A Better Model for Job Redundancy: Decoupling Server Slowdown and Job Size

Kristen Gardner
Computer Science Department
Carnegie Mellon University

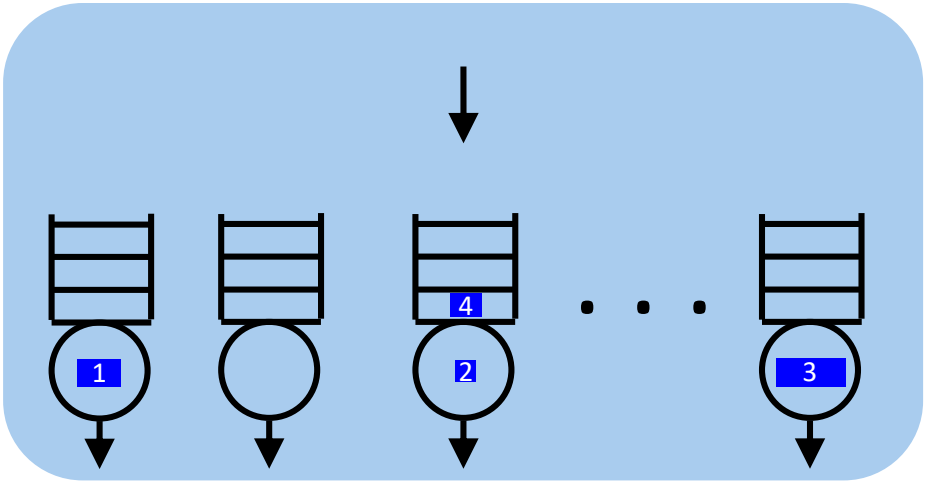
Mor Harchol-Balter
Computer Science Department
Carnegie Mellon University

Alan Scheller-Wolf
Tepper School of Business
Carnegie Mellon University

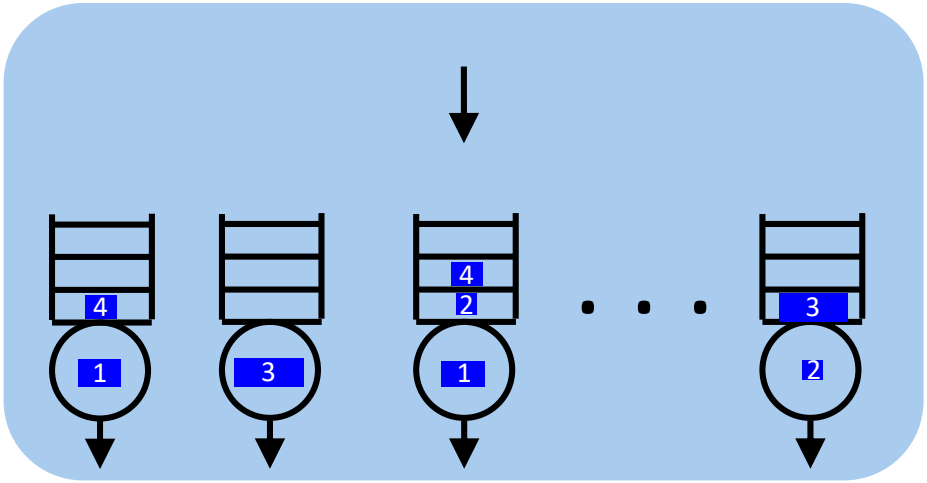
YEQT
November 7, 2016

What is Redundancy?

