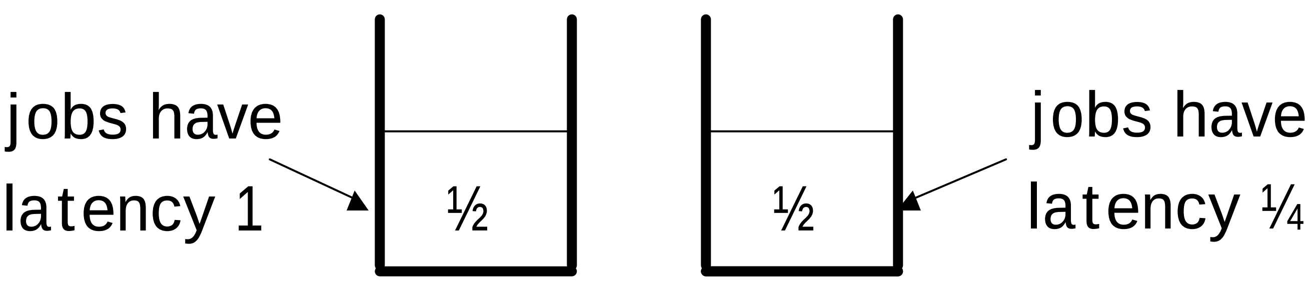
Stackelberg Scheduling Strategies

Tim Roughgarden
Cornell University

The Model

- m machines $1,2,\ldots,m$
- A quantity r of jobs
  - jobs are small (model in a cts way)
- For each machine i, a load-dependent latency function $l_i(\bullet)$
  - assume continuous, nondecreasing

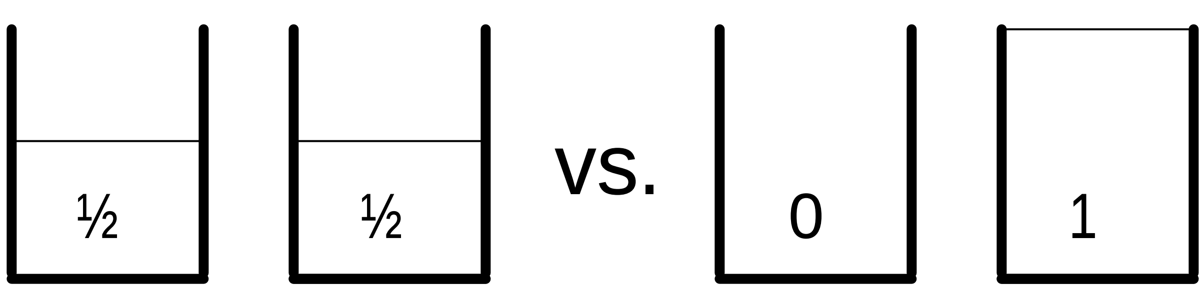
Example: $(r=1)$



$l_1(x)=1$ $l_2(x)=x^2$

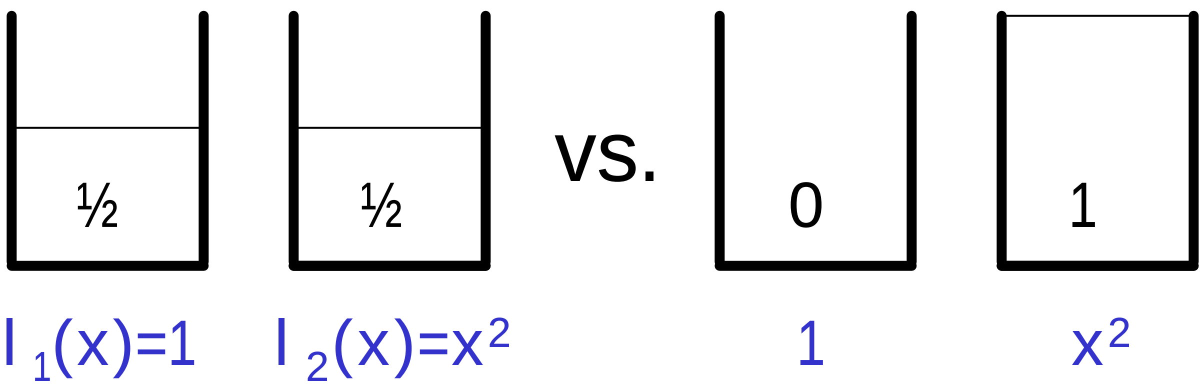
Equilibria

- which job assignments are "stable"?
  - jobs controlled by selfish, noncooperative agents
  - no job wants to switch machines (no job should be envious)



$l_1(x)=1$ $l_2(x)=x^2$ 1 x^2

More on Equilibrium Assignments



Def: an assignment is at Nash equilibrium (is a Nash assignment) if:

- all used machines have equal latency
- unused machines have greater latency

Fact: always have existence, uniqueness

How Good is an Assignment?

