Throughput and Heavy-traffic Optimality of General Load Balancing Algorithm in Cloud Networks

Xingyu Zhou

The Ohio State University

zhou.2055@osu.edu

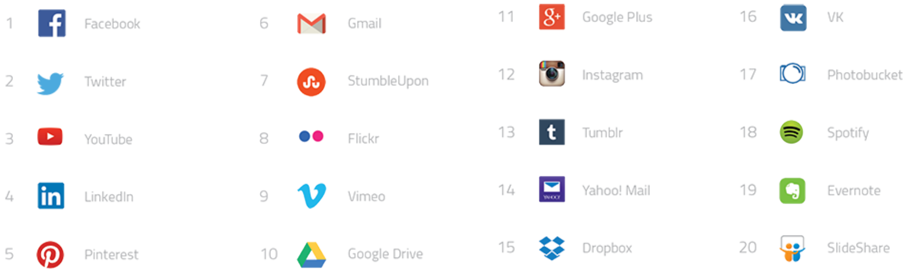
October 24, 2016

Outline

Outline

1. Introduction and Motivation
   - Why do we need an effective and fast cloud?
   - Load Balancing and Previous Works
2. Throughput and Heavy-traffic optimality of General Load Balancing
   - Model, Challenges, Contributions and Key insights
   - Methodology and Sufficient Conditions
   - Homogeneous Servers
   - Heterogeneous Servers
   - Conclusions

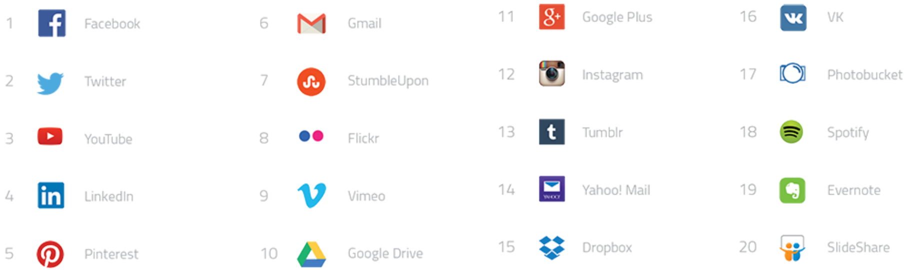
Cloud is everywhere in life



Source: *https://www.skyhighnetworks.com/*

Figure: Top 20 consumer cloud services in 2016

Cloud is everywhere in life



Source: *https://www.skyhighnetworks.com/*

Figure: Top 20 consumer cloud services in 2016

Question: What makes a good cloud service?

An Effective and Fast Cloud

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.
  - It determines how many units of information an application can process in a period of time.

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.
  - It determines how many units of information an application can process in a period of time.
  - High throughput means the application can handle more number of concurrent users.

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.
  - It determines how many units of information an application can process in a period of time.
  - High throughput means the application can handle more number of concurrent users.
- **Fast:** Low delay and response time

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.
  - It determines how many units of information an application can process in a period of time.
  - High throughput means the application can handle more number of concurrent users.
- **Fast:** Low delay and response time
  - It determines how much time an application takes to return a request to users.

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.
  - It determines how many units of information an application can process in a period of time.
  - High throughput means the application can handle more number of concurrent users.
- **Fast:** Low delay and response time
  - It determines how much time an application takes to return a request to users.
  - Low response time means a good user experience and also great profit for companies.

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.
  - It determines how many units of information an application can process in a period of time.
  - High throughput means the application can handle more number of concurrent users.
- **Fast:** Low delay and response time
  - It determines how much time an application takes to return a request to users.
  - Low response time means a good user experience and also great profit for companies.
  - Examples[1]:
    - ★ Google: an half second increase in loading time drops traffic by 20%
    - ★ Amazon: every 100ms of latency costs them 1% in sales

[1] Credit: Prof. Ness Shroff Mobihoc 2015 Talk

An Effective and Fast Cloud

- **Effective:** High utilization and throughput.
  - It determines how many units of information an application can process in a period of time.
  - High throughput means the application can handle more number of concurrent users.
- **Fast:** Low delay and response time
  - It determines how much time an application takes to return a request to users.
  - Low response time means a good user experience and also great profit for companies.
  - Examples[1]:
    - ★ Google: an half second increase in loading time drops traffic by 20%
    - ★ Amazon: every 100ms of latency costs them 1% in sales

Question: How do we design an effective and fast cloud system?

[1] Credit: Prof. Ness Shroff Mobihoc 2015 Talk

Outline

1. Introduction and Motivation
   - Why do we need an effective and fast cloud?
   - Load Balancing and Previous Works

2. Throughput and Heavy-traffic optimality of General Load Balancing
   - Model, Challenges, Contributions and Key insights
   - Methodology and Sufficient Conditions
   - Homogeneous Servers
   - Heterogeneous Servers
   - Conclusions

Load Balancing in Cloud

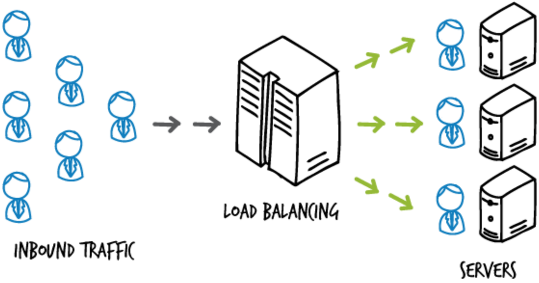


Figure: A typical model in cloud system

- **Load balancing:** Choose the right server(s) when requests coming.

Load Balancing in Cloud

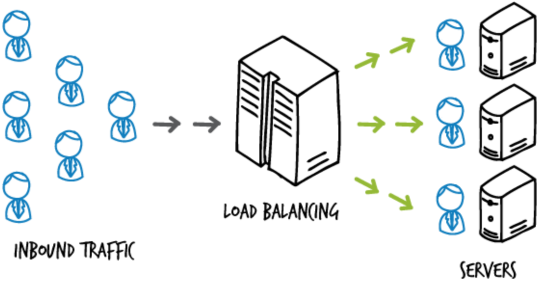


Figure: A typical model in cloud system

- **Load balancing:** Choose the right server(s) when requests coming.
  - It is the key to optimize resource use, maximize throughput, reduce response time in cloud system.

