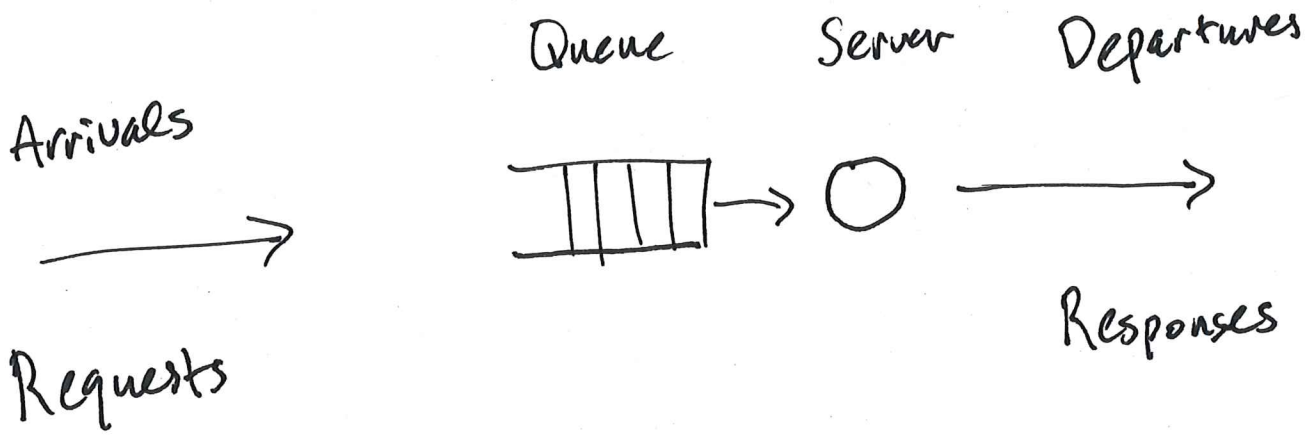
Lec 12

# Queueing Theory


Queue
Server
Departures
Arrivals
Requests
Responses

# Web Server example

- processes 1 request at a time

- takes 1 ms to process a request

- if requests arrive exactly every 10 ms, what is the throughput? latency?

- every 2 ms?

- 1 ms?

- one extra request arrives...

- every 0.99 ms?

# Definitions

response time = waiting time + service time

$R = W + S$

arrival rate $\lambda$ (avg, could be drawn from distribution)

service rate $\mu = 1/S$ how many requests can be served per unit time

throughput $X = \min(\lambda, \mu)$

number of tasks in system $N$

# Little's Law

In a stable system $(\lambda < \mu)$:

$$N = XR$$

avg number of tasks in system = (avg) throughput $\left(\frac{\text{req}}{\text{s}}\right)$ · avg response time $\left(\frac{\text{s}}{\text{req}}\right)$

Example: if $X = 100$ req/s $\quad R = \frac{1}{1000}$ s/req

then $N = 1/10$ tasks in system

→ queue is usually empty

# Evenly spaced arrivals (best case)

- tasks arrive <u>evenly spaced</u> with rate $\lambda$

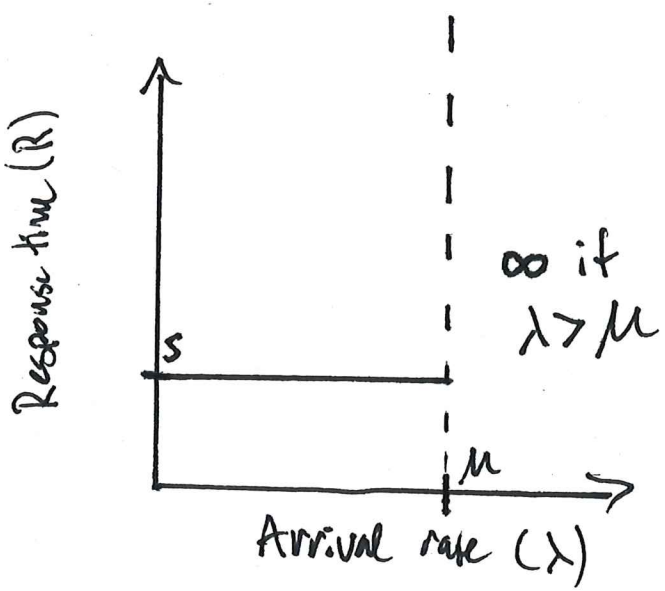
- if $\lambda < \mu$, no queueing, $W = 0$

