On the Power of (even a little) Centralization in Distributed Processing

Kuang Xu
Joint work with John N. Tsitsiklis

Massachusetts Institute of Technology

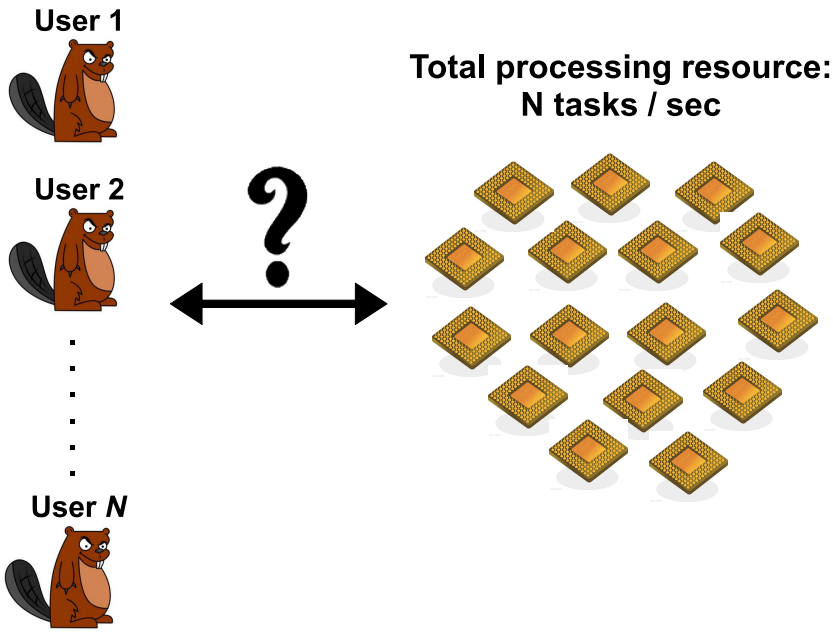
ACM Sigmetrics
June 10, 2011

Outline

Outline

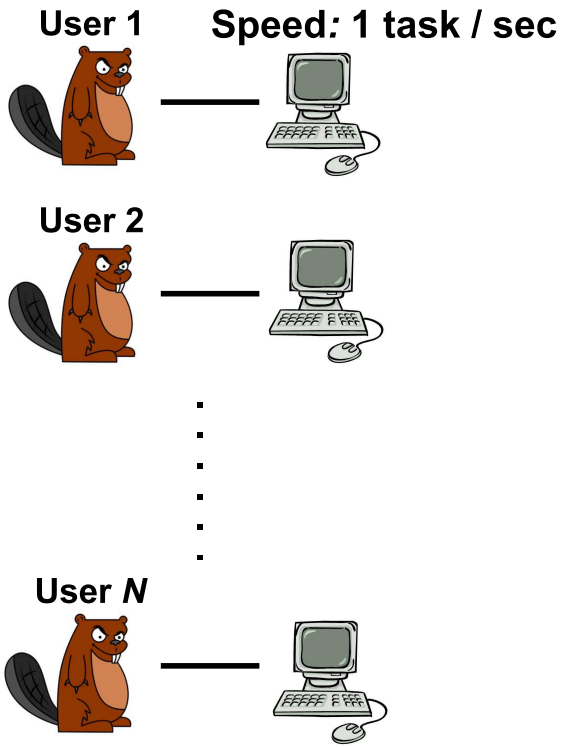
Centralized or Distributed?

How to allocate N divisible processing resources to N users.



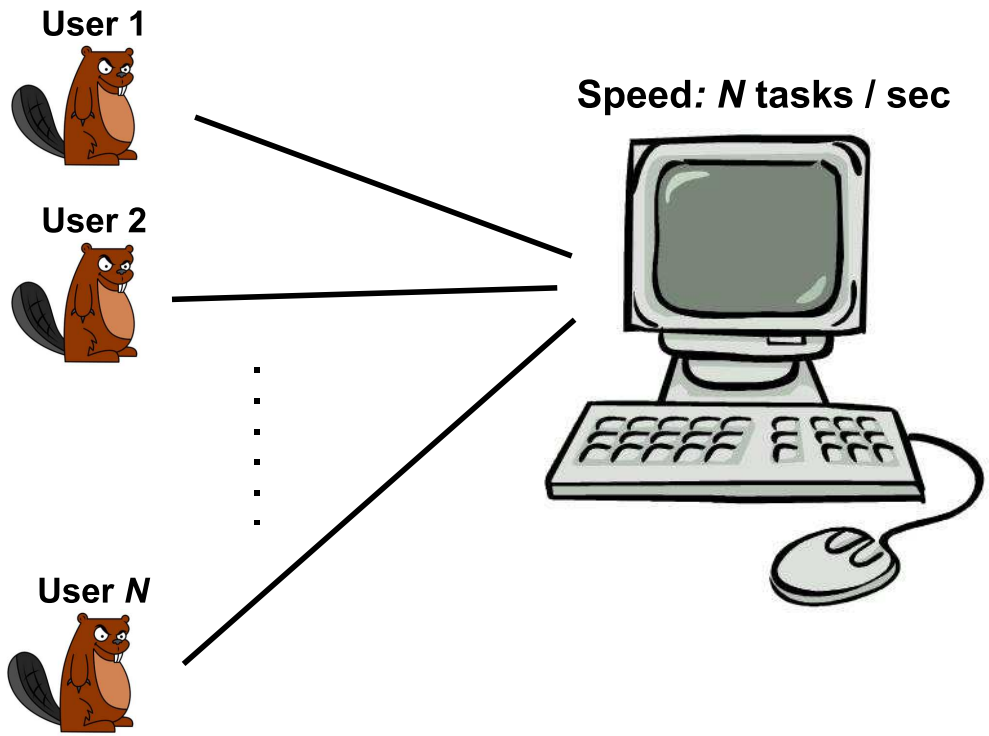
Centralized or Distributed?

Option 1: **fully distributed** (e.g., 1 PC per user).



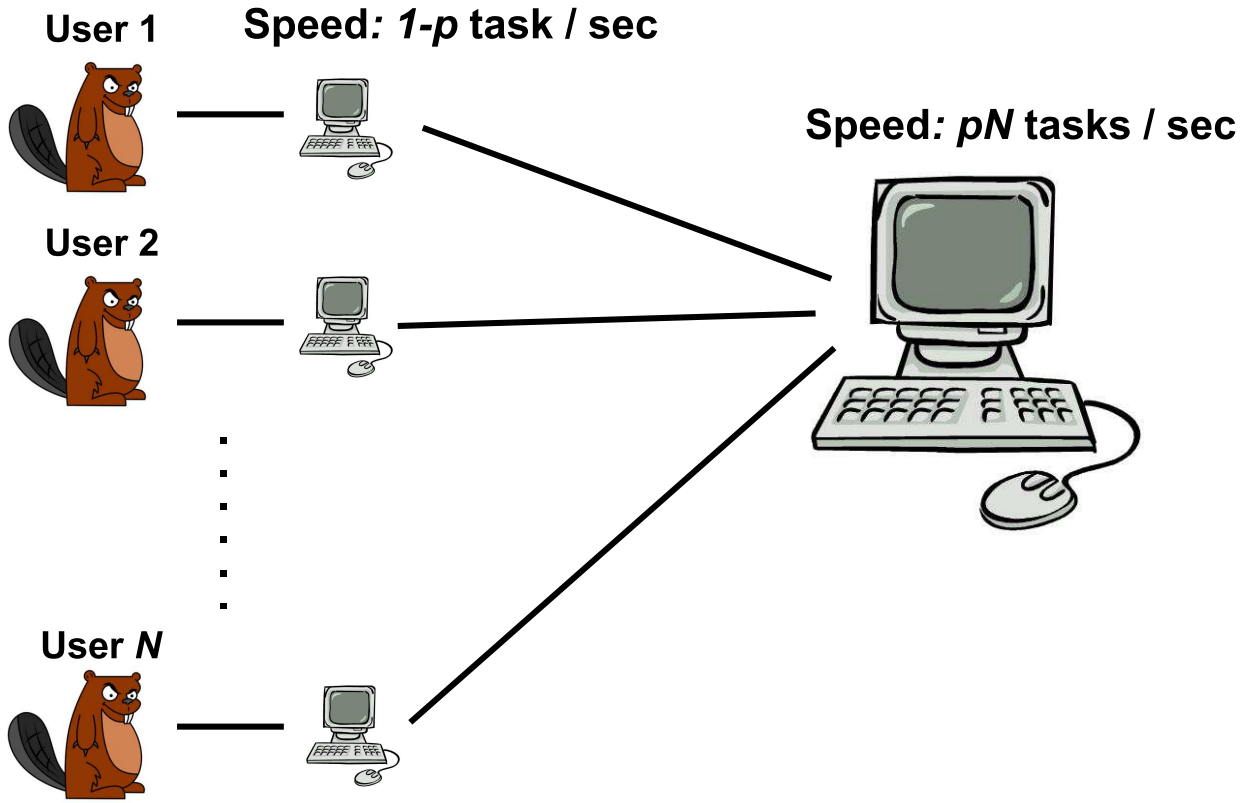
Centralized or Distributed?

Option 2: **fully centralized** (e.g. Cloud, mainframe)



Centralized or Distributed?

Option 3: $(1-p)N$ distributed, pN centralized $(0 < p < 1)$.



Centralized or Distributed?

Trade-off:

- More centralization (larger p) $\Rightarrow$ better utilization of resources.
- But also higher infrastructure and communications costs!

Centralized or Distributed?

This talk:

- Understand **performance impact of centralization** in distributed processing and resource allocation.
- Main message: even a **small amount of centralization** is significant!

Centralized or Distributed?

This talk:

- Understand **performance impact of centralization** in distributed processing and resource allocation.
- Main message: even a **small amount of centralization** is significant!

Motivations [1]

- Applications
  - Infrastructure planning and design of **server farms** and **cloud clusters**.
  - Scheduling with **limited state information**.
- A building block for finding optimal performance v.s. costs trade-off

[1] image source: http://www.opensolutions.ie/

Motivations [1]

- Applications
  - Infrastructure planning and design of **server farms** and **cloud clusters**.
  - Scheduling with **limited state information**.
- A building block for finding optimal performance v.s. costs trade-off

[1] image source: http://www.opensolutions.ie/

Motivations [1]

- Applications
  - Infrastructure planning and design of **server farms** and **cloud clusters**.
  - Scheduling with **limited state information**.
- A building block for finding optimal performance v.s. costs trade-off

[1] image source: http://www.opensolutions.ie/

Motivations [1]

- Applications
  - Infrastructure planning and design of **server farms** and **cloud clusters**.
  - Scheduling with **limited state information**.
- A building block for finding optimal performance v.s. costs trade-off

[1] image source: http://www.opensolutions.ie/

Related Work

Small coordination makes a big difference...

