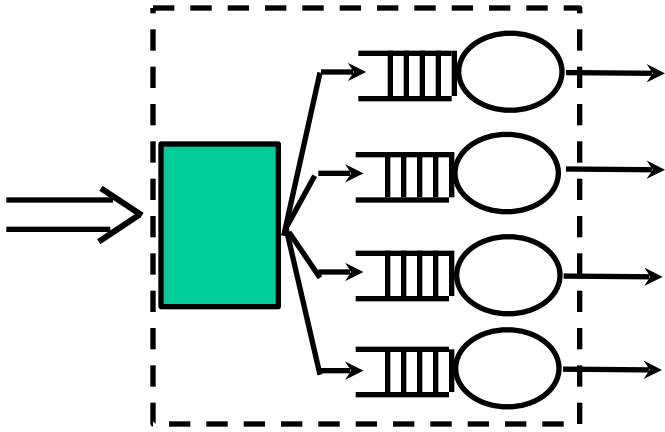
Scheduling in Server Farms

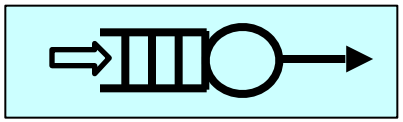
Mor Harchol-Balter
Associate Department Head
Computer Science Dept
Carnegie Mellon University

harchol@cs.cmu.edu

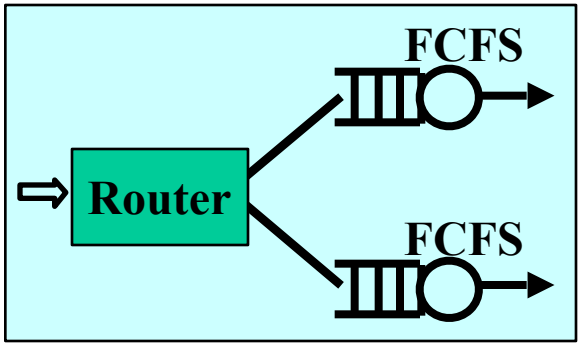


Outline

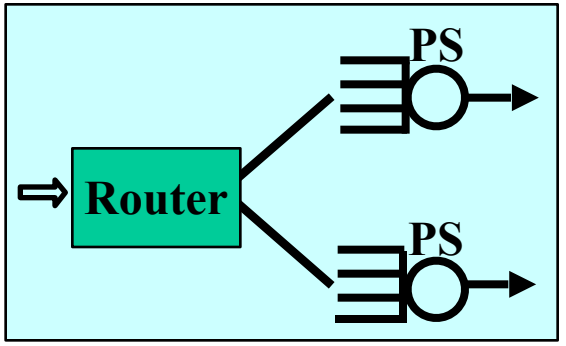
I. *Scheduling a single-server*



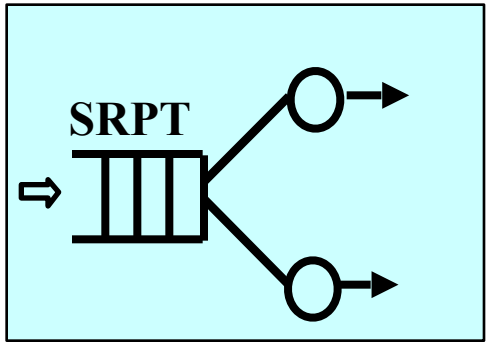
II. *Supercomputing*



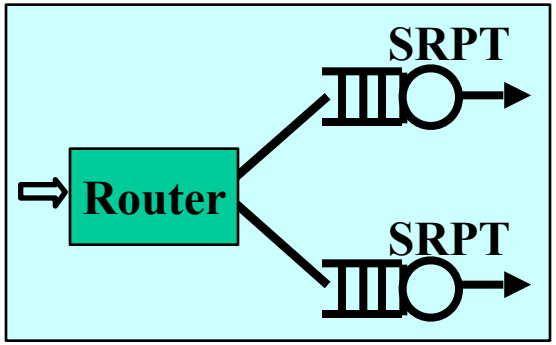
III. *Web server farm model*



IV. Towards Optimality …

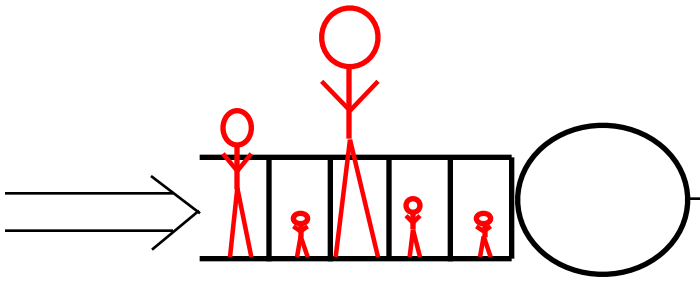


&



Metric: Mean Response Time, E[T]

Single Server Model (M/G/1)



Load $\rho = \lambda E[X] < 1$

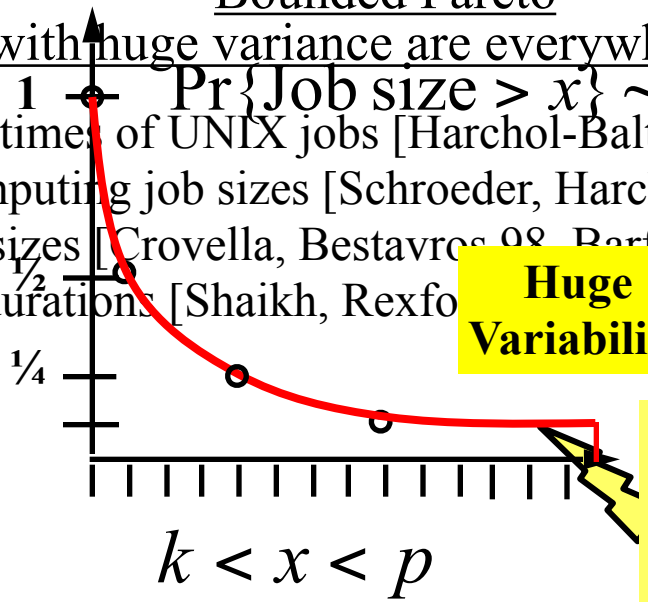
X: job size
(service requirement)

Bounded Pareto

Job sizes with huge variance are everywhere in CS:

- CPU Lifetimes of UNIX jobs [Harchol-Balter, Downey 96]
- Supercomputing job sizes [Schroeder, Harchol-Balter 00]
- Web file sizes [Crovella, Bestavros 98, Barford, Crovella 98]
- IP Flow durations [Shaikh, Rexfo

$$\Pr\{\text{Job size} > x\} \sim \frac{1}{x}$$



Huge Variability

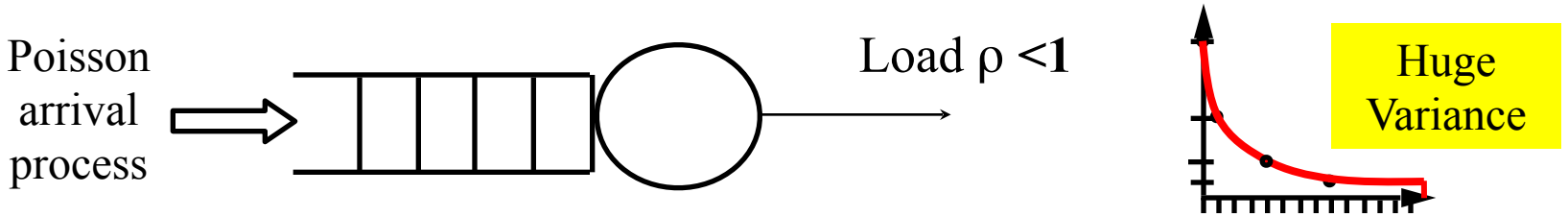
D.F.R.

Top-heavy: top 1% jobs make up half load

$$C_X^2 = \frac{Var(X)}{E[X]^2}$$

$8 < C_X^2 < 50$ is typical

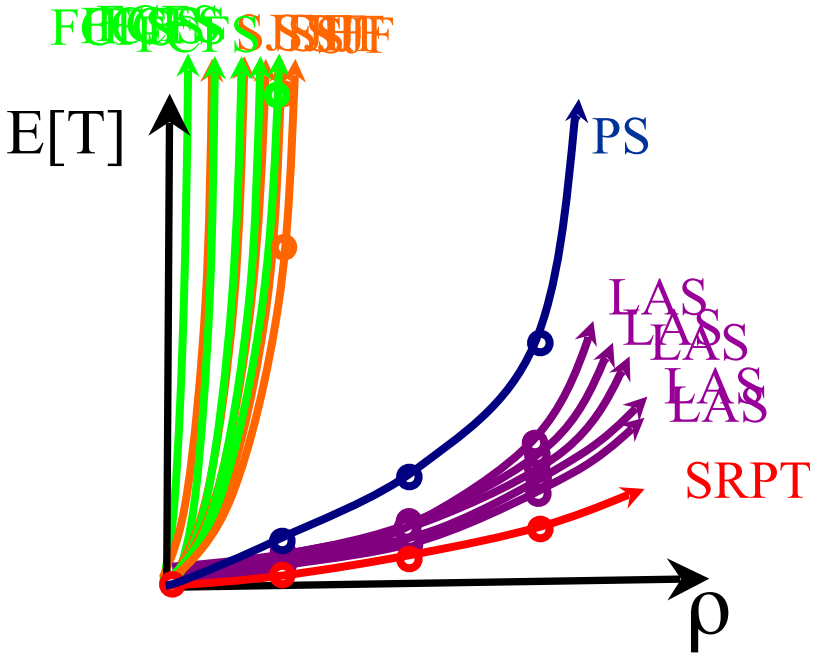
Scheduling Single Server (M/G/1)



Question: Order these scheduling policies for mean response time, E[T]:

1. FCFS (First-Come-First-Served, non-preemptive)
2. PS (Processor-Sharing, preemptive)
3. SJF (Shortest-Job-First, a.k.a. SPT, non-preemptive)
4. SRPT (Shortest-Remaining-Processing-Time, preemptive)
5. LAS (Least Attained Service, a.k.a., FB, preemptive)

