

On the Analysis of Caches with Pending Interest Tables



Mostafa Dehghan¹, Bo Jiang¹
Ali Dabirmoghaddam², Don Towsley¹

¹University of Massachusetts Amherst

²University of California Santa Cruz

ICN, October 2, 2015

Overview

- Named Data Networking
 - Data names instead of IP addresses

- Data Structures:
 - Content Store
 - Pending Interest Table
 - Forwarding Information Base

- Packets:
 - Interest packet
 - Data packet

Overview

Content Store

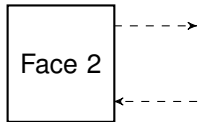
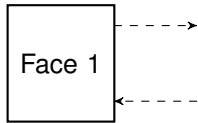
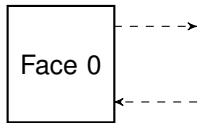
Name	Data

Pending Interest Table

Name	Face

Forwarding Information Base

Prefix	Face
/icn/	1



Overview

Content Store

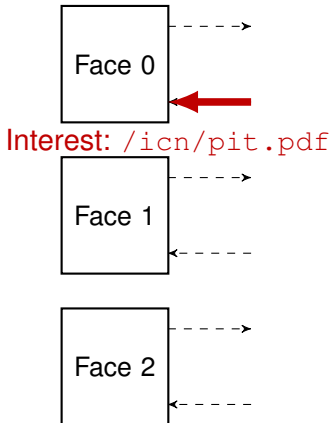
Name	Data

Pending Interest Table

Name	Face

Forwarding Information Base

Prefix	Face
/icn/	1



Overview

Content Store

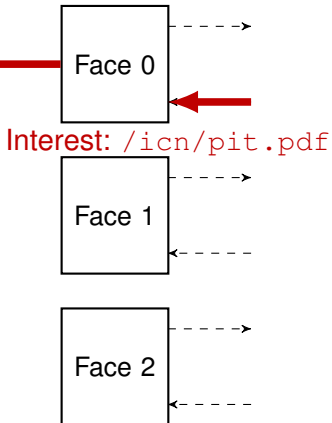
Name	Data

Pending Interest Table

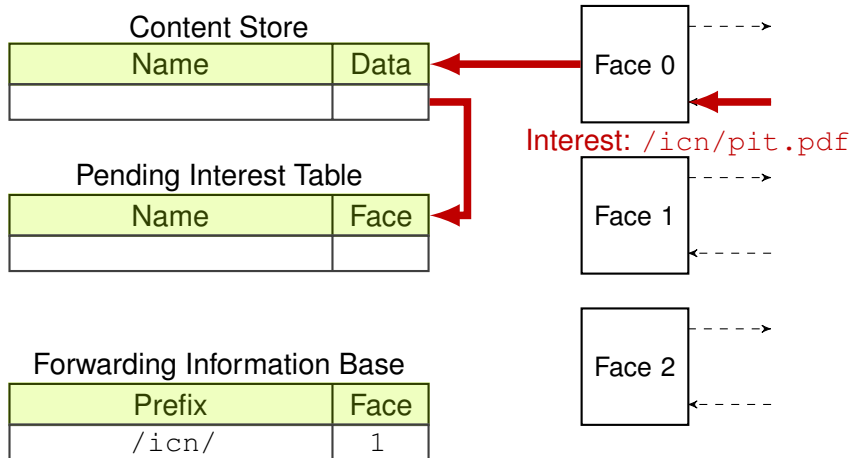
Name	Face

Forwarding Information Base

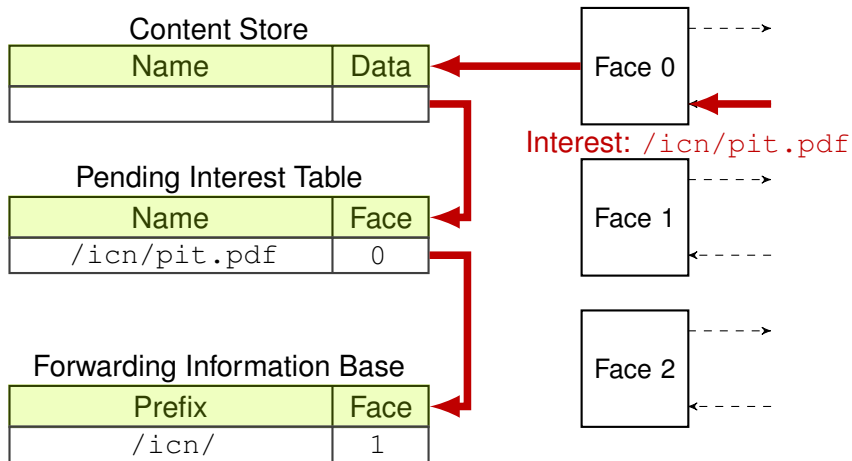
Prefix	Face
/icn/	1



Overview



Overview



Overview

Content Store

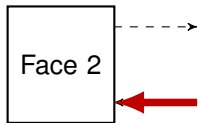
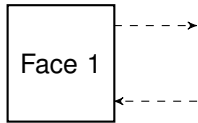
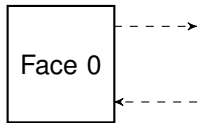
Name	Data