- Flexible routing to two queues
  [Foschini and Salz 78]
  - Routing a single stream to two stations
  - Constant factor delay improvement when having flexible customers (under diffusion heavy-traffic scaling).
- Super-market model in load-balancing ("power of two choices")
  [Vvedenskaya et al. 96] [Mitzenmacher 96] [Bramson et al. 2010]
  - Routing a single stream to N parallel stations.
  - Substantial benefits by routing to the shorter queue between two randomly chosen queues.
- Our work: also consider partial coordination, but very different in model, dynamics and analysis.

Related Work

Small coordination makes a big difference...

- Flexible routing to two queues
  [Foschini and Salz 78]
  - Routing a single stream to two stations
  - Constant factor delay improvement when having flexible customers (under diffusion heavy-traffic scaling).
- Super-market model in load-balancing ("power of two choices")
  [Vvedenskaya et al. 96] [Mitzenmacher 96] [Bramson et al. 2010]
  - Routing a single stream to N parallel stations.
  - Substantial benefits by routing to the shorter queue between two randomly chosen queues.
- Our work: also consider partial coordination, but very different in model, dynamics and analysis.

Related Work

Small coordination makes a big difference...

- Flexible routing to two queues
  [Foschini and Salz 78]
  - Routing a single stream to two stations
  - Constant factor delay improvement when having flexible customers (under diffusion heavy-traffic scaling).
- Super-market model in load-balancing ("power of two choices")
  [Vvedenskaya et al. 96] [Mitzenmacher 96] [Bramson et al. 2010]
  - Routing a single stream to N parallel stations.
  - Substantial benefits by routing to the shorter queue between two randomly chosen queues.
- Our work: also consider partial coordination, but very different in model, dynamics and analysis.

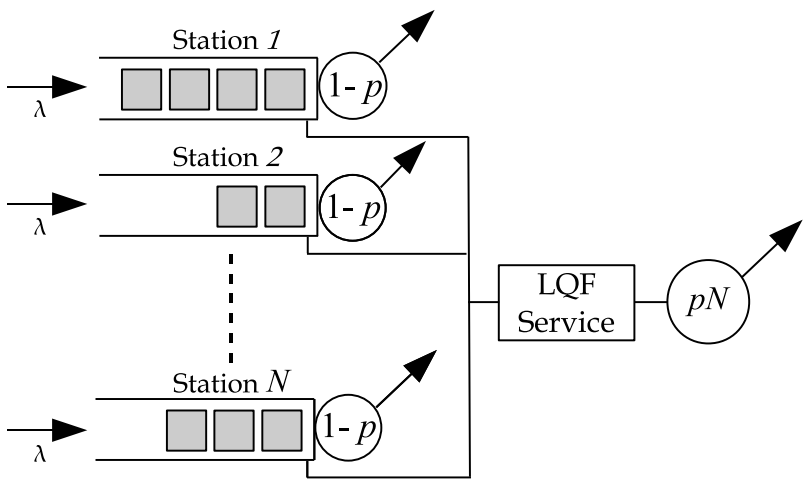
Related Work

Small coordination makes a big difference...

- Flexible routing to two queues
  [Foschini and Salz 78]
  - Routing a single stream to two stations
  - Constant factor delay improvement when having flexible customers (under diffusion heavy-traffic scaling).
- Super-market model in load-balancing ("power of two choices")
  [Vvedenskaya et al. 96] [Mitzenmacher 96] [Bramson et al. 2010]
  - Routing a single stream to N parallel stations.
  - Substantial benefits by routing to the shorter queue between two randomly chosen queues.
- Our work: also consider partial coordination, but very different in model, dynamics and analysis.

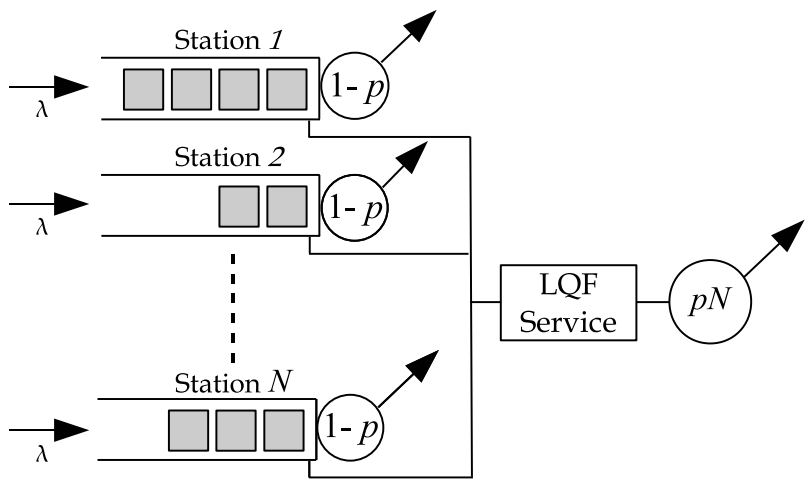
Model: Server Farm with Local and Central Servers

- Time is continuous.
- Arrivals and departures modeled by independent Poisson processes.
  - Events = clock ticks.
- N parallel stations. One queue per station to store unprocessed jobs: $Q_i(t)$.
- **System designer** chooses $p \in [0, 1]$.



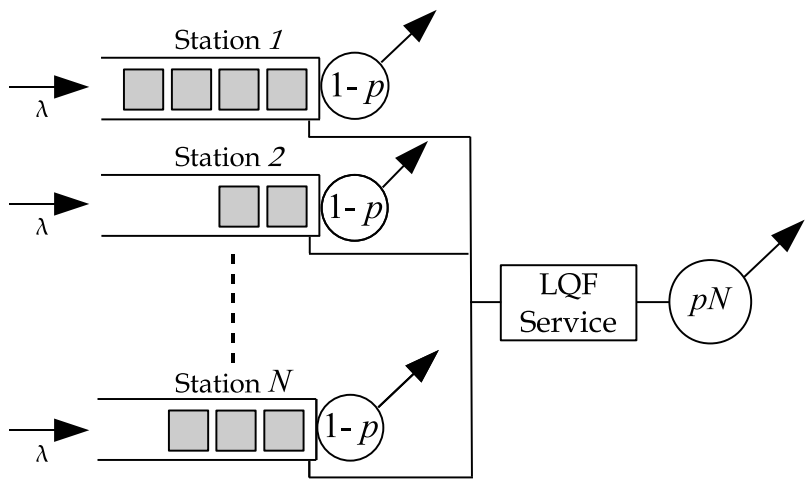
Model: Server Farm with Local and Central Servers

- Time is continuous.
- Arrivals and departures modeled by independent Poisson processes.
  - Events = clock ticks.
- N parallel stations. One queue per station to store unprocessed jobs: $Q_i(t)$.
- **System designer** chooses $p \in [0, 1]$.



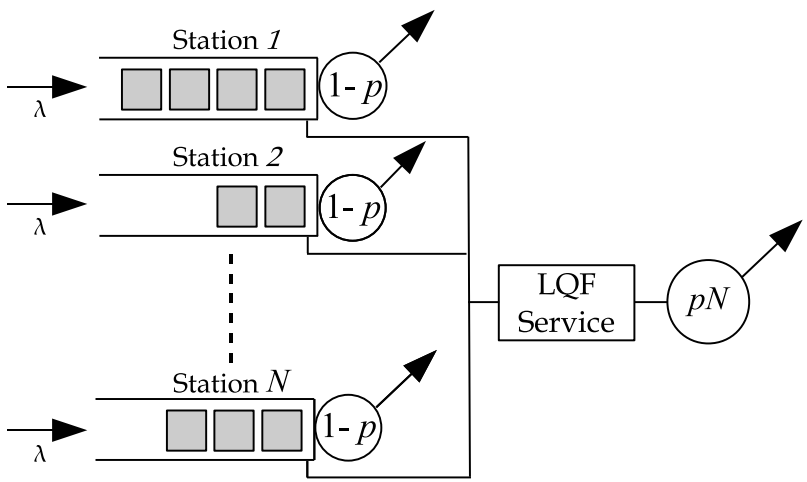
Model: Server Farm with Local and Central Servers

- Time is continuous.
- Arrivals and departures modeled by independent Poisson processes.
  - Events = clock ticks.
- N parallel stations. One queue per station to store unprocessed jobs: $Q_i(t)$.
- **System designer** chooses $p \in [0, 1]$.

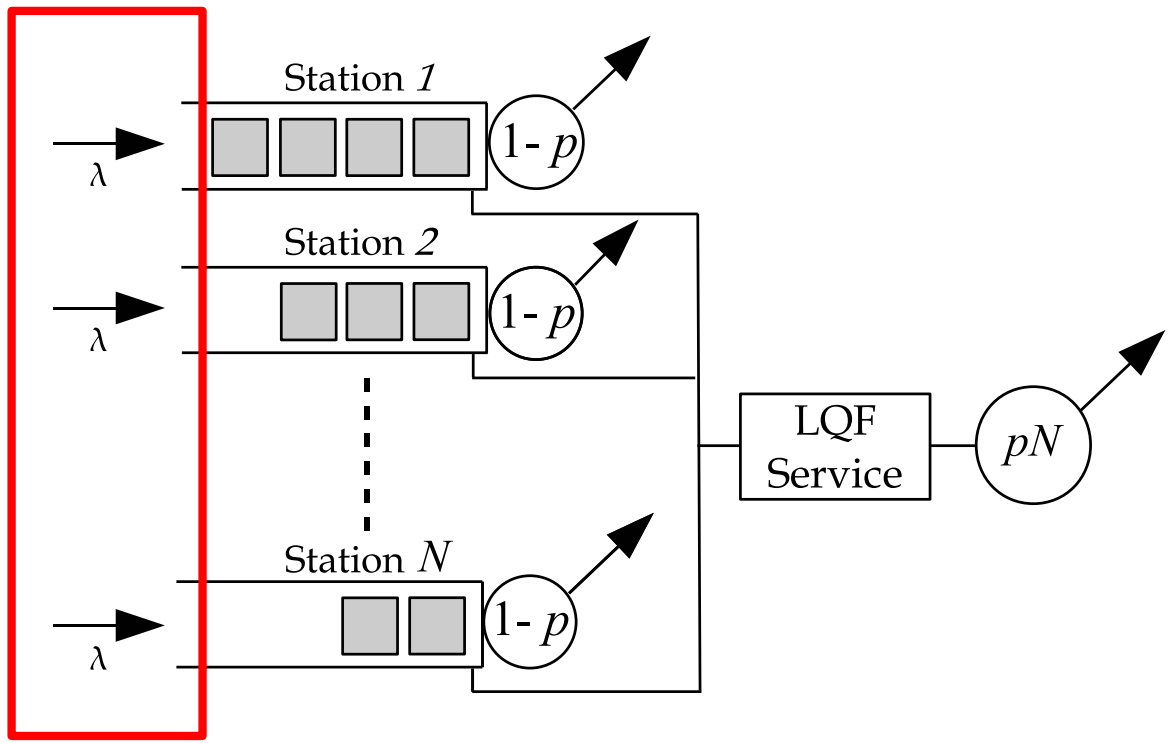


Model: Server Farm with Local and Central Servers

- Time is continuous.
- Arrivals and departures modeled by independent Poisson processes.
  - Events = clock ticks.
- N parallel stations. One queue per station to store unprocessed jobs: $Q_i(t)$.
- **System designer** chooses $p \in [0, 1]$.

