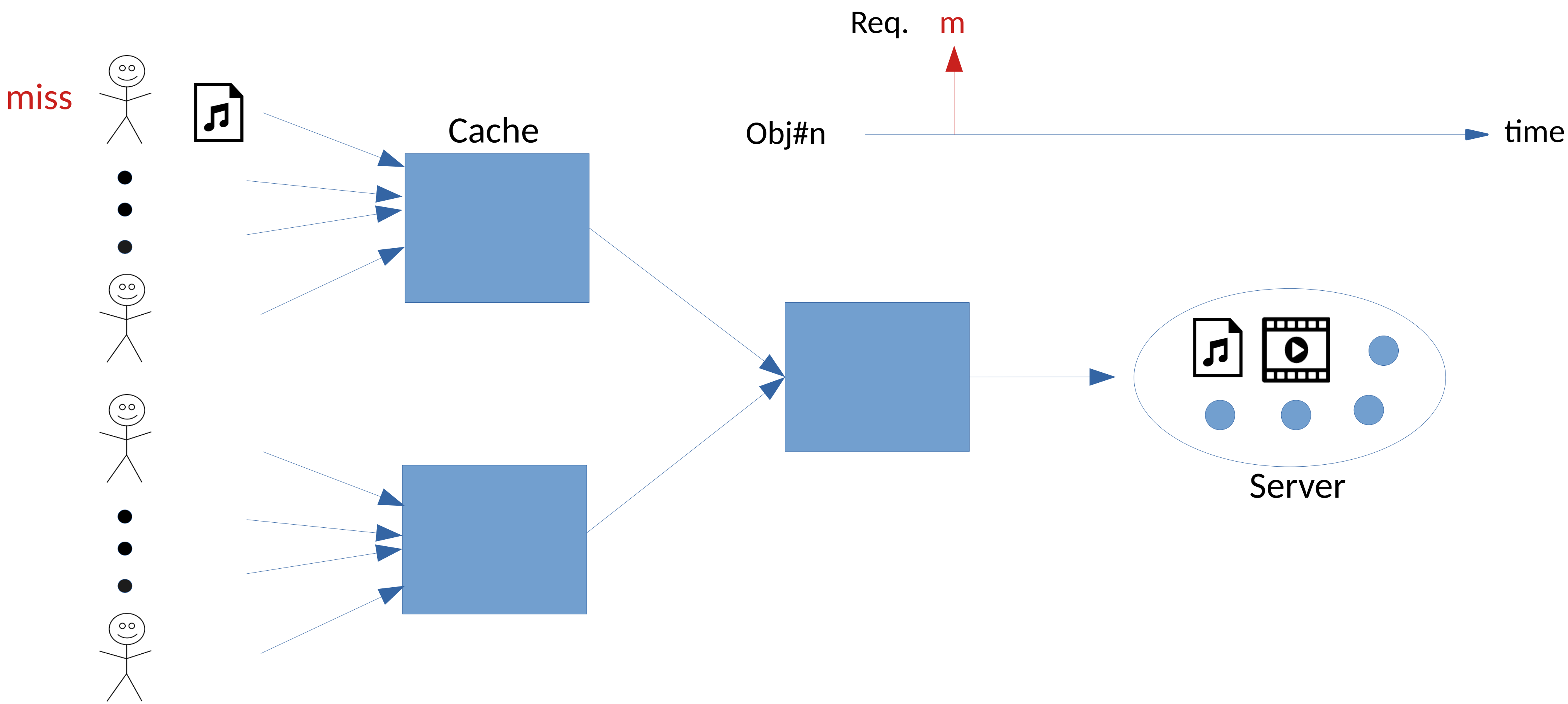
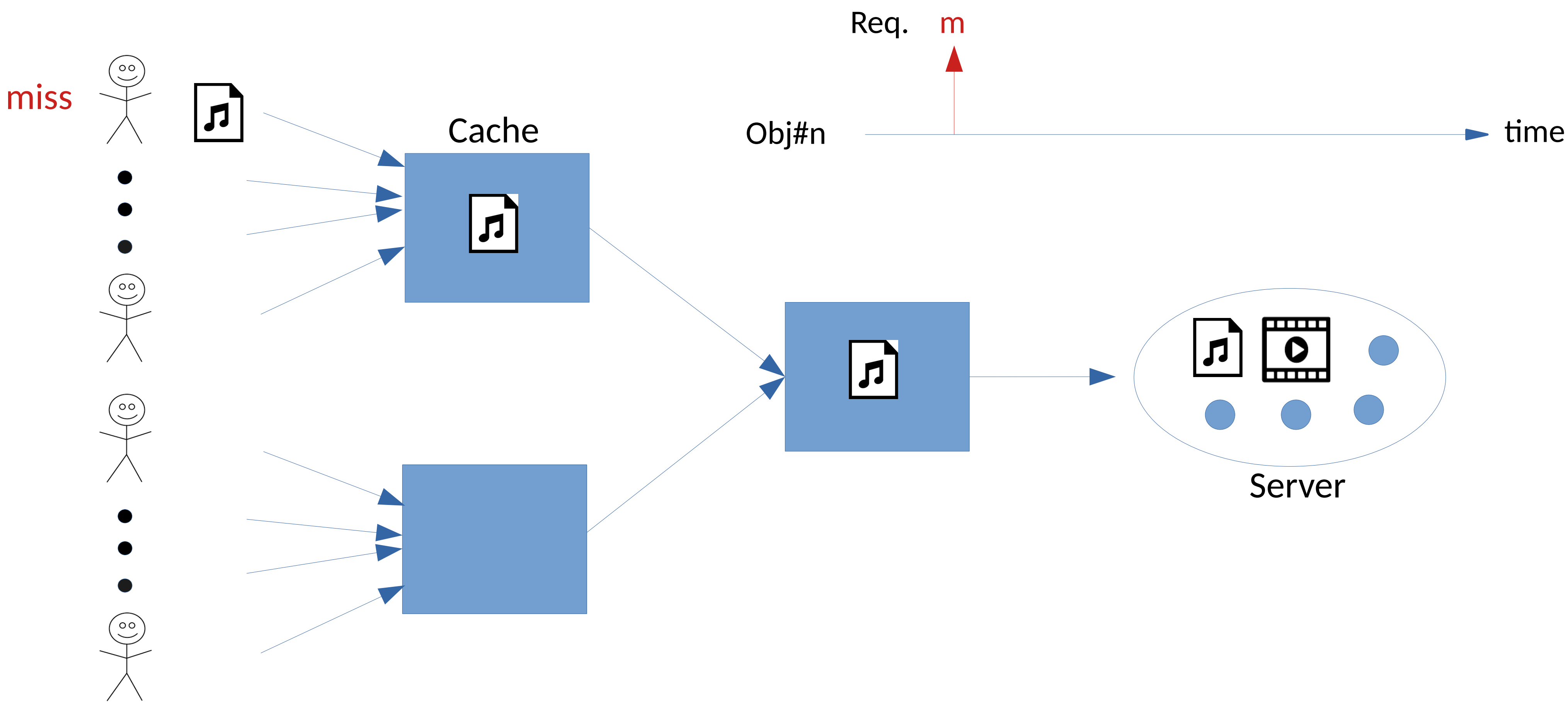
Response Times in Time-To-Live Caching Hierarchies under Random Network Delays

