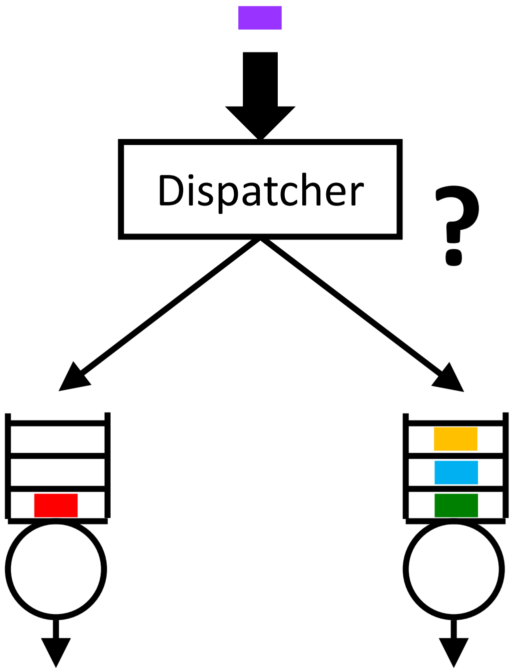
A General "Power-of-d" Dispatching Framework for Heterogeneous Systems

Kristy Gardner
Amherst College

Based on joint work with Jazeem Abdul Jaleel,
Sherwin Doroudi, and Alexander Wickeham
University of Minnesota

YEQT Conference
June 7, 2021

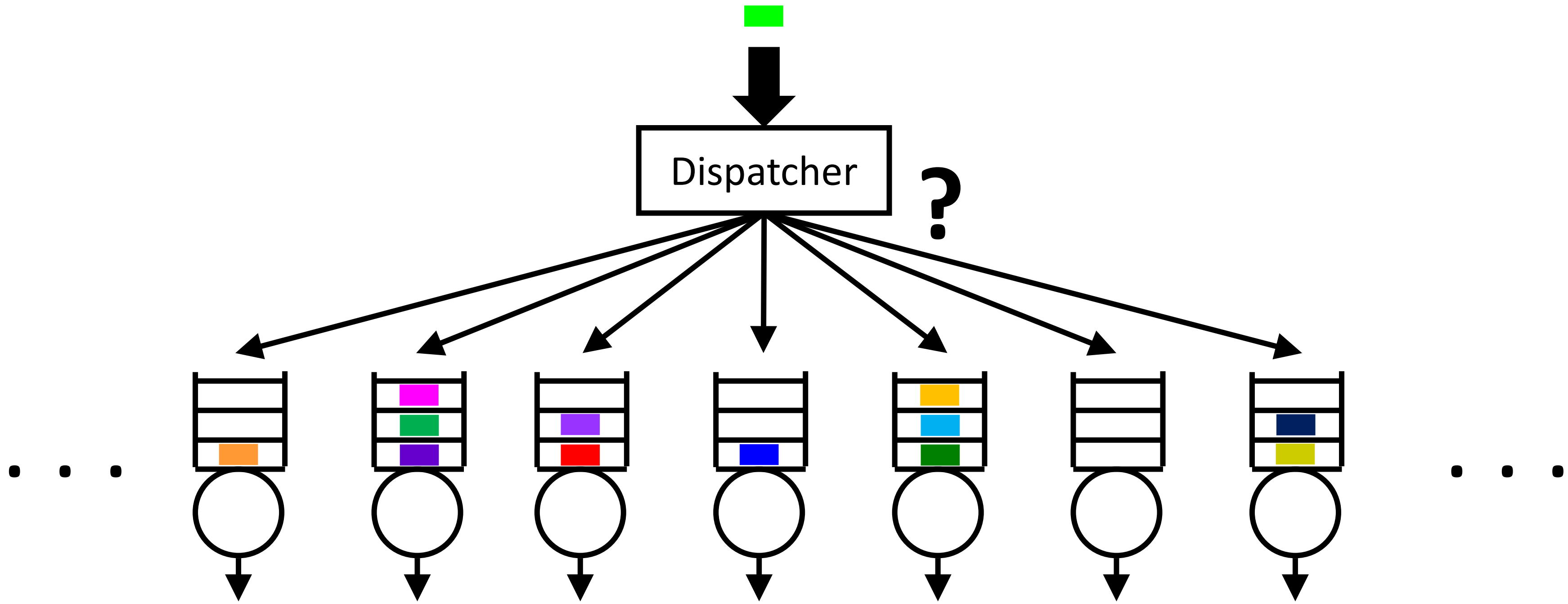
How to dispatch in multi-server systems?



Join the Shortest Queue (JSQ)

How to dispatch in multi-server systems?