Load Balancing in Cloud

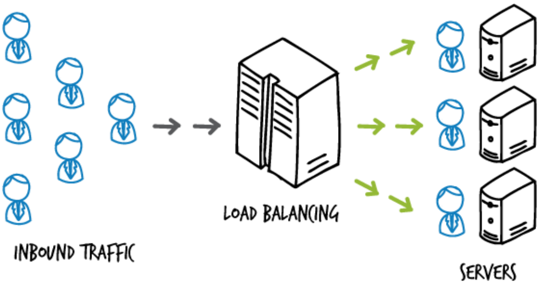


Figure: A typical model in cloud system

- **Load balancing:** Choose the right server(s) when requests coming.
  - It is the key to optimize resource use, maximize throughput, reduce response time in cloud system.
  - It becomes more and more critical due to explosive increase in the number of servers and traffic in cloud system.

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.
  - Task arrival: average-delay of Join-the-shortest-queue (JSQ), Power-of-d and Join-the-idle-queue (JIQ)[Mitzenmacher'96,'16, Lu'11]

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.
  - Task arrival: average-delay of Join-the-shortest-queue (JSQ), Power-of-d and Join-the-idle-queue (JIQ)[Mitzenmacher'96,'16, Lu'11]
  - Job arrival: Batch sampling is introduced to reduce sampling overhead. Srikant'15, Ousterhout'13]

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.
  - Task arrival: average-delay of Join-the-shortest-queue (JSQ), Power-of-d and Join-the-idle-queue (JIQ)[Mitzenmacher'96,'16, Lu'11]
  - Job arrival: Batch sampling is introduced to reduce sampling overhead. Srikant'15, Ousterhout'13]
- **Heavy traffic limit:** the utilization of server ρ goes to 1.

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.
  - Task arrival: average-delay of Join-the-shortest-queue (JSQ), Power-of-d and Join-the-idle-queue (JIQ)[Mitzenmacher'96,'16, Lu'11]
  - Job arrival: Batch sampling is introduced to reduce sampling overhead. Srikant'15, Ousterhout'13]
- **Heavy traffic limit:** the utilization of server ρ goes to 1.
  - Brownian motion condition: Sample-path heavy-traffic optimality in scaled time over a finite time interval of JSQ, power-of-d[Foschini et al,'78, Whiting'16]

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.
  - Task arrival: average-delay of Join-the-shortest-queue (JSQ), Power-of-d and Join-the-idle-queue (JIQ)[Mitzenmacher'96,'16, Lu'11]
  - Job arrival: Batch sampling is introduced to reduce sampling overhead. Srikant'15, Ousterhout'13]
- **Heavy traffic limit:** the utilization of server ρ goes to 1.
  - Brownian motion condition: Sample-path heavy-traffic optimality in scaled time over a finite time interval of JSQ, power-of-d[Foschini et al,'78, Whiting'16]
  - Lyapunov Drift condition: First moment heavy-traffic optimality of JSQ, power-of-d. [Erylimaz'13, Maguluri'12]

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.
  - Task arrival: average-delay of Join-the-shortest-queue (JSQ), Power-of-d and Join-the-idle-queue (JIQ)[Mitzenmacher'96,'16, Lu'11]
  - Job arrival: Batch sampling is introduced to reduce sampling overhead. Srikant'15, Ousterhout'13]
- **Heavy traffic limit:** the utilization of server ρ goes to 1.
  - Brownian motion condition: Sample-path heavy-traffic optimality in scaled time over a finite time interval of JSQ, power-of-d[Foschini et al,'78, Whiting'16]
  - Lyapunov Drift condition: First moment heavy-traffic optimality of JSQ, power-of-d. [Erylimaz'13, Maguluri'12]

Questions: Can we generalize existing load balancing algorithm?

Previous Works

Throughput analysis is relatively easy, however, delay is really difficult for exact analysis. Two alternatives are asymptotic analysis:

- **Large system limit:** the number of servers N goes to infinity.
  - Task arrival: average-delay of Join-the-shortest-queue (JSQ), Power-of-d and Join-the-idle-queue (JIQ)[Mitzenmacher'96,'16, Lu'11]
  - Job arrival: Batch sampling is introduced to reduce sampling overhead. Srikant'15, Ousterhout'13]
- **Heavy traffic limit:** the utilization of server ρ goes to 1.
  - Brownian motion condition: Sample-path heavy-traffic optimality in scaled time over a finite time interval of JSQ, power-of-d[Foschini et al,'78, Whiting'16]
  - Lyapunov Drift condition: First moment heavy-traffic optimality of JSQ, power-of-d. [Erylimaz'13, Maguluri'12]

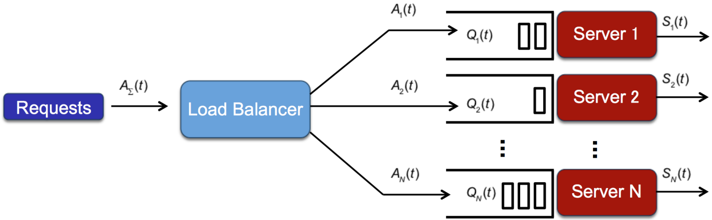
Questions: Can we generalize existing load balancing algorithm?

Question: Do we really need sampling for each arrival?

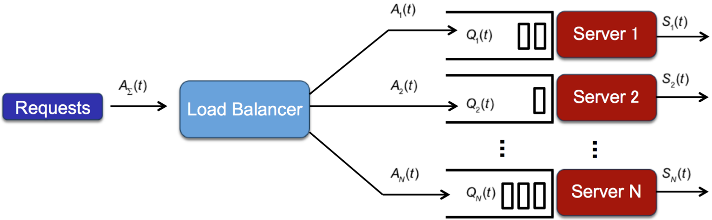
Outline

Outline

System Model and Assumptions

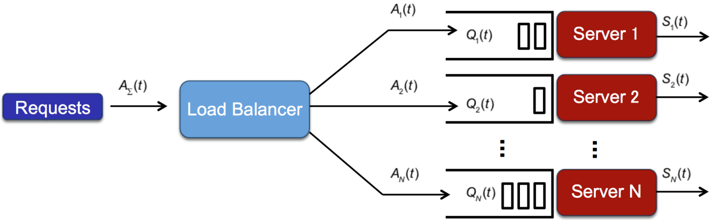


System Model and Assumptions



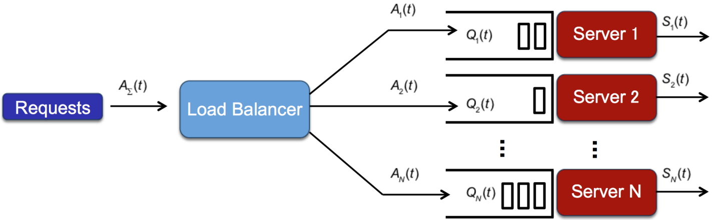
- The exogenous arrival process $A_\Sigma(t)$ with mean rate λ_Σ is i.i.d and also independent of service, queue length, and routing decision. $A_\Sigma(t) \leq A_{\max} < \infty$ for all $t \geq 0$

