An Efficient Simulation Algorithm for Cache of Random Replacement Policy

Shuchang Zhou

Zheng Zhou, Sep. 13, 2010

Organization

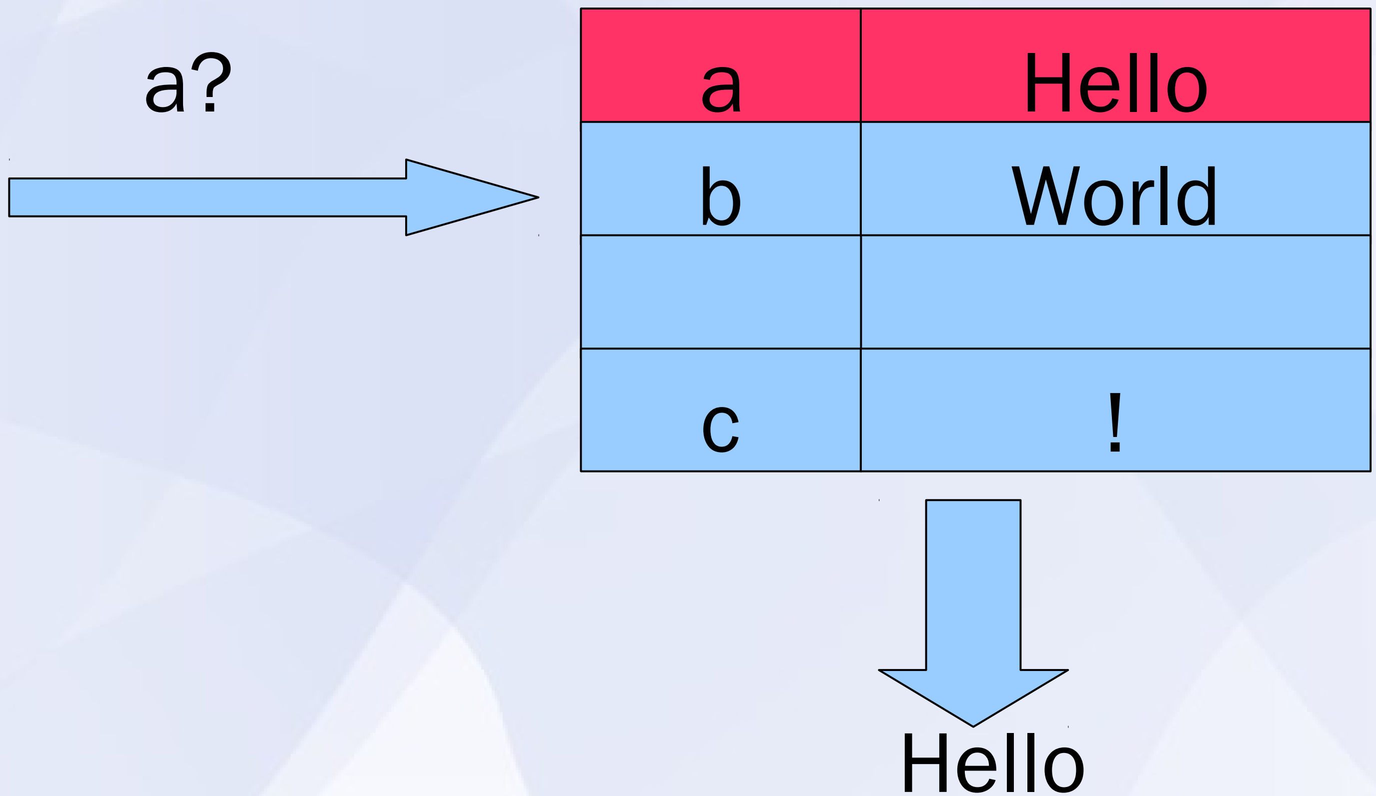
- Background
- Algorithm
- Evaluation

Cache Hit

a?

a	Hello
b	World
c	!

Cache Hit

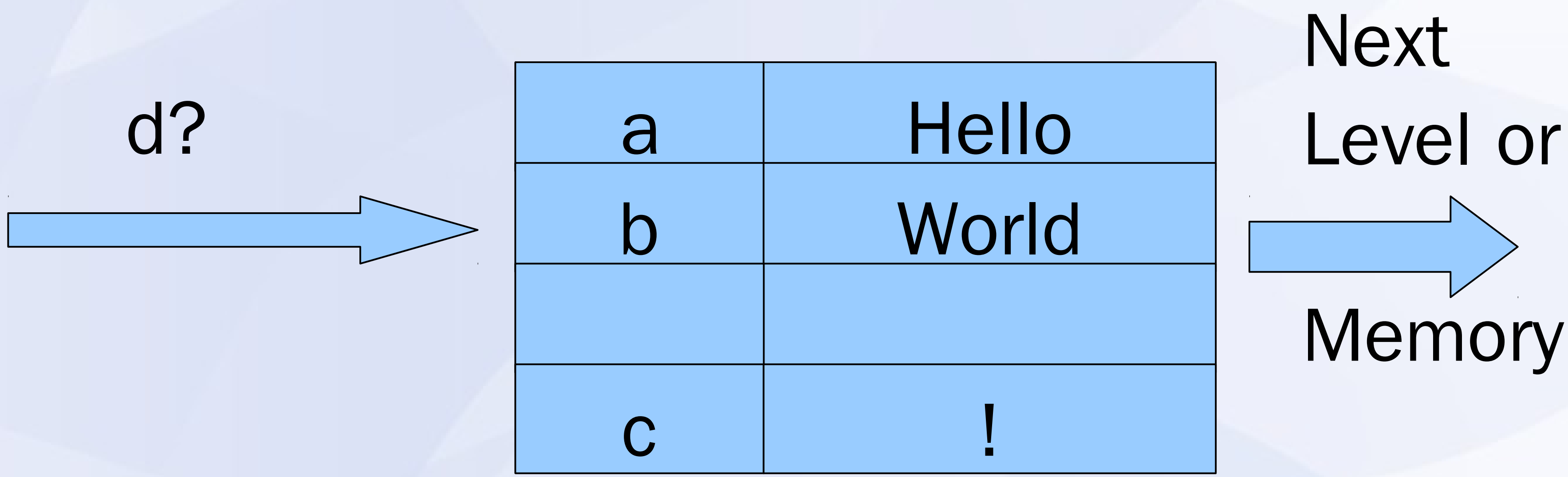


Cache Miss

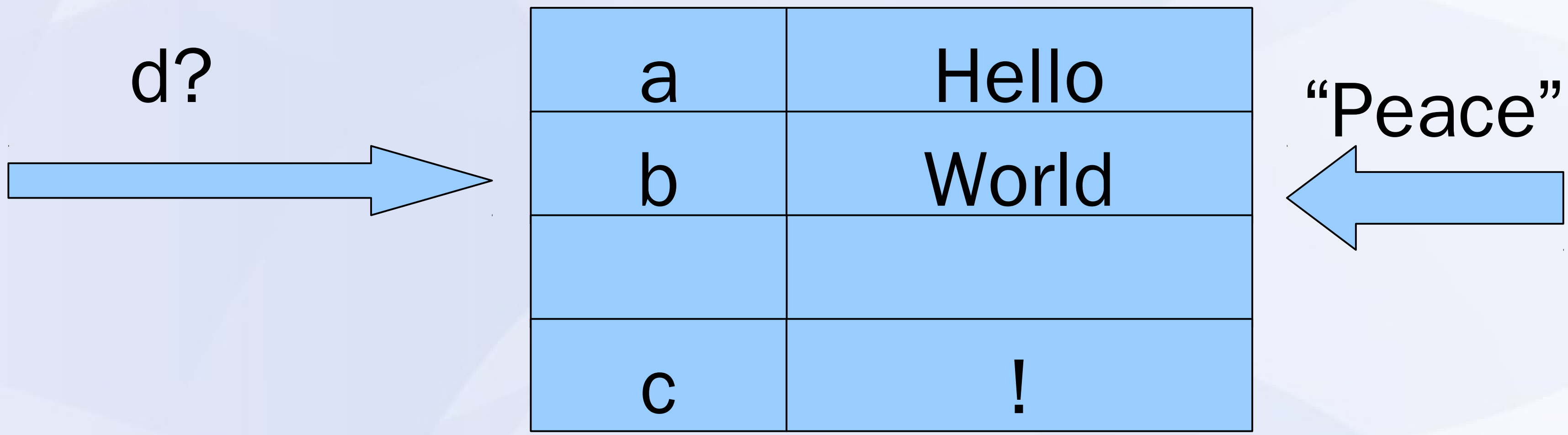
d?

a	Hello
b	World
c	!