Answers

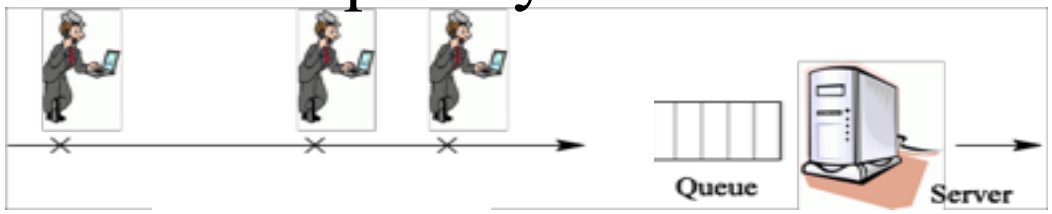


$C^2 =$

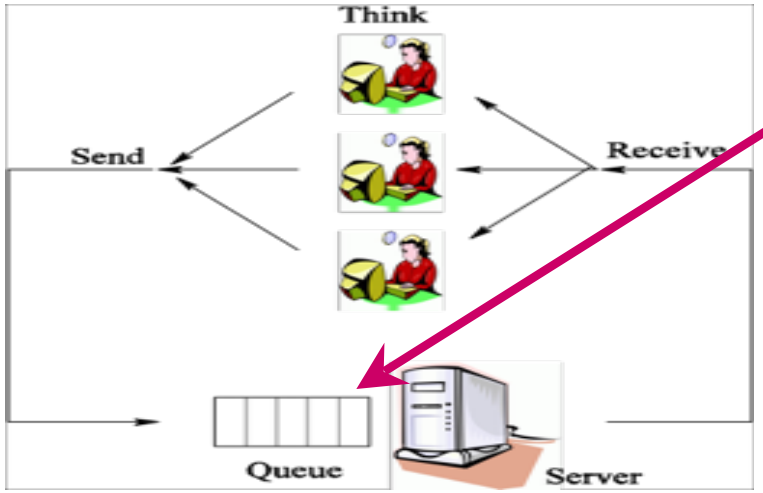
Bounded Pareto job sizes

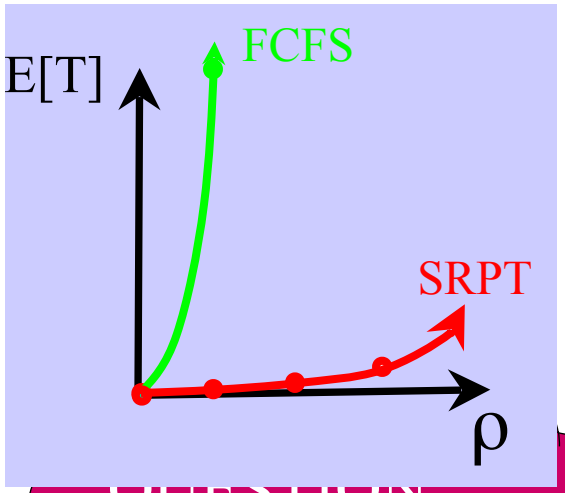
Closed vs. Open Systems

Open System

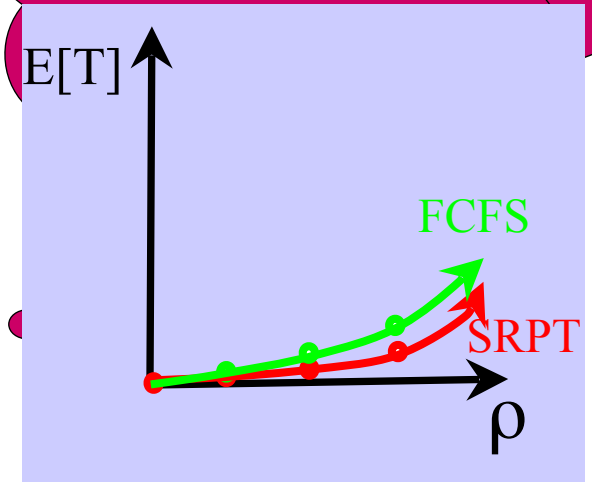


Closed System





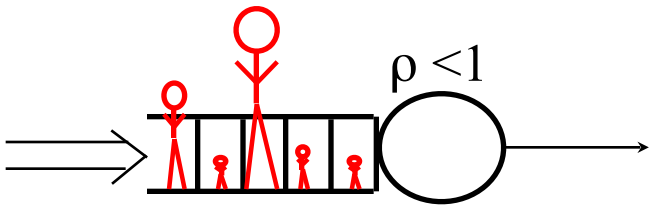
QUESTION:
What's the effect of



[Schroeder, Wierman, Harchol-Balter, NSDI 06]

Summary Single-Server

Single-server system



X: job size -- highly-variable

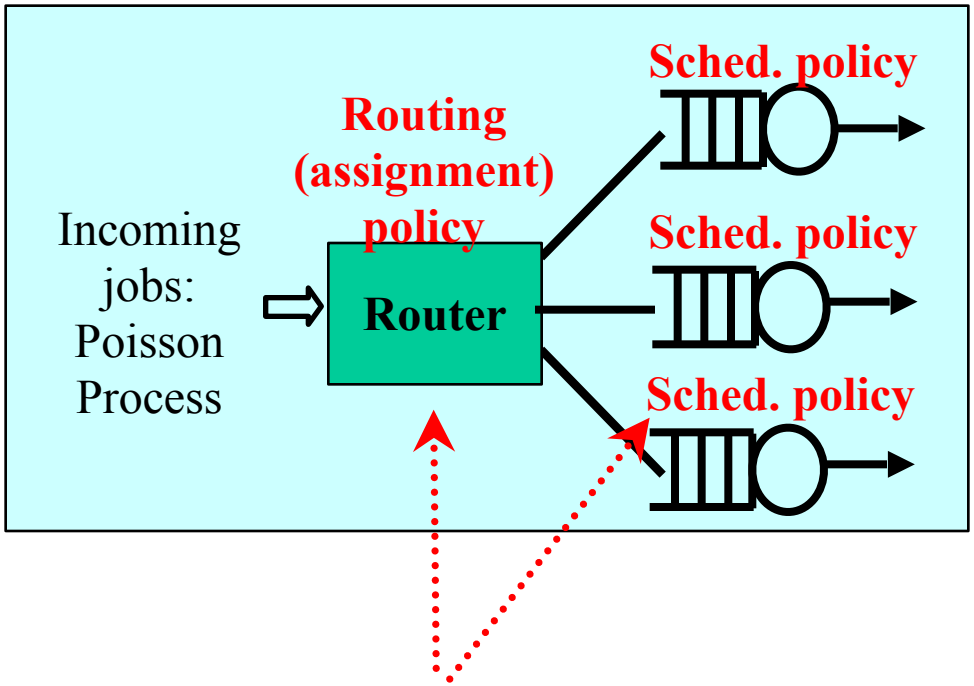
LESSONS LEARNED:

- ❑ Smart scheduling greatly improves mean response time.
- ❑ Variability of job size distribution is key.

Multiserver Model

Server farms:

+ Cheap
+ Scalable capacity

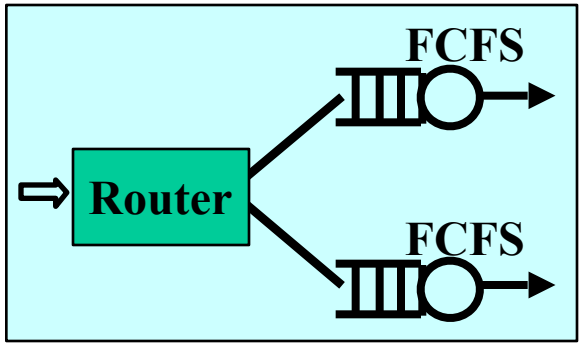


(Sometimes scheduling policy is fixed – legacy system)

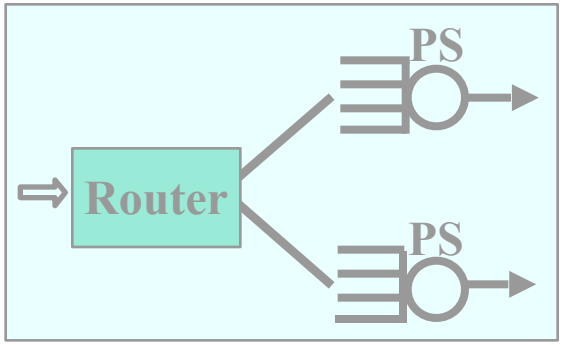
Outline

I. *Review of scheduling in single-server*

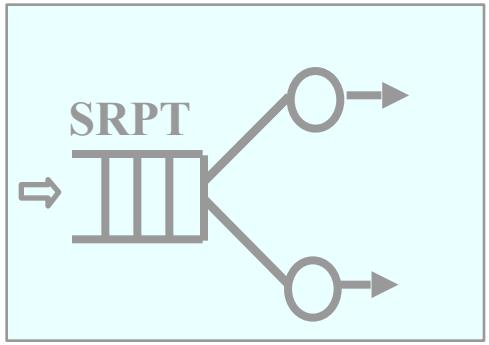
II. ***Supercomputing***



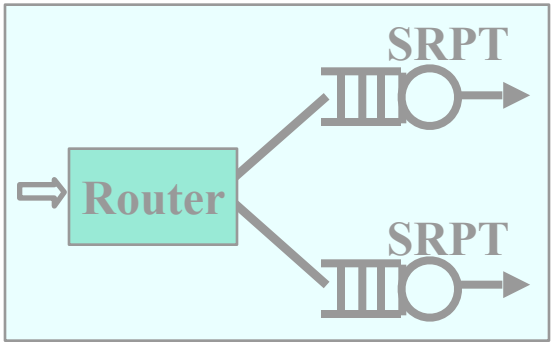
III. *Web server farm model*



IV. Towards Optimality …

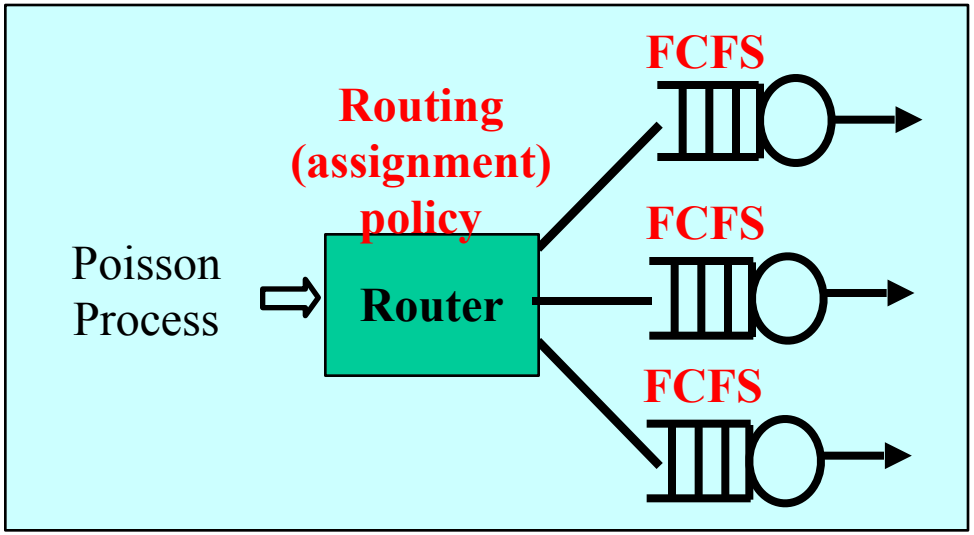


&



Metric: Mean Response Time, E[T]

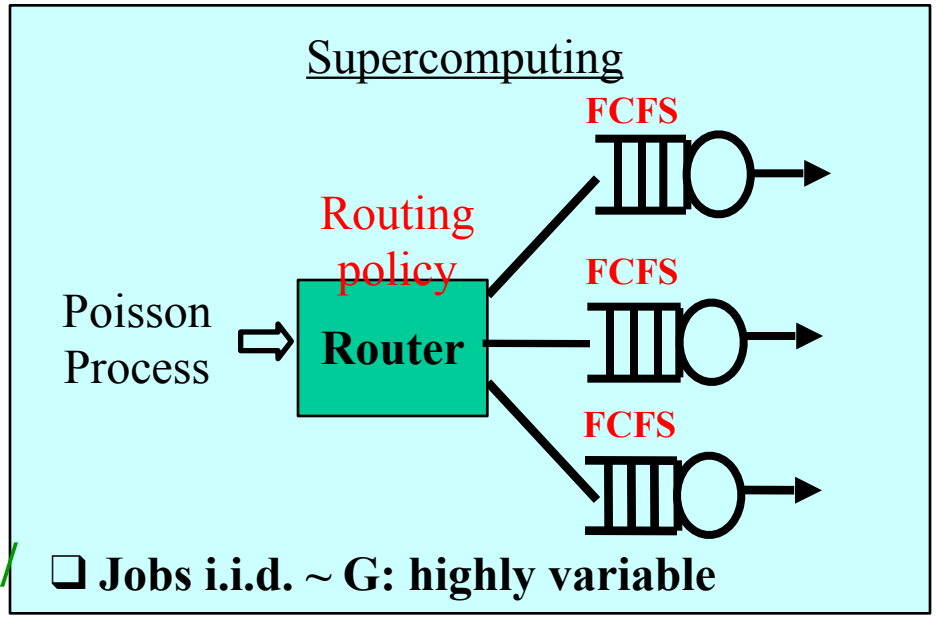
Supercomputing Model



- Jobs are not preemptible.
- Jobs processed in FCFS order.
- Assume hosts are identical.
- Jobs i.i.d. ~ G: highly variable size distribution.
- Size may or may not be known. Initially assume known.

