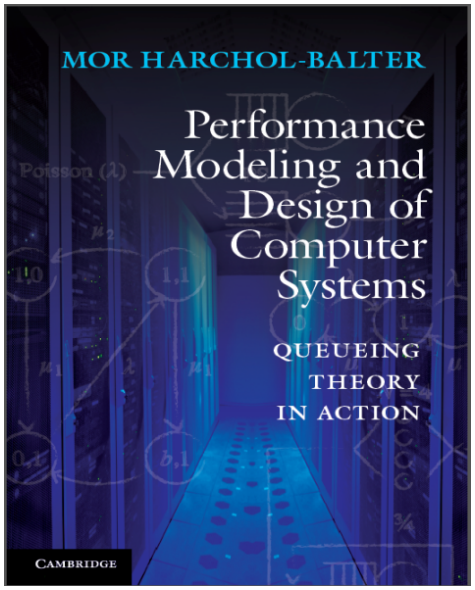
Introductory Queueing Theory Tutorial

Mor Harchol-Balter
Computer Science Dept, Carnegie Mellon Univ.



Outline

I. Basic Vocabulary

- Avg arrival rate, λ
- Avg service rate, μ
- Avg load, ρ
- Avg throughput, X
- Response time, T
- Little's Law
- Exponential vs. Pareto/Heavy-tailed
- Poisson Process

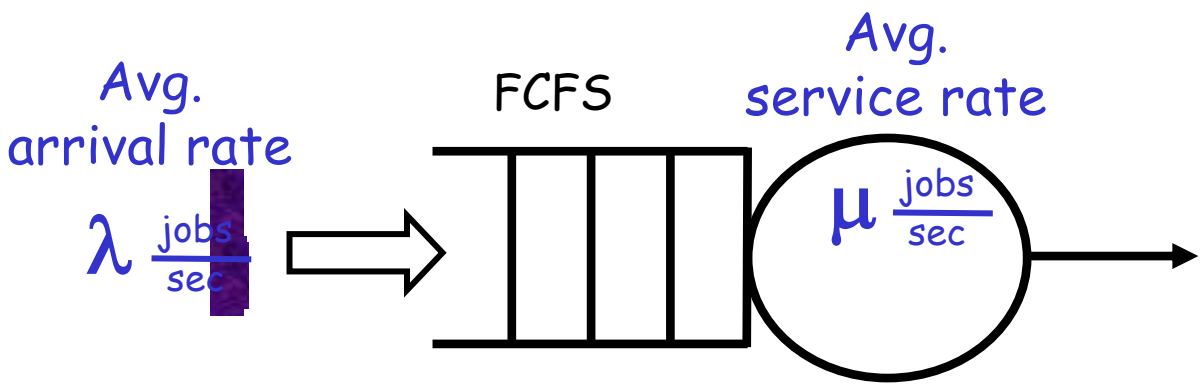
II. Single-server queues

- M/G/1 response time
- Inspection Paradox
- Effect of job size variability
- Effect of load
- Scheduling: FCFS, PS, SJF, LAS, SRPT
- Scheduling: Priority Classes
- Scheduling: SOAP Framework (New)

III. Multi-server queues

- Single shared queue, M/G/k
- Load balancing across queues
- Cycle stealing
- Replication of jobs (New)
- Multi-task jobs and fork-join (New)
- Networks of queues

Vocabulary



$\lambda < \mu$ throughout

S = job size = service requirement

$$E[S] = \frac{1}{\mu} \text{ sec}$$

Example:

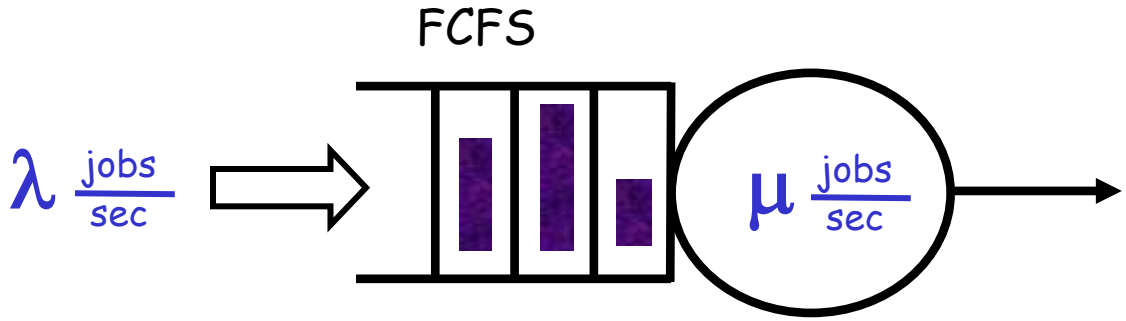
- On average, job needs 3×10^6 cycles
- Server executes 9×10^6 cycles/sec

Avg service rate $\mu = 3 \frac{\text{jobs}}{\text{sec}}$

Avg size of job on this server:

$E[S] = \frac{1}{3}$ sec.

Vocabulary: Load



S: job size $\quad E[S] = \frac{1}{\mu}$

ρ = Load (utilization) = Frac. time server busy = $\lambda E[S] = \frac{\lambda}{\mu}$

Example:

- $\lambda = 2$ $\frac{\text{jobs}}{\text{sec}}$ arrive
- Each job requires $E[S] = \frac{1}{3}$ sec on avg

$\rho = \frac{2}{3}$

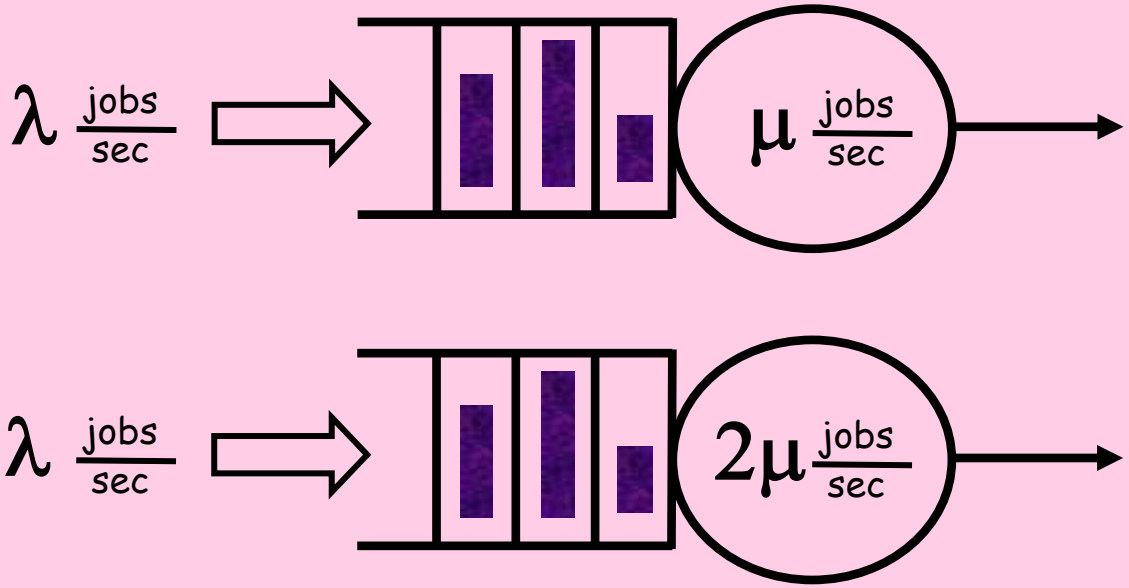
Vocabulary: Throughput

Defn: Throughput X is the average rate at which jobs complete (jobs/sec)

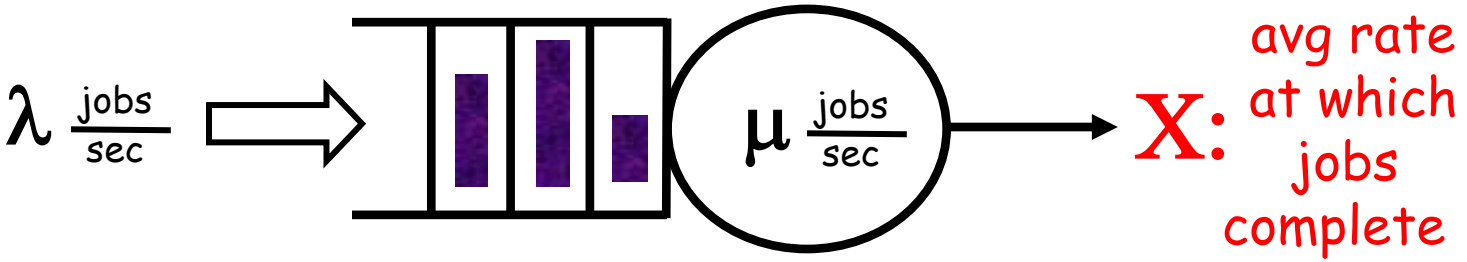
QUESTION:

Which has higher throughput, X?

Assume $\lambda < \mu$

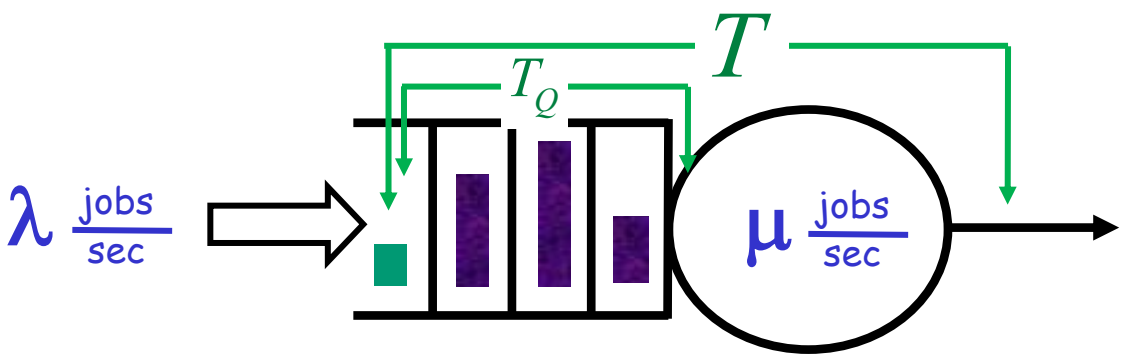


Vocabulary: Throughput



$$X = \lambda$$ (assuming no jobs dropped)

Vocabulary: Response Time



S : job size

$$E[S] = \frac{1}{\mu}$$

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

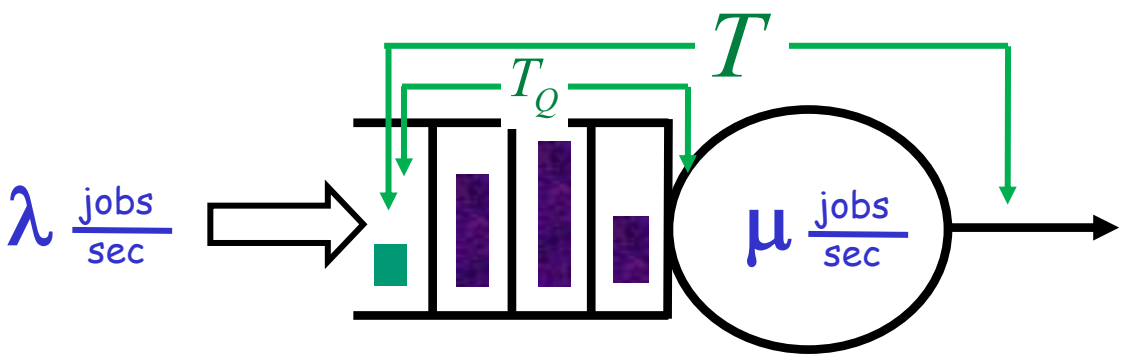
$T =$ response time

$T_Q =$ queueing time (waiting time)

$N =$ Number jobs in system

Little's Law: $E[T] = \dfrac{E[N]}{\lambda}$.