Karim Elsayed
Joint work with Amr Rizk

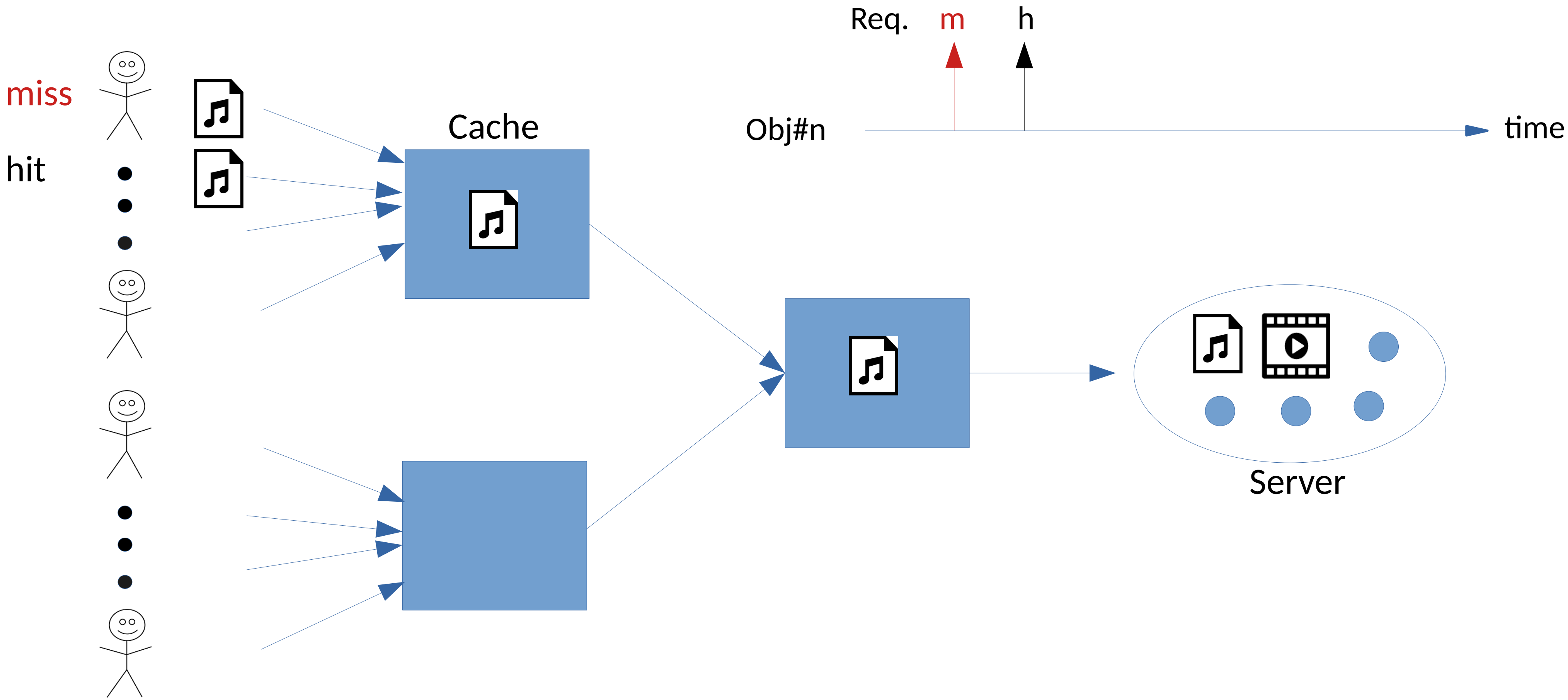
Motivation



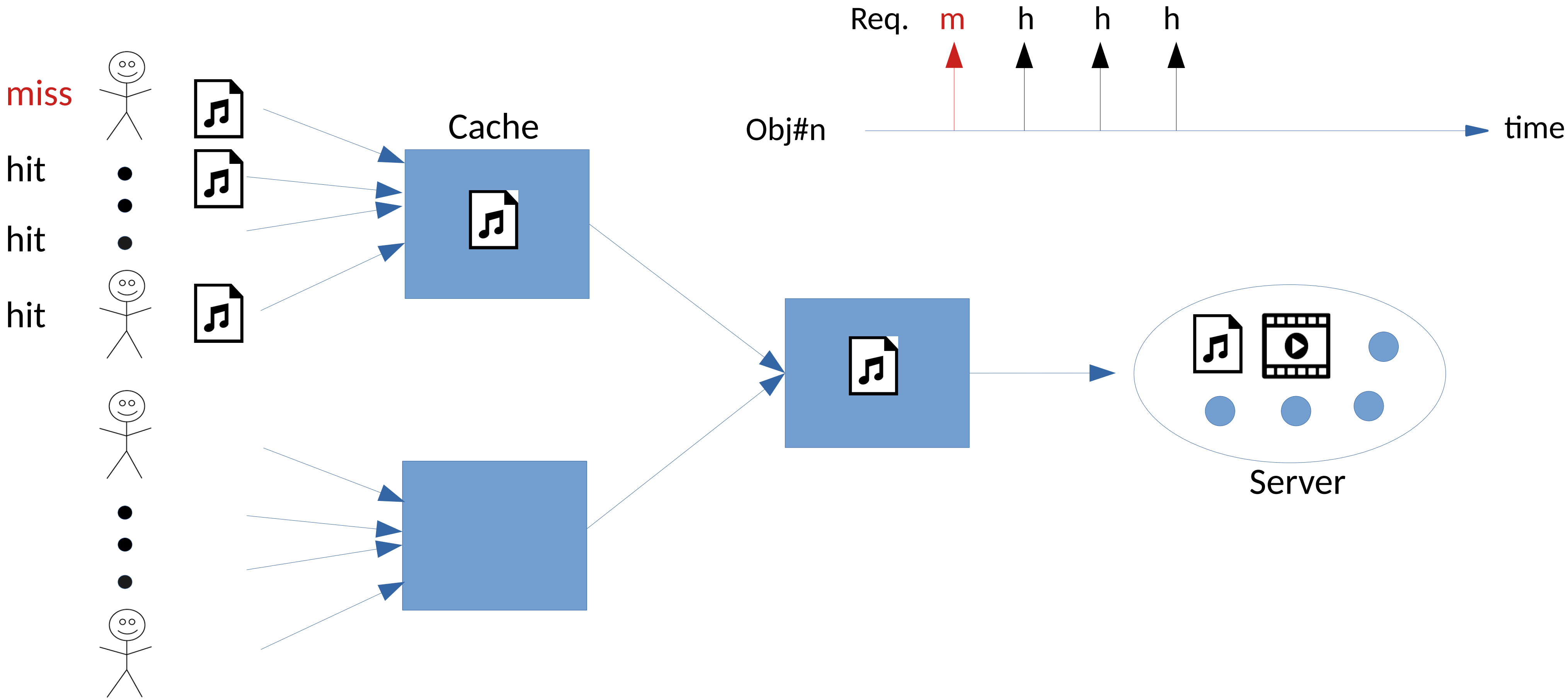
Motivation



Motivation

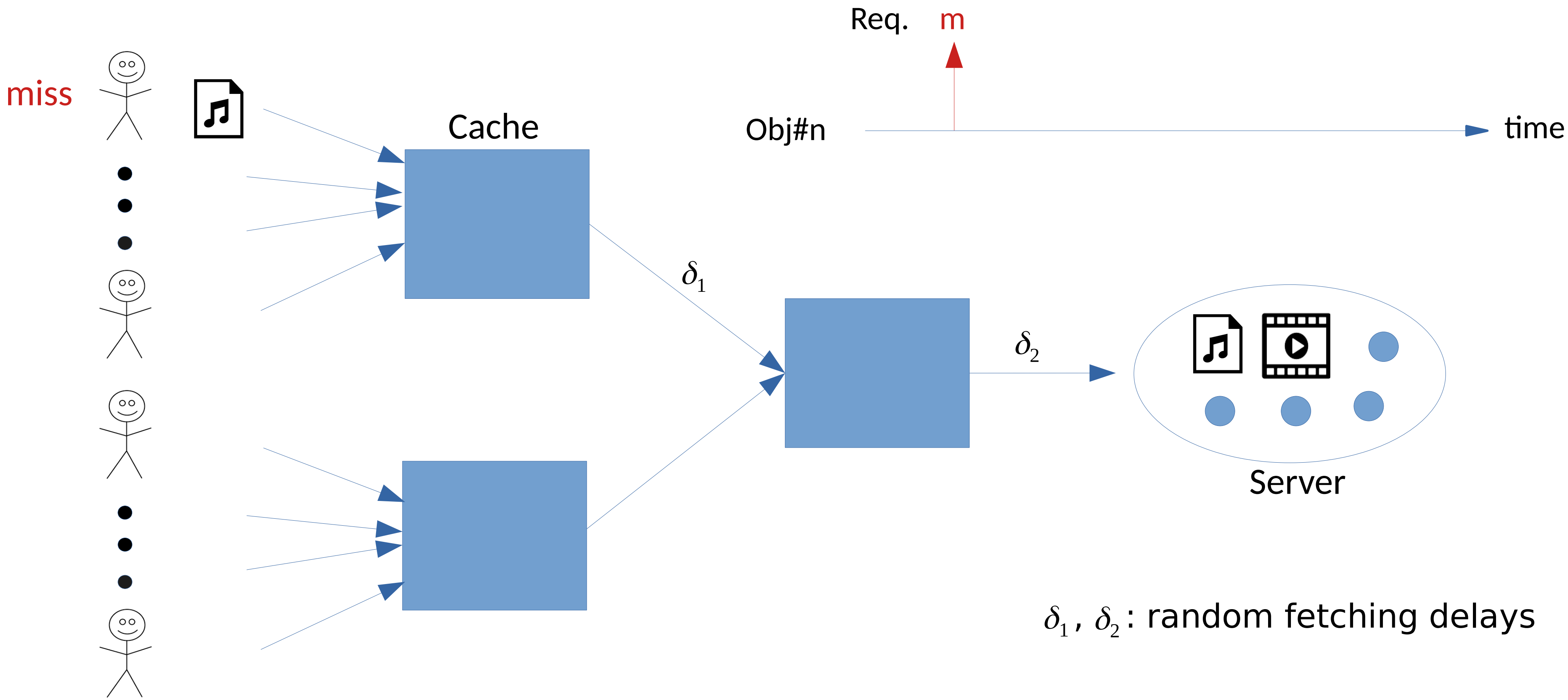


Motivation



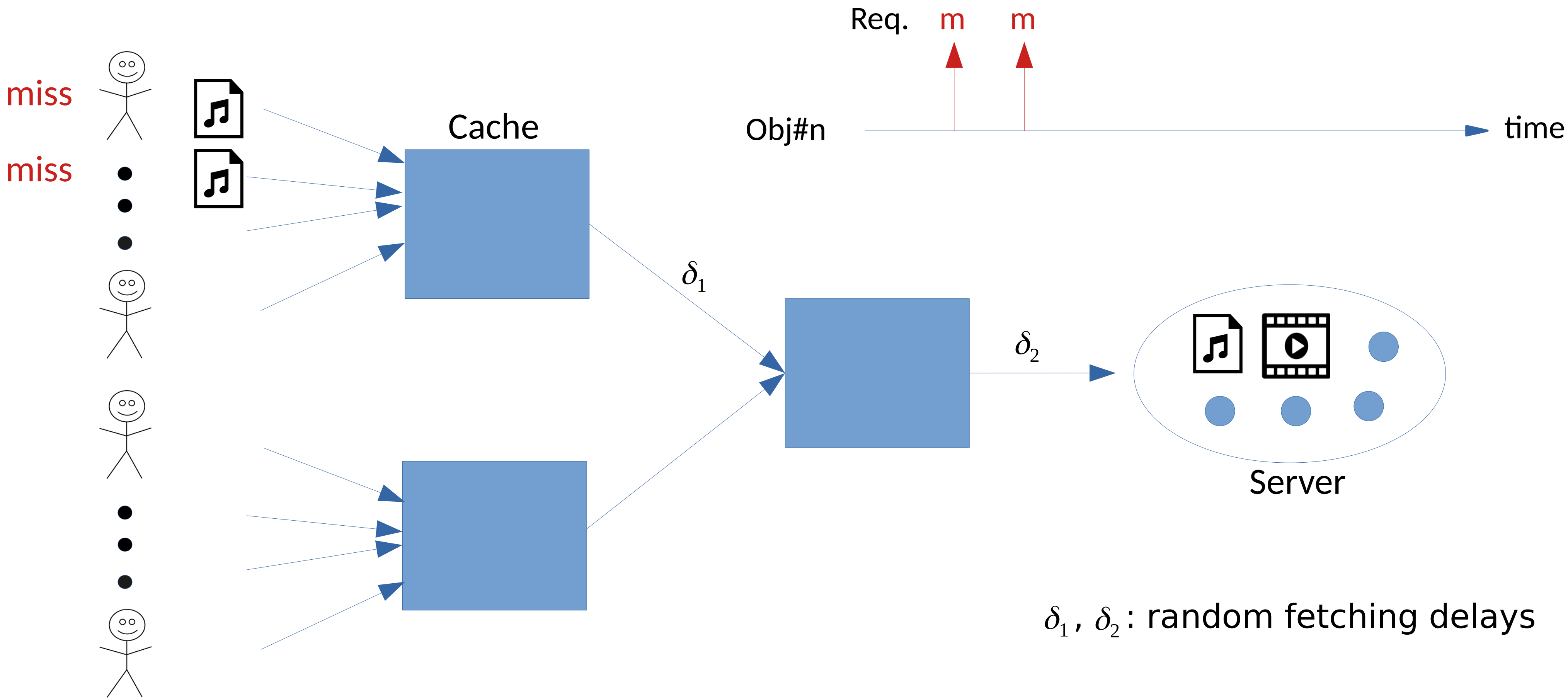
Motivation

- Object admission to the cache is not instantaneous



Motivation

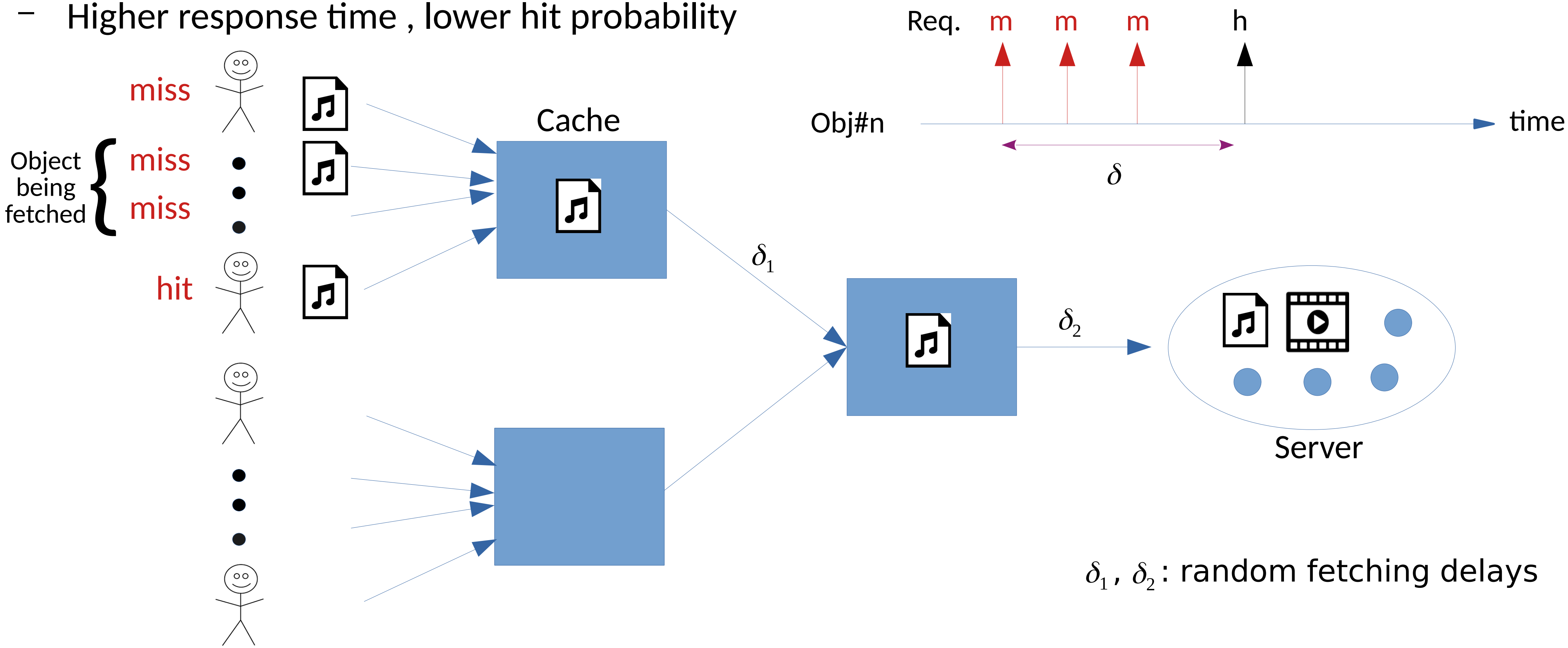
- Aggregate requests during random fetching delays impact the performance



δ_1 , δ_2 : random fetching delays

Motivation

- Aggregate requests during random fetching delays impact the performance
  - Higher response time , lower hit probability



Contributions

- Extending an **exact** model of the caching hierarchy under **random network delays**

Contributions

- Extending an **exact** model of the caching hierarchy under **random network delays**
  - Calculating the **exact** mean response time for cache hierarchies → Importance?