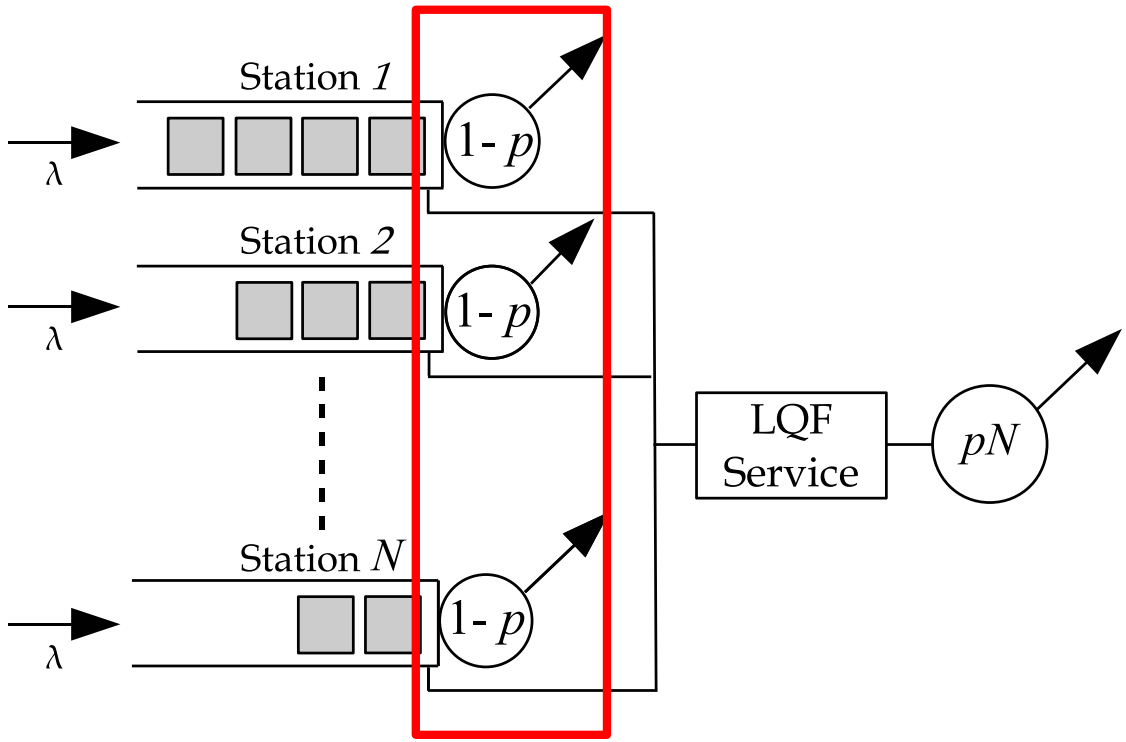


Model: Server Farm with Local and Central Servers



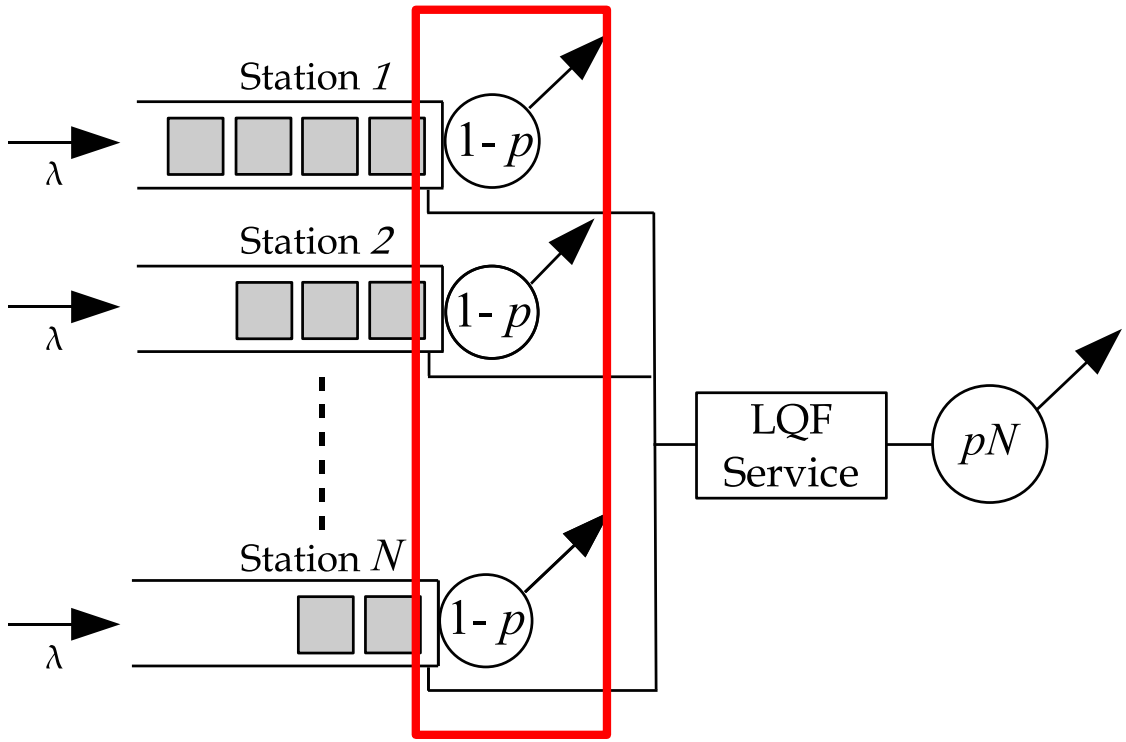
- **Job arrivals**: λ jobs / sec to each station, $0 \leq \lambda < 1$.

Model: Server Farm with Local and Central Servers



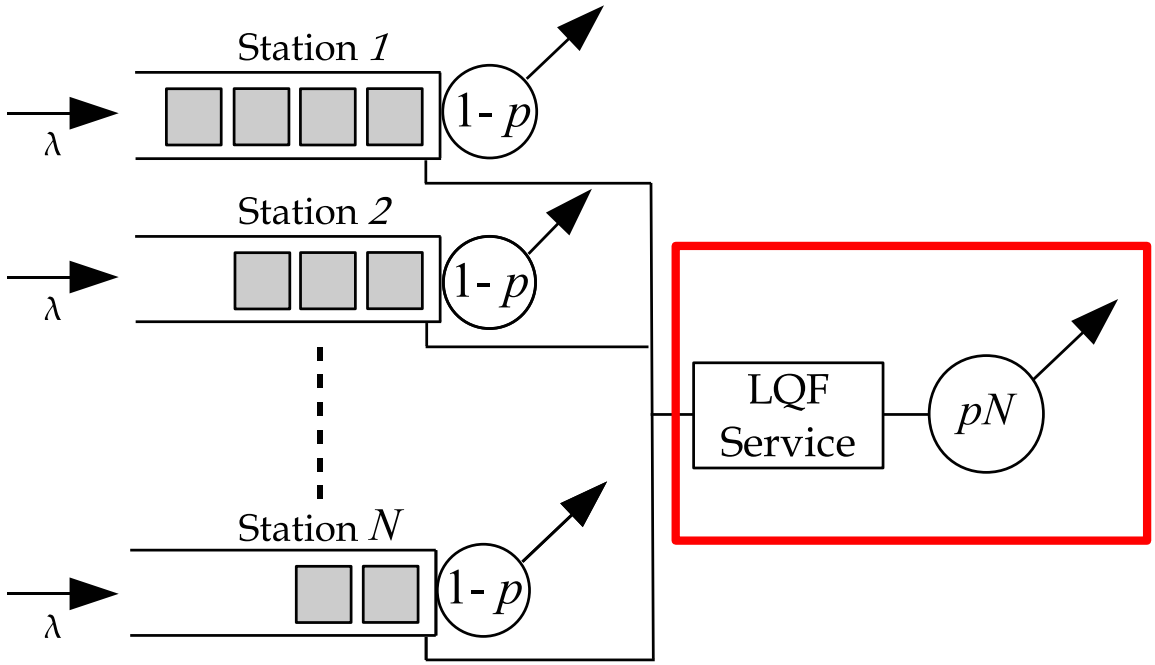
- N **local servers**, working at $1 - p$ job / sec.
- Each serves the **assigned station only**.

Model: Server Farm with Local and Central Servers



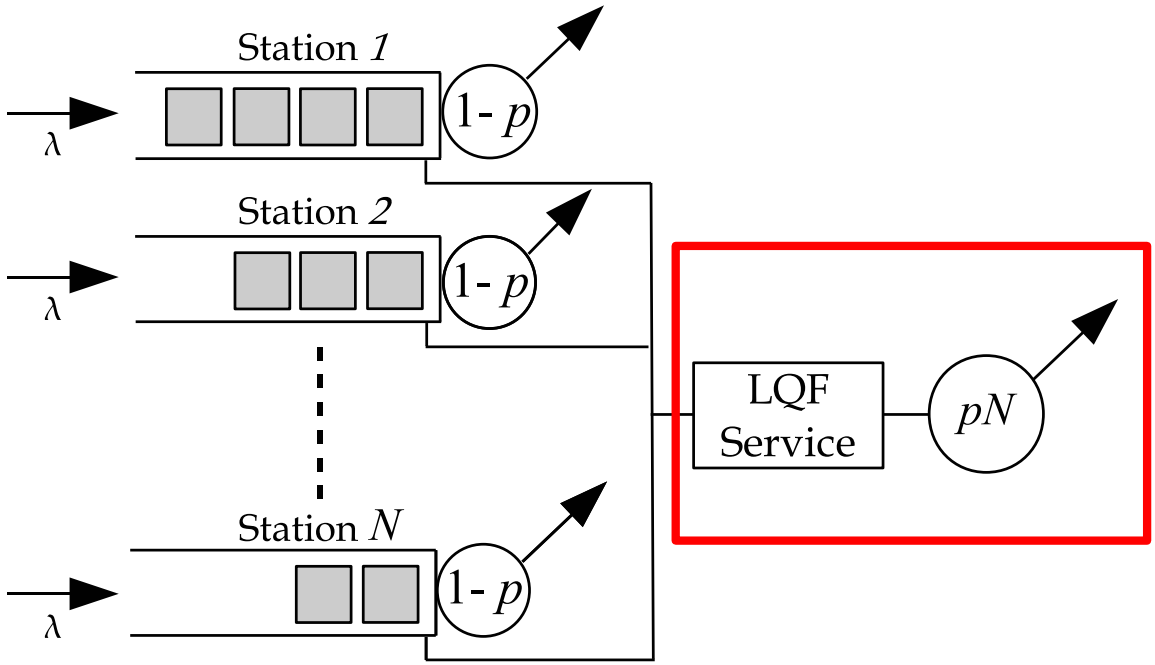
- N **local servers**, working at $1 - p$ job / sec.
- Each serves the **assigned station only**.

