清華大學

Tsinghua University

Modeling and Analyzing Latency in the Memcached System

Wenxue Cheng, Fengyuan Ren, Wanchun Jiang (CSU), Tong Zhang

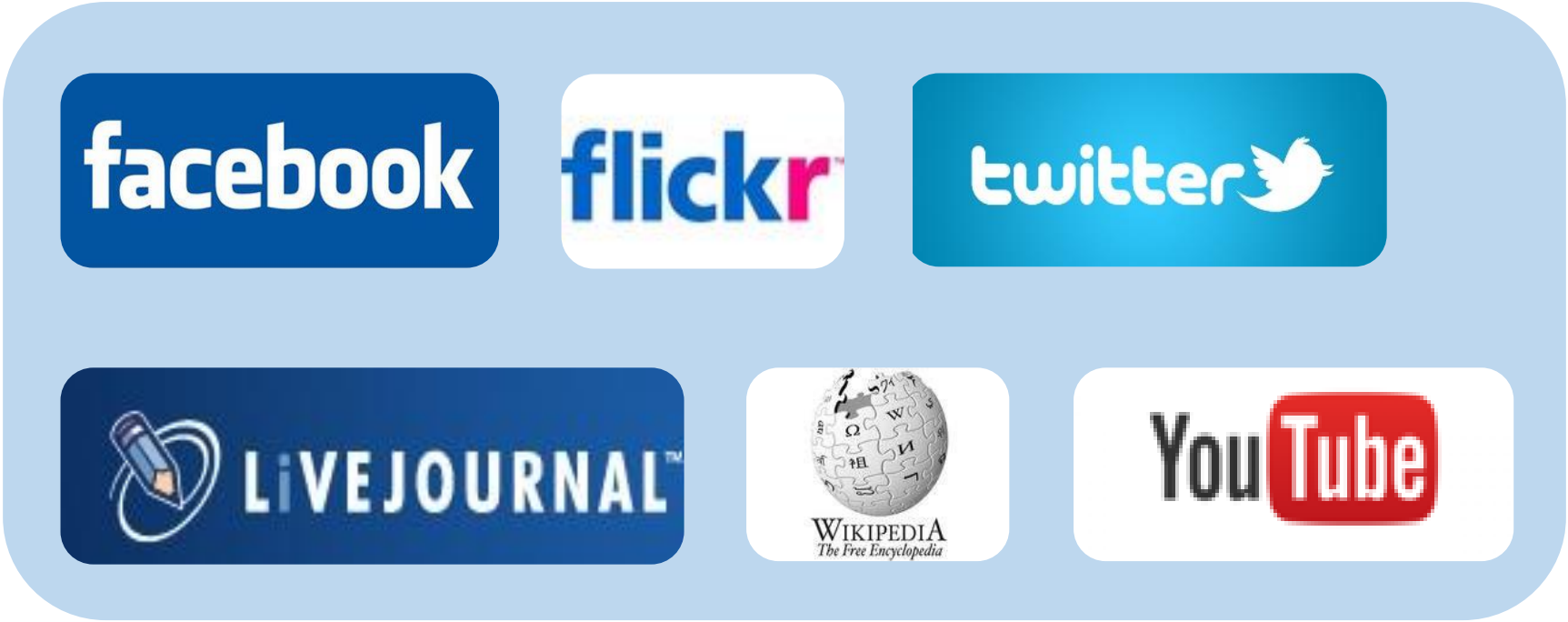
NNS Group @ Tsinghua University

IEEE ICDCS 2017, Atlanta, USA

Background

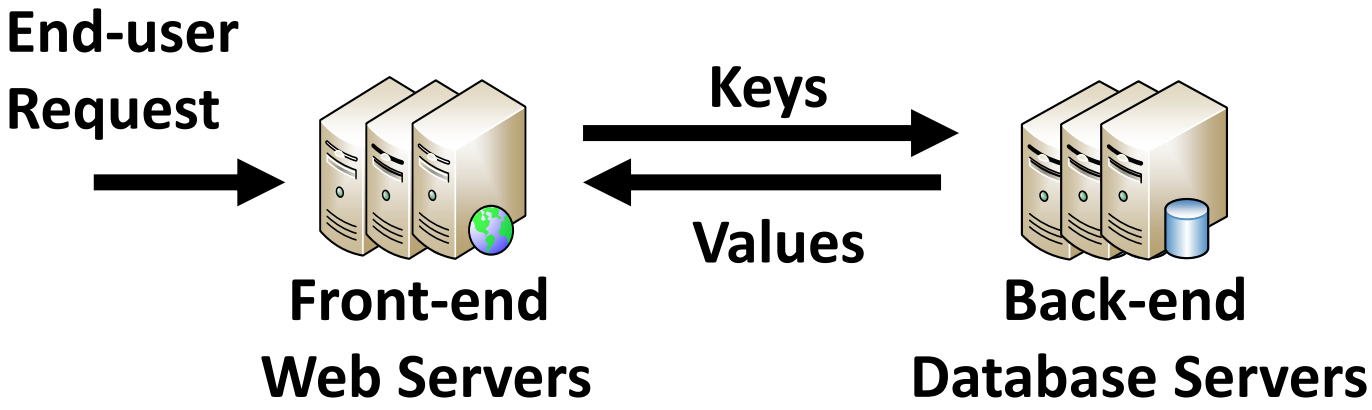
Memcached System

- Key-Value system
- In-memory caching solution
- Speed up searching applications



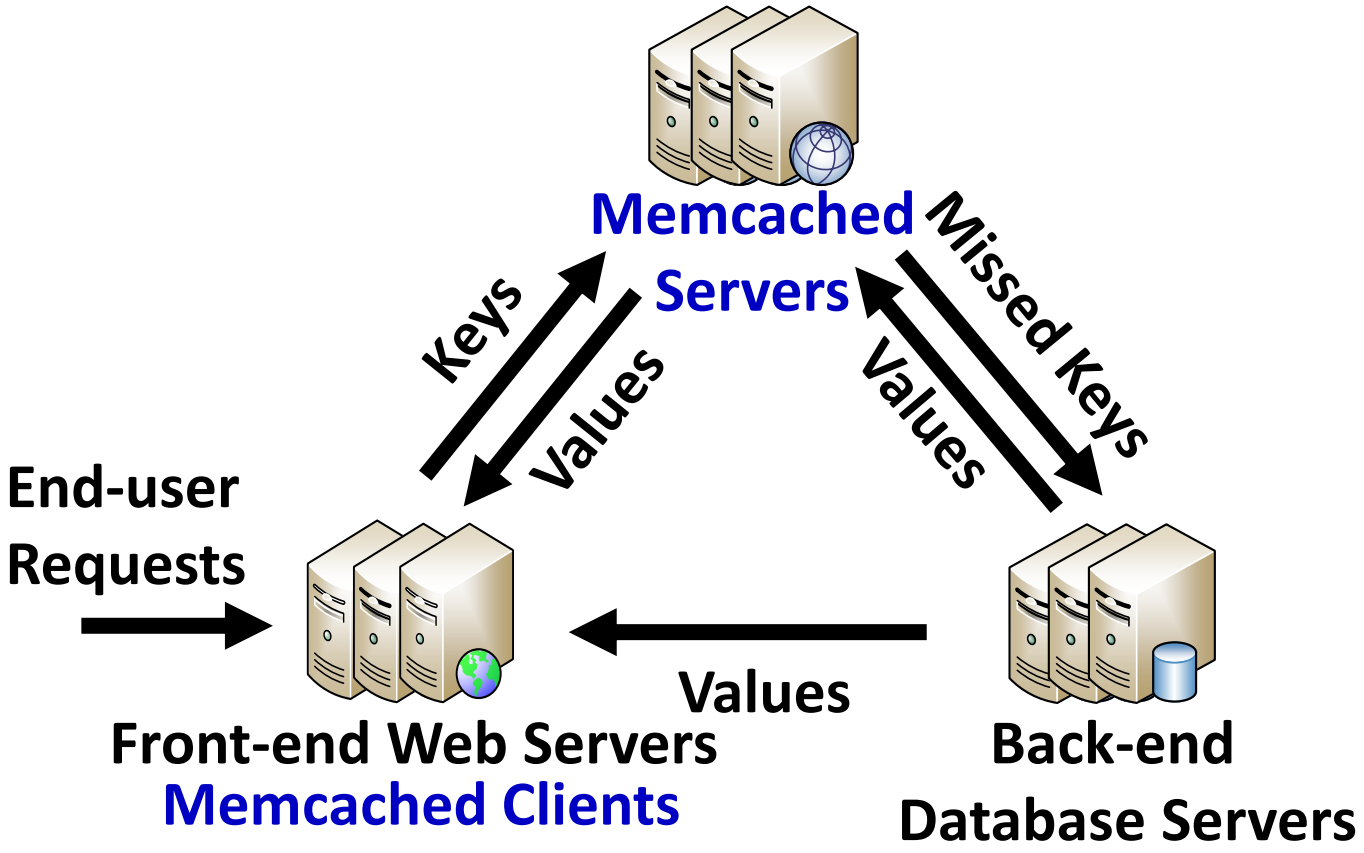
Background

Traditional Websites



Background

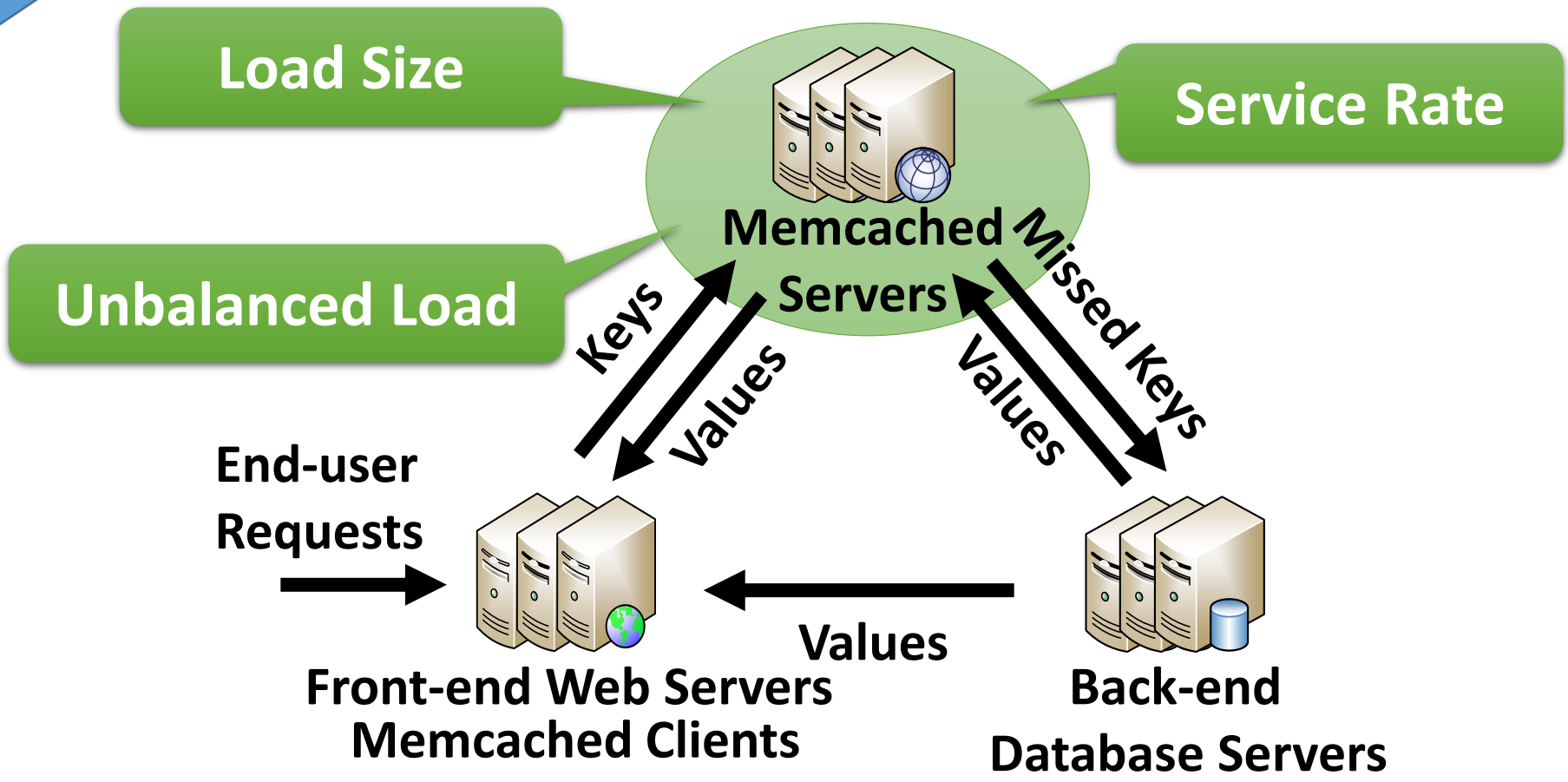
Memcached Architecture



Latency is the most pivotal performance metric.

