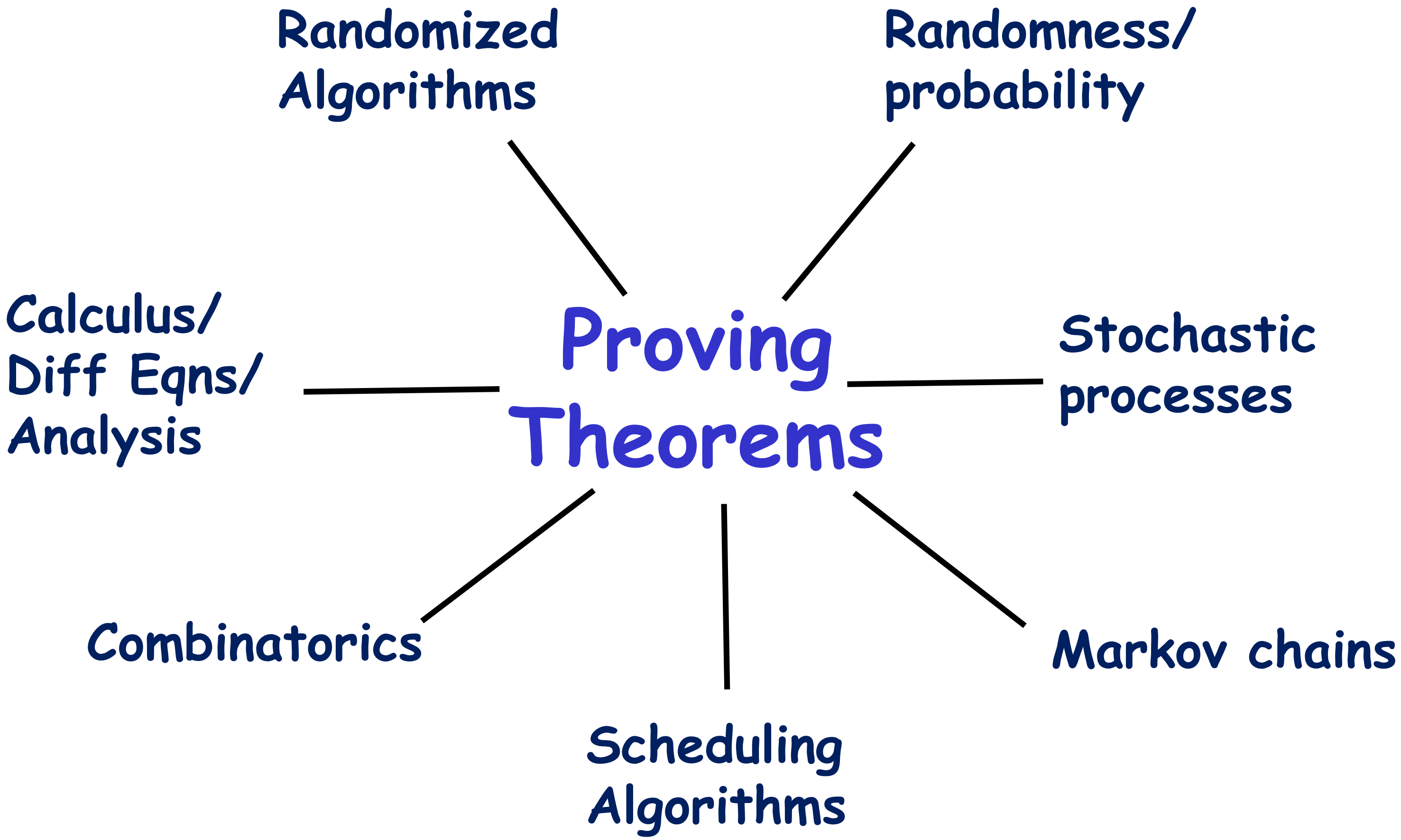
Computer Performance:

Queueing and Scheduling

Mor Harchol-Balter
Computer Science Dept, CMU

Accepting new student

Work with me if you like …



Work with me if you want to do theory that computer systems folks care about …

Typical Systems Goals

- "Reducing tail latency"
- "Guaranteeing flow fairness"
- "Combatting transient overload"
- "Optimally assigning heterogeneous resources to heterogeneous jobs"
- "Minimizing energy usage while meeting QoS latency goals"
- "Dynamic capacity provisioning"

⇒ **Theory**

Work with me if you want to do theory that computer systems folks care about …

Typical Systems Goals

- "Reducing tail latency"
- "Guaranteeing flow fairness"
- "Combatting transient overload"
- "Optimally assigning heterogeneous resources to heterogeneous jobs"
- "Minimizing energy usage while meeting QoS latency goals"
- "Dynamic capacity provisioning"

→ Scheduling Theory

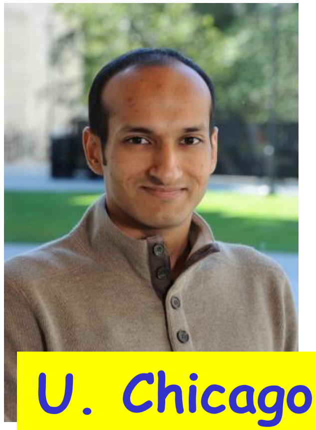
Work with me if you want to end up in academia …

U. Toronto

Caltech

IBM Research

StonyBrook



U. Chicago

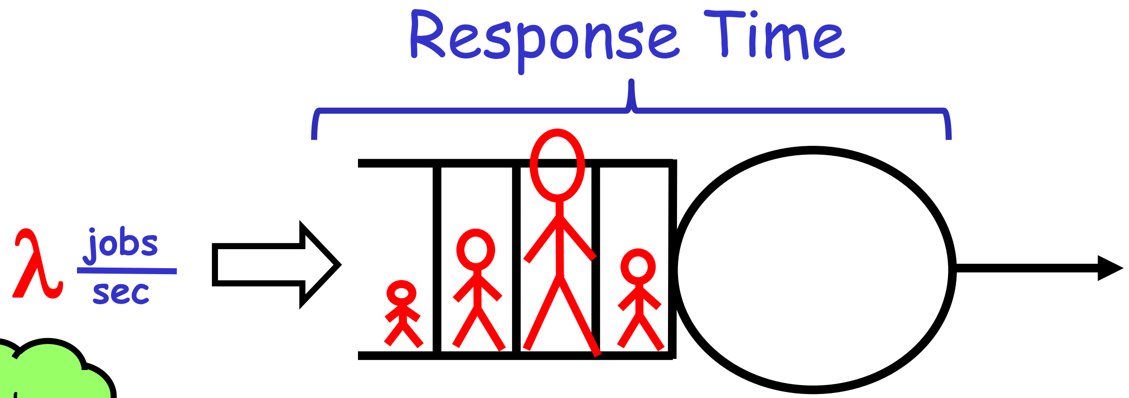
U. Minnesota

Amherst

Penn State

- Analysis
- Speaking
- Teaching
- Writing
- Health

Scheduling 101



$\lambda = 2 \frac{jobs}{sec}$

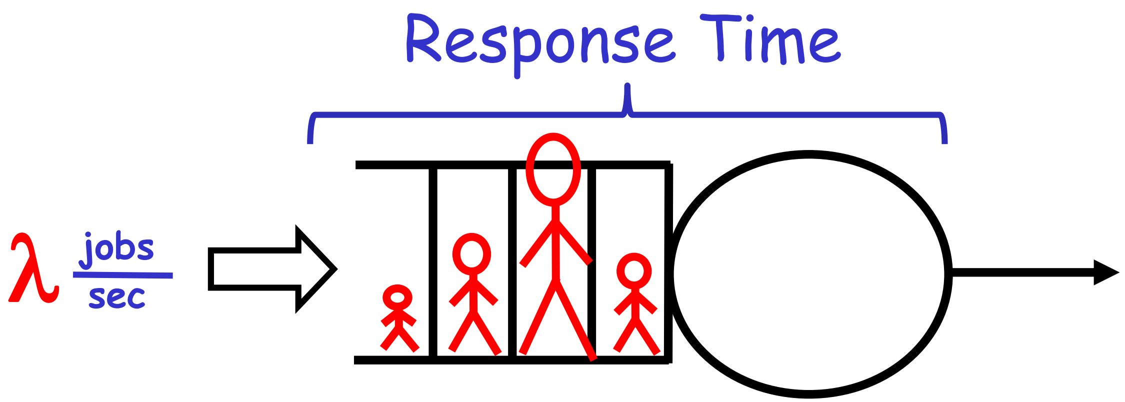
S = job size (sec) $E[S] = \frac{1}{3}$ sec