large-scale

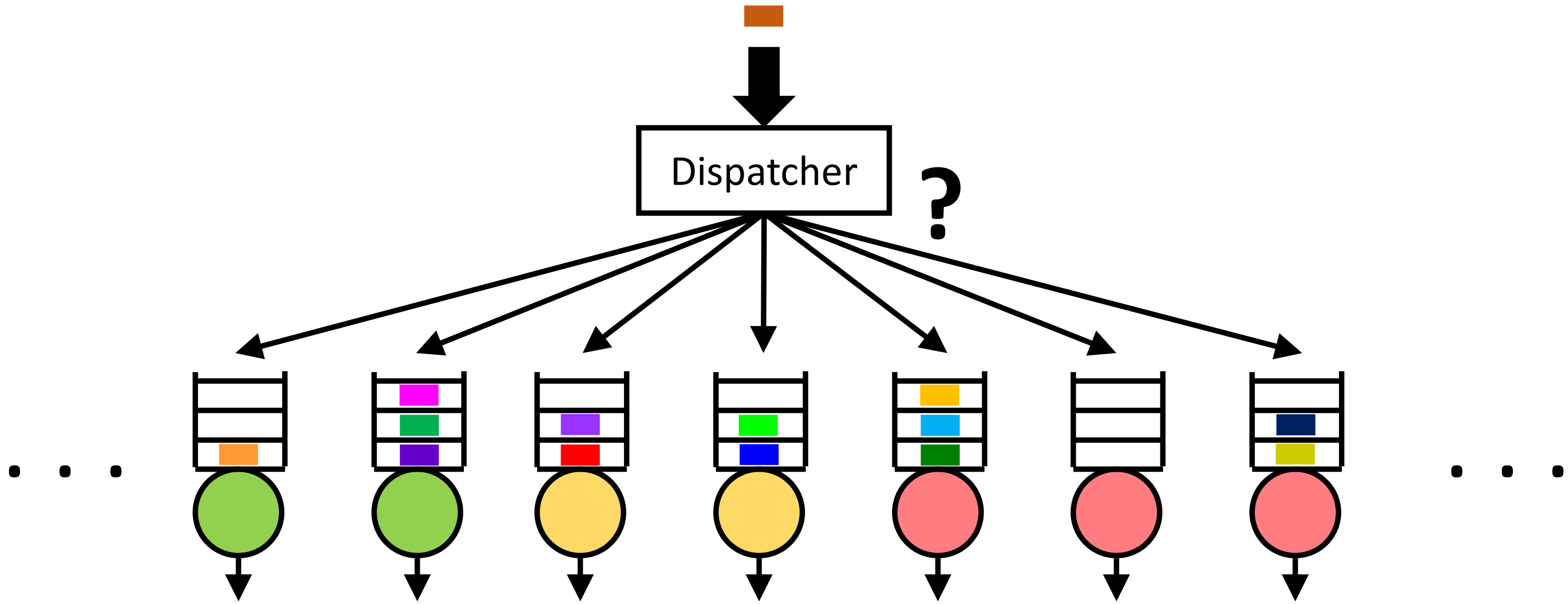


Join the Shortest Queue among d (JSQ-d)

How to dispatch in multi-server systems?

large-scale

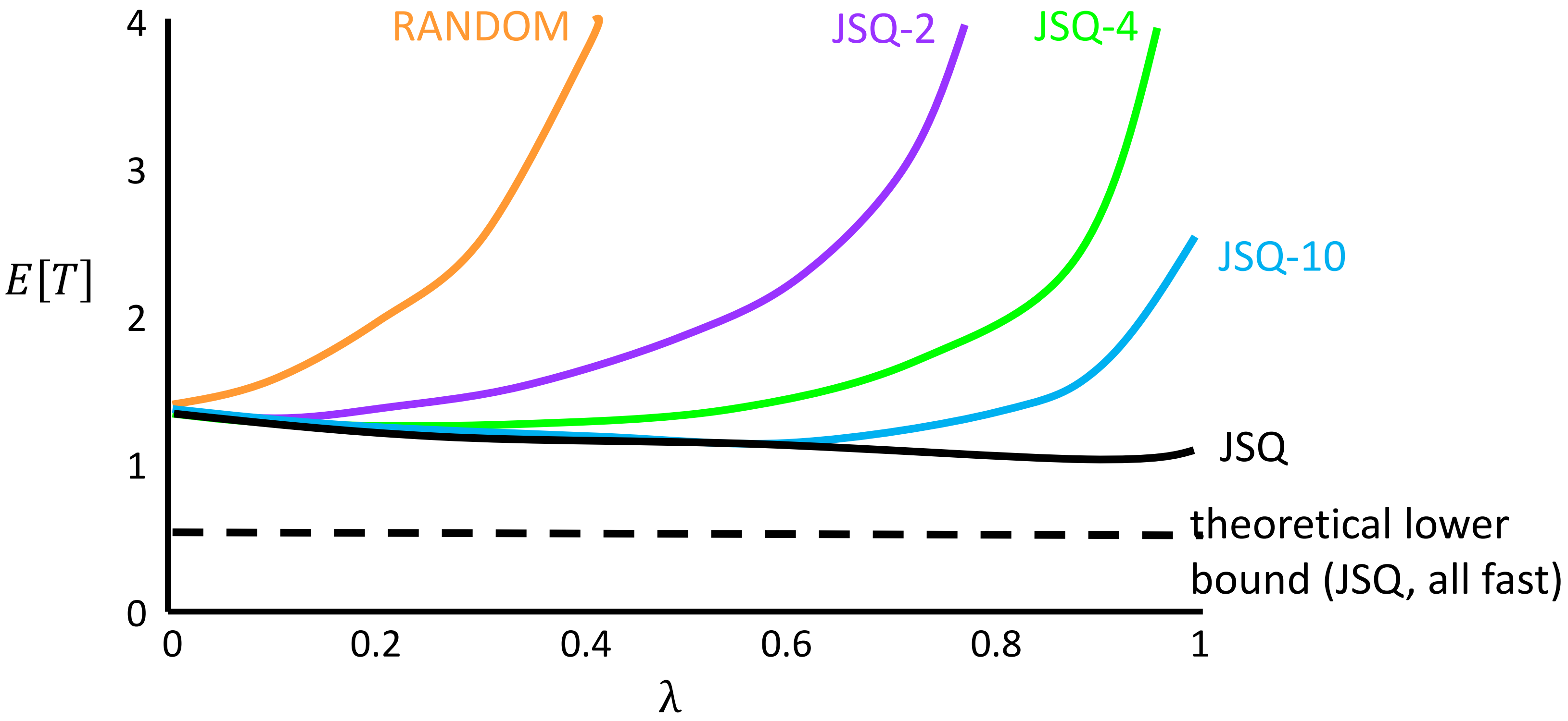
heterogeneous



How to dispatch in multi-server systems?