Model: Server Farm with Local and Central Servers



- **One central server**, working at pN jobs / sec.
- Serves a job from a **most loaded queue** whenever it becomes available.

Model: Server Farm with Local and Central Servers



- **One central server**, working at pN jobs / sec.
- Serves a job from a **most loaded queue** whenever it becomes available.

Performance Metric: Average Queue Length

- **Performance metric**: the **average queue length** at time t

$$Q_{avg}(t) = \frac{\text{total number of jobs at time } t}{N}$$

- Q_{avg} closely related to **average waiting time** (delay) by Little's Law.

Simple Observations

Some simple observations before calculations:

- If $p = 0$, system degenerates into N separate $M/M/1$ queues.
- Classical queuing theory: delay scales as $\frac{1}{1-\lambda}$ as $\lambda \to 1$.

Simple Observations

Some simple observations before calculations:

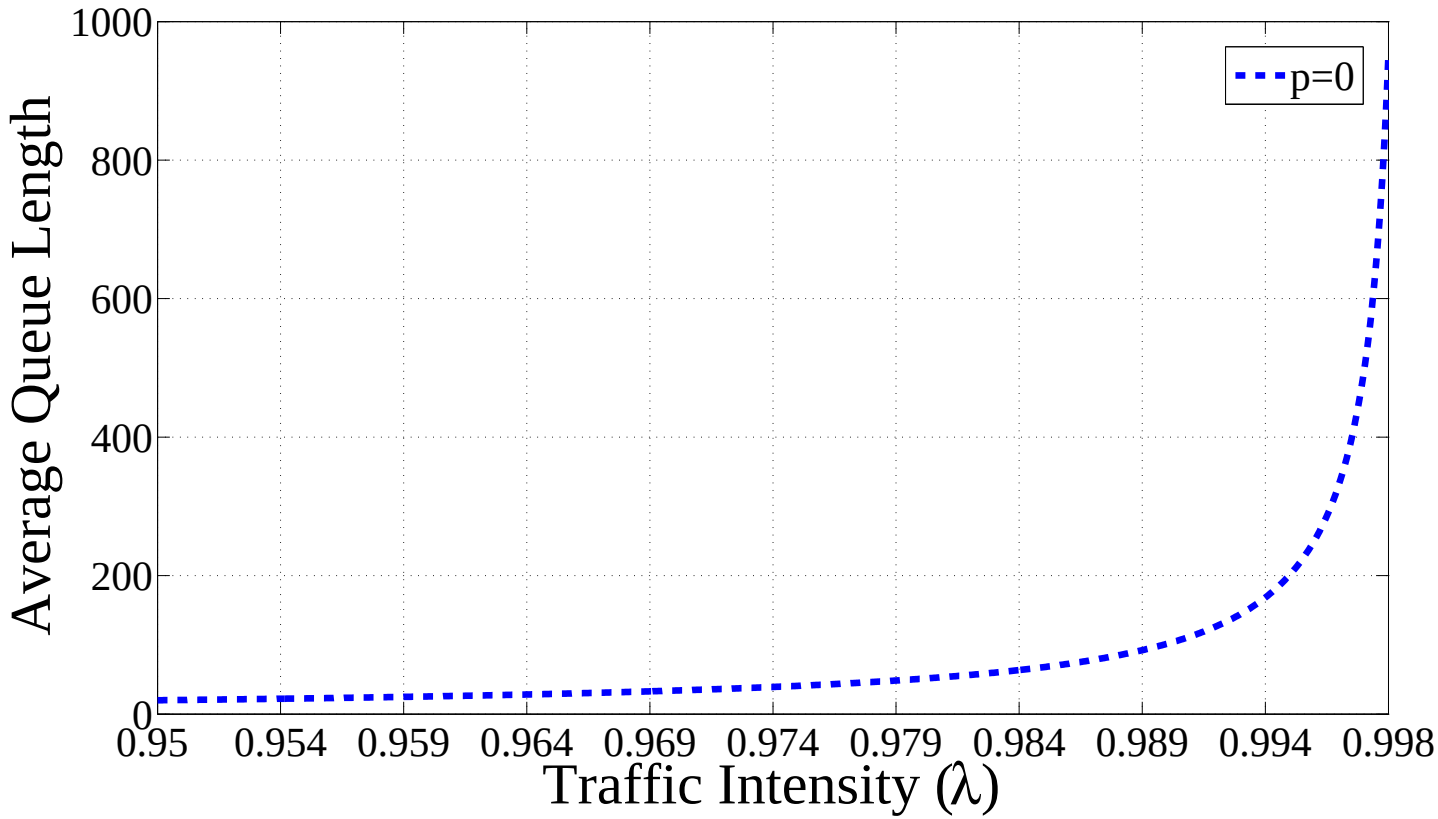
- If $p = 0$, system degenerates into N separate $M/M/1$ queues.
- Classical queuing theory: delay scales as $\frac{1}{1-\lambda}$ as $\lambda \to 1$.

Simple Observations

Some simple observations before calculations:

- If $p = 0$, system degenerates into N separate $M/M/1$ queues.
- Classical queuing theory: delay scales as $\frac{1}{1-\lambda}$ as $\lambda \to 1$.

Simple Observations



Simple Observations

For $p > 0$, more observations:

- **Central server** always busy whenever the system is non-empty.
- Some **local servers** may be **idling** while other local servers are **busy**.
- Idling leads to **waste of processing resources** at local servers.
- Reasonable to expect:

$$\text{larger } p \Longrightarrow \text{ less idling resources} \Longrightarrow \text{ better performance}$$

Simple Observations

For $p > 0$, more observations:

- **Central server** always busy whenever the system is non-empty.
- Some **local servers** may be **idling** while other local servers are **busy**.
- Idling leads to **waste of processing resources** at local servers.
- Reasonable to expect:

$$\text{larger } p \Longrightarrow \text{ less idling resources} \Longrightarrow \text{ better performance}$$

Simple Observations

For $p > 0$, more observations:

- **Central server** always busy whenever the system is non-empty.
- Some **local servers** may be **idling** while other local servers are **busy**.
- Idling leads to **waste of processing resources** at local servers.
- Reasonable to expect:

$$\text{larger } p \Longrightarrow \text{ less idling resources} \Longrightarrow \text{ better performance}$$

Simple Observations

For $p > 0$, more observations:

- **Central server** always busy whenever the system is non-empty.
- Some **local servers** may be **idling** while other local servers are **busy**.
- Idling leads to **waste of processing resources** at local servers.
- Reasonable to expect:

$$\text{larger } p \Longrightarrow \text{ less idling resources} \Longrightarrow \text{ better performance}$$

Simple Observations

For $p > 0$, more observations:

- **Central server** always busy whenever the system is non-empty.
- Some **local servers** may be **idling** while other local servers are **busy**.
- Idling leads to **waste of processing resources** at local servers.
- Reasonable to expect:

$$\text{larger } p \implies \text{ less idling resources} \implies \text{ better performance}$$

Simple Observations

To analyze the system when $p > 0$:

- The central server induces correlations among queues, making exact analysis very difficult for finite N.
- We consider the regime as $N \to \infty$.
- The rest of the talk:
  - Exact performance characterization for any p and λ using a **fluid approximation**, in the limit of $N \to \infty$.
  - How to **justify the approximation**.

Simple Observations

To analyze the system when $p > 0$:

- The central server induces correlations among queues, making exact analysis very difficult for finite N.
- We consider the regime as $N \to \infty$.
- The rest of the talk:
  - Exact performance characterization for any p and λ using a **fluid approximation**, in the limit of $N \to \infty$.
  - How to **justify the approximation**.

Simple Observations

To analyze the system when $p > 0$:

- The central server induces correlations among queues, making exact analysis very difficult for finite N.
- We consider the regime as $N \to \infty$.
- The rest of the talk:
  - Exact performance characterization for any p and λ using a **fluid approximation**, in the limit of $N \to \infty$.
  - How to **justify the approximation**.

System State

To describe the system state:

- Let $\mathbf{S}_i^N(t)$ be the **fraction of queues** with at least i jobs:

$$\mathbf{S}_i^N(t) \stackrel{\triangle}{=} \frac{1}{N} \sum_{k=1}^{N} \mathbb{I}_{[i,\infty)} \left(Q_k(t) \right), \quad i \geq 0.$$

- System is Markov w.r.t $\left\{ \mathbf{S}_i^N(t) \right\}_{i=1}^{\infty}$.

System State

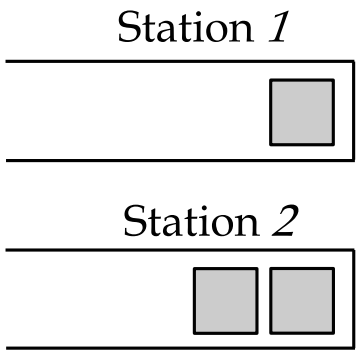
To describe the system state:

- Let $\mathbf{S}_i^N(t)$ be the **fraction of queues** with at least i jobs:

$$\mathbf{S}_i^N(t) \stackrel{\triangle}{=} \frac{1}{N} \sum_{k=1}^{N} \mathbb{I}_{[i,\infty)} \left(Q_k(t) \right), \quad i \geq 0.$$

- System is Markov w.r.t $\left\{ \mathbf{S}_i^N(t) \right\}_{i=1}^{\infty}$.

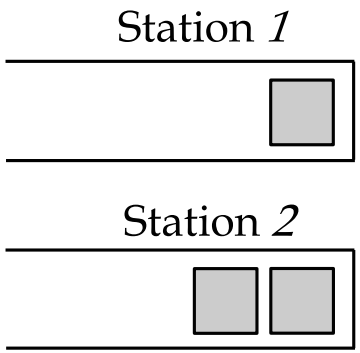
System State: Example



A quick warm-up...