ρ = load = Fraction of time server is busy

$\rho = \lambda E[S] = \frac{2}{3}$

Q: What scheduling policy minimizes E[RespTime]?

Scheduling 101



Non-Preemptive

FCFS: First Come First Served

LCFS: Last Come First Served

Random: Pick Random Job

SJF: Shortest Job First

Q: How do these compare wrt E[Resp Time]?

Preemptive

PS: Processor Sharing

SRPT: Shortest Remaining Processing Time

Q: Is SRPT optimal?

Optimality of SRPT

arrival time

size

Defn: Arrival Sequence $A = \{(a_1, s_1), (a_2, s_2), (a_3, s_3), \ldots\}$

Claim: $\forall A,\ SRPT(A)$ is optimal for $E[\text{RespTime}]$

Proof Sketch: ?

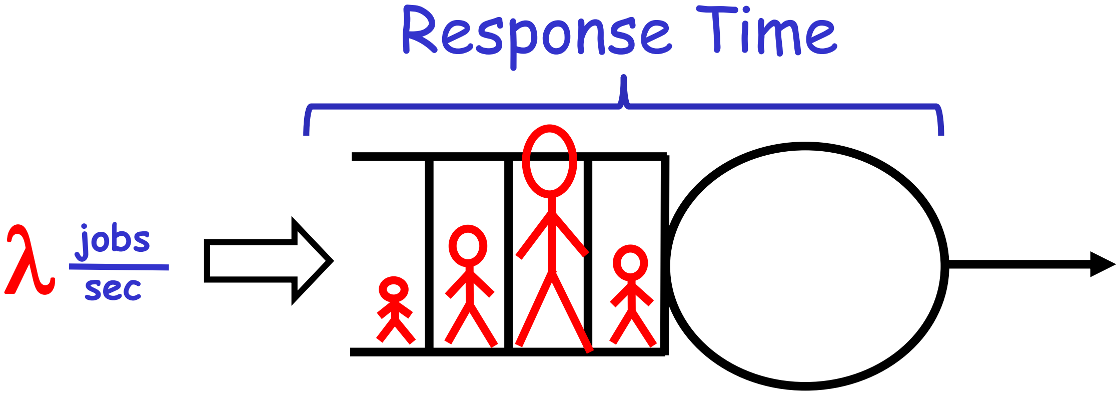
Consider a given A.

Define $OPT(A)$ as yielding optimal response time for A.

Assume that $SRPT(A) \neq OPT(A)$.

Prove there's a contradiction.