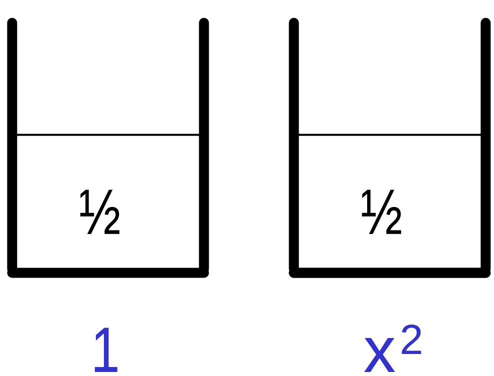
The Cost of an Assignment:

- $C(x)$ = cost or total latency experienced by assignment x:

$$\Sigma_i \, x_i \bullet l_i(x_i)$$

- our notion of system performance
- can optimize in poly-time

Example:



cost =

$\frac{1}{2} \bullet 1 + \frac{1}{2} \bullet \frac{1}{4}$ = ?

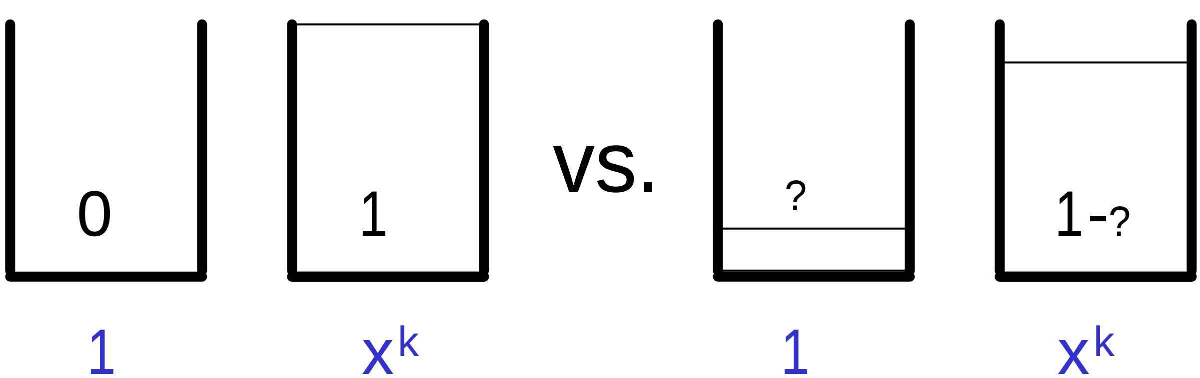
How Good are Nash Assignments?

Goal: prove that Nash assignments are near-optimal

- want a laissez faire approach to regulating users

Problem: false in general!

Example: (r=1, k large)



Nash costs 1 OPT costs ≈ 0

Near-Optimal Nash Assignments

Old Approach: weaken model, compare Nash vs. OPT (due to [Roughgarden/Tardos 00])

- general latency fns, weaker OPT

  Thm 1: cost of Nash = cost of OPT at rate $2r$

- stronger OPT, linear latency fns ($l_i(x)=a_i x+b_i$)

  Thm 2: cost of Nash = 4/3 cost of OPT (at rate r)

Taming Selfishness through a Manager

New Approach:

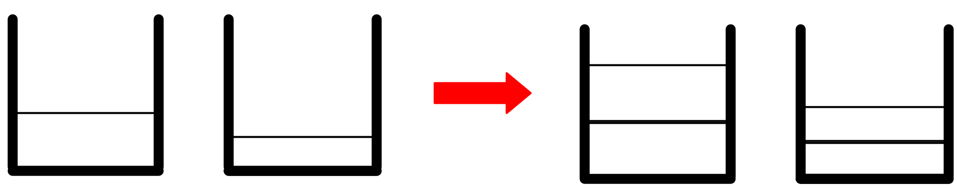
- not all jobs need be controlled by selfish users
  - "centrally controlled" vs. "selfishly controlled" jobs
  - behavior of selfish users depends on assignment of managed jobs

Goal:

- assign centrally controlled jobs to induce "good" selfish behavior
  - see also [Korlis, Lazar, Orda 97]

Stackelberg Strategies

- Stackelberg strategy = assignment of centrally controlled jobs
  - Þ yields an induced equilibrium



- Basic Questions:
  - what's the best strategy?
    - can we compute/characterize it?
  - how inefficient is the best induced equilibrium?
    - are we provably near-optimal?