- $N = 2$
- $Q_1(t) = 1$, $Q_2(t) = 2$
- $\mathbf{S}_0^N(t) = \frac{2}{2} = 1$ (always 1)
- $\mathbf{S}_1^N(t) = \frac{1+1}{2} = 1$
- $\mathbf{S}_2^N(t) = \frac{1}{2}$
- $\mathbf{S}_i^N(t) = 0$ for all $i \geq 3$

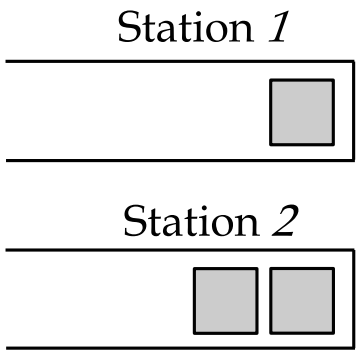
System State: Example



A quick warm-up...

- $N = 2$
- $Q_1(t) = 1$, $Q_2(t) = 2$
- $\mathbf{S}_0^N(t) = \frac{2}{2} = 1$ (always 1)
- $\mathbf{S}_1^N(t) = \frac{1+1}{2} = 1$
- $\mathbf{S}_2^N(t) = \frac{1}{2}$
- $\mathbf{S}_i^N(t) = 0$ for all $i \geq 3$

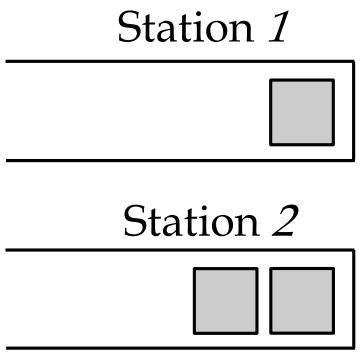
System State: Example



A quick warm-up...

- $N = 2$
- $Q_1(t) = 1$, $Q_2(t) = 2$
- $\mathbf{S}_0^N(t) = \frac{2}{2} = 1$ (always 1)
- $\mathbf{S}_1^N(t) = \frac{1+1}{2} = 1$
- $\mathbf{S}_2^N(t) = \frac{1}{2}$
- $\mathbf{S}_i^N(t) = 0$ for all $i \geq 3$

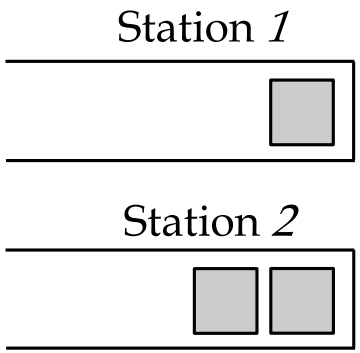
System State: Example



A quick warm-up...

- $N = 2$
- $Q_1(t) = 1$, $Q_2(t) = 2$
- $\mathbf{S}_0^N(t) = \frac{2}{2} = 1$ (always 1)
- $\mathbf{S}_1^N(t) = \frac{1+1}{2} = 1$
- $\mathbf{S}_2^N(t) = \frac{1}{2}$
- $\mathbf{S}_i^N(t) = 0$ for all $i \geq 3$

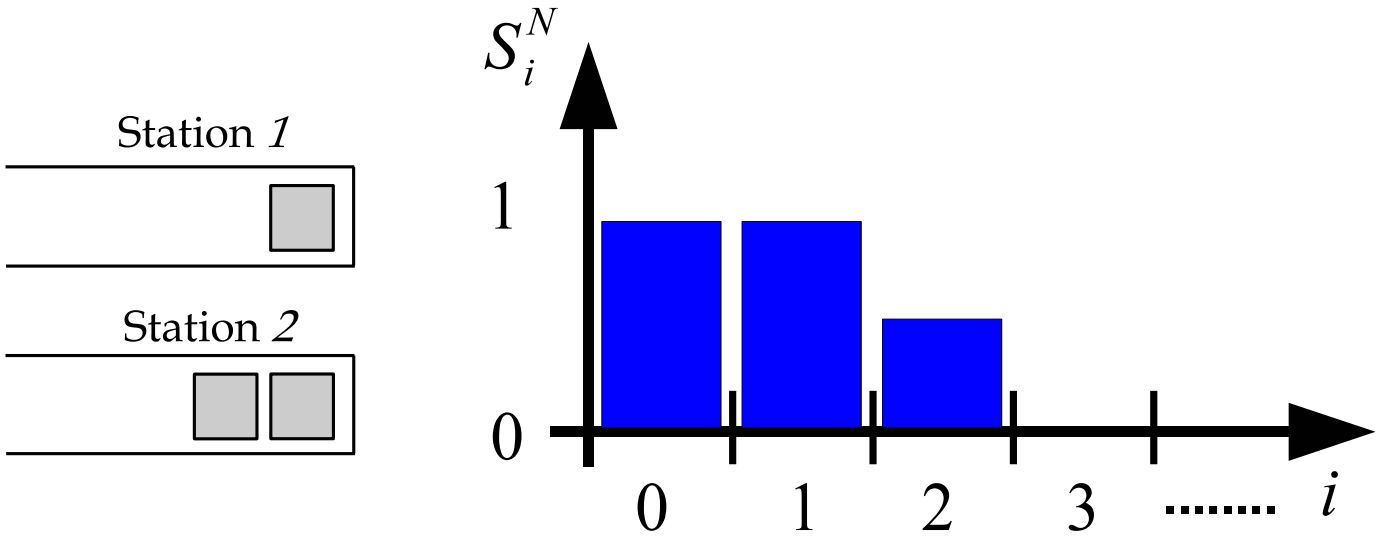
System State: Example



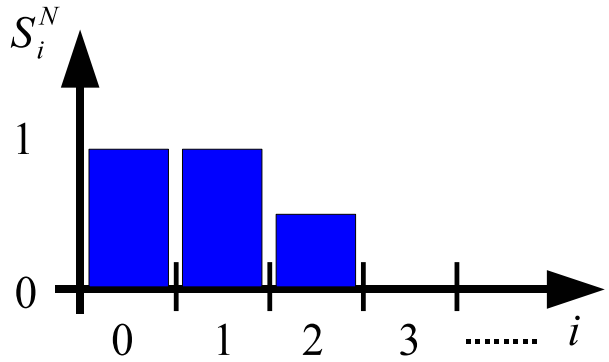
A quick warm-up...

- $N = 2$
- $Q_1(t) = 1$, $Q_2(t) = 2$
- $\mathbf{S}_0^N(t) = \frac{2}{2} = 1$ (always 1)
- $\mathbf{S}_1^N(t) = \frac{1+1}{2} = 1$
- $\mathbf{S}_2^N(t) = \frac{1}{2}$
- $\mathbf{S}_i^N(t) = 0$ for all $i \geq 3$

System State: Example



System State



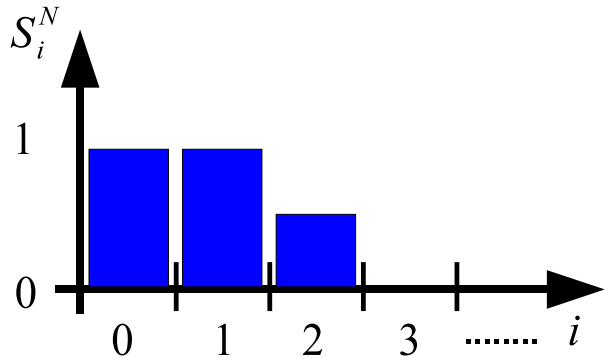
- Interpretation of $\mathbf{S}_i^N(t)$: **tail probabilities**

  $\mathbf{S}_i^N(t) \approx$ prob. of typical arrival sees a queue with $\geq i$ jobs.

- In particular, **average queue length** in the system at time t is

$$Q_{avg}(t) = \frac{\sum_{i=1}^{\infty} \sum_{k=1}^{N} \mathbb{I}_{[i,\infty)}(Q_k(t))}{N} = \sum_{i=1}^{\infty} S_i^N(t)$$

System State



- Interpretation of $\mathbf{S}_i^N(t)$: **tail probabilities**

  $\mathbf{S}_i^N(t) \approx$ prob. of typical arrival sees a queue with $\geq i$ jobs.

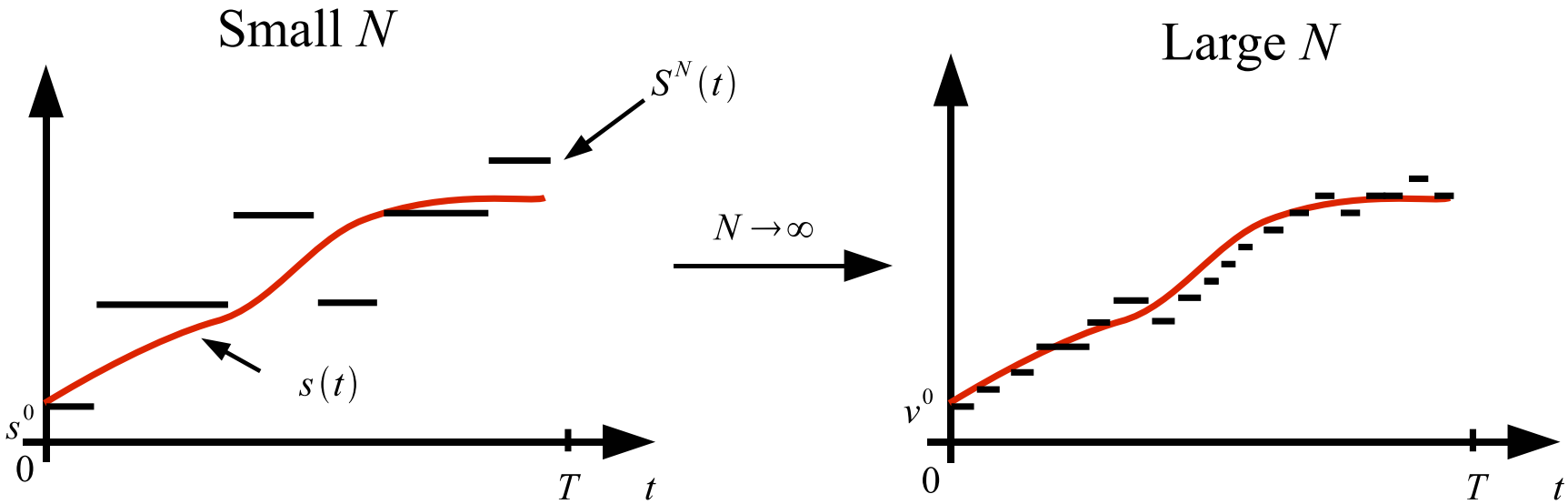
- In particular, **average queue length** in the system at time t is

$$Q_{avg}(t) = \frac{\sum_{i=1}^{\infty} \sum_{k=1}^{N} \mathbb{I}_{[i,\infty)}\left(Q_k(t)\right)}{N} = \sum_{i=1}^{\infty} S_i^N(t)$$

Outline

Fluid Approximation

Sample paths of $\mathbf{S}^N(t)$ "converge" to some smooth trajectory $\mathbf{s}(t)$, as $N \to \infty$



Fluid Approximation

Why use fluid approximations?

