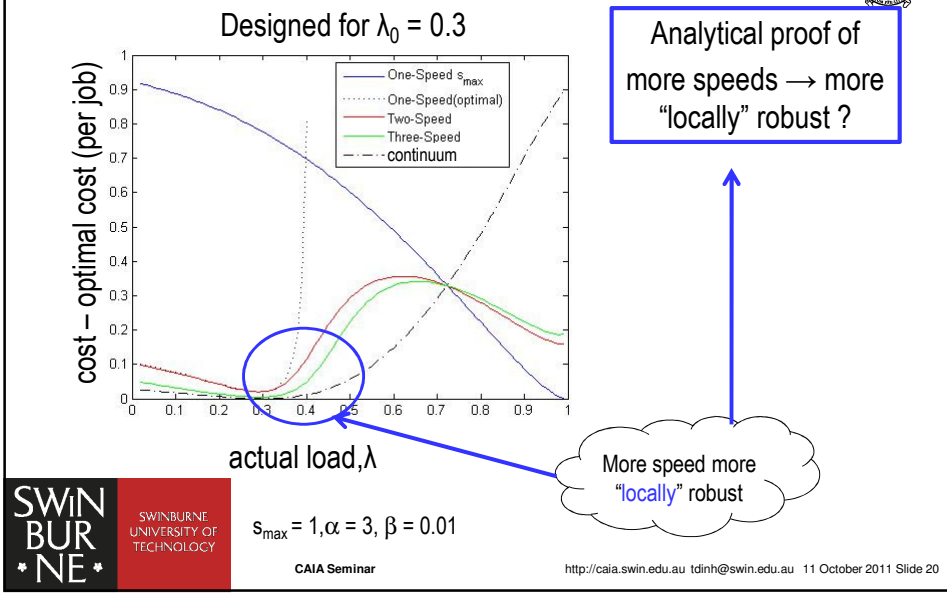
$s_{max} = 1, \alpha = 3, \beta = 0.01$

Numerical Robustness Experiment



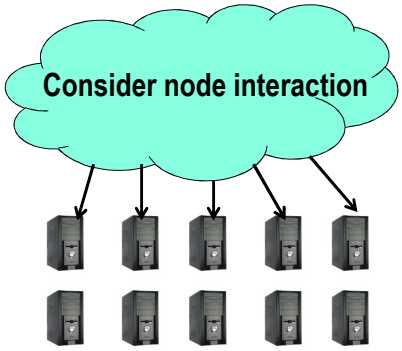
$s_{max} = 1, \alpha = 3, \beta = 0.01$

Numerical Robustness Experiment

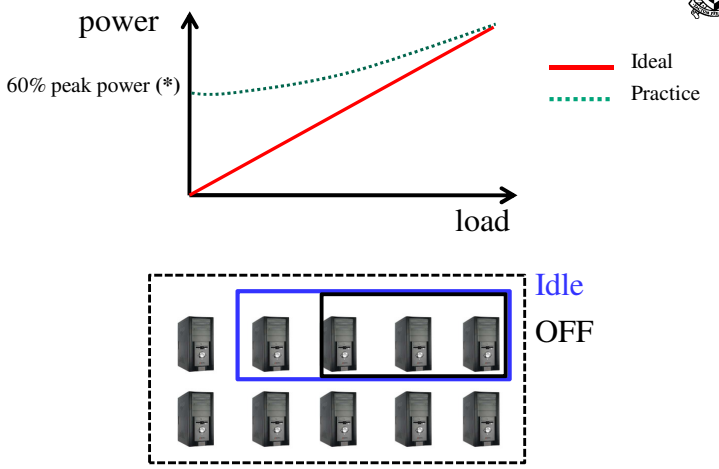


$s_{max} = 1, \alpha = 3, \beta = 0.01$

Approaches



Power Proportionality



(*) J. S Chase et al., "Managing Energy and Server Resources in Hosting Center", 2001

