

Understanding Sharded Caching Systems

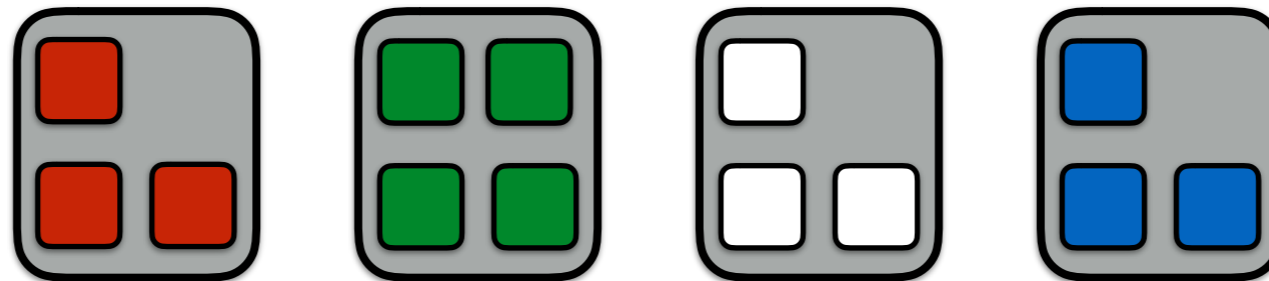
Lorenzo Saino
l.saino@ucl.ac.uk

Ioannis Paras
i.psaras@ucl.ac.uk

George Pavlov
g.pavlou@ucl.ac.uk

Department of Electronic and Electrical Engineering
University College London

What is sharding?



Problem and contributions

There is little theoretical understanding of sharding performance under realistic operational conditions.

We model sharded systems and shed light on their properties in terms of:

- ▶ Load balancing
- ▶ Caching performance

Load balancing

System model

Assumptions and notation

- ▶ N items are sharded across K nodes ($N > K$)
- ▶ Each item is requested with probability p_i (p_1, p_2, \dots, p_N)
- ▶ Each shard receives a random fraction L of requests.

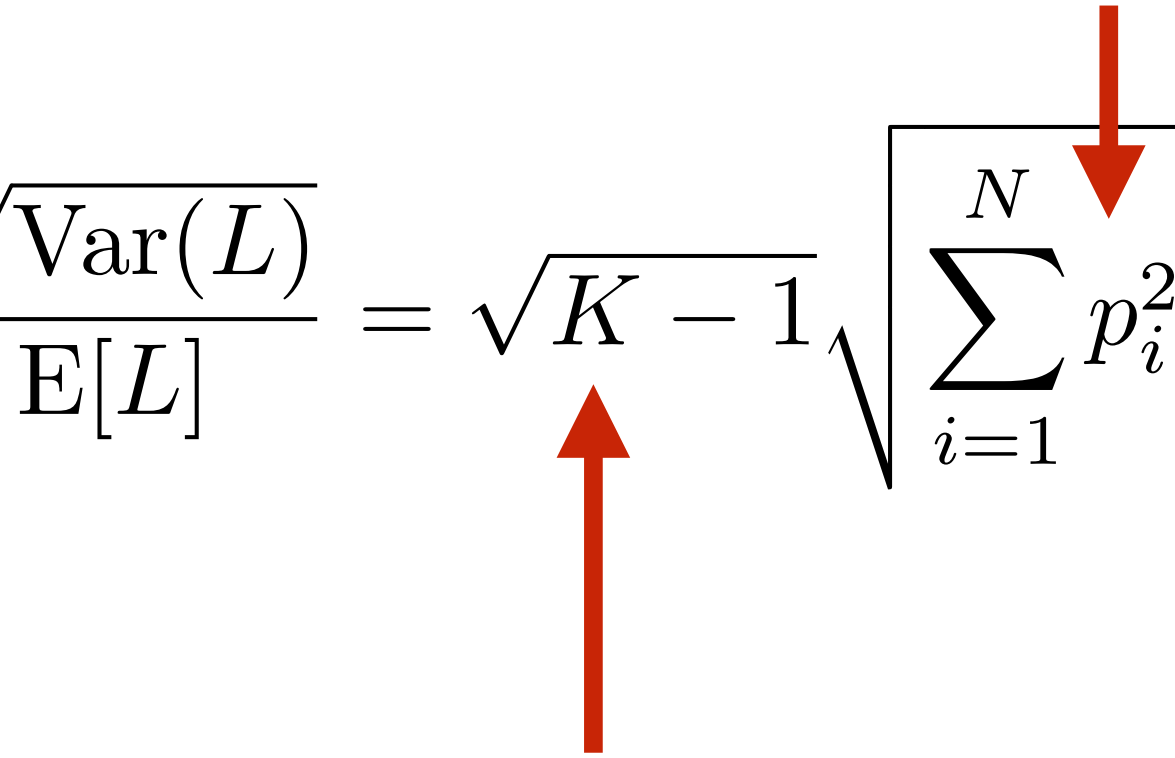
Quantify load imbalance

- ▶ Coefficient of variation of load L of a shard

$$c_v(L) = \frac{\sqrt{\text{Var}(L)}}{\text{E}[L]}$$

General formulation

Load imbalance increases with skewness of item popularity distribution

$$c_v(L) = \frac{\sqrt{\text{Var}(L)}}{\text{E}[L]} = \sqrt{K - 1} \sqrt{\sum_{i=1}^N p_i^2}$$


Load imbalance increases with # of shards (K)

Impact of item popularity distribution

Let's assume that the demand follows a Zipf distribution with exponent α

$$p_i = \frac{i^{-\alpha}}{\sum_{j=1}^N j^{-\alpha}} = \frac{i^{-\alpha}}{H_N^{(\alpha)}}$$