large-scale

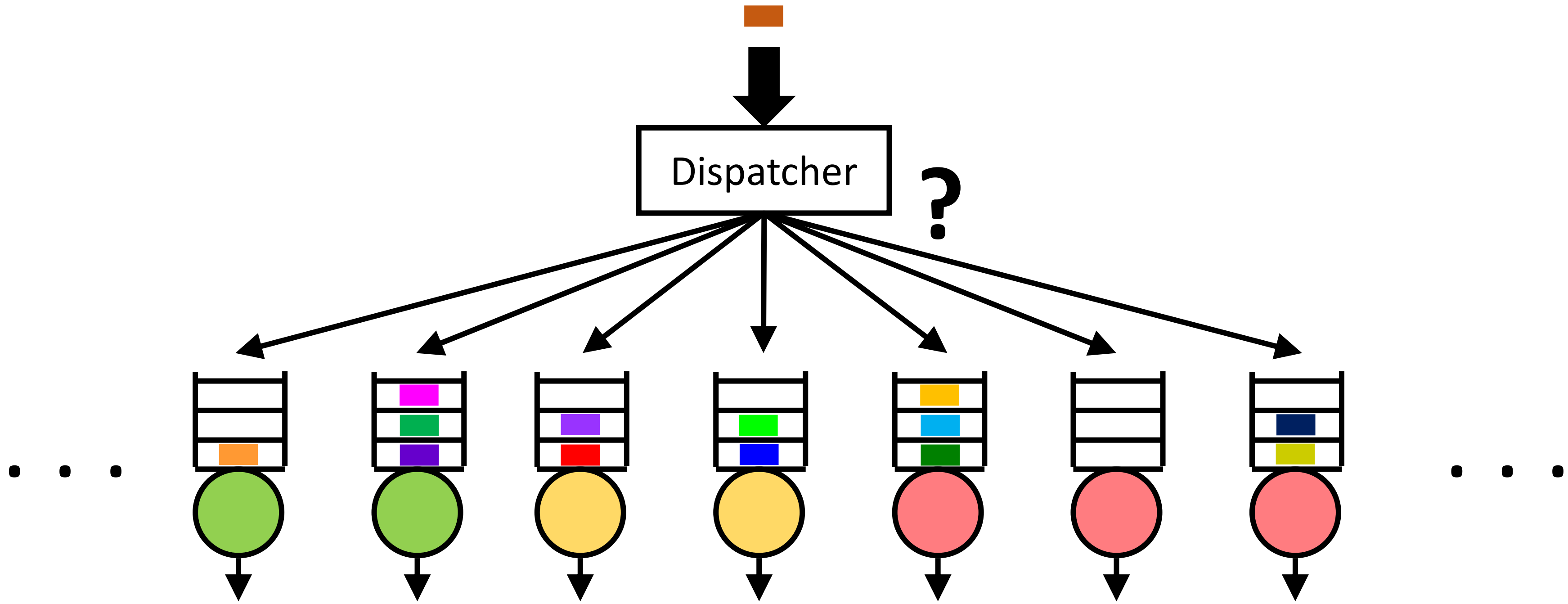
heterogeneous



300 fast servers, rate 1.95 jobs/sec
700 slow servers, rate 0.6 jobs/sec

How to dispatch in multi-server systems?

large-scale | heterogeneous

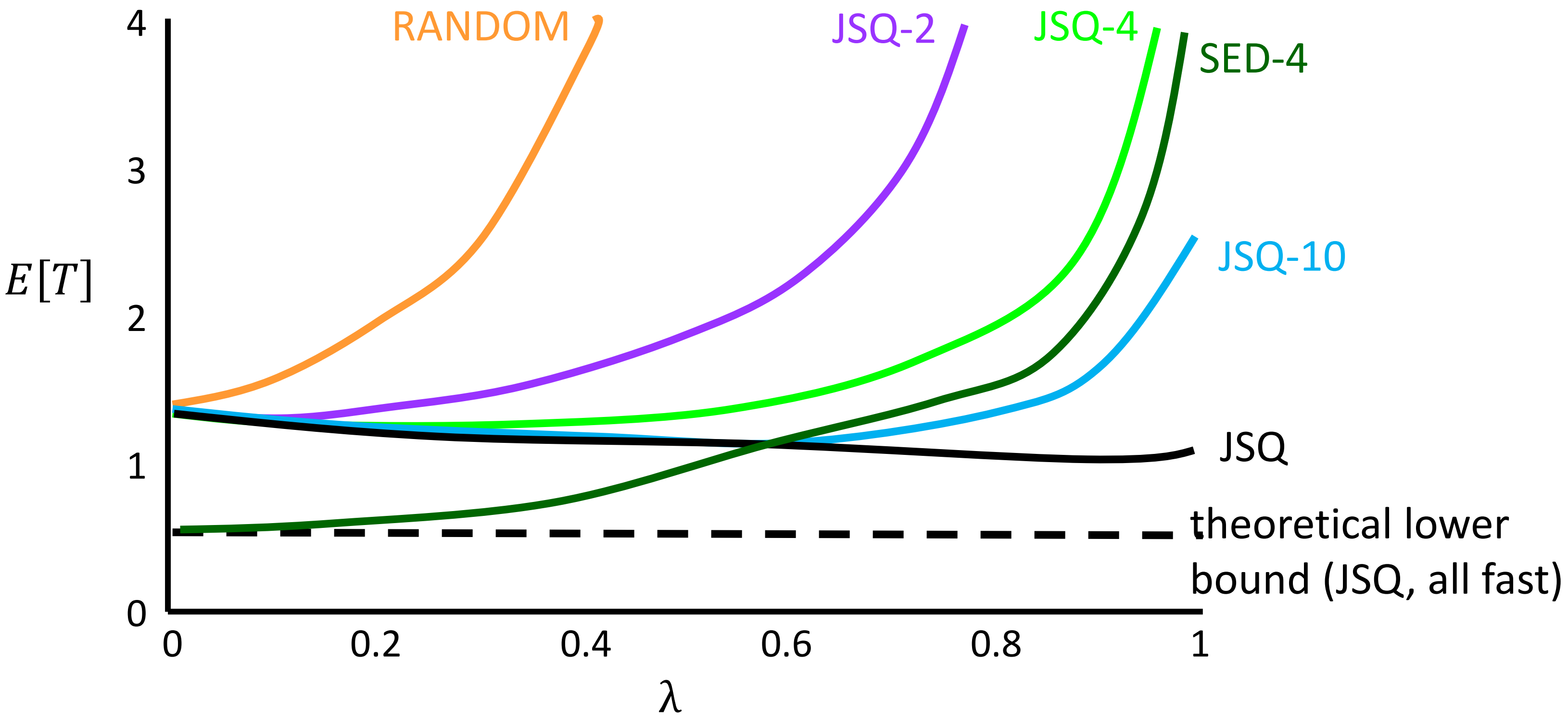


Shortest Expected Delay among d (SED-d)

How to dispatch in multi-server systems?

large-scale

heterogeneous



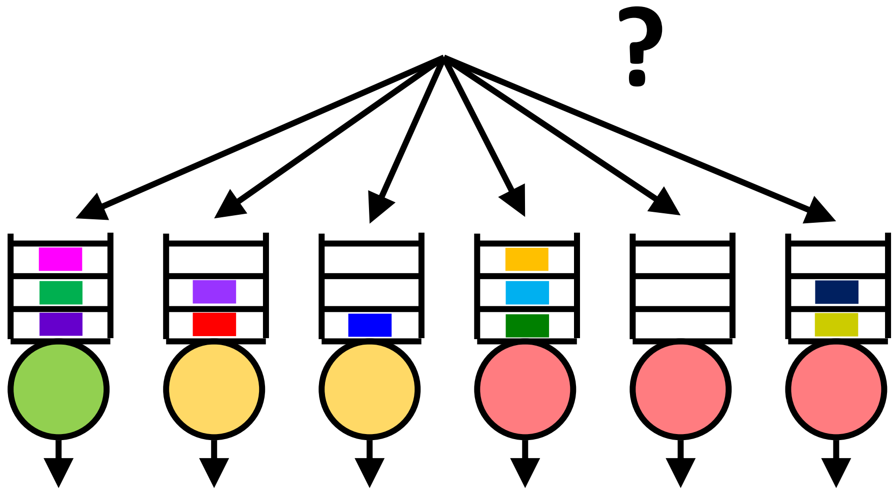
300 fast servers, rate 1.95 jobs/sec
700 slow servers, rate 0.6 jobs/sec

