CACHE OPTIMIZATION FOR THE MODERN WEB

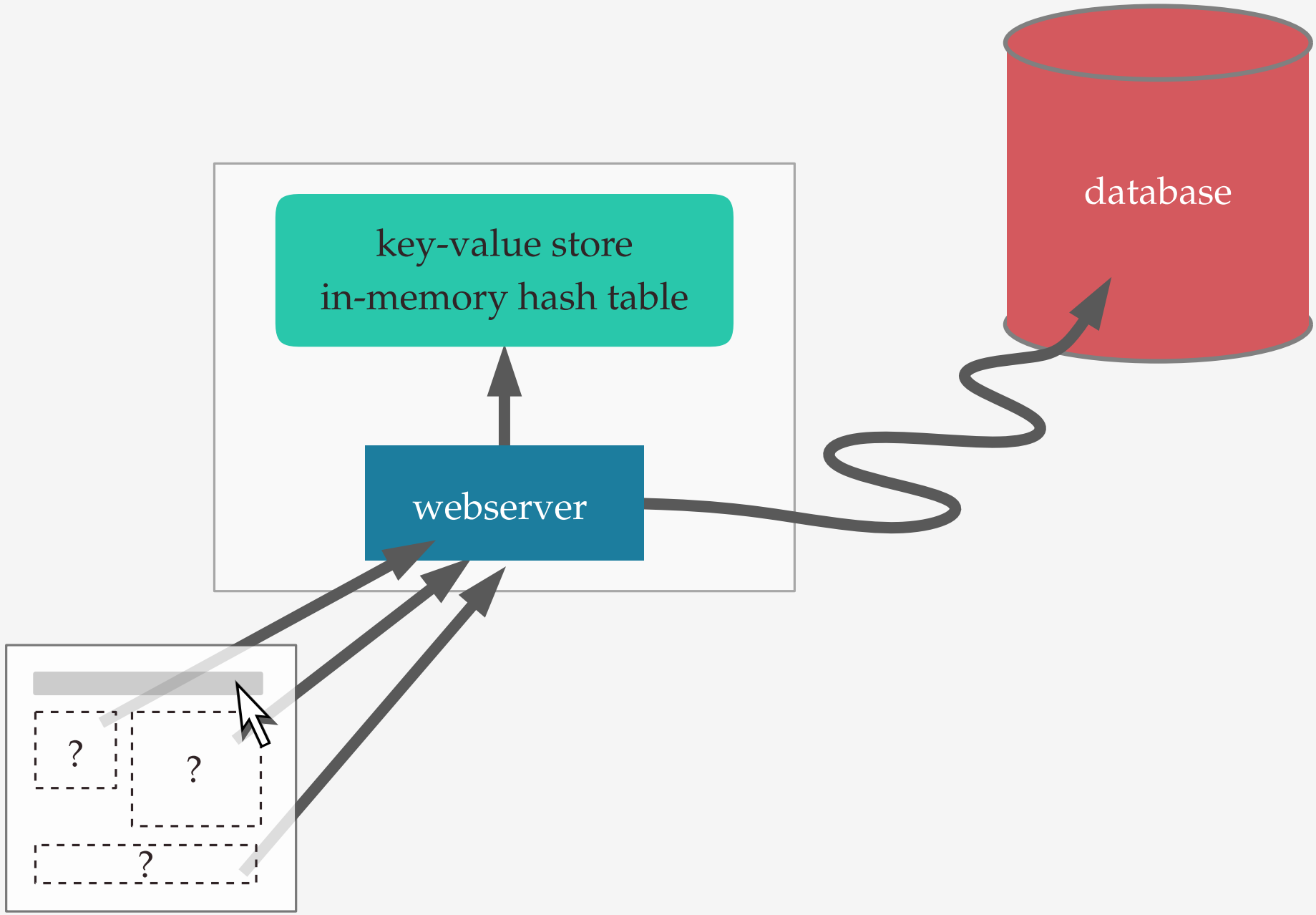
Jenny Lam

Sandy Irani (chair)