- $\mathbf{s}(t)$ is solution to a system of ordinary differential equations

$$\dot{\mathbf{s}}(t) = \mathbf{F}(\mathbf{s}(t)).$$

- Much easier to analyze than the original finite stochastic system.

Fluid Approximation

General steps when applying fluid approximation

- Step 1: Write down fluid model ($\dot{\mathbf{s}}(t) = \mathbf{F}(\mathbf{s}(t))$).
- Step 2: Solve for **invariant state(s)** of fluid model, $\mathbf{s}^I$, as an approximation of the **steady state of** $\mathbf{S}^N(t)$, i.e.

$$\mathbf{F}(\mathbf{s}^I) = 0.$$

- Step 3: Prove **convergence results** to justify why $\mathbf{s}^I$ is a good approximation as $N \to \infty$.

Fluid Approximation

General steps when applying fluid approximation

- Step 1: Write down fluid model ($\dot{\mathbf{s}}(t) = \mathbf{F}(\mathbf{s}(t))$).
- Step 2: Solve for **invariant state(s)** of fluid model, $\mathbf{s}^I$, as an approximation of the **steady state of** $\mathbf{S}^N(t)$, i.e.

$$\mathbf{F}(\mathbf{s}^I) = 0.$$

- Step 3: Prove **convergence results** to justify why $\mathbf{s}^I$ is a good approximation as $N \to \infty$.

Fluid Approximation

General steps when applying fluid approximation

- Step 1: Write down fluid model ($\dot{\mathbf{s}}(t) = \mathbf{F}(\mathbf{s}(t))$).
- Step 2: Solve for **invariant state(s)** of fluid model, $\mathbf{s}^I$, as an approximation of the **steady state of** $\mathbf{S}^N(t)$, i.e.

$$\mathbf{F}(\mathbf{s}^I) = 0.$$

- Step 3: Prove **convergence results** to justify why $\mathbf{s}^I$ is a good approximation as $N \to \infty$.

Outline

Summary of Results

Qualitative results for system performance:

- Closed-form expressions for the invariant state of fluid model, $\mathbf{s}^I$, for all values of p and λ.
- A phase transition in delay scaling: **exponentially better scaling** for any $p > 0$, compared to $p = 0$.

Summary of Results

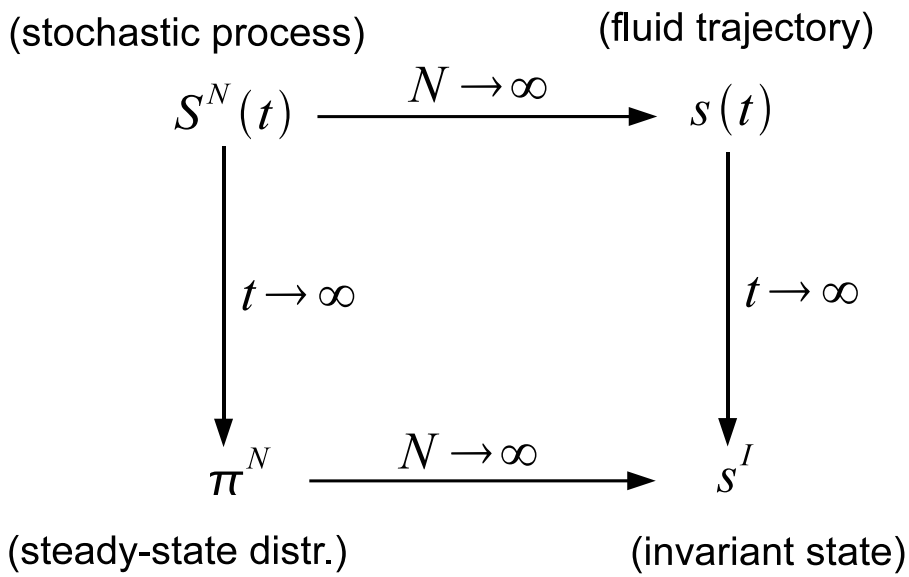
Qualitative results for system performance:

- Closed-form expressions for the invariant state of fluid model, $\mathbf{s}^I$, for all values of p and λ.
- A phase transition in delay scaling: **exponentially better scaling** for any $p > 0$, compared to $p = 0$.

Summary of Results

Convergence results to justify approximation:

- **Transient regime**: $\mathbf{S}^N(t) \to \mathbf{s}(t)$ uniformly over any interval $[0, T]$ w.h.p. as $N \to \infty$.
- **Steady-state regime**: Steady state distributions of $\mathbf{S}^N(t)$ concentrates on the invariant state of fluid model, $\mathbf{s}^I$, as $N \to \infty$.
- Require non-standard work due to **intrinsic discontinuities** in fluid model.



Fluid Model

Definition of Fluid Model

$\mathbf{s}(t)$ is a **solution to fluid model** if

- (**initial condition**) $\mathbf{s}(0)$ is equal to some finite $\mathbf{s}^0$,
- (**boundary condition**) $\mathbf{s}_0(t) = 1$ and $1 \geq \mathbf{s}_i(t) \geq \mathbf{s}_{i+1}(t) \geq 0$ for all $i \geq 0$,
- (**drift**) $\mathbf{s}(t)$ is differentiable a.e. on $[0, \infty)$ and

$$\dot{\mathbf{s}}_i(t) = \lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) - (1-p)(\mathbf{s}_i - \mathbf{s}_{i+1}) - g_i(\mathbf{s}),$$

where

$$g_i(\mathbf{s}) = \begin{cases} 0, & \mathbf{s}_i > 0, \mathbf{s}_{i+1} > 0, \\ p - \min\{\lambda \mathbf{s}_i, p\}, & \mathbf{s}_i > 0, \mathbf{s}_{i+1} = 0, \\ \min\{\lambda \mathbf{s}_{i-1}, p\}, & \mathbf{s}_i = 0, \mathbf{s}_{i-1} > 0, \\ 0, & \mathbf{s}_i = 0, \mathbf{s}_{i-1} = 0. \end{cases}$$

Interpretation of Drift

Drift of Fluid Model

$$\dot{\mathbf{s}}_i(t) = \lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) - (1-p)(\mathbf{s}_i - \mathbf{s}_{i+1}) - g_i(\mathbf{s})$$

- $\lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) \Longrightarrow$ **job arrivals.**

Interpretation of Drift

Drift of Fluid Model

$$\dot{\mathbf{s}}_i(t) = \lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) - (1-p)(\mathbf{s}_i - \mathbf{s}_{i+1}) - g_i(\mathbf{s})$$

- $\lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) \Longrightarrow$ **job arrivals**.
- $(1-p)(\mathbf{s}_i - \mathbf{s}_{i+1}) \Longrightarrow$ **job departures** due to **local servers**.

Interpretation of Drift

Drift of Fluid Model

$$\dot{\mathbf{s}}_i(t) = \lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) - (1-p)(\mathbf{s}_i - \mathbf{s}_{i+1}) - g_i(\mathbf{s})$$

- $\lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) \Longrightarrow$ **job arrivals**.
- $(1-p)(\mathbf{s}_i - \mathbf{s}_{i+1}) \Longrightarrow$ **job departures** due to **local servers**.
- $g_i(\mathbf{s}) \Longrightarrow$ **job departures** due to the **central server**.

Interpretation of Drift

Drift of Fluid Model

$$\dot{\mathbf{s}}_i(t) = \lambda(\mathbf{s}_{i-1} - \mathbf{s}_i) - (1-p)(\mathbf{s}_i - \mathbf{s}_{i+1}) - g_i(\mathbf{s})$$

Technical challenge: $g(\mathbf{s})$ admits many **discontinuities**:

$$g_i(\mathbf{s}) = \begin{cases} 0, & \mathbf{s}_i > 0, \mathbf{s}_{i+1} > 0, \\ p - \min\{\lambda \mathbf{s}_i, p\}, & \mathbf{s}_i > 0, \mathbf{s}_{i+1} = 0, \\ \min\{\lambda \mathbf{s}_{i-1}, p\}, & \mathbf{s}_i = 0, \mathbf{s}_{i-1} > 0, \\ 0, & \mathbf{s}_i = 0, \mathbf{s}_{i-1} = 0. \end{cases}$$

Solution: use a different state representation to establish uniqueness and convergence results.

Details are given in the paper.

Expressions for Limiting Tail Probabilities

Theorem: Limiting Tail Probabilities

For every fixed p and λ, the fluid model admits a **unique invariant state**, $\mathbf{s}^I$, where

- If $p = 0$, then $\mathbf{s}_i^I = \lambda^i$, $\forall i \geq 0$.
- If $p \geq \lambda$, then $\mathbf{s}_i^I = 0$, $\forall i \geq 0$.
- If $0 < p < \lambda$, and $\lambda = 1 - p$, then

$$\mathbf{s}_i^I = \begin{cases} 1 - \left(\frac{p}{1-p}\right) i, & 1 \leq i \leq \tilde{i}^*(p,\lambda), \\ 0, & i > \tilde{i}^*(p,\lambda), \end{cases}$$

where $\tilde{i}^*(p,\lambda) \stackrel{\triangle}{=} \left\lfloor \frac{1-p}{p} \right\rfloor$.

Expressions for Limiting Tail Probabilities

Theorem: Limiting Tail Probabilities (cont.)

- If $0 < p < \lambda$, and $\lambda \neq 1 - p$, then

$$\mathbf{s}_i^I = \begin{cases} \frac{1-\lambda}{1-(p+\lambda)} \left(\frac{\lambda}{1-p}\right)^i - \frac{p}{1-(p+\lambda)}, & 1 \leq i \leq i^*(p,\lambda), \\ 0, & i > i^*(p,\lambda), \end{cases}$$

where $i^*(p,\lambda) \stackrel{\triangle}{=} \left\lfloor \log_{\frac{\lambda}{1-p}} \frac{p}{1-\lambda} \right\rfloor$

- $0 < p < \lambda$: A most important case, because it includes the **heavy-traffic regime** ($\lambda \to 1$).
- $\mathbf{s}^I$ always have **finite support**.

Expressions for Limiting Tail Probabilities

Theorem: Limiting Tail Probabilities (cont.)