Contributions

- Extending an **exact** model of the caching hierarchy under **random network delays**
  - Calculating the **exact** mean response time for cache hierarchies → Importance?
- Standing on the shoulders of
  - On the Impact of network delays on Time-to-Live caching [Elsayed]

[Elsayed] K. Elsayed, and A. Rizk, "On the Impact of Network Delays on Time-to-Live Caching," *ArXiv abs/2201.1157, 2022.*

Contributions

- Extending an **exact** model of the caching hierarchy under **random network delays**
  - Calculating the **exact** mean response time for cache hierarchies → Importance?
- Standing on the shoulders of
  - On the Impact of network delays on Time-to-Live caching [Elsayed]

[Elsayed] K. Elsayed, and A. Rizk, “On the Impact of Network Delays on Time-to-Live Caching,” *ArXiv abs/2201.1157, 2022.*

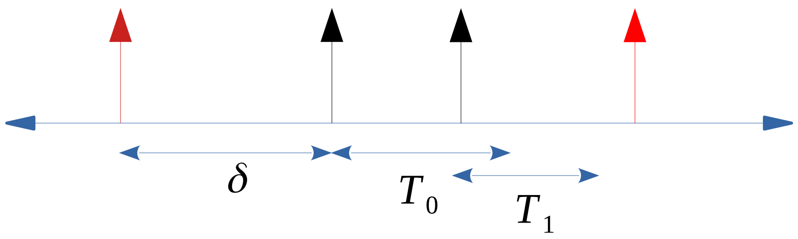
Contributions

- Extending an **exact** model of the caching hierarchy under **random network delays**
  - Calculating the **exact** mean response time for cache hierarchies → Importance?
- Standing on the shoulders of
  - On the Impact of network delays on Time-to-Live caching [Elsayed]
  - Exact TTL Cache Hierarchy model under zero delay [Berger, Ciucu, 2014]

[Elsayed] K. Elsayed, and A. Rizk, “On the Impact of Network Delays on Time-to-Live Caching,” *ArXiv abs/2201.1157, 2022.*
[Berger] D. S. Berger et al. “Exact Analysis of TTL Cache Networks,” *Performance Evaluation, vol. 79, pp. 2 – 23, 2014.*

TTL Cache Model

- Admission → object is assigned a time to live (TTL)
- Eviction → TTL expiration
- Hit → TTL gets renewed



T_0, T_1 TTL

- Objects are decoupled in the cache

Cache Model

Markov arrival process

- A model for Markovian point processes

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:
  - $(D_0, D_1) \leftrightarrow$ (Hidden, Active) transition matrices
  - D_1 includes the transitions contributing to the **counting process**, where we count **misses**

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:
  - $(D_0, D_1) \leftrightarrow$ (Hidden, Active) transition matrices
  - D_1 includes the transitions contributing to the **counting process**, where we count **misses**
- Model assumptions:

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:
  - $(D_0, D_1) \leftrightarrow$ (Hidden, Active) transition matrices
  - D_1 includes the transitions contributing to the **counting process**, where we count **misses**
- Model assumptions:
  - Inter-request times, TTLs and delays are I.I.D, generally PH distributed

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:
  - $(D_0, D_1) \leftrightarrow$ (Hidden, Active) transition matrices
  - D_1 includes the transitions contributing to the **counting process**, where we count **misses**
- Model assumptions:
  - Inter-request times, TTLs and delays are I.I.D, generally PH distributed
  - TTLs can be random or deterministic

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:
  - $(D_0, D_1) \leftrightarrow$ (Hidden, Active) transition matrices
  - D_1 includes the transitions contributing to the **counting process**, where we count **misses**
- Model assumptions:
  - Inter-request times, TTLs and delays are I.I.D, generally PH distributed
  - TTLs can be random or deterministic
  - Tree-like cache hierarchy

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:
  - $(D_0, D_1) \leftrightarrow$ (Hidden, Active) transition matrices
  - D_1 includes the transitions contributing to the **counting process**, where we count **misses**
- Model assumptions:
  - Inter-request times, TTLs and delays are I.I.D, generally PH distributed
  - TTLs can be random or deterministic
  - Tree-like cache hierarchy
- Notation:

Cache Model

Markov arrival process

- A model for Markovian point processes
- Definition:
  - $(D_0, D_1) \leftrightarrow$ (Hidden, Active) transition matrices
  - D_1 includes the transitions contributing to the **counting process**, where we count **misses**
- Model assumptions:
  - Inter-request times, TTLs and delays are I.I.D, generally PH distributed
  - TTLs can be random or deterministic
  - Tree-like cache hierarchy
- Notation: M/ M/ M

Inter-request time

TTL

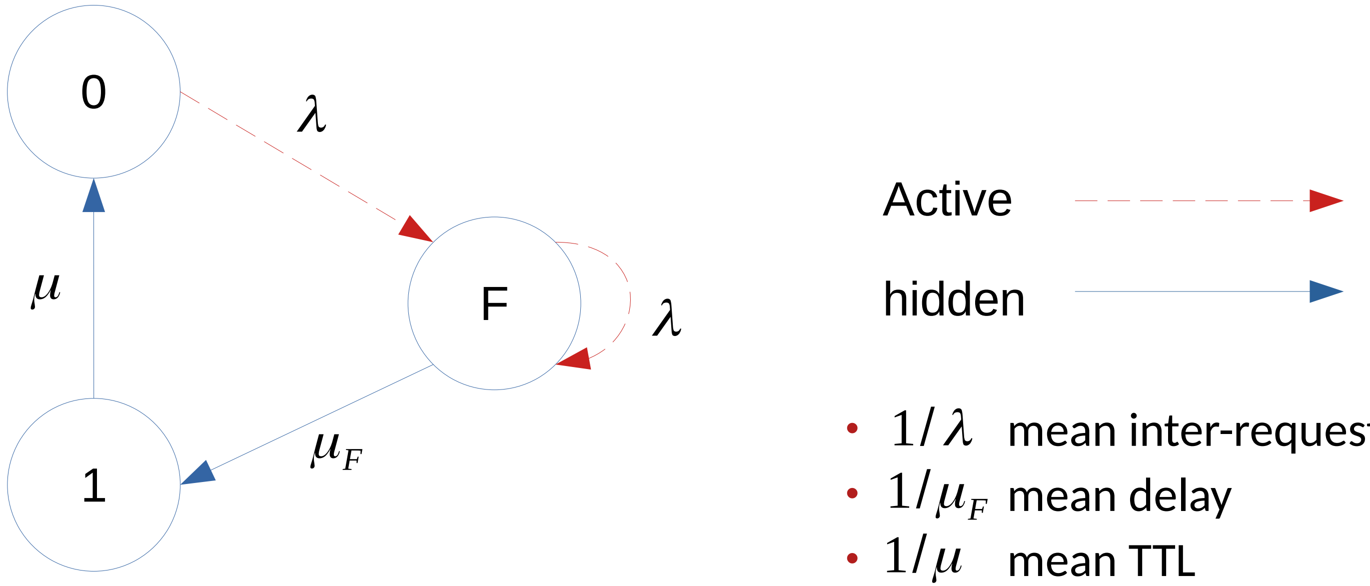
Delay

<u>Our work:</u>

M: exponentially distributed, PH: phase type, E: Erlang

Single M/M/M cache

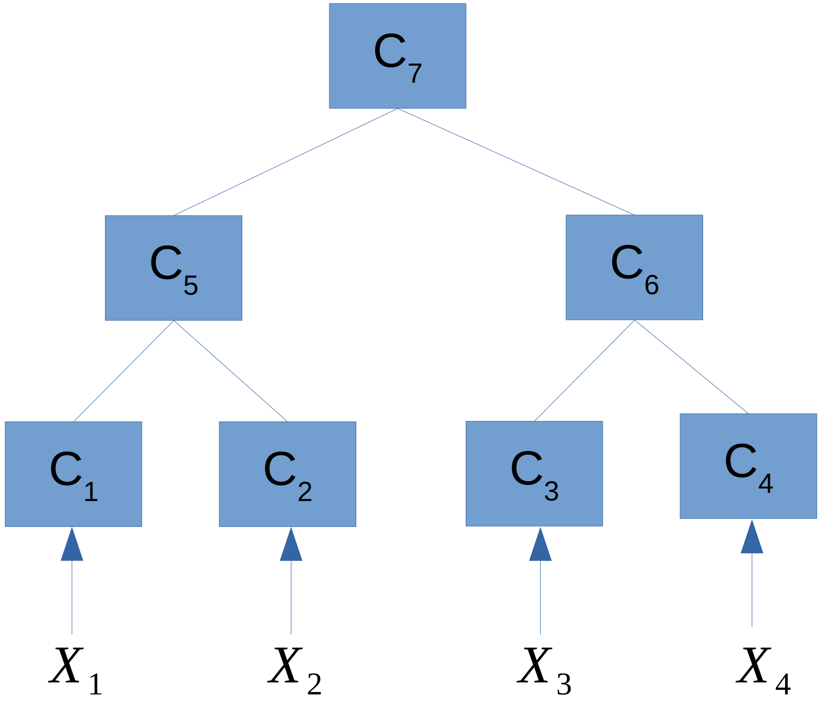
- One object in/out of the cache is modelled using MAPs
- MAP has 3 states:
  - State “1”: Object in the cache
  - State “0”: Object out of the cache
  - State “F”: Object being fetched



- $1/\lambda$ mean inter-request time
- $1/\mu_F$ mean delay
- $1/\mu$ mean TTL

Cache Hierarchy MAP

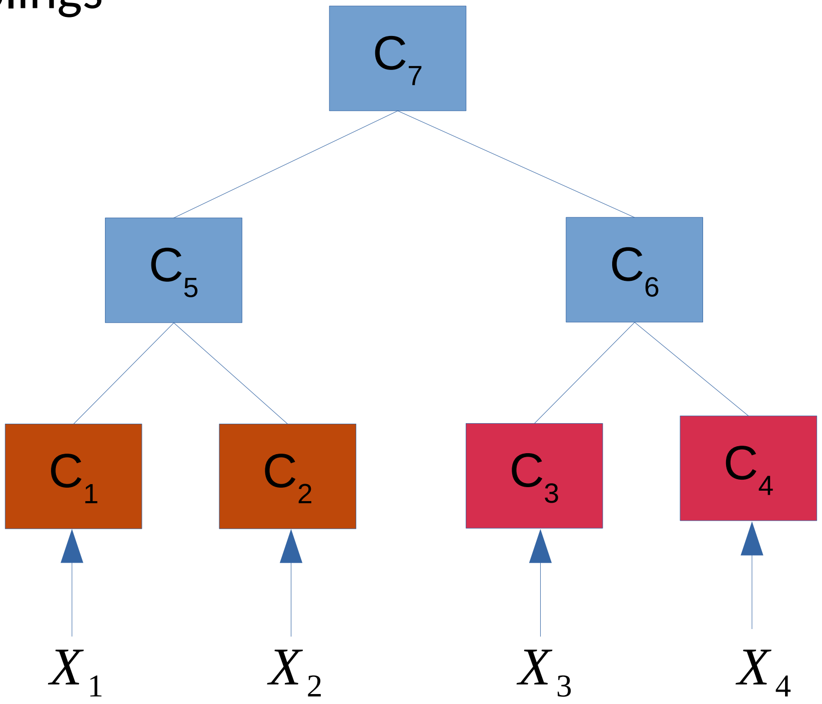
- Goal: model the cache hierarchy using a total MAP
- Approach: **Exact recursive** superposition of single cache MAPs



X : Request process

Cache Hierarchy MAP

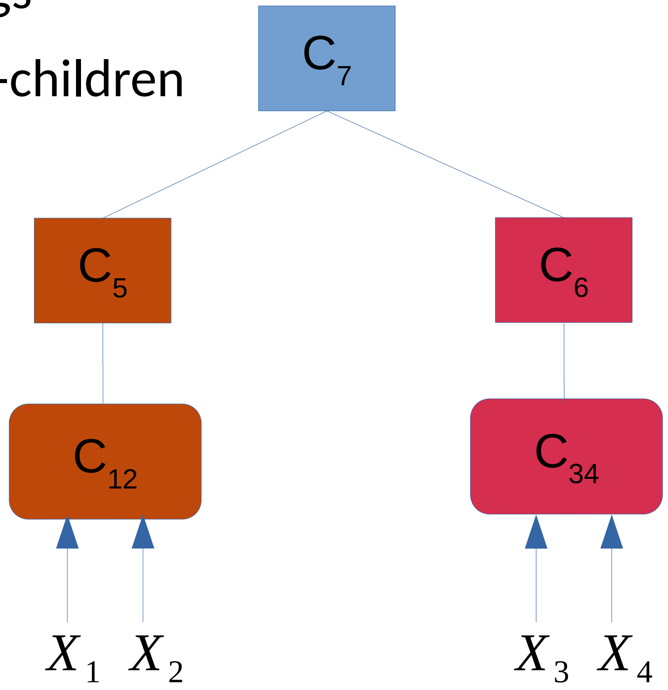
- Goal: model the cache hierarchy using a total MAP
- Approach: **Exact recursive** superposition of single cache MAPs
- From leaf caches:
  - Level superposition of siblings