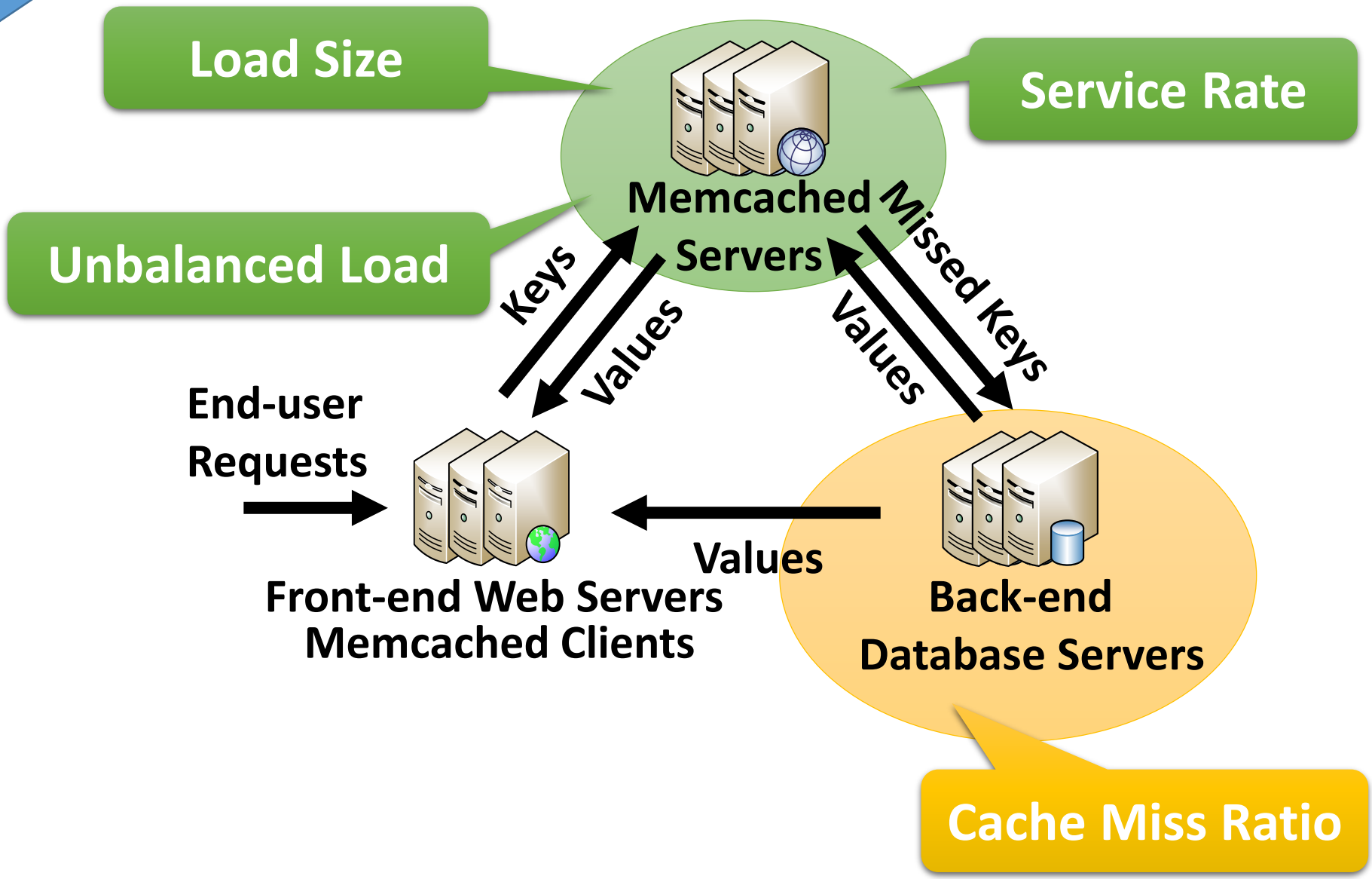
Background

Factors on Latency



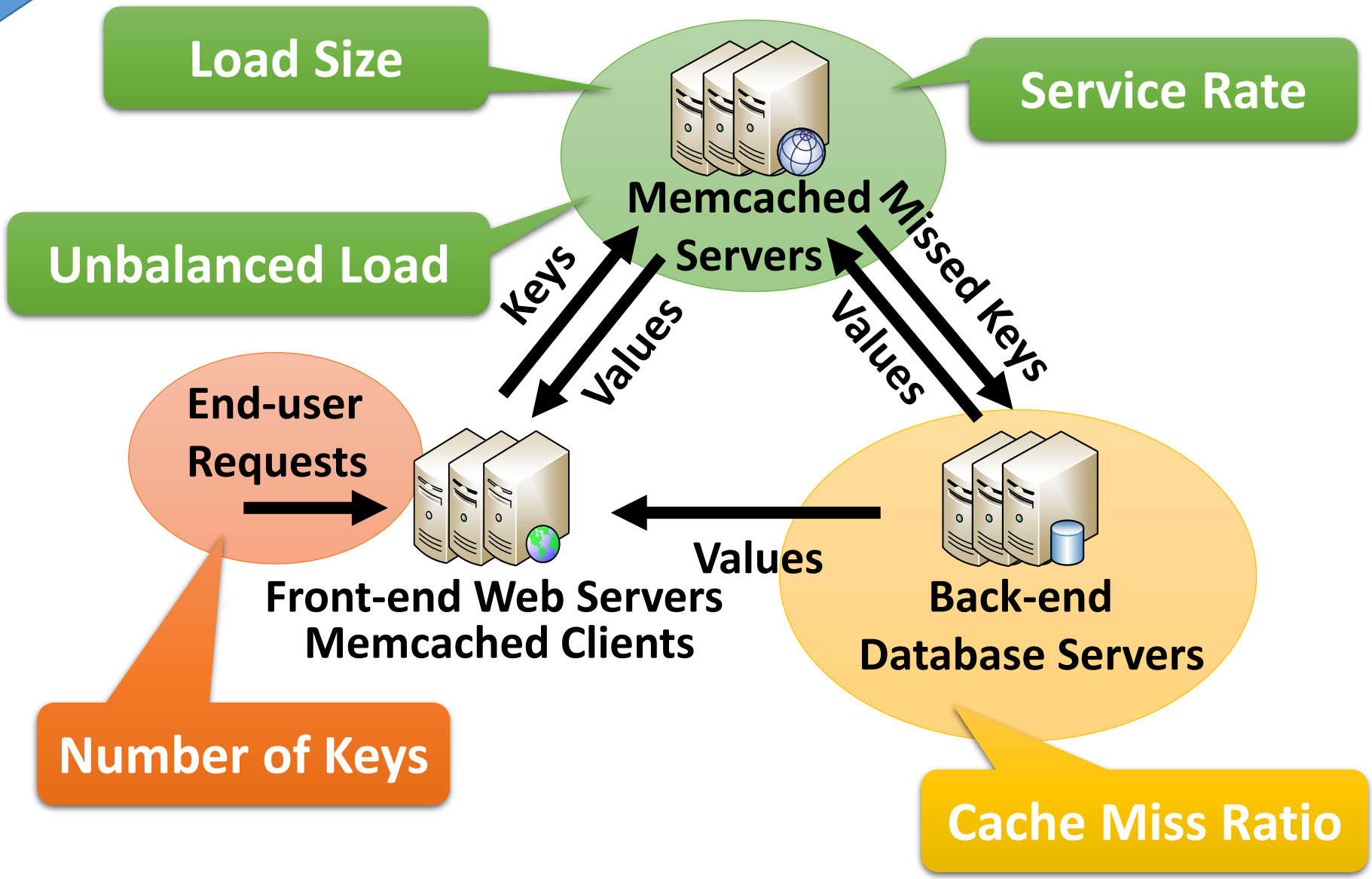
Background

Factors on Latency



Background

Factors on Latency



Background

Factors on Latency

Load Size

Service Rate

- How much improvement on latency can be achieved by optimizing each factor?

Values

Front-end Web Servers
Memcached Clients

Back-end
Database Servers

Number of Keys

Cache Miss Ratio

Background

Factors on Latency

Load Size

Service Rate

- How much improvement on latency can be achieved by optimizing each factor?