Q: Compare Routing Policies for E[T]?

1. **Round-Robin**
2. **Join-Shortest-Queue**
   Go to host w/ fewest # jobs.
3. **Least-Work-Left**
   Go to host with least total work.
4. **Central-Queue-FCFS (M/G/k/FCFS)**
   Host grabs shortest job when free.
5. **Central-Queue-Shortest-Job (M/G/k/SJF)**
   Host grabs shortest job when free.
6. **Size-Interval Splitting**
   Jobs are split up by size among hosts.



A: Size-Interval Splitting: best

High E[T]

1. **Round-Robin**

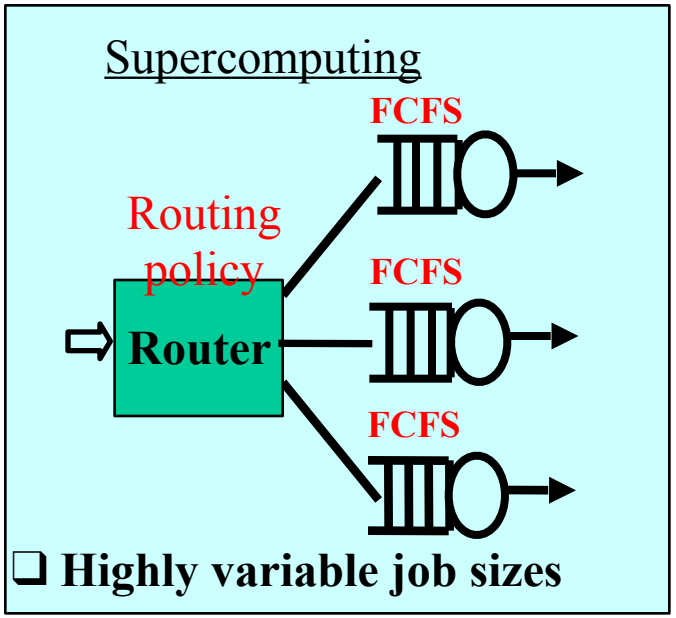
2. **Join-Shortest-Queue**
   Go to host w/ fewest # jobs.

3. **Least-Work-Left, equivalent to M/G/k/FCFS**
   Go to host with least total work.

4. **Central-Queue-Shortest-Job (M/G/k/SJF)**
   Host grabs shortest job when free.

5. **Size-Interval Splitting**
   Jobs are split up by size among hosts.

Low E[T]



[Harchol-Balter, Crovella, Murta, JPDC 99]

Routing Policies: Remarks

High E[T]

Low E[T]

1. **Round-Robin**

2. **Join-Shortest-Queue**
   Go to host w/ fewest # jobs.

3. **Least-Work-Left, equivalent to M/G/k/FCFS**
   Go to host with least total work.

4. **Central-Queue-Shortest-Job (M/G/k/SJF)**
   Host grabs shortest job when free.

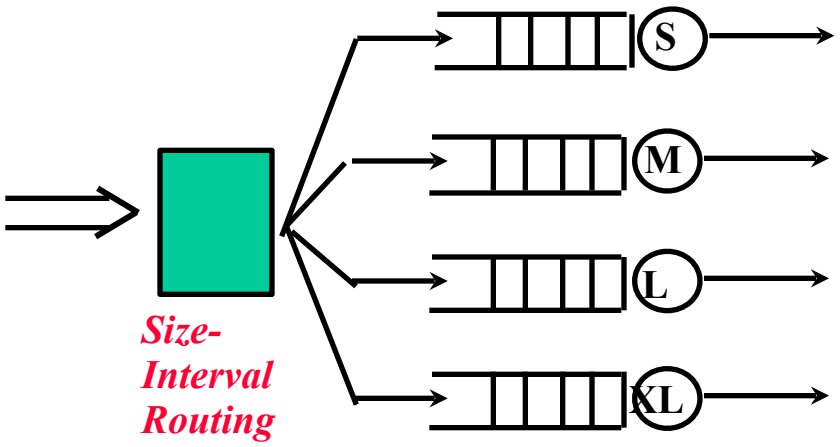
5. **Size-Interval Splitting**
   Jobs are split up by size among hosts.

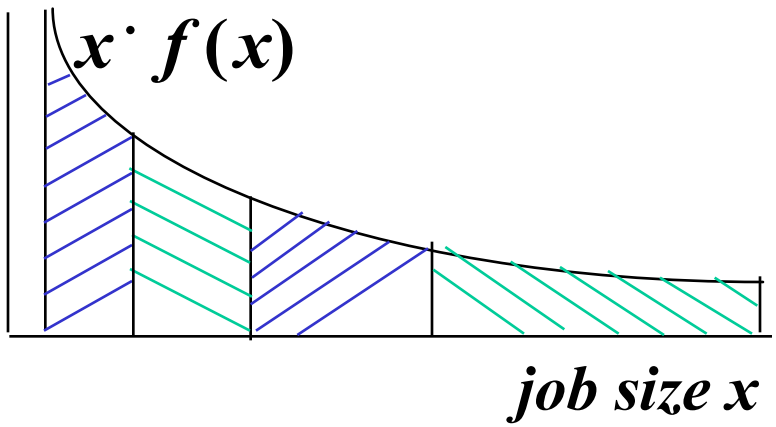
Central-Queue:
+ Good utilization of servers.
+ Some isolation for smalls

Size-Interval WAY Better!
- Worse utilization of servers.
+ Great isolation for smalls!

[Harchol-Balter, Crovella, Murta, JPDC 99].

Size-Interval Splitting

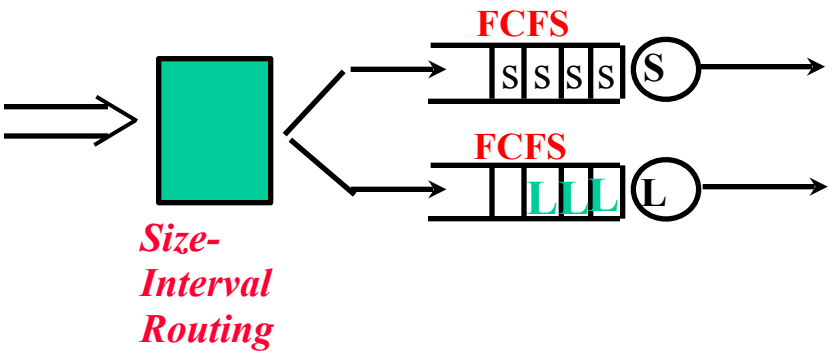


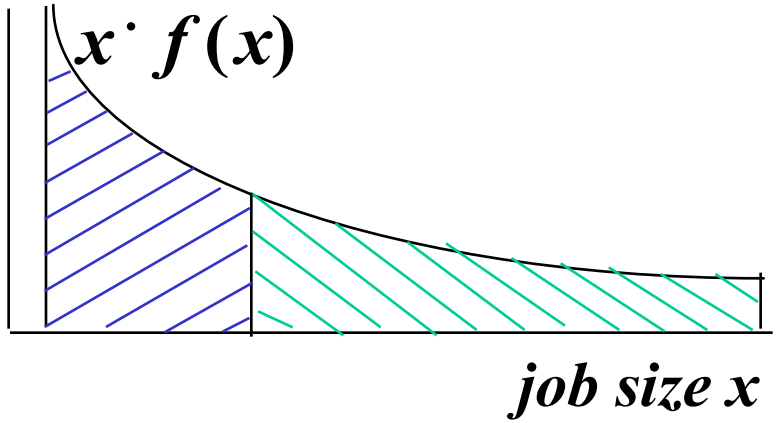


Question: How to choose the size cutoffs?
"To Balance Load or Not to Balance Load?"

$$\int_S xf(x)dx \stackrel{?}{=} \int_M xf(x)dx \stackrel{?}{=} \int_L xf(x)dx \stackrel{?}{=} \int_{XL} xf(x)dx$$

Size-Interval Splitting





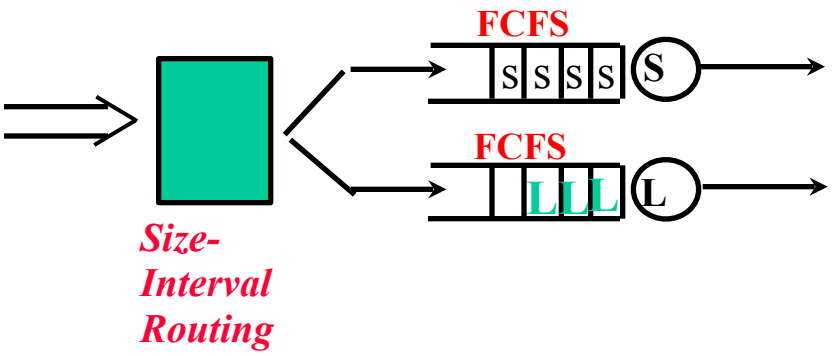
Answer: Recent Research for case of Bounded Pareto

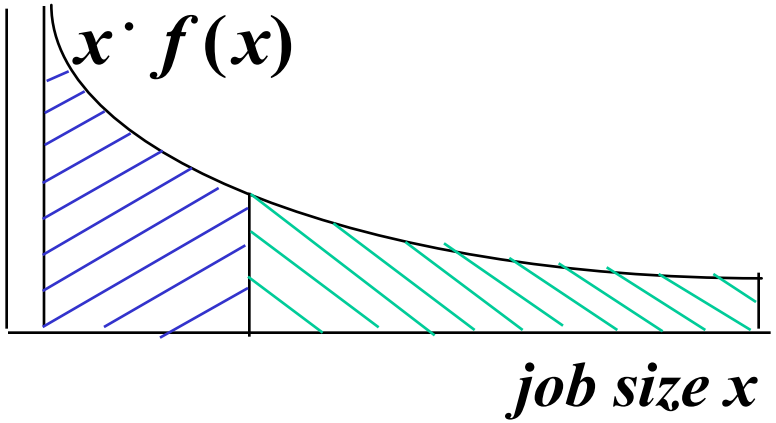
job size: $\Pr\{X>x\} \sim x^{-\alpha}$

$\alpha<1$	$\alpha=1$	$\alpha>1$
UNBALANCE favor smalls	**BALANCE LOAD**	**UNBALANCE favor larges**

[Harchol-Balter,Vesilo, 08]

Beyond Size-Interval Splitting

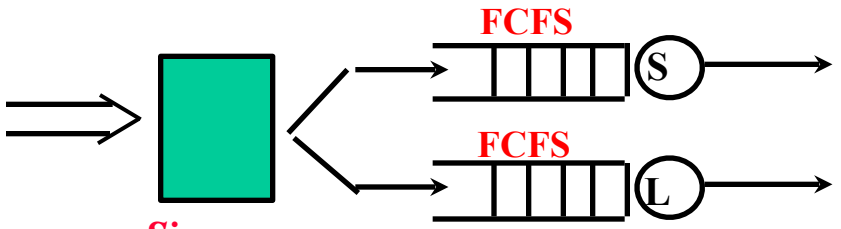




Q: Is Size-Interval Splitting as good as it gets?