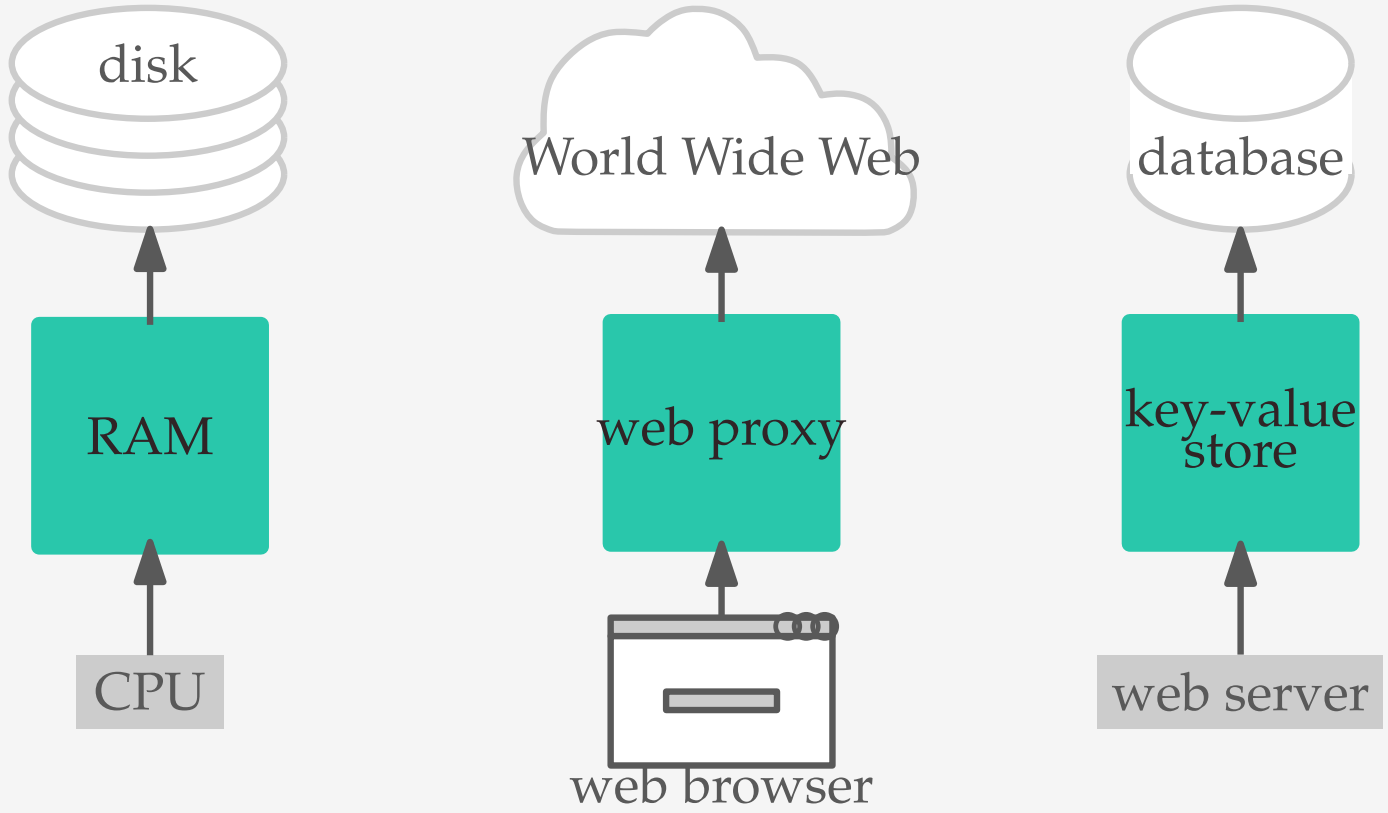
Michael Dillencourt

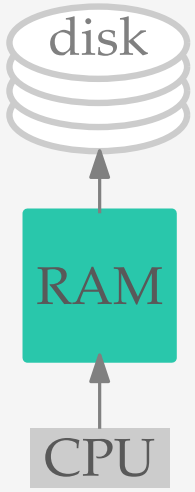
Michael T. Goodrich

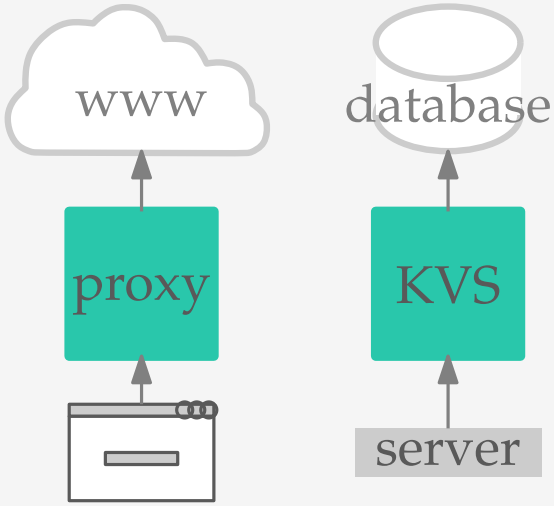
11/24/2015

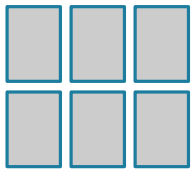


Scaling Memcache at Facebook, NFGKLLMPPSSTTV, NSDI 2013


disk
RAM
CPU
World Wide Web
web proxy
web browser
database
key-value store
web server



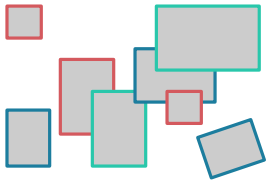




PAGING

minimize