Scheduling Optimality



Non-Preemptive

FCFS: First Come First Served

LCFS: Last Come First Served

Random: Pick Random Job

SJF: Shortest Job First

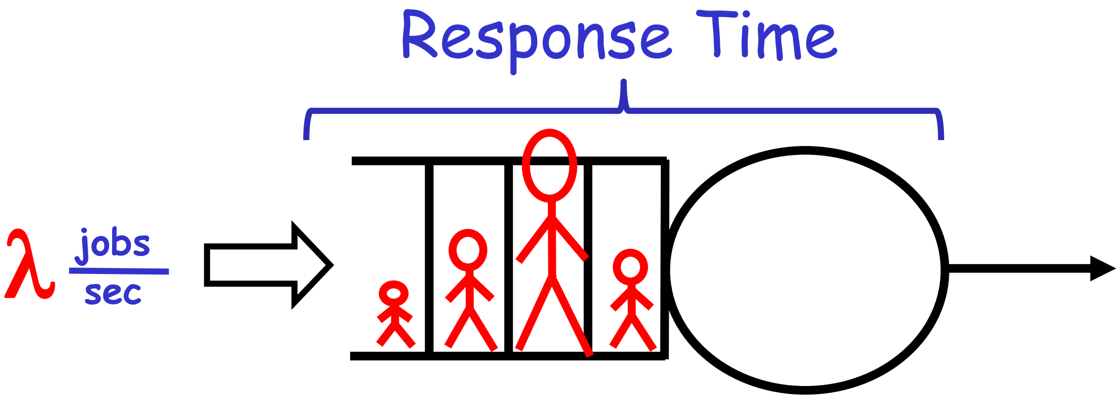
Preemptive

PS: Processor Sharing

SRPT: Shortest Remaining Processing Time

Q: Is SRPT optimal for E[Resp Time]?

Scheduling Optimality



Non-Preemptive

FCFS: First Come First Served

LCFS: Last Come First Served

Random: Pick Random Job

SJF: Shortest Job First

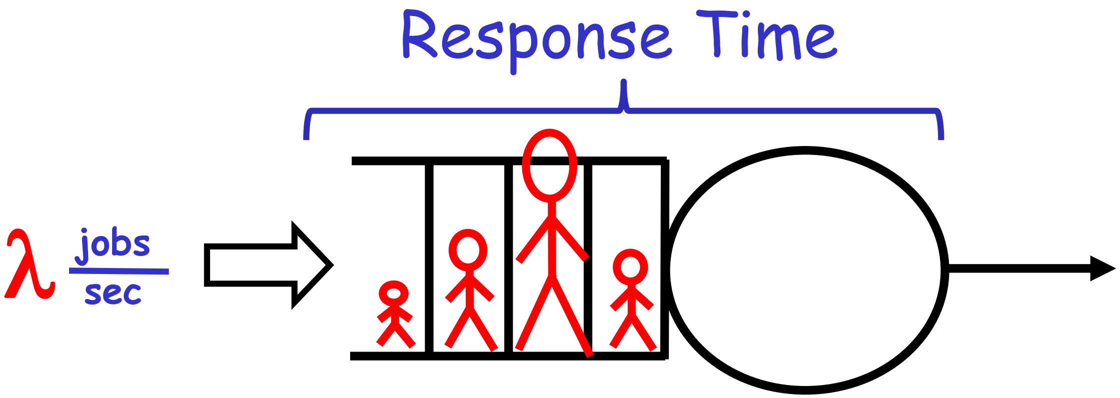
Preemptive

PS: Processor Sharing

SRPT: Shortest Remaining Processing Time

Q: Is SRPT optimal for 99th %-tile Resp Time?

Scheduling Optimality



Non-Preemptive

FCFS: First Come First Served

LCFS: Last Come First Served

Random: Pick Random Job

SJF: Shortest Job First

Preemptive

PS: Processor Sharing

SRPT: Shortest Remaining Processing Time

Q: What is optimal for Max response time?

Slowdown Metric

$$\text{Slowdown(job)} = \frac{\text{RespTime(job)}}{\text{Size(job)}}$$

Q: Does SRPT minimize E[Slowdown]?