System Model and Assumptions



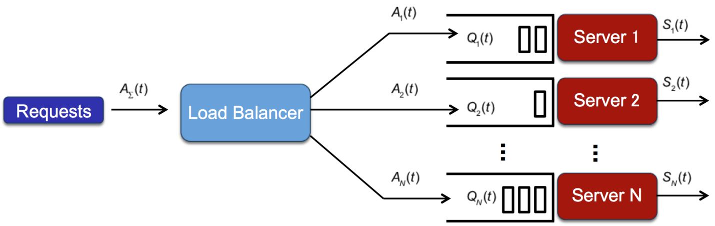
- The exogenous arrival process $A_\Sigma(t)$ with mean rate λ_Σ is i.i.d and also independent of service, queue length, and routing decision. $A_\Sigma(t) \leq A_{\max} < \infty$ for all $t \geq 0$
- Each service process $S_n(t)$ with mean rate μ_n is i.i.d and also independent of other service, arrival, queue length, and routing decision. $S_n(t) \leq S_{\max} < \infty$ for all $t \geq 0$ and all $n \in \{1, 2, \ldots, N\}$, and $S_\Sigma(t) = \sum S_n(t)$ is the hypothetical total service with mean $\mu_\Sigma = \sum \mu_n$. $\epsilon = \mu_\Sigma - \lambda_\Sigma$ indicates how close the arrival rate to the boundary.

System Model and Assumptions



- The exogenous arrival process $A_\Sigma(t)$ with mean rate λ_Σ is i.i.d and also independent of service, queue length, and routing decision. $A_\Sigma(t) \leq A_{\max} < \infty$ for all $t \geq 0$
- Each service process $S_n(t)$ with mean rate μ_n is i.i.d and also independent of other service, arrival, queue length, and routing decision. $S_n(t) \leq S_{\max} < \infty$ for all $t \geq 0$ and all $n \in \{1, 2, \ldots, N\}$, and $S_\Sigma(t) = \sum S_n(t)$ is the hypothetical total service with mean $\mu_\Sigma = \sum \mu_n$. $\epsilon = \mu_\Sigma - \lambda_\Sigma$ indicates how close the arrival rate to the boundary.
- Requests cannot be removed from server to server.

System Model and Assumptions



- The exogenous arrival process $A_\Sigma(t)$ with mean rate λ_Σ is i.i.d and also independent of service, queue length, and routing decision. $A_\Sigma(t) \leq A_{\max} < \infty$ for all $t \geq 0$
- Each service process $S_n(t)$ with mean rate μ_n is i.i.d and also independent of other service, arrival, queue length, and routing decision. $S_n(t) \leq S_{\max} < \infty$ for all $t \geq 0$ and all $n \in \{1, 2, \ldots, N\}$, and $S_\Sigma(t) = \sum S_n(t)$ is the hypothetical total service with mean $\mu_\Sigma = \sum \mu_n$. $\epsilon = \mu_\Sigma - \lambda_\Sigma$ indicates how close the arrival rate to the boundary.
- Requests cannot be removed from server to server.
- Queue length dynamic is

$$
\begin{aligned}
Q_n(t+1) &= [Q_n(t) + A_n(t) - S_n(t)]^+ \\
&= Q_n(t) + A_n(t) - S_n(t) + U_n(t).
\end{aligned}
\tag{1}
$$

Definitions of Throughput and Heavy-traffic Optimal

Definitions of Throughput and Heavy-traffic Optimal

Definition (Throughput Optimality)

In general, a load balancing policy is said to be throughput optimal if it can stabilize any set of arrival rates which can be stabilized by another policy. In our model, a load balancing policy is throughput optimal if it can stabilize any exogenous rate $\lambda_\Sigma < \mu_\Sigma$.

Definitions of Throughput and Heavy-traffic Optimal

Definition (Throughput Optimality)

In general, a load balancing policy is said to be throughput optimal if it can stabilize any set of arrival rates which can be stabilized by another policy. In our model, a load balancing policy is throughput optimal if it can stabilize any exogenous rate $\lambda_\Sigma < \mu_\Sigma$.

Definition (Equivalent Single Queue)

A single queue process $q(t)$ is said to be the equivalent single queue of the load balancing system if $a(t) = A_\Sigma(t)$ and $s(t) = S_\Sigma(t)$. Clearly, $q(t) \leq_{st} \sum Q_n(t)$

Definitions of Throughput and Heavy-traffic Optimal

Definition (Throughput Optimality)

In general, a load balancing policy is said to be throughput optimal if it can stabilize any set of arrival rates which can be stabilized by another policy. In our model, a load balancing policy is throughput optimal if it can stabilize any exogenous rate $\lambda_\Sigma < \mu_\Sigma$.

Definition (Equivalent Single Queue)

A single queue process $q(t)$ is said to be the equivalent single queue of the load balancing system if $a(t) = A_\Sigma(t)$ and $s(t) = S_\Sigma(t)$. Clearly, $q(t) \leq_{st} \sum Q_n(t)$

Definition (Heavy-traffic Optimality)

A load balancing policy is said to be heavy-traffic optimal if the expected steady-state sum queue length is asymptoticly the same as the equivalent single queue when the arrival rate approaches to the capacity boundary.

Main Contributions

Main Contributions

- We give sufficient conditions for a policy to be throughput and heavy-traffic optimal under certain assumptions.

Main Contributions

- We give sufficient conditions for a policy to be throughput and heavy-traffic optimal under certain assumptions.
- We introduce a new load balancing policy, called Join-Below-Average (JBA), which is both throughput and heavy-traffic optimal. Instead of joining the shortest queue, it needs only to randomly selects a queue with queue length below average to join.

Main Contributions

- We give sufficient conditions for a policy to be throughput and heavy-traffic optimal under certain assumptions.
- We introduce a new load balancing policy, called Join-Below-Average (JBA), which is both throughput and heavy-traffic optimal. Instead of joining the shortest queue, it needs only to randomly selects a queue with queue length below average to join.
  - it is often easy to obtain the sum queue length of a system, and hence the average queue length.