Theoretical Model & Quantitative Analysis

Values

Front-end Web Servers
Memcached Clients

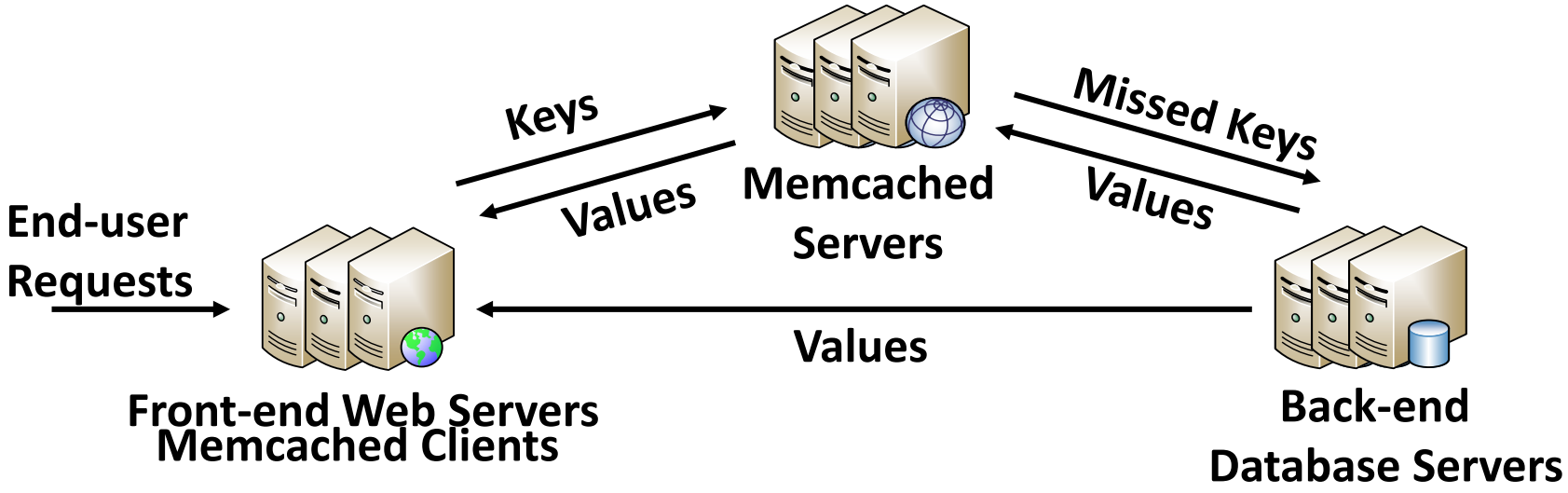
Back-end
Database Servers

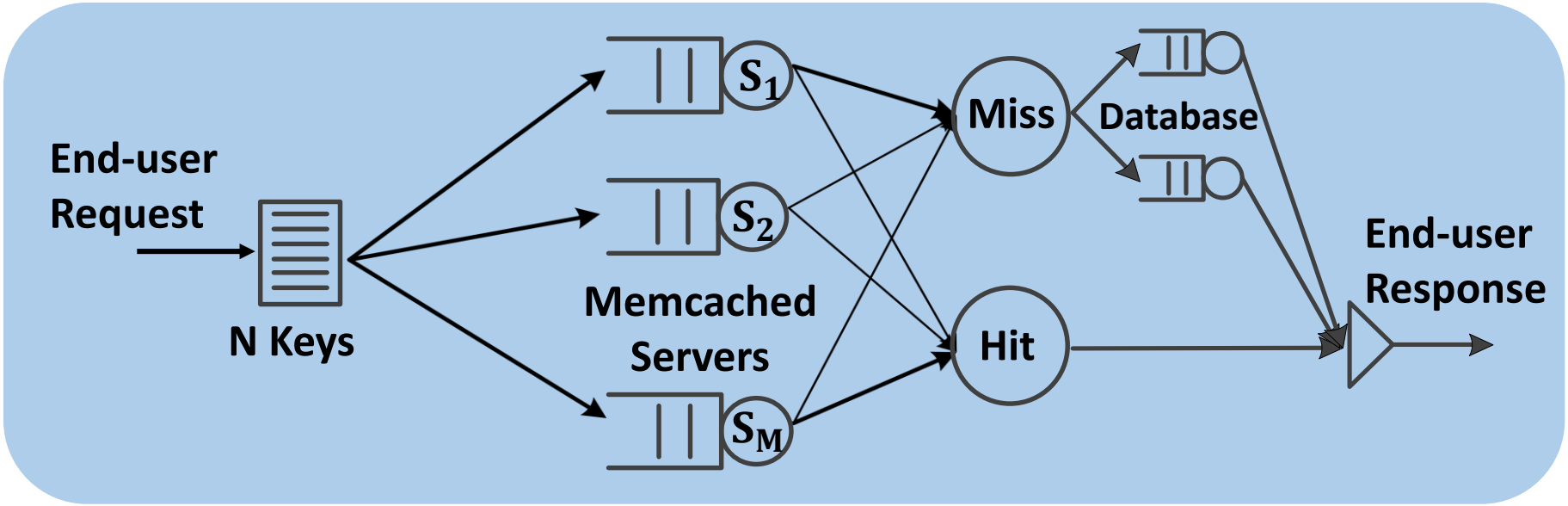
Number of Keys

Cache Miss Ratio

Model

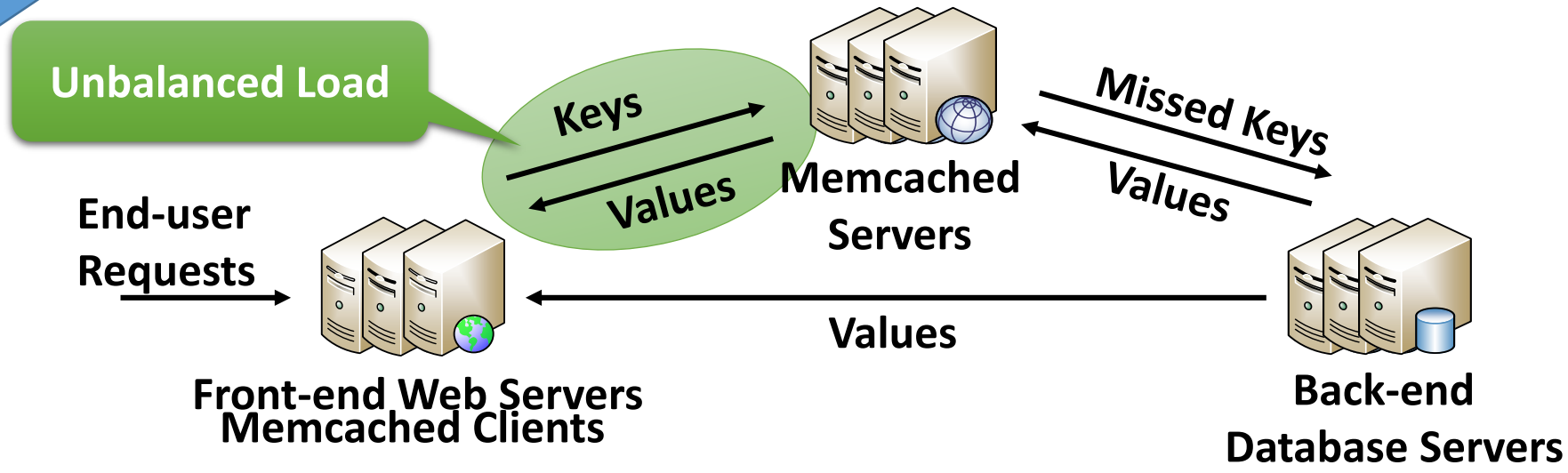
Model for Memcached

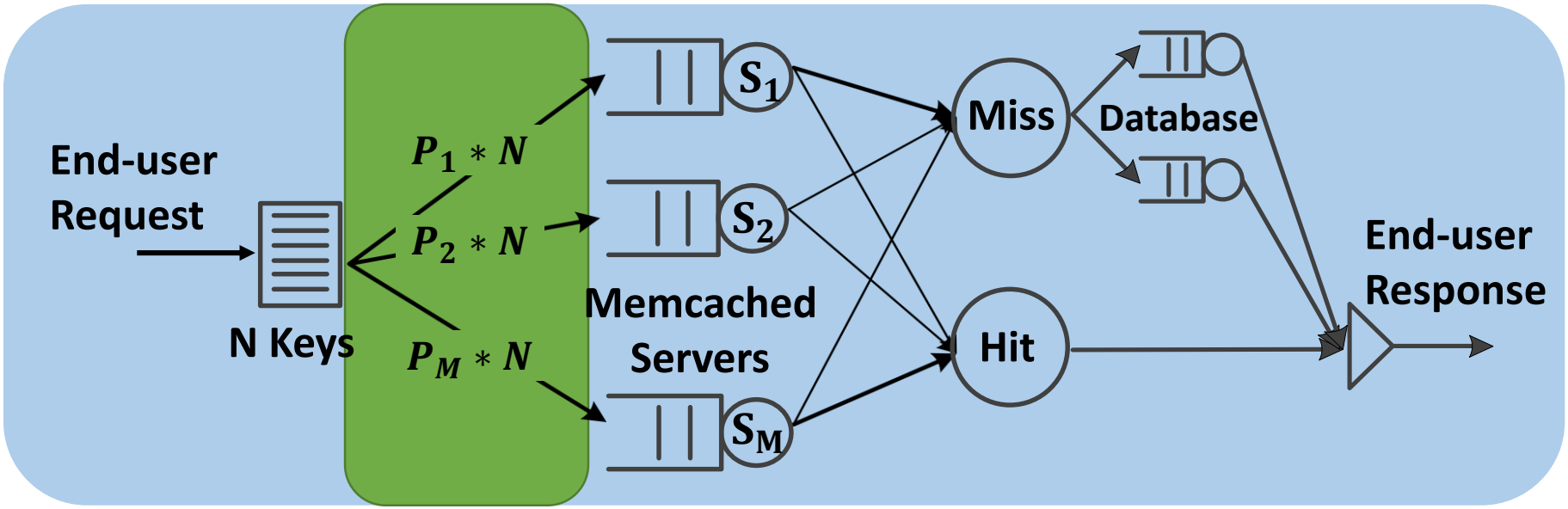




Model

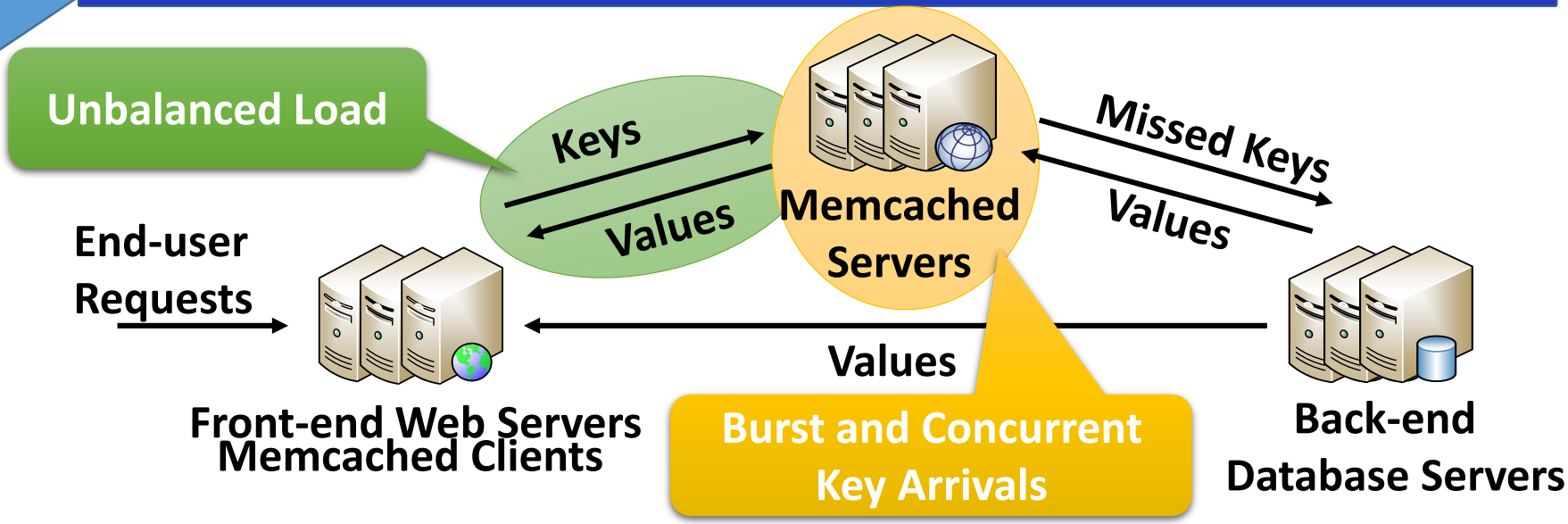
Specific Features

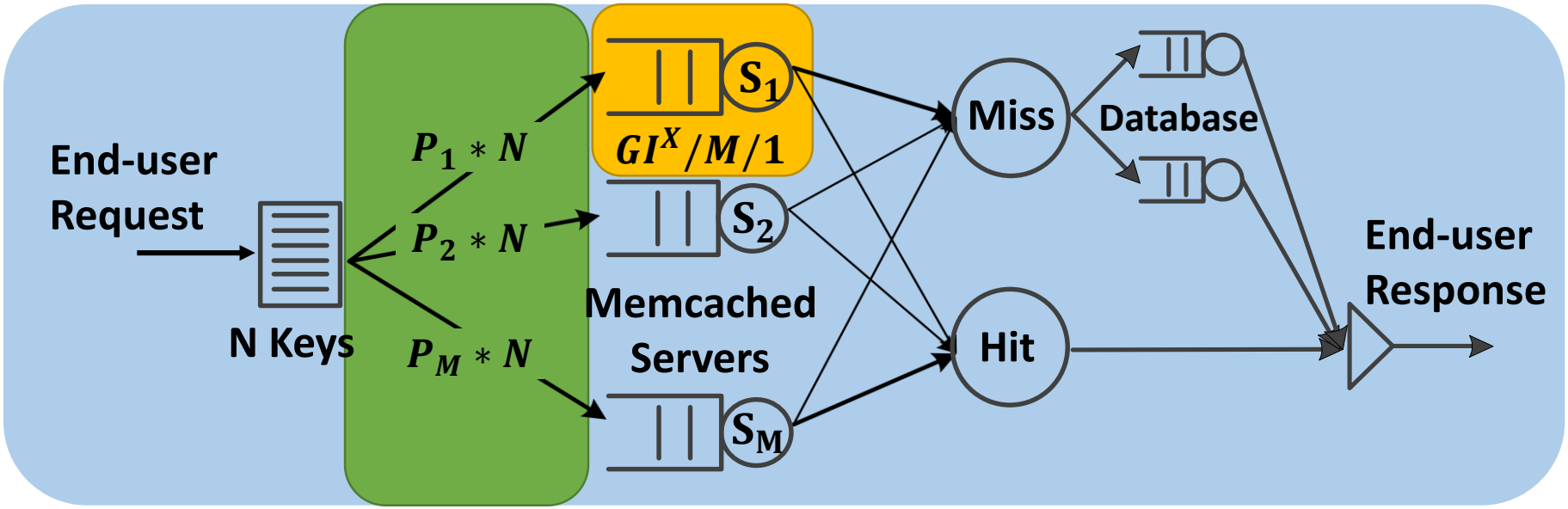




Model

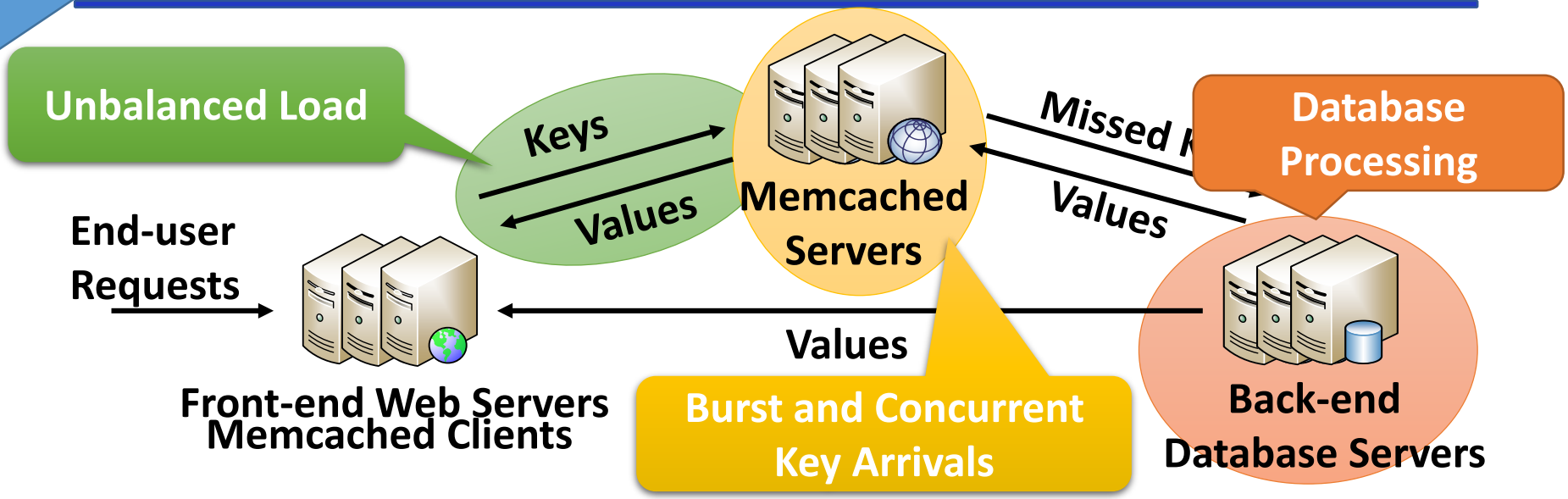
Specific Features

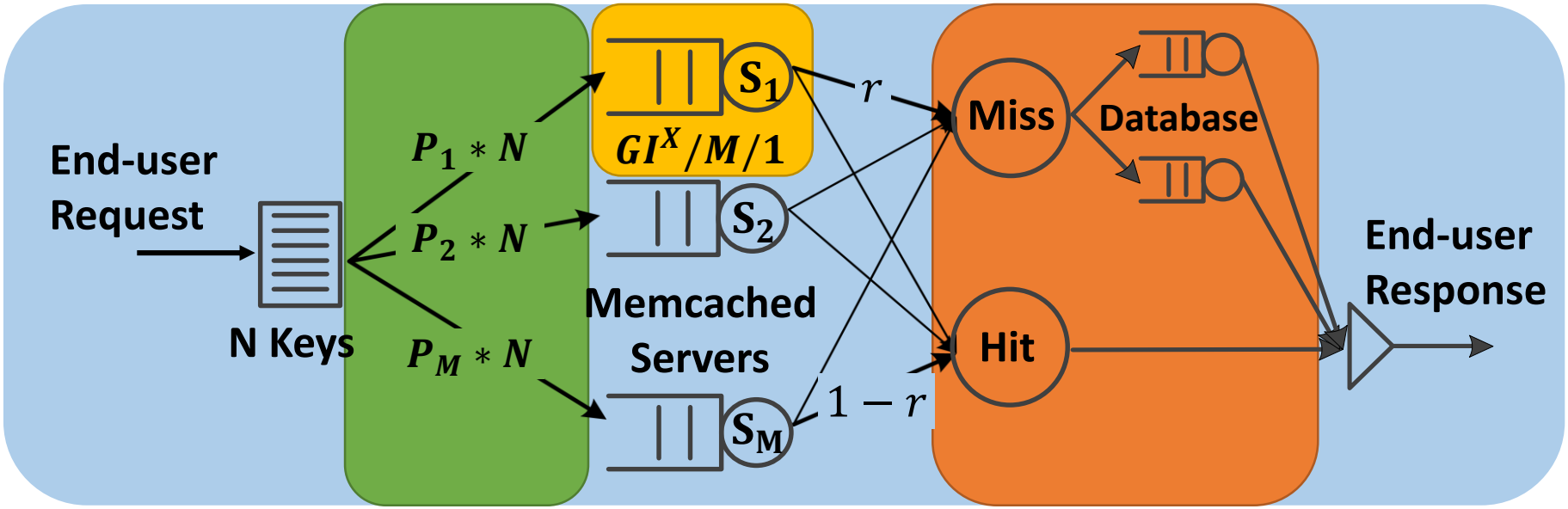




Model

Specific Features