Traditional Dispatching

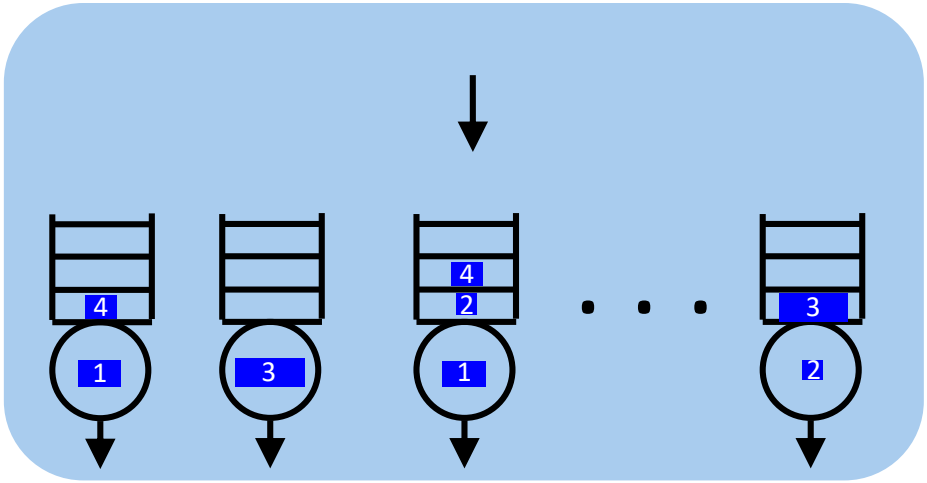


New Idea: Redundancy



Idea: create multiple copies of the same job and wait for only the first copy to complete service

Why Redundancy?



Variable Queue Lengths

- Load from different applications
- Variability in job size

Variable Service Times

Same job can take up to 27X longer on one server than on another [Xu et al. '13]

Redundancy reduces mean response time by…

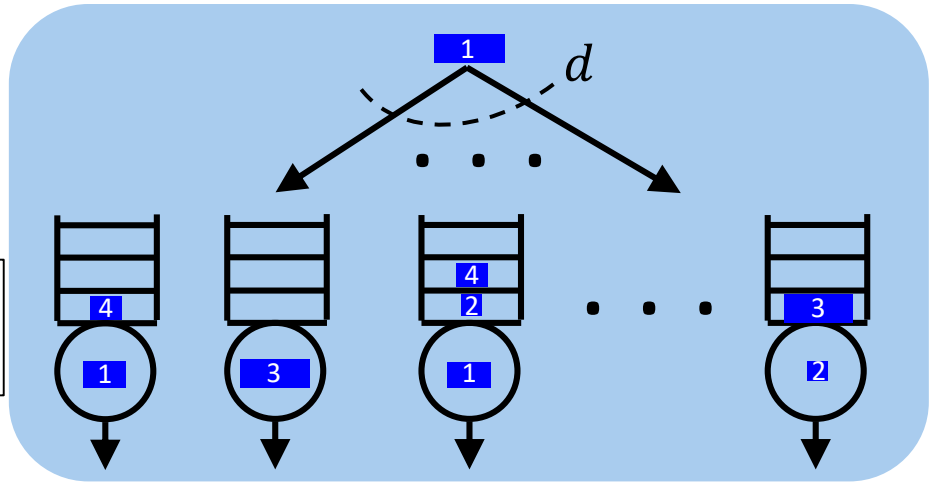
16% [Google's BigTable, Dean & Barroso '13]

34-46% [Facebook's Hadoop cluster, Ananthanarayanan et al. '13]

50% [DNS queries, Vulimiri et al. '13]

…with even bigger improvements at the tail!