Leveraging Server Heterogeneity

There are **two** points at which the dispatcher can use server speed information

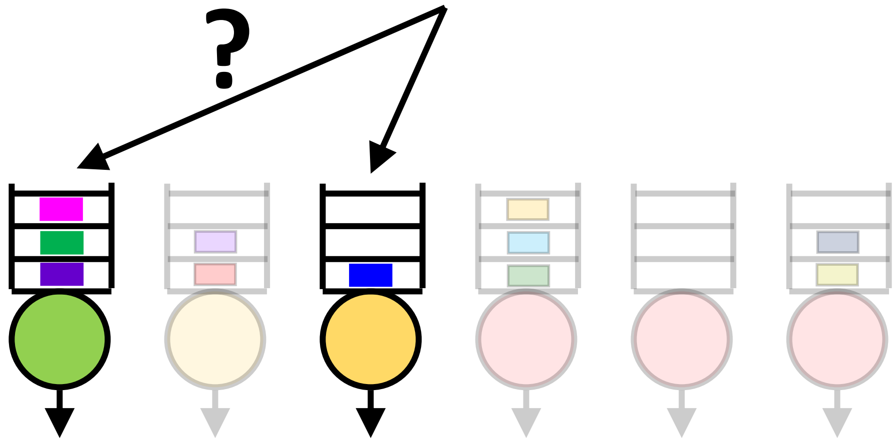
Querying Rule

Which d servers to query?

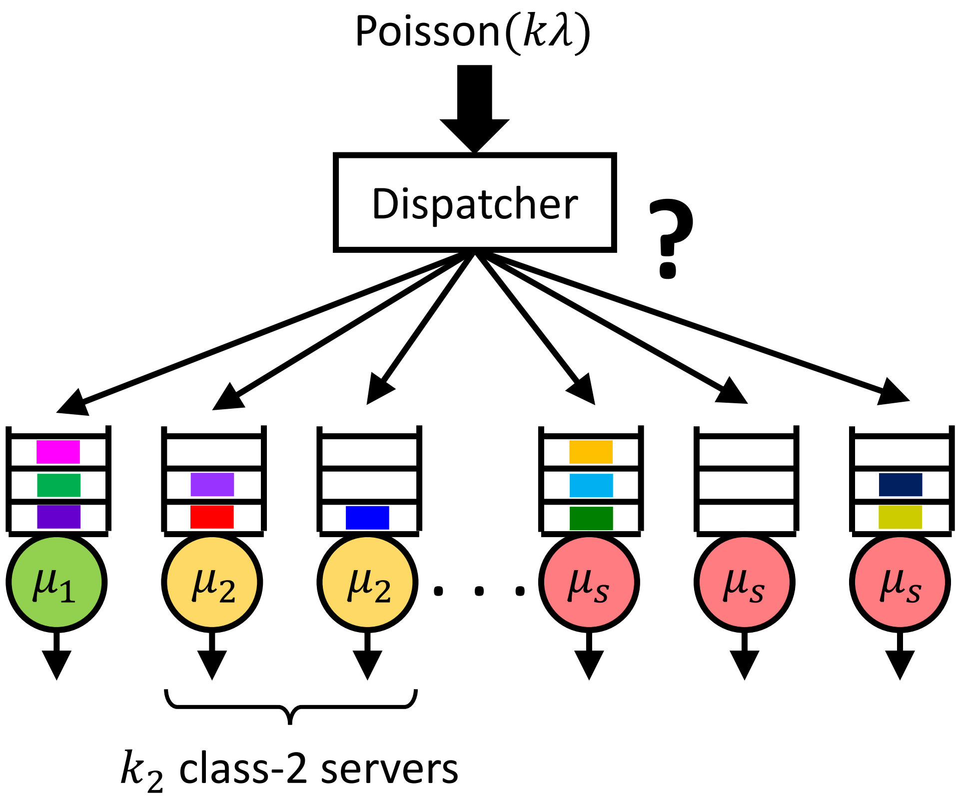


Assignment Rule

After querying, where to send the job?



System Model



Modeling Assumptions

k servers, s server speed classes $k_i = kq_i$ servers in class i

Independent service times, distribution G_i with mean $1/\mu_i$

$\mu_1 > \mu_2 > \cdots > \mu_s$

Poisson arrival process with rate $k\lambda$

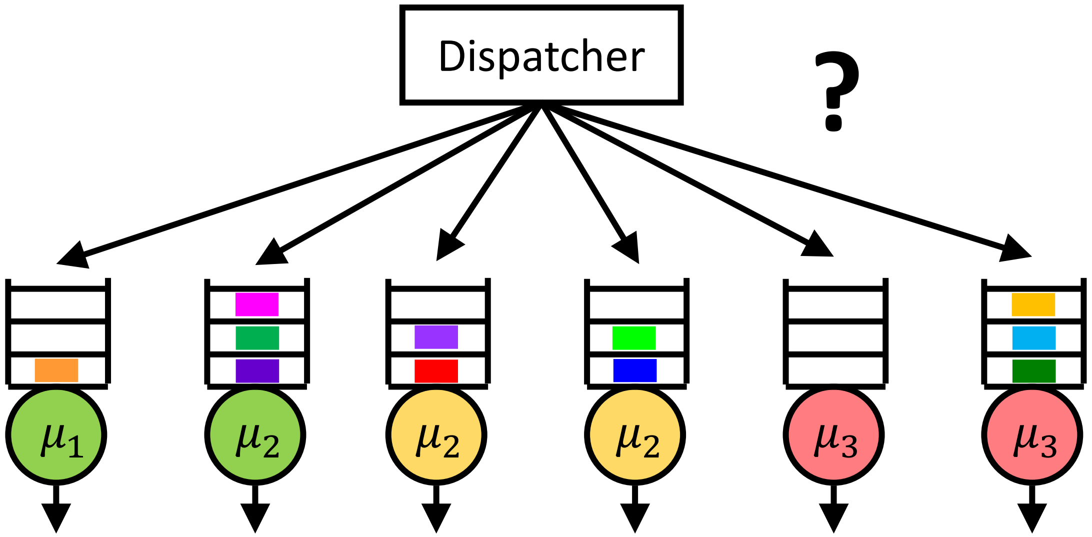
Dispatcher decides immediately to which server to send a job