Pending Interest Table

Name	Face
/icn/pit.pdf	0

Forwarding Information Base

Prefix	Face
/icn/	1



Interest: /icn/pit.pdf

Overview

Content Store

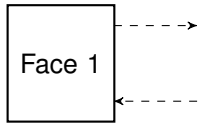
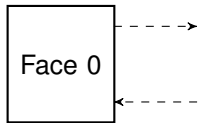
Name	Data

Pending Interest Table

Name	Face
/icn/pit.pdf	0

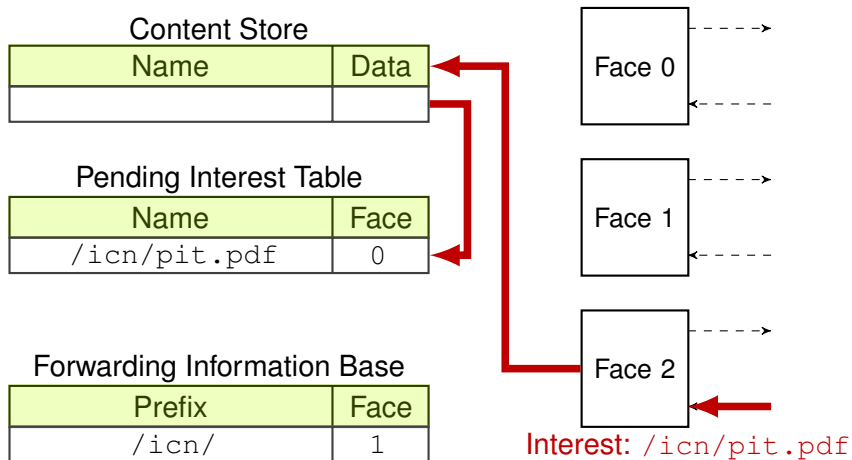
Forwarding Information Base

Prefix	Face
/icn/	1

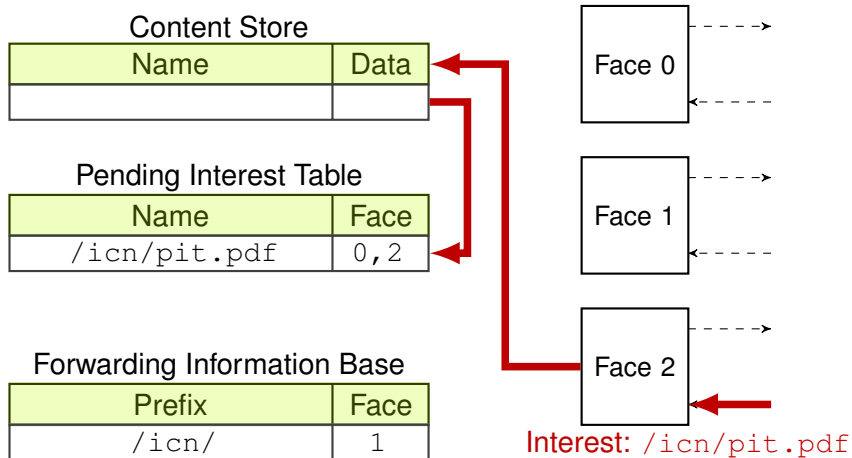


Interest: /icn/pit.pdf

Overview



Overview



Overview

Content Store

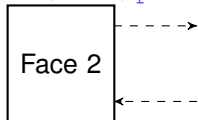
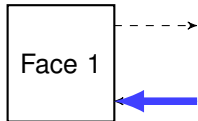
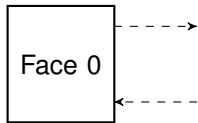
Name	Data

Pending Interest Table

Name	Face
/icn/pit.pdf	0, 2

Forwarding Information Base

Prefix	Face
/icn/	1



Overview

Content Store

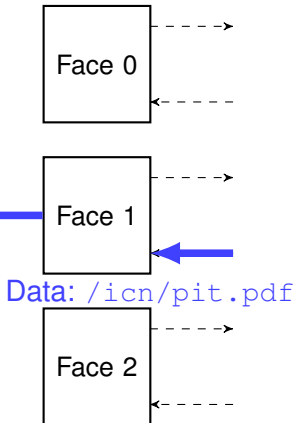
Name	Data

Pending Interest Table

Name	Face
/icn/pit.pdf	0, 2

Forwarding Information Base

Prefix	Face
/icn/	1



Overview

Content Store

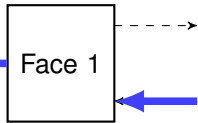
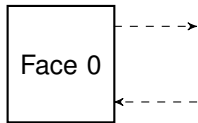
Name	Data
/icn/pit.pdf	...