- If $0 < p < \lambda$, and $\lambda \neq 1 - p$, then

$$\mathbf{s}_i^I = \begin{cases} \frac{1-\lambda}{1-(p+\lambda)} \left(\frac{\lambda}{1-p}\right)^i - \frac{p}{1-(p+\lambda)}, & 1 \leq i \leq i^*(p,\lambda), \\ 0, & i > i^*(p,\lambda), \end{cases}$$

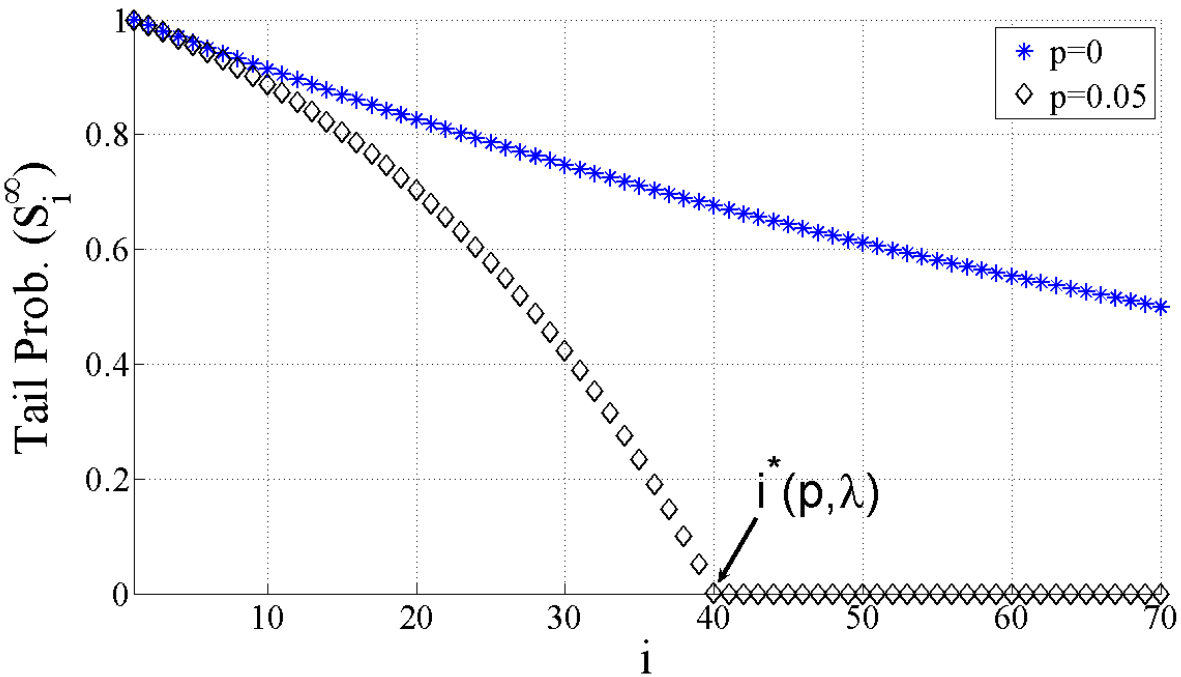
where $i^*(p,\lambda) \stackrel{\triangle}{=} \left\lfloor \log_{\frac{\lambda}{1-p}} \frac{p}{1-\lambda} \right\rfloor$

- $0 < p < \lambda$: A most important case, because it includes the **heavy-traffic regime** ($\lambda \to 1$).
- $\mathbf{s}^I$ always have **finite support**.

Interpretation: Finite Support of $\{\mathbf{s}_i^I\}_{i=1}^{\infty}$

Intuition behind finite-support property:

- **Longest queues** receive **all of the attention** from the central server.
- "Very long" queues **almost never emerge** when $p > 0$!
- "Very long" $\approx i^*(p, \lambda)$.



Phase Transition in Expected Queue Length Scaling

Corollary: Phase Transition in Expected Queue Length Scaling

For any fixed $p > 0$,

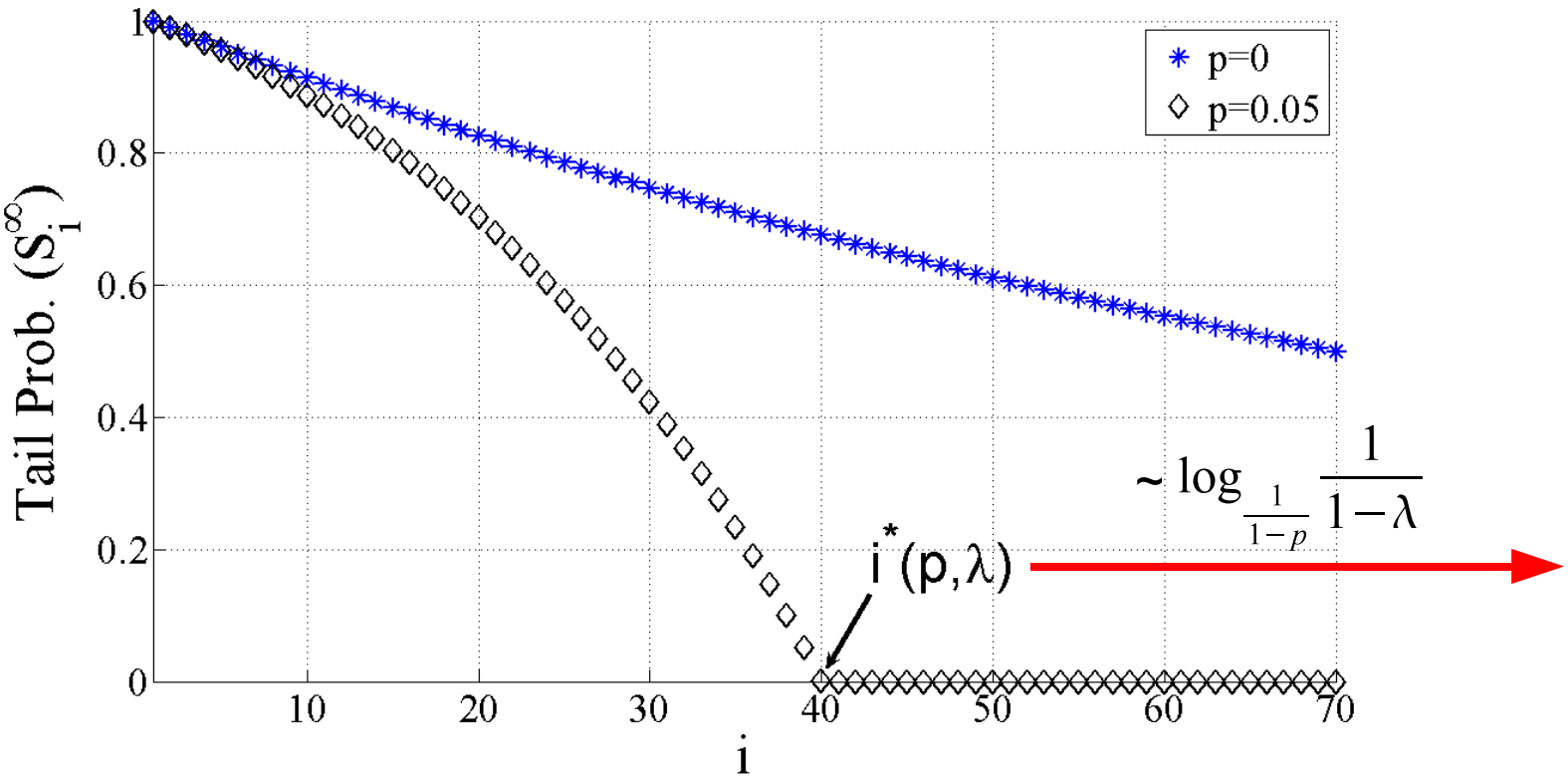
$$\mathbb{E}[Q_{avg}] \sim \log_{\frac{1}{1-p}} \frac{1}{1-\lambda}, \qquad \text{as } \lambda \to 1.$$

- **Exponentially better** delay scaling from $p = 0$, where

$$\mathbb{E}[Q_{avg}] \sim \frac{1}{1-\lambda} \qquad \text{as } \lambda \to 1.$$