Cache Miss



Cache Miss



Where to store the new pair?

Evict the oldest

* Least Recently Used

** found in Intel/AMD

Randomly pick a slot

* Random Replacement

** found in ARM/Loongson

Random Replacement

d	Peace
b	World
c	!

Random Replacement

a	Hello
d	Peace
c	!

Random Replacement

a	Hello
b	World
d	Peace
c	!

Random Replacement

a	Hello
b	World
d	Peace

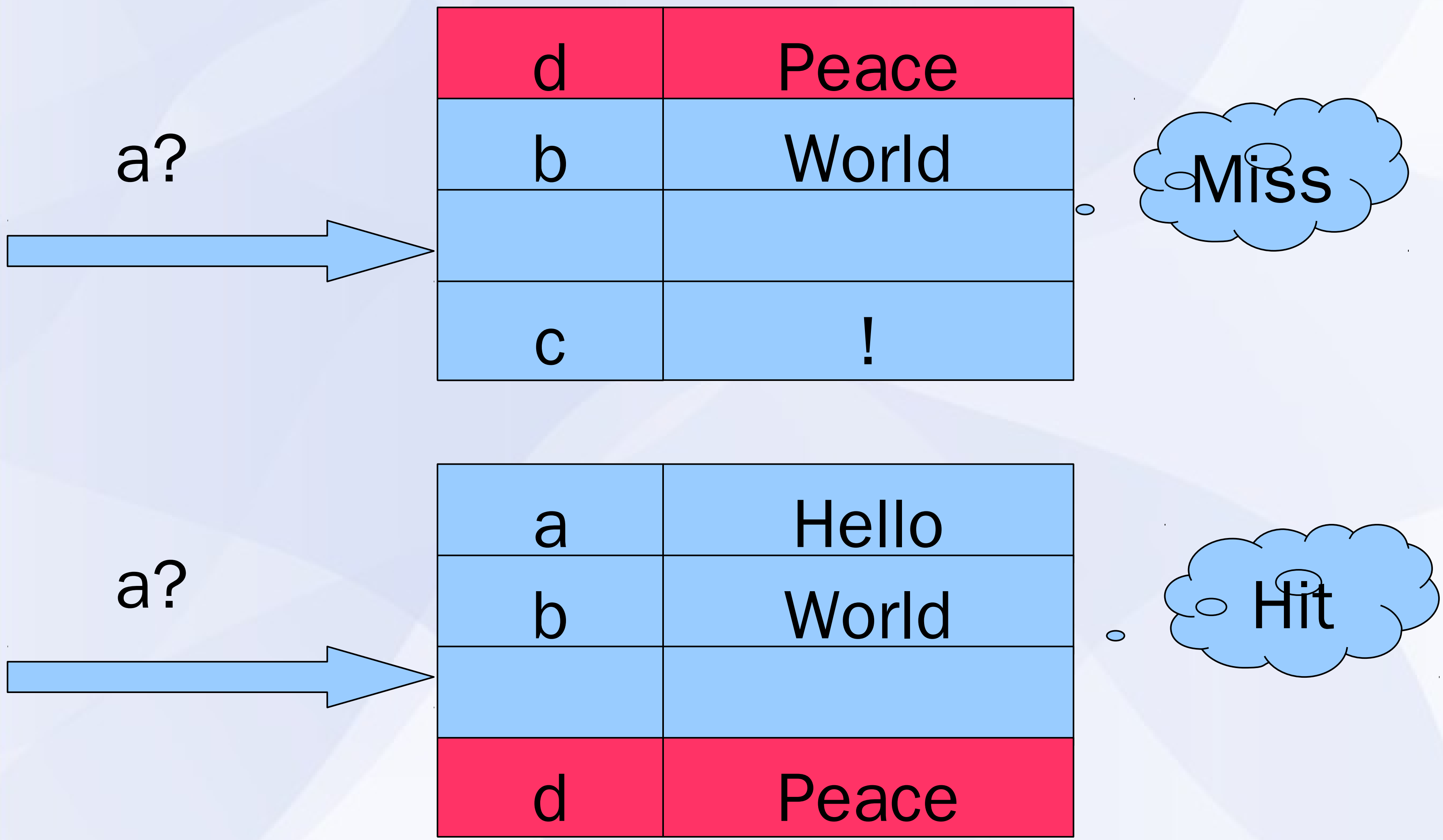
Different Behavior

a?

d	Peace
b	World
c	!