Pending Interest Table

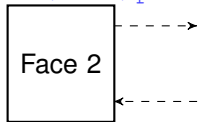
Name	Face
/icn/pit.pdf	0, 2

Forwarding Information Base

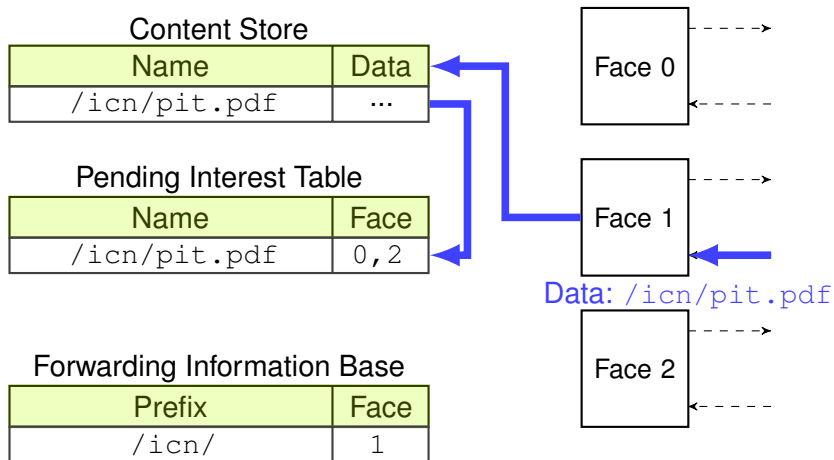
Prefix	Face
/icn/	1



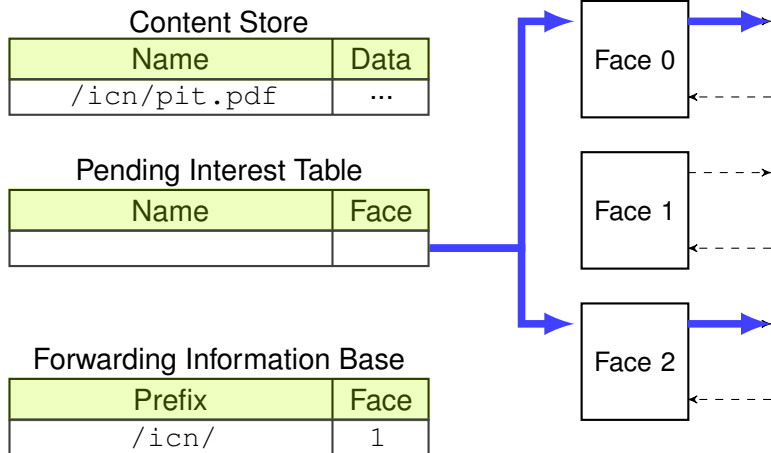
Data: /icn/pit.pdf



Overview



Overview



Pending Interest Table (PIT)

- Performs request aggregation
 - Keep track of Interests forwarded but not yet satisfied

Pending Interest Table (PIT)

- Performs request aggregation
 - Keep track of Interests forwarded but not yet satisfied

- Advantages:
 - lower bandwidth usage
 - communication without knowledge of source and destination
 - better security
 - ...

Pending Interest Table (PIT)

- Performs request aggregation
 - Keep track of Interests forwarded but not yet satisfied

- Advantages:
 - lower bandwidth usage
 - communication without knowledge of source and destination
 - better security
 - ...

- Affects cache behavior
 - hit probability
 - response time

Motivation

- Need analytical models to predict cache behavior
 - Better design choices

Motivation

- Need analytical models to predict cache behavior
 - Better design choices

- PIT sizing

Motivation

- Need analytical models to predict cache behavior
 - Better design choices

- PIT sizing

- State of the art:
 - Ignores download delay
 - Does not consider PIT

Motivation

- Need analytical models to predict cache behavior
 - Better design choices

- PIT sizing

- State of the art:
 - Ignores download delay
 - Does not consider PIT

Goal: Analytically model cache with PIT

Outline

- Time-To-Live Caches
- Model and Analysis
- Replacement Based Policies
- Simulation Results

Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration

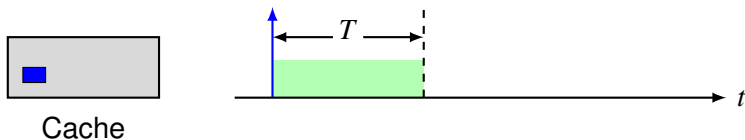
Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration



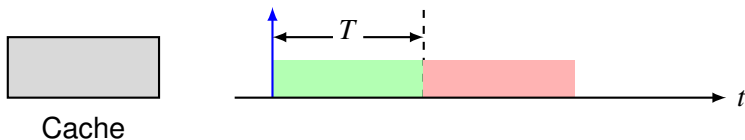
Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration



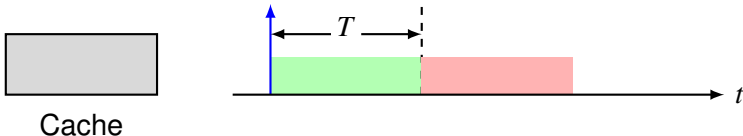
Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration



Time-To-Live (TTL) Caches

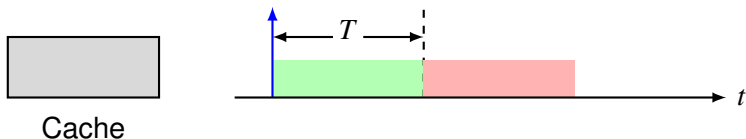
- Content eviction occurs upon timer expiration



- Reset TTL cache
 - reset timer upon each hit

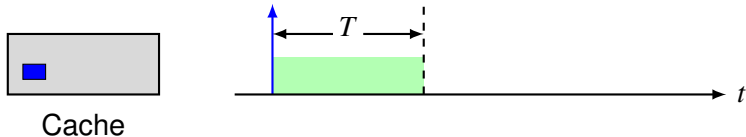
Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration



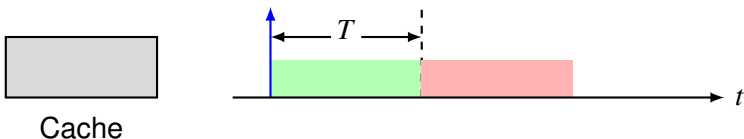
- Reset TTL cache

- reset timer upon each hit



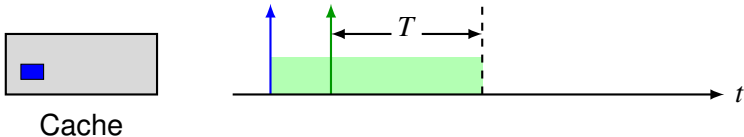
Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration



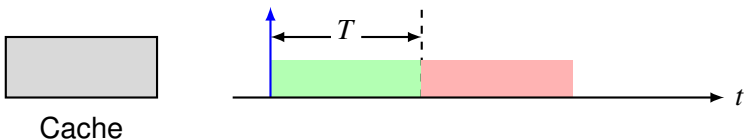
- Reset TTL cache

- reset timer upon each hit



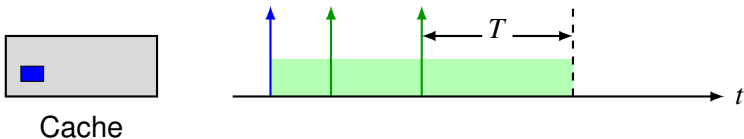
Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration



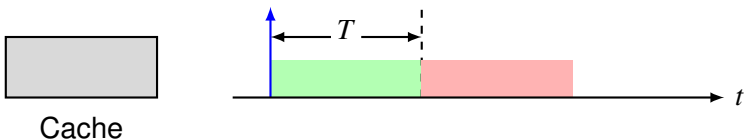
- Reset TTL cache

- reset timer upon each hit



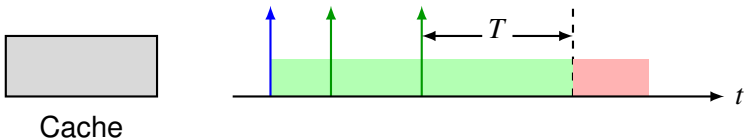
Time-To-Live (TTL) Caches

- Content eviction occurs upon timer expiration



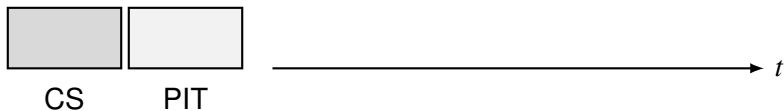
- Reset TTL cache

- reset timer upon each hit



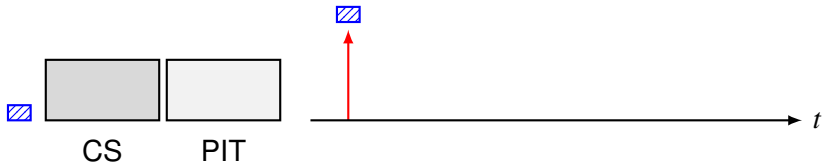
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



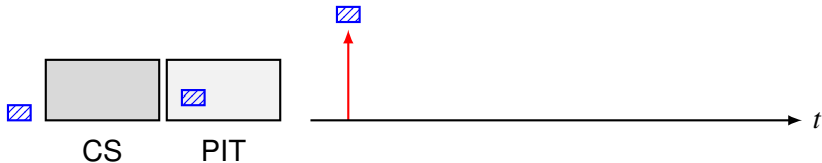
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



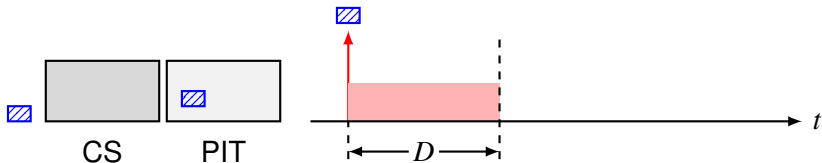
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



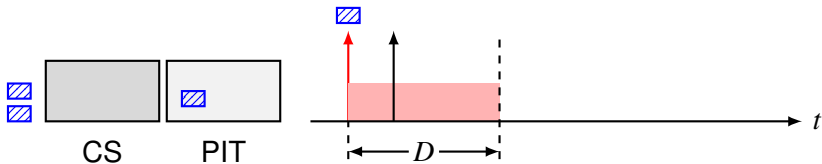
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



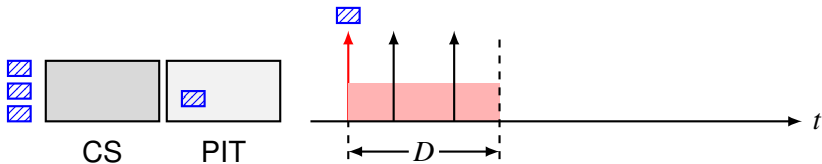
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



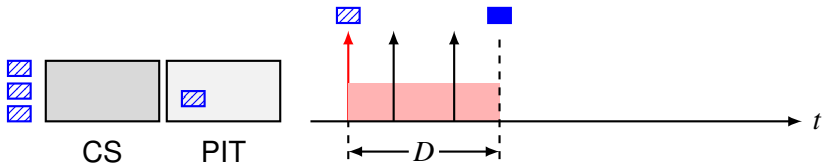
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