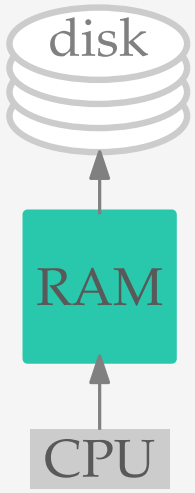
number of cache misses

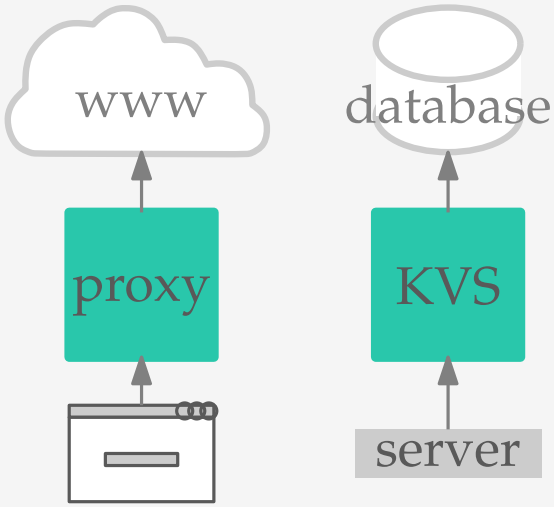


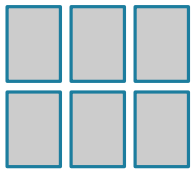
GENERALIZED CACHING

minimize

total cost of cache misses





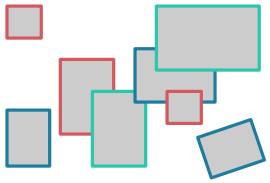


PAGING

minimize

number of cache misses

Least Recently Used (LRU)



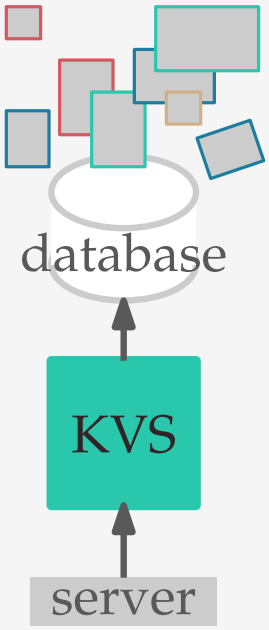
GENERALIZED CACHING

minimize

total cost of cache misses

GreedyDual-Size (GDS)