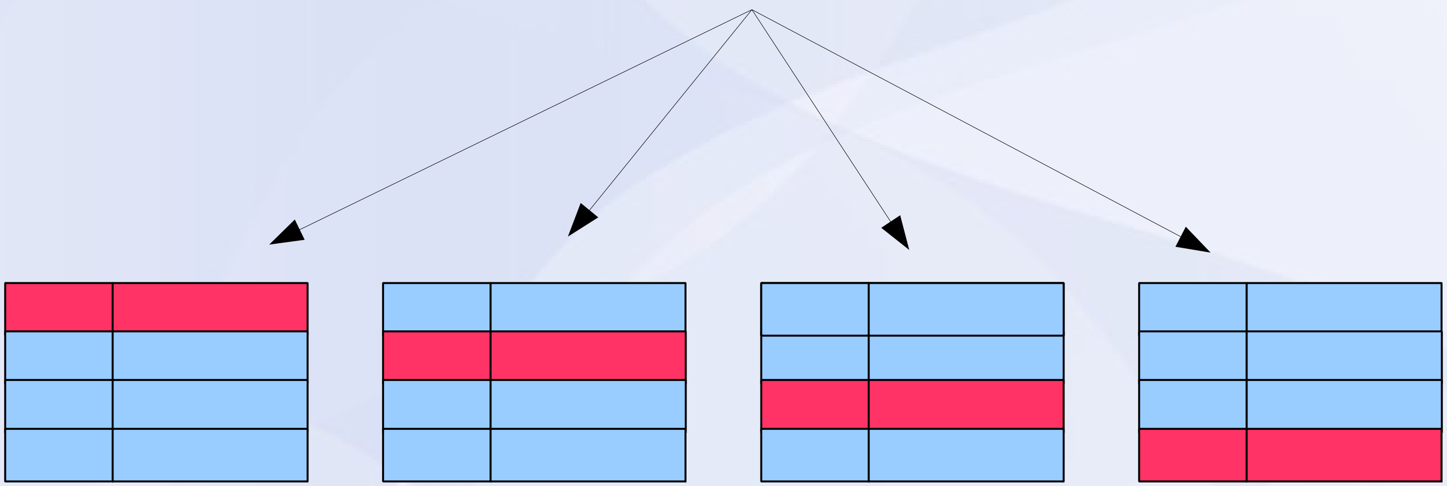
Miss

Different Behavior

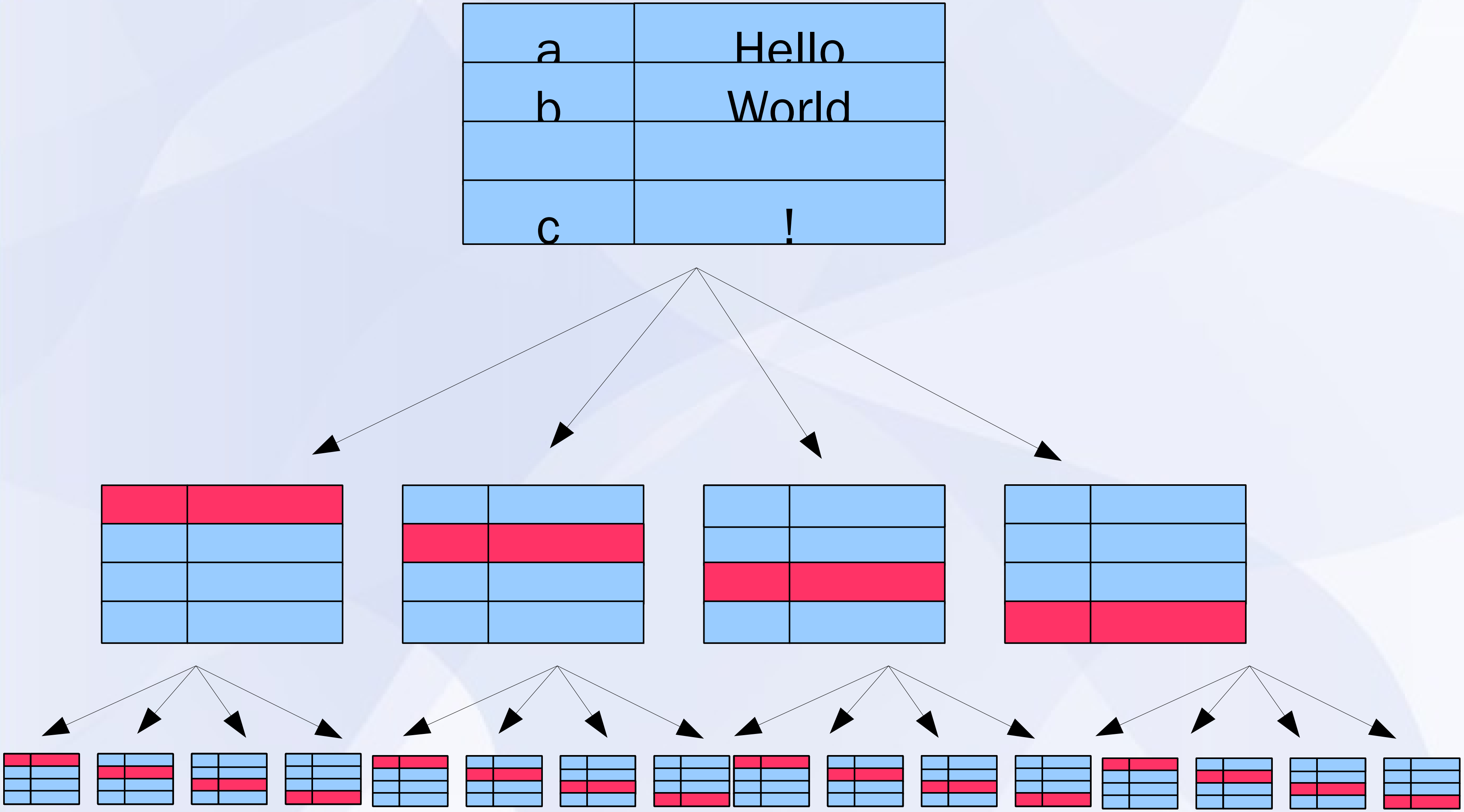


Combinatorial Explosion

a	Hello
b	World
c	!



Combinatorial Explosion



Solution?

Solution?

Recast the problem in probability setting.

Settings

* Let the input be the sequence of cache line index

** a_0, a_1, a_2, … a_n,

* Assume associativity to be M

** there are M candidates for eviction upon a cache miss

Indicator Random Variable

X=1 if an event happens

X=0 if an event *does not* happen

$E(X) = 1*P(X=1) + 0*P(X=0) = P(X=1)$

The expectation equals the probability of an event.

Indicator Random Variable

Let X_i indicate the miss event for a_i

$$a_0, a_1, a_2, \ldots a_n$$

↓

$$X_0, X_1, X_2, \ldots X_n$$

Reuse window

Reuse window

$Z_i = \infty$ if a_i never occurs before

Z_i = sum of X_i in the reuse window

Key Observations

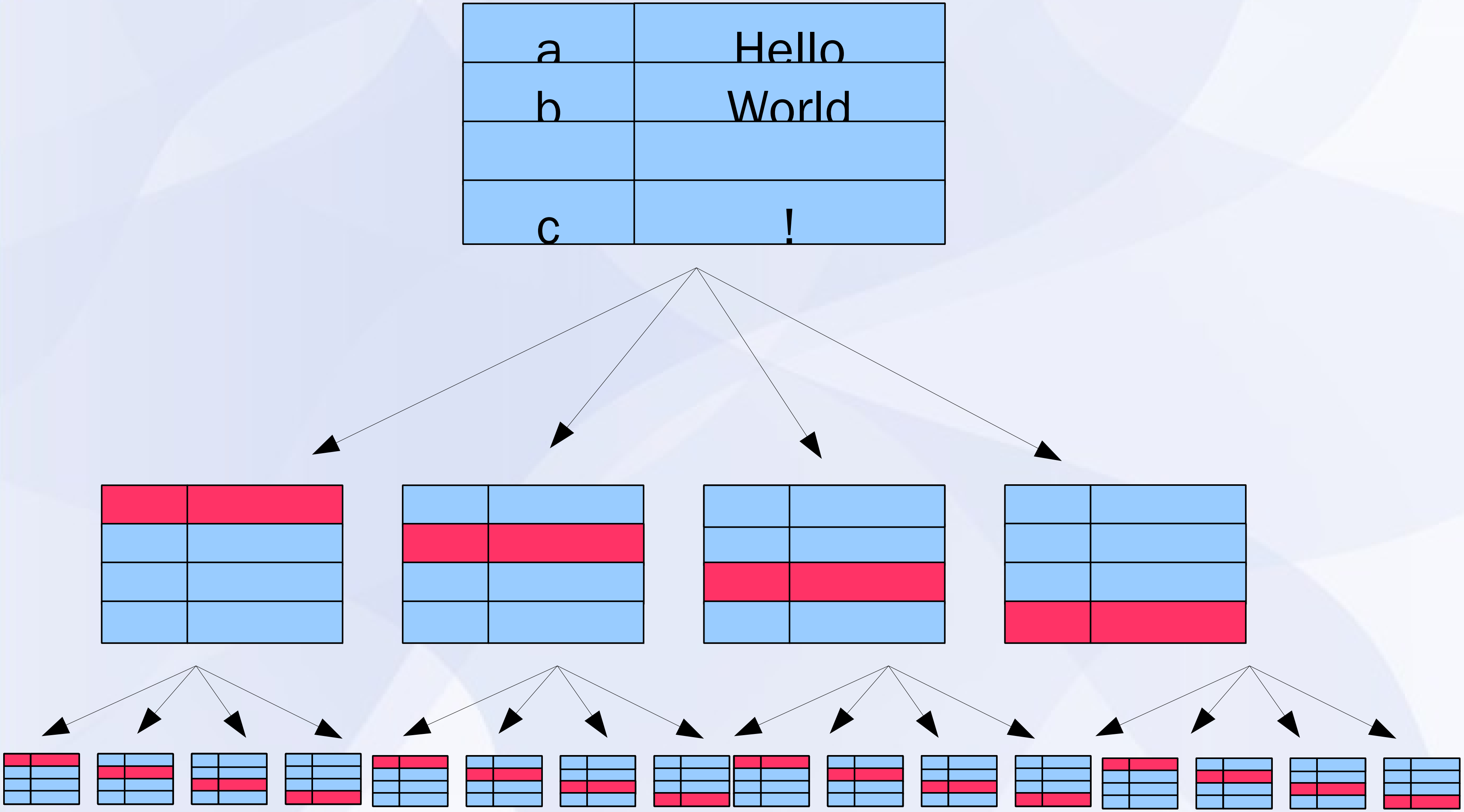
A cache line is definitely in cache *after the access*

Key Observations

A cache line is definitely in cache *after the access*

Every cache miss will have a 1/M probability of evicting a cache line in the same cache set.

Combinatorial Explosion



Key Observations

A cache line is definitely in cache *after the access*

Every cache miss will have a 1/M probability of evicting a cache line in the same cache set.