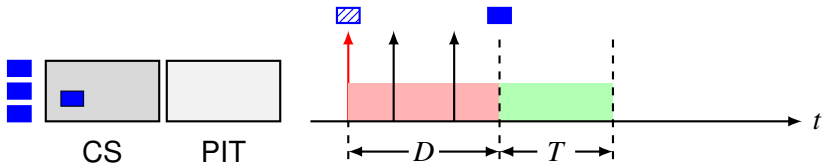
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



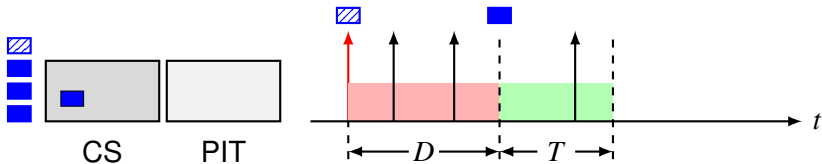
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



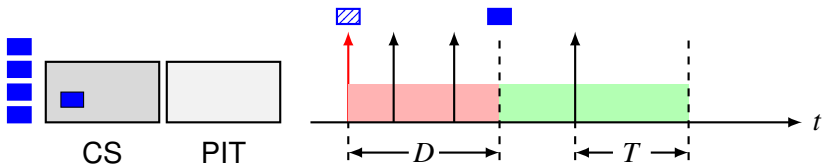
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



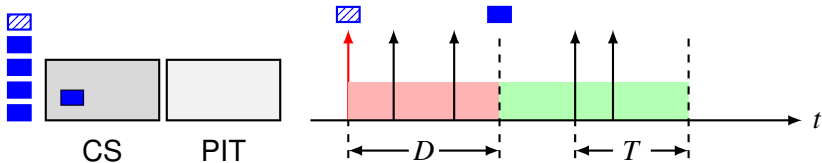
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



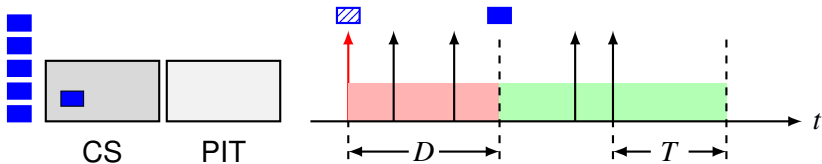
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



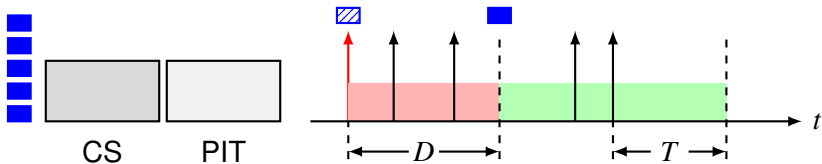
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