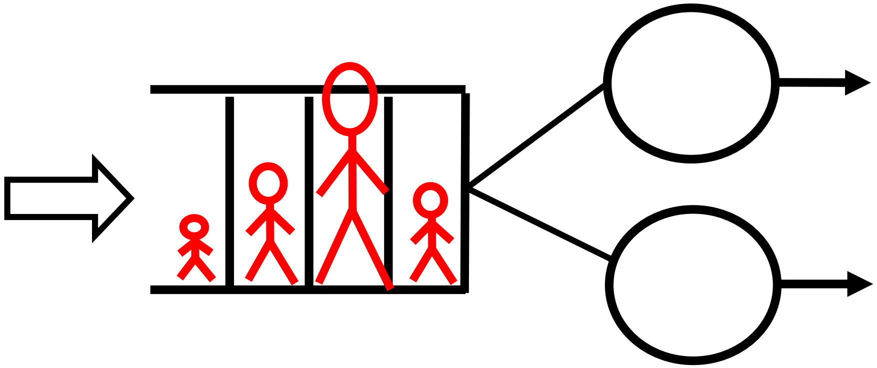
Yes?

Q:Prove it!

No?

Q:What scheduling policy is optimal for E[Slowdown]?

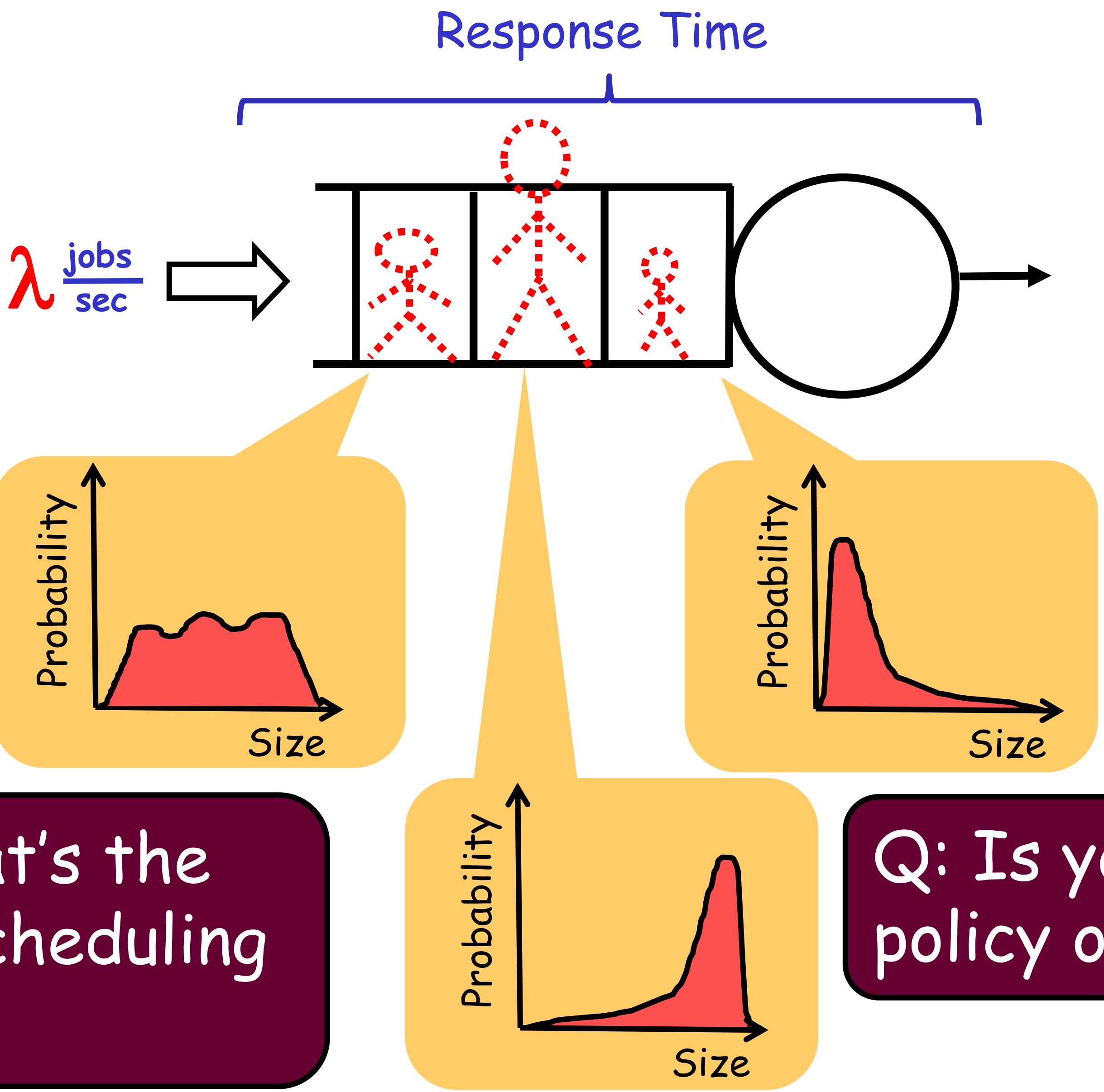
Minimizing Response Time for 2 Servers



Q: How should we schedule to minimize E[Response Time] given 2 servers? (assume you can preempt jobs)

Q: Is your policy optimal?

What if Don't Know Size?



Q: What's the right scheduling policy?

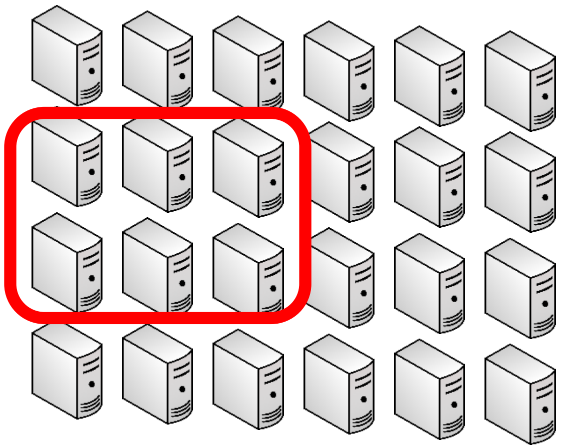
Q: Is your policy optimal?