A cache line will be in the cache with $(1-1/M)^Z$ probability, if there were Z misses in the reuse window.

Foundation

$$E(X_i|Z_i) = 1 - (1 - 1/M)^{Z_i}$$

Foundation

$$E(X_i|Z_i) = 1 - (1 - 1/M)^{Z_i}$$

$$E(X_i) = 1 - E((1 - 1/M)^{Z_i})$$

Foundation

$$E(X_i|Z_i) = 1 - (1 - 1/M)^{Z_i}$$

$$E(X_i) = 1 - E((1 - 1/M)^{Z_i})$$

Impossible to solve directly, for the correlation between consecutive miss events.

Approximation

$$E(X_i) = 1 - E((1 - 1/M)^{Z_i})$$

$$\approx 1 - (1 - 1/M)^{E(Z_i)}$$

See paper for precision of approximation

Reuse window

$Z_i = \infty$ if a_i never occurs before

Z_i = sum of X_i in the reuse window

Reuse window

$EZ_i = \infty$ if a_i never occurs before

EZ_i = sum of EX_i in the reuse window

Approximation

We just formulate a circular relation between EX_i and EZ_i

Can solve and get all EX_i, hence knowing the hit/miss probability of each cache reference.

EZ, EX, M=4

a, b, a, c, d, b

EZ

EX

EZ, EX, M=4

a, b, a, c, d, b

EZ ∞

EX

EZ, EX, M=4

a, b, a, c, d, b

EZ ∞

EX 1

EZ, EX, M=4

a, b, a, c, d, b

EZ ∞ ∞

EX 1 1

EZ, EX, M=4

a, b, a, c, d, b

EZ ∞ ∞ 1

EX 1 1

EZ, EX, M=4

a, b, a, c, d, b

EZ ∞ ∞ 1

EX 1 1 ¼

EZ, EX, M=4

a, b, a, c, d, b

EZ ∞ ∞ 1 ∞ ∞ 2.25

EX 1 1 ¼ 1 1 0.48

EZ, EX, M=4

a, b, a, c, d, b

EZ	∞	∞	1	∞	∞	2.25
EX	1	1	¼	1	1	0.48

#Total misses = sum of EX

≈ 4.73

Efficient Implementation

Problem: Upon each cache miss, all EZ_i need be updated.

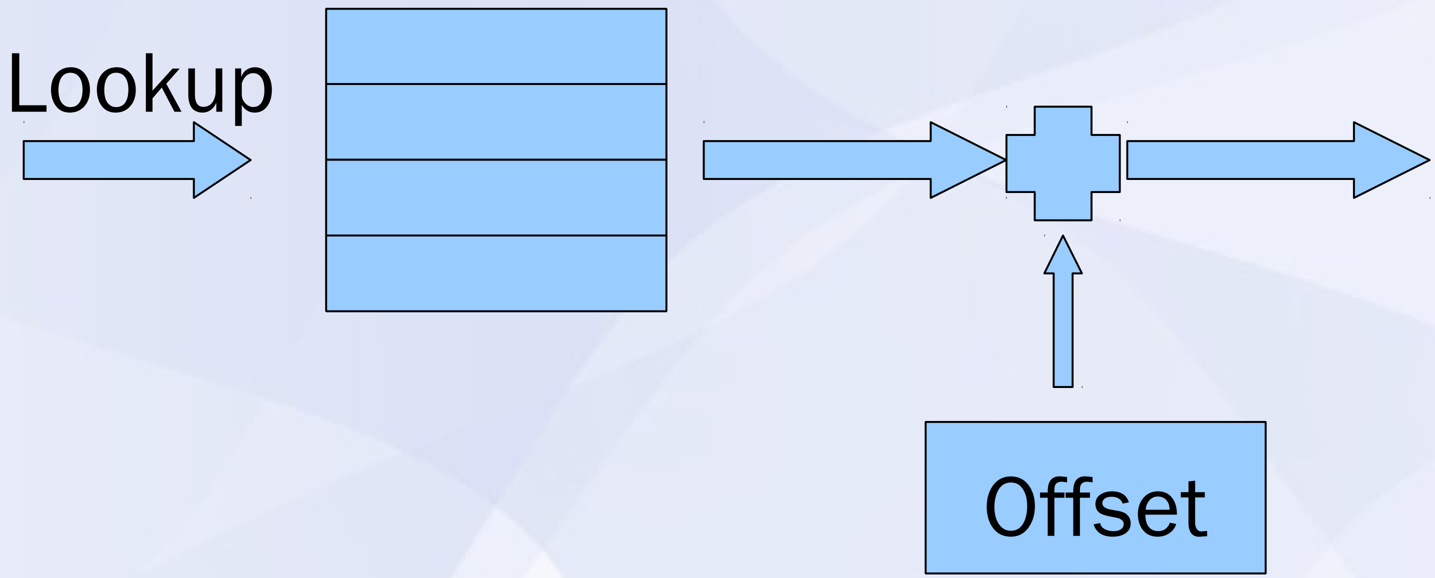
Efficient Implementation

Problem: Upon each cache miss, all EZ_i need be updated.

* But almost all incremented by the same value: EX_i , except the one corresponding to a_i is set to 0