X : Request process

Cache Hierarchy MAP

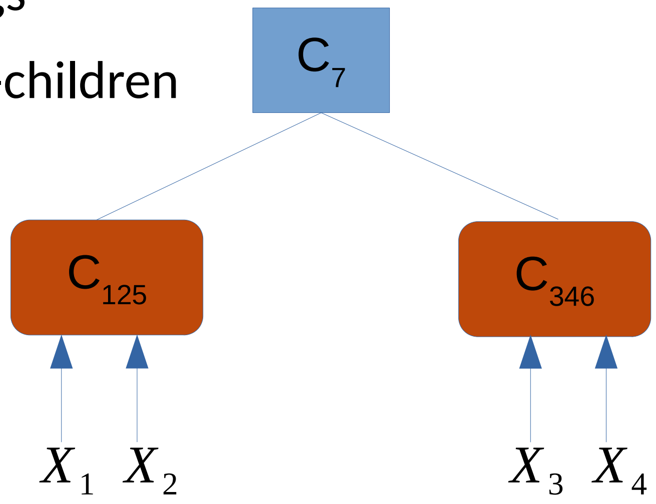
- Goal: model the cache hierarchy using a total MAP
- Approach: **Exact recursive** superposition of single cache MAPs
- From leaf caches:
  - Level superposition of siblings
  - Line superposition of parent-children



X : Request process

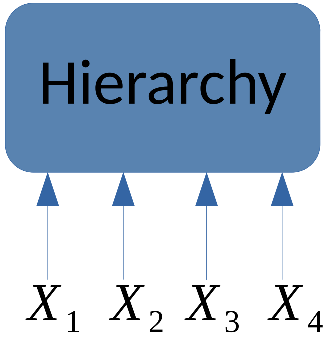
Cache Hierarchy MAP

- Goal: model the cache hierarchy using a total MAP
- Approach: **Exact recursive** superposition of single cache MAPs
- From leaf caches:
  - Level superposition of siblings
  - Line superposition of parent-children



Cache Hierarchy MAP

- Goal: model the cache hierarchy using a total MAP
- Approach: **Exact recursive** superposition of single cache MAPs
- From leaf caches:
  - Level superposition of siblings
  - Line superposition of parent-children



Superposition

- Construction of a system MAP from individual MAPs

Superposition

- Construction of a system MAP from individual MAPs
- Based on the Kronecker sum of individual MAPs

$$M = M_1 \oplus M_2$$

$$(D_0, D_1) = (D_0^{(1)}, D_1^{(1)}) \oplus (D_0^{(2)}, D_1^{(2)})$$

Superposition

UNIVERSITÄT DUISBURG ESSEN

Offen im Denken

- Construction of a system MAP from individual MAPs
- Based on the Kronecker sum of individual MAPs

$$M = M_1 \oplus M_2$$

$$(D_0, D_1) = (D_0^{(1)}, D_1^{(1)}) \oplus (D_0^{(2)}, D_1^{(2)})$$

- All the combination of states with the corresponding transitions.

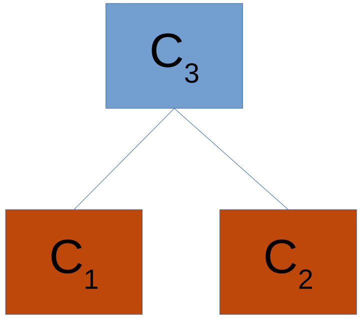
Superposition

- Construction of a system MAP from individual MAPs
- Based on the Kronecker sum of individual MAPs

$$M = M_1 \oplus M_2$$

$$(D_0, D_1) = (D_0^{(1)}, D_1^{(1)}) \oplus (D_0^{(2)}, D_1^{(2)})$$

- All the combination of states with the corresponding transitions.



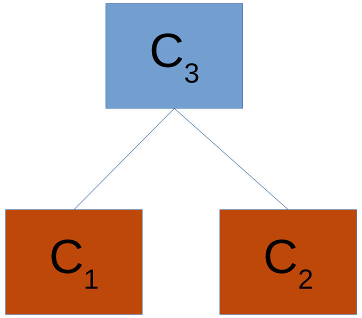
Superposition

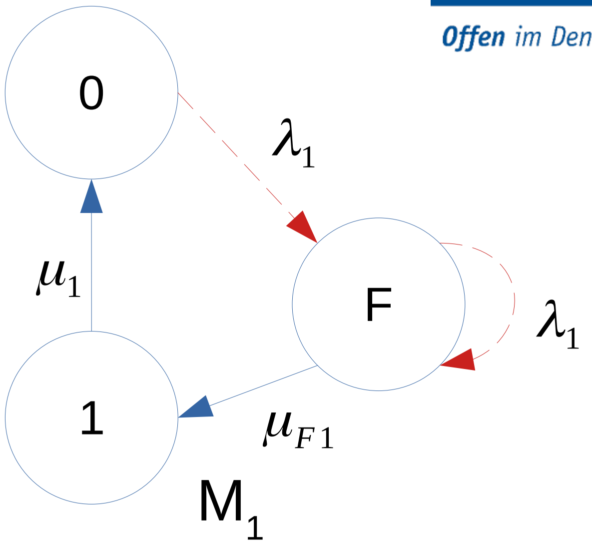
- Construction of a system MAP from individual MAPs
- Based on the Kronecker sum of individual MAPs

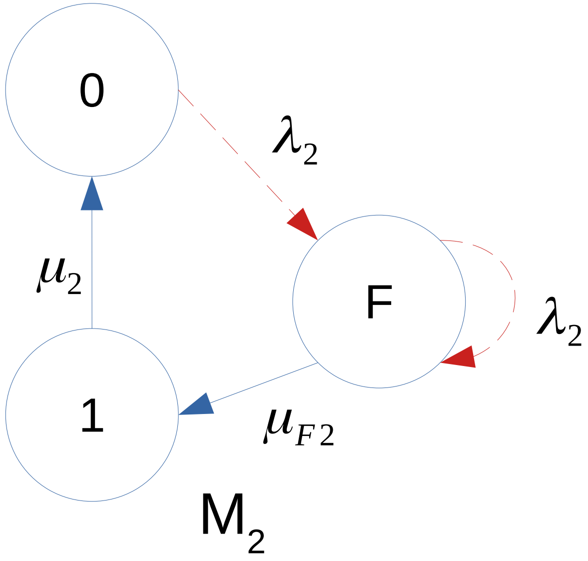
$$M = M_1 \oplus M_2$$

$$(D_0, D_1) = (D_0^{(1)}, D_1^{(1)}) \oplus (D_0^{(2)}, D_1^{(2)})$$

- All the combination of states with the corresponding transitions.





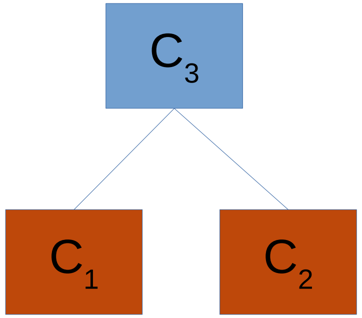


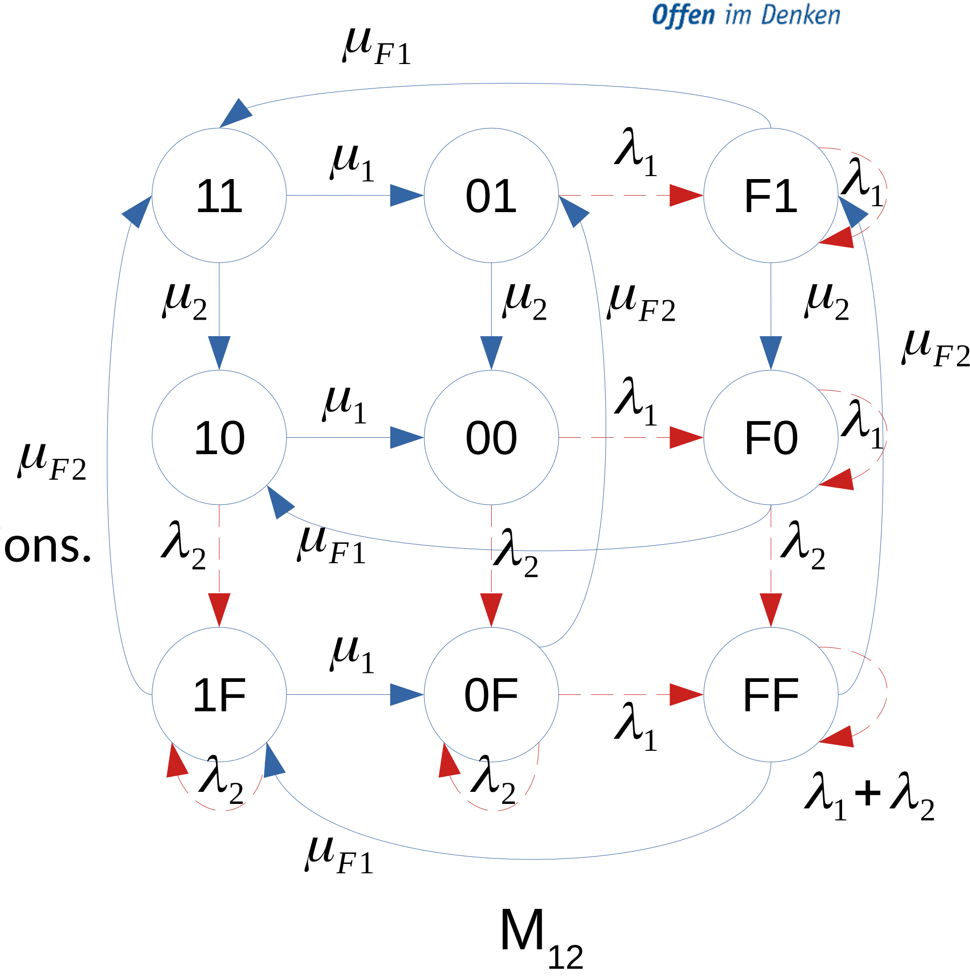
Superposition

- Construction of a system MAP from individual MAPs
- Based on the Kronecker sum of individual MAPs

$$M = M_1 \oplus M_2$$
$$(D_0, D_1) = (D_0^{(1)}, D_1^{(1)}) \oplus (D_0^{(2)}, D_1^{(2)})$$

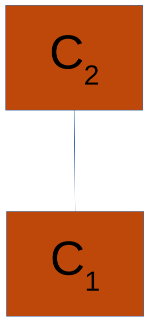
- All the combination of states with the corresponding transitions.
- **Independent** caches → Level superposition





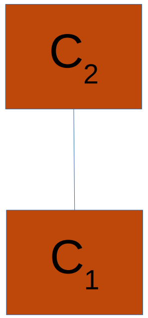
Superposition

- Line superposition → Dependent caches



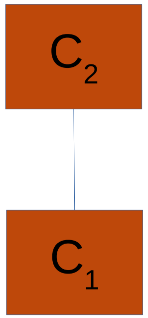
Superposition

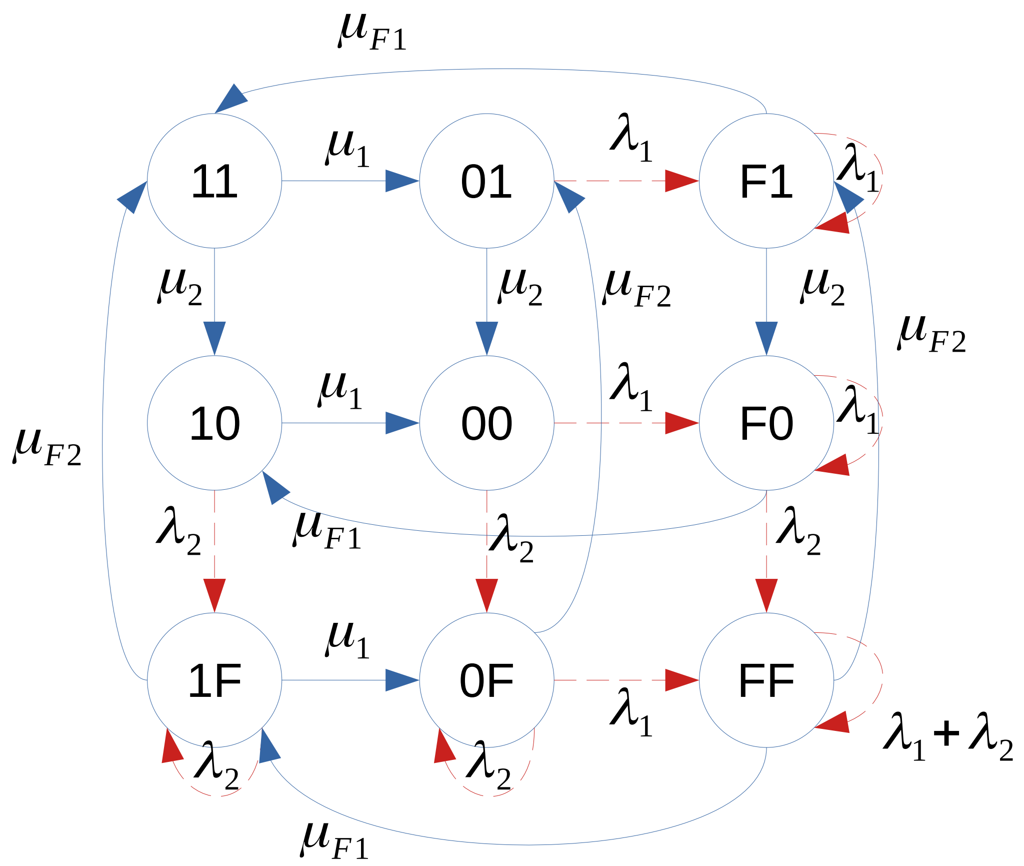
- Line superposition → Dependent caches
- Approach?