Vocabulary: Response Time



S : job size

$$E[S] = \frac{1}{\mu}$$

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

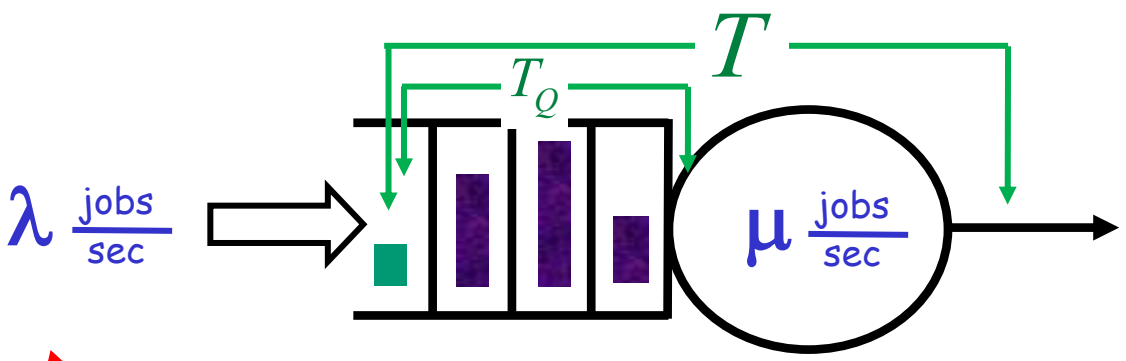
T = response time

T_Q = queueing time (waiting time)

Q: Given that $\lambda < \mu$, what causes wait?

A: Variability in the arrival process & service requirements

Vocabulary: Response Time



S : job size

$$E[S] = \frac{1}{\mu}$$

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

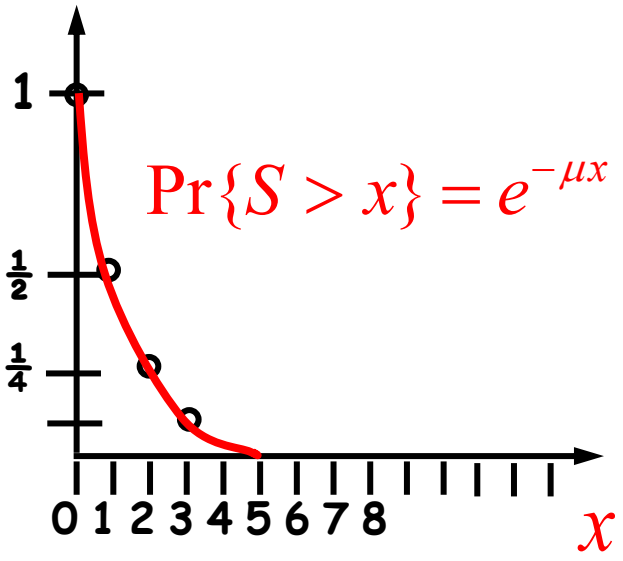
Variability in arrival process

Variability in job size, S

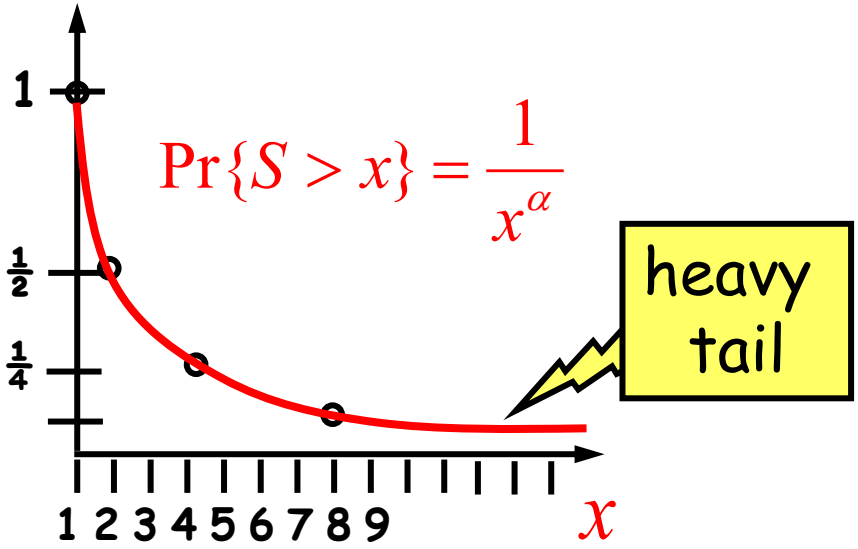
Job Size Distributions

"Most jobs are small; few jobs are large"

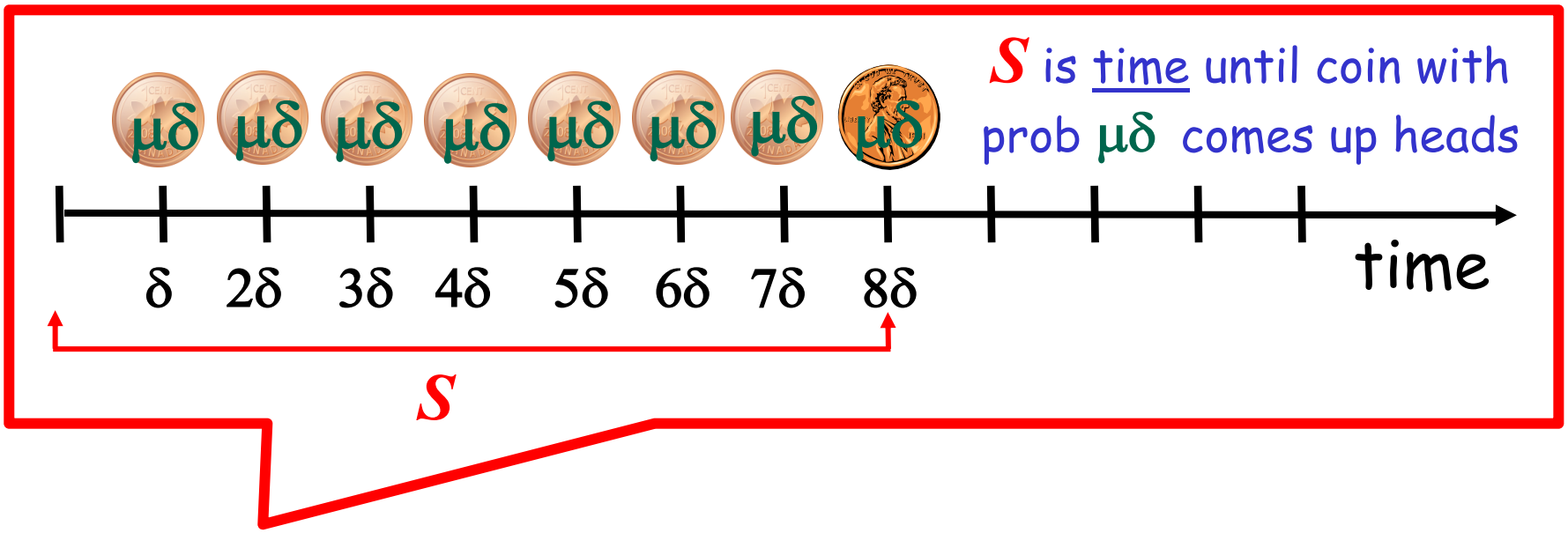
$S \sim \text{Exp}(\mu)$



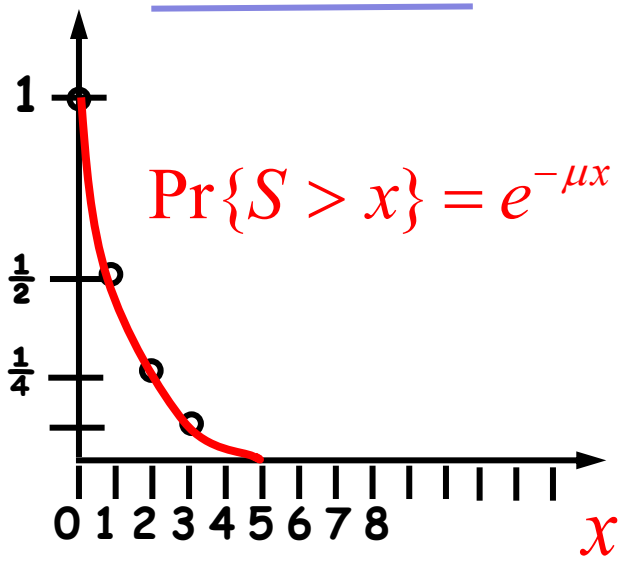
$S \sim \text{Pareto}(\alpha)$



Job Size Distributions

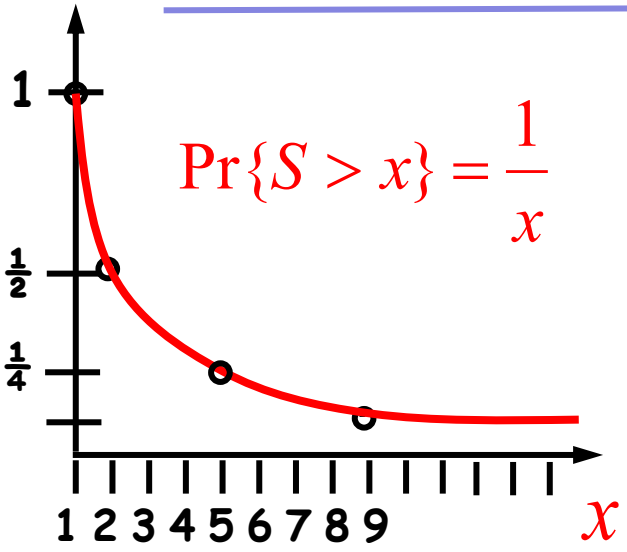


$S \sim \text{Exp}(\mu)$



$$\Pr\{S > x\} = e^{-\mu x}$$

$S \sim \text{Pareto}(\alpha = 1)$



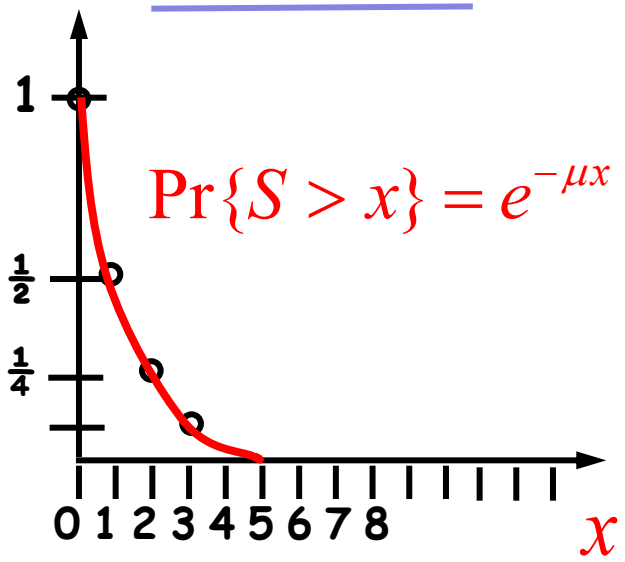
$$\Pr\{S > x\} = \frac{1}{x}$$

Job Size Distributions

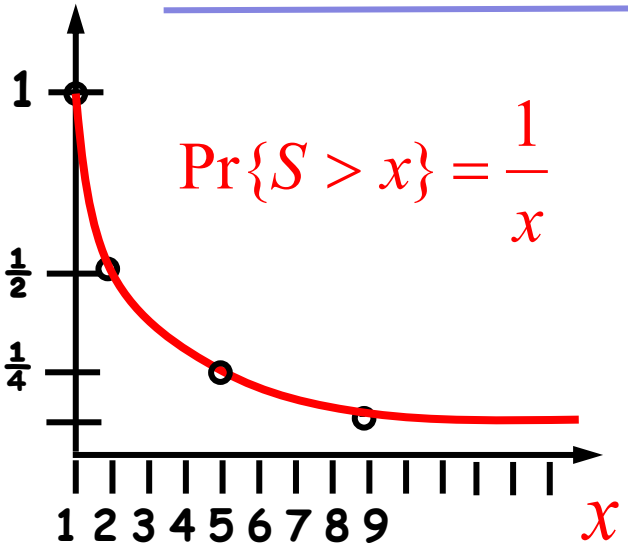
- "Memoryless"
- Lower variability
- Light-tail:
  top 1% of jobs
  comprise 5% load.