Size-Interval Splitting with Stealing

Answer: Allow Cycle Stealing!



Size-Interval Routing with Cycle Stealing

Send Shorts Here

Send Longs Here.
But, if idle, send Short.

- Gain to Shorts is very high
- Pain to Longs is very small.

Cycle Stealing analysis very hard: Fayolle, Iasnogorodski, Konheim, Meilijson, Melkman, Cohen, Boxma, van Uitert, Jelenkovic, Foley, McDonald, Harrison, Borst, Williams …

New easy approach: Dimensionality Reduction 2D→1D
[Harchol-Balter, Osogami, Scheller-Wolf, Squillante SPAA03]

What if Don't Know Job Size?

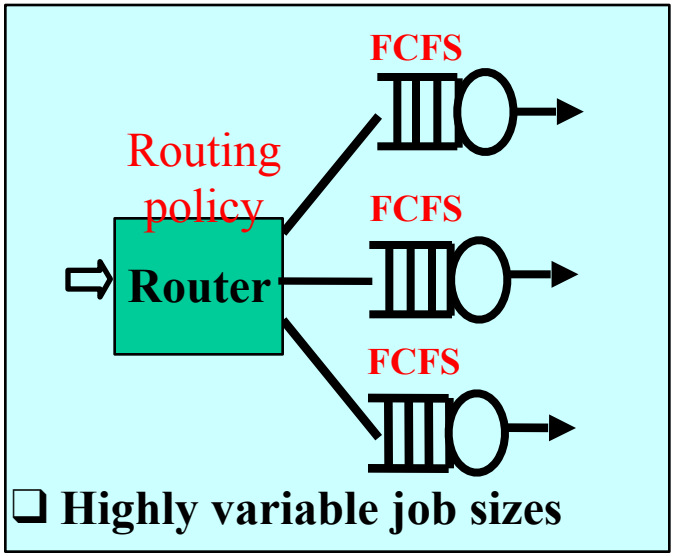
1. **Round-Robin**

2. **Join-Shortest-Queue**
   Go to host w/ fewest # jobs.

3. **Least-Work-Left, equivalent to M/G/k/FCFS**
   Go to host with least total work.

4. **Central-Queue-Shortest-Job (M/G/k/SJF)**
   Host grabs shortest job when free.

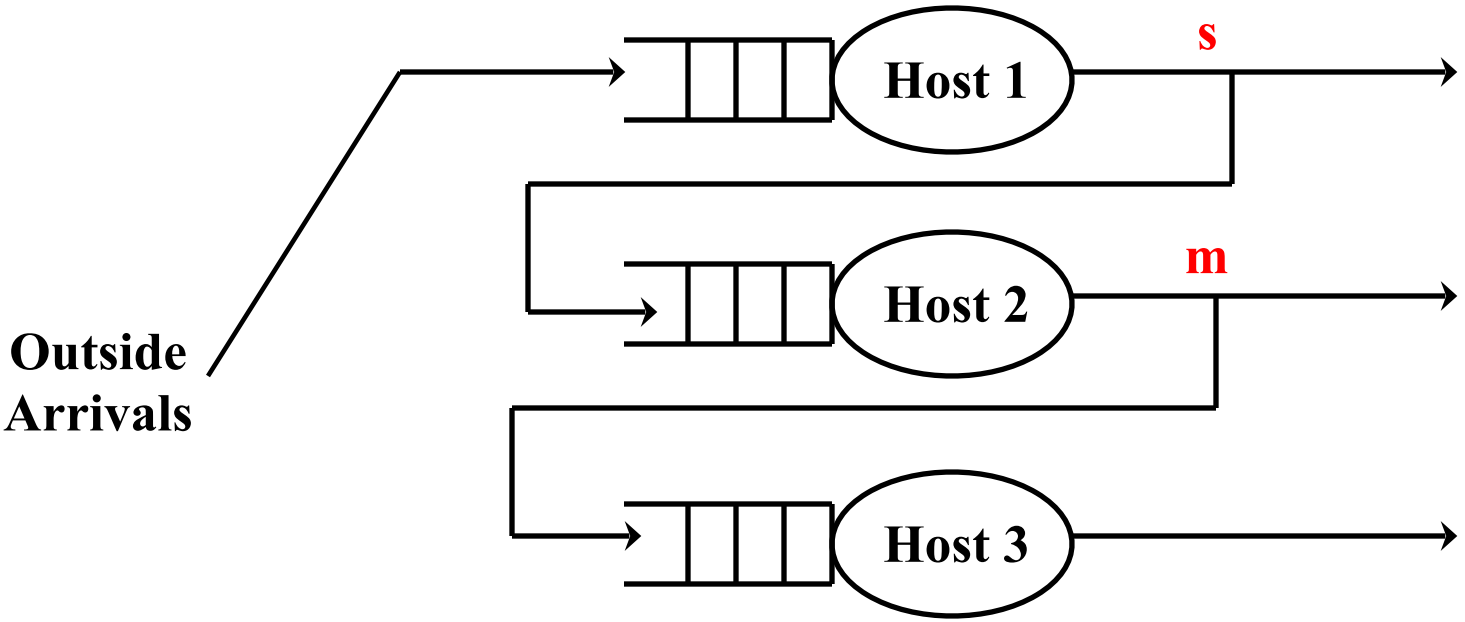
5. **Size-Interval Splitting**
   Jobs are split up by size among hosts.



❑ Highly variable job sizes

Q: What can we do to minimize E[T] when don't know job size?

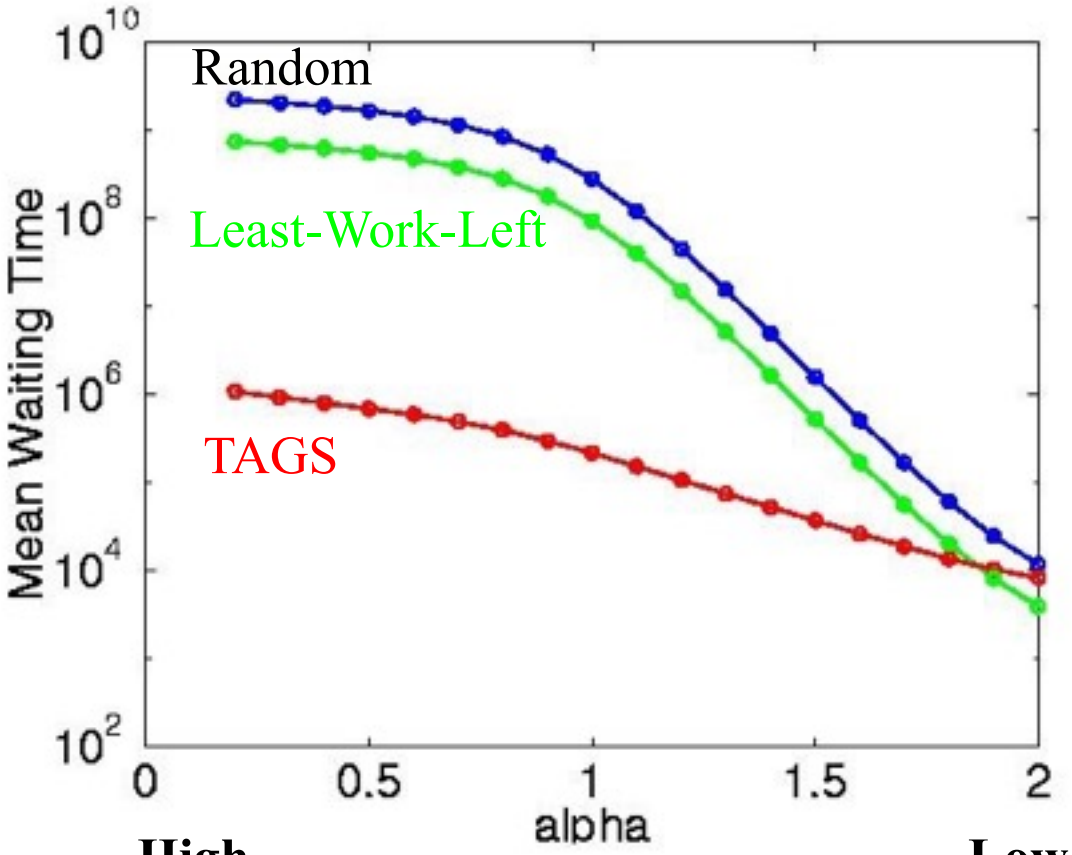
The TAGS algorithm
"Task Assignment by Guessing Size"



Answer:
When job reaches size limit for host, then it is killed and restarted from scratch at next host.

[Harchol-Balter, JACM 02]

Results of Analysis

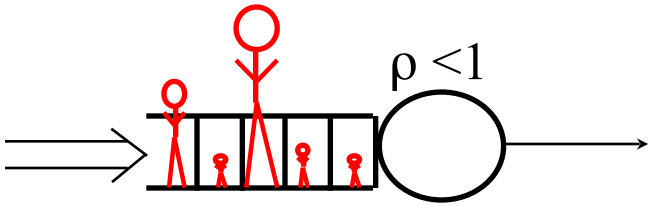


High variability

Lower variability

Summary so far …

Single-server system

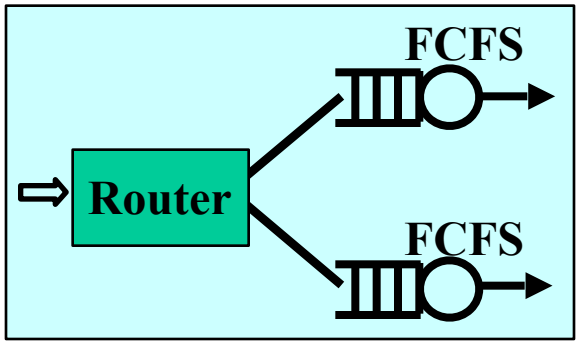


X: job size -- highly-variable

LESSONS LEARNED:

- ❑ Smart scheduling greatly improves mean response time.
- ❑ Variability of job size distribution is key.