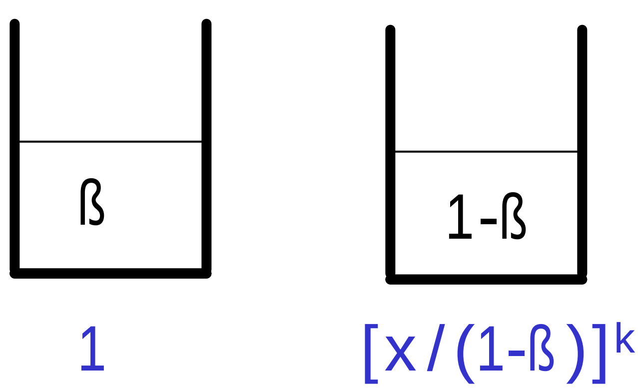
Our Results - General Latency Functions

Theorem 1: Can efficiently compute a strategy inducing an equilibrium with cost

$= (1/ß) \times$ cost of opt assignment

($ß$ = fraction of centrally assigned jobs)

Fact:
$(1/ß) \times \text{OPT}$
is best possible



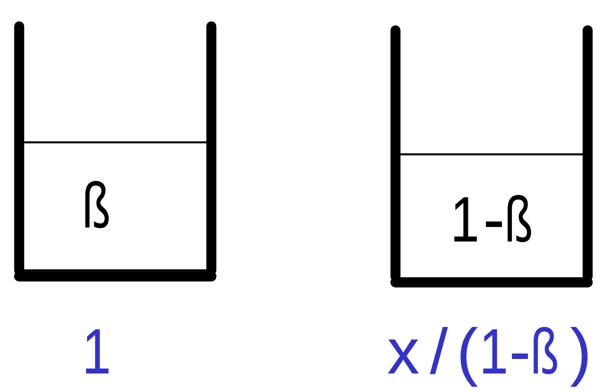
Our Results - Linear Latency Functions

Theorem 2: Can efficiently compute a strategy inducing an equilibrium with cost

= [4/(3+ß)] × cost of OPT

(ß = fraction of centrally assigned jobs)

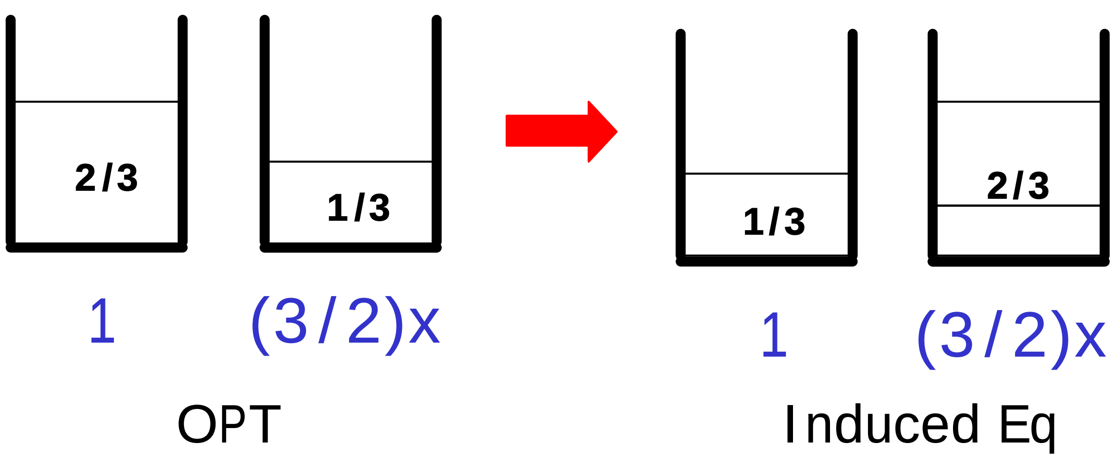
Fact: [4/(3+ß)] × OPT is best possible



What makes a strategy (in)effective?