- Decreasing hazard rate
- Infinite variance
- Heavy-tail:
  top 1% of jobs
  comprise 50% load.

$S \sim \text{Exp}(\mu)$



$S \sim \text{Pareto}(\alpha = 1)$

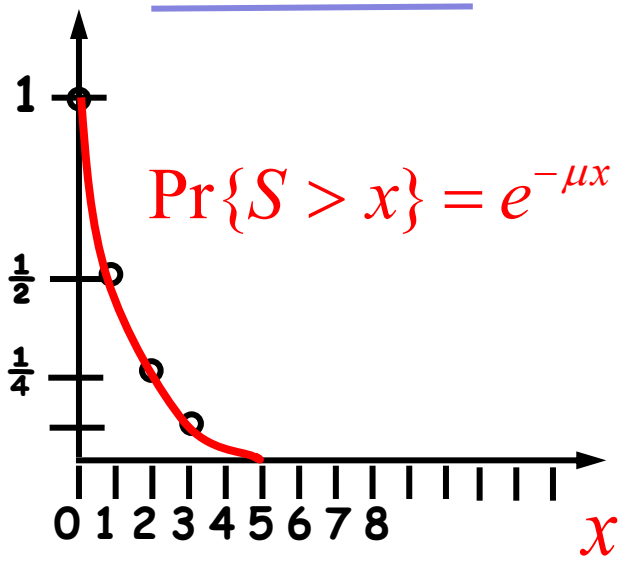


Job Size Distributions

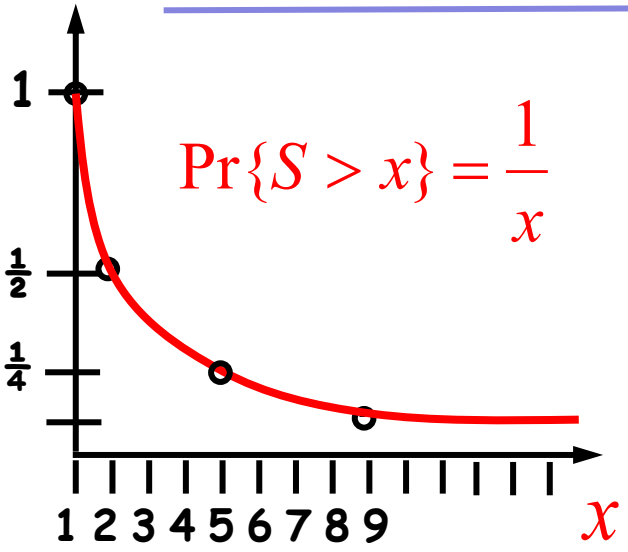
- "Memoryless"
- Lower variability
- Light-tail:
  top 1% of jobs
  comprise 5% load.

Representative of:
-- UNIX job sizes sizes
-- Supercomputing job sizes
-- File sizes
-- Human wealth
-- Damage due to forest fires, earthquakes, etc.

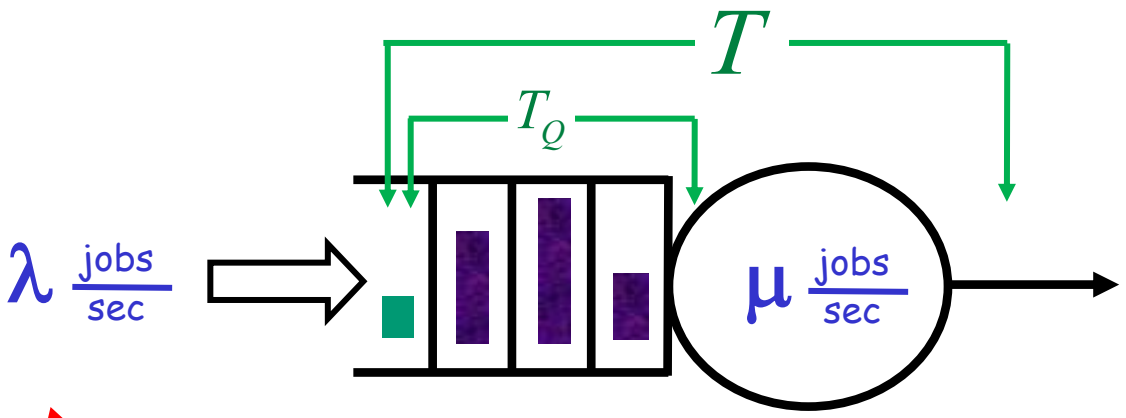
$S \sim \text{Exp}(\mu)$



$S \sim \text{Pareto}(\alpha = 1)$



Variability



S : job size

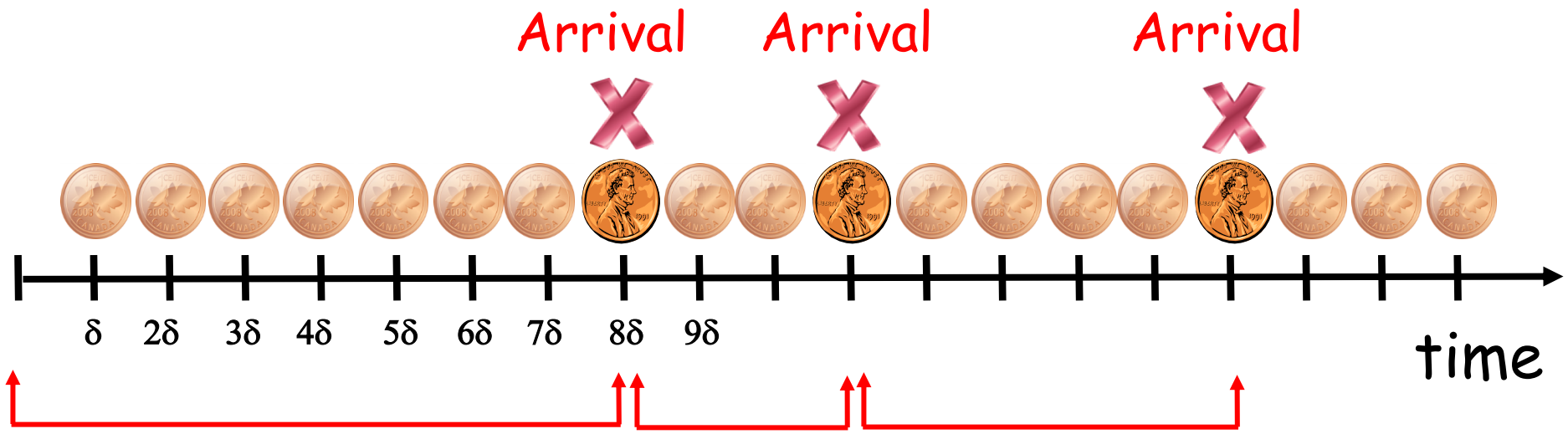
$$E[S] = \frac{1}{\mu}$$

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

Variability in arrival process

Variability in job size, S

Vocabulary: Poisson Process with rate λ



$S \sim \text{Exp}(\lambda)$ $S \sim \text{Exp}(\lambda)$ $S \sim \text{Exp}(\lambda)$

(Poisson process comes up when aggregating many users)

Outline

I. Basic Vocabulary

- Avg arrival rate, λ
- Avg service rate, μ
- Avg load, ρ
- Avg throughput, X
- Response time, T
- Little's Law
- Exponential vs. Pareto/Heavy-tailed
- Poisson Process

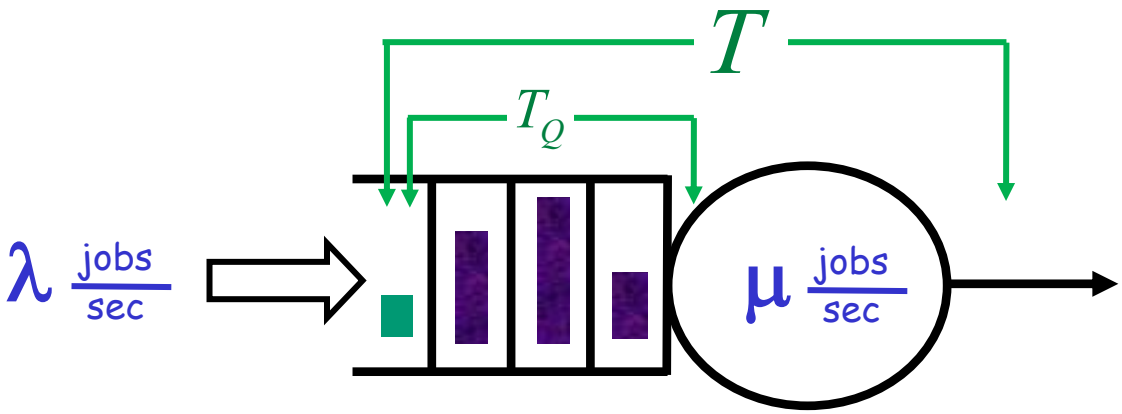
II. Single-server queues

- M/G/1 response time
- Inspection Paradox
- Effect of job size variability
- Effect of load
- Scheduling: FCFS, PS, SJF, LAS, SRPT
- Scheduling: Priority Classes
- Scheduling: SOAP Framework (New)

III. Multi-server queues

- Single shared queue, M/G/k
- Load balancing across queues
- Cycle stealing
- Replication of jobs (New)
- Multi-task jobs and fork-join (New)
- Networks of queues

Single-Server Queue



S : job size

$$E[S] = \frac{1}{\mu}$$

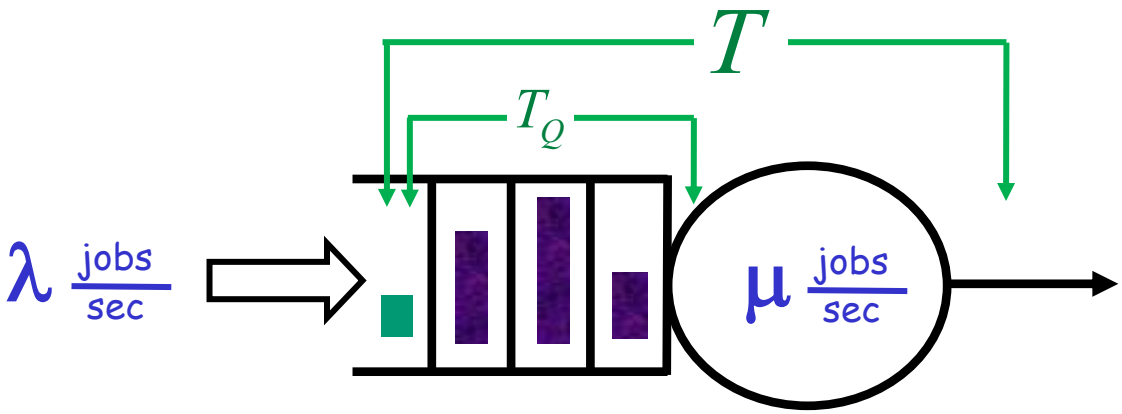
$$\rho = \lambda E[S] = \frac{\lambda}{\mu}$$

M/G/1

- Exponential inter-arrival times (M = memoryless)
- General i.i.d. service times
- 1 server

Q: Does low ρ ➔ low $E[T_Q]$?