Main Contributions

- We give sufficient conditions for a policy to be throughput and heavy-traffic optimal under certain assumptions.
- We introduce a new load balancing policy, called Join-Below-Average (JBA), which is both throughput and heavy-traffic optimal. Instead of joining the shortest queue, it needs only to randomly selects a queue with queue length below average to join.
  - it is often easy to obtain the sum queue length of a system, and hence the average queue length.
  - sometimes, it is prohibited for a request to join the shortest queue due to data locality.

Main Contributions

- We give sufficient conditions for a policy to be throughput and heavy-traffic optimal under certain assumptions.
- We introduce a new load balancing policy, called Join-Below-Average (JBA), which is both throughput and heavy-traffic optimal. Instead of joining the shortest queue, it needs only to randomly selects a queue with queue length below average to join.
  - it is often easy to obtain the sum queue length of a system, and hence the average queue length.
  - sometimes, it is prohibited for a request to join the shortest queue due to data locality.
- We show that a class of load balancing policies is throughput and heavy-traffic optimal.

Main Contributions

- We give sufficient conditions for a policy to be throughput and heavy-traffic optimal under certain assumptions.
- We introduce a new load balancing policy, called Join-Below-Average (JBA), which is both throughput and heavy-traffic optimal. Instead of joining the shortest queue, it needs only to randomly selects a queue with queue length below average to join.
  - it is often easy to obtain the sum queue length of a system, and hence the average queue length.
  - sometimes, it is prohibited for a request to join the shortest queue due to data locality.
- We show that a class of load balancing policies is throughput and heavy-traffic optimal.
  - we need only use any 'good' load balancing policy once every T time slots, for any finite T, and random routing in other time slots to achieve throughput and heavy-traffic optimality

Main Contributions

- We give sufficient conditions for a policy to be throughput and heavy-traffic optimal under certain assumptions.
- We introduce a new load balancing policy, called Join-Below-Average (JBA), which is both throughput and heavy-traffic optimal. Instead of joining the shortest queue, it needs only to randomly selects a queue with queue length below average to join.
  - it is often easy to obtain the sum queue length of a system, and hence the average queue length.
  - sometimes, it is prohibited for a request to join the shortest queue due to data locality.
- We show that a class of load balancing policies is throughput and heavy-traffic optimal.
  - we need only use any 'good' load balancing policy once every T time slots, for any finite T, and random routing in other time slots to achieve throughput and heavy-traffic optimality
  - The 'good' policy can be any one time-slot sampling heavy traffic optimal policy, such as JBA, Power-of-d, and JSQ depending on different situations.

Key Insights

Key Insights

- We know that purely random routing is not heavy-traffic optimal, but, it is not so bad as we might think.

Key Insights

- We know that purely random routing is not heavy-traffic optimal, but, it is not so bad as we might think.
  - For homogeneous servers, it actually does no harm (of course no good) to heavy-traffic optimality.

Key Insights

- We know that purely random routing is not heavy-traffic optimal, but, it is not so bad as we might think.
  - For homogeneous servers, it actually does no harm (of course no good) to heavy-traffic optimality.
  - For heterogeneous servers, the harm it does will tend to zero as traffic becomes more heavy.

Key Insights

- We know that purely random routing is not heavy-traffic optimal, but, it is not so bad as we might think.
  - For homogeneous servers, it actually does no harm (of course no good) to heavy-traffic optimality.
  - For heterogeneous servers, the harm it does will tend to zero as traffic becomes more heavy.
- As the traffic gets heavier, it actually becomes more and more easy to achieve heavy-traffic optimality.

Key Insights

- We know that purely random routing is not heavy-traffic optimal, but, it is not so bad as we might think.
  - For homogeneous servers, it actually does no harm (of course no good) to heavy-traffic optimality.
  - For heterogeneous servers, the harm it does will tend to zero as traffic becomes more heavy.
- As the traffic gets heavier, it actually becomes more and more easy to achieve heavy-traffic optimality.
  - The upper bound on the sampling interval T gets larger when the traffic becomes heavier.

Key Insights

- We know that purely random routing is not heavy-traffic optimal, but, it is not so bad as we might think.
  - For homogeneous servers, it actually does no harm (of course no good) to heavy-traffic optimality.
  - For heterogeneous servers, the harm it does will tend to zero as traffic becomes more heavy.
- As the traffic gets heavier, it actually becomes more and more easy to achieve heavy-traffic optimality.
  - The upper bound on the sampling interval T gets larger when the traffic becomes heavier.
  - In some sense, load balancing becomes easier (kind of counterintuitive, but can be explained) when traffic becomes heavier.

Outline

Main Ideas

- Choose the Lyapunov function $Z(\mathbf{Q}) \triangleq \|\mathbf{Q}\|_1^2 = \left(\sum_{n=1}^{N} Q_n\right)^2$, and set the mean drift to zero in steady-state to get bounds.

Main Ideas

- Choose the Lyapunov function $Z(\mathbf{Q}) \triangleq \|\mathbf{Q}\|_1^2 = \left(\sum_{n=1}^{N} Q_n\right)^2$, and set the mean drift to zero in steady-state to get bounds.
  - (R1) It requires the existence of a stationary distribution $\overline{\mathbf{Q}}$.

Main Ideas

- Choose the Lyapunov function $Z(\mathbf{Q}) \triangleq \|\mathbf{Q}\|_1^2 = \left(\sum_{n=1}^{N} Q_n\right)^2$, and set the mean drift to zero in steady-state to get bounds.
  - (R1) It requires the existence of a stationary distribution $\overline{\mathbf{Q}}$.
  - (R2) It requires bounded moments of the stationary distribution to avoid the situation $\infty - \infty$, i.e, $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}\right\|^2\right] \leq M$.

Main Ideas

- Choose the Lyapunov function $Z(\mathbf{Q}) \triangleq \|\mathbf{Q}\|_1^2 = \left(\sum_{n=1}^{N} Q_n\right)^2$, and set the mean drift to zero in steady-state to get bounds.
  - (R1) It requires the existence of a stationary distribution $\overline{\mathbf{Q}}$.
  - (R2) It requires bounded moments of the stationary distribution to avoid the situation $\infty - \infty$, i.e, $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}\right\|^2\right] \leq M$.
- Assume (R1) and (R2) satisfy, we obtain the expected sum queue length in steady-state

$$\epsilon\mathbb{E}\left[\sum_{n=1}^{N} \overline{Q}_n^{(\epsilon)}\right] = \frac{\zeta^{(\epsilon)}}{2} + \mathbb{E}\left[\left\|\overline{\mathbf{Q}}(t+1)\right\|_1 \left\|\overline{\mathbf{U}}(t)\right\|_1\right] - \mathbb{E}\left[\left\|\overline{\mathbf{U}}(t)\right\|_1^2\right] \quad (2)$$