Superposition

- Line superposition → Dependent caches
- Approach?
  - Kronecker sum

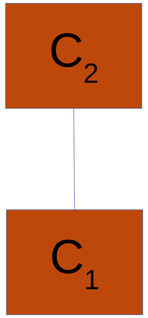


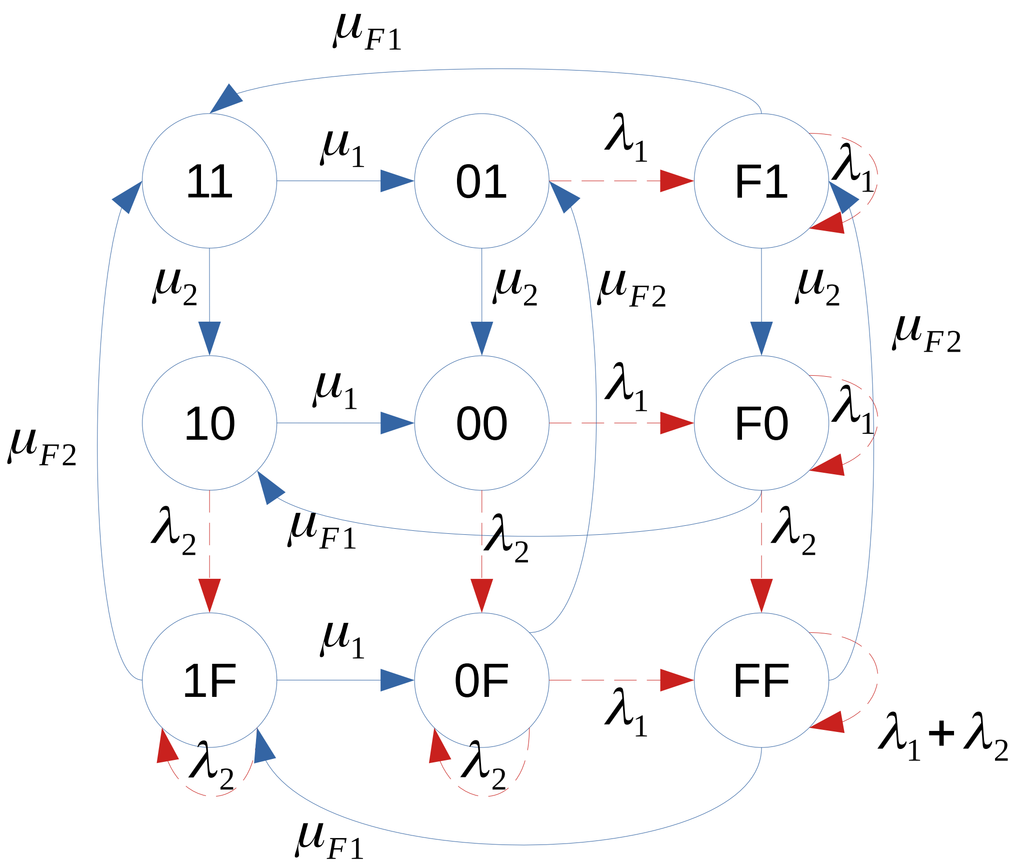


M_{12}

Superposition

- Line superposition → Dependent caches
- Approach?
  - Kronecker sum → problems?

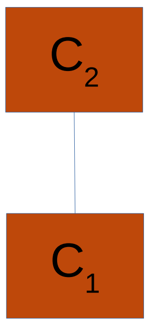


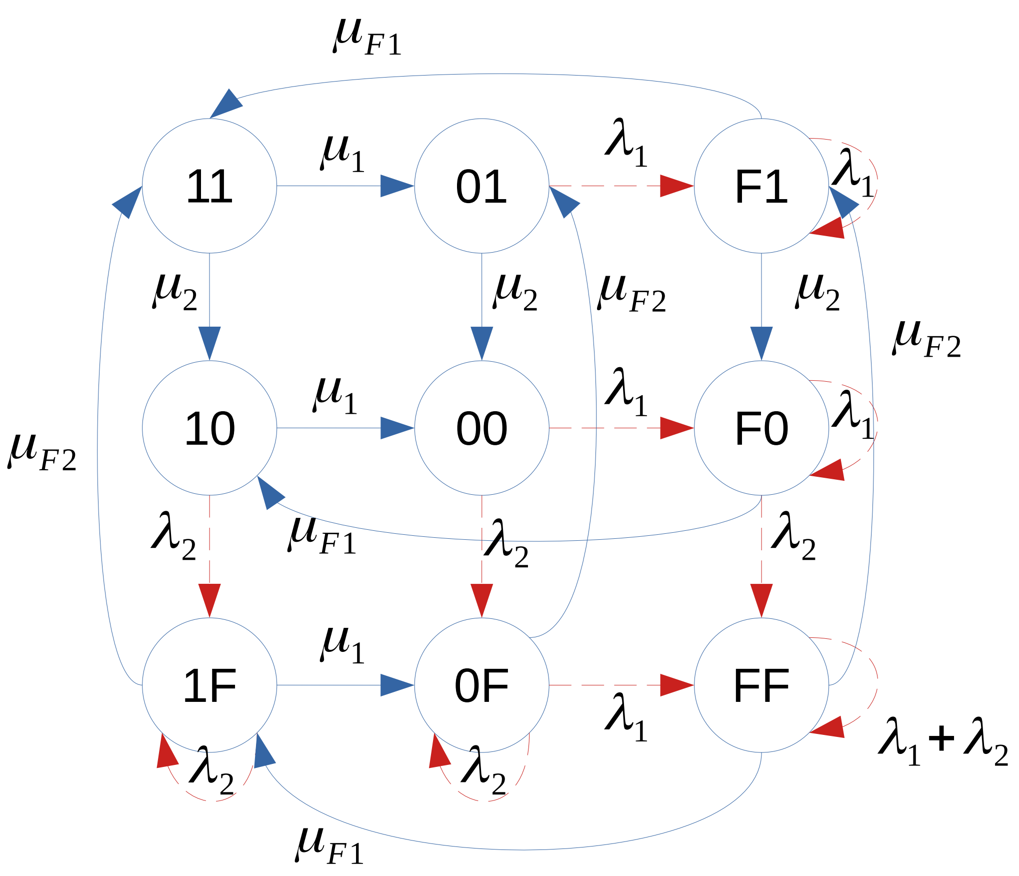


M_{12}

Superposition

- Line superposition → Dependent caches
- Approach?
  - Kronecker sum → problems?
  - e.g., “1F”→ Parent fetching while object in child cache





M_{12}

Complexity

- Model complexity
  - Number of states of the final MAP grows **exponentially** with the number of caches in the tree.

- Approach to reduce model complexity (while still exact) [Elsayed]
  - Leverage the symmetric structure within the tree.
  - Lumping the equivalent states.

[Elsayed] K. Elsayed, and A. Rizk, “On the Impact of Network Delays on Time-to-Live Caching,” *ArXiv abs/2201.1157, 2022.*

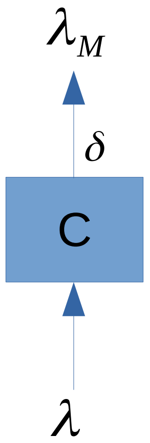
Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

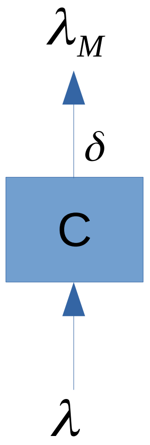
Single M/M/M cache:



Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:



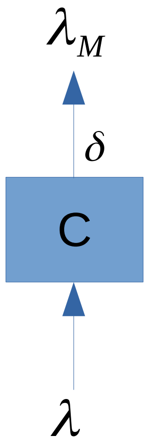
λ : request rate
λ_M : miss rate
$1/\mu_F$: mean fetching time
δ : random delay

Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:

- Hit → zero delay



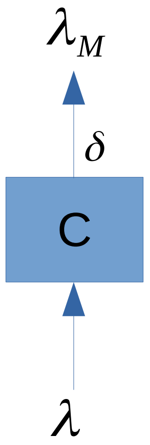
λ: request rate
λ_M: miss rate
$1/\mu_F$: mean fetching time
δ: random delay

Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:

- Hit → zero delay
- Miss → $1/\mu_F$



λ : request rate
λ_M : miss rate
$1/\mu_F$: mean fetching time
δ : random delay

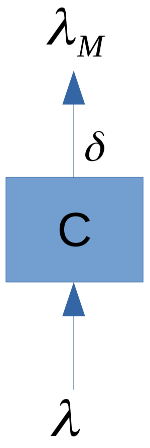
Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:

- Hit → zero delay
- Miss → $1/\mu_F$

$$\bar{R} = P_{hit} E[\delta | hit] + P_{miss} E[\delta | miss]$$



λ : request rate
λ_M : miss rate
$1/\mu_F$: mean fetching time
δ : random delay

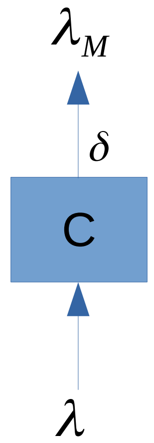
Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:

- Hit → zero delay
- Miss → $1/\mu_F$

$$\bar{R} = P_{hit}\underbrace{E[\delta|hit]}_{0} + P_{miss}E[\delta|miss]$$



λ : request rate
λ_M : miss rate
$1/\mu_F$: mean fetching time
δ : random delay

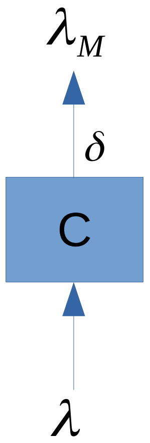
Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:

- Hit → zero delay
- Miss → $1/\mu_F$

$$\bar{R} = P_{hit}\underbrace{E[\delta|hit]}_{0} + P_{miss}\underbrace{E[\delta|miss]}_{1/\mu_F}$$



λ: request rate
λ_M: miss rate
$1/\mu_F$: mean fetching time
δ: random delay

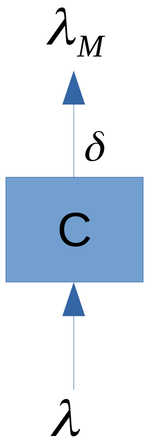
Response Time

- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:

- Hit → zero delay
- Miss → $1/\mu_F$

$$\bar{R}=P_{hit}\underbrace{E[\delta|hit]}_{0}+\underbrace{P_{miss}}_{\frac{\lambda_M}{\lambda}}\underbrace{E[\delta|miss]}_{1/\mu_F}$$



λ: request rate
λ_M: miss rate
$1/\mu_F$: mean fetching time
δ: random delay

Response Time

UNIVERSITÄT DUISBURG ESSEN
Offen im Denken

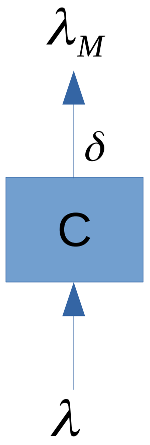
- The mean response time $\bar{R}$ depends on
  - The mean fetching delay
  - The average miss rate at each cache

Single M/M/M cache:

- Hit → zero delay
- Miss → $1/\mu_F$

$$\bar{R} = P_{hit}\underbrace{E[\delta|hit]}_{0} + \underbrace{P_{miss}}_{\frac{\lambda_M}{\lambda}}\underbrace{E[\delta|miss]}_{1/\mu_F}$$

- How to calculate λ_M ?