Single-Server Queue



S : job size

$$E[S] = \frac{1}{\mu}$$

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

M/G/1

- Exponential inter-arrival times (M = memoryless)
- General i.i.d. service times
- 1 server

A: low load does NOT ensure low wait

M/G/1

$$E[T_Q] = \frac{\rho}{1-\rho} \cdot \frac{E[S^2]}{2E[S]}$$

Where is this coming from?

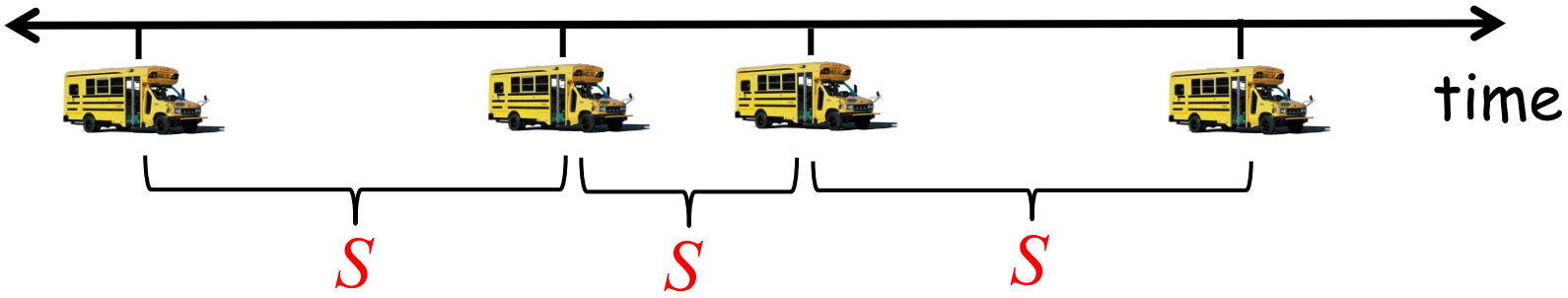
A: low load does NOT ensure low wait

Waiting for the bus

Waiting for the bus

S: time between buses

$$E[S] = 10\,\text{min}$$



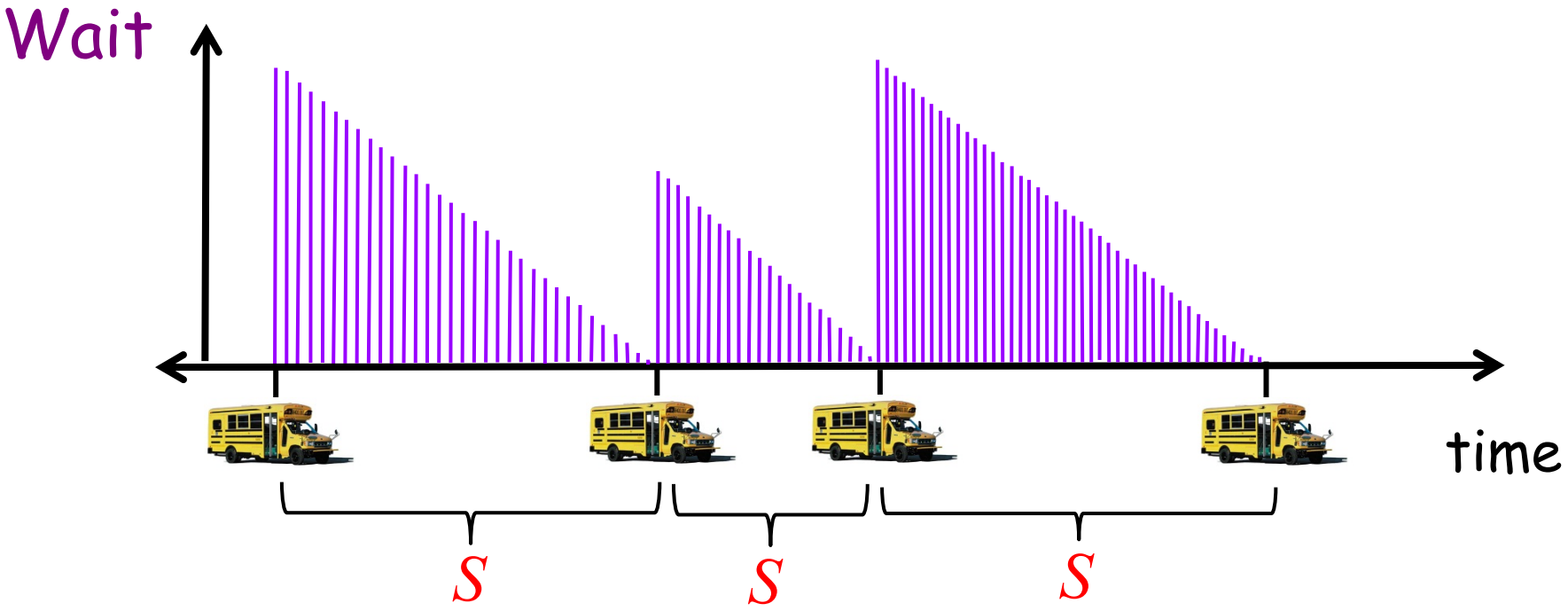
QUESTION:

On average, how long do I have to wait for a bus?

(a) < 5 min
(b) 5 min
(c) 10 min
(d) >10 min

Waiting for the bus

S: time between buses



$$E[\text{Wait}] = \frac{E[S^2]}{2E[S]} >> E[S]$$

"Inspection Paradox"

M/G/1

$$E[T_Q] = \frac{\rho}{1-\rho} \cdot \frac{E[S^2]}{2E[S]}$$

High load leads to high wait

High job size variability leads to high wait

To drop load, we can increase server speed.

Q: What can we do to combat job size variability?

A: Smarter scheduling!

Scheduling in M/G/1

Well-studied scheduling policies:

FCFS (First-Come-First-Served, non-preemptive)

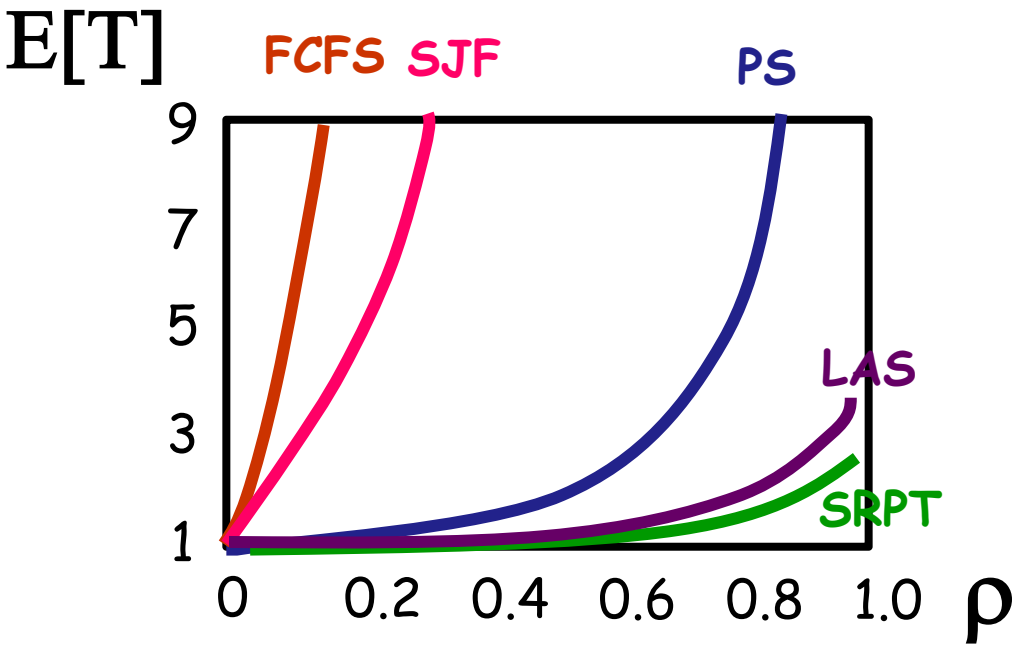
PS (Processor-Sharing, preemptive)

SJF (Shortest-Job-First, non-preemptive)

SRPT (Shortest-Remaining-Processing-Time, preemptive)

LAS (Least-Attained-Service First, preemptive)

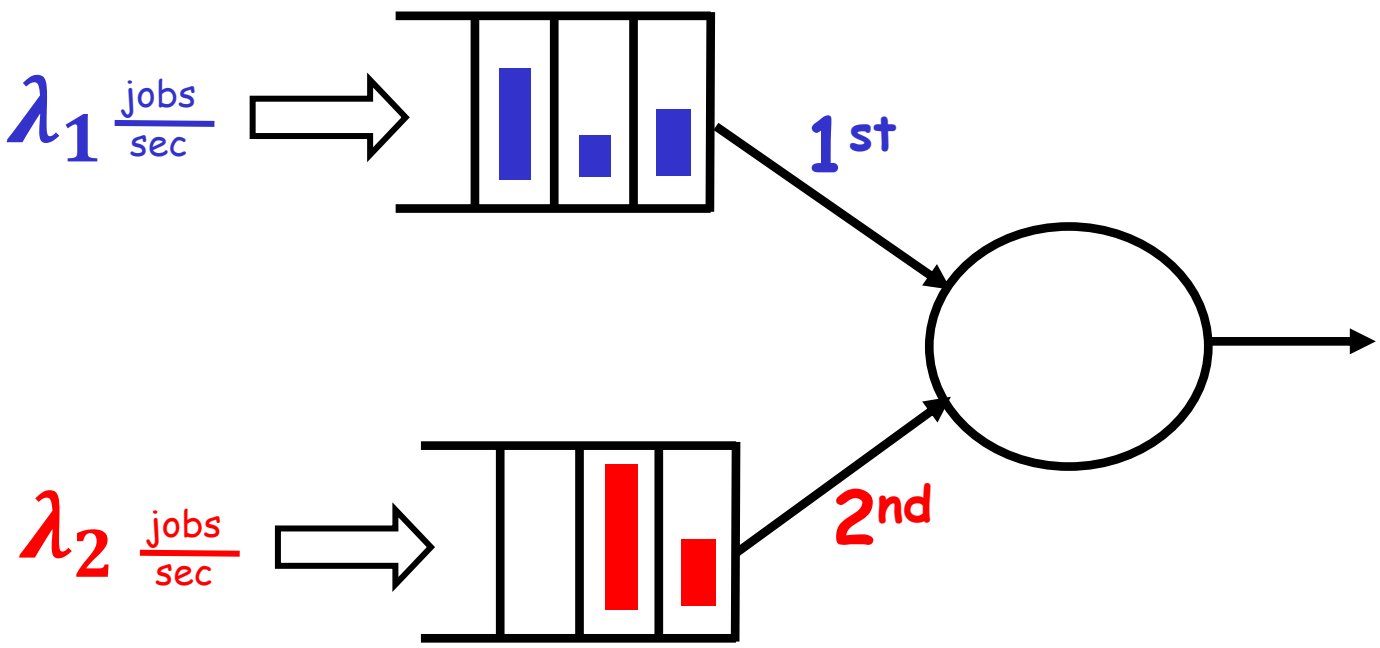
Scheduling in M/G/1



Under high job size variability

FCFS (First-Come-First-Served, non-preemptive)
PS (Processor-Sharing, preemptive)
SJF (Shortest-Job-First, non-preemptive)
SRPT (Shortest-Remaining-Processing-Time, preemptive)
LAS (Least-Attained-Service First, preemptive)

Priority Classes



<u>According to Ruth Williams (genetic networks)</u>:

- Jobs → molecules
- Server → enzyme
- Classes → protein species
- Reneging → dilution
- Class 1's load and variability can really affect class 2

Big Scheduling Breakthrough

[Scully, Harchol-Balter, Scheller-Wolf SIGMETRICS 2018]

The SOAP framework:

Enables first analysis of many previously intractable policies:

- **SERPT**: Prioritize jobs by Expected Remaining Size
- **Gittins**: Prioritize jobs by their Gittins Index
- **Discretized Policies**: Preemptions only at specific ages
- **Mixed Priority Classes**: Priority classes, where each class can have its own scheduling policy.