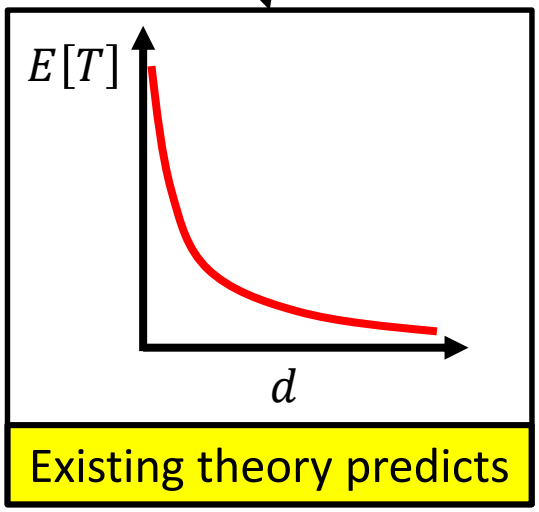
The Redundancy-d Policy



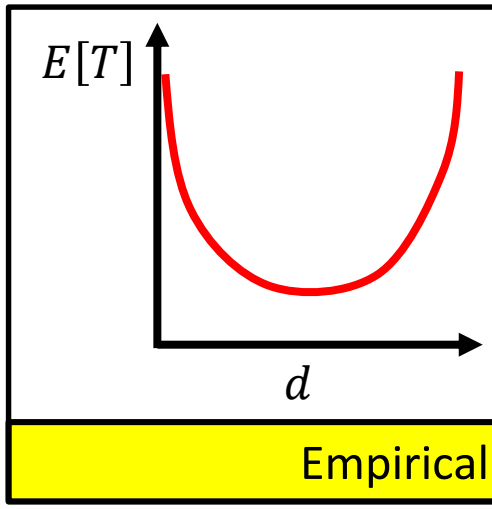
e.g., Gardner et al. ’16, Koole and Righter ’08

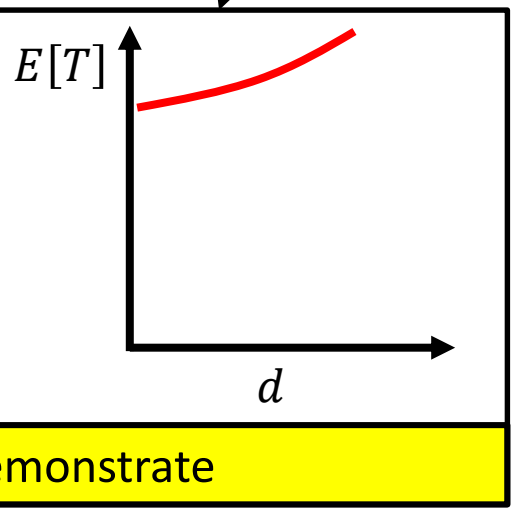
e.g., Vulimiri et al. ’13

What happens to mean response time as d increases?



Existing theory predicts





Empirical results demonstrate

Why don't theory and practice match?

Common Modeling Assumptions	vs.	Systems Realities
• **Exponentially** distributed runtimes		• **Generally** distributed runtimes
• **Instantaneous** cancellation of extra copies		• **Non-immediate** cancellation of extra copies
• **Independent** runtimes across servers		• **Correlated** runtimes across servers

Wrong assumptions → qualitatively misleading results

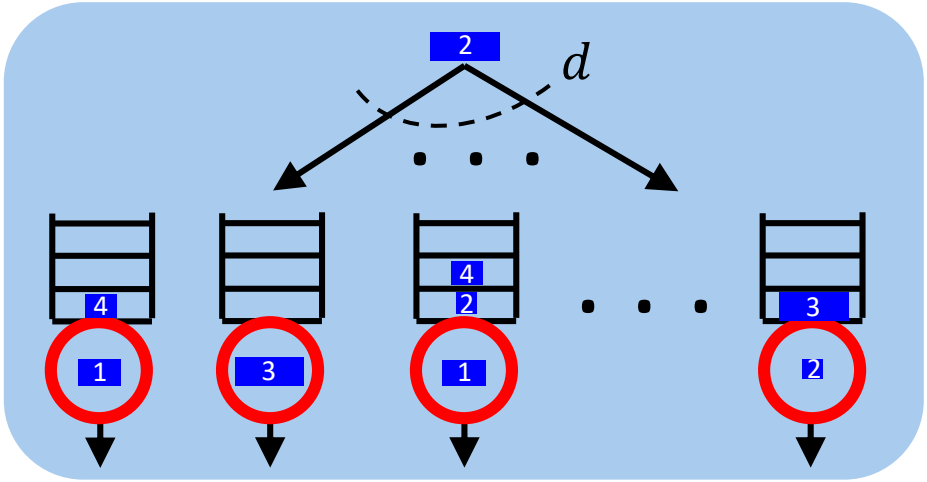
Our goal: Better theoretical model

Our Solution: The S&X Model

server slowdown

inherent job size

Redundancy-d in S&X Model



Traditional queueing theory uses only one "service time" variable. This is **inadequate** for redundancy.