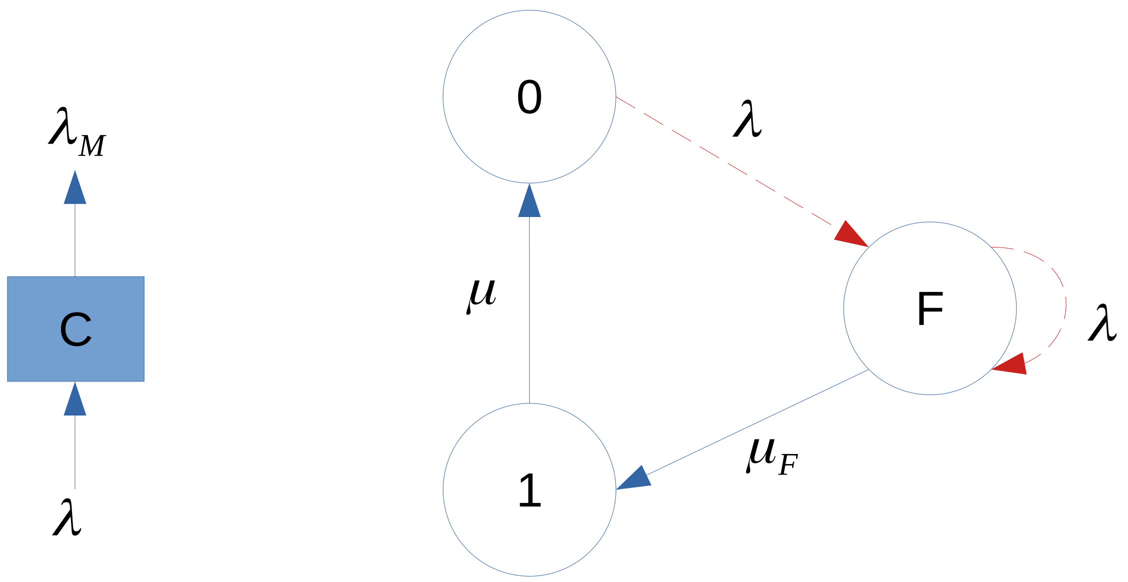


λ : request rate
λ_M : miss rate
$1/\mu_F$: mean fetching time
δ : random delay

Response Time

Single cache miss rate

- From the MAP $\rightarrow$ $\boldsymbol{D_1}$ contains the active transitions



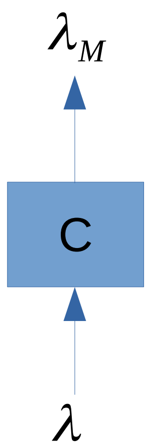
Response Time

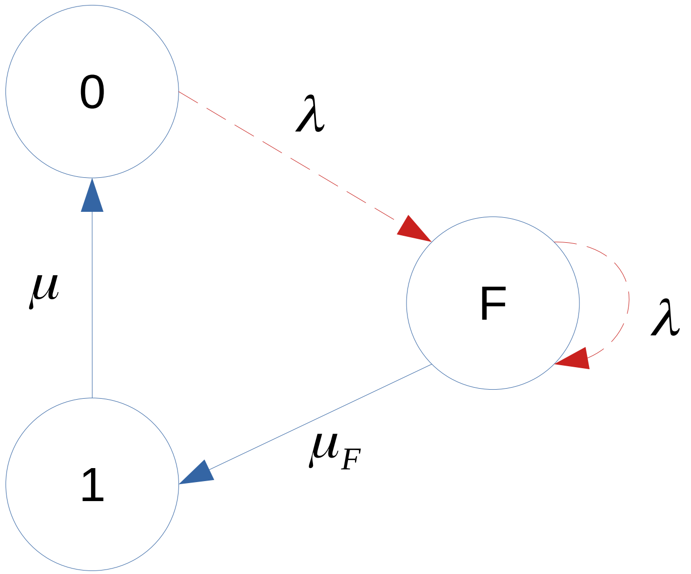
Single cache miss rate

- From the MAP $\rightarrow$ $\boldsymbol{D}_1$ contains the active transitions

$$\lambda_M = \boldsymbol{\pi} \boldsymbol{D}_1 \mathbf{1},$$

$\boldsymbol{\pi}$: Steady state probability vector
$\mathbf{1}$: All ones vector





Response Time

Single cache miss rate

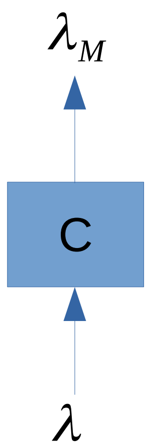
- From the MAP $\rightarrow$ $\boldsymbol{D}_1$ contains the active transitions

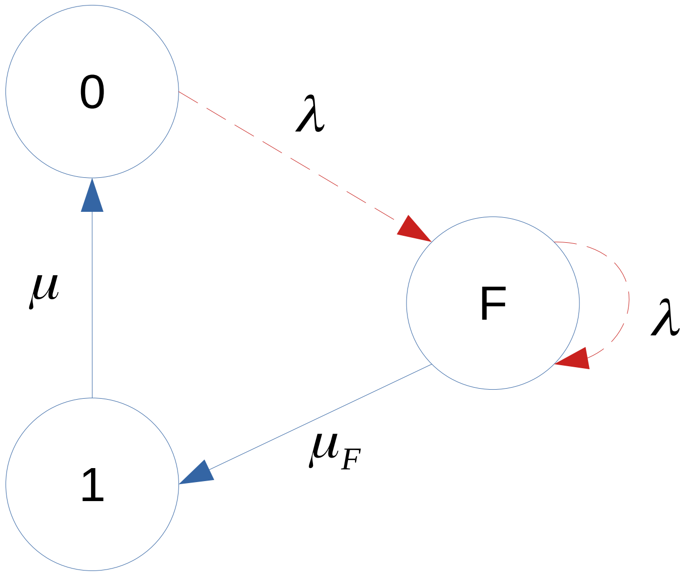
$$\lambda_M = \boldsymbol{\pi} \boldsymbol{D}_1 \mathbf{1},$$

$\boldsymbol{\pi}$: Steady state probability vector
$\mathbf{1}$: All ones vector

- Exact mean response time

$$E[R] = \frac{\boldsymbol{\pi} \boldsymbol{D}_1 \mathbf{1}}{\lambda \mu_f}$$

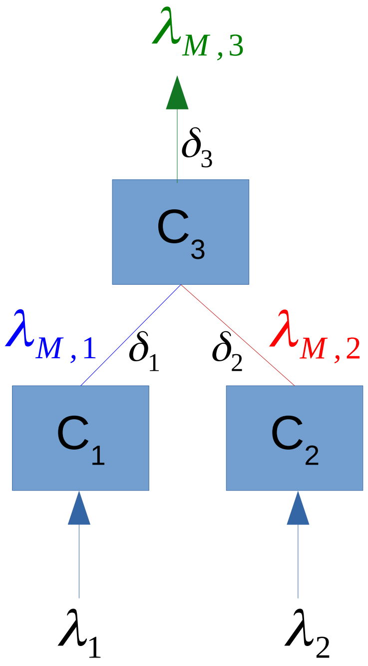




Response Time

M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

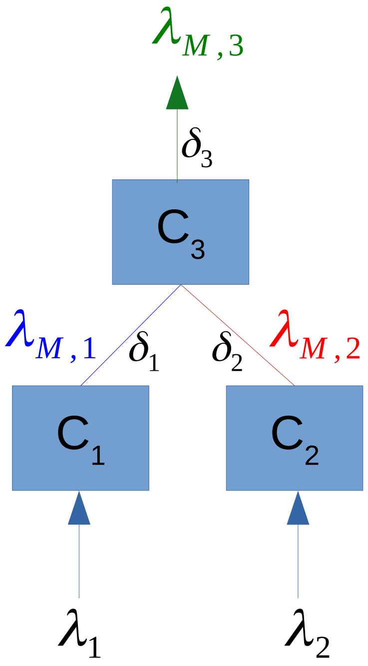


Response Time

M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} =$$

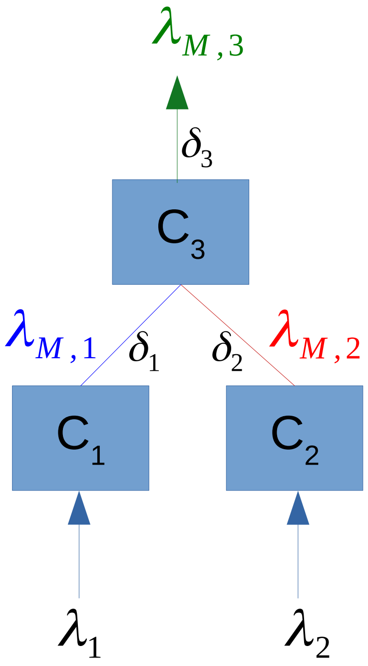


Response Time

M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1]}{\phantom{\lambda_{M,1} E[\delta_1] + \lambda_{M,2} E[\delta_2] + \lambda_{M,3} E[\delta_3]}}$$



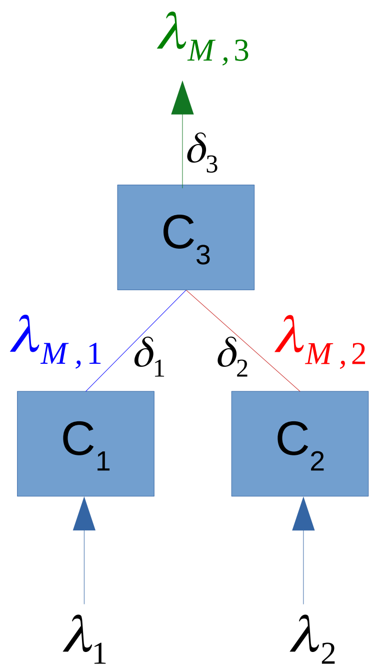
Response Time

M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1]}{}$$

$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1



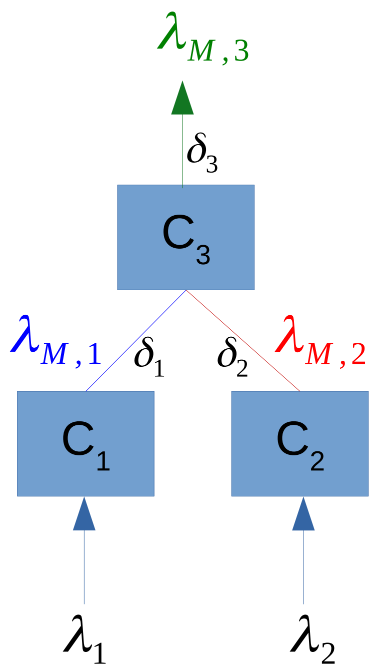
Response Time

M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1] + \lambda_{M,2} E[\delta_2]}{}$$

$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1



Response Time

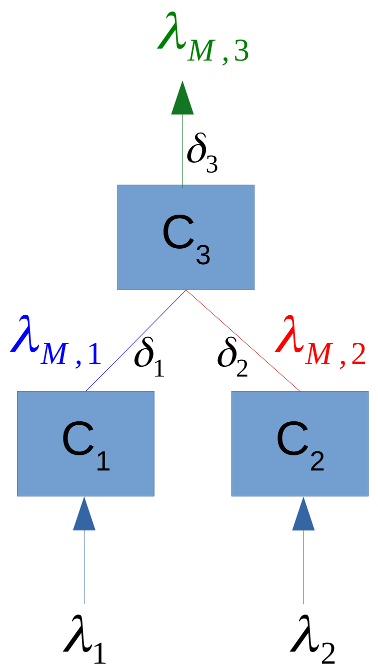
M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1] + \lambda_{M,2} E[\delta_2]}{\quad}$$

$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1

$\lambda_{M,2} \rightarrow$ MAP M_2 modelling C_2



Response Time

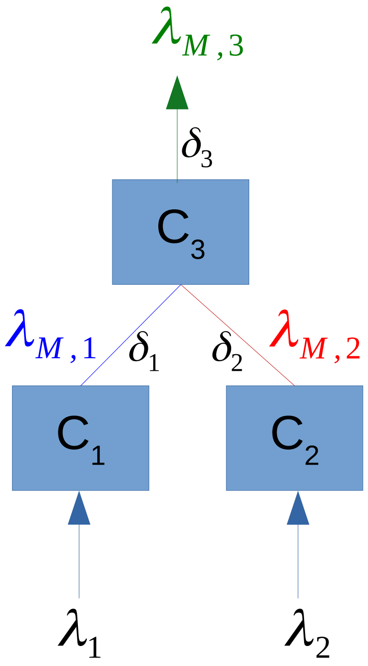
M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R}=\frac{\lambda_{M,1}E[\delta_1]+\lambda_{M,2}E[\delta_2]+\lambda_{M,3}E[\delta_3]}{}$$