Outline

I. Basic Vocabulary

- Avg arrival rate, λ
- Avg service rate, μ
- Avg load, ρ
- Avg throughput, X
- Response time, T
- Waiting time, T_Q
- Exponential vs. Pareto/Heavy-tailed
- Poisson Process

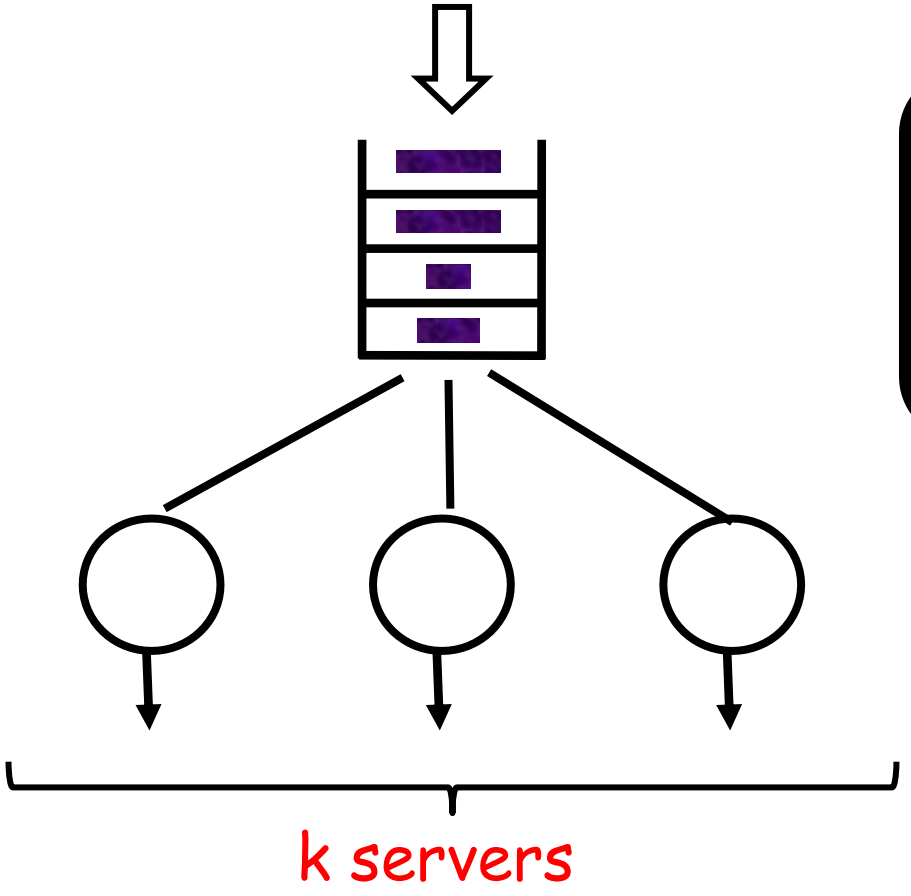
II. Single-server queues

- M/G/1 response time
- Inspection Paradox
- Effect of job size variability
- Effect of load
- Scheduling: FCFS, PS, SJF, LAS, SRPT
- Scheduling: Priority Classes
- Scheduling: SOAP Framework (New)

III. Multi-server queues

- Single shared queue, M/G/k
- Load balancing across queues
- Cycle stealing
- Replication of jobs (New)
- Multi-task jobs and fork-join (New)
- Networks of queues

M/G/k Model



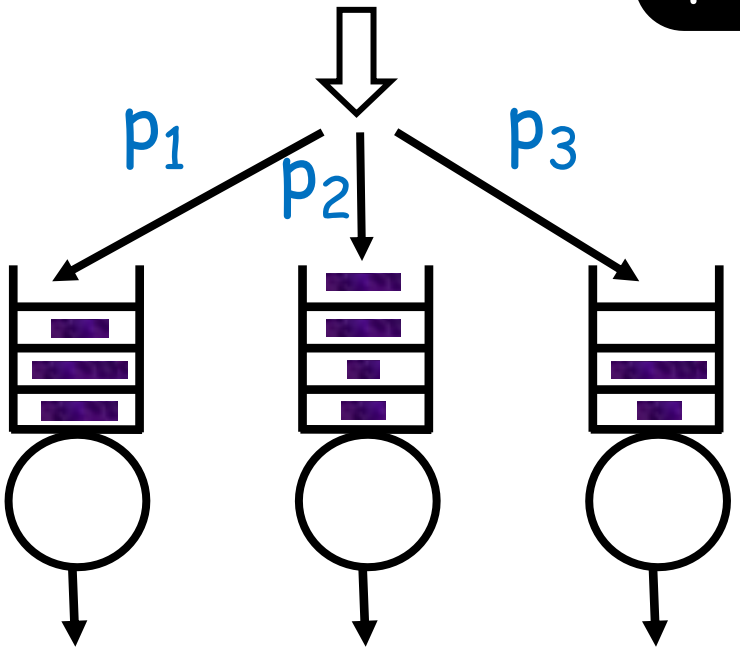
When server frees up, it grabs next available job

Q: How does M/G/k compare with M/G/1 at k-speed?

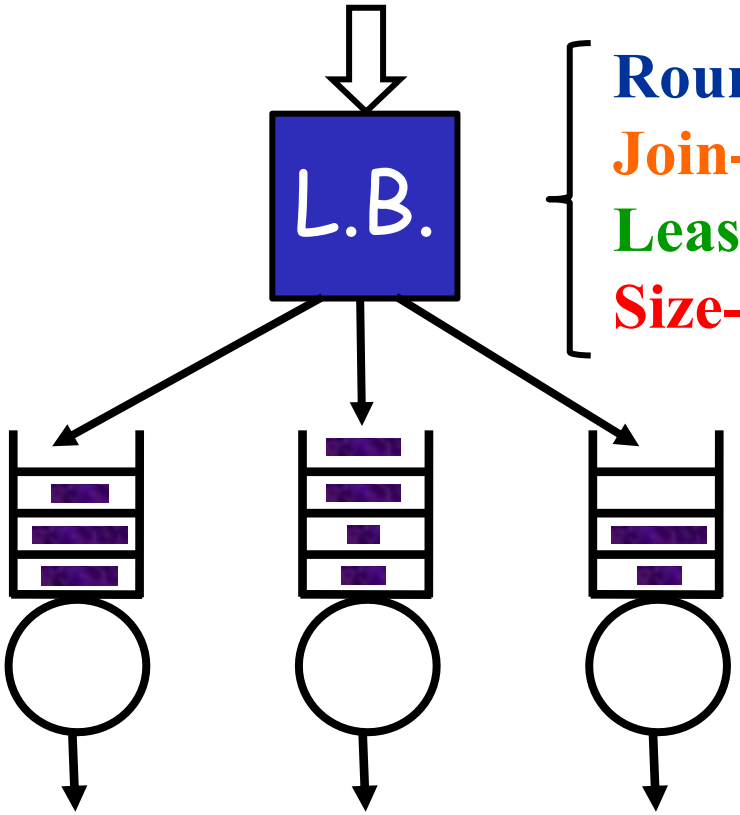
A: Both worse and better!

Load Balancing Model

Probabilistically split into independent queues.



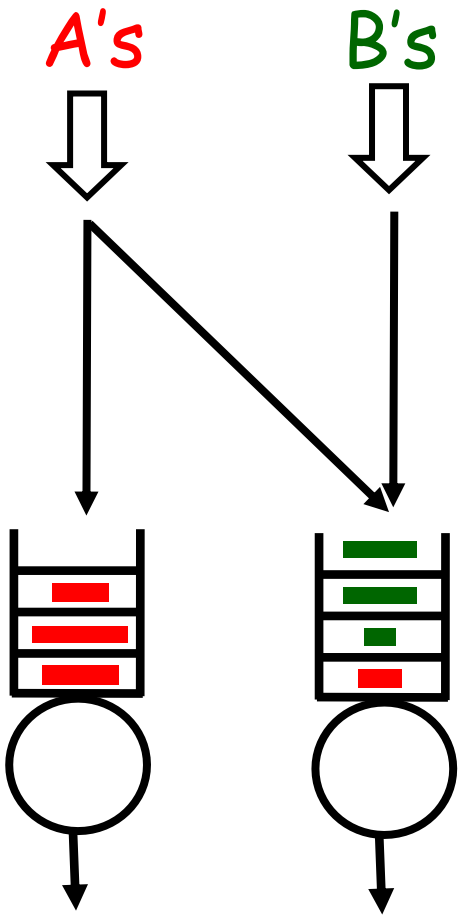
Load Balancing Model



Round-Robin
Join-Shortest-Queue
Least-Work-Left
Size-Interval Assignment

Smart Load Balancing → Much reduced mean response time

Cycle Stealing Model (N-model)



2D-inf Markov Chain

Only A's.

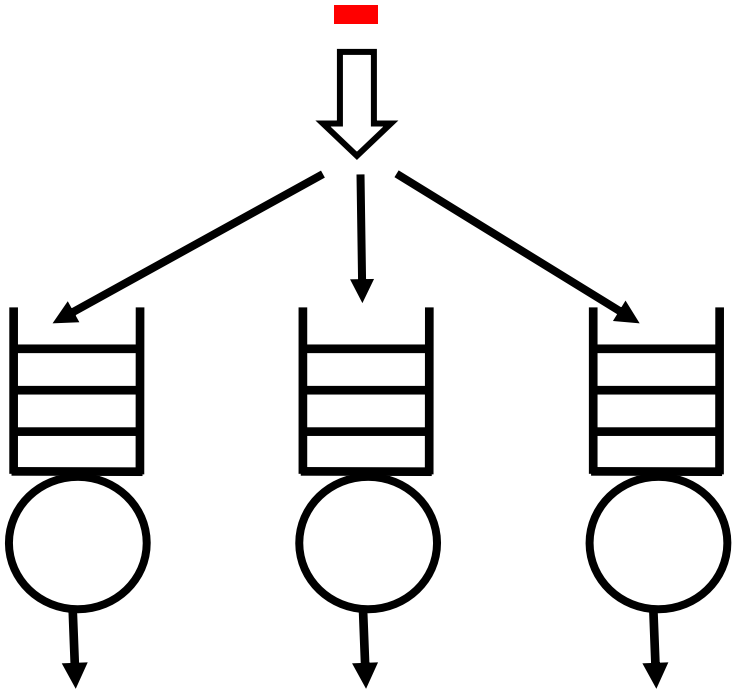
B's have priority, but if idle, then work on A's.

Replication Model

[Gardner, Harchol-Balter, Scheller-Wolf Transactions on Networking 2017]
[Gardner, Harchol-Balter, Scheller-Wolf Operations Research 2017]

Replication Model

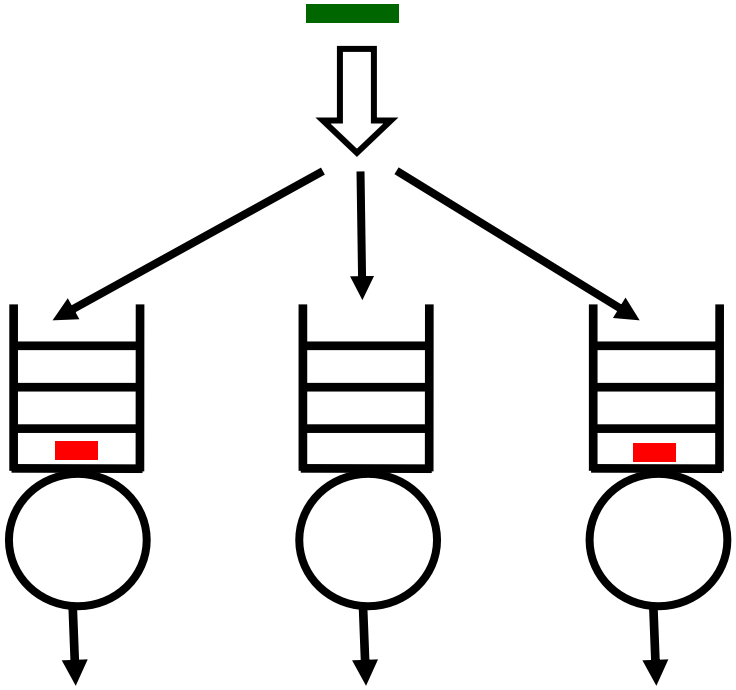
[Gardner, Harchol-Balter, Scheller-Wolf Transactions on Networking 2017]
[Gardner, Harchol-Balter, Scheller-Wolf Operations Research 2017]



Same job goes to multiple queues. Job is "done" as soon as first copy completes.