$$\text{Runtime} = R = S \cdot X$$

$$\text{Response Time} = T = T_Q + R$$

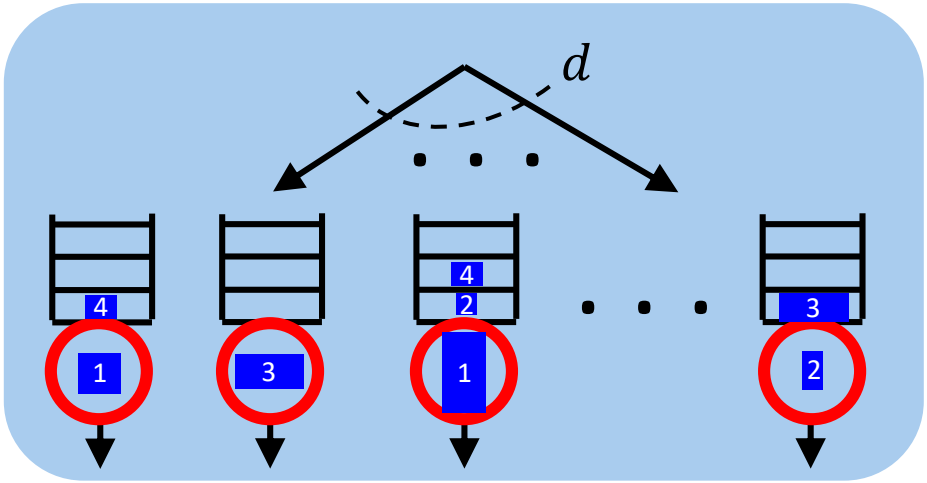
When $d > 1$, see min of d response times

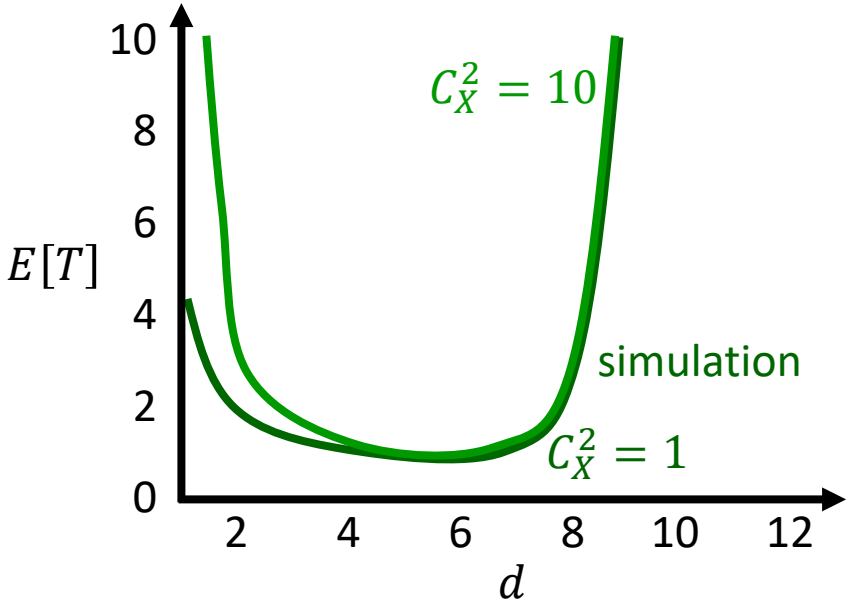
Redundancy-d in the S&X Model

server slowdown

inherent job size

Redundancy-d in S&X Model





$k = 1000$ servers

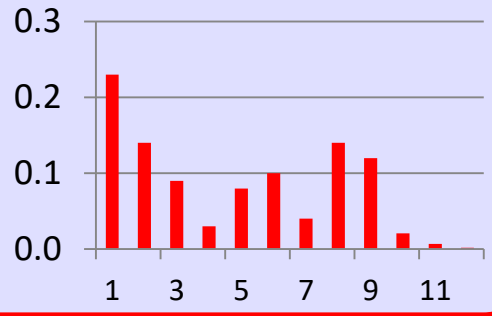
Arr. rate $\lambda = 0.7k$

$\boldsymbol{X} \sim H_2$, $E[X] = \frac{1}{4.7}$

$\boldsymbol{S}$~Dolly empirical dist.