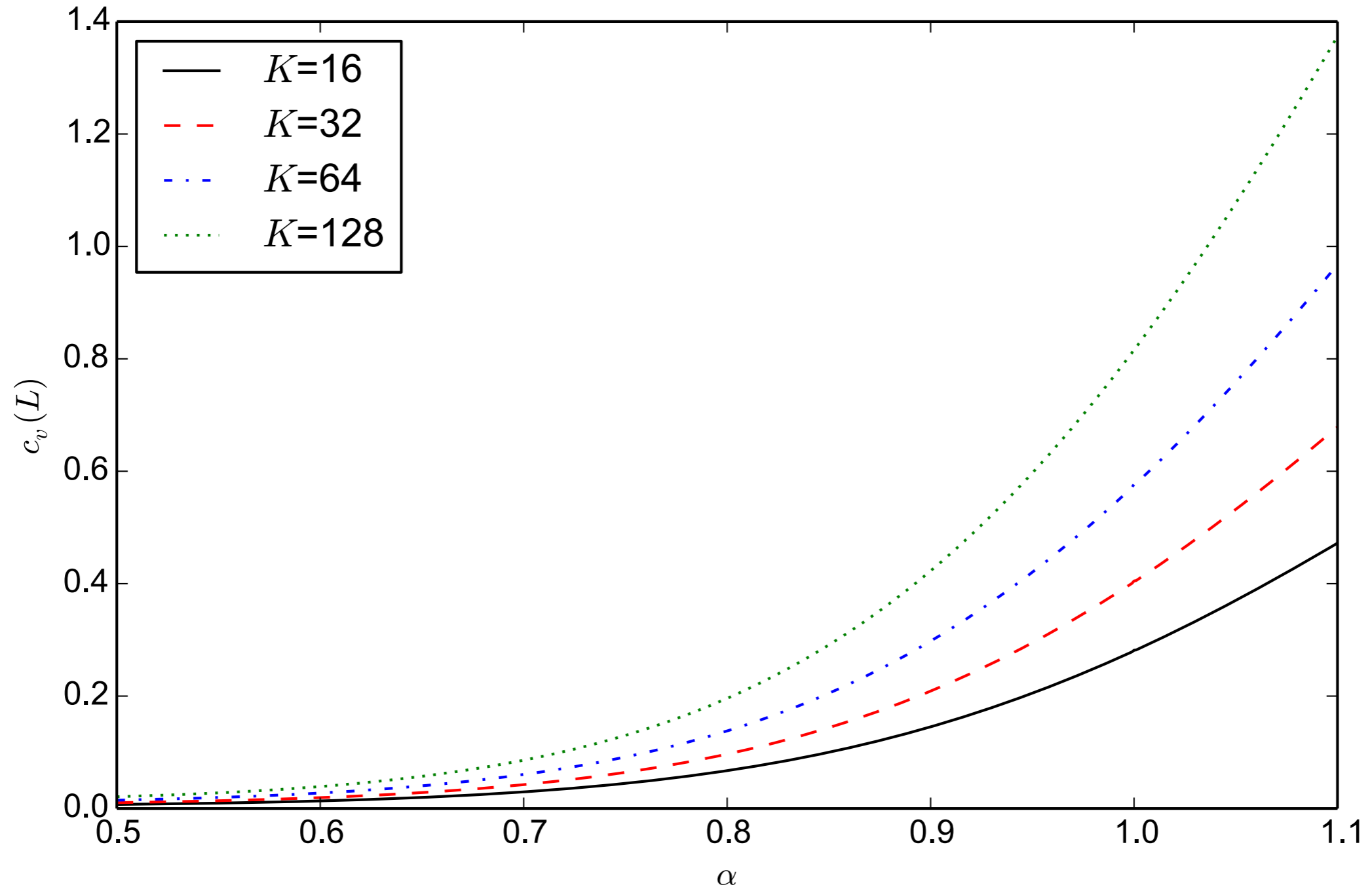
We can derive a closed-form expression for $c_v(L)$ by approximating $H_N^{(\alpha)}$ with its integral expression evaluated in $[1, N+1]$

$$\sum_{i=1}^N \frac{1}{i^\alpha} = H_N^{(\alpha)} \approx \int_1^{N+1} \frac{dx}{x^\alpha} = \begin{cases} \frac{(N+1)^{1-\alpha} - 1}{1-\alpha}, & \alpha \neq 1 \\ \log(N+1), & \alpha = 1 \end{cases}$$

Impact of item popularity distribution

$$c_v(L) \approx \begin{cases} \sqrt{K-1} \frac{\sqrt{(N+1)^{1-2\alpha} - 1} \cdot (1-\alpha)}{\sqrt{1-2\alpha} \cdot [(N+1)^{1-\alpha} - 1]} & \alpha \neq \left\{\frac{1}{2}, 1\right\} \\ \frac{\sqrt{(K-1) \log(N+1)}}{2(\sqrt{N+1} - 1)} & \alpha = \frac{1}{2} \\ \sqrt{\frac{N(K-1)}{(N+1) \log^2(N+1)}} & \alpha = 1 \end{cases}$$

Impact of item popularity distribution



The skewness of item popularity distribution considerably affects load imbalance

Impact of chunking

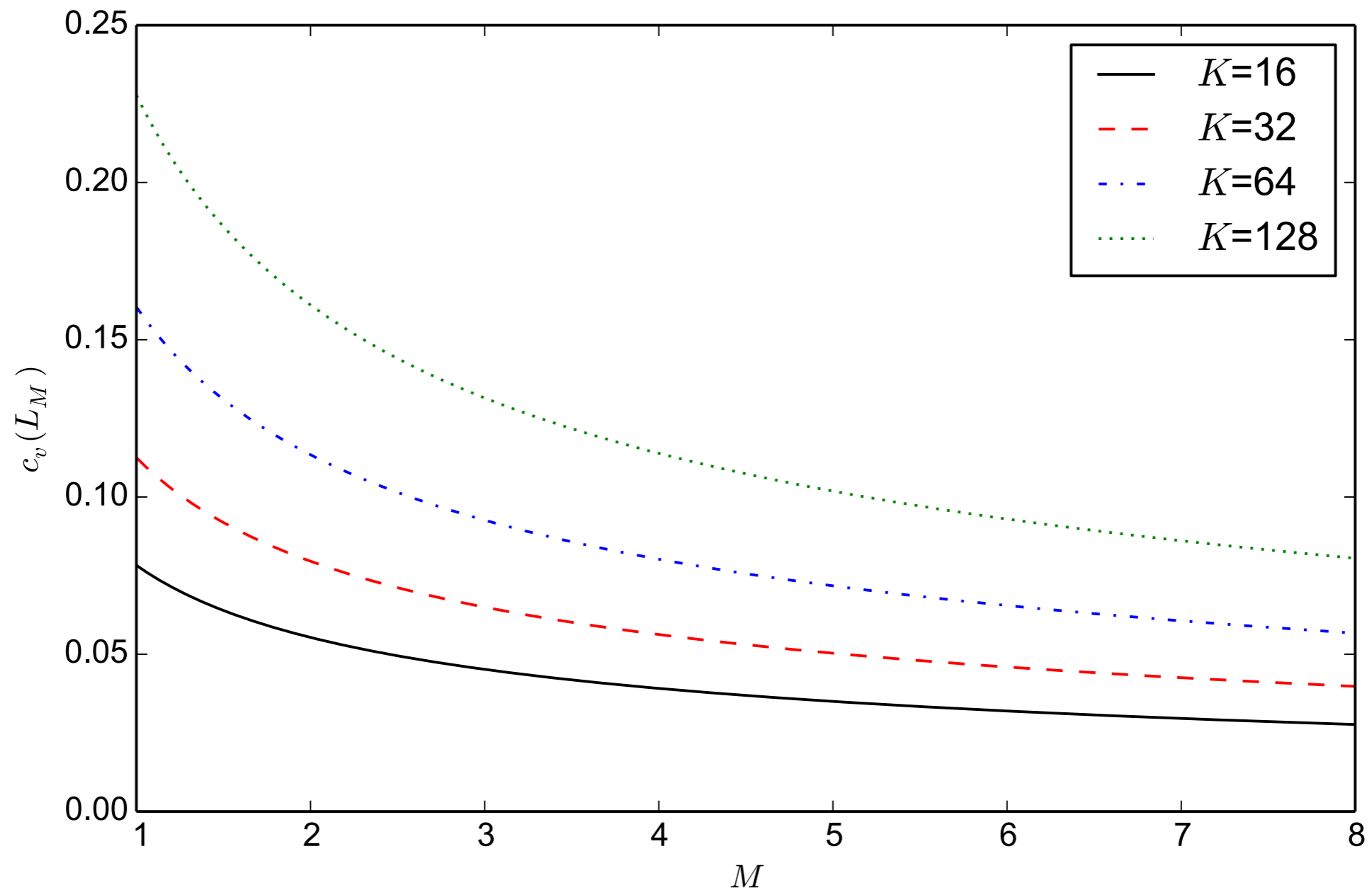
Let's split each item in M chunks and hash them to shards independently