$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1

$\lambda_{M,2} \rightarrow$ MAP M_2 modelling C_2



Response Time

M/M/M Cache Hierarchy

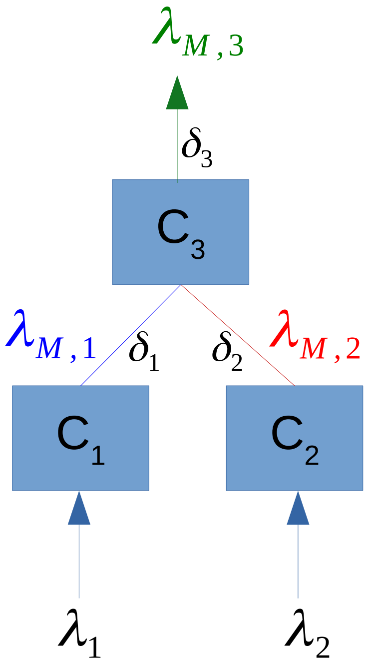
- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1] + \lambda_{M,2} E[\delta_2] + \lambda_{M,3} E[\delta_3]}{}$$

$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1

$\lambda_{M,2} \rightarrow$ MAP M_2 modelling C_2

$\lambda_{M,3} \rightarrow$ MAP M modelling the tree



Response Time

M/M/M Cache Hierarchy

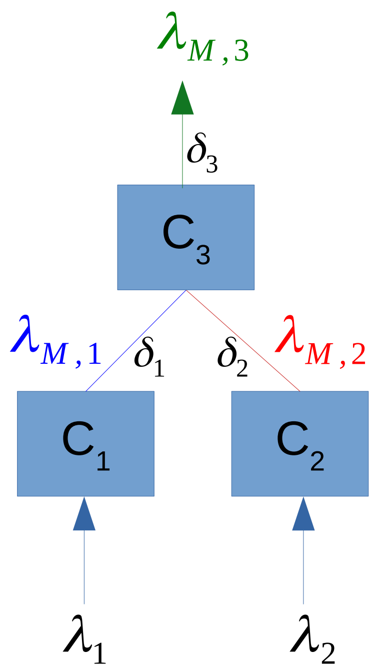
- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1] + \lambda_{M,2} E[\delta_2] + \lambda_{M,3} E[\delta_3]}{\lambda_1 + \lambda_2}$$

$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1

$\lambda_{M,2} \rightarrow$ MAP M_2 modelling C_2

$\lambda_{M,3} \rightarrow$ MAP M modelling the tree



Response Time

M/M/M Cache Hierarchy

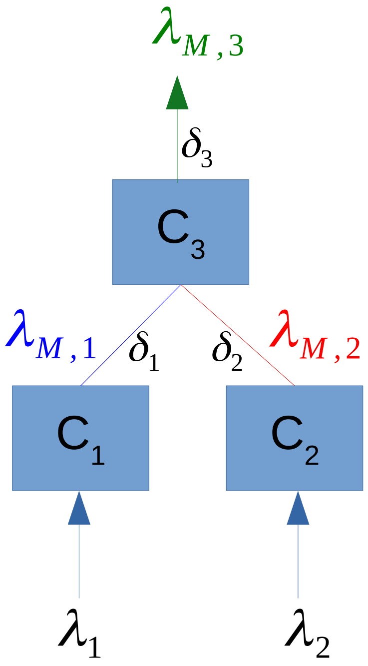
- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1] + \lambda_{M,2} E[\delta_2] + \lambda_{M,3} E[\delta_3]}{\lambda_1 + \lambda_2}$$

$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1

$\lambda_{M,2} \rightarrow$ MAP M_2 modelling C_2

$\lambda_{M,3} \rightarrow$ MAP M modelling the tree



- For any number of caches:

Response Time

M/M/M Cache Hierarchy

- Iterative accumulation of fetching delays due to the misses at each cache

$$\bar{R} = \frac{\lambda_{M,1} E[\delta_1] + \lambda_{M,2} E[\delta_2] + \lambda_{M,3} E[\delta_3]}{\lambda_1 + \lambda_2}$$

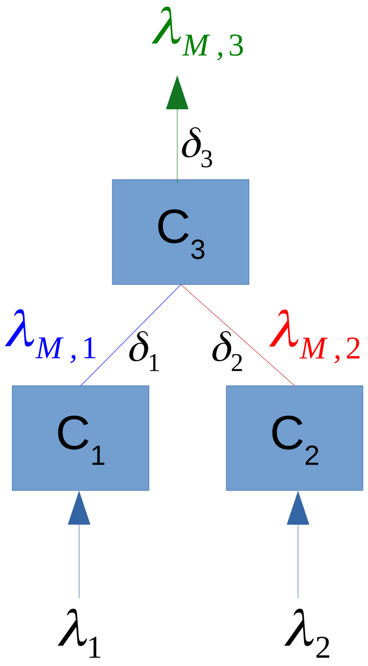
$\lambda_{M,1} \rightarrow$ MAP M_1 modelling C_1

$\lambda_{M,2} \rightarrow$ MAP M_2 modelling C_2

$\lambda_{M,3} \rightarrow$ MAP M modelling the tree

- For any number of caches:

$$\bar{R} = \frac{\sum_i \boldsymbol{\pi}^i \boldsymbol{D}_1^i E[\delta_i] \mathbf{1}}{\sum_i \lambda_i}$$



Response Time

PH fetching delay

- The fetching process is represented by multiple states

Response Time

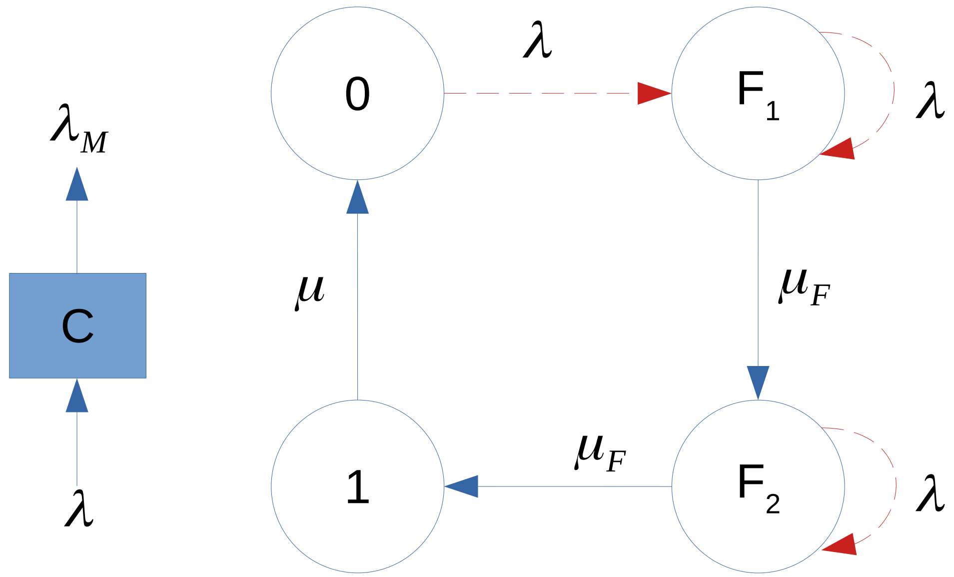
PH fetching delay

- The fetching process is represented by multiple states
  - Example: Erlang-2 distribution

Response Time

PH fetching delay

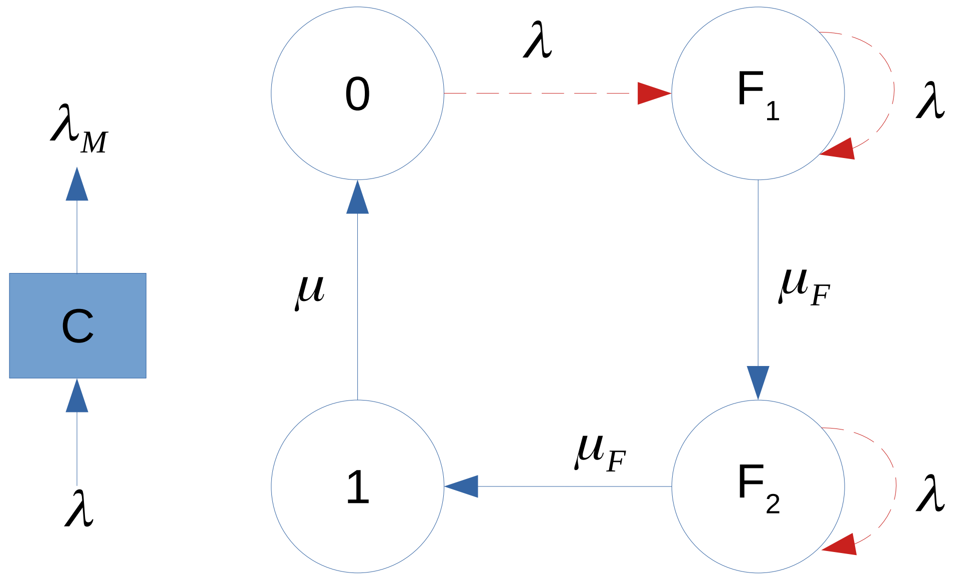
- The fetching process is represented by multiple states
  - Example: Erlang-2 distribution



Response Time

PH fetching delay

- The fetching process is represented by multiple states
  - Example: Erlang-2 distribution
- Aggregate requests see different mean delays

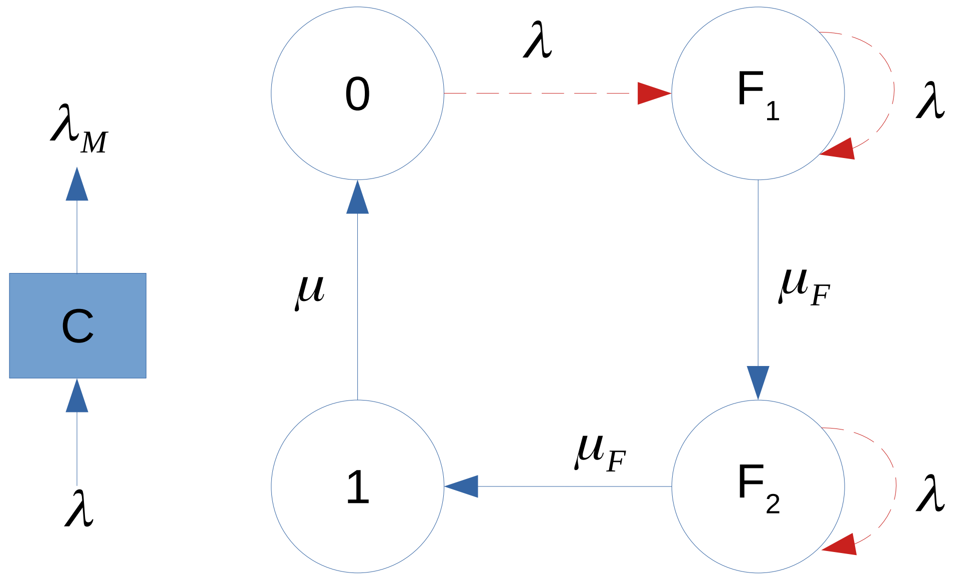


Response Time

PH fetching delay

- The fetching process is represented by multiple states
  - Example: Erlang-2 distribution
- Aggregate requests see different mean delays
- Arrivals at states [1, 0, F_1, F_2] see mean delays

$$\boldsymbol{\alpha}=[0,\frac{2}{\mu_F},\frac{2}{\mu_F},\frac{1}{\mu_F}]$$

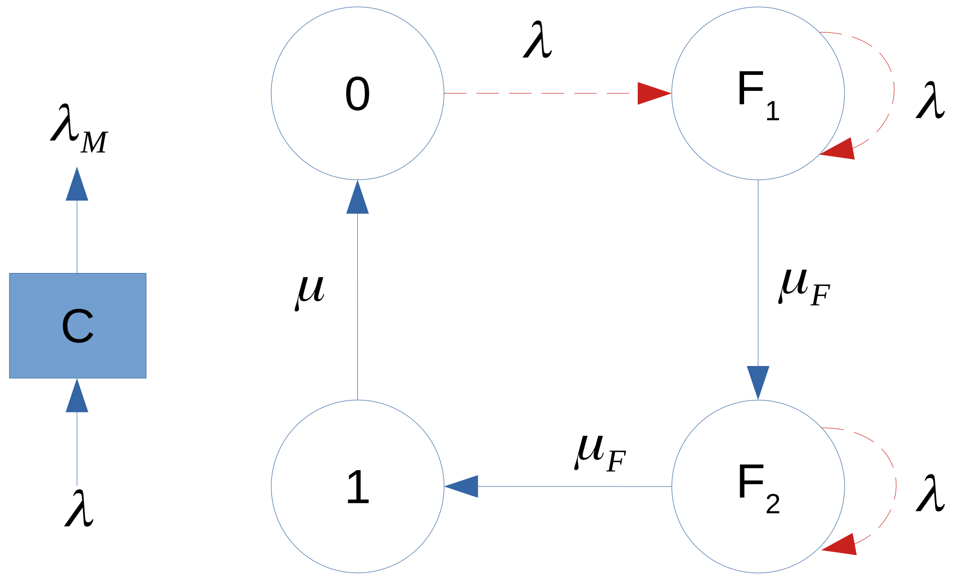


Response Time

PH fetching delay

- The fetching process is represented by multiple states
  - Example: Erlang-2 distribution
- Aggregate requests see different mean delays
- Arrivals at states $[1, 0, F_1, F_2]$ see mean delays