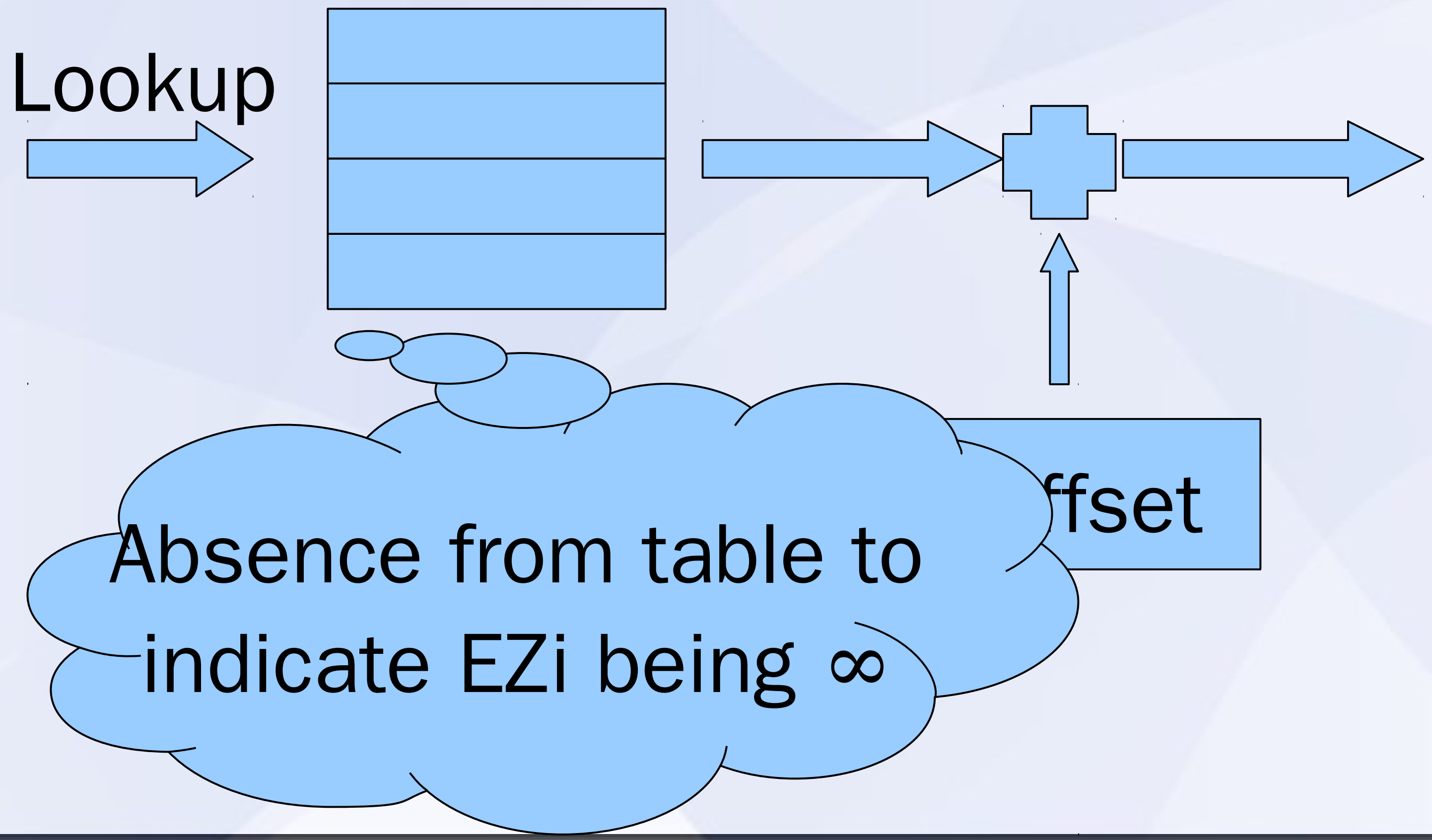
Efficient Implementation

Use a hash map with an offset to store EZ_i



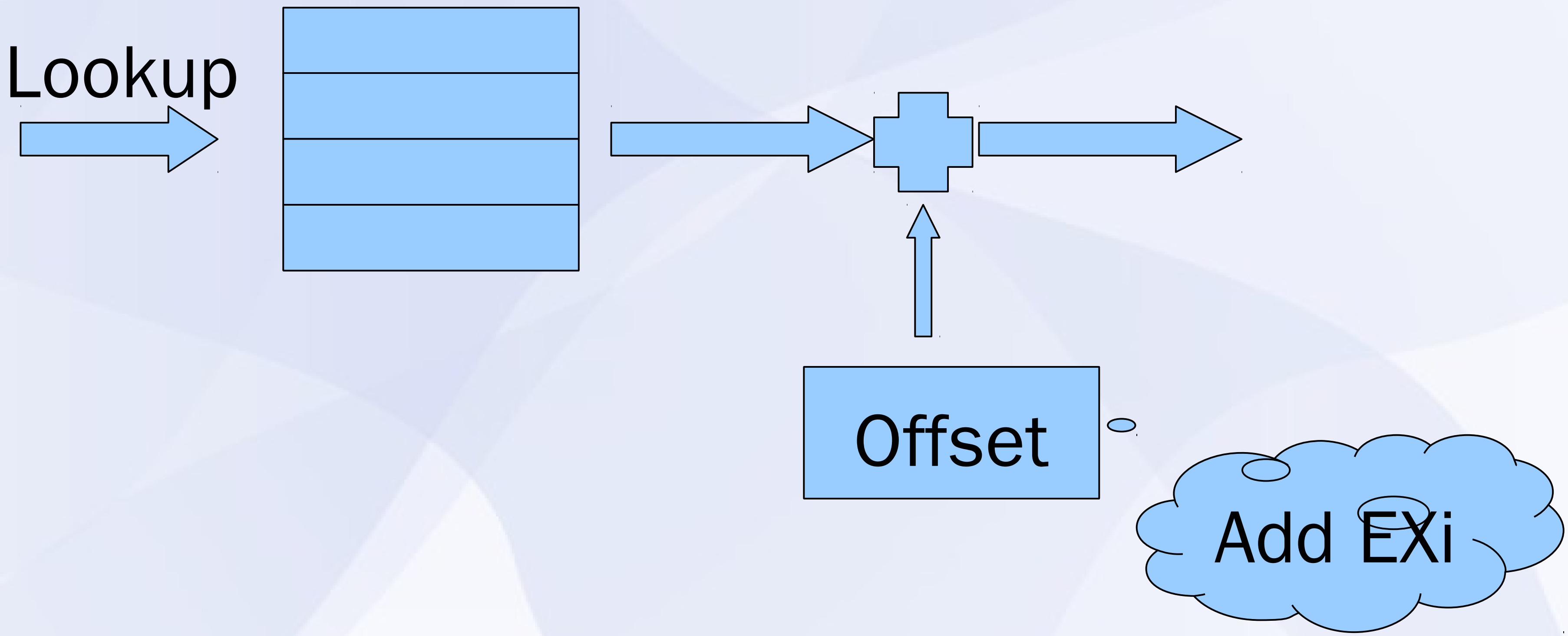
Efficient Implementation

Use a hash map with an offset to store EZ



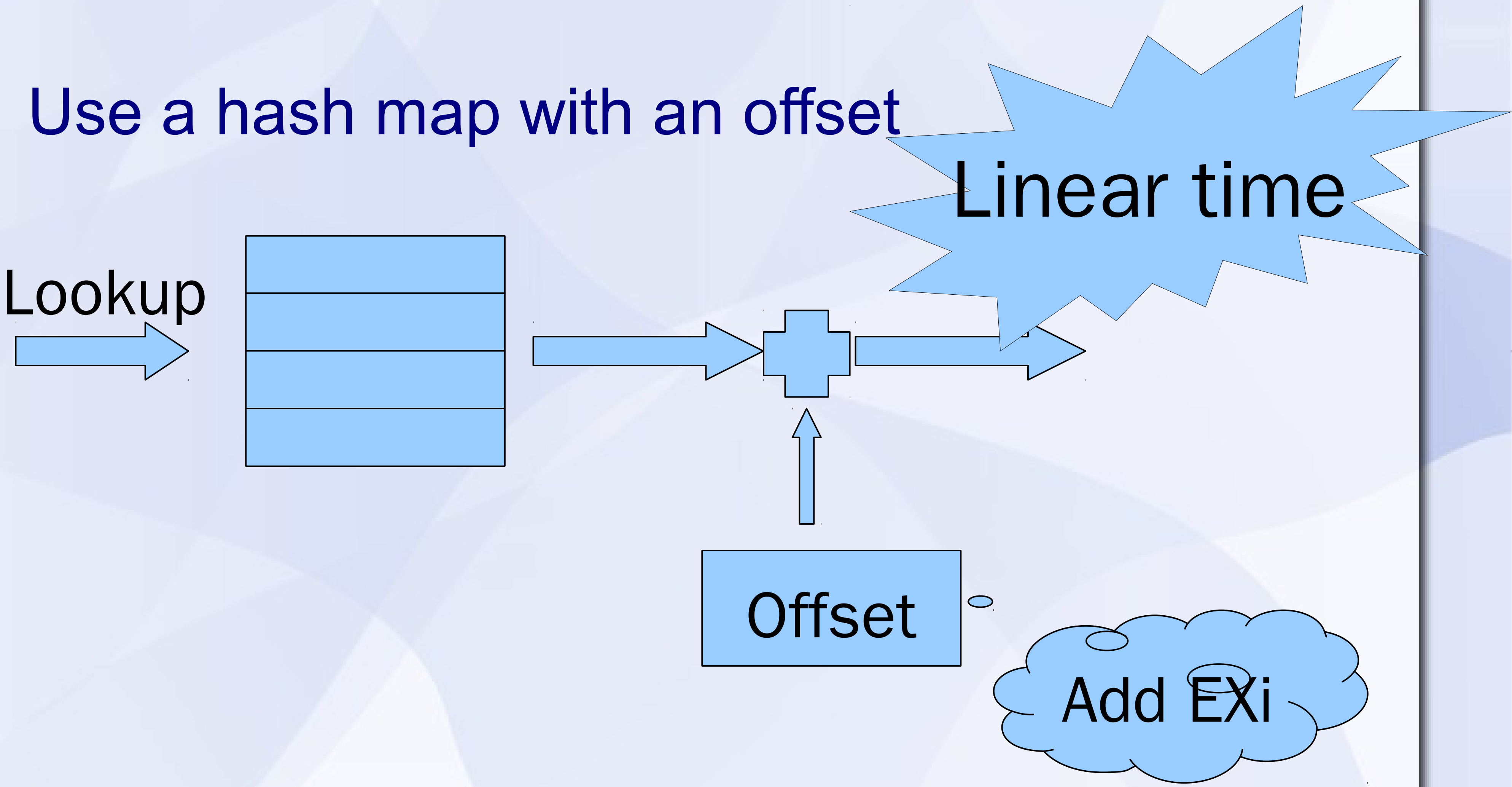
Efficient Implementation

Use a hash map with an offset



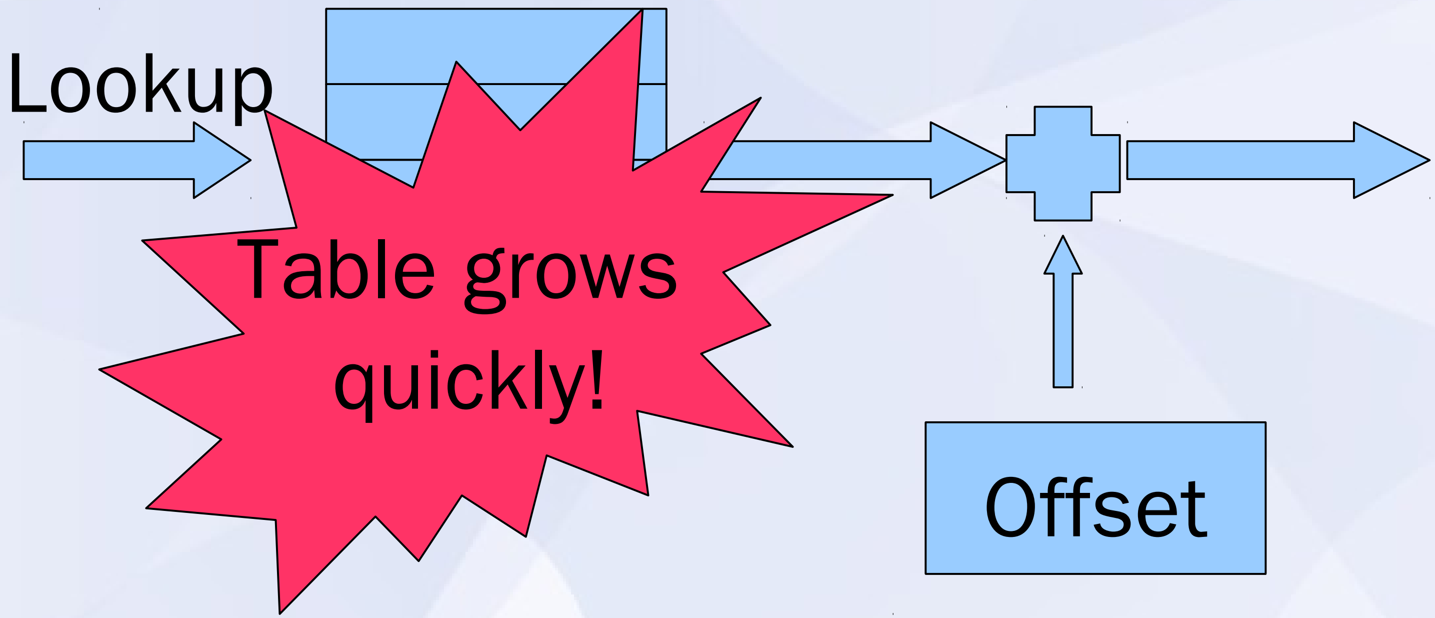