$$c_v(L_M) = \frac{c_v(L)}{\sqrt{M}}$$



Chunking reduces
load imbalance

Impact of chunking



Splitting items in few chunks is sufficient to reap most of the benefits

Impact of heterogeneous item size

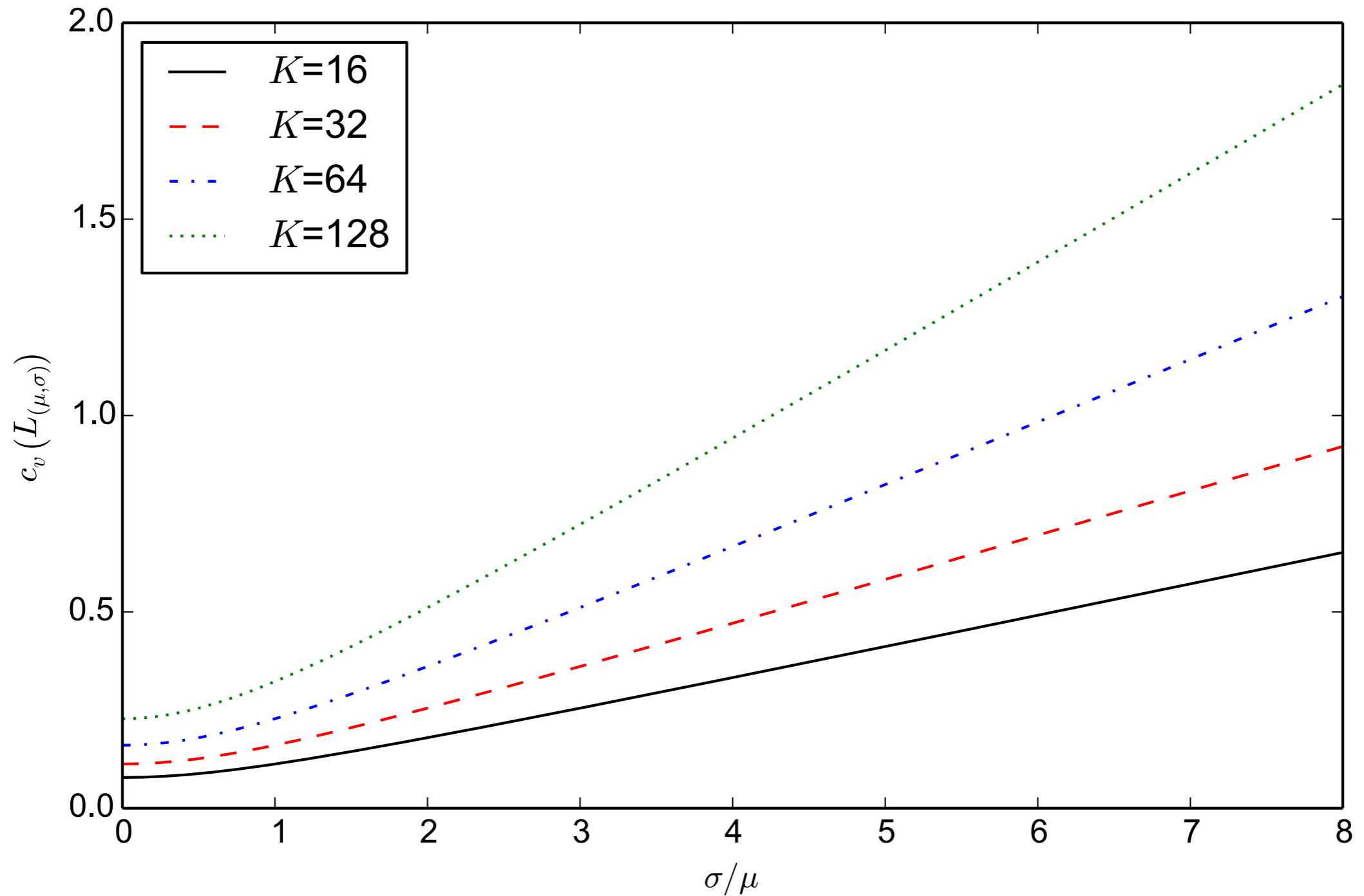
Item size: arbitrary random variable with mean μ and variance σ^2

$$c_v(L_{(\mu,\sigma)}) = \sqrt{K} \sqrt{\left(1 - \frac{1}{K}\right) + \frac{\sigma^2}{\mu^2}} \sqrt{\sum_{i=1}^N p_i^2}$$