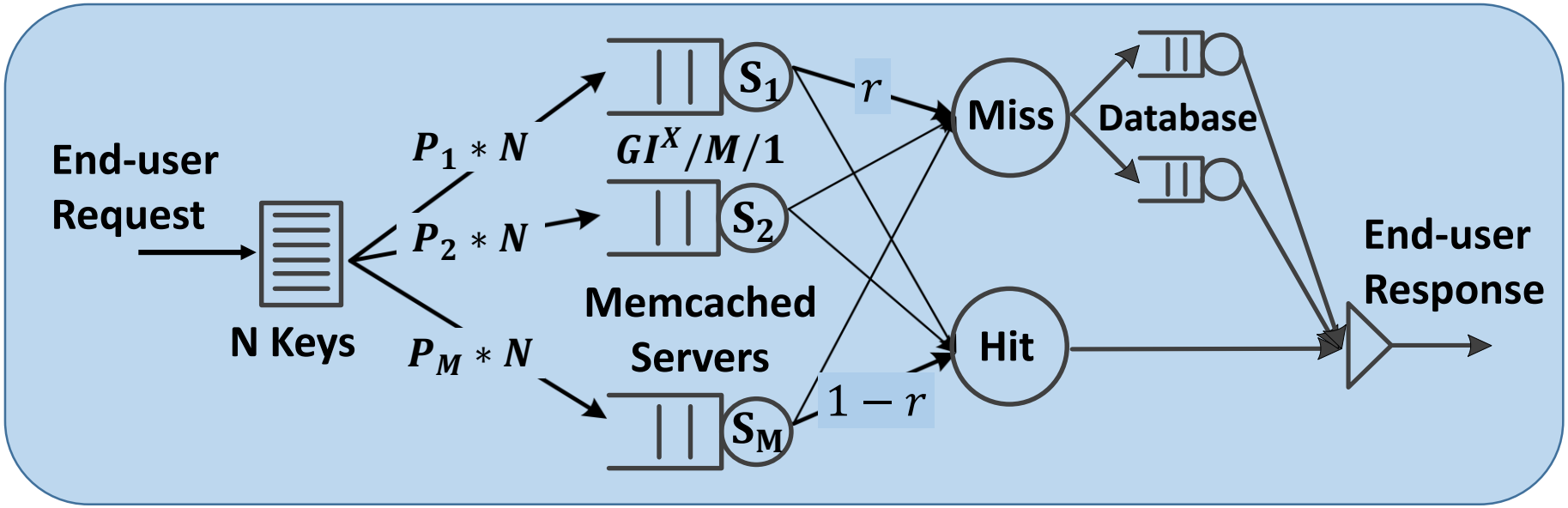




Model

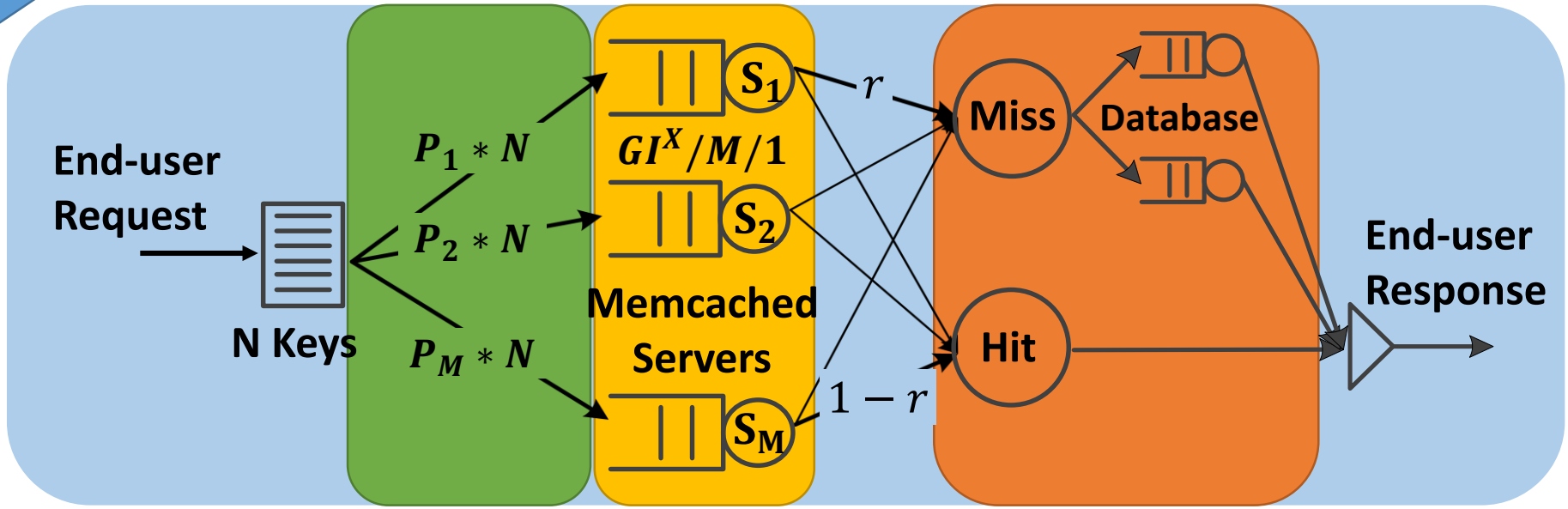
Assumptions

- One end-user request simultaneously generates multiple independent keys.
- Service times are exponential, both at Memcached servers and database.[CoNEXT'13][NSDI'15]



Estimation

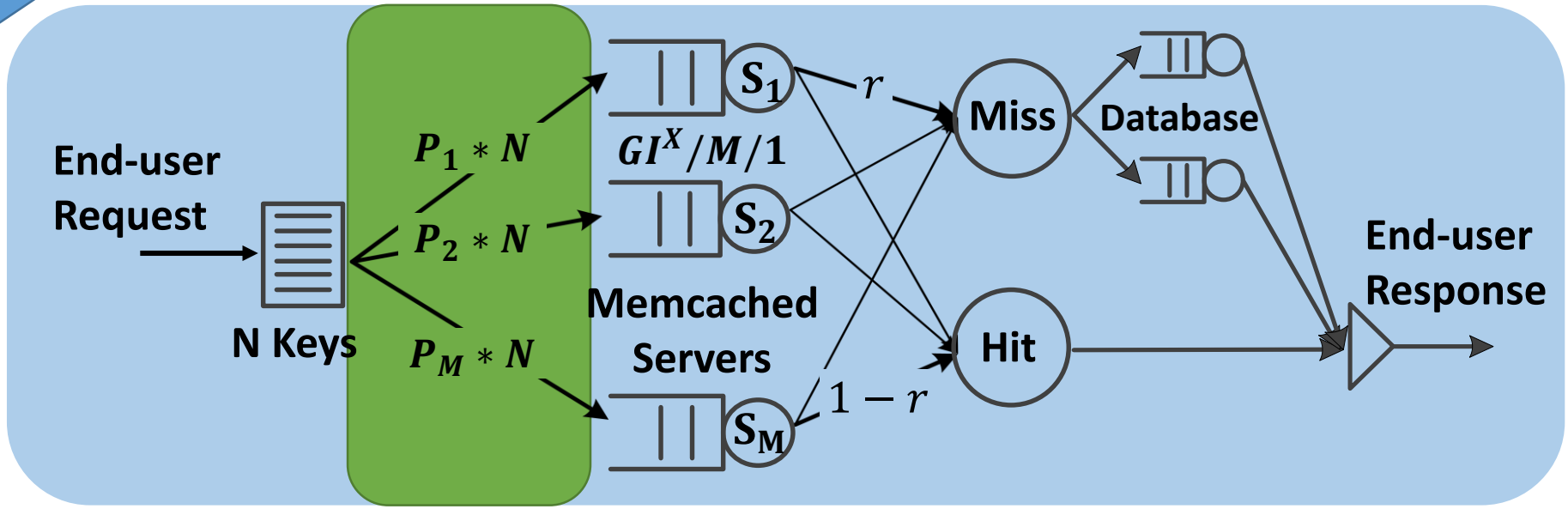
Latency Composition



- Network Latency
- Processing Latency at Memcached Servers
- Processing Latency at Database

Estimation

Network Latency



- Network Utilization << 10% ➔ No queueing in network transmission
  - Request rate: $10^5/s$
  - Request size: Key < 200B, Value < 1KB[SIGMETRICS'12]