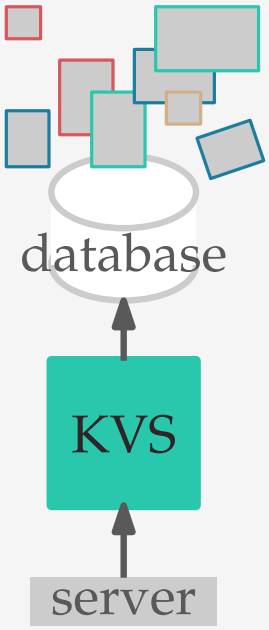
GDS → CAMP



generalized caching → managed memory caching

2-level cache → multi-level cache

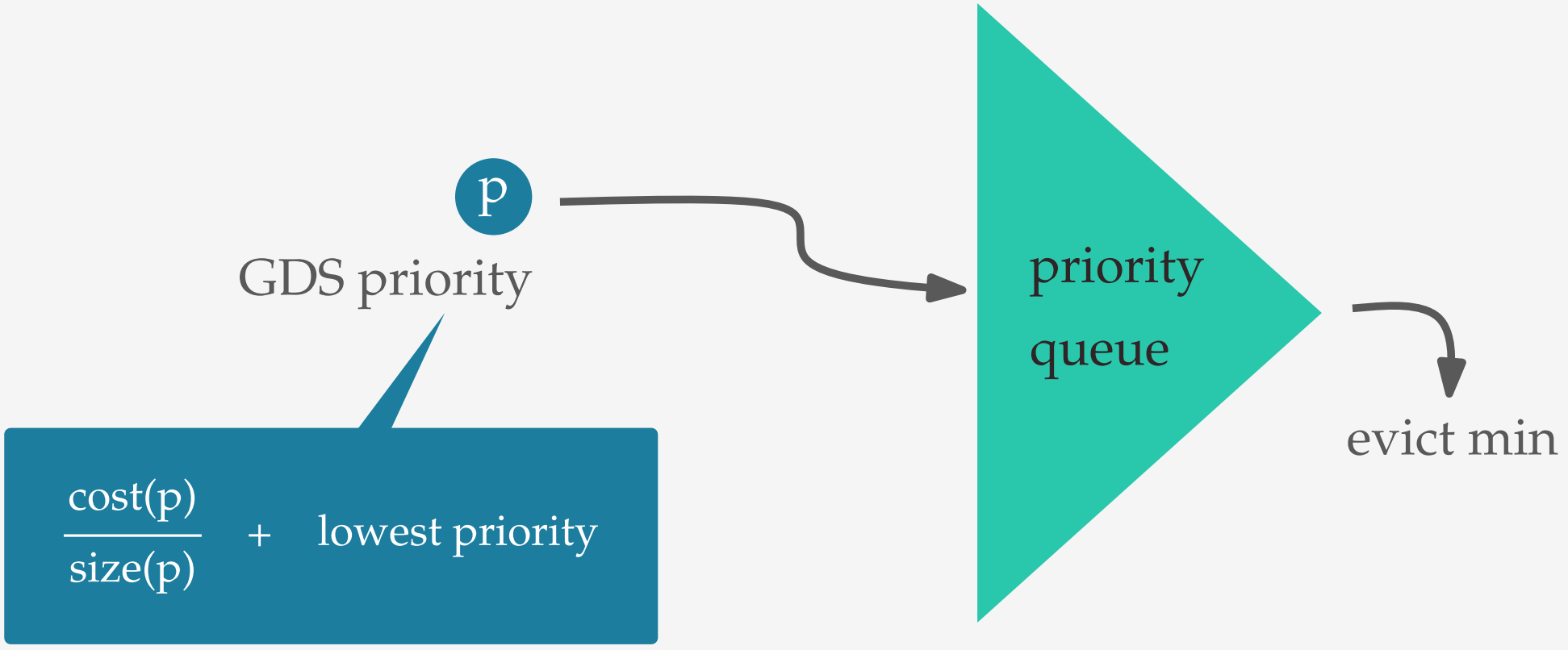
GDS ⟶ CAMP



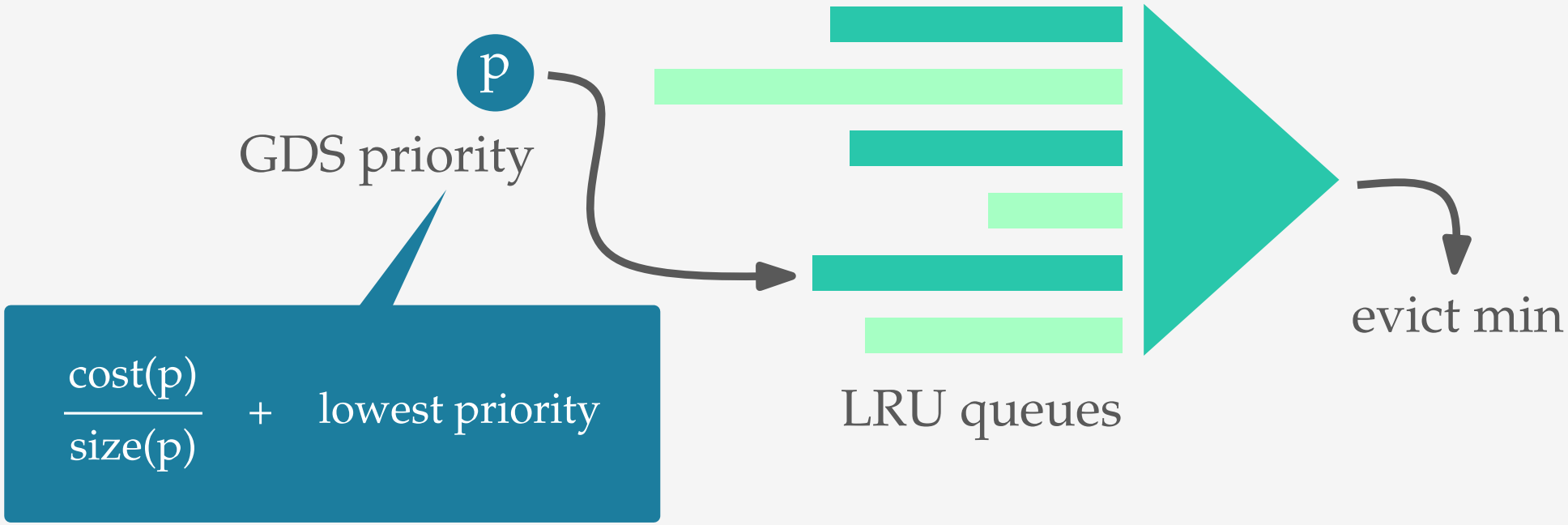
generalized caching ⟶ managed memory caching

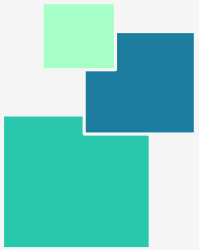
2-level cache ⟶ multi-level cache