Replication Model

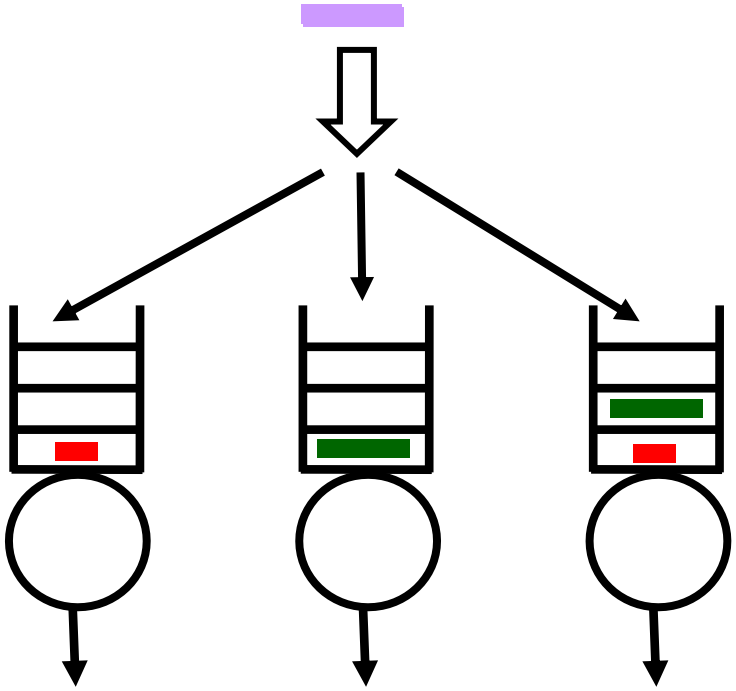
[Gardner, Harchol-Balter, Scheller-Wolf Transactions on Networking 2017]
[Gardner, Harchol-Balter, Scheller-Wolf Operations Research 2017]



Same job goes to multiple queues. Job is "done" as soon as first copy completes.

Replication Model

[Gardner, Harchol-Balter, Scheller-Wolf Transactions on Networking 2017]
[Gardner, Harchol-Balter, Scheller-Wolf Operations Research 2017]

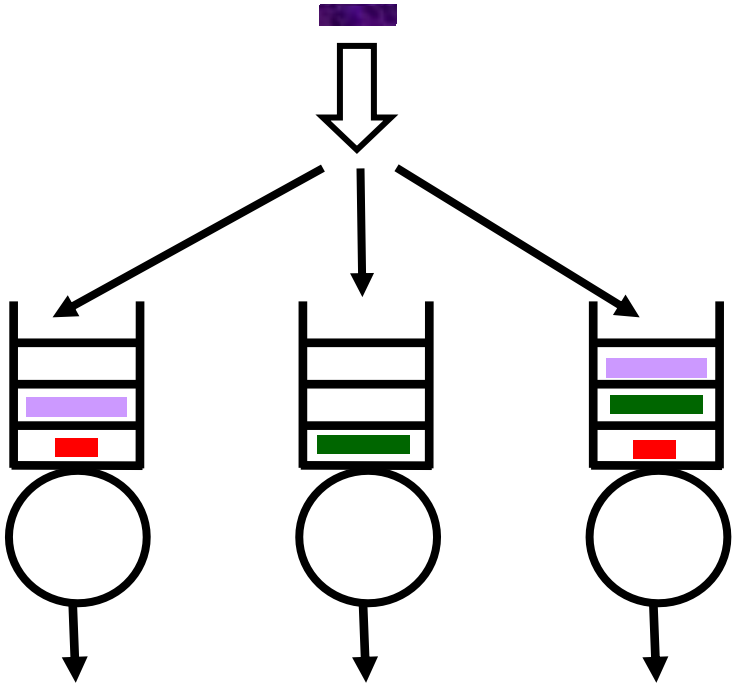


Same job goes to multiple queues. Job is "done" as soon as first copy completes.

Replication Model

[Gardner, Harchol-Balter, Scheller-Wolf Transactions on Networking 2017]
[Gardner, Harchol-Balter, Scheller-Wolf Operations Research 2017]

nD-inf Markov Chain

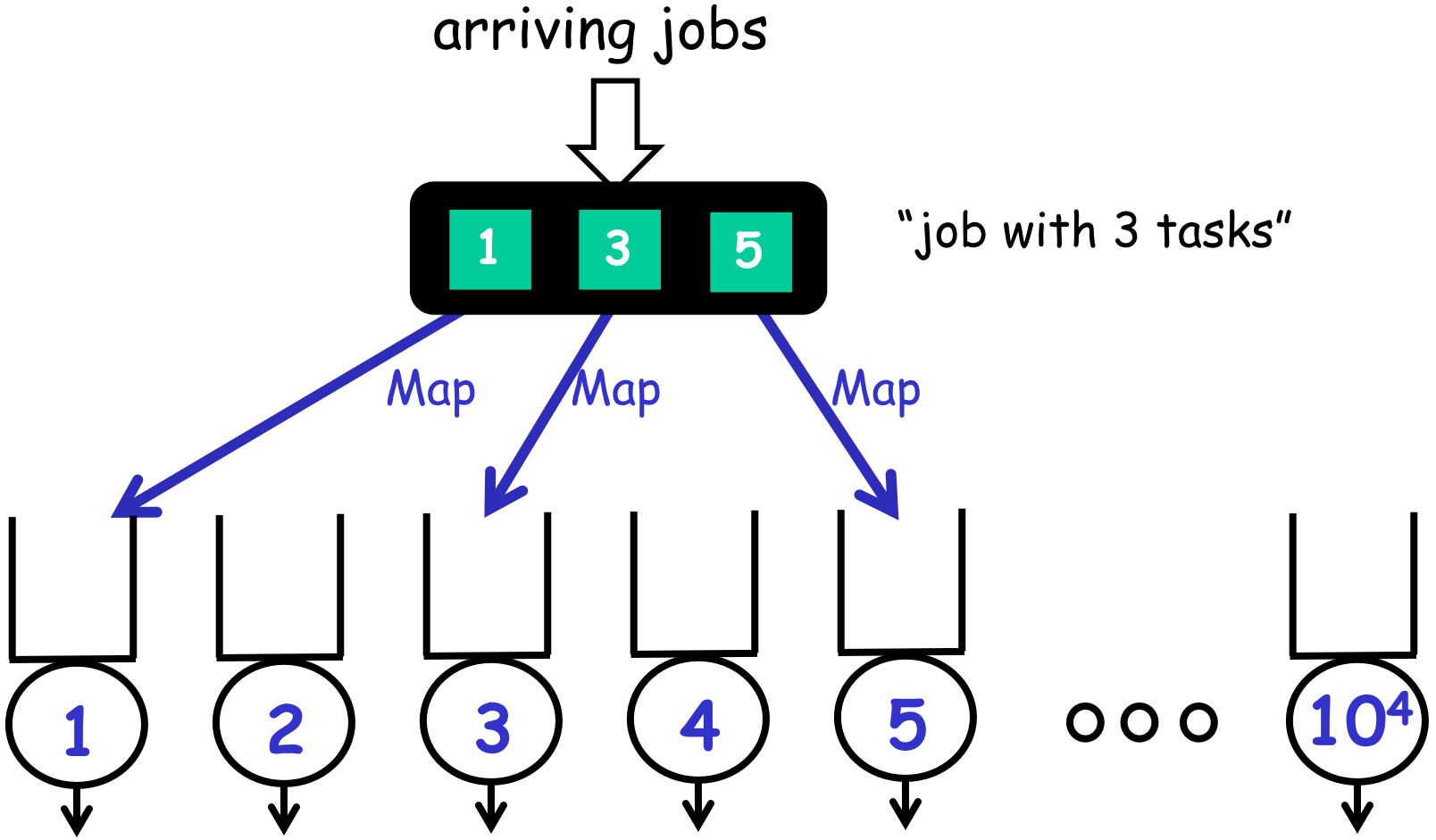


Same job goes to multiple queues. Job is "done" as soon as first copy completes.

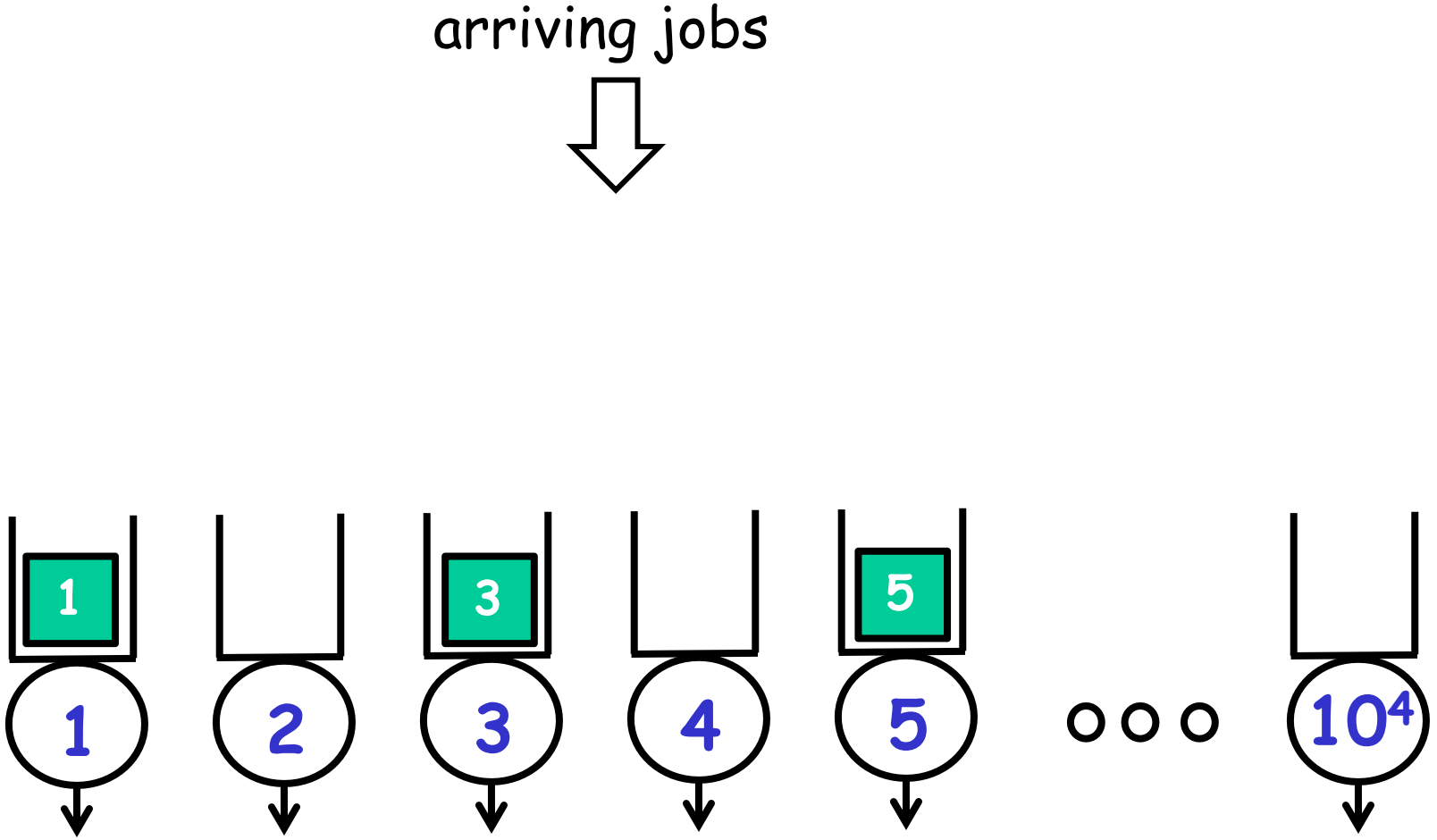
Replication Tradeoff:

+ Lower response time because only need first completion.
+ Higher response time due to extra load.