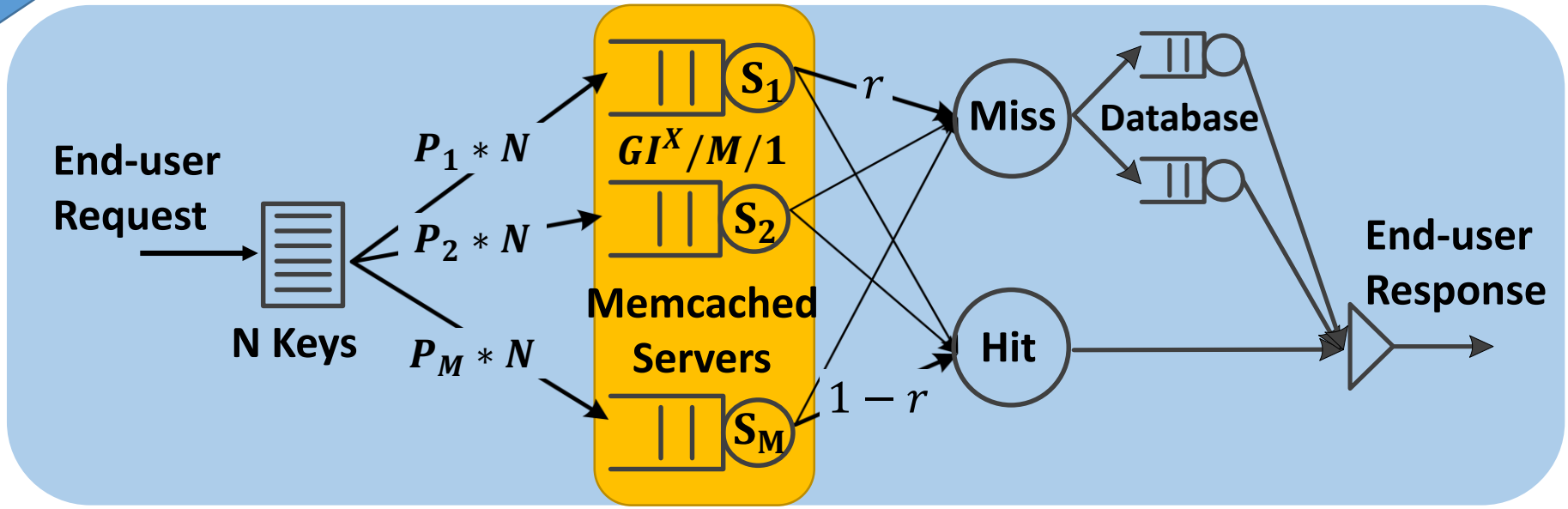
➔Workload << 1Gbps

➔Network bandwidth 10Gbps

Network latency is almost constant

Estimation

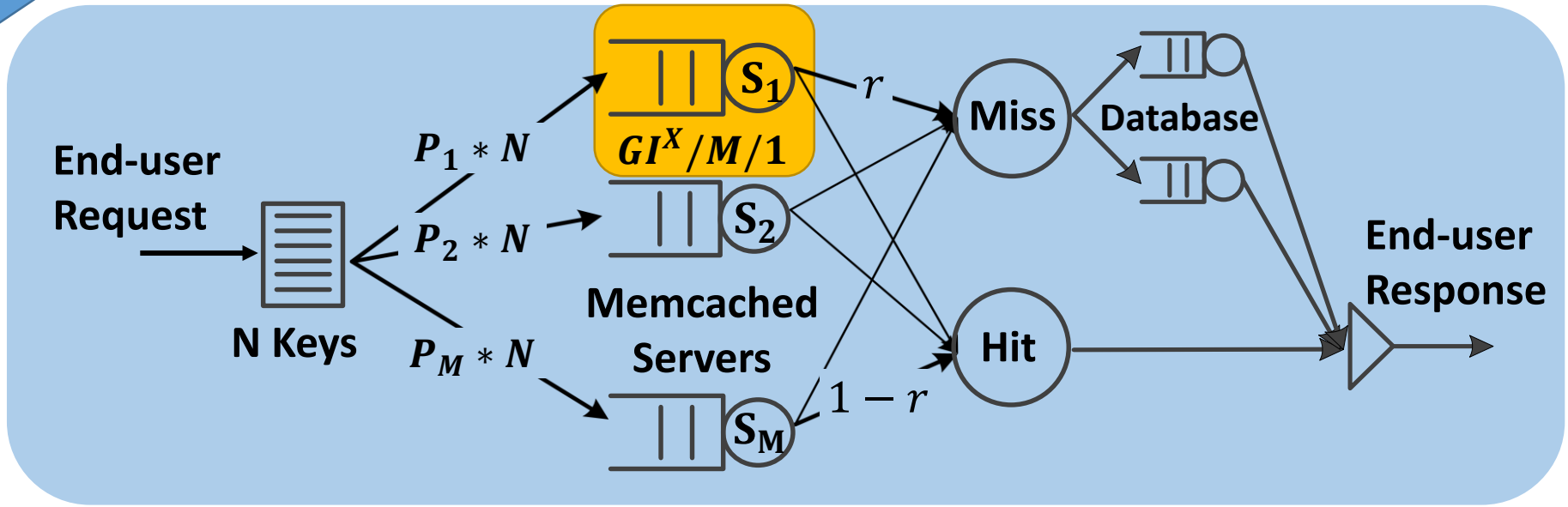
Latency at Memcached servers



- Latency for 1 Key → Maximum Latency for N Keys

Estimation

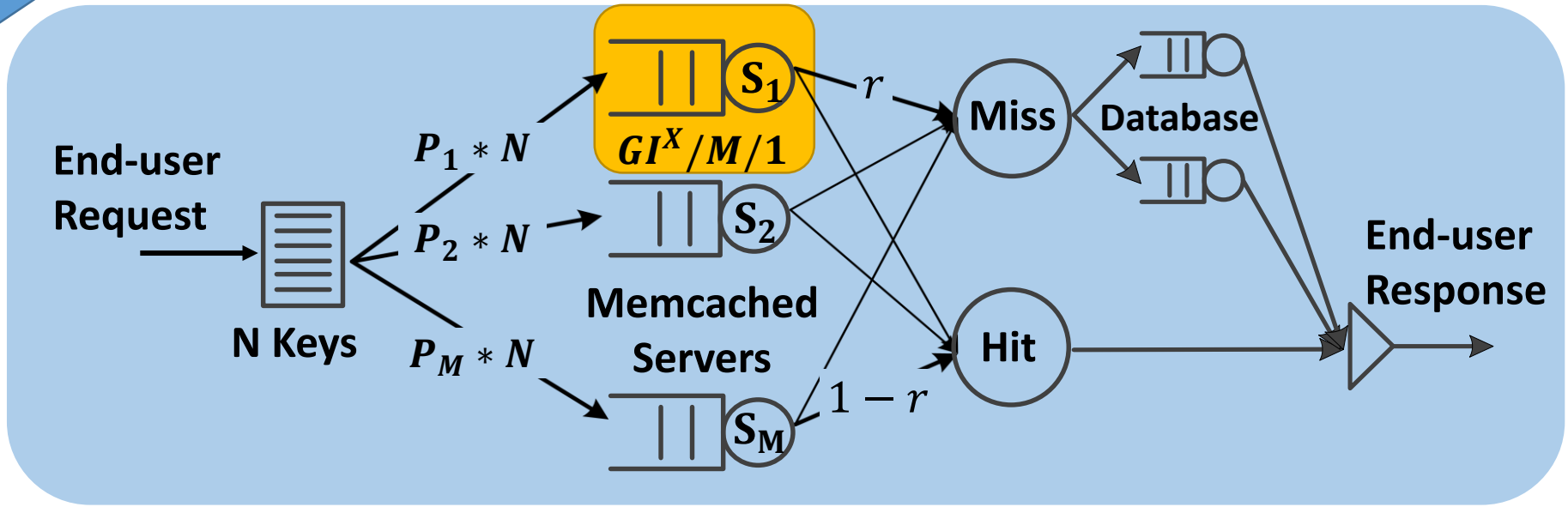
Latency at Memcached servers



- Latency for 1 Key
  - Queuing at each Memcached server : $GI^X/M/1$
    - GI: Any distribution ➔ Burst arrival of keys
    - X: Batch blocks ➔ Concurrent arrival of keys
  - $GI^X/M/1$ ➔ $GI/M/1$
    - Response time for a batch < Latency for 1 Key ≤ Response time for a batch

Estimation

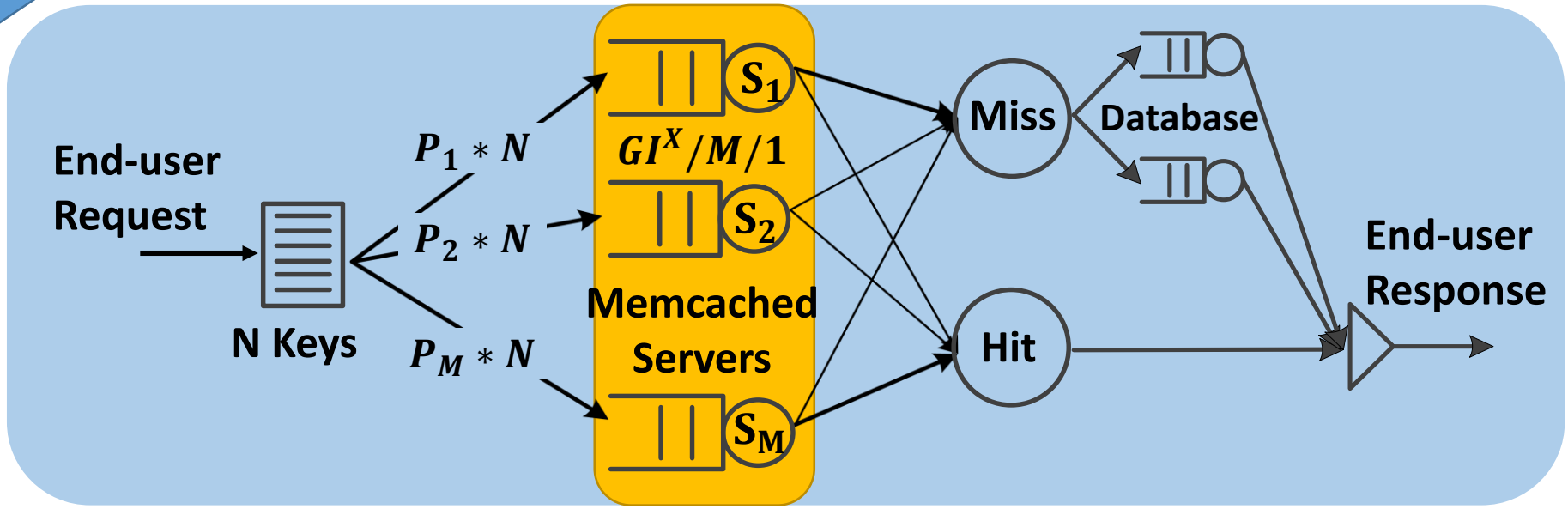
Latency at Memcached servers



- Latency for 1 Key
  - Queuing at each Memcached server : $GI^X/M/1$
    - GI: Any distribution ➔ Burst arrival of keys
    - X: Batch blocks ➔ Concurrent arrival of keys
  - $GI^X/M/1$ ➔ $GI/M/1$
    - Waiting time for a batch < Latency for 1 Key ≤ Response time for a batch

Estimation

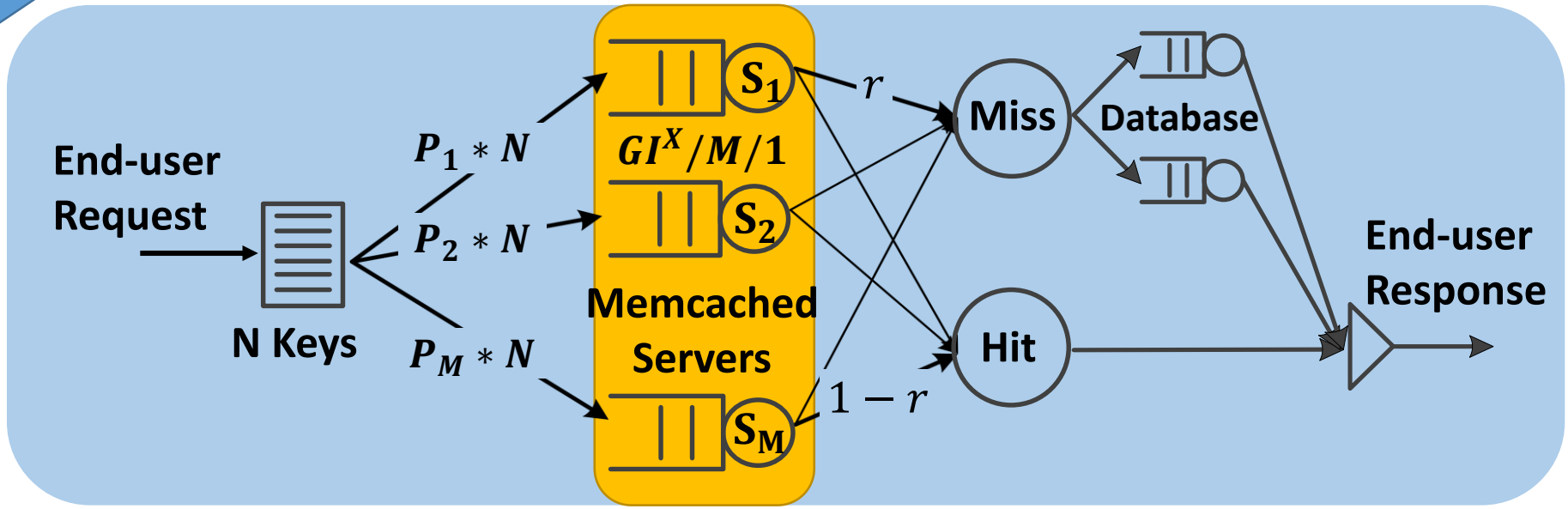
Latency at Memcached servers



- Maximum Latency for N Keys
  - Balanced load ➔ $\approx \left(\frac{N}{N+1}\right)_{th}$ of latency for 1 key
  - Unbalanced load among Memcached servers : $\{P_j\}_{j=1}^{M}$
    - $\in \left[\left(\frac{N}{N+1}\right)^{\frac{1}{p_{max}}}, \left(\frac{N}{N+1}\right)\right]_{th}$ of latency for a key at the heaviest server

Estimation

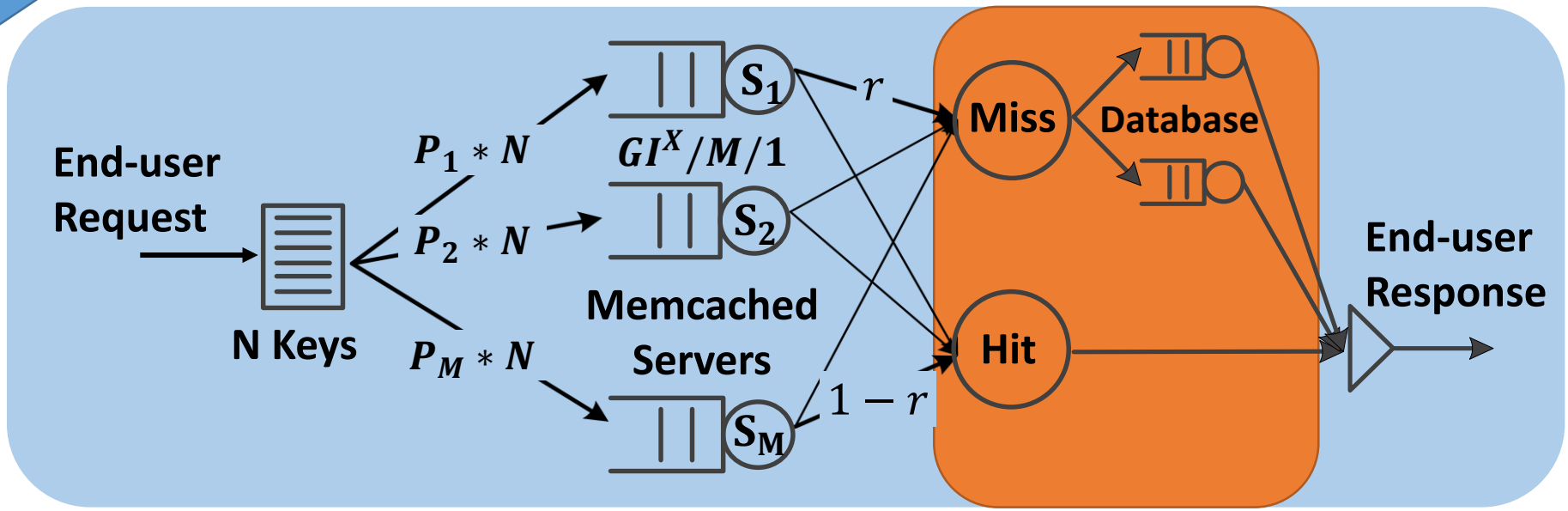
Latency at Memcached servers



- Maximum Latency for N Keys
  - Balanced load ➔ $\approx \left(\frac{N}{N+1}\right)_{\text{th}}$ of latency for 1 key
  - Unbalanced load among Memcached servers : $\{P_j\}_{j=1}^{M}$
    - $\in \left[\left(\frac{N}{N+1}\right)^{\frac{1}{p_{max}}}, \left(\frac{N}{N+1}\right)\right]_{th}$ of latency for 1 key at the heaviest server

Estimation

Latency at Database



- Depends on the number of keys N and cache miss ratio r
  - $(1-r)^N$: All-hit cases➔ Zero latency.
  - $1-(1-r)^N$: Miss cases ➔ Large latency.

Estimation

Symbols

Symbol	Definition
N	Number of keys generated from an end user request.
M	Number of Memcached servers.
$\{p_j\}_{j=1}^{M}$	Load distribution among Memcached servers.
q	Concurrent probability of keys.
X	Batch size for concurrent keys.
T_X	Inter-arrival gap of batches.
λ	Arrival rate of keys.
μ_S	Service rate of one Memcached server.
r	Cache miss ratio.
μ_D	Service rate at database.