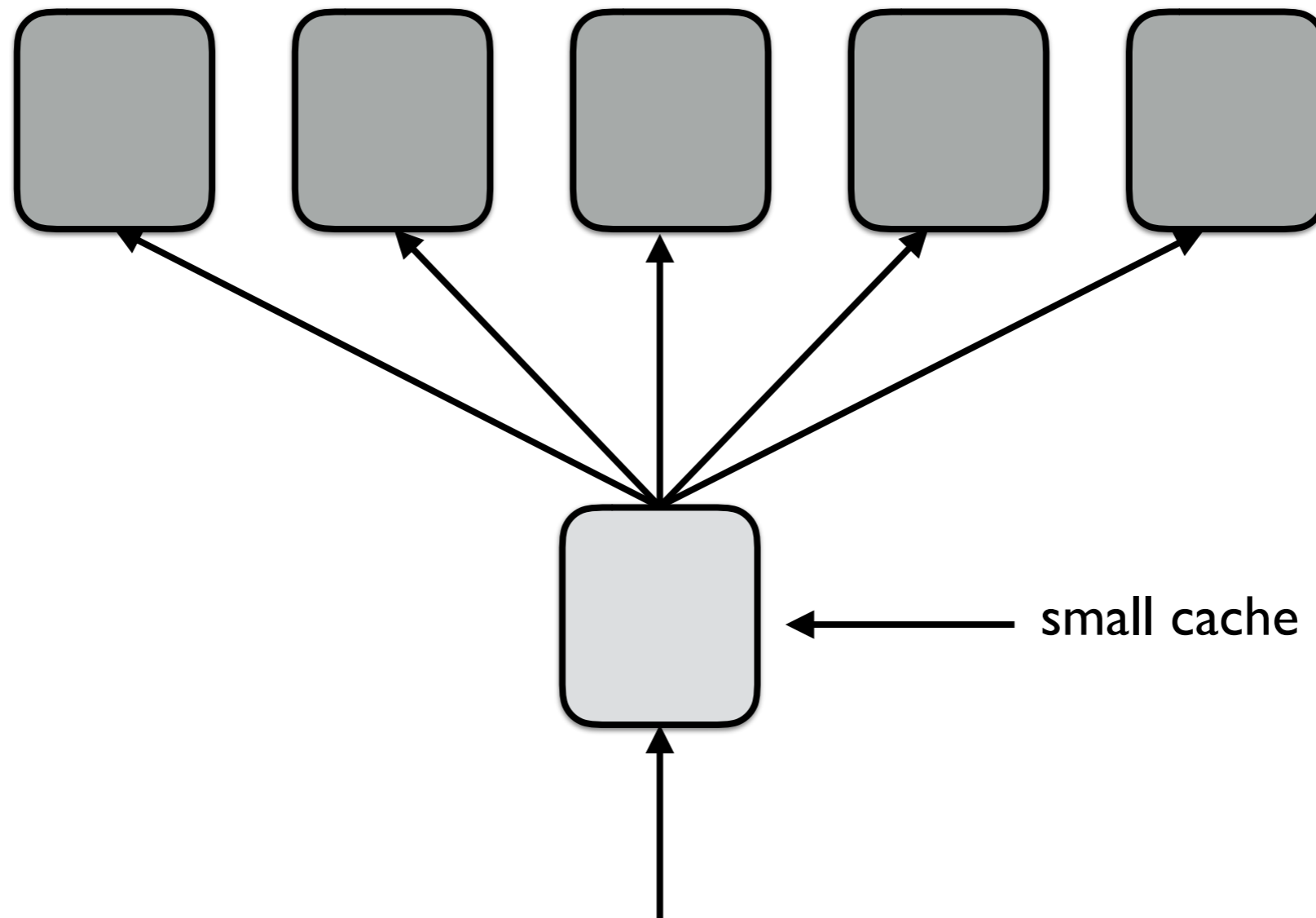


Load imbalance proportional to σ/μ

Impact of heterogeneous item size

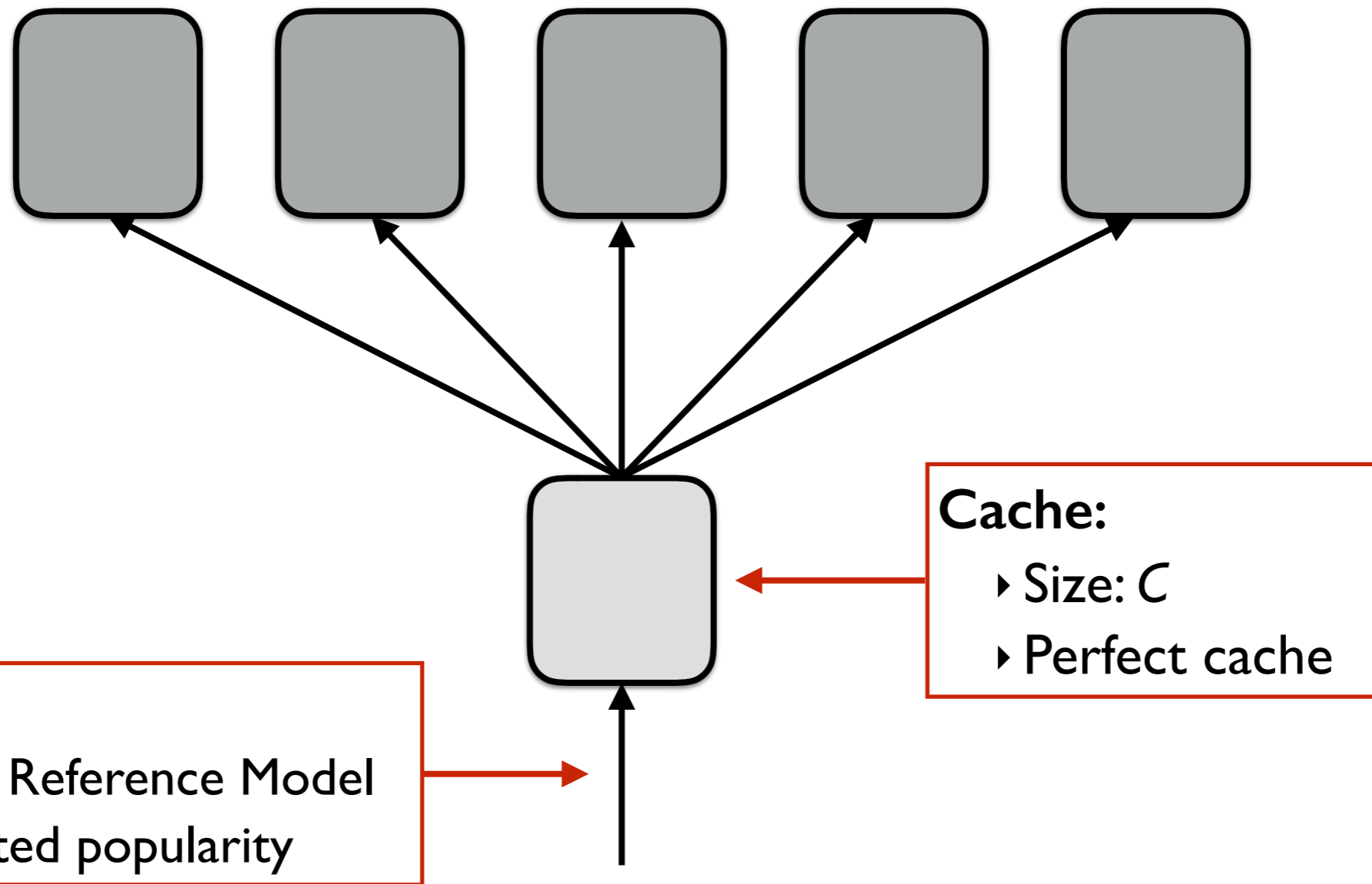


Impact of frontend caching



Would placing a frontend cache in front a sharded system reduce load imbalance?