Supercomputing

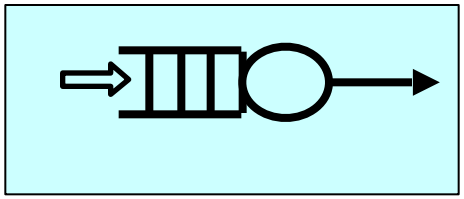


LESSONS LEARNED:

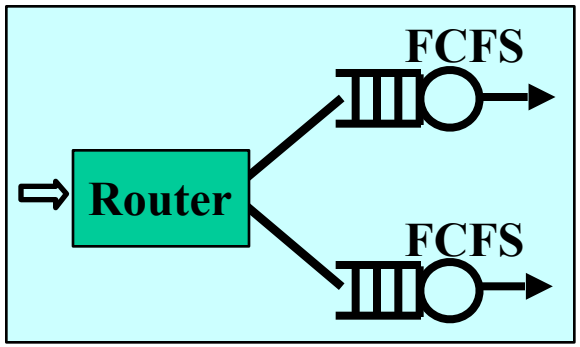
- ❑ Greedy routing policies, like JSQ, LWL are poor.
- ❑ To combat variability, need size-interval splitting.
- ❑ By isolating smalls, can achieve effects of smart single-server policies
- ❑ Don't need to know size
- ❑ Load UN-balancing

Outline

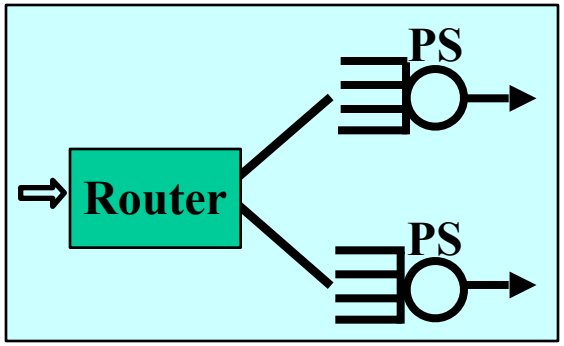
I. *Review of scheduling in single-server*



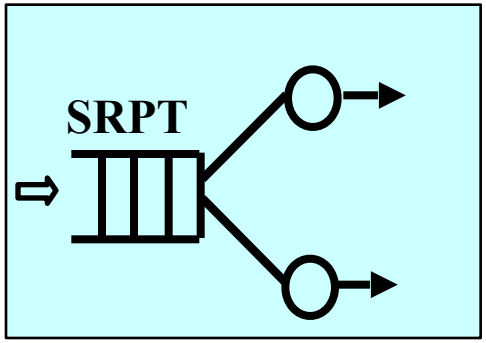
II. *Supercomputing*



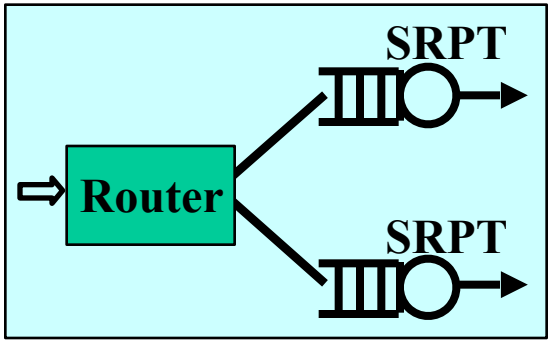
III. *Web server farm model*



IV. Towards Optimality …

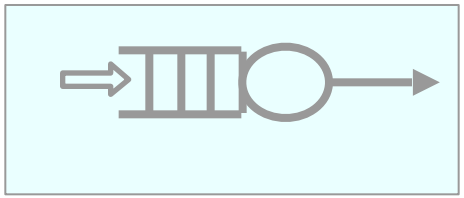


&

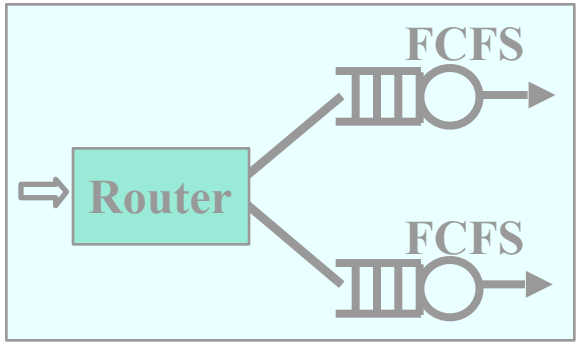


Outline

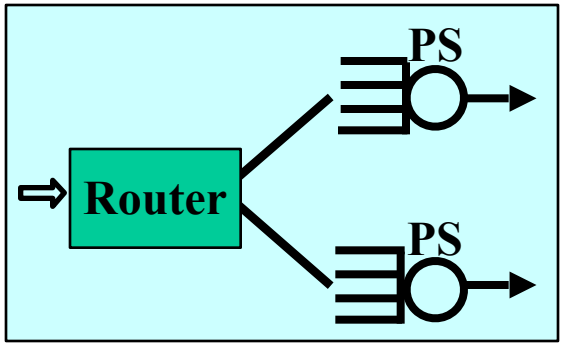
I. *Review of scheduling in single-server*



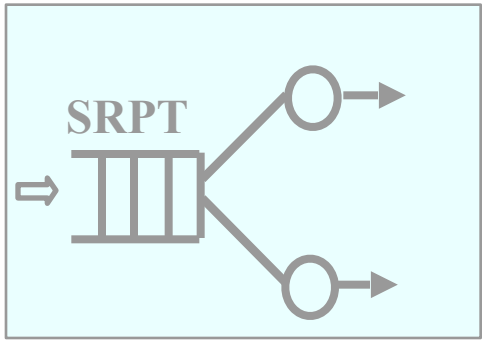
II. *Supercomputing*



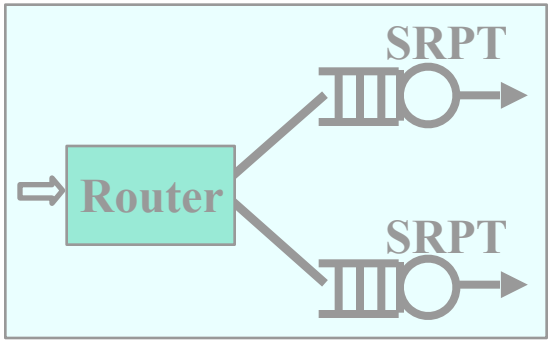
III. ***Web server farm model***



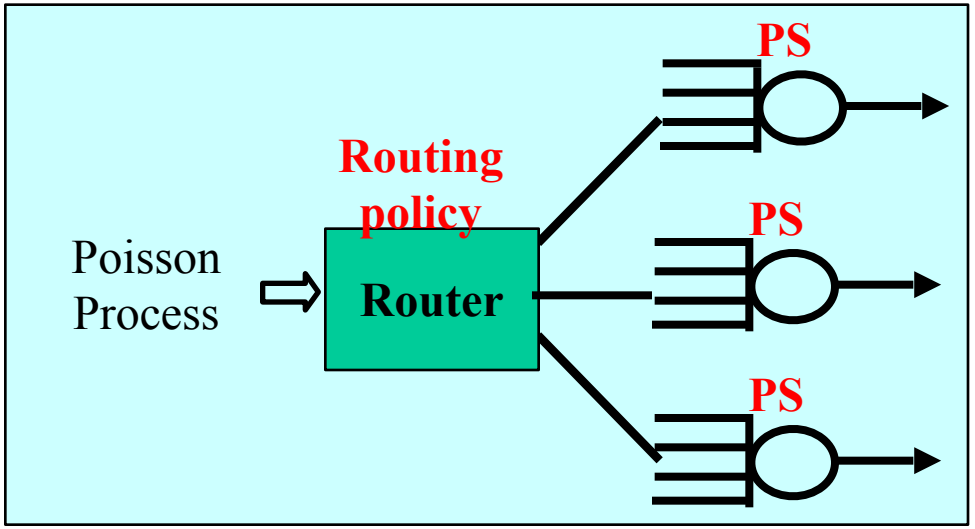
IV. Towards Optimality …



&



Web Server Farm Model



- ❑ **Cisco Local Director**
- ❑ **IBM Network Dispatcher**
- ❑ **Microsoft SharePoint**
- ❑ **F5 Labs BIG/IP**

- ❑ HTTP requests are immediately dispatched to server.
- ❑ Requests are fully preemptible.
- ❑ Commodity servers utilized → Do Processor-Sharing.
- ❑ Jobs i.i.d. ~ G: highly variable size distribution,
  7 orders magnitude difference in job size
  [Crovella, Bestavros 98].

Q: Compare Routing Policies for E[T]?

High $E[T]^{FCFS}$

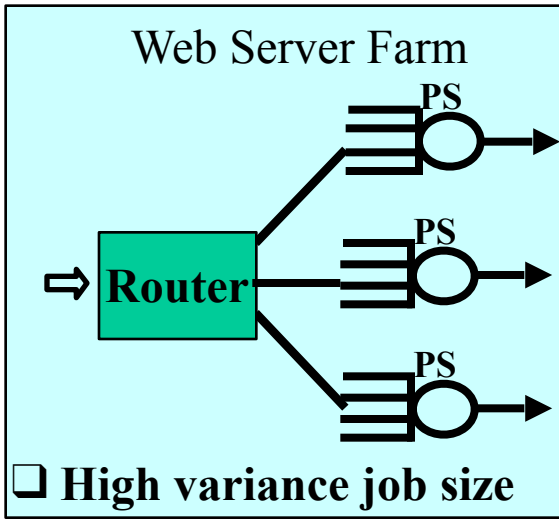
1. **Random**

2. **Join-Shortest-Queue**
   Go to host w/ fewest # jobs.

3. **Least-Work-Left**
   Go to host with least total work.

4. **Size-Interval Splitting**
   Jobs are split up by size among hosts.

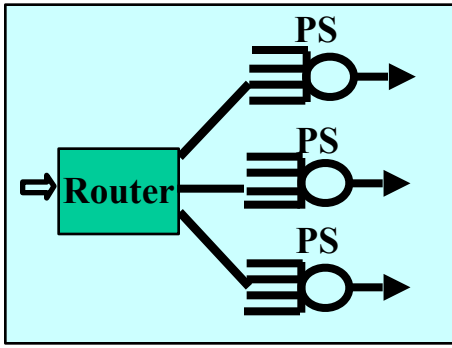
Low $E[T]^{FCFS}$



?

?

Q: Compare Routing Policies for E[T]?



1. **Random**

2. **Join-Shortest-Queue**
   Go to host w/ fewest # jobs.

3. **Least-Work-Left**
   Go to host with least total work.

4. **Size-Interval Splitting**
   Jobs are split up by size among hosts.

Answer: Shortest-Queue is greedier & better.

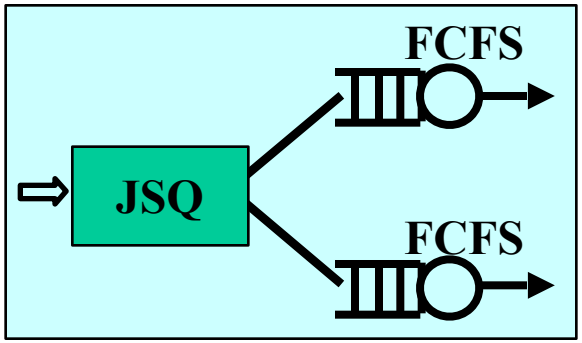
Answer: Same high E[T]. Also, **want** to *balance* load!

$$E[T]^{\text{M/G/1/PS}} = \frac{1}{\lambda} \cdot \frac{\rho}{1-\rho}$$

$$E[T]^{\text{PS farm}} = \sum_i p_i \left(\frac{1}{\lambda p_i} \cdot \frac{\rho}{1-\rho} \right)$$

Prior Analysis of JSQ Routing

All prior JSQ analysis assumes FCFS servers



2-server:

[Kingman 61] , [Flatto, McKean 77],
[Wessels, Adan, Zijm 91]

[Foschini, Salz 78],
[Knessl, Makkowsky, Schuss, Tier 87]

[Conolly 84], [Rao, Posner 87], [Blanc 87],
[Grassmann 80], [Muntz, Lui, Towsley 95]

[Cohen, Boxma 83]

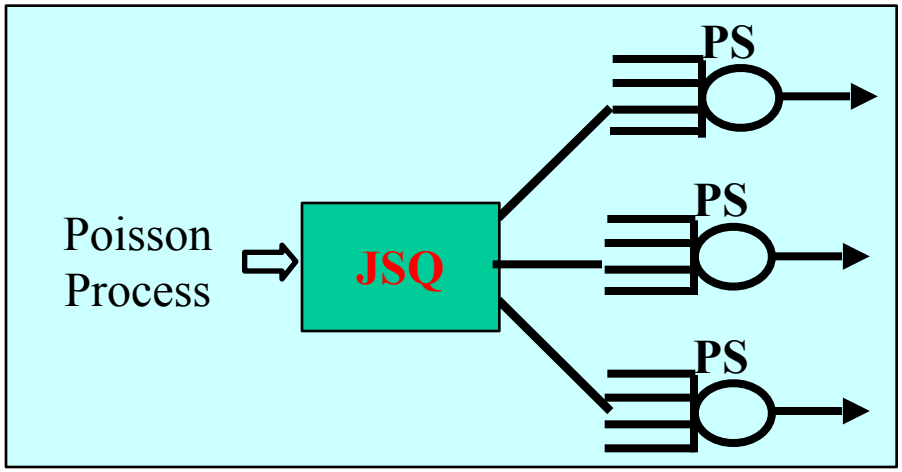
>2-server approximations:

[Nelson, Philips, Sigmetrics 89]
[Nelson, Philips, Perf.Eval. 93]

[Lin, Raghavendra, TPDS 96]

First Analysis of JSQ for PS

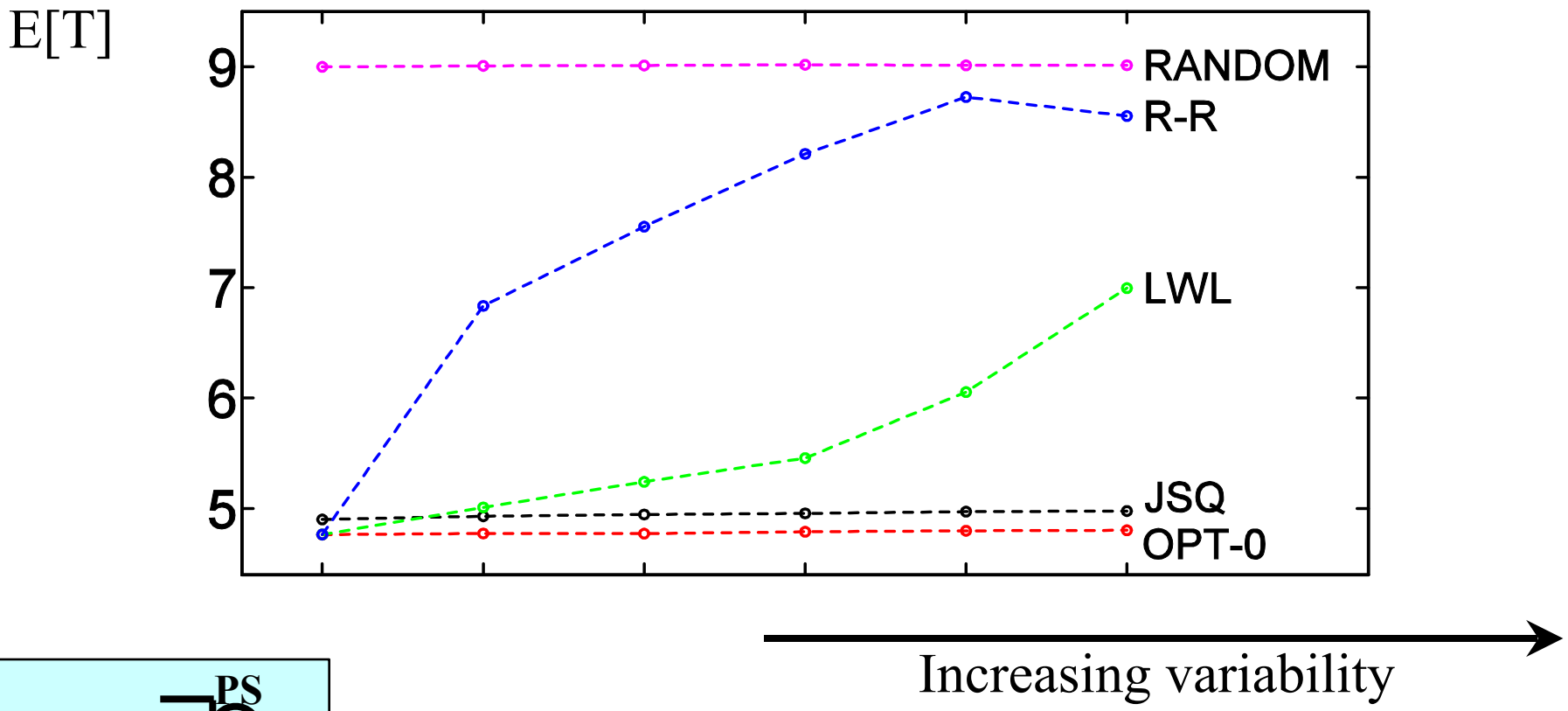
[Gupta, Harchol-Balter, Sigman, Whitt, Performance 07]

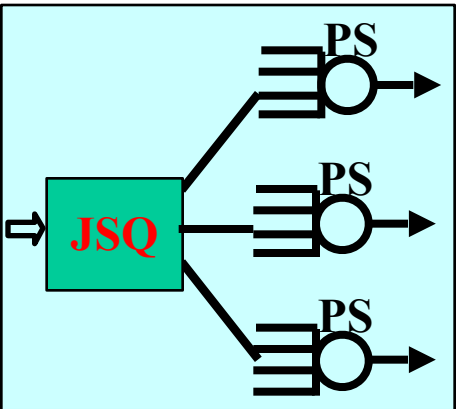


<u>Single-Queue-Approximation</u>:
M/G/k/JSQ-PS system ~ M_n/G/1/PS

First Analysis of JSQ for PS

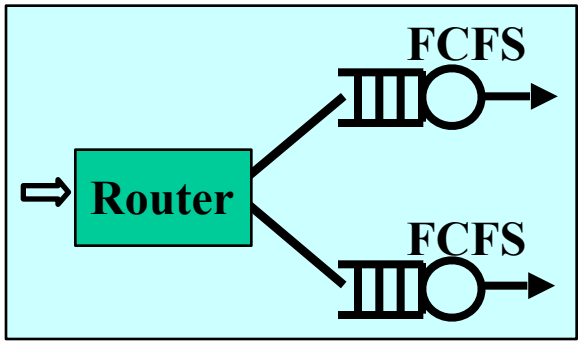
[Gupta, Harchol-Balter, Sigman, Whitt, Performance 07]





Summary so far

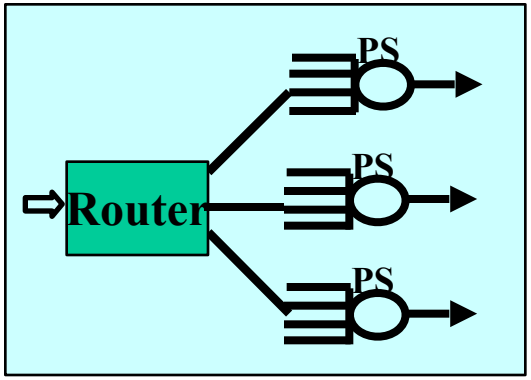
Supercomputing



LESSONS LEARNED:

- ❑ Greedy routing policies, like JSQ, LWL are poor.
- ❑ To combat variability, need size-interval splitting.
- ❑ By isolating smalls, can achieve effects of smart single-server policies
- ❑ Don't need to know size.
- ❑ Load UN-balancing

Web server farm



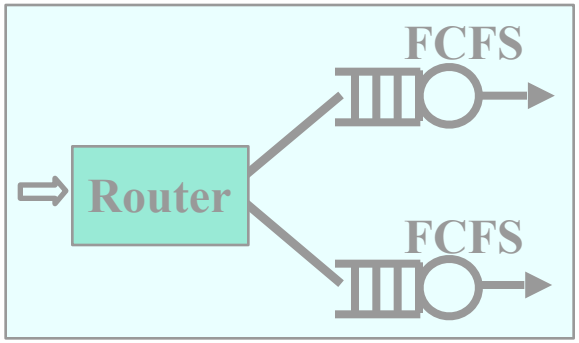
LESSONS LEARNED:

- ❑ JSQ routing is good!
- ❑ Job size variability not a problem.
- ❑ Load Balancing

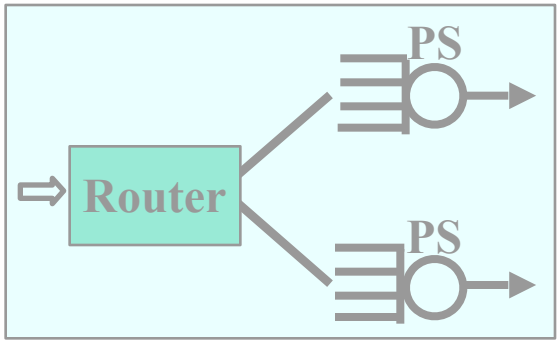
Outline

I. Review of scheduling in single-server

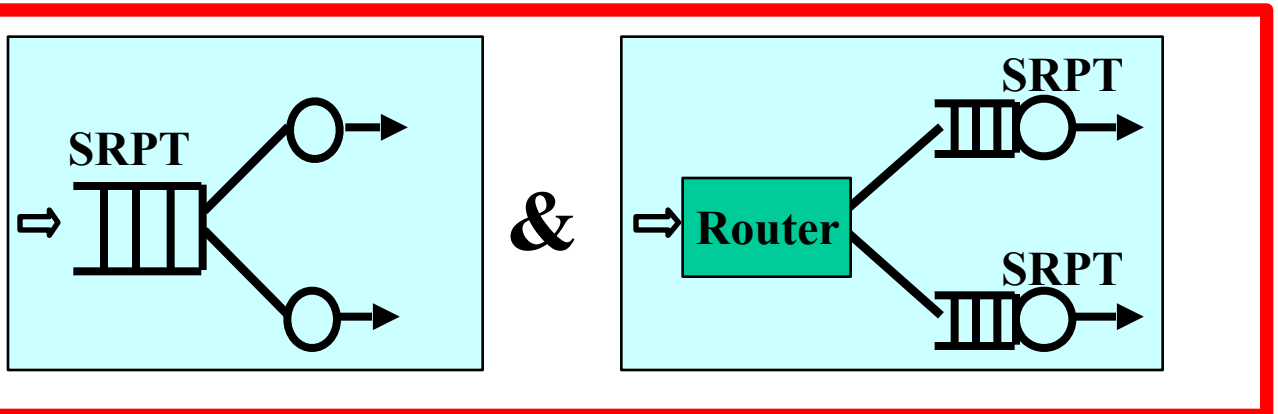
II. *Supercomputing/Manufacturing*



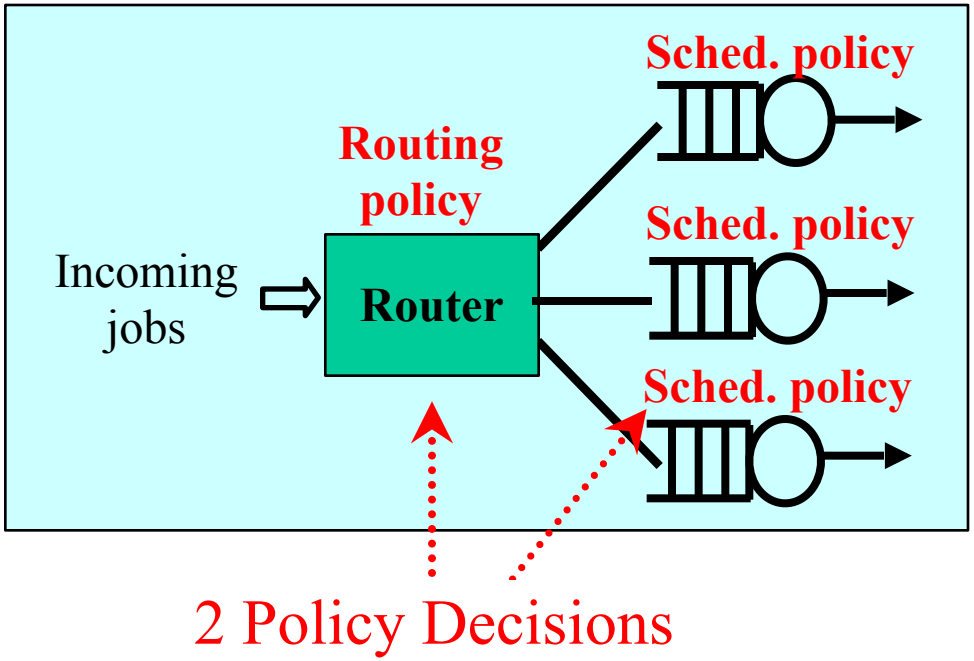
III. *Web server farm model*



IV. Towards Optimality …



What is Optimal Routing/Scheduling?