Main Ideas

- Choose the Lyapunov function $Z(\mathbf{Q}) \triangleq \|\mathbf{Q}\|_1^2 = \left(\sum_{n=1}^{N} Q_n\right)^2$, and set the mean drift to zero in steady-state to get bounds.
  - (R1) It requires the existence of a stationary distribution $\overline{\mathbf{Q}}$.
  - (R2) It requires bounded moments of the stationary distribution to avoid the situation $\infty - \infty$, i.e, $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}\right\|^2\right] \leq M$.
- Assume (R1) and (R2) satisfy, we obtain the expected sum queue length in steady-state

$$\epsilon\mathbb{E}\left[\sum_{n=1}^{N} \overline{Q}_n^{(\epsilon)}\right] = \frac{\zeta^{(\epsilon)}}{2} + \mathbb{E}\left[\left\|\overline{\mathbf{Q}}(t+1)\right\|_1 \left\|\overline{\mathbf{U}}(t)\right\|_1\right] - \mathbb{E}\left[\left\|\overline{\mathbf{U}}(t)\right\|_1^2\right] \quad (2)$$

- By letting $N = 1$, expected queue length for the corresponding Equivalent Single Queue is simply

$$\epsilon\mathbb{E}\left[q^{(\epsilon)}\right] = \frac{\zeta^{(\epsilon)}}{2} + \mathbb{E}\left[q(t+1)u(t)\right] - \mathbb{E}\left[u^2\right] \quad (3)$$

Main Ideas (Cont'd)

- Foe the equivalent single queue, by exploiting the most important equation $q(t+1)u(t) = 0$ and the fact $\mathbb{E}\left[u^2\right]$ is $o(\epsilon)$, we have

$$\begin{aligned} \epsilon\mathbb{E}\left[q^{(\epsilon)}\right] &= \frac{\zeta^{(\epsilon)}}{2} + \mathbb{E}\left[q(t+1)u(t)\right] - \mathbb{E}\left[u^2\right] \\ &= \frac{\zeta^{(\epsilon)}}{2} - o(\epsilon) \end{aligned}$$

Main Ideas (Cont'd)

- Foe the equivalent single queue, by exploiting the most important equation $q(t+1)u(t) = 0$ and the fact $\mathbb{E}\left[u^2\right]$ is $o(\epsilon)$, we have

$$\begin{aligned}\epsilon\mathbb{E}\left[q^{(\epsilon)}\right] &= \frac{\zeta^{(\epsilon)}}{2} + \mathbb{E}\left[q(t+1)u(t)\right] - \mathbb{E}\left[u^2\right] \\ &= \frac{\zeta^{(\epsilon)}}{2} - o(\epsilon)\end{aligned}$$

- Foe the N server, by exploiting the equation $Q_n(t+1)U_n(t) = 0$ and the fact $\mathbb{E}\left[\left\|\overline{\mathbf{U}}(t)\right\|_1^2\right]$ is $o(\epsilon)$, we have

$$\begin{aligned}\epsilon\mathbb{E}\left[\sum_{n=1}^{N}\overline{Q}_n^{(\epsilon)}\right] &= \frac{\zeta^{(\epsilon)}}{2} + \mathbb{E}\left[\left\|\overline{\mathbf{Q}}(t+1)\right\|_1\left\|\overline{\mathbf{U}}(t)\right\|_1\right] - \mathbb{E}\left[\left\|\overline{\mathbf{U}}(t)\right\|_1^2\right] \\ &= \frac{\zeta^{(\epsilon)}}{2} + N\mathbb{E}\left[\langle\overline{\mathbf{U}}, -\overline{\mathbf{Q}}_\perp(t+1)\rangle\right] - o(\epsilon) \\ &\leq \frac{\zeta^{(\epsilon)}}{2} + \sqrt{\mathbb{E}\left[\left\|\overline{\mathbf{Q}}_\perp\right\|^2\right]o(\epsilon)} - o(\epsilon)\end{aligned}$$

in which $\mathbf{Q}_\perp$ is the perpendicular component of $\mathbf{Q}$ with respect to $\mathbf{c} = \frac{1}{\sqrt{N}}\mathbf{1}$

Sufficient Conditions for Throughput Heavy-traffic Optimal

- (R1) If $\overline{\mathbf{Q}}$ exists, i.e., the underline Markov chain is positive recurrent.
- (R2) If $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}\right\|^2\right] \le M$, i.e., the second moments is bounded by a constant.
- (R3) If $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}_\perp\right\|^2\right] \le K$, which is independent of ϵ. This is often called steady-state collapse, which indicates that under heavy-traffic, queue-length vector concentrates around the line $\mathbf{c}$.

Sufficient Conditions for Throughput Heavy-traffic Optimal

- (R1) If $\overline{\mathbf{Q}}$ exists, i.e., the underline Markov chain is positive recurrent.
- (R2) If $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}\right\|^2\right] \leq M$, i.e., the second moments is bounded by a constant.
- (R3) If $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}_\perp\right\|^2\right] \leq K$, which is independent of ϵ. This is often called steady-state collapse, which indicates that under heavy-traffic, queue-length vector concentrates around the line $\mathbf{c}$.

How can we bound the moments?

A Very Useful Lemma

Lemma

For an irreducible aperiodic and positive Markov chain $\{X(t), t \geq 0\}$ *over a countable state space* $\mathcal{X}$*, which converges in distribution to* $\overline{X}$*, and suppose* $V : \mathcal{X} \to \mathbb{R}_+$ *is a Lyapunov function. We define the* T *time slot drift of* V *at* X *as*

$$\Delta V(X) := [V(X(t_0 + T)) - V(X(t_0))]\mathcal{I}(X(t_0) = X),$$

where $\mathcal{I}(.)$ *is the indicator function. Suppose for some positive finite integer* T*, the* T *time slot drift of* V *satisfies the following conditions:*

- *(C1) There exists an* $\eta > 0$ *and a* $\gamma < \infty$ *such that for any* $t_0 = 1, 2, \ldots$ *and for all* $X \in \mathcal{X}$ *with* $V(X) \geq \gamma$*,*

$$\mathbb{E}\left[\Delta V(X) | X(t_0) = X\right] \leq -\eta.$$

- *(C2) There exists a constant* $D < \infty$ *such that for all* $X \in \mathcal{X}$*,*

$$\mathbb{P}(|\Delta V(X)| \leq D) = 1.$$

Then, there exist finite constants $\{M_r, r \in \mathbb{N}\}$ *such that for each positive* r*,* $\mathbb{E}\left[V(\overline{X})^r\right] \leq M_r$*, where* M_r *are fully determined by* η*,* γ *and* D*.*

General Sufficient Conditions

Assuming bounded support of arrival and departure process, by exploiting the useful lemma and properties of projection to a convex cone, we are able to give sufficient condition to more general cases

General Sufficient Conditions

Assuming bounded support of arrival and departure process, by exploiting the useful lemma and properties of projection to a convex cone, we are able to give sufficient condition to more general cases

- (S1) If there exists a finite constant T_1 and $K_1 > 0$ and $\delta > 0$ such that for all t_0

$$\mathbb{E}\left[\sum_{t=t_0}^{t_0+T_1-1} \langle \mathbf{Q}(t), \mathbf{A}(t) - \mathbf{S}(t) \rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \leq -\delta \|\mathbf{Q}\| + K_1 \tag{4}$$

  holds, then the system is throughput optimal and has a stationary distribution with all moments bounded.