Scheduling: any work-conserving policy

Goal: design dispatching policies to achieve low mean response time

Key idea: Dispatching policy = Querying rule + Assignment rule

Querying Rules



Notation

d: # of servers queried (constant, $\ll k$)

d_i: # of class-i servers queried

$\vec{\boldsymbol{d}} = (d_1, \dots, d_s)$: query mix

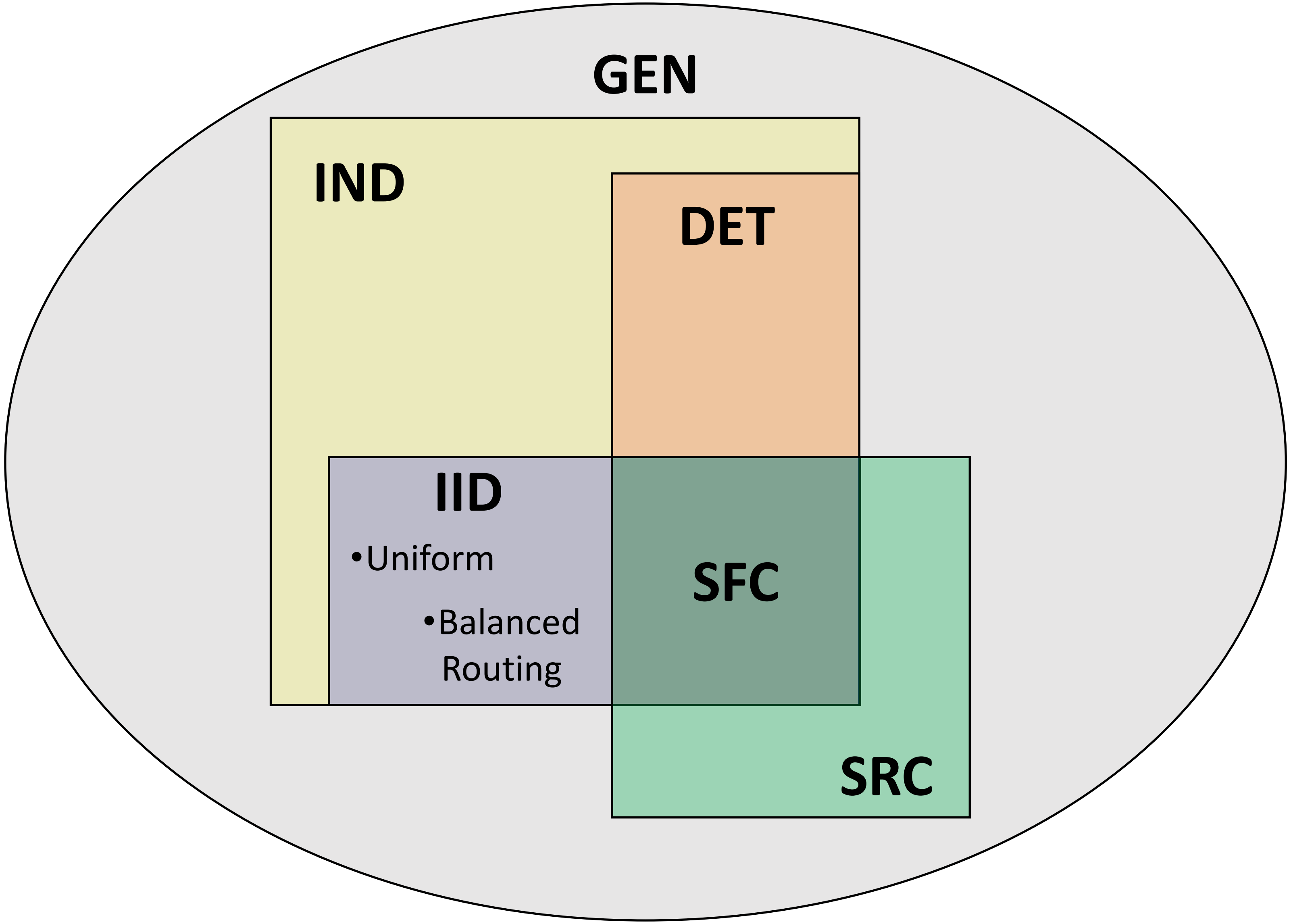
$\mathcal{D} = \{\vec{\boldsymbol{d}}: d_1 + \cdots + d_s = d\}$: set of possible query mixes

Two important properties:

- **Symmetric**: servers of the same class are treated *ex ante* identically
- **Static**: dispatcher doesn't maintain any history

Querying rule: $p(\cdot): \mathcal{D} \to [0,1]$

Querying Rules



GEN: any distribution over $\mathcal{D}$

IND: each of the d queried server classes is chosen independently

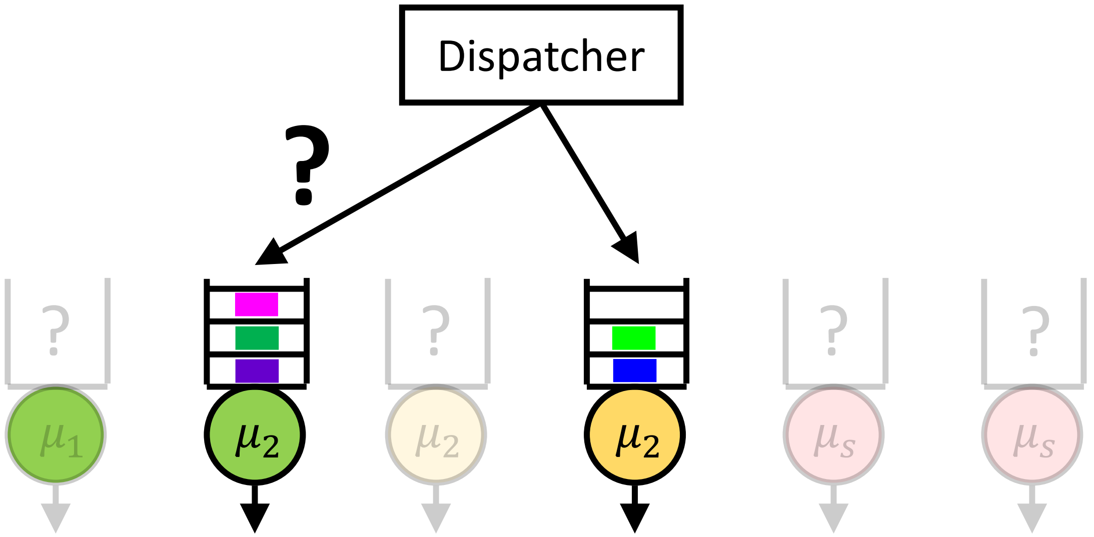
IID: each of the d queried server classes is chosen independently *from the same distribution*