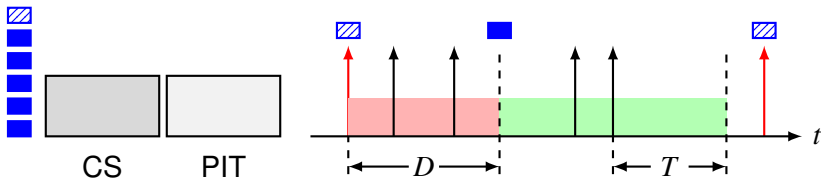
Model

- Requests: renewal arrivals
- Reset TTL cache with PIT

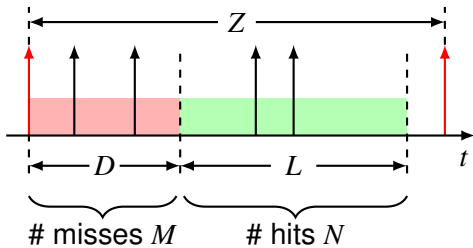


Model

- Requests: renewal arrivals
- Reset TTL cache with PIT



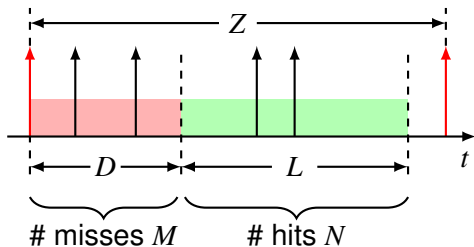
Metrics



Metrics

■ Cache hit prob

$$h = \frac{\mathbb{E}[N]}{\mathbb{E}[M] + \mathbb{E}[N]}$$



Metrics

- Cache hit prob

$$h = \frac{\mathbb{E}[N]}{\mathbb{E}[M] + \mathbb{E}[N]}$$

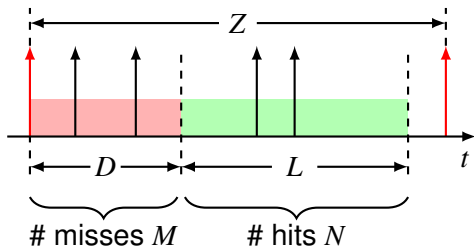
- Cache response time

$$r = \mathbb{E}[w(D)] / (\mathbb{E}[M] + \mathbb{E}[N])$$

$$w(t) = t + \int_0^t (t - x) dm(x)$$

$m(t)$: renewal function

Poisson: $m(t) = \lambda t$



Metrics

- Cache hit prob

$$h = \frac{\mathbb{E}[N]}{\mathbb{E}[M] + \mathbb{E}[N]}$$

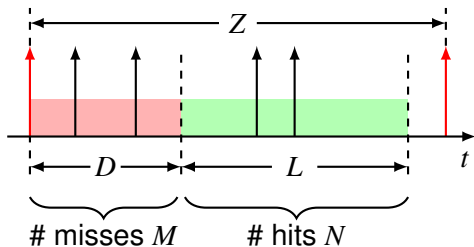
- Cache response time

$$r = \mathbb{E}[w(D)] / (\mathbb{E}[M] + \mathbb{E}[N])$$

$$w(t) = t + \int_0^t (t-x) dm(x)$$

$m(t)$: renewal function

Poisson: $m(t) = \lambda t$



- Cache occupancy prob

$$o = \mathbb{E}[L] / \mathbb{E}[Z]$$

Metrics

- Cache hit prob

$$h = \frac{\mathbb{E}[N]}{\mathbb{E}[M] + \mathbb{E}[N]}$$

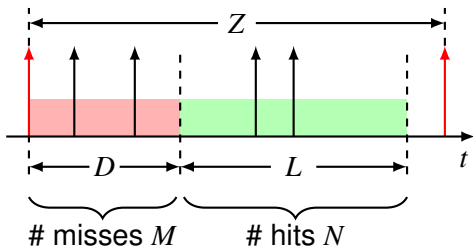
- Cache response time

$$r = \mathbb{E}[w(D)] / (\mathbb{E}[M] + \mathbb{E}[N])$$

$$w(t) = t + \int_0^t (t-x) dm(x)$$

$m(t)$: renewal function

Poisson: $m(t) = \lambda t$



- Cache occupancy prob

$$o = \mathbb{E}[L] / \mathbb{E}[Z]$$

- PIT occupancy prob

$$p = \mathbb{E}[D] / \mathbb{E}[Z]$$

Replacement Based Policies

Characteristic Time Approximation

- Introduced by Che et al. for LRU policy w/ Poisson arrivals
- Extended to other policies, e.g. FIFO, Random, . . .
- Extended to general request arrival processes
- Characteristic time T solution of

$$\sum_i o_i(T) = C$$

- o_i – cache occupancy probability of file i
- C – cache size

LRU Cache with Poisson Arrivals

- Modeled as reset TTL with constant T
- Cache hit/occupancy probability

$$h_i = o_i = \frac{e^{\lambda_i T} - 1}{\lambda_i \mathbb{E}D_i + e^{\lambda_i T}}$$

- λ_i – arrival rate of requests for file i
- $\mathbb{E}D_i$ – expected download delay for file i
- characteristic time T solution of

$$\sum_i \frac{e^{\lambda_i T} - 1}{\lambda_i \mathbb{E}D_i + e^{\lambda_i T}} = C$$

LRU Cache with Poisson Arrivals

- PIT occupancy probability

$$p_i = \frac{\lambda_i \mathbb{E}D_i}{\lambda_i \mathbb{E}D_i + e^{\lambda_i T}}$$

LRU Cache with Poisson Arrivals

- PIT occupancy probability

$$p_i = \frac{\lambda_i \mathbb{E}D_i}{\lambda_i \mathbb{E}D_i + e^{\lambda_i T}}$$

- PIT size

$$S \sim \mathcal{N} \left(\sum_i p_i, \sum_i p_i (1 - p_i) \right)$$

LRU Cache with Poisson Arrivals

- PIT occupancy probability

$$p_i = \frac{\lambda_i \mathbb{E}D_i}{\lambda_i \mathbb{E}D_i + e^{\lambda_i T}}$$

- PIT size

$$S \sim \mathcal{N} \left(\sum_i p_i, \sum_i p_i(1 - p_i) \right)$$

- Average cache response time

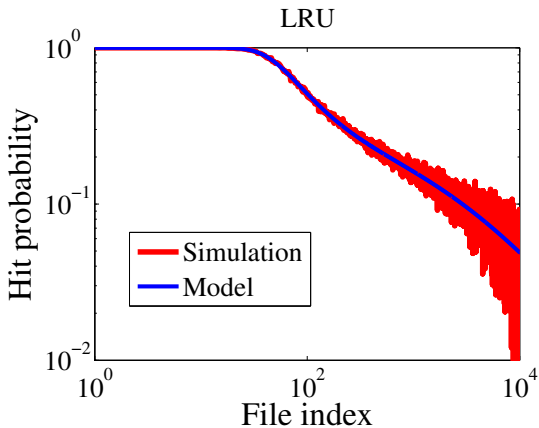
$$r_i = \frac{\mathbb{E}D_i + 0.5\lambda_i \mathbb{E}D_i^2}{\lambda_i \mathbb{E}D_i + e^{\lambda_i T}}$$

Simulation

- Cache size = 10^3
- # different files = 10^4
- Requests: Poisson process w/ rate $\lambda = 10^5$ per second
- File popularity: Zipf distribution, $\lambda_i \propto i^{-0.8}$
- Download delay: 100 ms
- One simulation run: total of 10^7 requests simulated
- Base model:
 - ZDD: Zero-Download Delay