$$\boldsymbol{\alpha} = [0, \frac{2}{\mu_F}, \frac{2}{\mu_F}, \frac{1}{\mu_F}]$$



Response Time

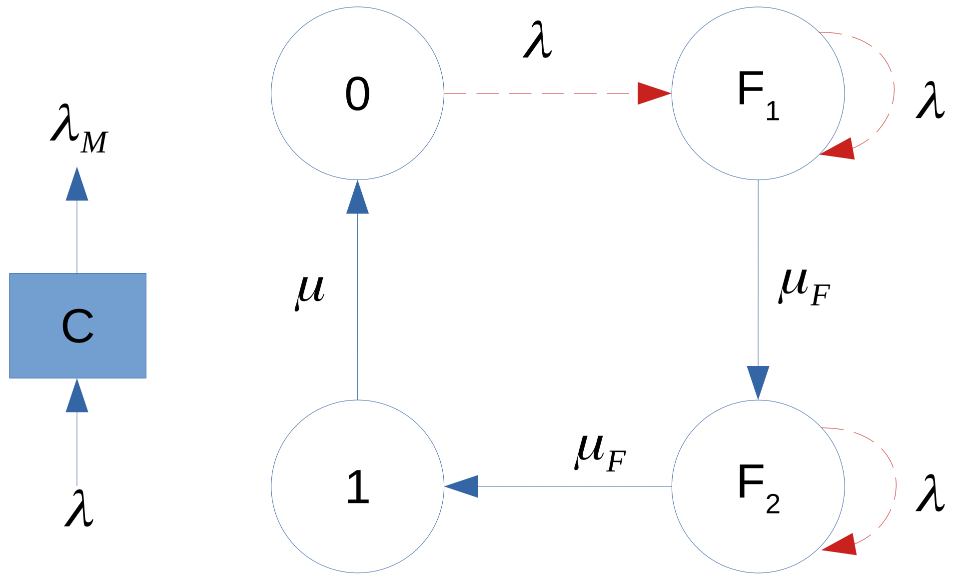
PH fetching delay

- The fetching process is represented by multiple states
  - Example: Erlang-2 distribution
- Aggregate requests see different mean delays
- Arrivals at states $[1, 0, F_1, F_2]$ see mean delays

$$\boldsymbol{\alpha}=[0, \frac{2}{\mu_F}, \frac{2}{\mu_F}, \frac{1}{\mu_F}]$$

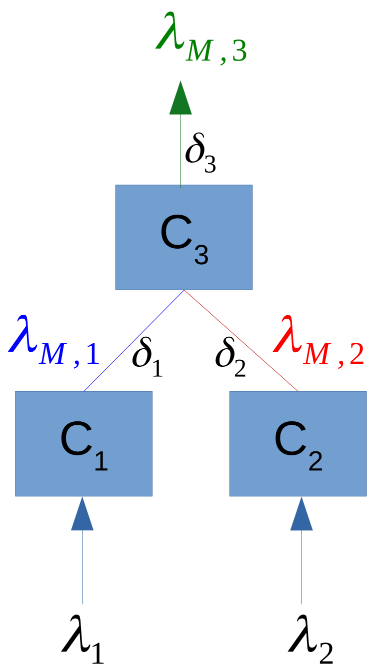
- For any fetching delay distribution

$$\bar{R}=\frac{\sum_i (\boldsymbol{\pi}^i \odot \boldsymbol{\alpha}) \boldsymbol{D}_1^i \mathbf{1}}{\sum_i \lambda_i}, \quad \odot: \text{Hadamard prodcut}$$



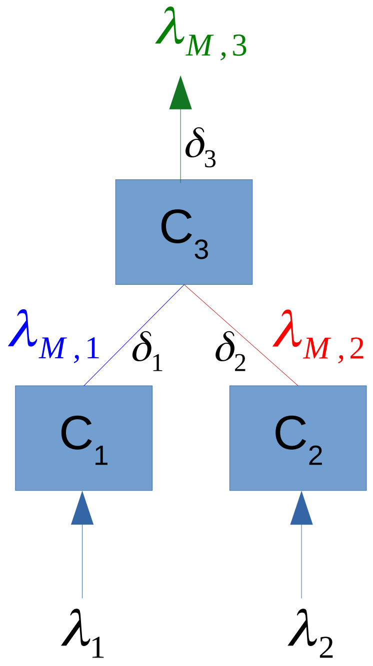
Response Time

- Using the same concept we can calculate



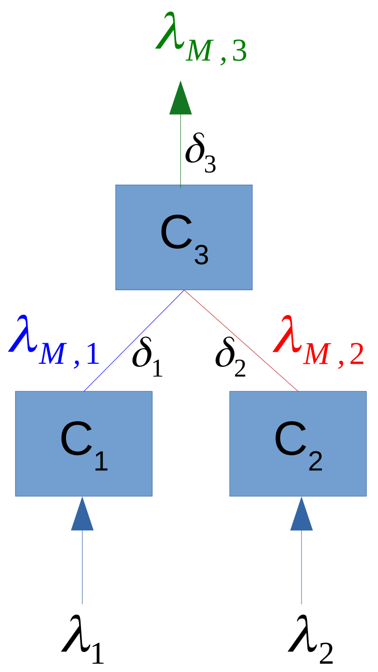
Response Time

- Using the same concept we can calculate
  - Mean response time for each input stream



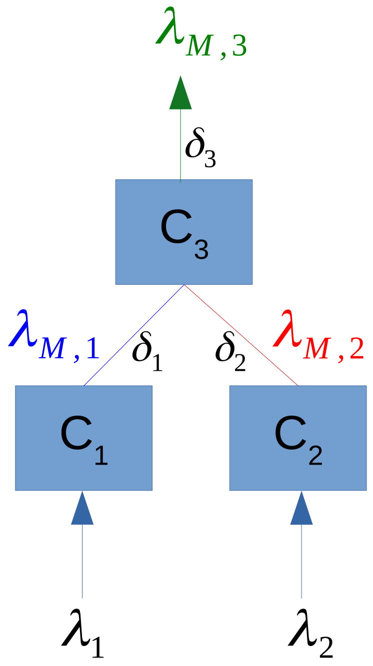
Response Time

- Using the same concept we can calculate
  - Mean response time for each input stream
  - Mean response time given a system hit/miss



Response Time

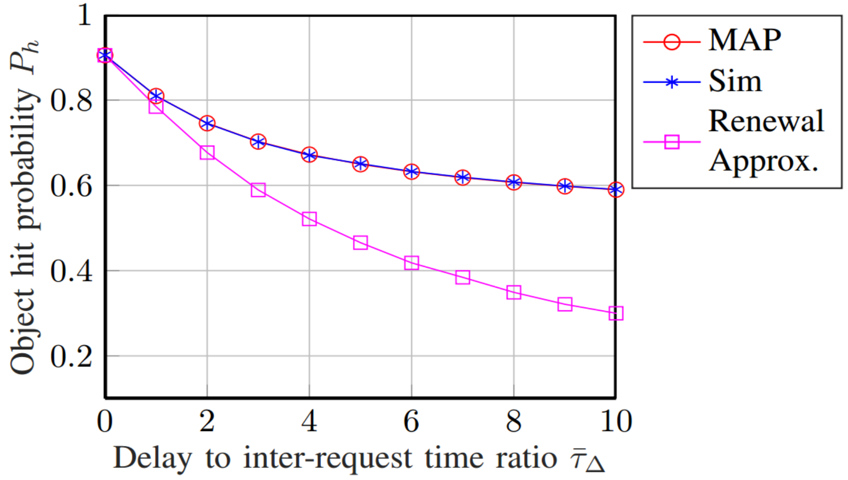
- Using the same concept we can calculate
  - Mean response time for each input stream
  - Mean response time given a system hit/miss
  - Mean response time given a PH/PH/PH hierarchy

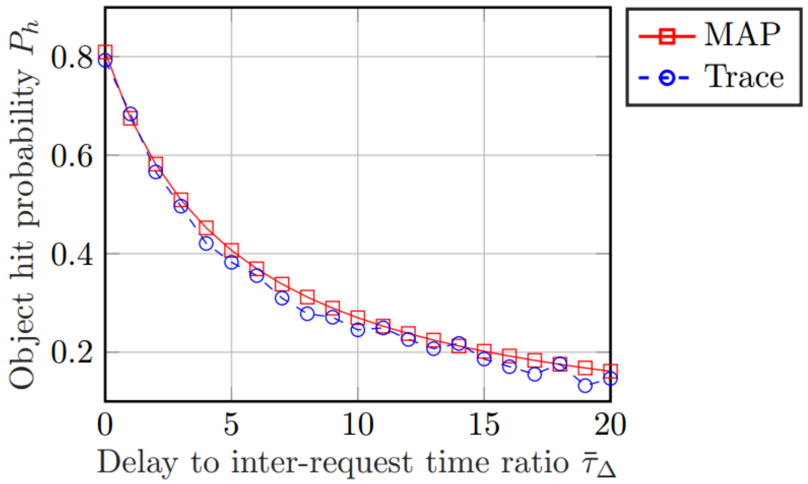


Evaluation

Delay impact on hit probability

- Two level M/M/M hierarchy
  - Simulation (only for validation)
  - MAP (Exact model)
  - Renewal approximation (based on Related work)





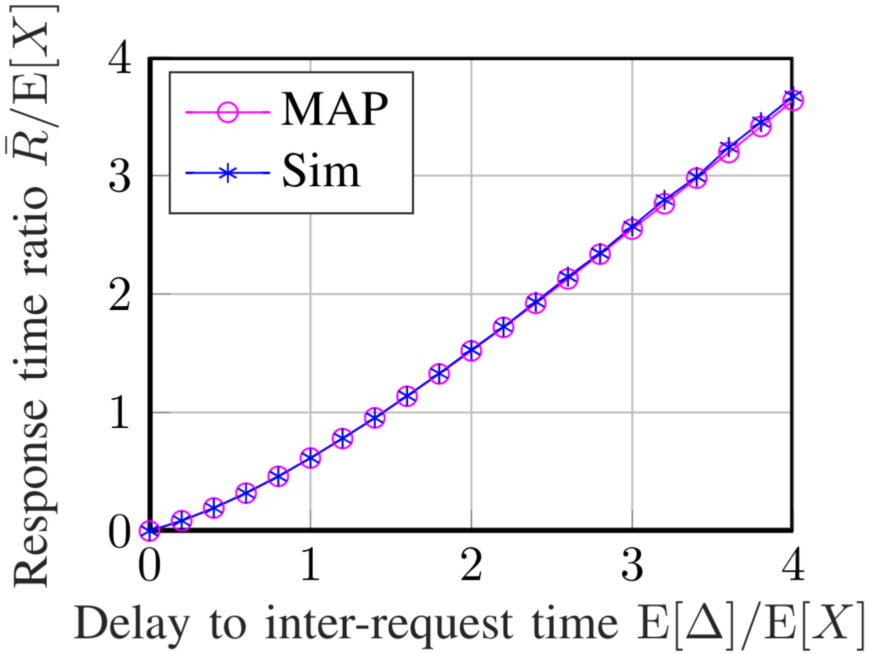
$\bar{\tau}_\Delta$ mean delay time / mean Inter-request time

Trace from SNIA, 2011. "Storage Networking Industry Association's Input/Output Traces, Tools, and Analysis Technical Work Group". Iotta.snia.org

Evaluation

Response time

- Two level M/ E_2 / E_2 hierarchy
  - Simulation (only for validation)
  - MAP (Exact model)



Conclusion & Future direction

- Fetching delays in cache hierarchies remarkably impact the performance (response time and hit probability)
- MAPs for cache hierarchies are formed recursively to provide an exact model with delays
- Mean response time is iteratively calculated from the MAP
- **<u>Open topic</u>:** The Response time distribution derivation given the MAP of a cache hierarchy