Efficient Implementation

Use a hash map with an offset



Control Space

Use a hash map with an offset



Another Approximation

Tolerate a little absolute error of the hit probability $1 - E(X_i)$

Another Approximation

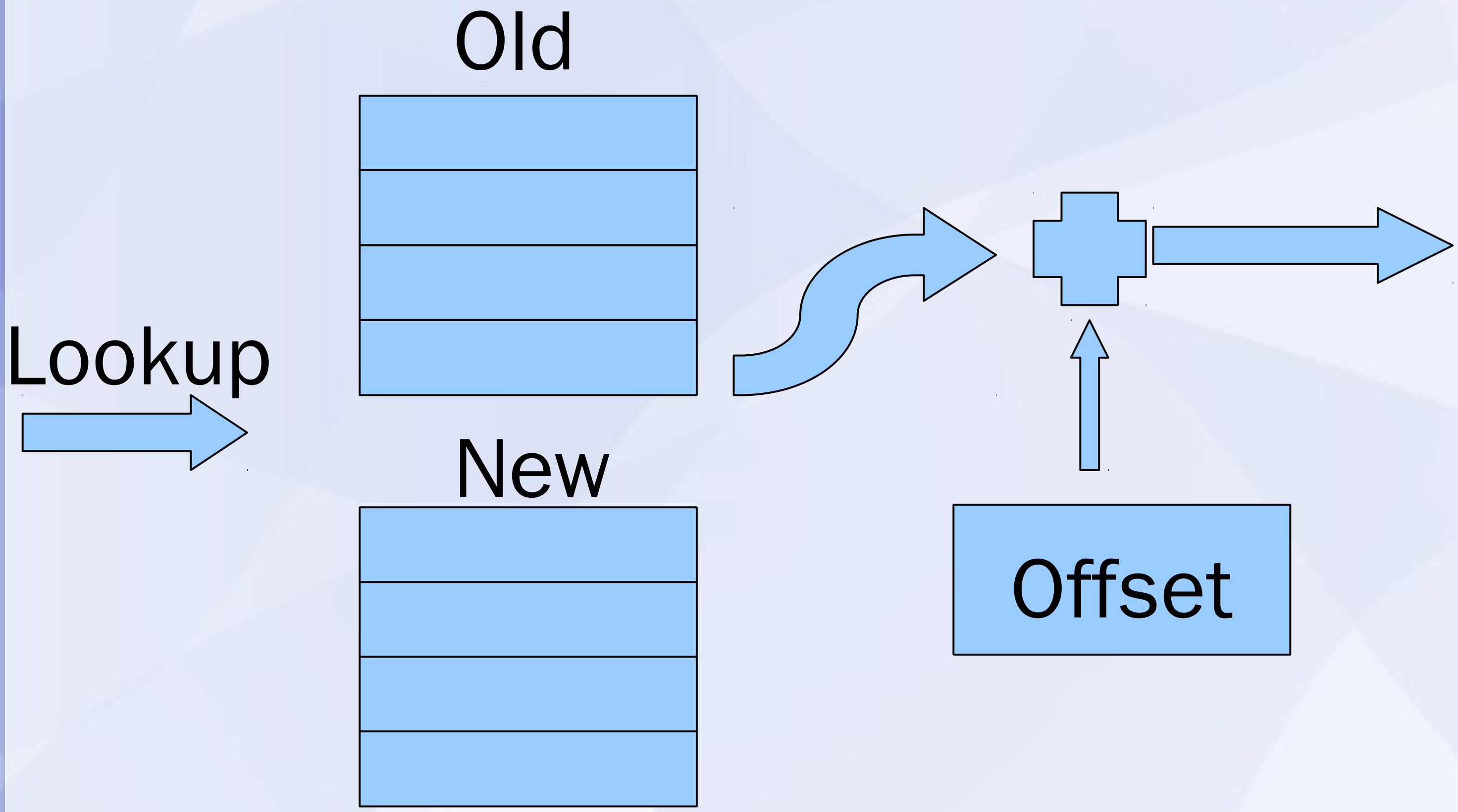
Tolerate a little absolute error of the hit probability 1 − E(Xi)