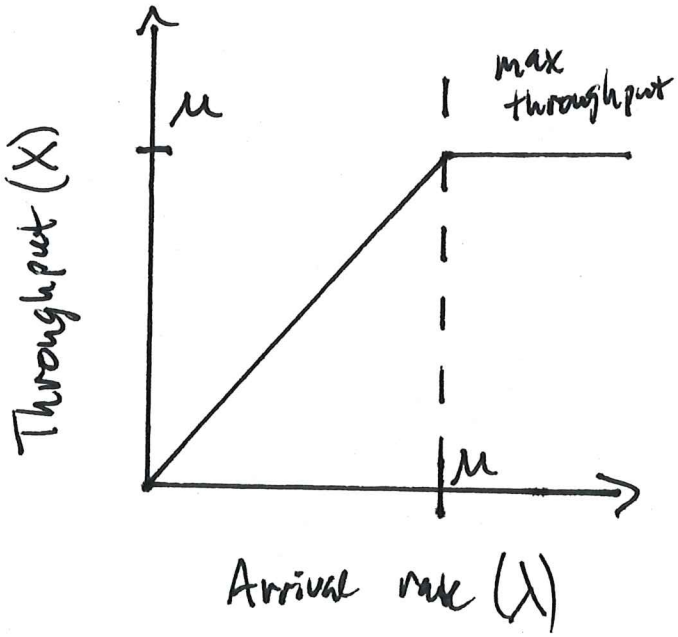
$$R = W + S = 0 + S = S$$
$$X = \lambda$$

- if $\lambda = \mu$, queue size does not change
  - empty initially → always empty
  - initially 10 in queue → always 10

- if $\lambda > \mu$, unbounded queueing

$$R = \infty$$
$$X = \mu$$



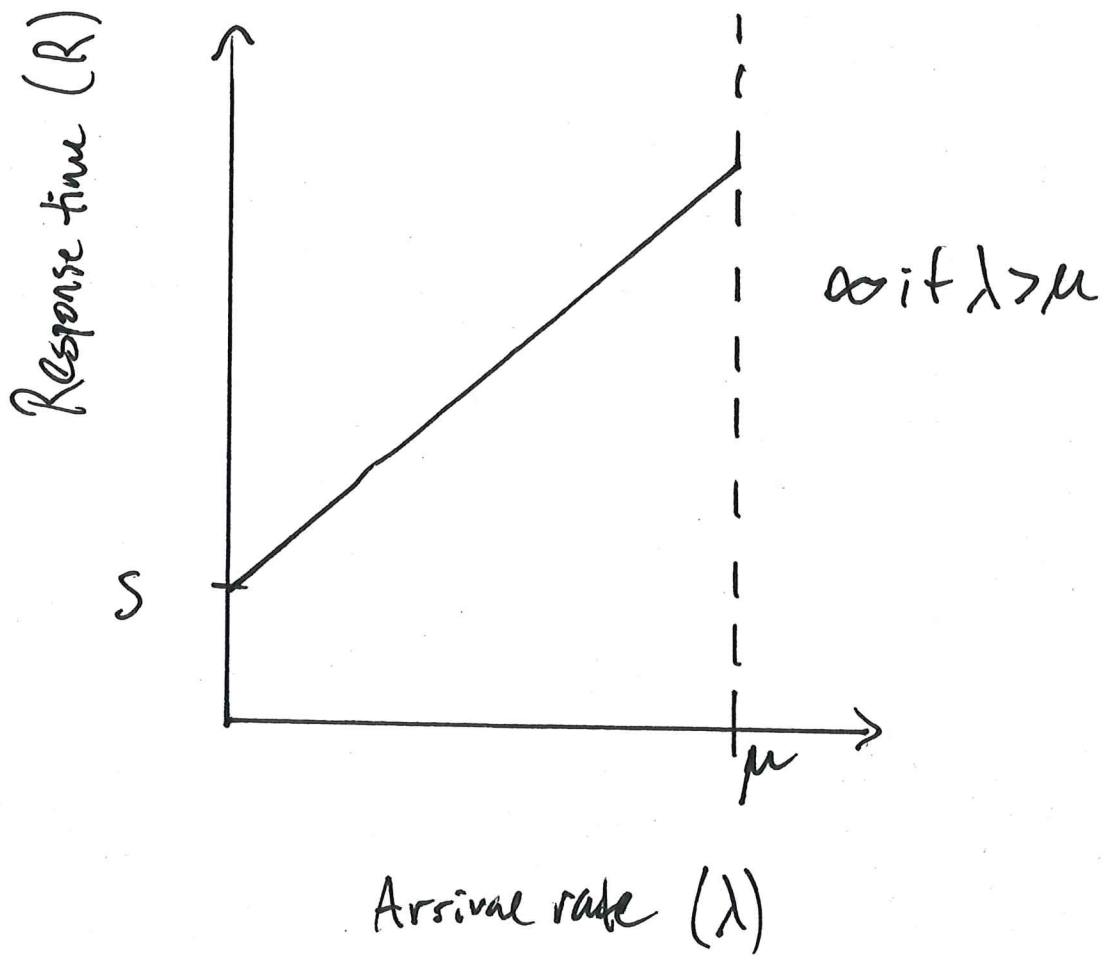



# Bursty Arrivals (worst case)

- tasks arrive in groups
  say 1 group per second
  group size = $K$ $(= \lambda)$