Estimation

Latency Estimation

Latency $T(N)$ of an end-user request that generates N keys is separately bounded by three parts,

$$\max\{T_N(N), T_S(N), T_D(N)\} \leq T(N) \leq T_N(N) + T_S(N) + T_D(N)$$

- The network latency $T_N(N)$ is almost constant.
- The processing latency at Memcached servers $T_S(N)$ satisfies

$$\max\left\{\frac{\ln\delta + \frac{1}{p_{max}}\ln(N+1)}{(1-\delta)(1-q)\mu_S}, 0\right\} \leq E[T_S(N)] \leq \frac{\frac{1}{p_{max}}\ln(N+1)}{(1-\delta)(1-q)\mu_S}$$

- The processing latency at database $T_D(N)$ can be estimated by

$$E[T_D(N)] \approx \frac{1-(1-r)^N}{\mu_D} * \ln\left(\frac{N*r}{1-(1-r)^N} + 1\right)$$

Validation

Basic Validation

- Experimental Setup
  - 2 client machines and 4 server machines
    - Intel CoreTM i5-5200U CPU and 8GB memory
    - Same Rank, 1Gbps links
  - Workload
    - Mutilate: 512 connections
    - Measurements of Facebook [Sigmetrics'12]

Latency	Estimation	Experiment	Confidence Interval
Network	$20\mu s$	$20\mu s$	$[18.12\mu s, 21.68\mu s]$
Memcached Servers	$351\mu s \sim 366\mu s$	$368\mu s$	$[362\mu s, 373\mu s]$
Database	$836\mu s$	$867\mu s$	$[855\mu s, 879\mu s]$
Total	$\mathbf{836\mu s \sim 1222\mu s}$	$\mathbf{1144\mu s}$	$[\mathbf{1128\mu s}, \mathbf{1160\mu s}]$

Background

Factors on Latency

Load Size

Service Rate

- How much improvement on latency can be achieved by optimizing each factor?

Quantitative Analysis

Values

Front-end Web Servers
Memcached Clients

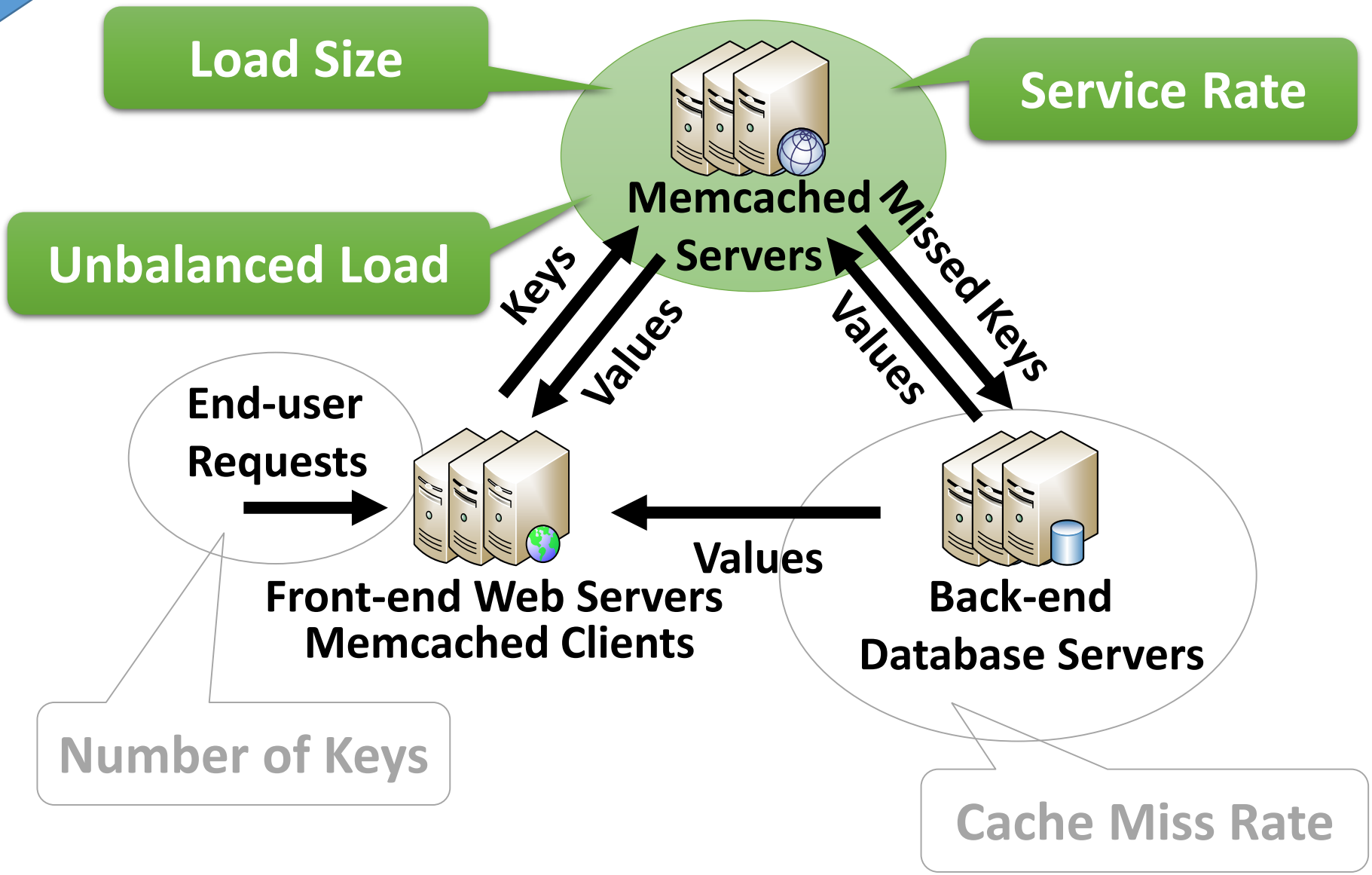
Back-end
Database Servers

Number of Keys

Cache Miss Ratio

Analysis

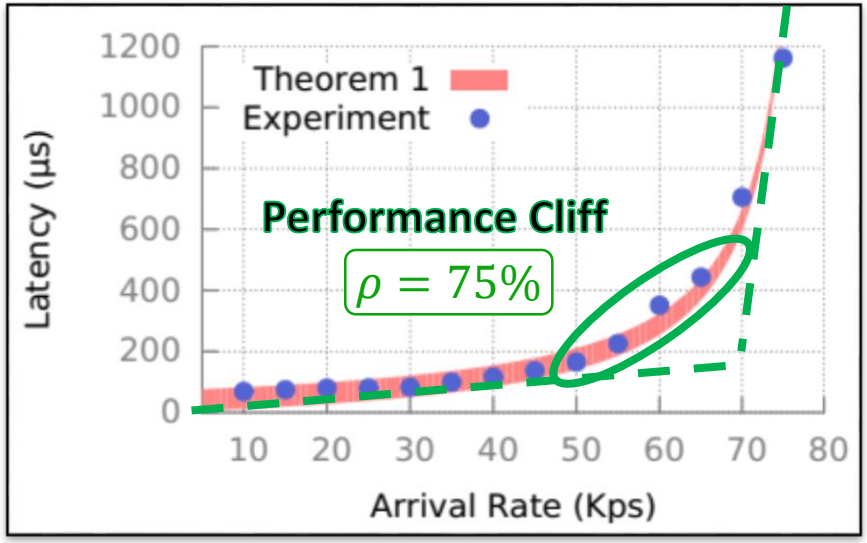
Factors on Latency

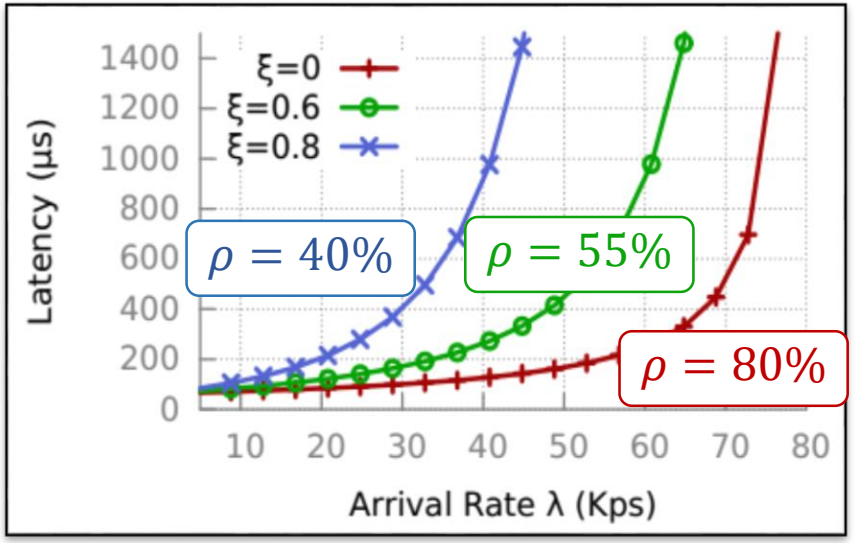


Analysis

Load Size

- Workload: Generalized Pareto Distribution
  - Arrival rate: $\lambda(\text{default} = 62.5Kbps)$
  - Burst degree: ξ $(\text{default} = 0.15)$
- Service rate: $\mu_s(\text{default} = 80Kps)$

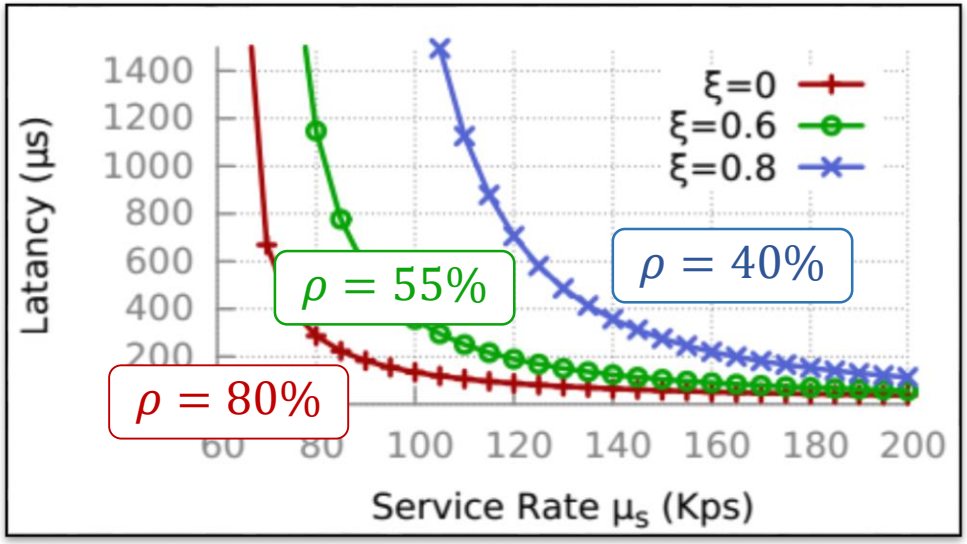




Analysis

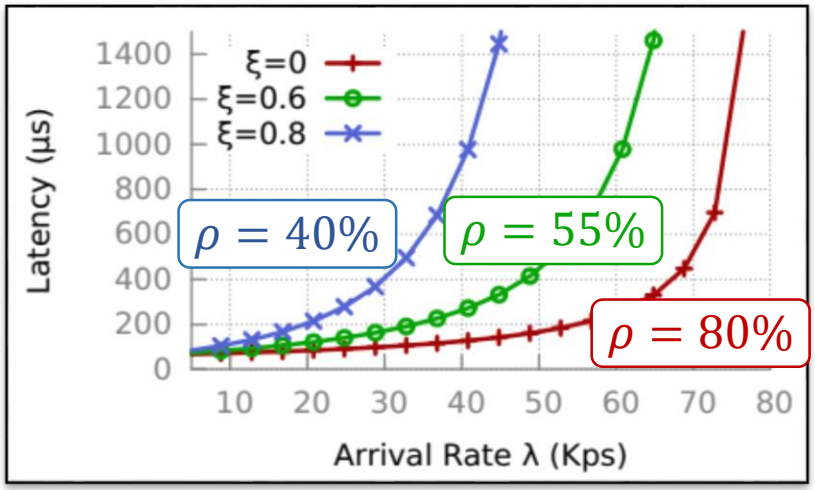
Service Rate

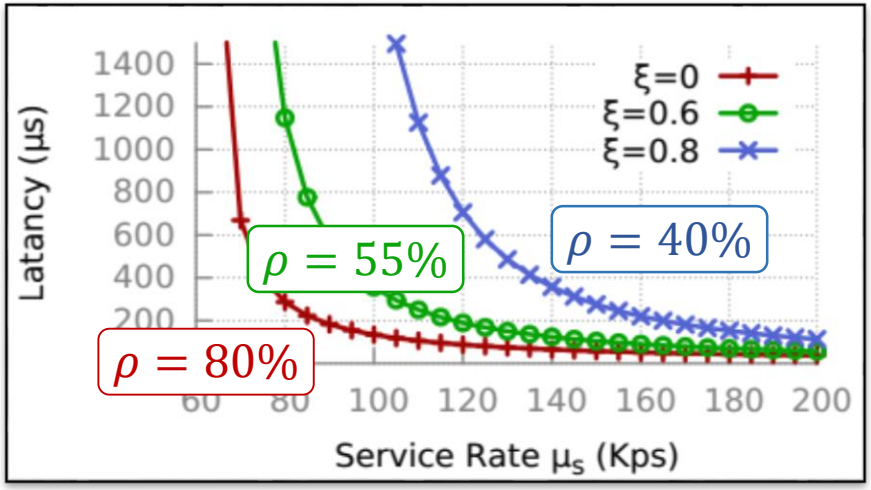
- Workload: Generalized Pareto Distribution
  - Arrival rate: $\lambda(\text{default} = 62.5Kbps)$
  - Burst degree: ξ $(\text{default} = 0.15)$
- Service rate: $\mu_s(\text{default} = 80Kps)$