Then can set large EZi to ∞

* The same as removing from the table

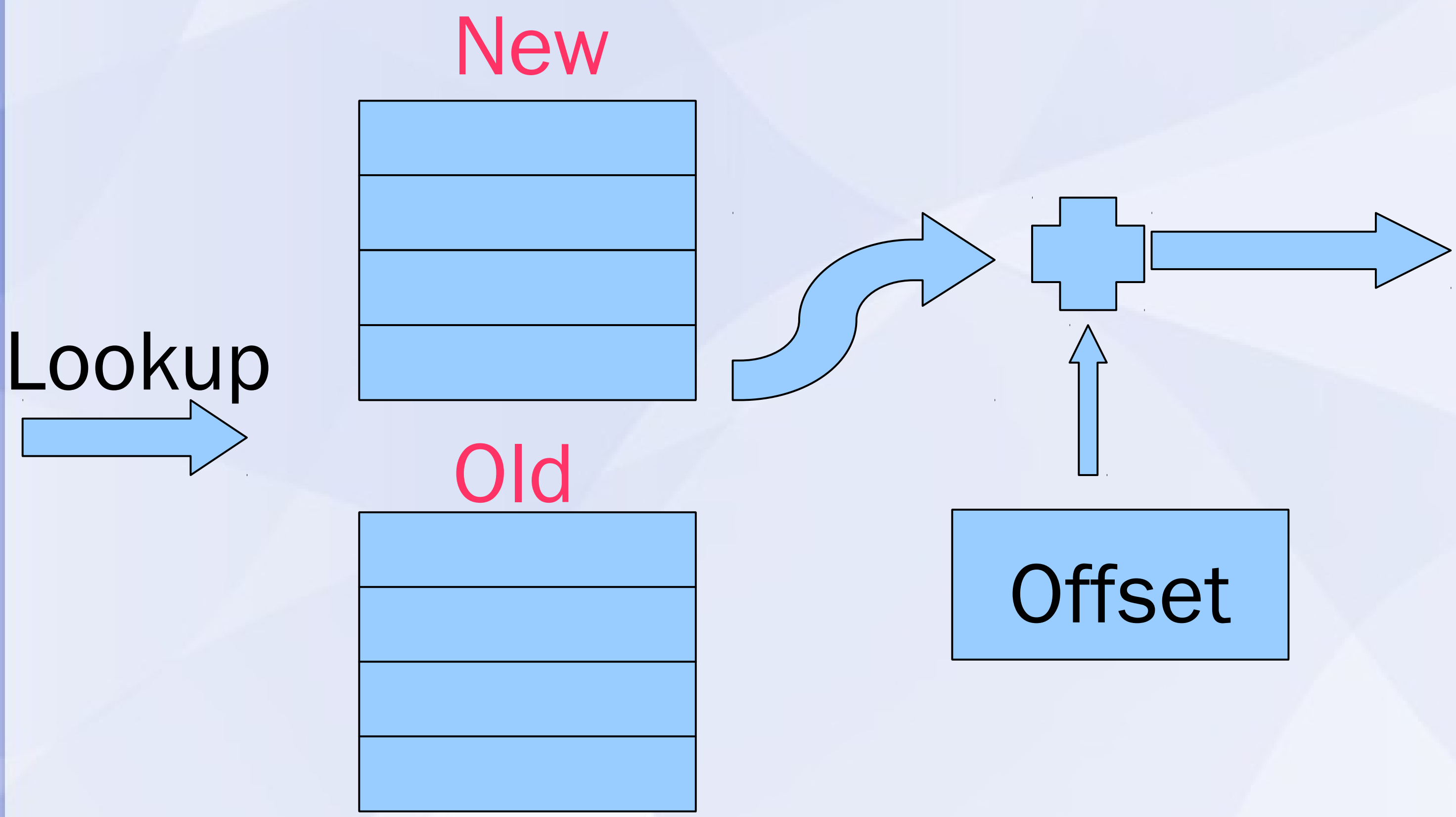
Sliding Windows

Use two hash maps with an offset to store EZ_i



Sliding Windows

Use two hash maps with an offset to store EZ_i



Summary

Time: linear

Space: $\log \varepsilon / \log (1-1/M) \approx M \log(1/\varepsilon)$

Precision: smaller with larger M

Summary

Time: linear

Space: $\log \varepsilon / \log (1-1/M) \approx M \log(1/\varepsilon)$

Precision: smaller with larger M

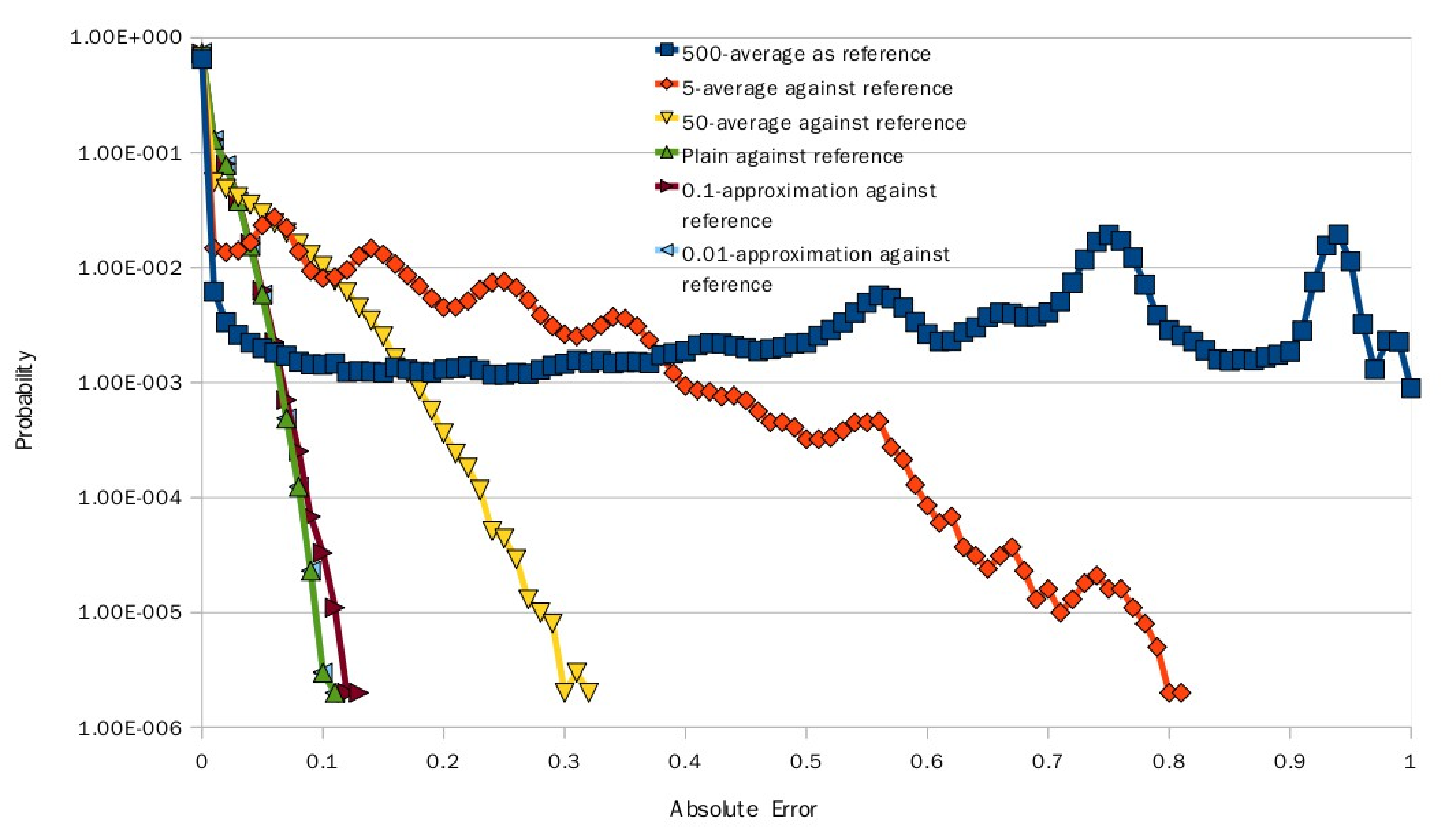
Extends to set-associative cache

Evaluation Setup

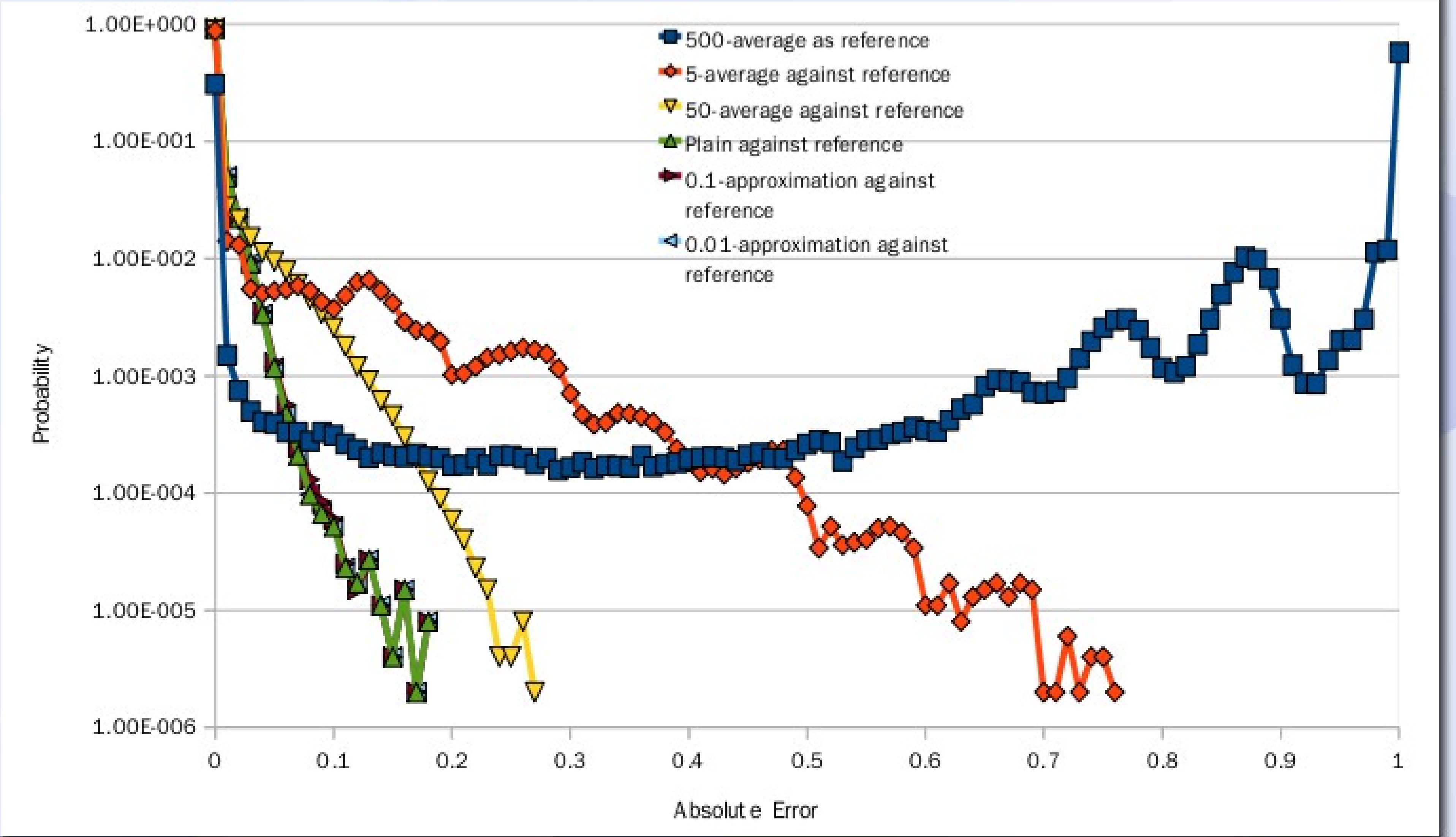
Evaluate against multiple rounds of Monte Carlo simulation

Realistic traces (1GB) collected by HMTT for CPU2000/LINPACK.