Dolly empirical server slowdown distribution [Ananthanarayanan '13]

$E[S] = 4.7$



S&X model effectively captures important characteristics of real systems

↓

Right qualitative trends

BUT . . .

1) <u>Robustness:</u> Redundancy-d can lead to overload in the **S&X** model

2) <u>Analytical tractability:</u> Redundancy-d cannot be analyzed in the **S&X** model

Our Solution: Redundant-to-Idle-Queue (RIQ)

Limits extra load added by replicas → no overload, more robust

Analytically tractable (approximately)

$\forall S$ $\forall X$ $\forall Z$

server slowdown

inherent job size

cancellation time

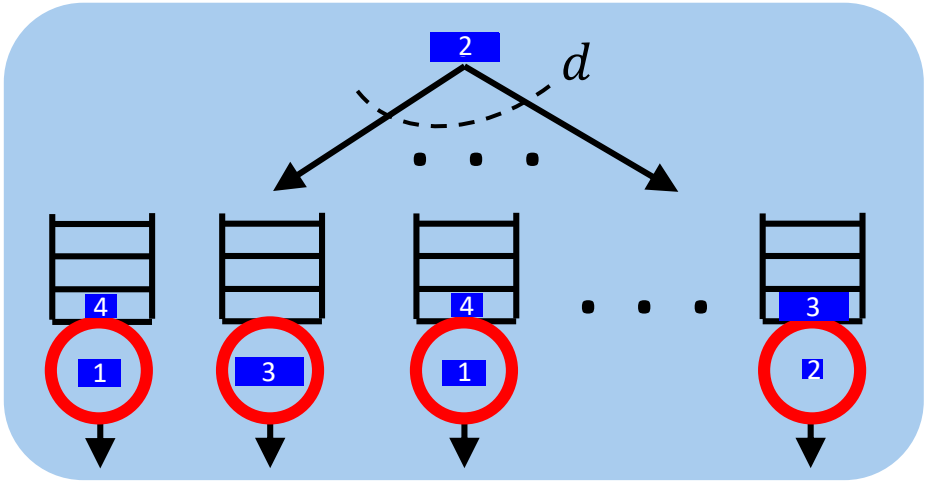
Redundant-to-Idle-Queue (RIQ)

RIQ policy:

Arrival polls d servers chosen at random

- Replicate to *all* idle servers among the d
- If none are idle, pick one at random among the d