General Sufficient Conditions

Assuming bounded support of arrival and departure process, by exploiting the useful lemma and properties of projection to a convex cone, we are able to give sufficient condition to more general cases

- (S1) If there exists a finite constant T_1 and $K_1 > 0$ and $\delta > 0$ such that for all t_0

$$\mathbb{E}\left[\sum_{t=t_0}^{t_0+T_1-1} \langle \mathbf{Q}(t), \mathbf{A}(t) - \mathbf{S}(t) \rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \leq -\delta \|\mathbf{Q}\| + K_1 \tag{4}$$

  holds, then the system is throughput optimal and has a stationary distribution with all moments bounded.
- (S2) If there exists a finite constant T_2, and constants $K_2 > 0$ and $\eta > 0$, both independent of ϵ such that for all t_0

$$\mathbb{E}\left[\sum_{t=t_0}^{t_0+T_2-1} \langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t) \rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \leq -\eta \|\mathbf{Q}_\perp\| + K_2 \tag{5}$$

  holds, then the moments of perpendicular component with respect to any convex set $\mathcal{C}$ is bounded. (steady-state collapse to a convex set $\mathcal{C}$)

General Sufficient Conditions

Assuming bounded support of arrival and departure process, by exploiting the useful lemma and properties of projection to a convex cone, we are able to give sufficient condition to more general cases

- (S1) If there exists a finite constant T_1 and $K_1 > 0$ and $\delta > 0$ such that for all t_0

$$\mathbb{E}\left[\sum_{t=t_0}^{t_0+T_1-1} \langle \mathbf{Q}(t), \mathbf{A}(t) - \mathbf{S}(t) \rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \leq -\delta \|\mathbf{Q}\| + K_1 \quad (4)$$

  holds, then the system is throughput optimal and has a stationary distribution with all moments bounded.
- (S2) If there exists a finite constant T_2, and constants $K_2 > 0$ and $\eta > 0$, both independent of ϵ such that for all t_0

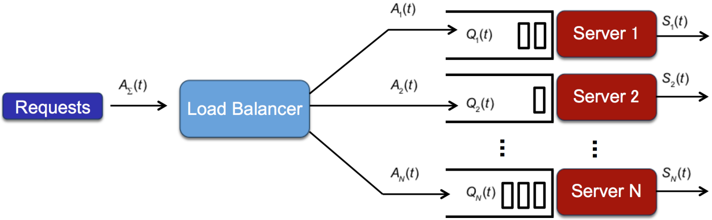
$$\mathbb{E}\left[\sum_{t=t_0}^{t_0+T_2-1} \langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t) \rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \leq -\eta \|\mathbf{Q}_\perp\| + K_2 \quad (5)$$

  holds, then the moments of perpendicular component with respect to any convex set $\mathcal{C}$ is bounded. (steady-state collapse to a convex set $\mathcal{C}$)

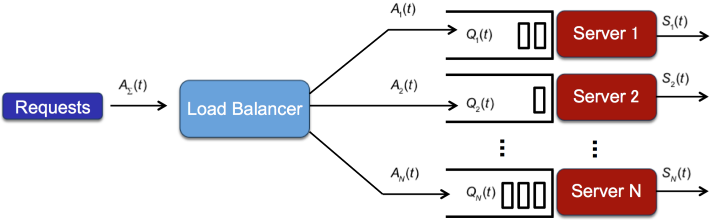
Clearly, if the convex set is $\mathbf{c} = \frac{1}{\sqrt{N}}\mathbf{1}$, (S1) implies (R1-R2) and (S2) implies (R3).

Outline

JBA in Homogeneous Servers

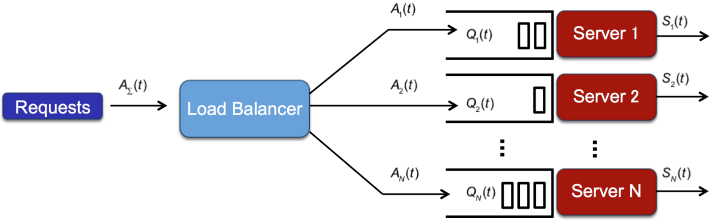


JBA in Homogeneous Servers



- All the N servers have the same average serve rate μ.

JBA in Homogeneous Servers



- All the N servers have the same average serve rate μ.
- The load balancer, under the JBA policy at each time-slot, randomly chose a queue among the queues that have workload less than the average workload at that time slot, and then forward all the incoming requests to that server.

S1 is Satisfied

- Let us first check (S1) by the choice $T = 1$:

$$\begin{aligned}
\mathbb{E}\left[\langle \mathbf{Q}(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}\right] &= \langle \mathbf{Q}, \mathbb{E}\left[\mathbf{A}|\mathbf{Q}\right]\rangle - \langle \mathbf{Q}, \boldsymbol{\mu}\rangle \\
&= \frac{\lambda_\Sigma}{L}\sum_{n=1}^{L} Q_n - \mu \sum_{n=1}^{N} Q_n \\
&= (\frac{\lambda_\Sigma}{N} - \mu)\sum_{n=1}^{N} Q_n - (\frac{\lambda_\Sigma}{N} - \frac{\lambda_\Sigma}{L})\sum_{n=1}^{N} - \frac{\lambda}{N}\sum_{n=L+1}^{N} Q_n \\
&\leq -\frac{\epsilon}{N}\left\|\mathbf{Q}\right\|_1 - \lambda_\Sigma \frac{N-L}{N}(Q_{L+1} - Q_L) \\
&\leq -\frac{\epsilon}{N}\left\|\mathbf{Q}\right\|
\end{aligned} \tag{6}$$

assume $Q_1(t) \leq Q_2(t) \leq \cdots \leq Q_L(t) \leq Q^*(t) < Q_{L+1}(t) \leq \cdots \leq Q_N(t)$, and $Q^*(t) = \frac{1}{N}\sum Q_n(t)$ is the average queue length.