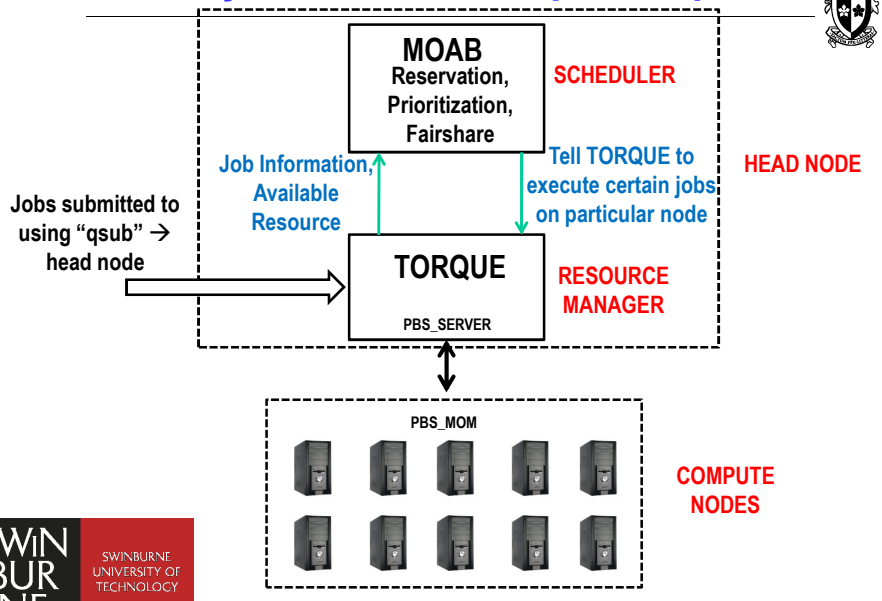
Prior work

Ignore switching cost

- J.S Chase et al., 2001: adjusts the active servers while avoiding violating SLA(s)
- E. Pinherio et al., 2001: load balancing vs. load concentration
- P. Bodik et al., 2008: Limited ON/OFF cycles before failure of Disks and IC chips
- M. Lin et al., 2011: dynamically change the size in order to minimize weighted sum of operational cost and switching cost

SWINBURNE UNIVERSITY OF TECHNOLOGY

Case Study – Swinburne Supercomputer



SWINBURNE UNIVERSITY OF TECHNOLOGY

Problem formulation

- Wear and tear cost (W&T):
  - adding delay (switching ON)
  - cost increases if node is turned ON/OFF too often
- One can form this optimization problem:

$$\min_{\text{active nodes at each time}} \sum_{time} f(E, D, W\&T)$$

wear and tear cost

delay cost

energy cost

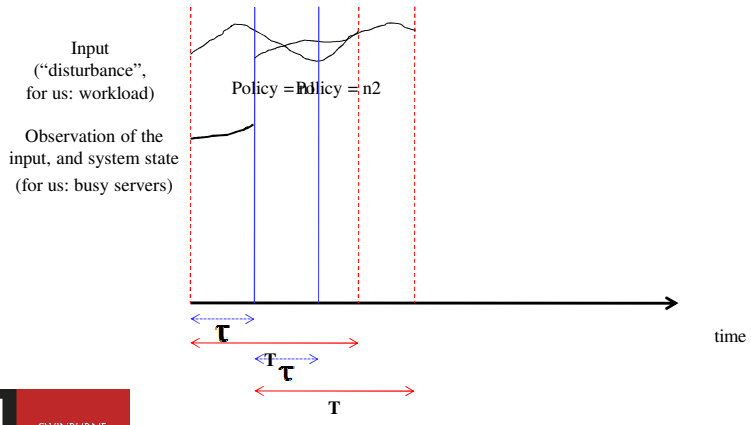
s.t no. active nodes > demand resources

Challenge: not knowing the future workload

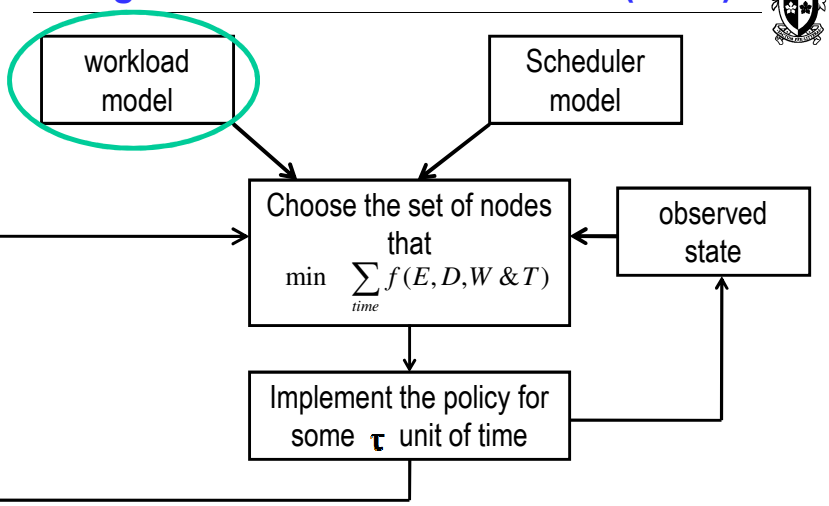
→optimization problem is not straight forward