The Scale Strategy

- compute optimal assignment x of all jobs, assign centrally controlled jobs via ß • x

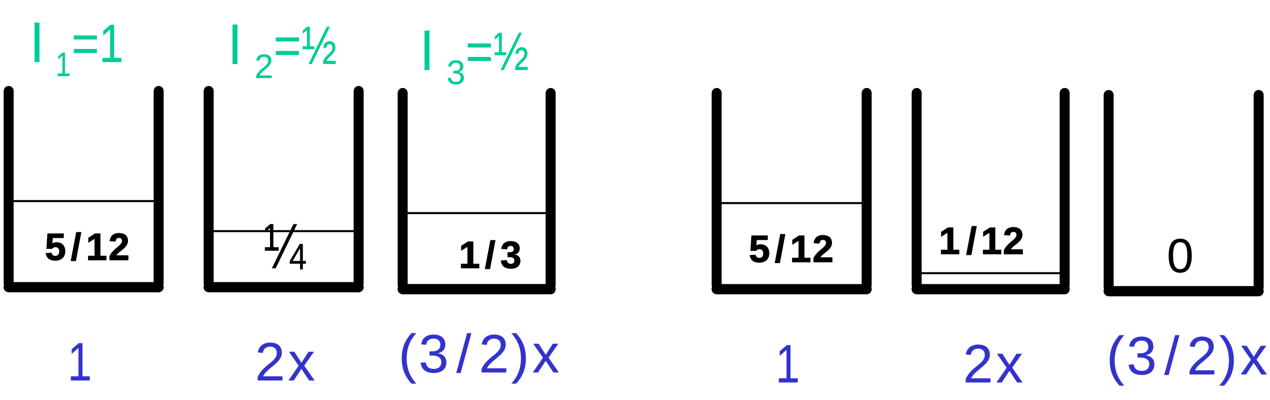


Moral: avoid machines that selfish users will (over)use anyways

The LLF Strategy

Largest Latency First (LLF):

- compute opt, x, for all jobs
- assign x_i jobs to i in order of decreasing $l_i(x_i)$'s
  (until no managed jobs remain)



OPT (r=1)

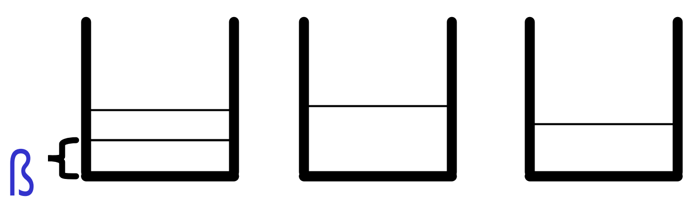
LLF (ß=½)

LLF with General Latency Functions

Theorem 1: The LLF strategy induces an equilibrium with cost = (1/ß) × cost of opt assignment.

Proof idea: Exploit iterative structure of LLF to proceed by induction on # of machines.

Base case:



common latency = L

•LLF Þ Machine 1's latency ≥ L

•OPT has ≥ ß jobs on machine 1

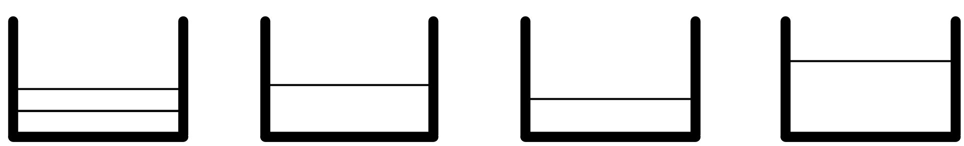
Þ OPT pays ≥ ßL, we pay L

LLF with Linear Latency Functions

Theorem 2: The LLF strategy induces an equilibrium with cost $= [4/(3+ß)] \times$ cost of OPT.

Main difficulty:

- previous argument too weak for:



- need detailed study of Nash, OPT when all latencies are linear

Computing Optimal Strategies

We've seen: LLF has the best possible worst-case guanantee

Question: is the LLF strategy optimal on all instances?

Bad news: no, in fact:

Theorem 3: Computing the optimal strategy is NP-hard (even for linear latency fns).

- Compare to: Opt, Nash assignments

Open Questions

Approximating the optimal strategy:

- LLF is best possible using OPT as a lower bound
- better guarantees for LLF via a better lower bound?
- more sophisticated algorithms?
  - Thm 3 is reduction from Partition
  - existence of a (F)PTAS?

General Graphs

Open:

- for general latency fns, fixed ß: a strategy inducing an equilibrium w/cost = f(ß) × opt
  - 1/ß not achievable in general graphs (!)
  - maybe 2/ß? (or O(n))
- for linear latency fns, a strategy w/cost < 4/3 × opt
  - e.g., is 8/7 achievable for ß=½?