DET: the same query mix is always used ($\vec{\boldsymbol{d}}$ is deterministic)

SRC: probabilistically chooses one server class, then queries d servers from that class

SFC: always queries d class-i servers for some fixed class i

Assignment Rules



Notation

$\vec{\boldsymbol{d}} = (d_1, \ldots, d_s)$: query mix

$\mathcal{D} = \{\vec{\boldsymbol{d}}: d_1 + \cdots + d_s = d\}$: set of possible query mixes

$\vec{\boldsymbol{x}}(\vec{\boldsymbol{d}})$: information obtained about the d queried servers

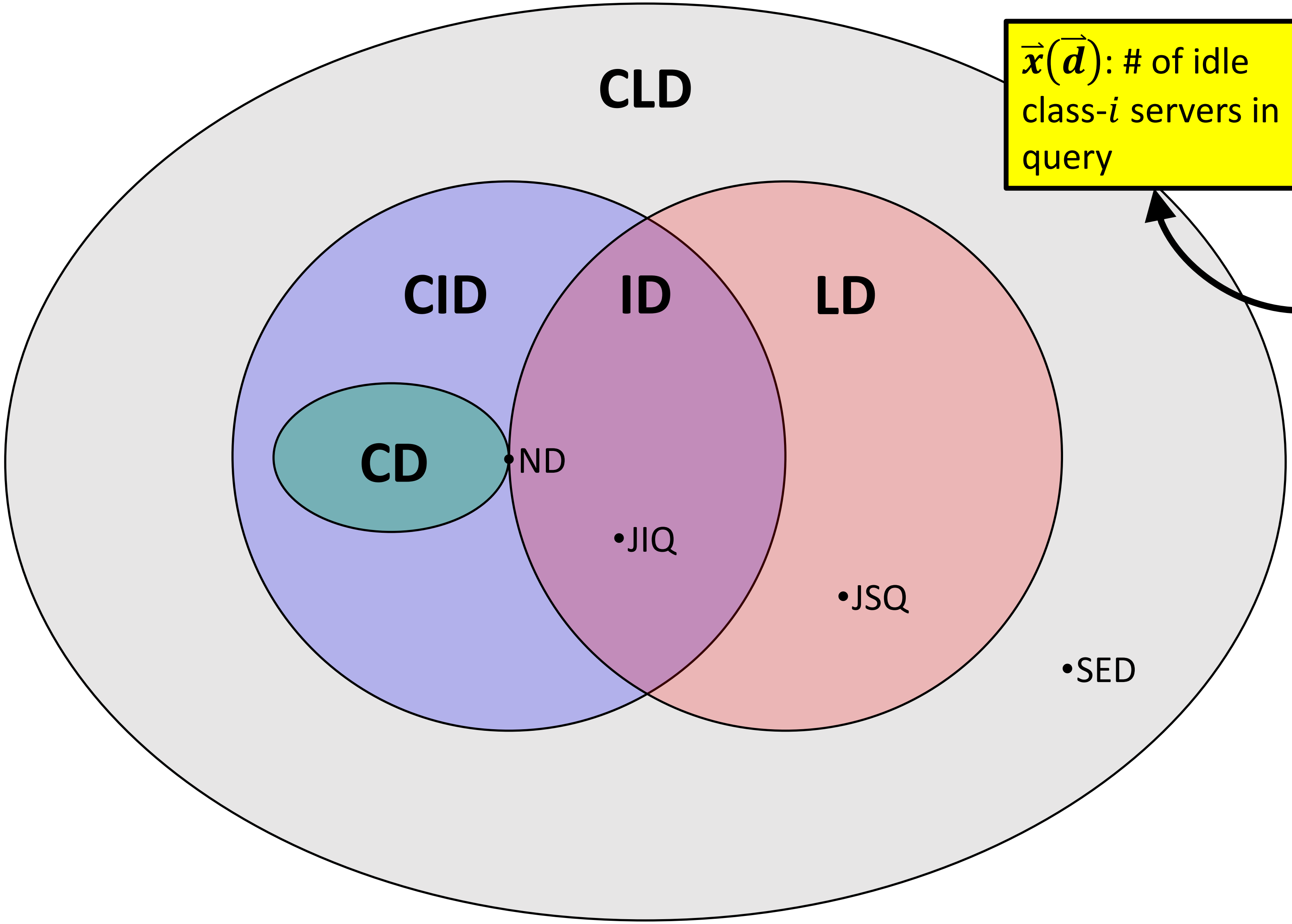
$\mathcal{X} = \{\vec{\boldsymbol{x}}(\vec{\boldsymbol{d}}): \vec{\boldsymbol{d}} \in \mathcal{D}\}$: set of possible query results

Two important properties:

- **Symmetric**: servers with the same state are treated identically
- **Static**: dispatcher doesn't maintain any history

Assignment rule: $\alpha(\cdot,\cdot): \mathcal{D} \times \mathcal{X} \to [0,1]$

Assignment Rules



$\vec{x}(\vec{d})$: # of idle class-i servers in query

CLD: can use class and queue-length information

CID: can use class and idleness information

LD: can use *only* queue-length information

ID: can use *only* idleness information

CD: can use *only* class information

Analyzing ⟨**GEN**, **CID**⟩: Approach

Goal: Find $E[T]$ given querying and assignment rules

<u>Assumptions</u>:

- $k \to \infty$, with q_i fixed
- States of all servers of the same class evolve stochastically identically
- All server states are independent

<u>Main analytic technique</u>: tag a class-i server and examine what happens whenever a new job arrives in steady-state, via mean-field analysis:

- Is the tagged server queried?
- If so, does it get the job?