Analysis

Load Size & Service Rate

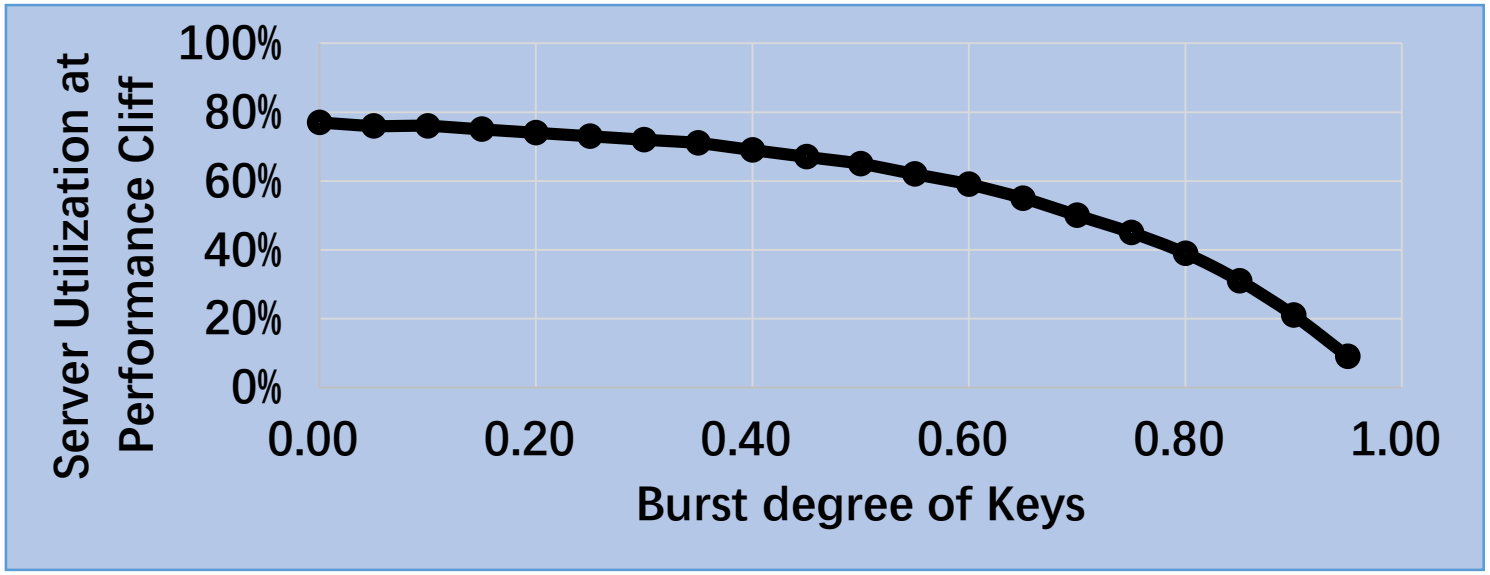




- *Latency reaches a **cliff point** when the Memcached **server utilization** gets to a specific value.*
- *This specific utilization is only determined by the **burst degree** of workload.*

Analysis

Server Utilization

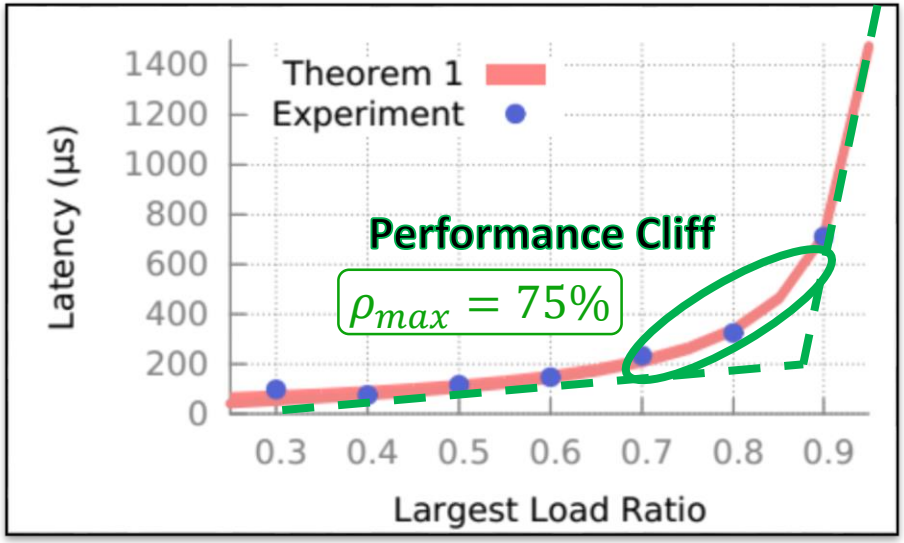


- *Latency reaches a **cliff point** when the Memcached **server utilization** gets to a specific value.*
- *This specific utilization is only determined by the **burst degree** of workload.*

Analysis

Unbalanced Load

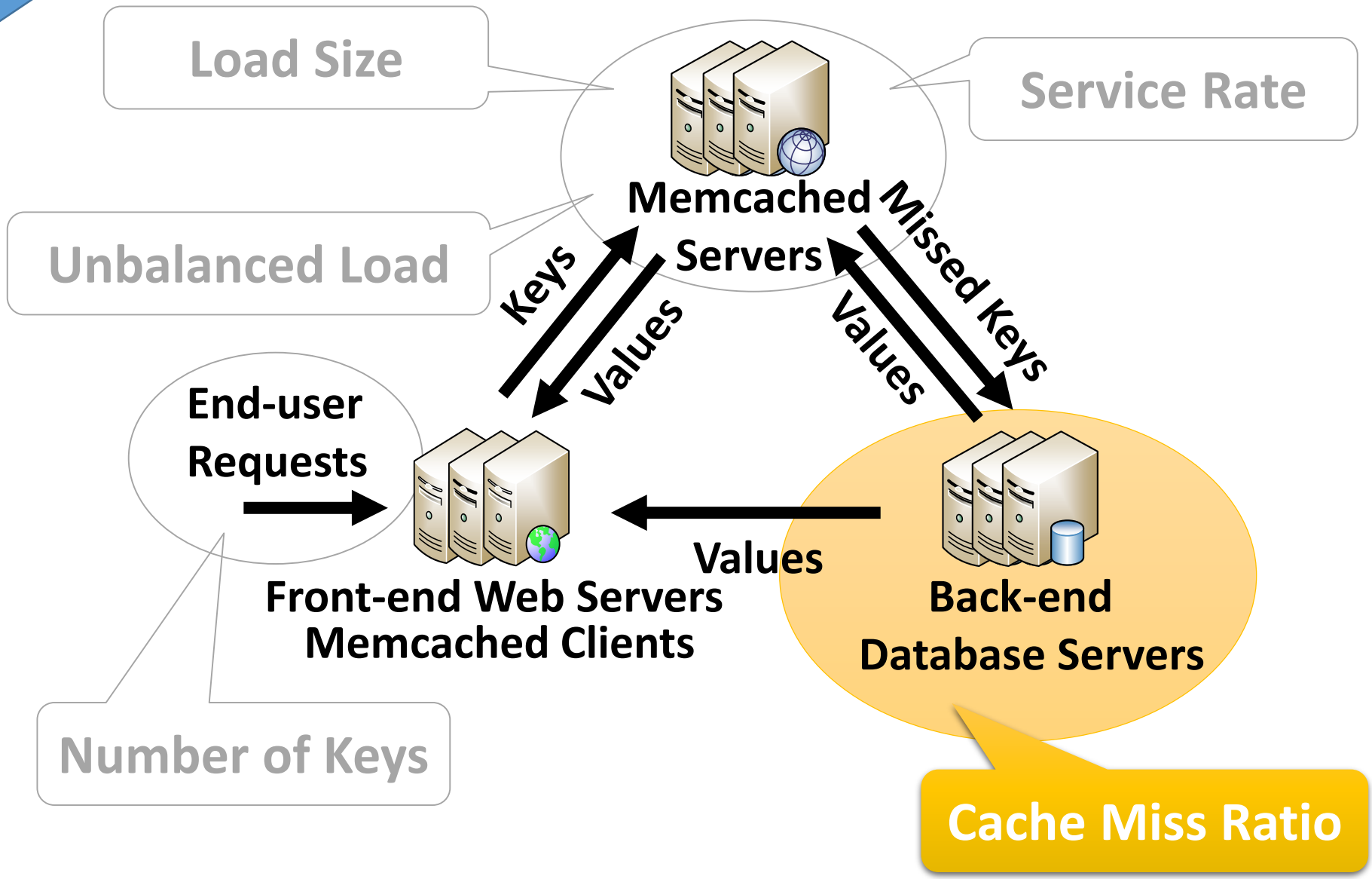
- 4 Memcached servers: Largest load ratio
  - Total load size: $\Lambda = 80Kbps$
  - Burst degree: $\xi = 0.15$
  - Service rate: $\mu_s = 80Kbps$



- Latency still reaches a **cliff point** when the **heaviest** server gets to the **specific utilization**.

Analysis

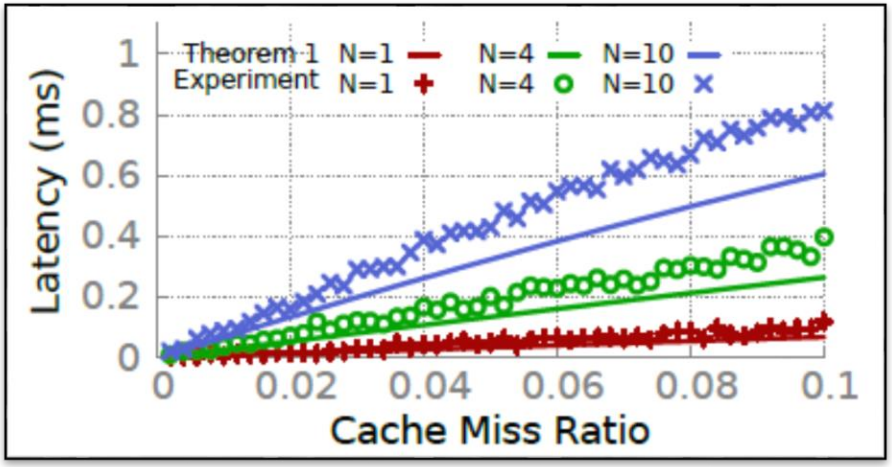
Factors on Latency

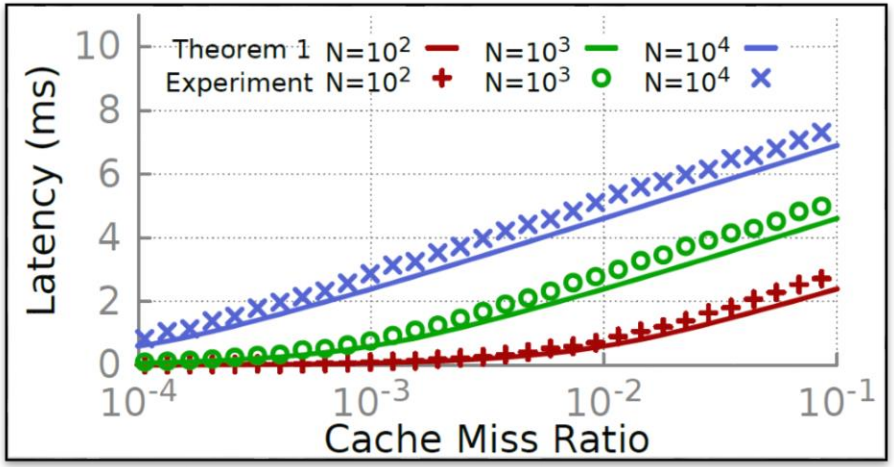


Analysis

Cache Miss Ratio

- Average cost at database: $\frac{1}{\mu_D} = 1ms$



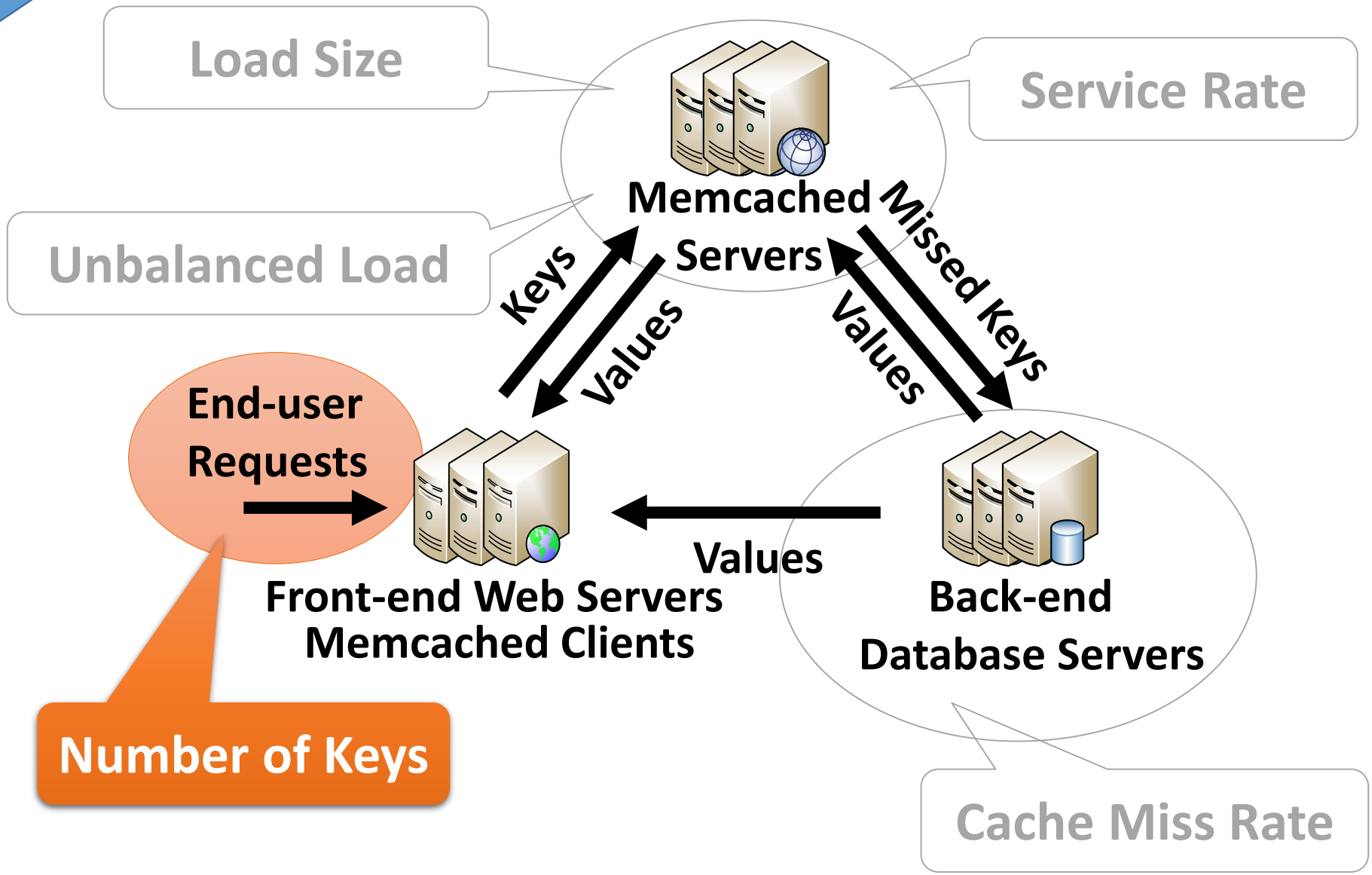


- $r = 0.1 \rightarrow 0.01$
  - $N = 1$ ➔ latency: 0.8ms → 0.08ms, ↓ 90%
  - $N = 10^4$ ➔ latency: 7.2ms → 4.3ms, ↓ 40%

- Latency grows **logarithmically** with the increase of the **cache miss ratio**.