Impact of frontend caching



Impact of frontend caching

Load imbalance is convex with respect to C , with a global minimum at C^*

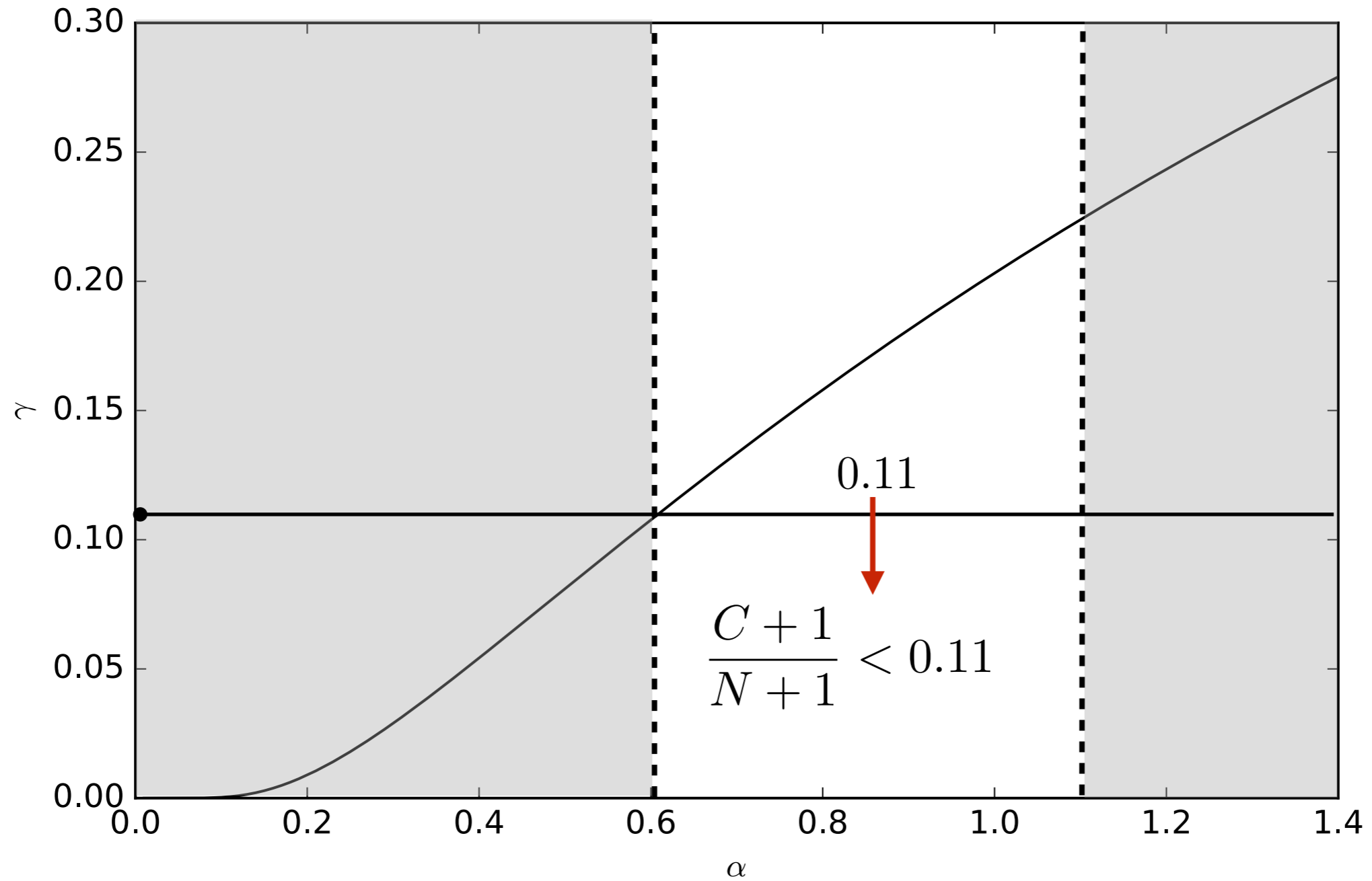
$$\frac{C^* + 1}{N + 1} = \gamma = f(\alpha)$$

Observations:

- ▶ A frontend cache reduced load imbalance as long as $C < C^*$
- ▶ The point of minimum load imbalance does not depend on the absolute value of C but on the ratio between C and N
- ▶ The parameter γ is exclusively a function of Zipf exponent α

How small in practice can a frontend cache be to reduce load imbalance?