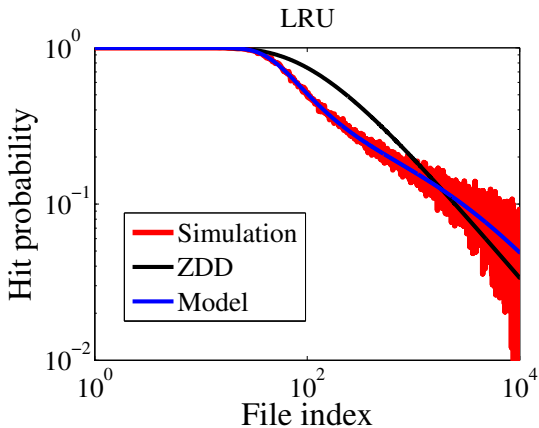
Result – Cache Hit Probability

- Hit probability for individual files ($D = 100$ ms)



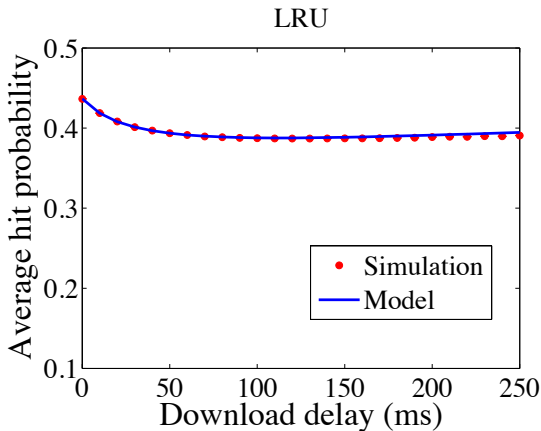
Result – Cache Hit Probability

- Hit probability for individual files ($D = 100$ ms)



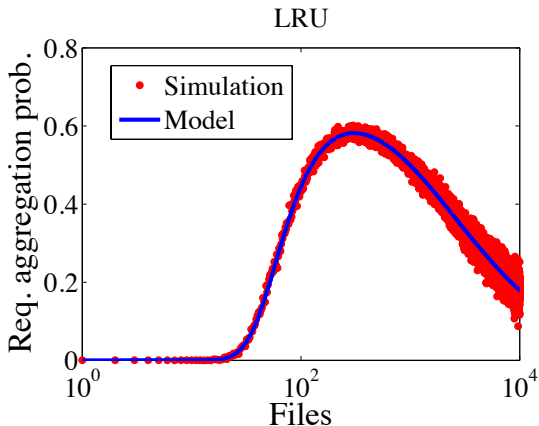
Result – Cache Hit Probability

- Average hit probability vs. download delay



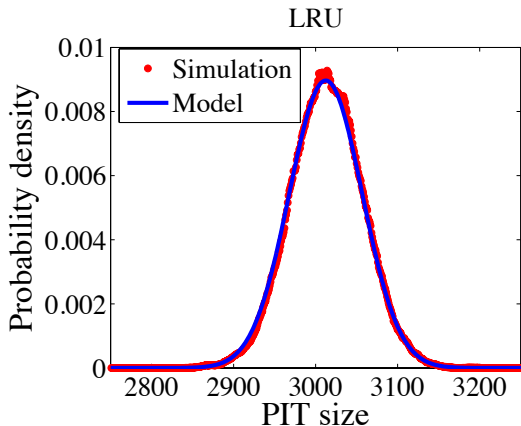
Result – Request Aggregation

- PIT occupancy probability for individual files ($D = 100$ ms)



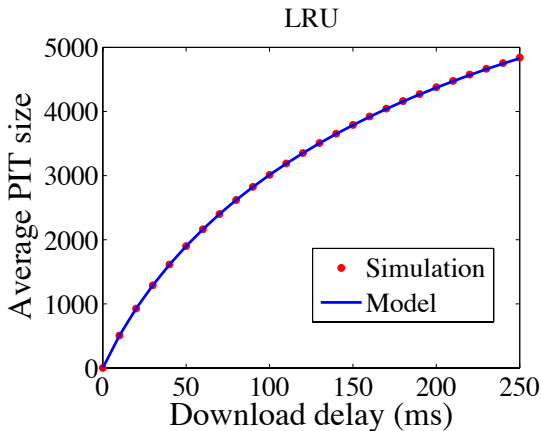
Result – PIT Size

- PIT size distribution



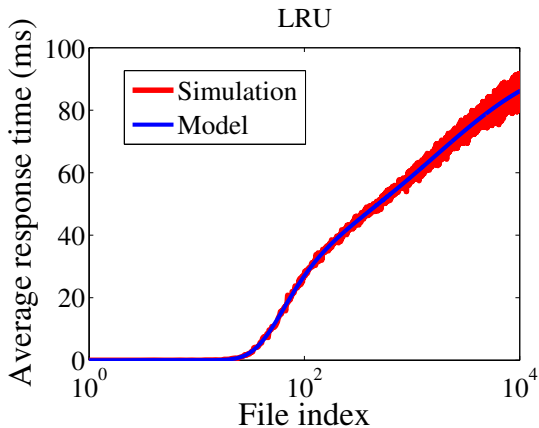
Result – PIT Size

- Average PIT size vs. download delay



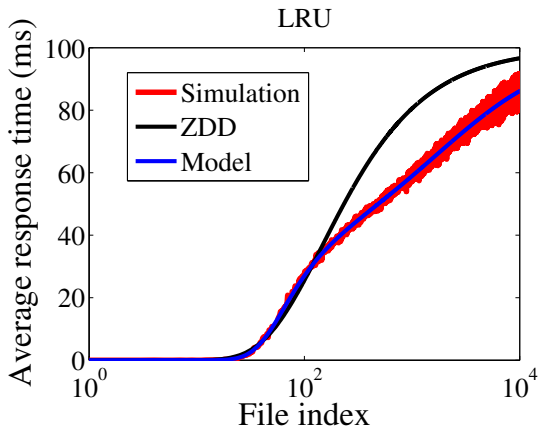
Result – Cache Response Time

- Response time for individual files ($D = 100$ ms)