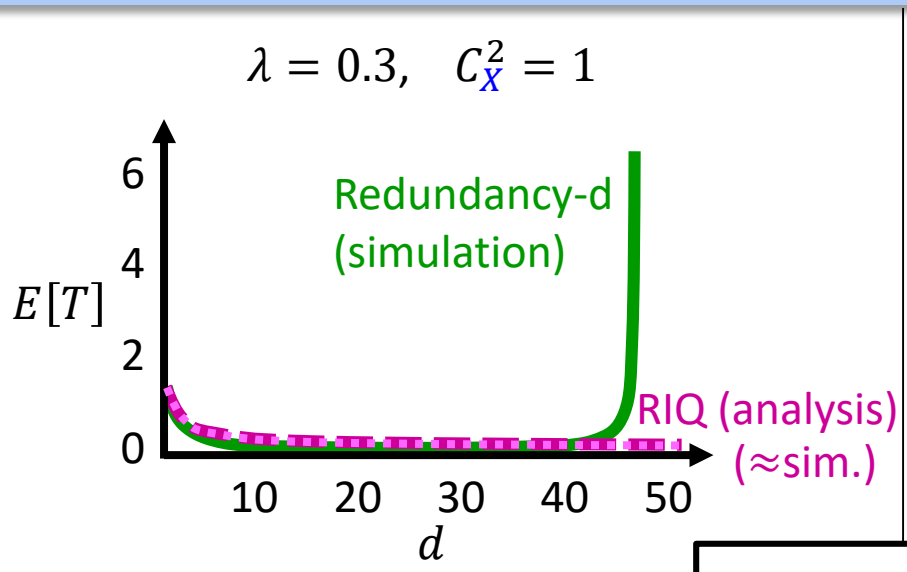
RIQ in **S&X** Model

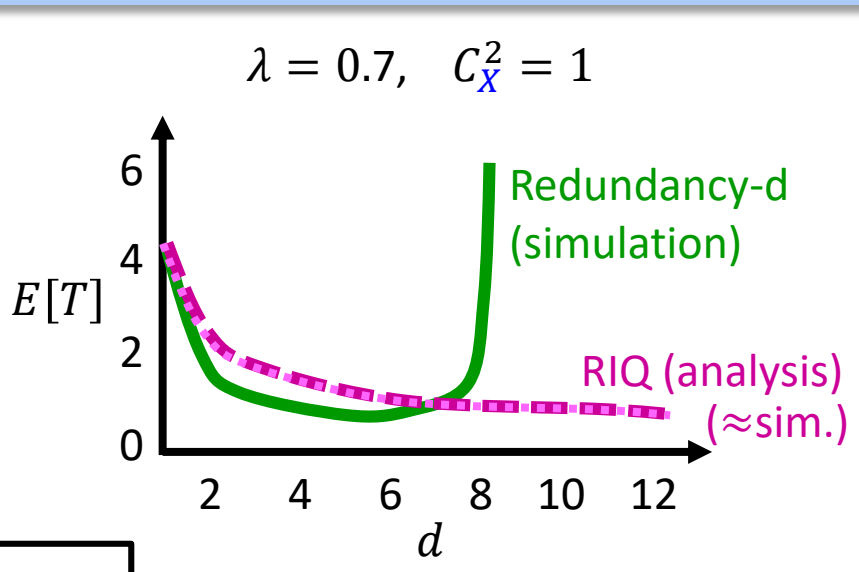


Intuition: RIQ only adds load by creating extra copies if the system has capacity to spare

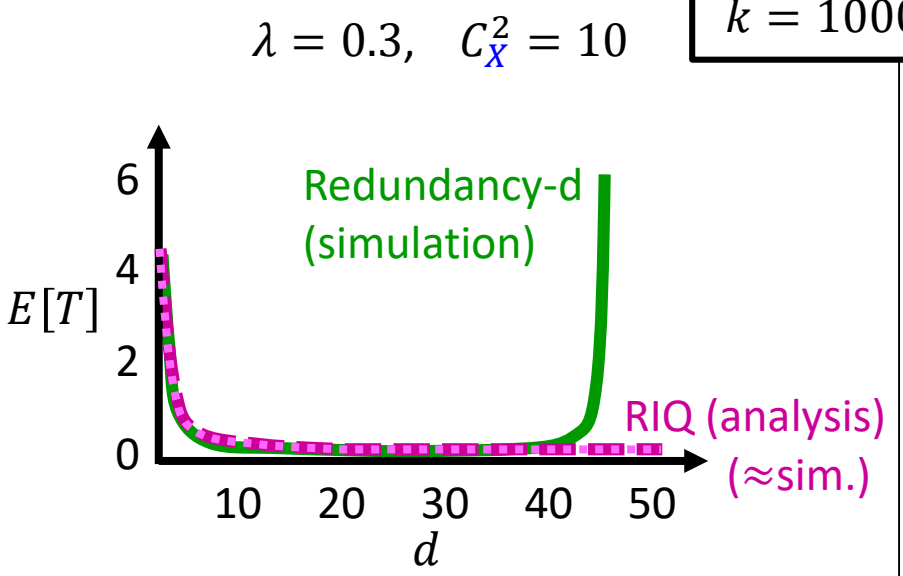
Analysis: replicas only affect first runtime in a busy period → M/G/1/exceptional 1st service