S1 is satisfied, and hence JBA is throughput optimal

S2 is Satisfied

- Let us turn to check (S2) with the line $\mathbf{c}$ as the projection direction:

$$\begin{aligned}
\mathbb{E}\left[\langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t) = \mathbf{Q}\right] &\overset{(a)}{=} \langle \mathbf{Q}_\perp, \mathbb{E}\left[\mathbf{A}|\mathbf{Q}\right]\rangle - \langle \mathbf{Q}_\perp, \boldsymbol{\mu}\rangle \\
&= \langle \mathbf{Q}_\perp, \mathbb{E}\left[\mathbf{A}|\mathbf{Q}\right]\rangle \\
&= \frac{\lambda_\Sigma}{L}\sum_{n=1}^{L}(Q_n - Q^*) \\
&= -\frac{\lambda_\Sigma}{L}\sum_{n=1}^{L}|Q_n - Q^*| \\
&\leq -\frac{\lambda_\Sigma}{2N}\|\mathbf{Q}_\perp\| \\
&\leq -\frac{\mu}{2}\|\mathbf{Q}_\perp\|
\end{aligned} \tag{7}$$

for all $0 < \epsilon \leq \frac{N\mu}{2}$. (a) follows $\langle \mathbf{Q}_\perp, \mathbf{1}\rangle = 0$

S2 is verified and hence JBA is heavy-traffic optimality

Theorems

Theorem

For any λ_Σ *in the interior of* $\mathcal{R}$, *i.e.,* $\lambda_\Sigma < \mu_\Sigma$, *the JBA routing policy stabilizes the system, and all the moments of the stationary distribution are bounded, i.e., there exist finite constants* $\{M_r, r \in \mathbb{N}\}$ *such that* $\mathbb{E}\left[\left\|\overline{\mathbf{Q}}\right\|^r\right] \leq M_r$.

Theorem

Consider a set of load balancing system under JBA policy with the exogenous arrival process $\{A_\Sigma^{(\epsilon)}(t), t \geq 0\}$, *parameterized by* $\epsilon > 0$. *Then, each of these systems, the expectation of the sum queue length in steady state is lower bounded by*

$$\mathbb{E}\left[\sum_{n=1}^{N} \overline{Q}_n^{(\epsilon)}\right] \geq \frac{\zeta^{(\epsilon)}}{2\epsilon} - K \tag{8}$$

where $\zeta^{(\epsilon)} = (\sigma_\Sigma^{(\epsilon)})^2 + \nu_\Sigma^2 + \epsilon^2$, $K = \frac{NS_{\max}}{2}$. *Therefore, in the heavy-traffic limit as* $\epsilon \downarrow 0$, *assuming the* $(\sigma_\Sigma^{(\epsilon)})^2$ *converges to a constant* σ_Σ^2, *we have*

$$\liminf_{\epsilon \downarrow 0} \epsilon \mathbb{E}\left[\sum_{n=1}^{N} \overline{Q}_n^{(\epsilon)}\right] \geq \frac{\zeta}{2}, \tag{9}$$

Theorems (Cont'd)

Theorem

Consider a set of load balancing system with the exogenous arrival process $\{A_\Sigma^{(\epsilon)}(t), t \geq 0\}$*, parameterized by* $\epsilon > 0$*, such that the mean arrival rate is* $\lambda_\Sigma^{(\epsilon)} = \mu_\Sigma - \epsilon$ *and variance is denoted as* $(\sigma_\Sigma^{(\epsilon)})^2$*. Under the JBA algorithm,* $\{\mathbf{Q}^{(\epsilon)}(t), t \geq 0\}$ *converges in distribution to* $\overline{\mathbf{Q}}^{(\epsilon)}$*. Then, there exist finite constants* $\{M_r, r \in \mathbb{N}\}$ *which are independent of* ϵ *such that for all* $r \in \mathbb{N}$*,*

$$\mathbb{E}\left[\left\|\overline{\mathbf{Q}}_\perp^{(\epsilon)}\right\|\right] \leq M_r, \tag{10}$$

for all system with $0 < \epsilon \leq \frac{N\mu}{2}$*.*

Theorems (Cont'd)

Theorem

Consider a set of load balancing system with the exogenous arrival process $\{A_{\Sigma}^{(\epsilon)}(t), t \geq 0\}$, *parameterized by* $\epsilon > 0$, *such that the mean arrival rate is* $\lambda_{\Sigma}^{(\epsilon)} = \mu_{\Sigma} - \epsilon$ *and variance is denoted as* $(\sigma_{\Sigma}^{(\epsilon)})^2$. *Under the JBA algorithm,* $\{\mathbf{Q}^{(\epsilon)}(t), t \geq 0\}$ *converges in distribution to* $\overline{\mathbf{Q}}^{(\epsilon)}$. *For each system with* $0 < \epsilon \leq \frac{N\mu}{2}$, *the steady state average queue length satisfies*

$$\mathbb{E}\left[\sum_{n=1}^{N} \overline{Q}_n^{(\epsilon)}\right] \leq \frac{\zeta^{(\epsilon)}}{2\epsilon} + \overline{B}^{(\epsilon)}, \tag{11}$$

where $\zeta^{(\epsilon)}$ *is the same as in the lower bound, and* $\overline{B}^{(\epsilon)}$ *is* $o(\frac{1}{\epsilon})$, *i.e.,* $\lim_{\epsilon \downarrow 0} \epsilon \overline{B}^{(\epsilon)} = 0$.

Therefore, assuming the variance $(\sigma_{\Sigma}^{(\epsilon)})^2$ *converges to a constant* σ_{Σ}^2, *the upper bound becomes*

$$\limsup_{\epsilon \downarrow 0} \epsilon \mathbb{E}\left[\sum_{n=1}^{N} \overline{Q}_n^{(\epsilon)}\right] \leq \frac{\zeta}{2} \tag{12}$$

How about Random Load Balancing

Under random load balancing, we have $\mathbf{A} - \mathbf{S} = -\frac{\epsilon}{N}\mathbf{1}$

- For (S1), we have $\mathbb{E}\left[\langle \mathbf{Q}(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}\right] = -\frac{\epsilon}{N}\left\|\mathbf{Q}\right\| \leq -\frac{\epsilon}{2N}\left\|\mathbf{Q}\right\|$
- For (S2), we have $\mathbb{E}\left[\langle \mathbf{Q}_{\perp}(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t) = \mathbf{Q}\right] = 0$ for all t

Consider steady-state collapse, random load balancing actually does no harm in the sense that it would not incur any positive drift.