GDS


p
GDS priority
cost(p)
size(p)
+
lowest priority
priority
queue
evict min

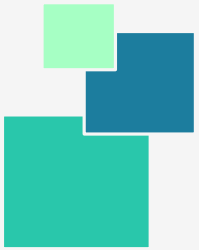
CAMP


p
GDS priority
cost(p)
size(p)
+
lowest priority
LRU queues
evict min

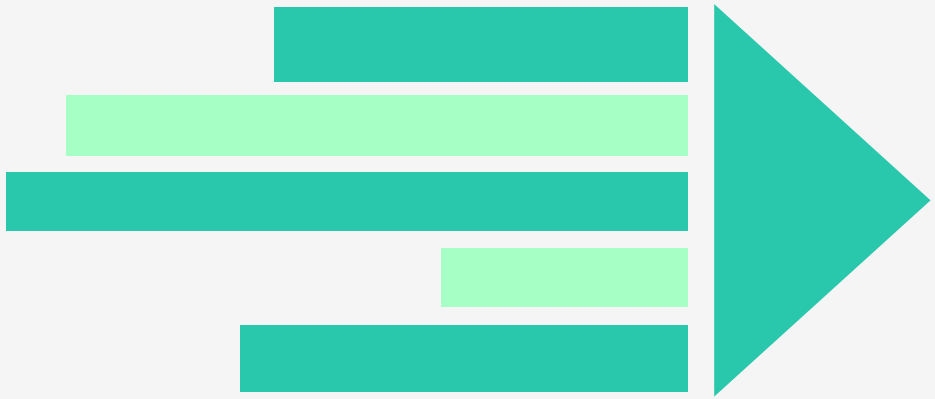


CAMP

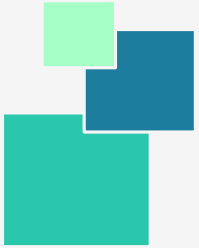
$$\text{round}\left(\frac{\text{cost(p)}}{\text{size(p)}}\right)$$