Analyzing ⟨**GEN**, **CID**⟩: Preliminaries

Observation: tagged class-i server has different arrival rates when **idle** vs **busy**

- When **idle**: Poisson$\left(\lambda_i^I\right)$
- When **busy**: Poisson$\left(\lambda_i^B\right)$

Observation: once busy, server behaves like an $M/G_i/1$ with arrival rate λ_i^B

Mean busy period duration:

$$E[B_i] = \frac{1}{\mu_i - \lambda_i^B}$$

Fraction of time a server is busy:

$$\boxed{\rho_i = \frac{E[B_i]}{1/\lambda_i^I + E[B_i]} = \frac{\lambda_i^I}{\mu_i - \lambda_i^B + \lambda_i^I}}$$

Analyzing ⟨**GEN, CID**⟩: $E[T]$

Condition on the server's class:

$$E[T] = \sum_{i=1}^{s} \left(\frac{k_i(1-\rho_i)\lambda_i^I + k_i \rho_i \lambda_i^B}{k\lambda} \right) E[T_i]$$

Mean response time at a class-i $M/G_i/1$ with arrival rate λ_i^B

Any work-conserving scheduling policy!

$$= \sum_{i=1}^{s} q_i \left(\frac{(1-\rho_i)\lambda_i^I + \rho_i \lambda_i^B}{\lambda} \right) E[T_i]$$

All that remains: finding λ_i^I and λ_i^B

Analyzing $\langle \mathbf{GEN}, \mathbf{CID} \rangle$: λ_i^I and λ_i^B

$\vec{\boldsymbol{d}} = (d_1, \dots, d_s)$: query mix

Given $\vec{\boldsymbol{d}}$, probability that the tagged server is in the query: $\frac{d_i}{k_i}$

Rate at which the tagged server is queried: $\lambda k \sum_{\vec{\boldsymbol{d}} \in \mathcal{D}} p(\vec{\boldsymbol{d}}) \frac{d_i}{k_i} = \frac{\lambda}{q_i} \sum_{\vec{\boldsymbol{d}} \in \mathcal{D}} p(\vec{\boldsymbol{d}})\, d_i$

Define $r_i^I(\vec{\boldsymbol{d}})$: prob. job is sent to the tagged server, given $\vec{\boldsymbol{d}}$ and tagged server is **idle**

$r_i^B(\vec{\boldsymbol{d}})$: analogous, tagged server is **busy**

$$\boxed{\lambda_i^I = \frac{\lambda}{q_i} \sum_{\vec{\boldsymbol{d}} \in \mathcal{D}} p(\vec{\boldsymbol{d}})\, d_i r_i^I(\vec{\boldsymbol{d}})}$$

$$\boxed{\lambda_i^B = \frac{\lambda}{q_i} \sum_{\vec{\boldsymbol{d}} \in \mathcal{D}} p(\vec{\boldsymbol{d}})\, d_i r_i^B(\vec{\boldsymbol{d}})}$$

All that remains: finding $r_i^I(\vec{\boldsymbol{d}})$ and $r_i^B(\vec{\boldsymbol{d}})$

Analyzing ⟨**GEN, CID**⟩: $r_i^I(\vec{\boldsymbol{d}})$

$r_i^I(\vec{\boldsymbol{d}})$: prob. job is sent to the tagged server, given $\vec{\boldsymbol{d}}$ and tagged server is **idle**

Define $b_i(\vec{\boldsymbol{d}})$: probability that all faster queried servers are busy

$$b_i(\vec{\boldsymbol{d}}) = \prod_{j=1}^{i-1} {\rho_j}^{d_j}$$

Job is then assigned to a class-i server w.p. $\alpha_i(i, \vec{\boldsymbol{d}})$

$$\boxed{r_i^I(\vec{\boldsymbol{d}}) = b_i(\vec{\boldsymbol{d}})\alpha_i(i, \vec{\boldsymbol{d}}) \sum_{a_i=1}^{d_i} \binom{d_i - 1}{a_i - 1} \frac{(1-\rho_i)^{a_i - 1} {\rho_i}^{d_i - a_i}}{a_i}}$$

Analyzing ⟨**GEN**, **CID**⟩: Putting it all together

Solve the system of equations:

$$\boxed{\rho_i = \frac{\lambda_i^I}{\mu_i - \lambda_i^B + \lambda_i^I}}$$

$$\boxed{\lambda_i^I = \frac{\lambda}{q_i} \sum_{\vec{\boldsymbol{d}} \in \mathcal{D}} p(\vec{\boldsymbol{d}})\, d_i r_i^I(\vec{\boldsymbol{d}})}$$

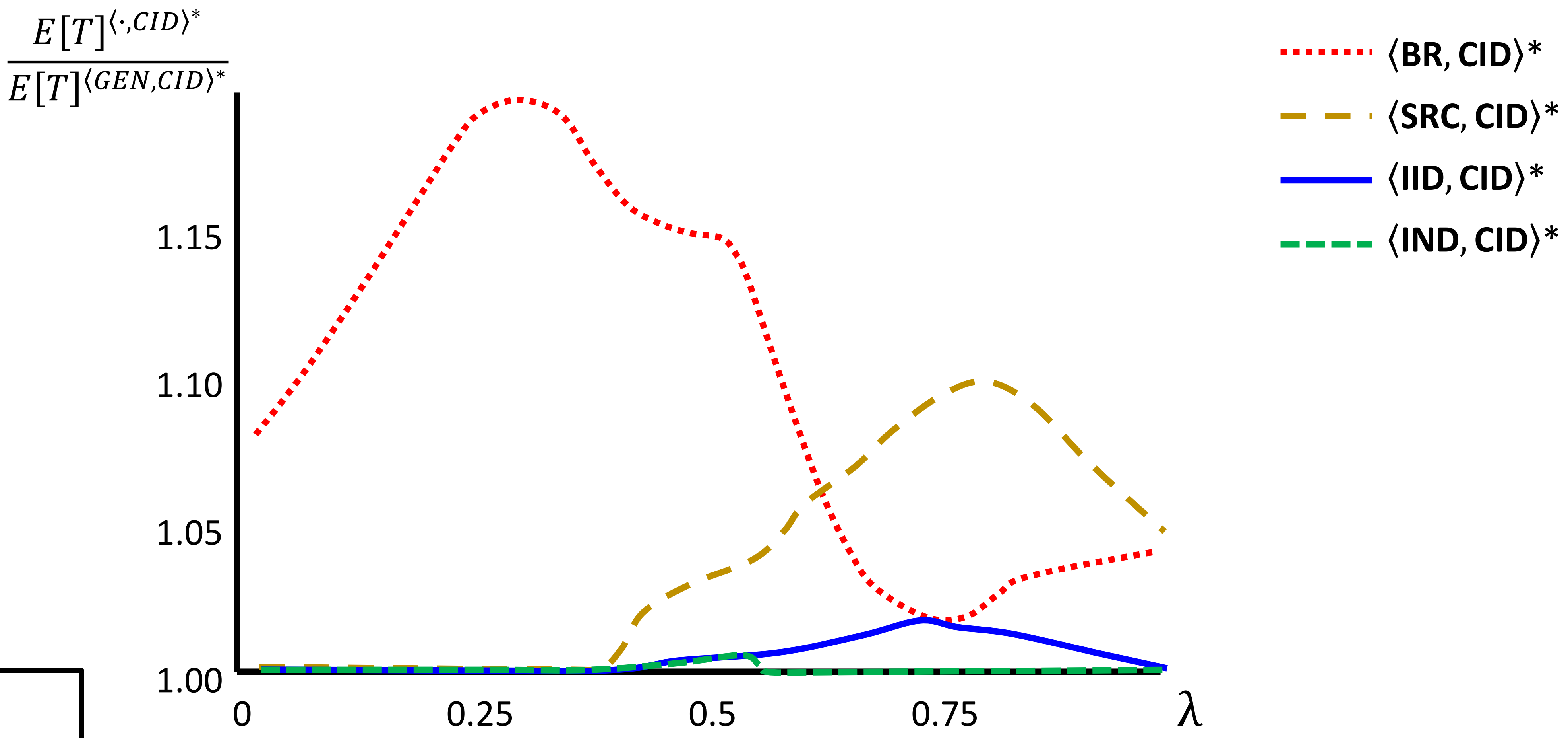
$$\boxed{\lambda_i^B = \frac{\lambda}{q_i} \sum_{\vec{\boldsymbol{d}} \in \mathcal{D}} p(\vec{\boldsymbol{d}})\, d_i r_i^B(\vec{\boldsymbol{d}})}$$