Can we utilize this fact to turn bad to good?

From 1 to any finite T

The load balancer uses JBA, Power of d, or JSQ every T time-slots, otherwise just random routing.

- For (S1), it is trivial to hold.
- For (S2), by letting $T_2 = T$, we have

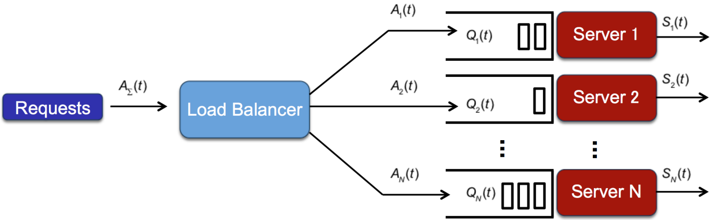
$$
\begin{aligned}
&\mathbb{E}\left[\sum_{t=t_0}^{t_0+T_2-1} \langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \\
&= \sum_{t=t_0}^{t_0+T_2-1} \mathbb{E}\left[\langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \\
&\overset{(a)}{=} \sum_{t=t_0}^{t_0+T_2-1} \mathbb{E}\left[\mathbb{E}\left[\langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t)\right] | \mathbf{Q}(t_0) = \mathbf{Q}\right] \\
&= \mathbb{E}\left[-\eta \left\|\mathbf{Q}_\perp(t^*)\right\| | \mathbf{Q}(t_0) = \mathbf{Q}\right] \\
&\leq -\eta \left\|\mathbf{Q}_\perp(t_0)\right\| + DT
\end{aligned}
\tag{13}
$$

where (a) follows from tower property of conditional expectation.

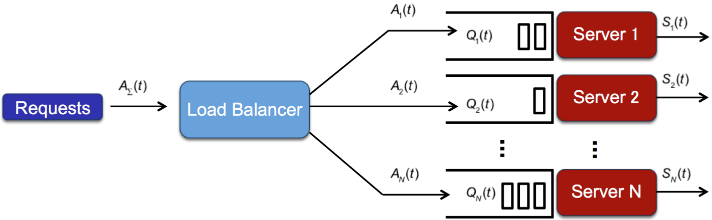
Outline

1. Introduction and Motivation
   - Why do we need an effective and fast cloud?
   - Load Balancing and Previous Works
2. Throughput and Heavy-traffic optimality of General Load Balancing
   - Model, Challenges, Contributions and Key insights
   - Methodology and Sufficient Conditions
   - Homogeneous Servers
   - Heterogeneous Servers
   - Conclusions

JBA in Heterogeneous Servers

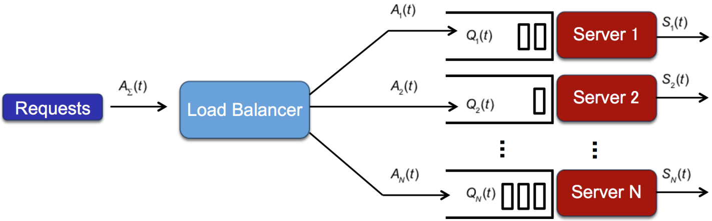


JBA in Heterogeneous Servers



- The N servers do not have the same average serve rate, each with μ_n.

JBA in Heterogeneous Servers



- The N servers do not have the same average serve rate, each with μ_n.
- The load balancer, under the JBA policy at each time-slot, randomly with probability proportional to the service rate μ_n to chose a queue among the queues that have workload less than the average workload at that time slot, and then forward all the incoming requests to that server.

$$\mathbb{P}(R_i) = \frac{\mu_i}{\sum_{i \leq L} \mu_i} \quad \text{if} \quad i \leq L \tag{14}$$

JBA in Heterogeneous Servers (Cont'd)

Under the JBA load balancing, it can be shown that (S1) and (S2) still hold, i.e.,

- For (S1), we have $\mathbb{E}\left[\langle \mathbf{Q}(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}\right] \leq -\delta \left\| \mathbf{Q} \right\|$ for some $\delta > 0$
- For (S2), we have $\mathbb{E}\left[\langle \mathbf{Q}_{\perp}(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t) = \mathbf{Q}\right] \leq -\eta \left\| \mathbf{Q}_{\perp}(t) \right\|$ for some $\eta > 0$ independent of ϵ

Under purely random load balancing with proportional probability, we have

- For (S1), we have $\mathbb{E}\left[\langle \mathbf{Q}(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}\right] \leq -\delta \left\| \mathbf{Q} \right\|$ for some $\delta > 0$
- For (S2), we have $\mathbb{E}\left[\langle \mathbf{Q}_{\perp}(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t) = \mathbf{Q}\right] \leq \epsilon \left\| \mathbf{Q}_{\perp}(t) \right\|$ for all t

From 1 to any finite T

The load balancer uses JBA, Power of d, or JSQ every T time-slots, otherwise just random routing with proportional probability over N servers.

- For (S1), it is trivial to hold.
- For (S2), by letting $T_2 = T$, we have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=t_0}^{t_0+T_2-1} \langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \\
&= \sum_{t=t_0}^{t_0+T_2-1} \mathbb{E}\left[\langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t_0) = \mathbf{Q}\right] \\
&= \sum_{t=t_0}^{t_0+T_2-1} \mathbb{E}\left[\mathbb{E}\left[\langle \mathbf{Q}_\perp(t), \mathbf{A}(t) - \mathbf{S}(t)\rangle | \mathbf{Q}(t)\right] | \mathbf{Q}(t_0) = \mathbf{Q}\right] \\
&\le ((T-1)\epsilon - \eta)\left\|\mathbf{Q}_\perp(t_0)\right\| + DT^2 \\
&\le -\frac{1}{2\eta}\left\|\mathbf{Q}_\perp(t_0)\right\| + DT^2
\end{aligned} \tag{15}$$

for all $0 < \epsilon \le \frac{\eta}{2T}$, hence smaller ϵ means a larger T!.

Outline

Conclusions

- Can we generalize existing load balancing algorithm?
  - Yes, the proposed JBA policy
- Do we really need sampling for each time-slot for heavy-traffic optimality?
  - No, we can actually sampling every T slots whenever $T\epsilon \leq \alpha$ is satisfied.

Thank you!

Q & A