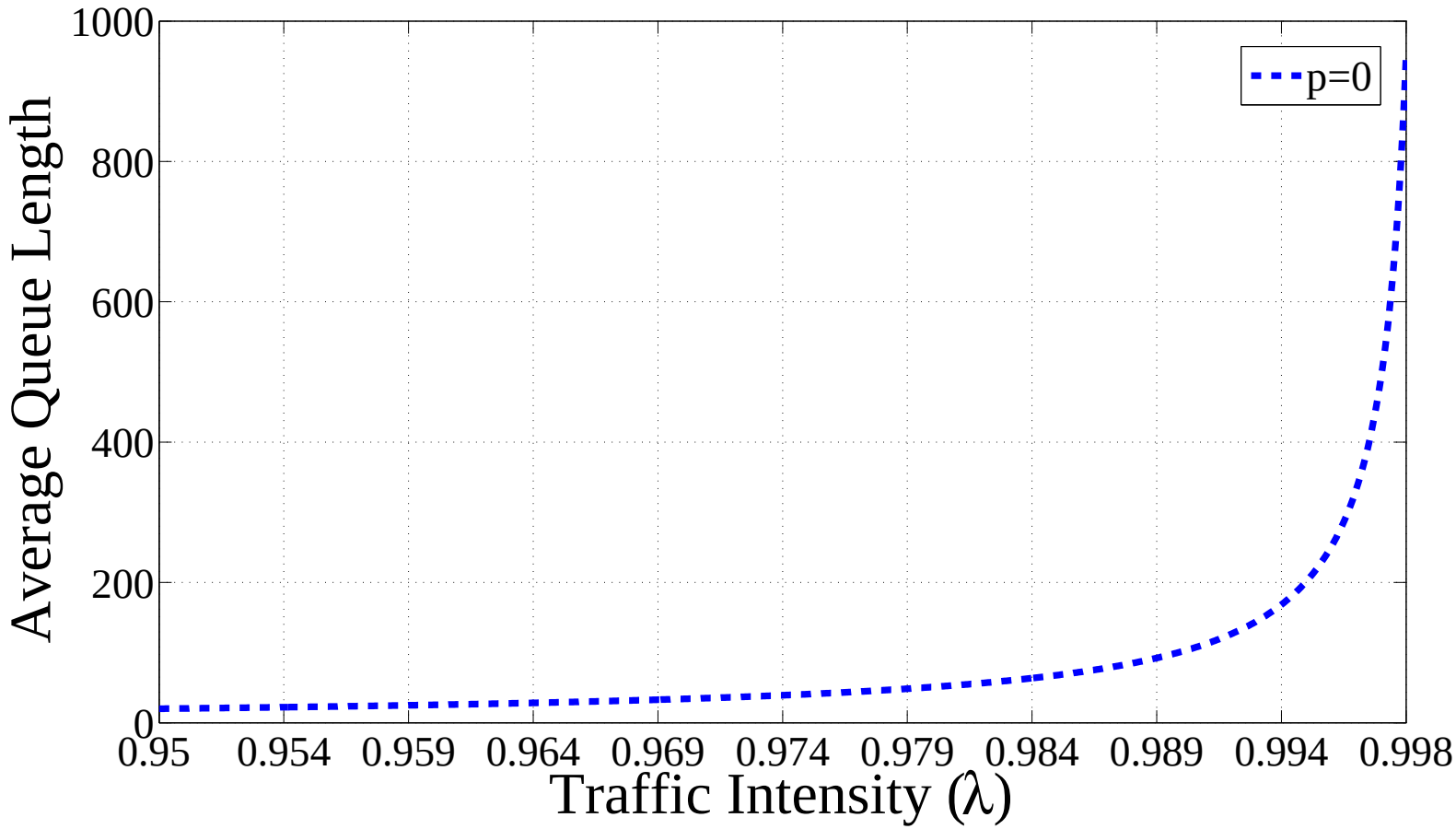
Interpretation: Phase Transition

- **Finite-support property** of $\left\{\mathbf{s}_i^I\right\}_{i=1}^{\infty}$ is crucial.
- Recall $i^*(p,\lambda) \stackrel{\triangle}{=} \left\lfloor \log_{\frac{\lambda}{1-p}} \frac{p}{1-\lambda} \right\rfloor$.
- Scaling of $\mathbb{E}[Q_{avg}] \approx$ scaling of $i^*(p,\lambda)$.



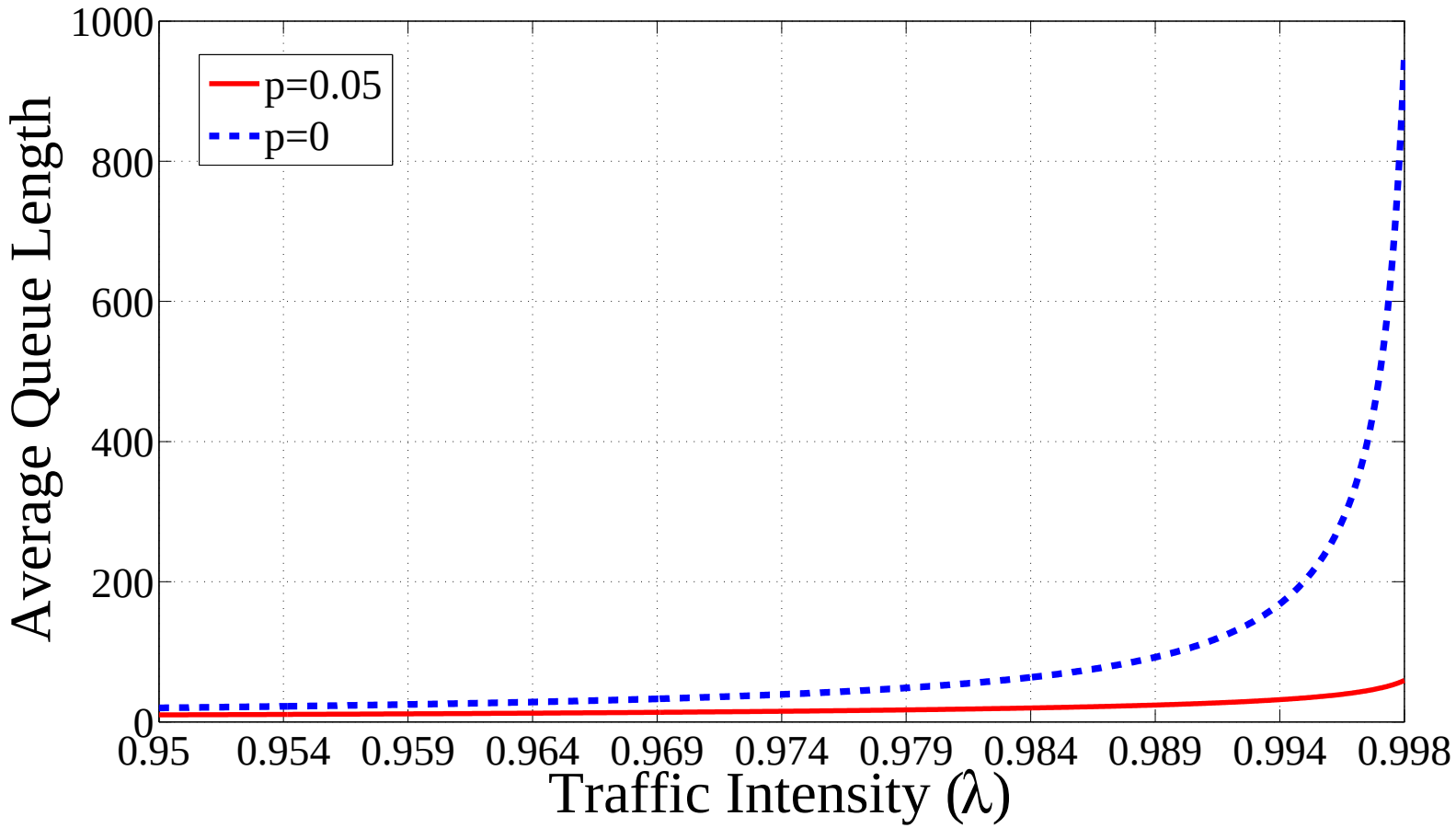
Phase Transition in Expected Queue Length Scaling

Scaling of $\mathbb{E}[Q_{avg}]$: $p > 0$ versus $p = 0$.



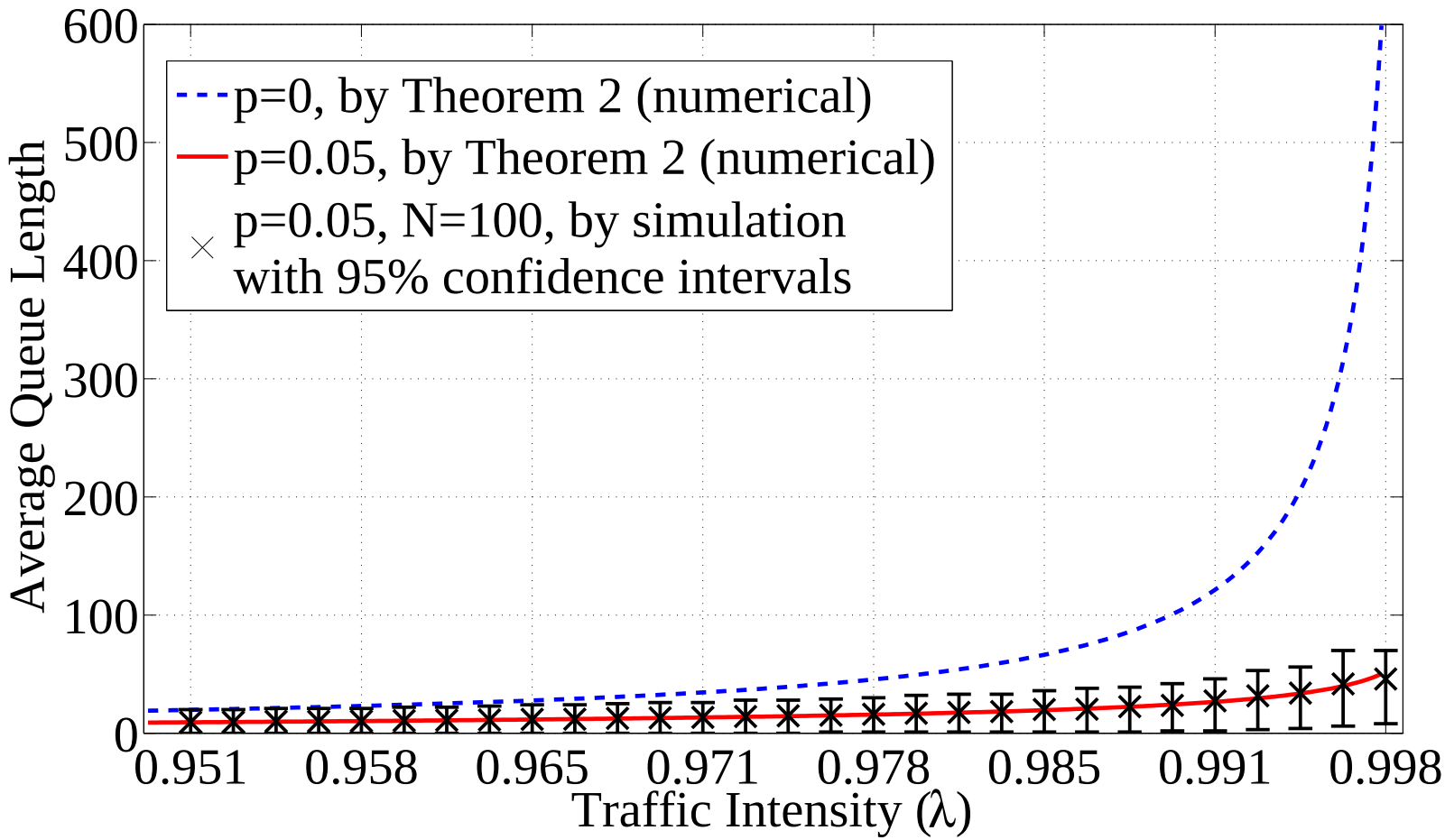
Phase Transition in Expected Queue Length Scaling

Scaling of $\mathbb{E}[Q_{avg}]$: $p > 0$ versus $p = 0$.



Simulations

Good approximation even for $N = 100$.



Outline

Summary

- **Small degree of centralization** or resource pooling has significant benefits.
- Model more realistic constraints?
  - **General** arrival and processing time distributions.
  - **Transmission delays** to central server.
- **Partial centralization** in other distributed systems?

Summary

- **Small degree of centralization** or resource pooling has significant benefits.
- Model more realistic constraints?
  - **General** arrival and processing time distributions.
  - **Transmission delays** to central server.
- **Partial centralization** in other distributed systems?

Summary

- **Small degree of centralization** or resource pooling has significant benefits.
- Model more realistic constraints?
  - **General** arrival and processing time distributions.
  - **Transmission delays** to central server.
- **Partial centralization** in other distributed systems?

Questions?

Thank you!

technical report available at:
`http://www.mit.edu/~kuangxu/papers/TsiXu11SIG_Ext.pdf`.