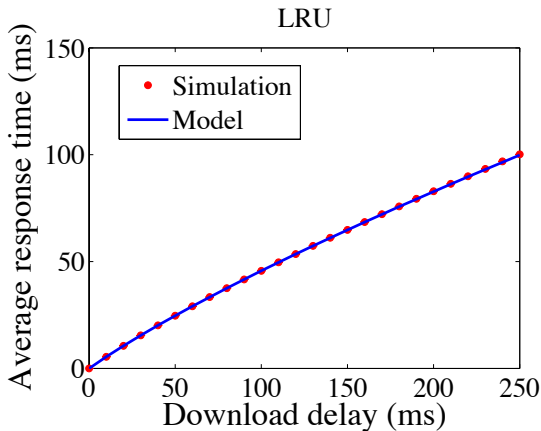
Result – Cache Response Time

- Response time for individual files ($D = 100$ ms)



Result – Cache Response Time

- Average response time vs. download delay

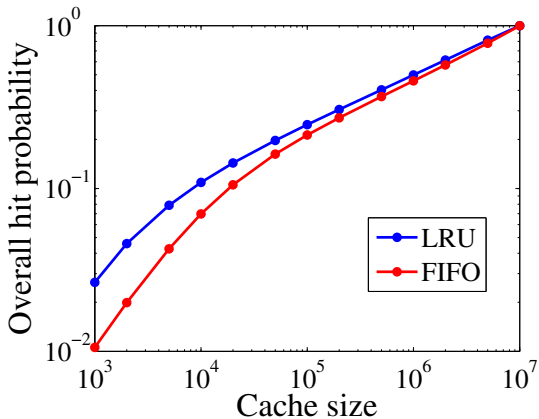


Numerical Results

- Cache size = 10^4
- # different files = 10^7
- Requests: Poisson process w/ rate $\lambda = 10^5$ per second
- File popularity: Zipf distribution, $\lambda_i \propto i^{-0.8}$
- Download delay: 100 ms
- Model only – no simulations

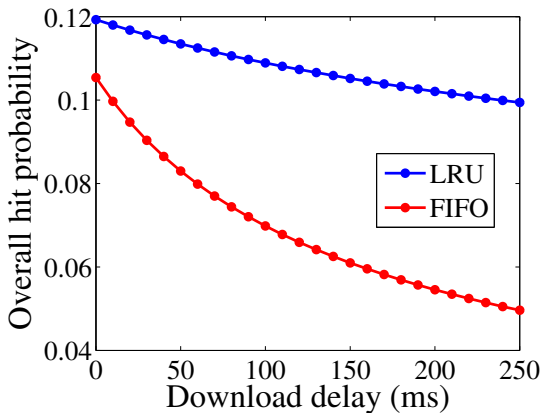
Result – Cache Hit Probability

- Average hit probability vs. cache size



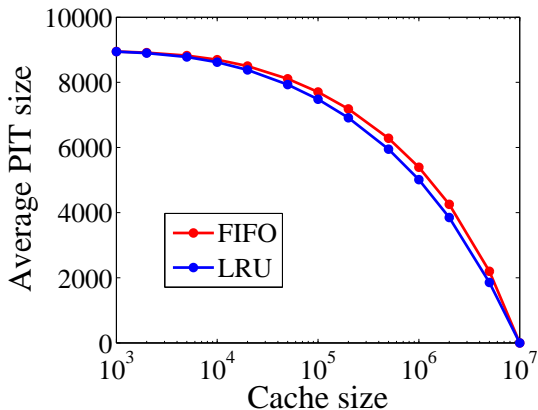
Result – Cache Hit Probability

- Average hit probability vs. download delay



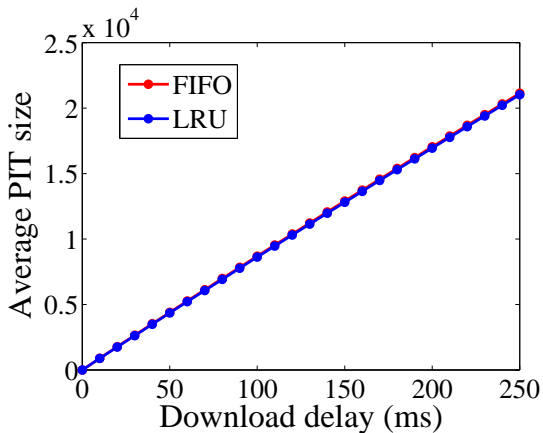
Result – PIT Size

■ Average PIT size vs. cache size



Result – PIT Size

- Average PIT size vs. download delay



Summary

■ Conclusions:

- Modeled cache with PIT and non-zero download delay
 - ◆ Cache hit probability
 - ◆ PIT size
 - ◆ Cache response time
- Analysis based on TTL caches
 - ◆ LRU, FIFO and Random
- Characteristic time holds with positive download delay

■ Future Work:

- Requests arriving for chunks of files (e.g. streaming video)
- Different scales of download delays