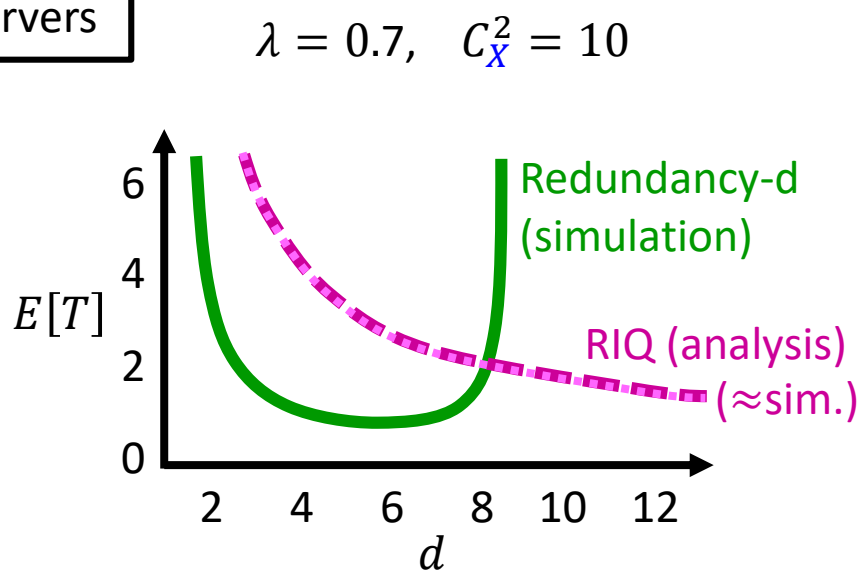
RIQ vs. Redundancy-d





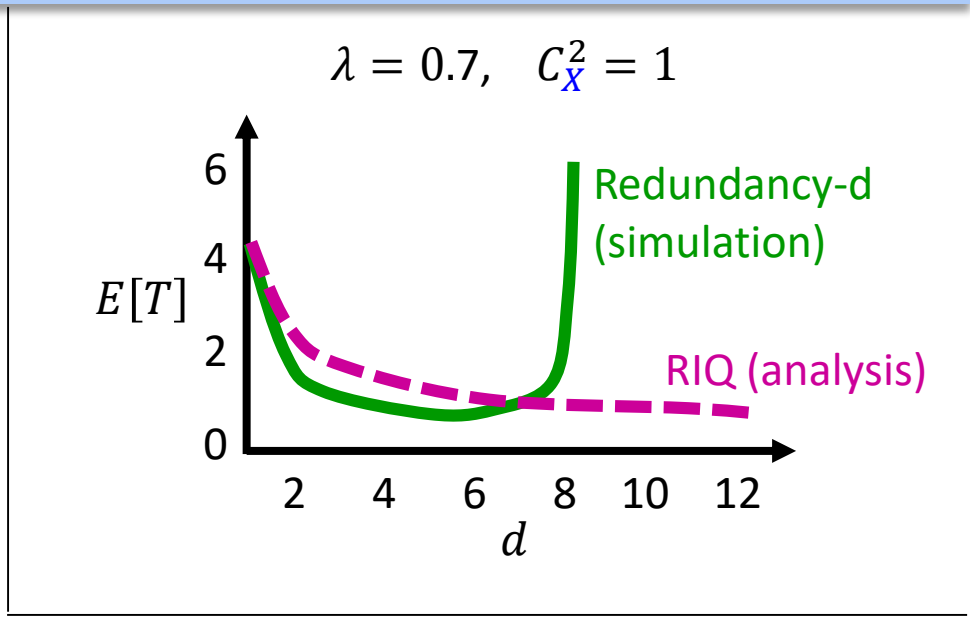
S&X
$k = 1000$ servers

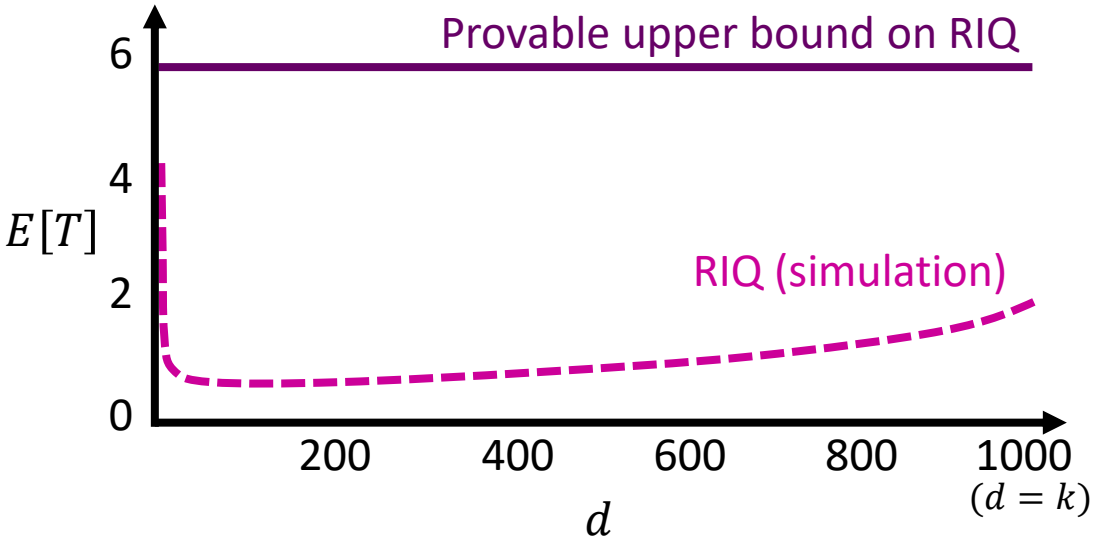




RIQ at High d

Thm. For all d,
$E[T]^{RIQ} \leq E[T]^{RIQ(d=1)} + E[(\boldsymbol{S} \cdot \boldsymbol{X})_{excess}]$



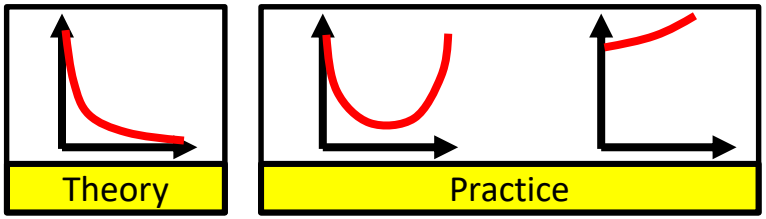


Thm. RIQ is stable iff
$\lambda \cdot E[\boldsymbol{S} \cdot \boldsymbol{X}] < 1$

Conclusions

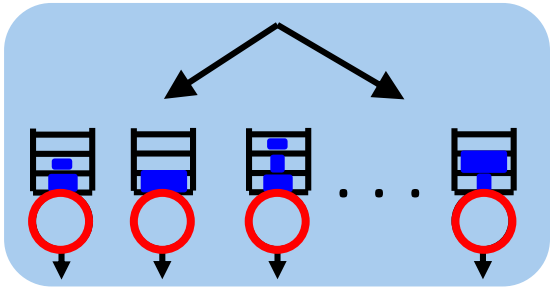
Problem

Mismatch between real systems trends and existing theoretical results

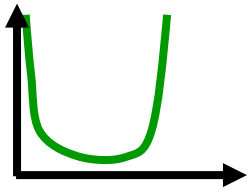


Our Solution

Introduce the S&X model



In **S&X** model, Redundancy-d is



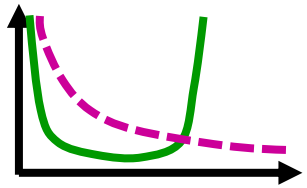
NOT robust

?

NOT analytically tractable

Introduce RIQ

PROVABLY robust



ANALYZABLE in approximation