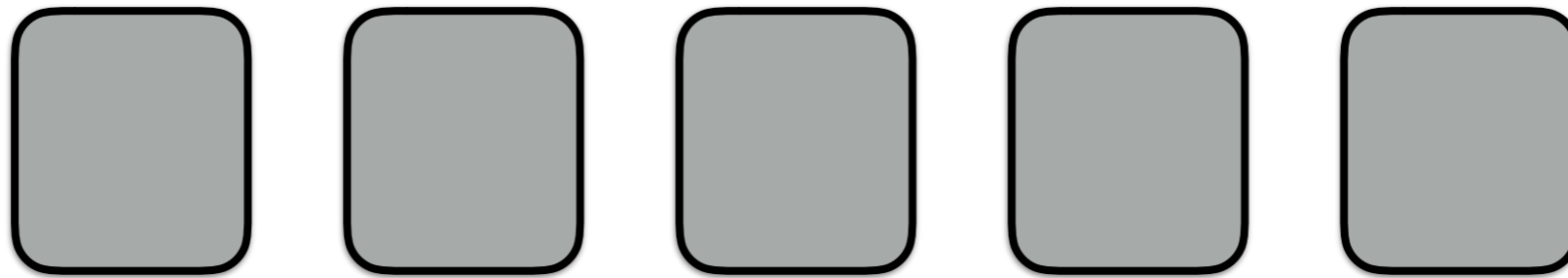
Impact of frontend caching



Any standard frontend cache reduces load imbalance

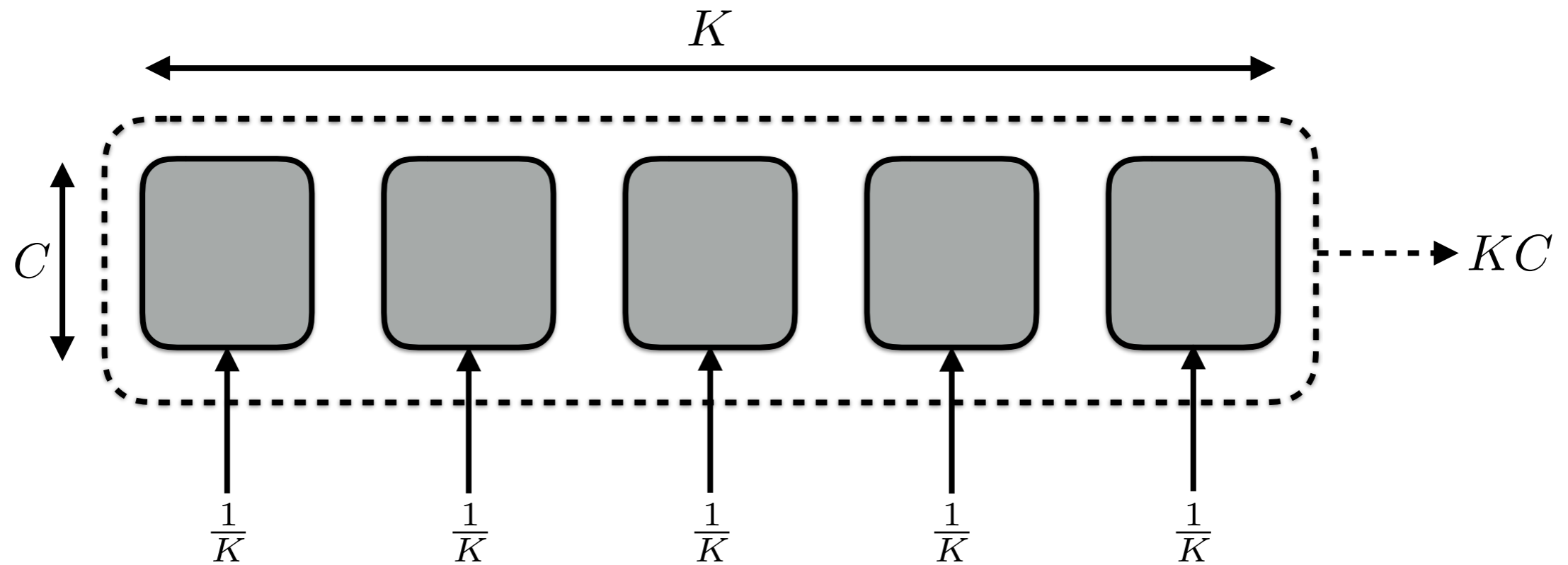
Caching performance

Problem statement



What is the cache hit ratio yielded by a sharded caching system compared to a single cache as large as the whole system?

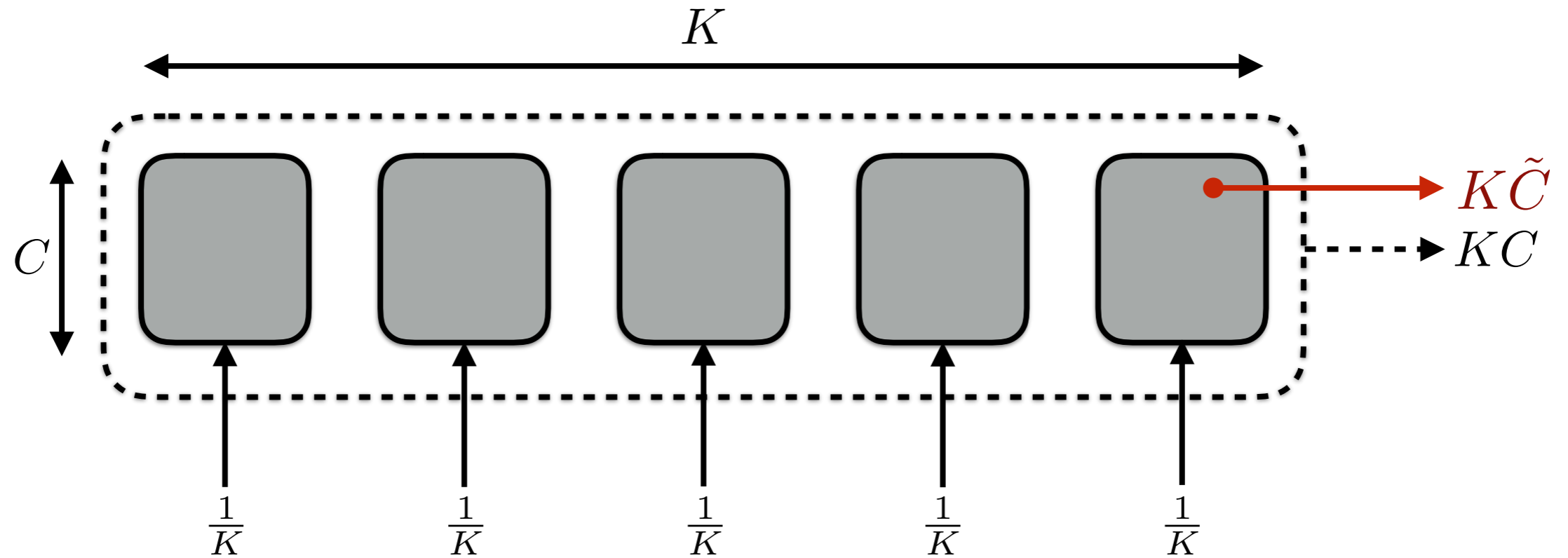
Model and assumptions



Assumptions

- ▶ Demand satisfies Independent Reference Model (stationary and finite catalog)
- ▶ All caches operate under the same policy
- ▶ Replacement policy is defined by its characteristic time (e.g. LRU, FIFO, LFU)
[Che *et al.*, IEEE JSAC], [Martina *et al.*, IEEE INFOCOM'14]