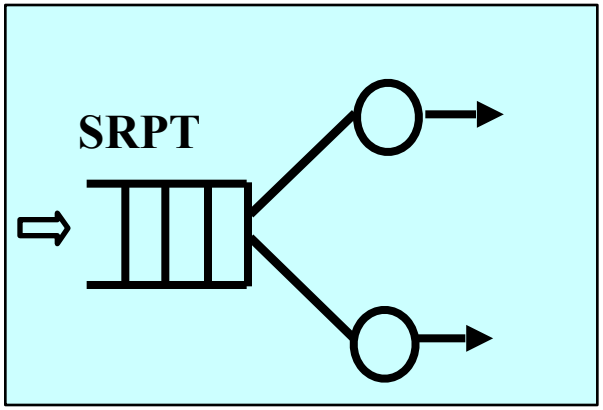


Assume no restrictions:

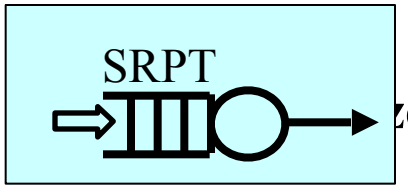
- Jobs are fully preemptible.
- Can have central queue if want it, or not.
- Know job size
  (of course don't know future jobs ...)

What is Optimal Routing/Scheduling?

Central-Queue-SRPT



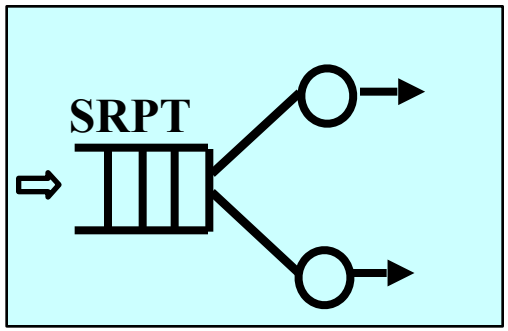
Recall:



zes E[T] <u>on every sample path</u>!
[Schrage 67]

<u>Question</u>: Central-Queue-SRPT looks pretty good!
Does it minimize E[T]?

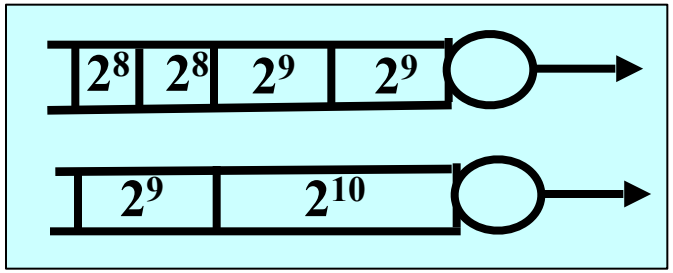
Central-Queue-SRPT



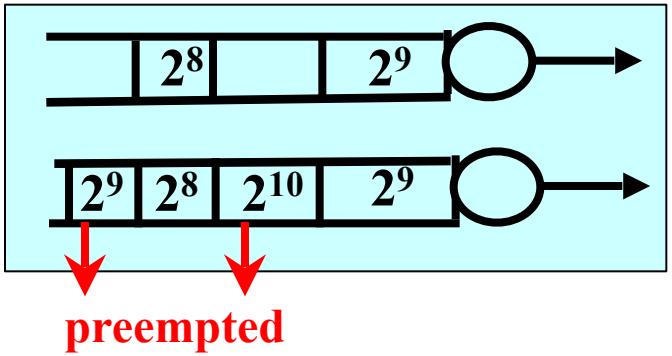
Answer: This does *not* minimize E[T] on every arrival sequence.

Bad Arrival Sequence:
@time 0: 2 jobs size 2^9, 1 job size 2^{10}
@time 2^{10}: 2 jobs size 2^8, 1 job size 2^9
@time $2^{10} + 2^9$: 2 jobs size 2^7, 1 job size 2^8, etc.

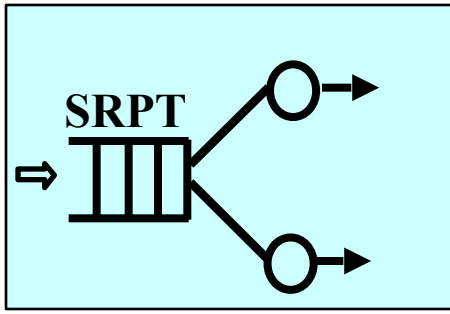
OPT



Central-Queue-SRPT



Central-Queue-SRPT



Adversarial (Worst-Case) Guarantees:

THM: [Leonardi, Raz, STOC 97]: Central-Queue-SRPT is

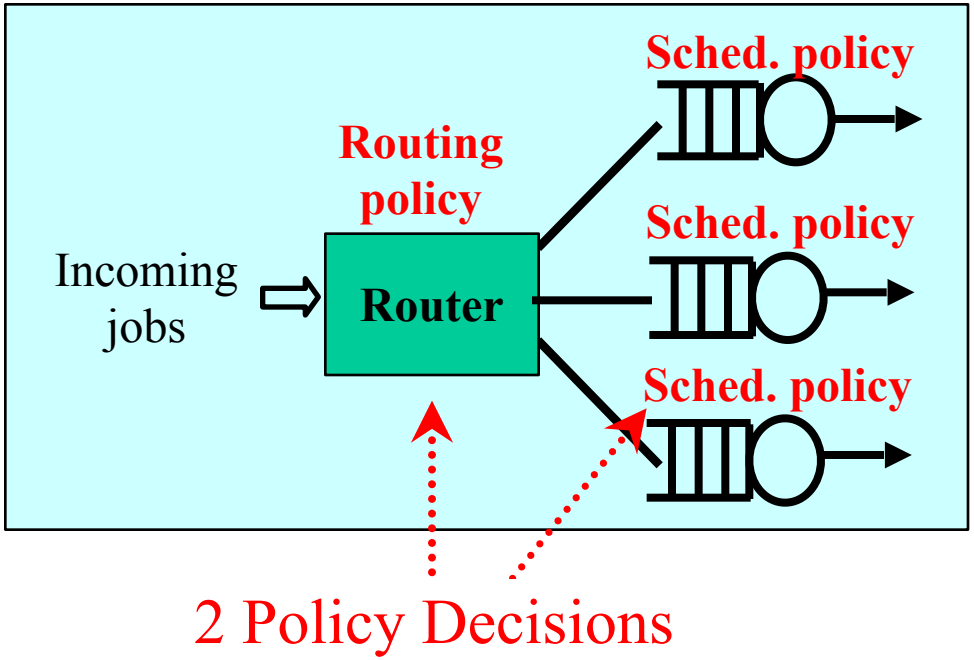
$$O\left(\log\left(\frac{\text{biggest size}}{\text{smallest size}}\right)\right)$$

competitive for E[T], and no online policy can beat this.

Remarks:

- log(biggest/smallest) could be factor 7 in practice!
- Closest *stochastic* result analyzes only central-queue w/priorities: [Harchol-Balter, Wierman, Osogami, Scheller-Wolf, QUESTA 05]

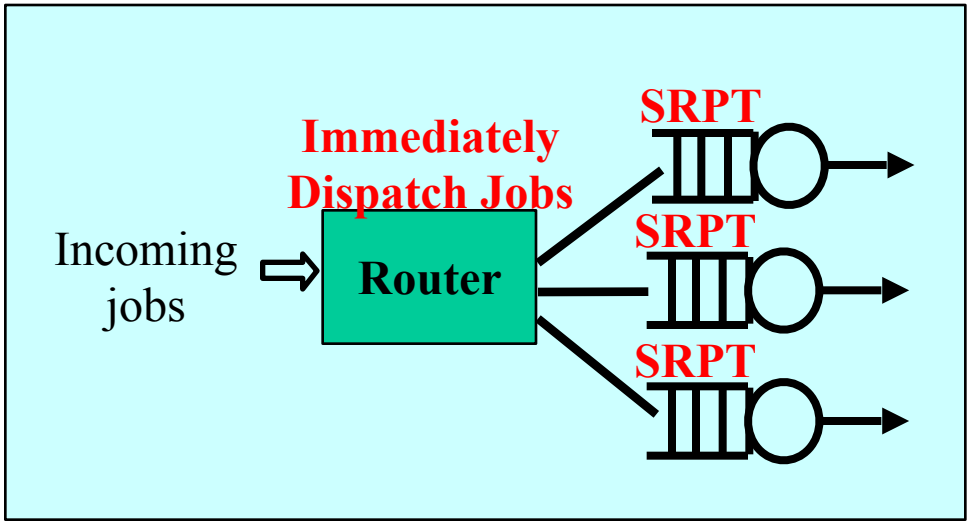
What is Optimal Routing/Scheduling with Immediate Dispatch?



Practical Assumption: jobs must be immediately dispatched!

- ❑ Jobs are fully preemptible within queue.
- ❑ Know job size.

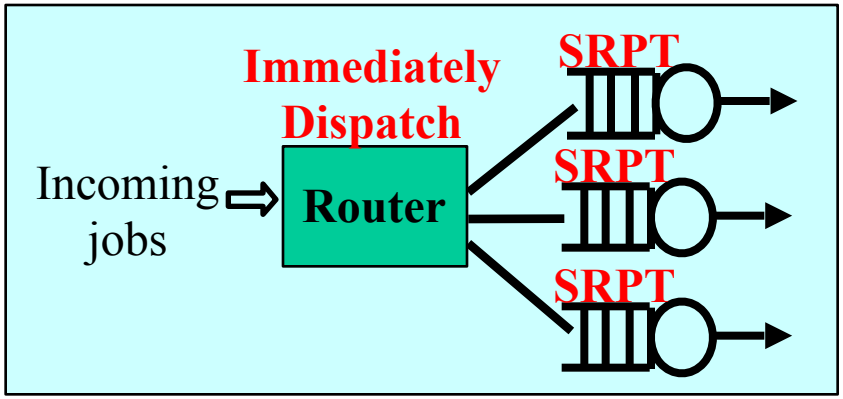
In search of good Immediate Dispatch Routing



Theorem:
The optimal routing/ scheduling pair uses SRPT at the hosts.

Q: What should immediate dispatch routing policy be, given SRPT sched. at hosts?

Smart Immediate Dispatch Policy



Answer: IMD Algorithm due to [Avrahami,Azar 03]:

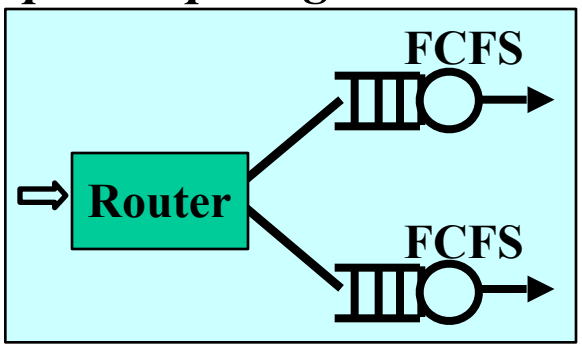
- Split jobs into size classes
- Assign each incoming job to server w/ fewest #jobs in that class

Remarks:

- IMD is $O\left(\log\left(\frac{\text{biggest}}{\text{smallest}}\right)\right)$-competitive for E[T].
  → Immediate Dispatching is "as good as" Central-Queue-SRPT
- Similar policy proposed by [Wu,Down 06] for heavy-traffic setting.