$R$ = avg response time

$$= (S + 2S + \cdots + KS)/K$$
$$= S(1 + 2 + \cdots + K)/K$$
$$= \frac{SK(K+1)}{2K}$$
$$= \frac{S}{2}\cdot(K+1) = S \cdot \frac{K+1}{2}$$
$$= O(K) \cdot S$$




worse by a factor of half the group size

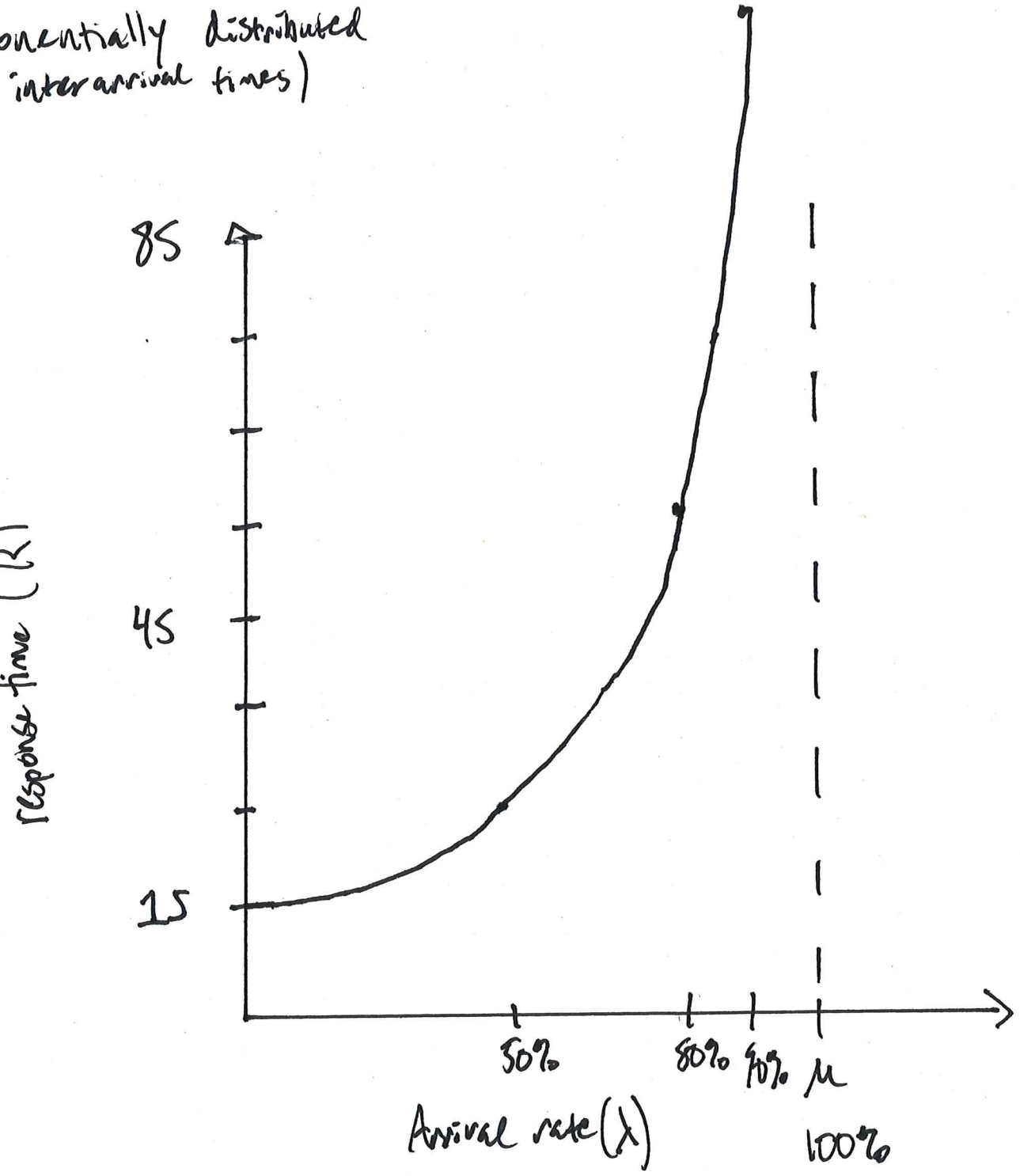
groups of 10 ↪ 5x worse!

# Independent Arrivals

- evenly spaced ⟶ perfectly anti-correlated
- grouped ⟶ perfectly correlated
- reality: different users send requests independently
  - ↳ somewhere between worst + best case
- good model: exponentially distributed interarrival times (Poisson process) ⟶ memoryless (Markov

↳ $R = \frac{S}{1-u}$ where utilization $u = \frac{\lambda}{\mu}$ $(\leq 1)$

(exponentially distributed inter arrival times)




$$R = \frac{S}{1-U}$$

$$= \frac{S}{1-\lambda/\mu}$$

# Overload Management

- if $\lambda > \mu$, response time unbounded: "overload"
- how to get out of overload?
  - reject some requests
  - make requests cheaper somehow (dyn → static)
- very common problem: systems try to do <u>more</u> work per request when overloaded
  - e.g. the more concurrent requests, the more lock contention
  - e.g. drop a request → sender retransmits

# Why do buses cluster?

- suppose bus schedule has one bus every 10 minutes
- they leave the first stop on time (10 mins apart)
- why do they tend to get to last stop at around the same time? (instead of 10 mins apart)

12.10