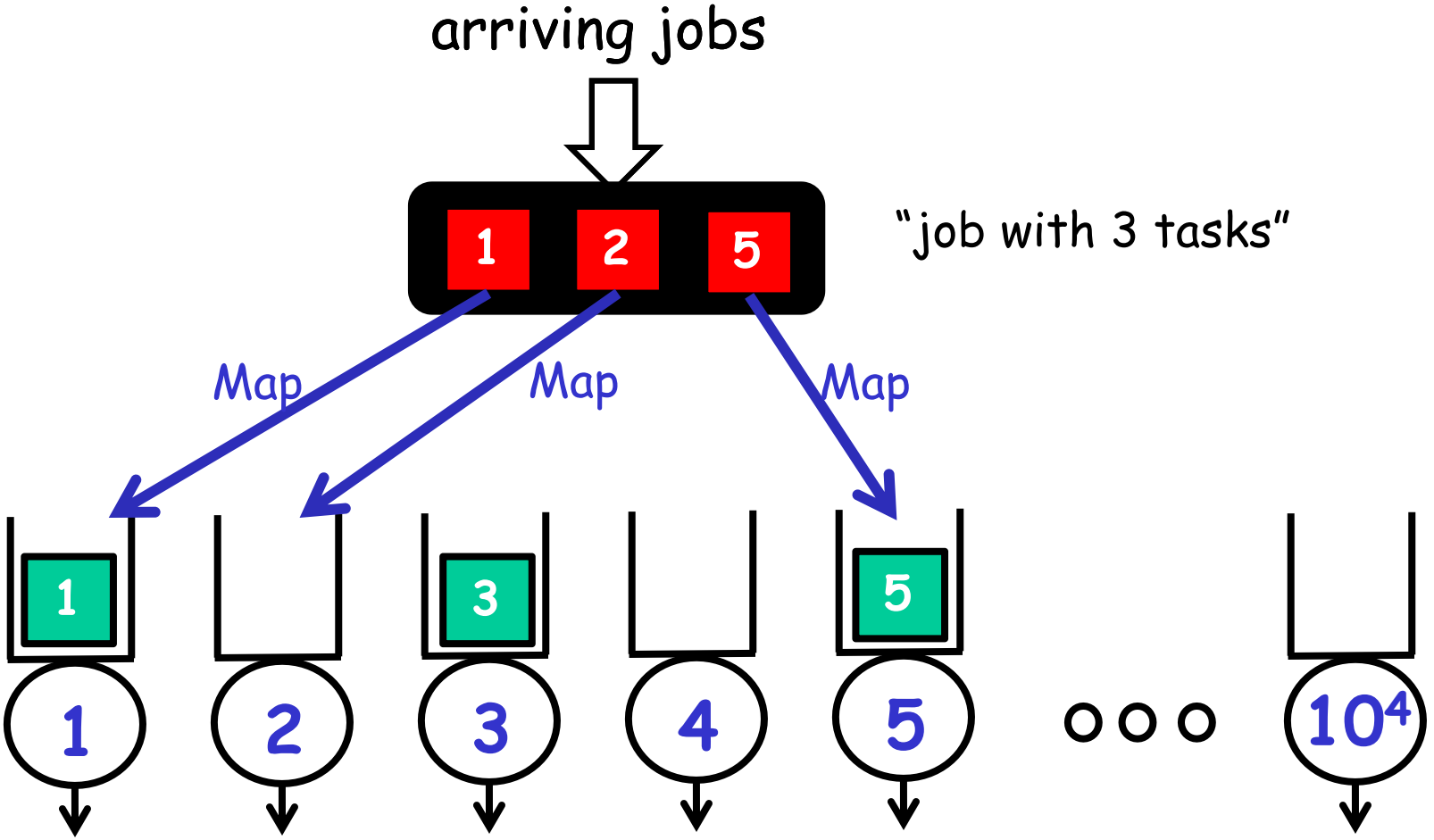
Multi-task Job Model



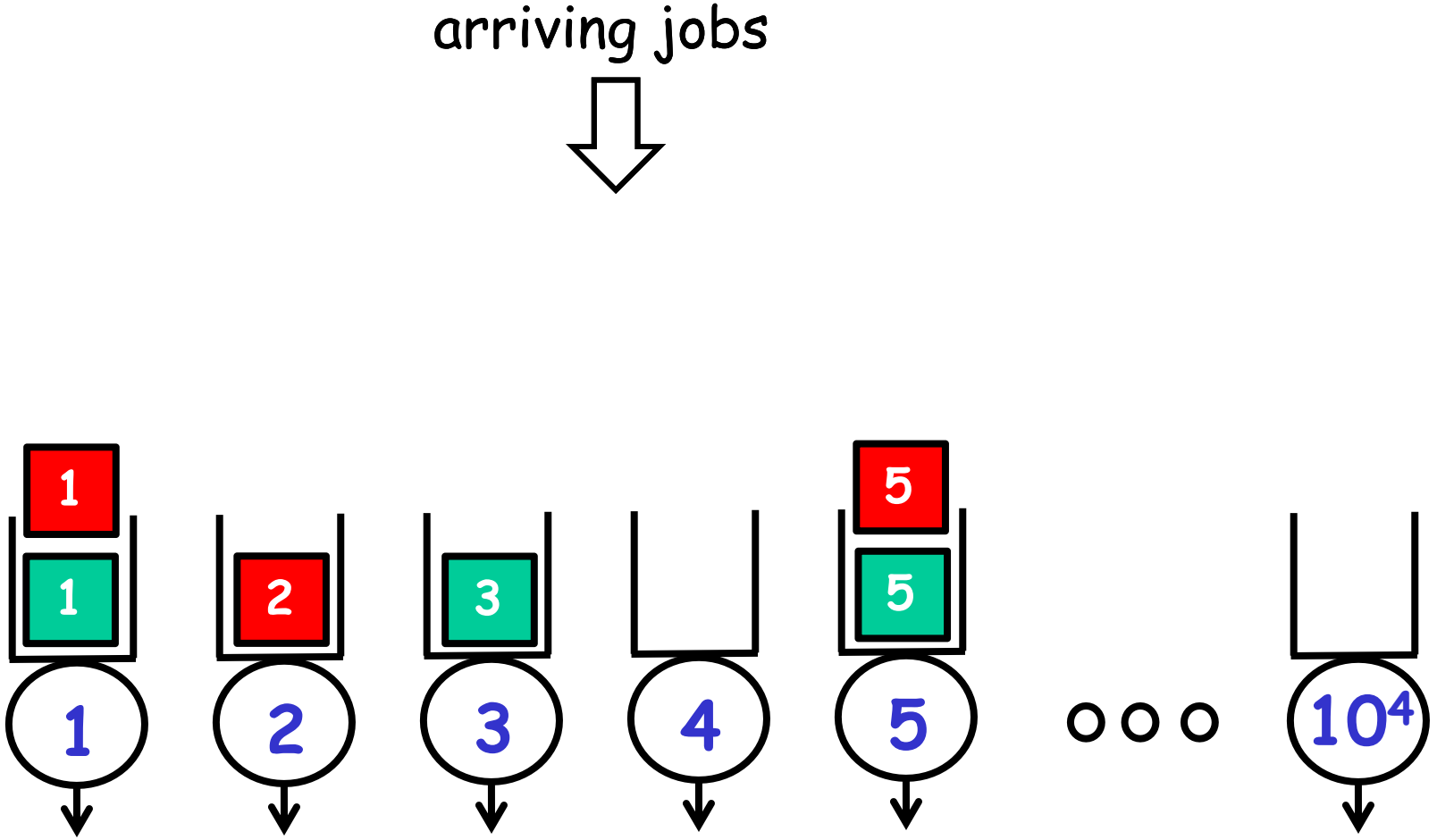
Multi-task Job Model



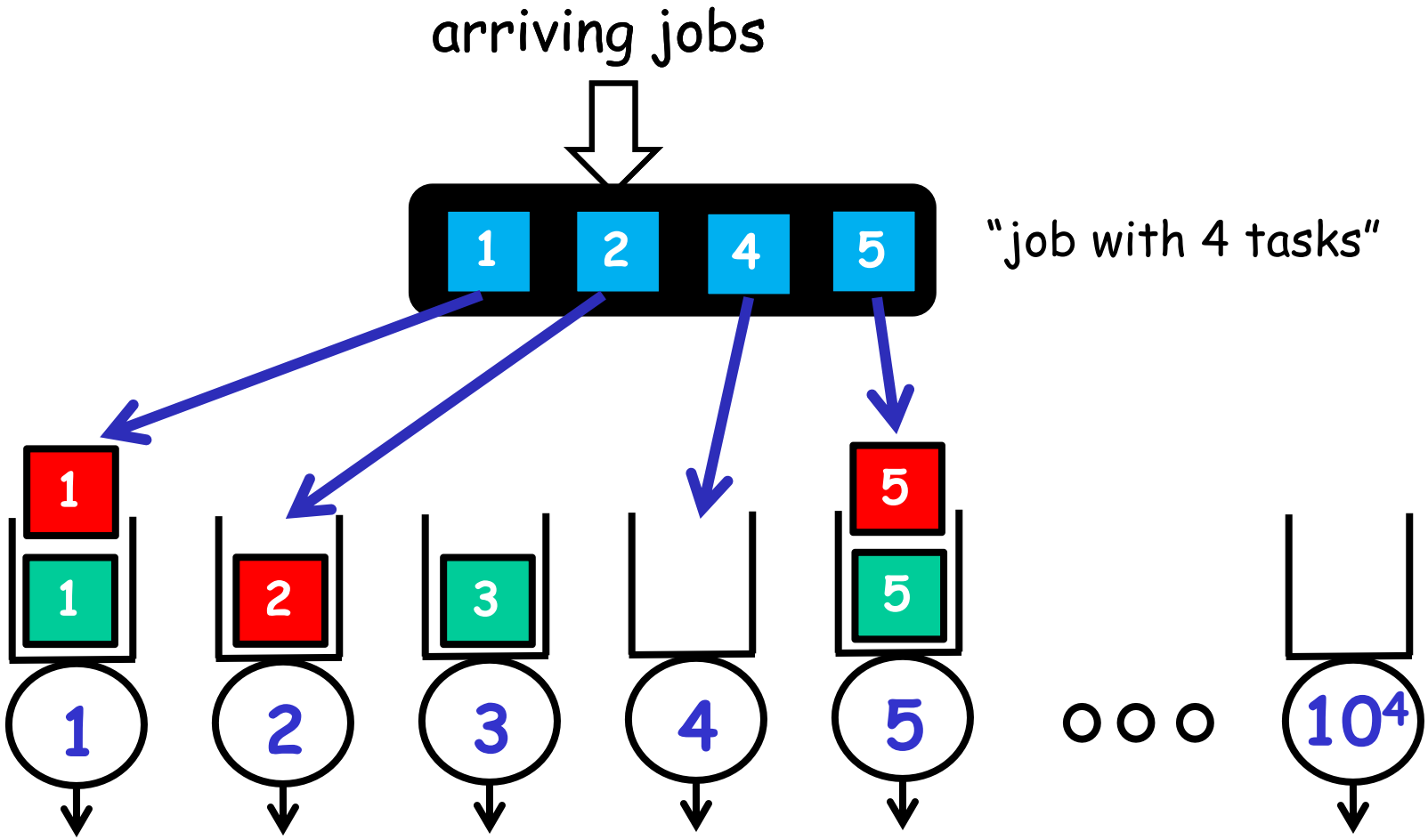
Multi-task Job Model



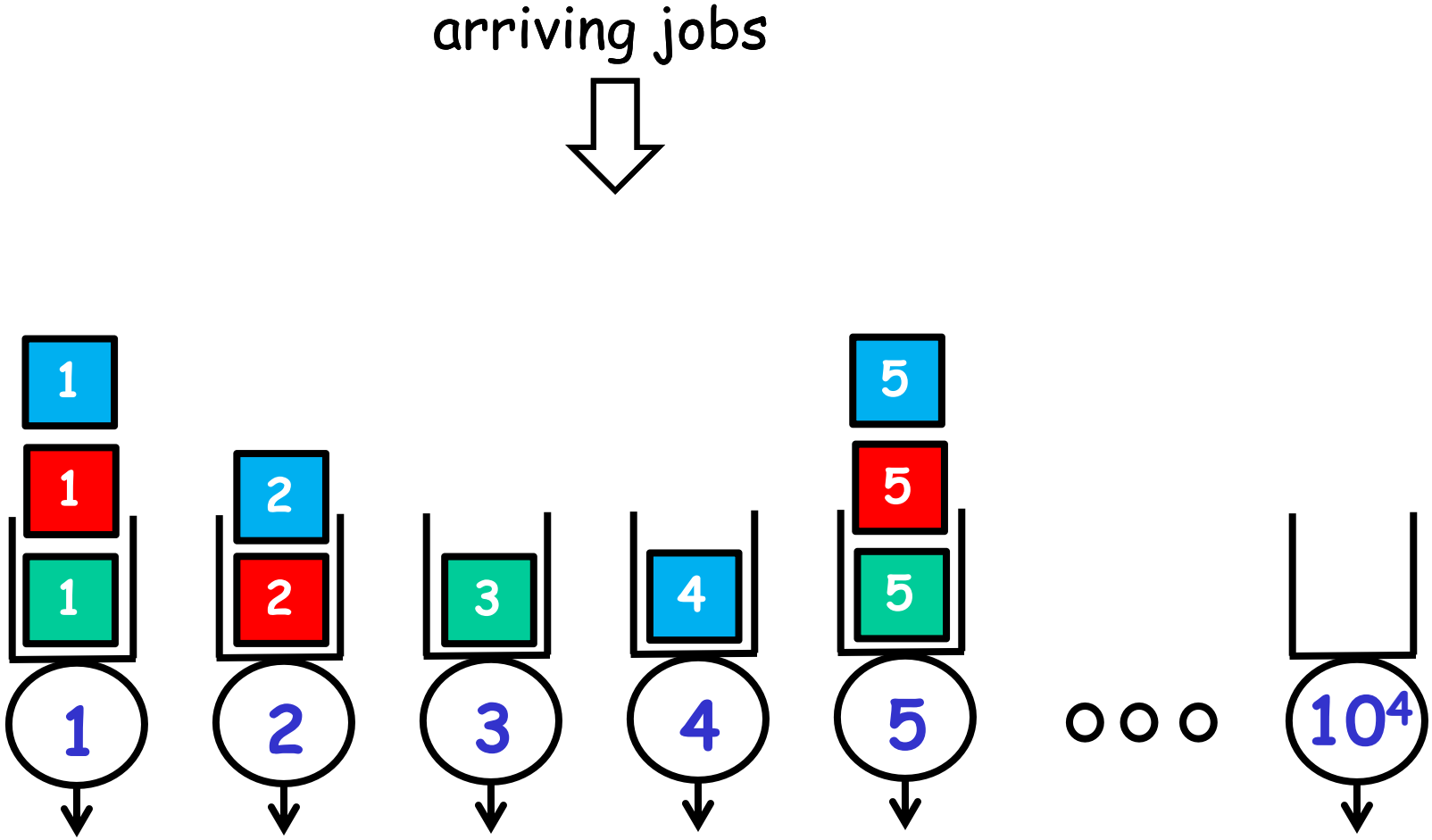
Multi-task Job Model



Multi-task Job Model



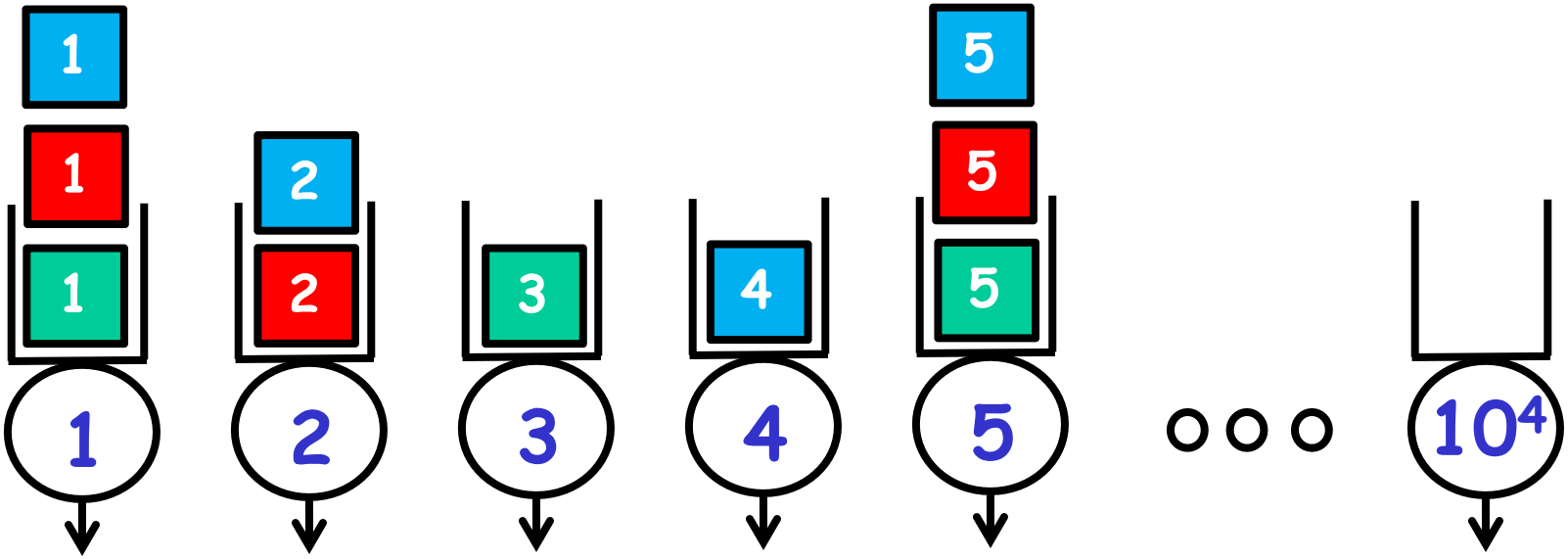
Multi-task Job Model



Multi-task Job Model

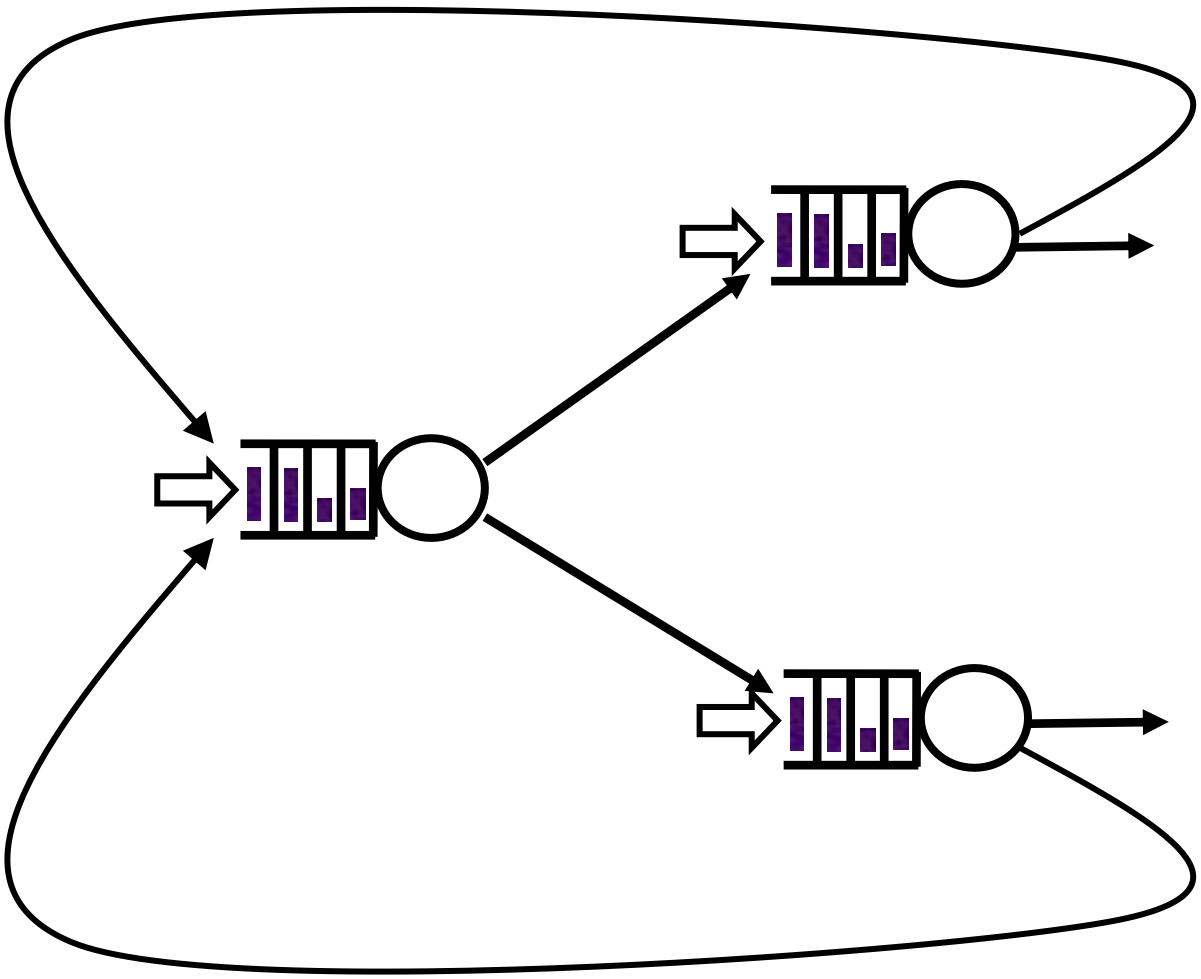
arriving jobs

Job not done until ALL its tasks are done



"Limited Fork-Join"
See [Wang, Harchol-Balter, Jiang, Scheller-Wolf, Srikant, 2018].

Networks of Queues Model



nD-inf
Markov
Chain

Conclusion

I. Basic Vocabulary

- Avg arrival rate, λ
- Avg service rate, μ
- Avg load, ρ
- Avg throughput, X
- Response time, T
- Little's Law
- Exponential vs. Pareto/Heavy-tailed
- Poisson Process

II. Single-server queues

- M/G/1 response time
- Inspection Paradox
- Effect of job size variability
- Effect of load
- Scheduling: FCFS, PS, SJF, LAS, SRPT
- Scheduling: Priority Classes
- Scheduling: SOAP Framework (New)

III. Multi-server queues

- Single shared queue, M/G/k
- Load balancing across queues
- Cycle stealing
- Replication of jobs (New)
- Multi-task jobs and fork-join (New)
- Network of queues

THANK YOU!

www.cs.cmu.edu/~harchol/