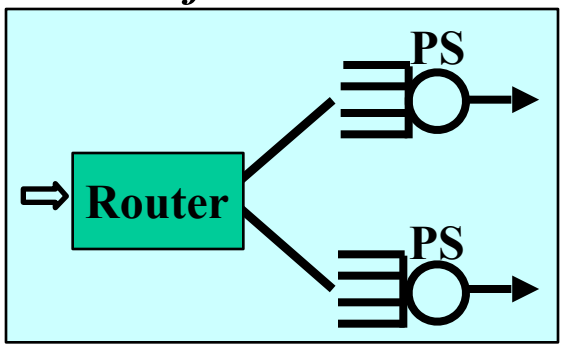
Some Key Points

Supercomputing



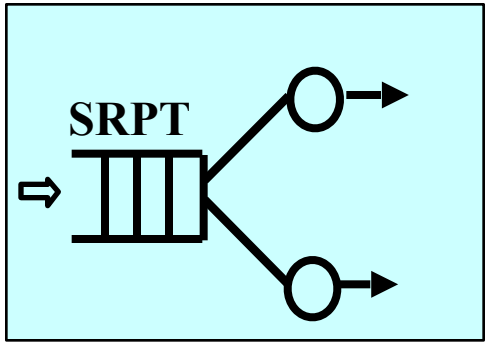
- Need Size-interval splitting to combat job size variability and enable good performance.

Web server farm model

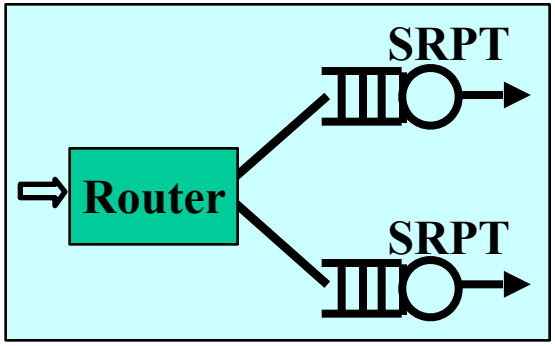


- Job size variability is not an issue.
- Greedy, JSQ, performs well.

Towards Optimality …



&



- Both these have similar worst-case E[T].
- Almost exclusively worst-case analysis, so hard to compare with above results.
- Need stochastic research here!

If you want to know more …

My class lectures are all available online.

15-857 Performance Modeling

**** Highly-recommended for CS theory, Math, TEPPER, and ACO doctoral students**

Queueing theory is an old area of mathematics which has recently become very hot. The goal of queueing theory has always been to improve the design/performance of systems, e.g. networks, servers, memory, disks, distributed systems, etc., by finding smarter schemes for allocating resources to jobs.

In this class we will study the beautiful mathematical techniques used in queueing theory, including stochastic analysis, discrete-time and continuous-time Markov chains, renewal theory, product-forms, transforms, supplementary random variables, fluid theory, scheduling theory, matrix-analytic methods, and more. Throughout we will emphasize realistic workloads, in particular heavy-tailed workloads.

This course is packed with open problems -- problems which if solved are not just interesting theoretically, but which have huge applicability to the design of computer systems today.

Instructor: Mor Harchol-Balter (harchol@cs.cmu.edu)

www.cs.cmu.edu/~harchol/

References

❑ N. Bansal and M. Harchol-Balter, "Analysis of SRPT Scheduling: Investigating Unfairness," *Proceedings of ACM Sigmetrics 2001 Conference on Measurement and Modeling of Computer Systems*.

❑ P. Barford and M. Crovella, “Generating Representative Web Workloads for Network and Server Performance Evaluation,” *Proceedings of Performance 1998/ SIGMETRICS 1998*, pp. 151-160.

❑ J. Blanc, “A Note on Waiting Times in Systems with Queues in Parallel,” *J. Appl. Prob.,* Vol. 24, 1987 pp 540-546.

❑ S. Borst, O. Boxma, and P. Jelenkovic, “Reduced load equivalence and induced burstiness in GPS queues with long-tailed traffic flows,” *Queueing Systems*, Vol. 43, 2003, pp. 274-285.

❑ S. Borst, O. Boxma, and M. van Uitert, “The asymptotic workload behavior of two coupled queues,” *Queueing Systems*, Vol. 43, 2003, pp. 81-102.

❑ J.W. Cohen and O. Boxma, *Boundary Value Problems in Queueing System Analysis*, North Holland, 1983

❑ B.W. Conolly, “The Autostrada Queueing Problem,” *J. Appl. Prob.*: Vol. 21., 1984,

References, cont.

❑ M. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: evidence and possible causes," Proceedings of the 1996 ACM Sigmetrics International Conference on Measurement and Modeling of Computer Systems, May 1996, pp. 160-169.

❑D. Down and R. Wu, "Multi-layered round robin scheduling for parallel servers," Queueing Systems: Theory and Applications, Vol. 53, No. 4, 2006, pp. 177-188.

❑G. Fayole and R. Iasnogorodski, "Two coupled processors: the reduction to a Riemann-Hilbert problem," *Zeitschrift fur Wahrscheinlichkeistheorie und vervandte Gebiete*, vol. 47, 1979, pp. 325-351.

❑ L. Flatto and H.P. McKean, "Two Queues in Parallel," *Communication on Pure and Applied Mathematics*, Vol. 30, 1977, pp. 255-263.

❑ R. Foley and D. McDonald, "Exact asymptotics of a queueing network with a cross-trained server," *Proceedings of INFORMS Annual Meeting*, October 2003, pp. MD-062.

❑ G. Foschini and J. Salz, "A Basic Dynamic Routing Problem and Diffusion," *IEEE Transactions on Communications*, Vol. Com-26, No. 3, March 1978.

References, cont.

❑ P. Glynn, M. Harchol-Balter, K. Ramanan, “Heavy-traffic Approach to Optimizing Size-Interval Task Assignment,” Work in progress, 2006.

❑ W. Grassmann, "Transient and Steady State Results for Two Parallel Queues," *Omega,* vol. 8, 1980, pp. 105-112.

❑ V. Gupta, M. Harchol-Balter, K. Sigman, and W. Whitt, “Analysis of Join-the-Shortest-Queue Policy for Web Server Farms.” To appear in *Performance Evaluation Review, 2007*.

❑ M. Harchol-Balter and A. Downey. "Exploiting Process Lifetime Distributions for Dynamic Load Balancing," *Proceedings of ACM Sigmetrics '96 Conference on Measurement and Modeling of Computer Systems* , May 1996, pp. 13-24.

❑ M. Harchol-Balter, M. Crovella, and C. Murta, "On Choosing a Task Assignment Policy for a Distributed Server System," *Journal of Parallel and Distributed Computing* , vol. 59, no. 2, Nov. 1999, pp. 204-228.

❑ M. Harchol-Balter, C. Li, T. Osogami, and A. Scheller-Wolf, and M. Squillante, “Cycle stealing under immediate dispatch task assignment,” *Proceedings of the Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, June

References, cont.

❑M. Harchol-Balter and R.Vesilo, "To Balance or Unbalance Load in Size-Interval Task Allocation," In submission to *Performance Evaluation*, 2008.

❑ M. Harchol-Balter, A. Wierman, T. Osogami, and A. Scheller-Wolf, "Multi-server queueing systems with multiple priority classes," *Queueing Systems: Theory and Applications (QUESTA)*, vol. 51, no. 3-4, 2005, pp. 331-360.

❑ J. Kingman, "Two Similar Queues in Parallel," *Biometrika*, Vol. 48, 1961, pp. 1316-1323.

❑A. Konheim, I. Meilijson, and A. Melkman, "Processor-sharing of two parallel lines," *J. Appl. Prob.*, Vol. 18, 1981, pp. 952-956.

❑ C. Knessl, B. Matkowsky, Z. Schuss, and C. Tier, "Two Parallel M/G/1 Queues where Arrivals Join the System with the Smaller Buffer Content," *IEEE Transactions on Communications*, Vol. Com-35, No. 11,1987, pp. 1153-1158.

❑ S. Leonardi and D. Raz, "Approximating Total Flow Time on Parallel Machines," *ACM Symposium on Theory of Computing (STOC)*, 1997.

References, cont.

❑ H. Lin, and C. Raghavendra, "An Approximate Analysis of the Join the Shortest Queue(JSQ) Policy", *IEEE Transactions on Parallel and Distributed Systems*, vol. 7, no. 3, March 1996.

❑ J. Lui, R. Muntz, D. Towsley, "Bounding the mean response time of the minimum expected delay routing policy: an algorithmic approach," *IEEE Transactions on Computers*, Vol. 44, No. 12, Dec 1995.

❑ S. Muthukrishnan, R. Rajaraman, A. Shaheen, and J. Gehrke, "Online Scheduling to Minimize Average Stretch," *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, October 1999, pp. 433.

❑ R. Nelson and T. Philips, "An approximation to the response time for shortest queue routing," *ACM SIGMETRICS Performance Evaluation Review*, Vol. 17 No. 1, May 1989, pp. 181-189.

❑ R. Nelson and T. Philips, "An approximation for the mean response time for shortest queue routing with general interarrival and service times," *Performance Evaluation*, Vol. 17 No. 2, March 1993 pp. 123-139.

References, cont.

- B. Rao and M. Posner, "Algorithmic and Approximate Analysis of the Shorter Queue," *Model Naval Research Logistics*, Vol. 34, 1987, pp. 381-398.

- T. Osogami, M. Harchol-Balter, and A. Scheller-Wolf, "Analysis of cycle stealing with switching cost," *Proceedings of the ACM Sigmetrics*, June 2003, pp. 184-195.

- R. Righter and J. Shanthikumar, "Scheduling multiclass single server queueing systems to stochastically maximize the number of successful departures**,"** *Probability in the Engineering and Informational Sciences*, Vol. 3, 1989, pp. 323-333.

- L.E. Schrage, "A proof of the optimality of the shortest processing remaining time discipline," *Operations Research*, Vol. 16, 1968, pp. 678-690.

- B. Schroeder and M. Harchol-Balter, "Evaluation of Task Assignment Policies for Supercomputing Servers: The Case for Load Unbalancing and Fairness," *9th IEEE Symposium on High Performance Distributed Computing (HPDC '00)* , August 2000.

References, cont.

❑ B. Schroeder, A. Wierman, and M. Harchol-Balter. "Closed versus Open System Models: a Cautionary Tale," *Proceedings of NSDI* , 2006.

❑A. Shaikh, J. Rexford, and K. Shin, "Load-sensitive routing of long-lived IP flows," *Proceedings of SIGCOMM*, September, 1999.

❑ J. Wessels, I. Adan, and W. Zijm, "Analysis of the asymmetric shortest queue problem," *Queueing Systems*, Vol. 8, 1991, pp. 1-58.

❑ A. Wierman and M. Harchol-Balter, "Classifying Scheduling Policies with respect to Unfairness in an M/GI/1," *Proceedings of ACM Sigmetrics 2003 Conference on Measurement and Modeling of Computer Systems* , June 2003.

Thank you for listening!