Model Predictive Control (MPC)

- MPC is a control method for stochastic optimization

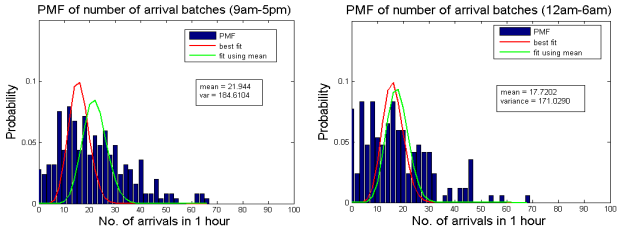


Using Model Predictive Control (MPC)



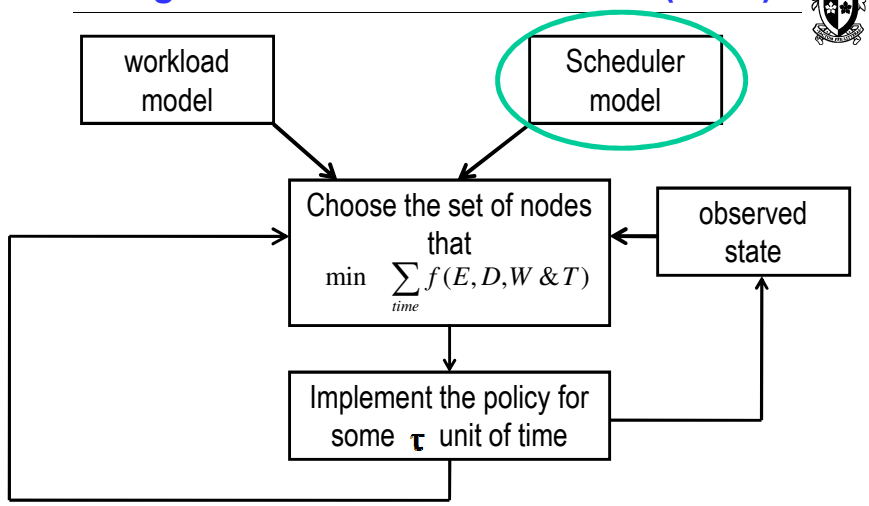
Green Machine historical workload

- Green Machine's workload is not Poisson

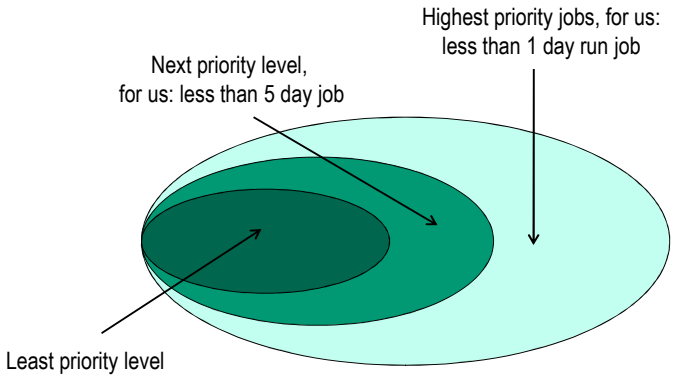


- Possible models for job arrival process
  - ☐ Time-Varying (batch) Poisson process
  - ☐ Using Markov Modulated (batch) Poisson Process
  - ☐ Per-user modelling

Using Model Predictive Control (MPC)

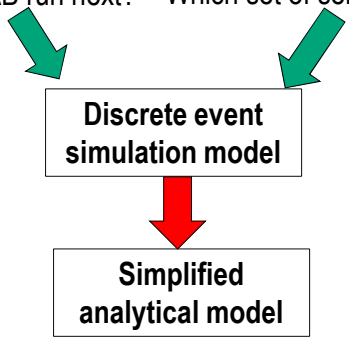


Limited machine choices



Where to from here?

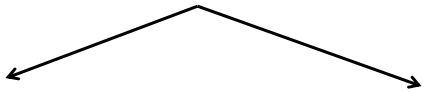
Which job does MOAB run next? Which set of servers can a job run on?



- Develop an optimization problem to find the best policy

Conclusion

- Energy consumption in HPC is significant and affects its operations.
- Most of HPC clusters are underutilised



Speed scaling

Having more speeds improves local robustness

Power ON/OFF

•Must consider W&T + delay + energy cost
•No knowledge of future workload → MPC