M=4, CPU2000/SWIM

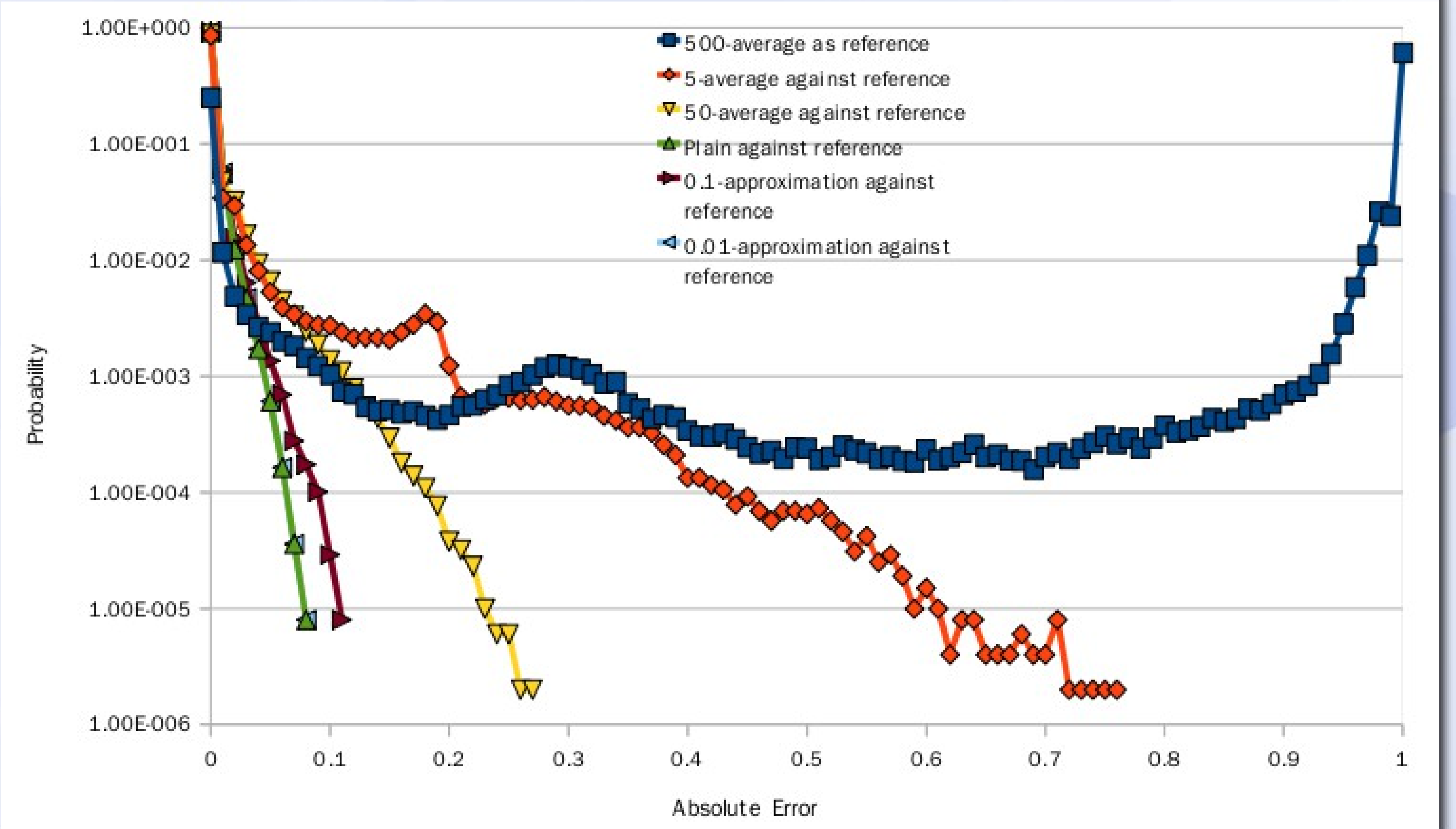


M=8, LINPACK



500-average as reference
5-average against reference
50-average against reference
Plain against reference
0.1-approximation against reference
0.01-approximation against reference
Probability
Absolute Error
1.00E+000
1.00E-001
1.00E-002
1.00E-003
1.00E-004
1.00E-005
1.00E-006
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

M=64, LINPACK



1.00E+000
1.00E-001
1.00E-002
1.00E-003
1.00E-004
1.00E-005
1.00E-006
Probability
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Absolute Error
500-average as reference
5-average against reference
50-average against reference
Plain against reference
0.1-approximation against reference
0.01-approximation against reference

Q&A

Thank you!