$$\boxed{r_i^I(\vec{\boldsymbol{d}}) = \alpha_i(i, \vec{\boldsymbol{d}}) b_i(\vec{\boldsymbol{d}}) \sum_{a_i=1}^{d_i} \binom{d_i - 1}{a_i - 1} \frac{(1-\rho_i)^{a_i - 1} \rho_i{}^{d_i - a_i}}{a_i}}$$

$$\boxed{r_i^B(\vec{\boldsymbol{d}}) = \frac{\alpha_i(0, \vec{\boldsymbol{d}})}{d_i} \cdot \frac{1}{\rho_i} \prod_{j=1}^{s} \rho_j{}^{d_j} + \sum_{j=i+1}^{s} \frac{\alpha_i(j, \vec{\boldsymbol{d}}) b_j(\vec{\boldsymbol{d}}) (1-\rho_j)^{d_j}}{d_i \rho_i}}$$

$$E[T] = \sum_{i=1}^{s} q_i \left(\frac{(1-\rho_i)\lambda_i^I + \rho_i \lambda_i^B}{\lambda} \right) E[T_i]$$

Numerical Results: $\langle \cdot, \mathbf{CID} \rangle$

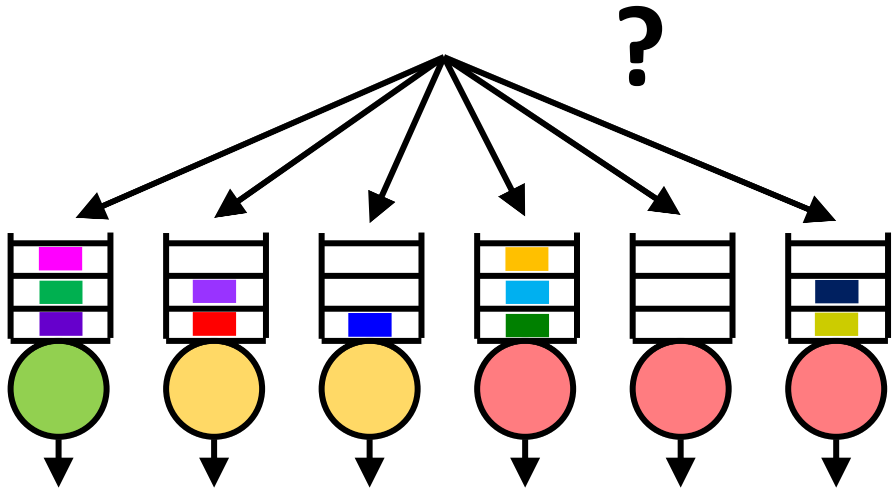


$s = 3$
$d = 3$
$(q_1, q_2, q_3) = \left(\frac{1}{3}, \frac{1}{6}, \frac{1}{2}\right)$
$(\mu_1, \mu_2, \mu_3) = \left(2, \frac{4}{5}, \frac{2}{5}\right)$

Conclusion

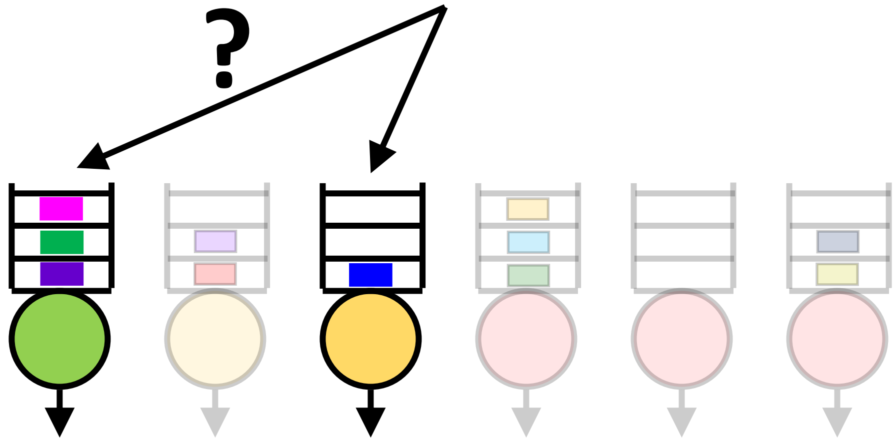
Querying Rule

Which d servers to query?



Assignment Rule

After querying, where to send the job?



Two Quick Advertisements…

- Amherst College Computer Science is hiring!

- I'll be visiting the Netherlands for (some part of) the upcoming academic year!

kgardner@amherst.edu