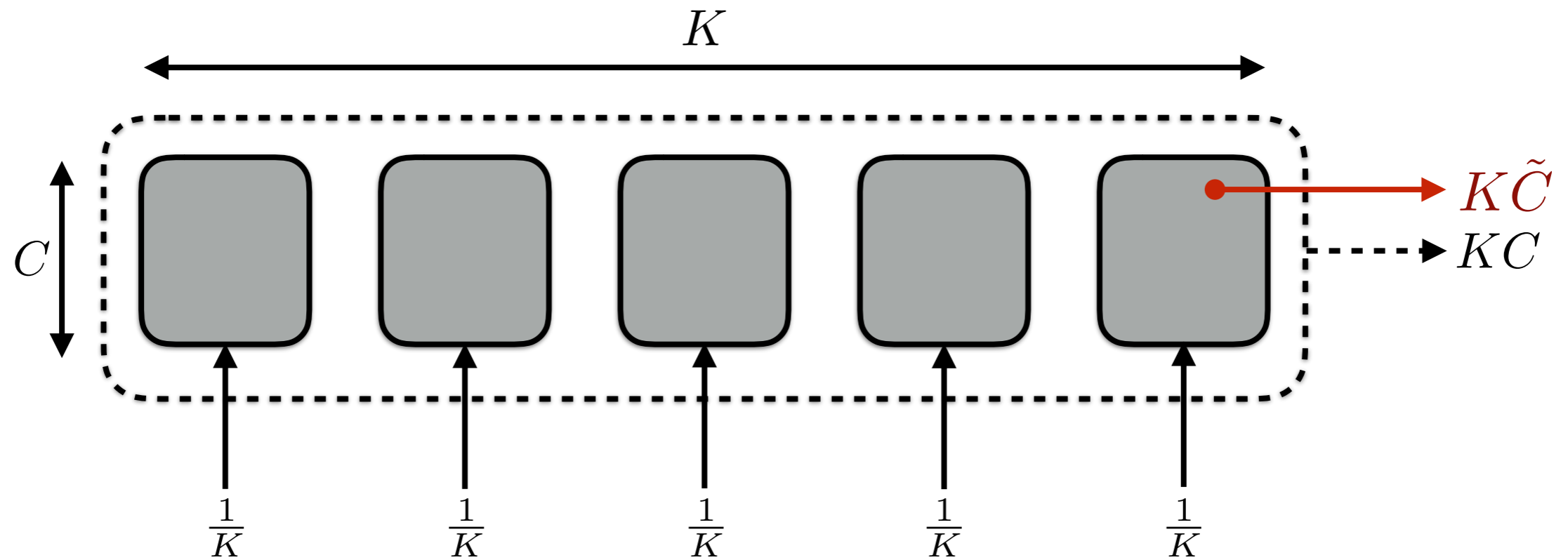
Main findings



$$P\left(|\tilde{C} - C| \geq \epsilon C\right) \leq \left(1 - \frac{1}{K}\right) \frac{1}{\epsilon^2 C}$$

Fluctuations decrease
as C increases

Main findings

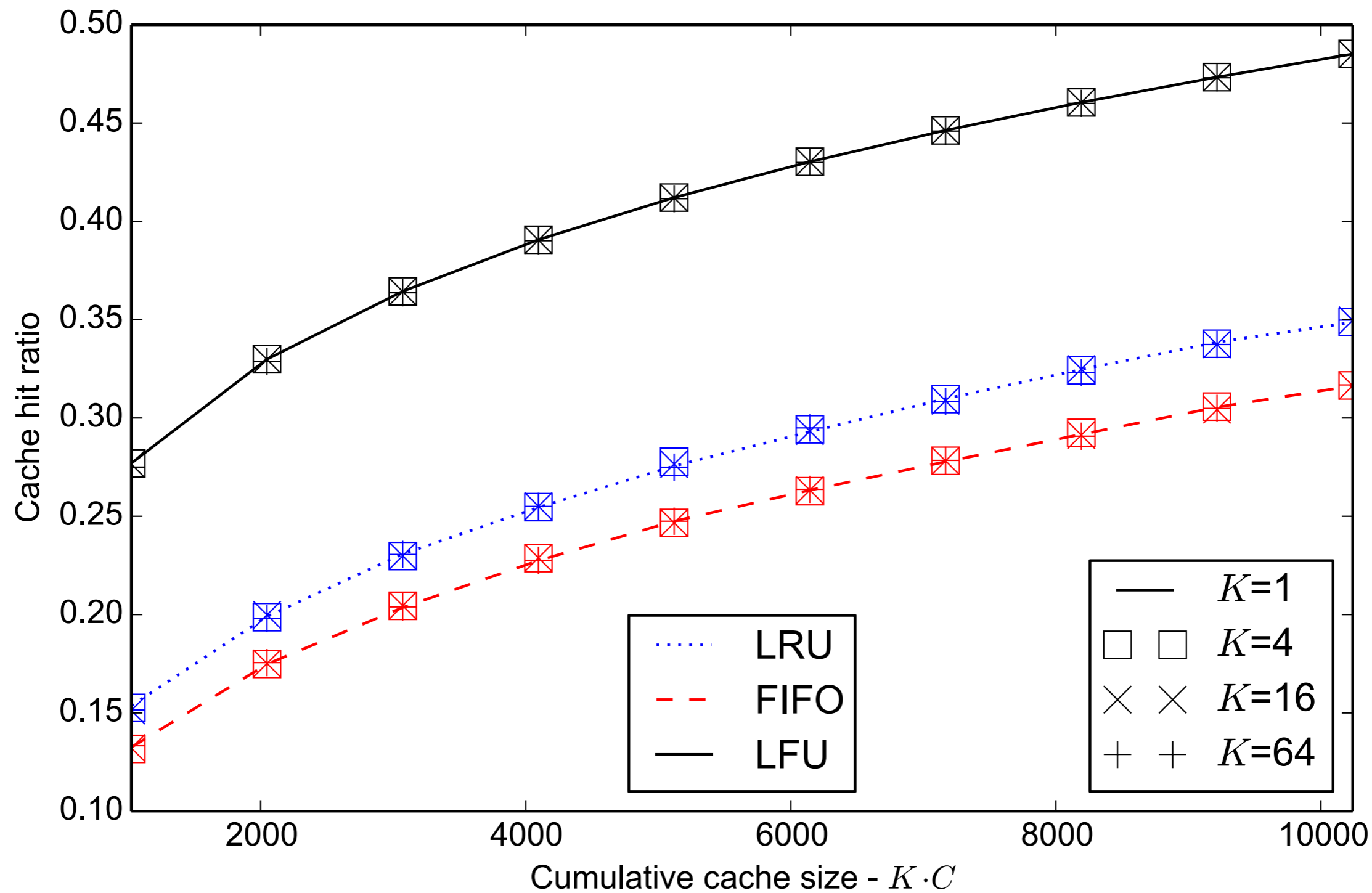


$$P\left(|\tilde{C} - C| \geq \epsilon C\right) \leq \left(1 - \frac{1}{K}\right) \frac{1}{\epsilon^2 C}$$