Analysis

Factors on Latency

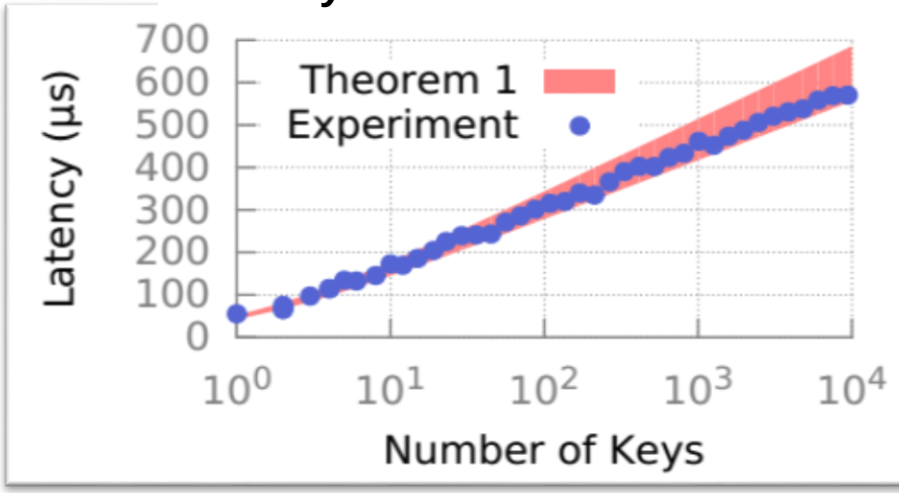


Analysis

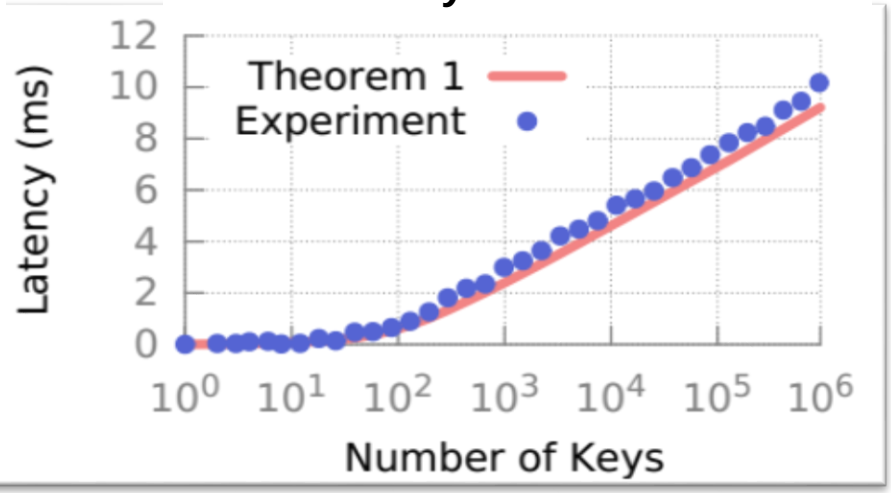
Number of Keys

- Workload burst degree $\xi = 0.15$
- Memcached server utilization $\rho = 0.75$
- Cache miss rate $r = 0.01$

Latency at Memcached Servers



Latency at Database



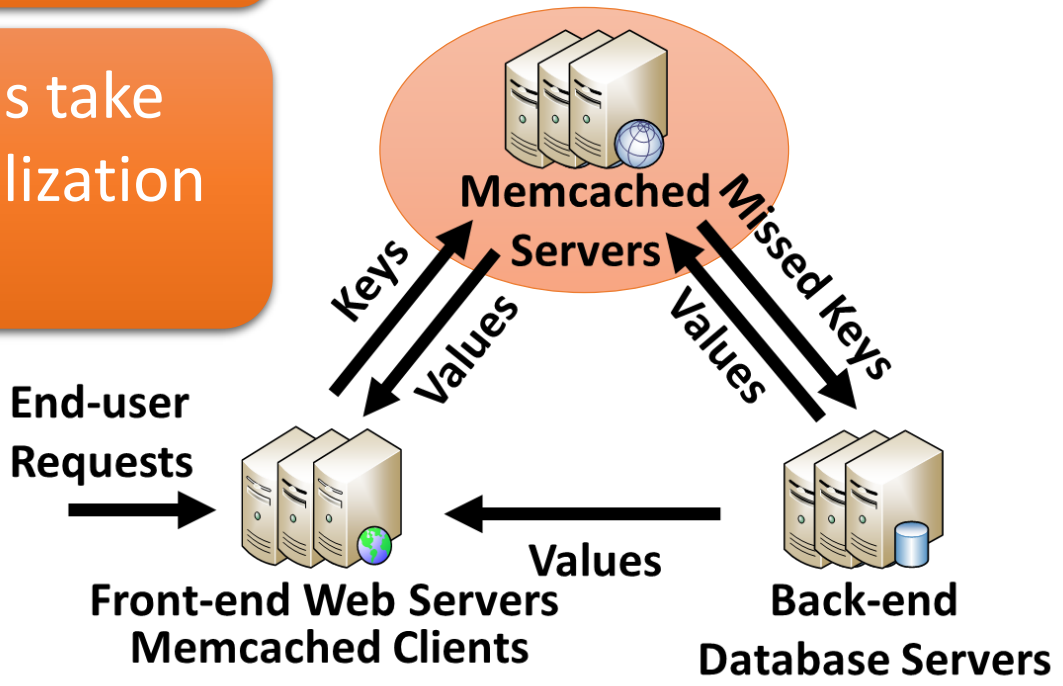
- Latency grows **logarithmically** with the increase of the **number of keys.**

Analysis

Recommendations

Server utilization should not exceed a specific value.

Load-balancing mechanisms take effect before the largest utilization extends the specific value.

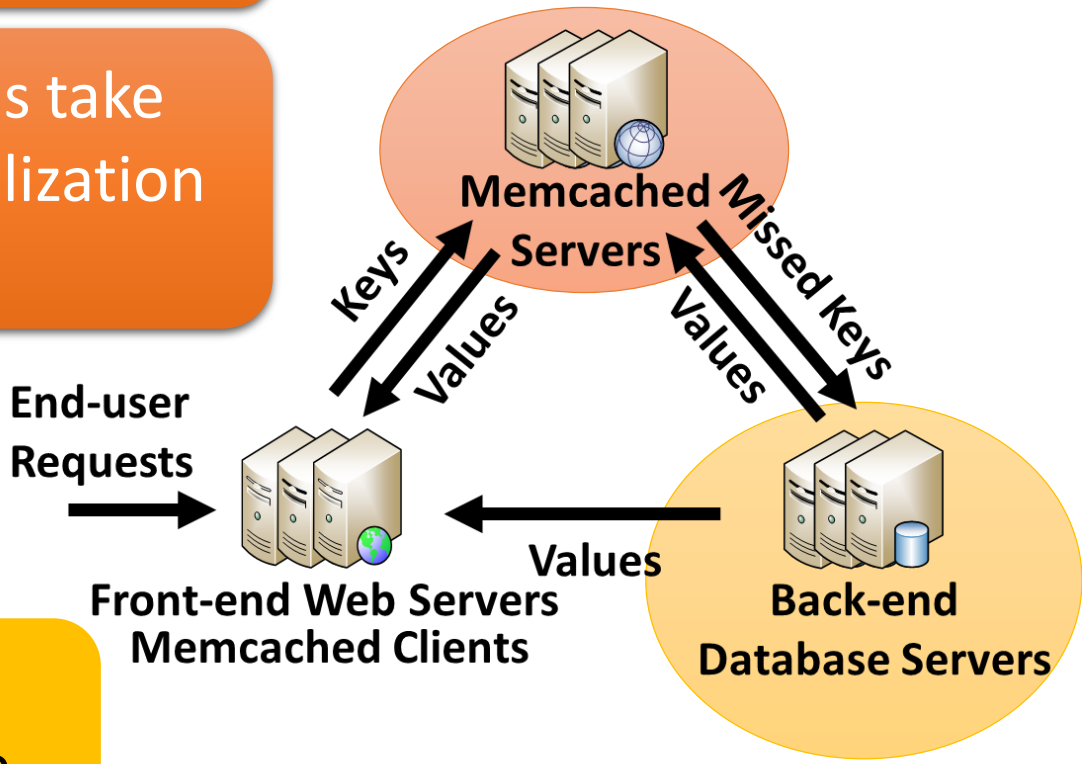


Analysis

Recommendations

Server utilization should not exceed a specific value.

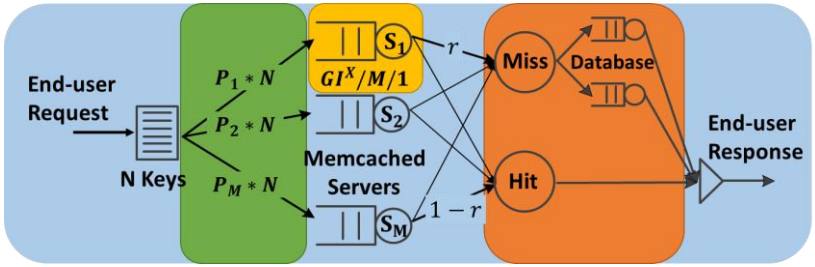
Load-balancing mechanisms take effect before the largest utilization extends the specific value.



Minimizing the number of keys instead of reducing the cache miss ratio.

Conclusion

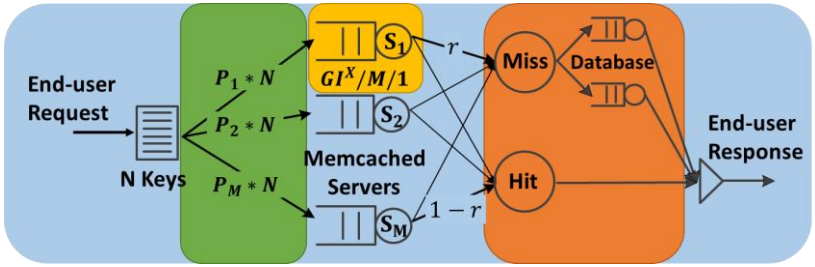
- Specific model for Memcached
  - Unbalanced load distribution
  - Burst and concurrent key arrivals
  - Database processing



- Latency estimation, validation & Analysis
  - Network latency is almost constant
  - Latency at Memcached servers is determined by server utilization
  - Latency at database is determined by the number of keys and cache miss ratio
- Recommendations
  - Server utilization should not exceed a specific value.
  - Load-balancing mechanisms take effect before the largest utilization extends the specific value.
  - Minimizing the number of keys instead of reducing the cache miss ratio.

Conclusion

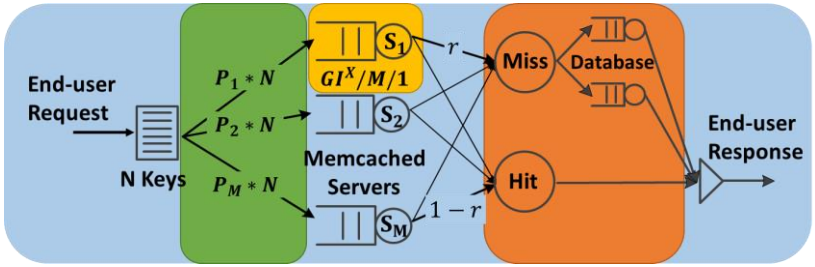
- Specific model for Memcached
  - Unbalanced load distribution
  - Burst and concurrent key arrivals
  - Database processing



- Latency estimation, validation & Quantitative analysis
  - Network latency is almost constant
  - Latency at Memcached servers is determined by server utilization
  - Latency at database is determined by the number of keys and cache miss ratio
- Recommendations
  - Server utilization should not exceed a specific value.
  - Load-balancing mechanisms take effect before the largest utilization extends the specific value.
  - Minimizing the number of keys instead of reducing the cache miss ratio.

Conclusion

- Specific model for Memcached
  - Unbalanced load distribution
  - Burst and concurrent key arrivals
  - Database processing



- Latency estimation, validation & Analysis
  - Network latency is almost constant
  - Latency at Memcached servers is determined by server utilization
  - Latency at database is determined by the number of keys and cache miss ratio
- Recommendations
  - Server utilization should not exceed a specific value.
  - Load-balancing mechanisms take effect before the largest utilization extends the specific value.
  - Minimizing the number of keys instead of reducing the cache miss ratio.

Thanks!