Real World Jobs are Multi-Dimensional

"job" = (# servers, duration)

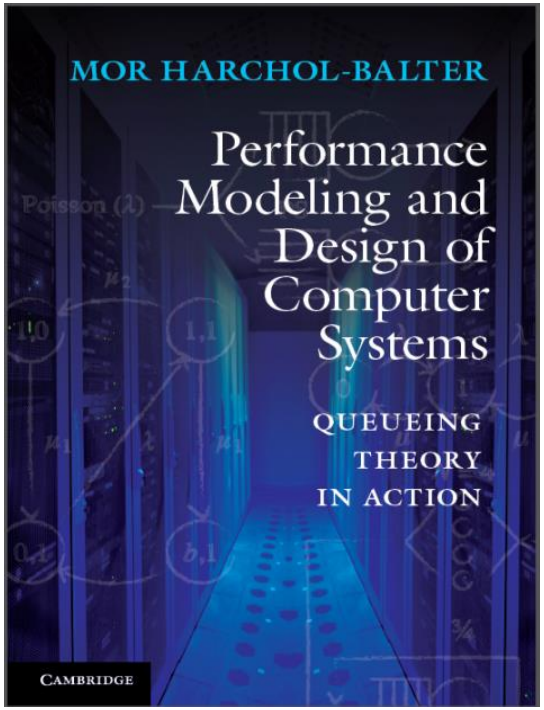
(500srvs, 1hr)
(575srvs, 1min)
(32srvs, 12hrs)
⋮

1 "cell" at Google = 10,000 servers



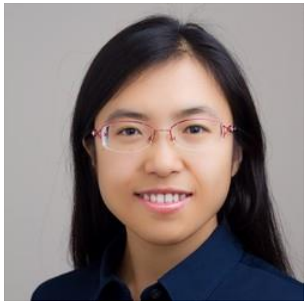
Q: Stability condition?
Q: E[Response time]?
Everything is open!

Want to learn more … take my class!



This Semester: 15-857
Analytical Performance Modeling
MWF 1:25 p.m.

co-Instructor:



Awesome TAs:

www.cs.cmu.edu/~harchol/