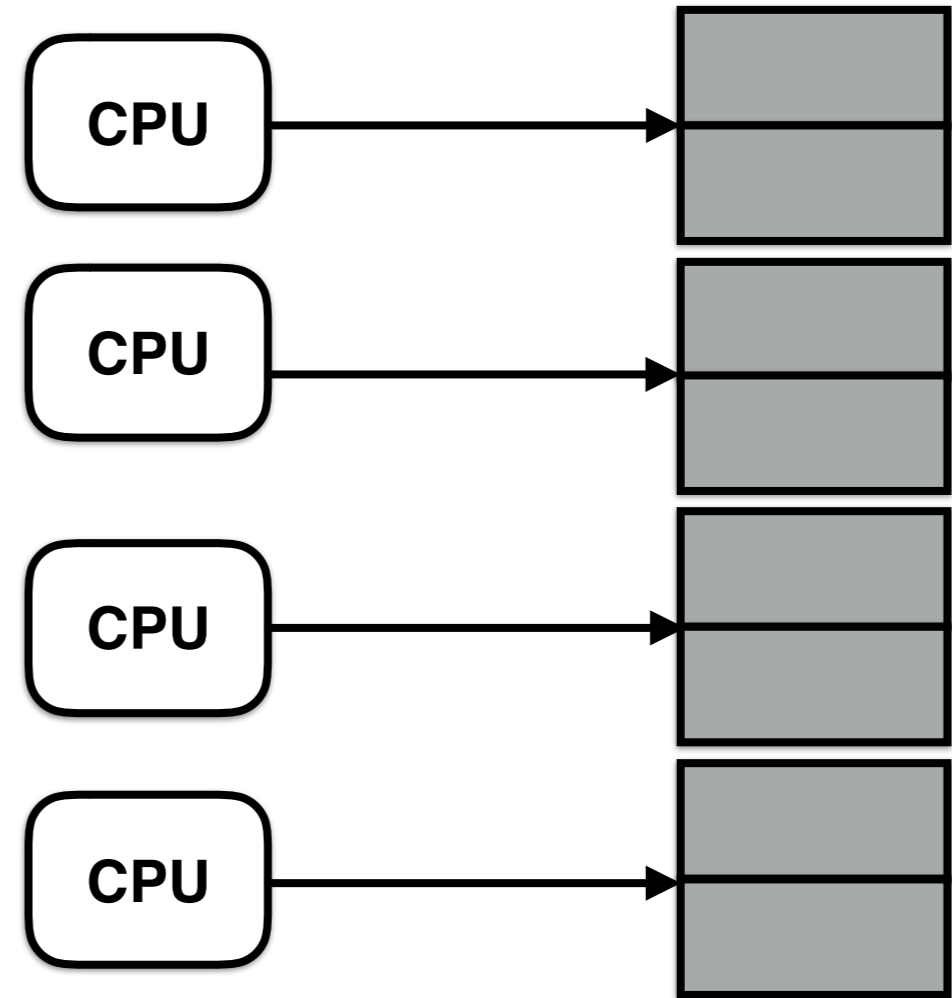
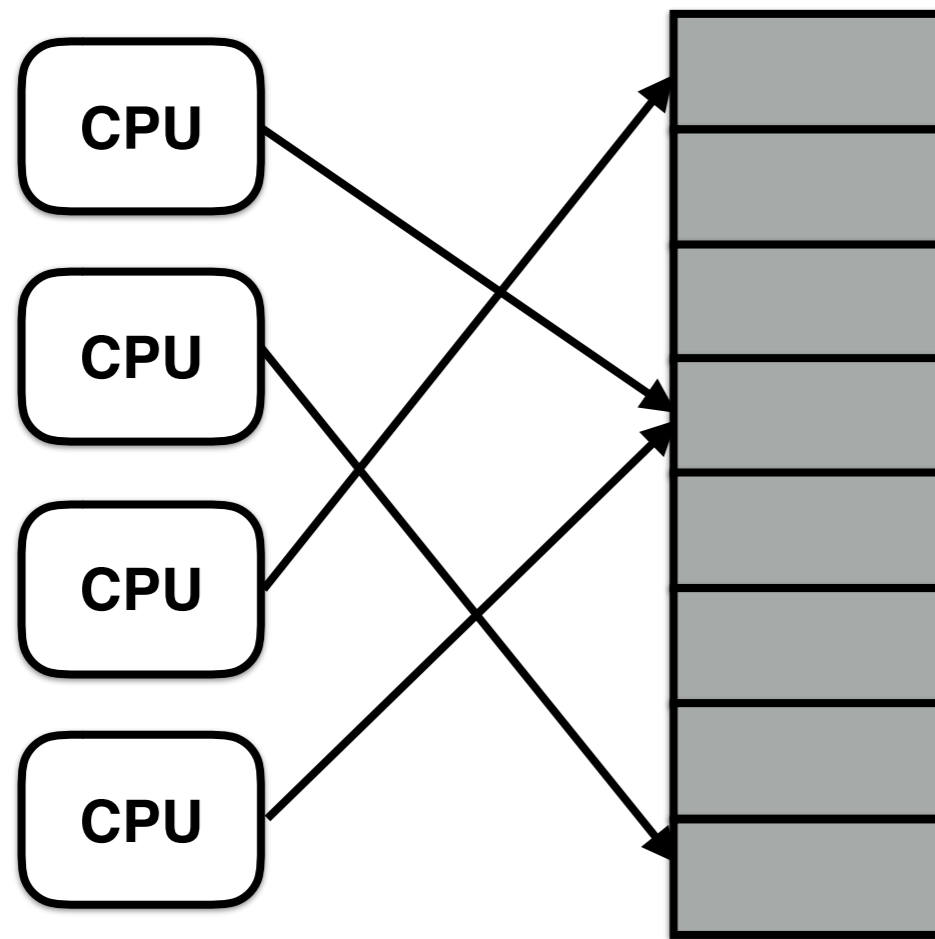
Web caching example

- ▶ The average size of a Web object is 24KB [HTTP archive]
- ▶ A caching shard with 24GB of storage we can store 10^6 items.
- ▶ The probability of having an error greater than $0.01 \times C$ is $< 1\%$

Validation



Why this result is important



Summary and conclusions

Load balancing

- ▶ Skewed item popularity distribution severely affects load imbalance
- ▶ Chunking and frontend caching are effective solutions

Caching performance

- ▶ Sharded caching systems yield performance identical to a single cache as long as each caching shard is large enough
- ▶ Sharded caching systems can be modeled as a single cache
- ▶ Effective technique for building scalable distributed systems