CAMP

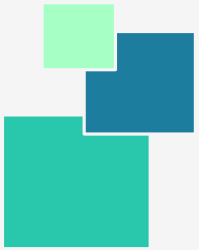


$$\text{round}\left(\frac{\text{cost(p)}}{\text{size(p)}}\right)$$



CAMP

$$\text{round}\left(\frac{\text{cost(p)}}{\text{size(p)}}\right)$$



PERFORMANCE

log (#items) per update

cost(GDS) $\leq$ k cost(OPT)

log (#queues) per update

cost(CAMP) $\leq (1 + \varepsilon)$k cost(OPT)

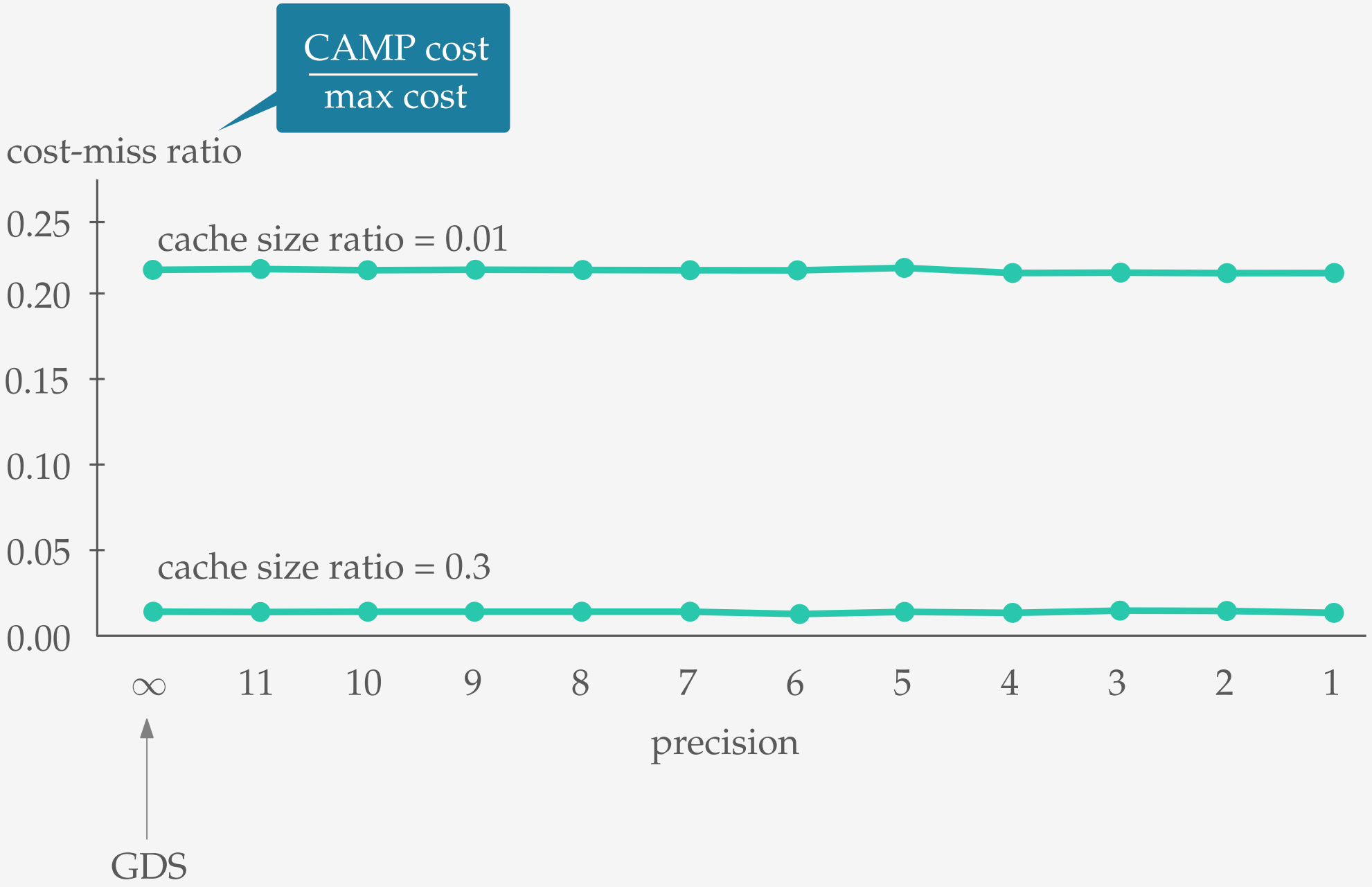
approximation parameter

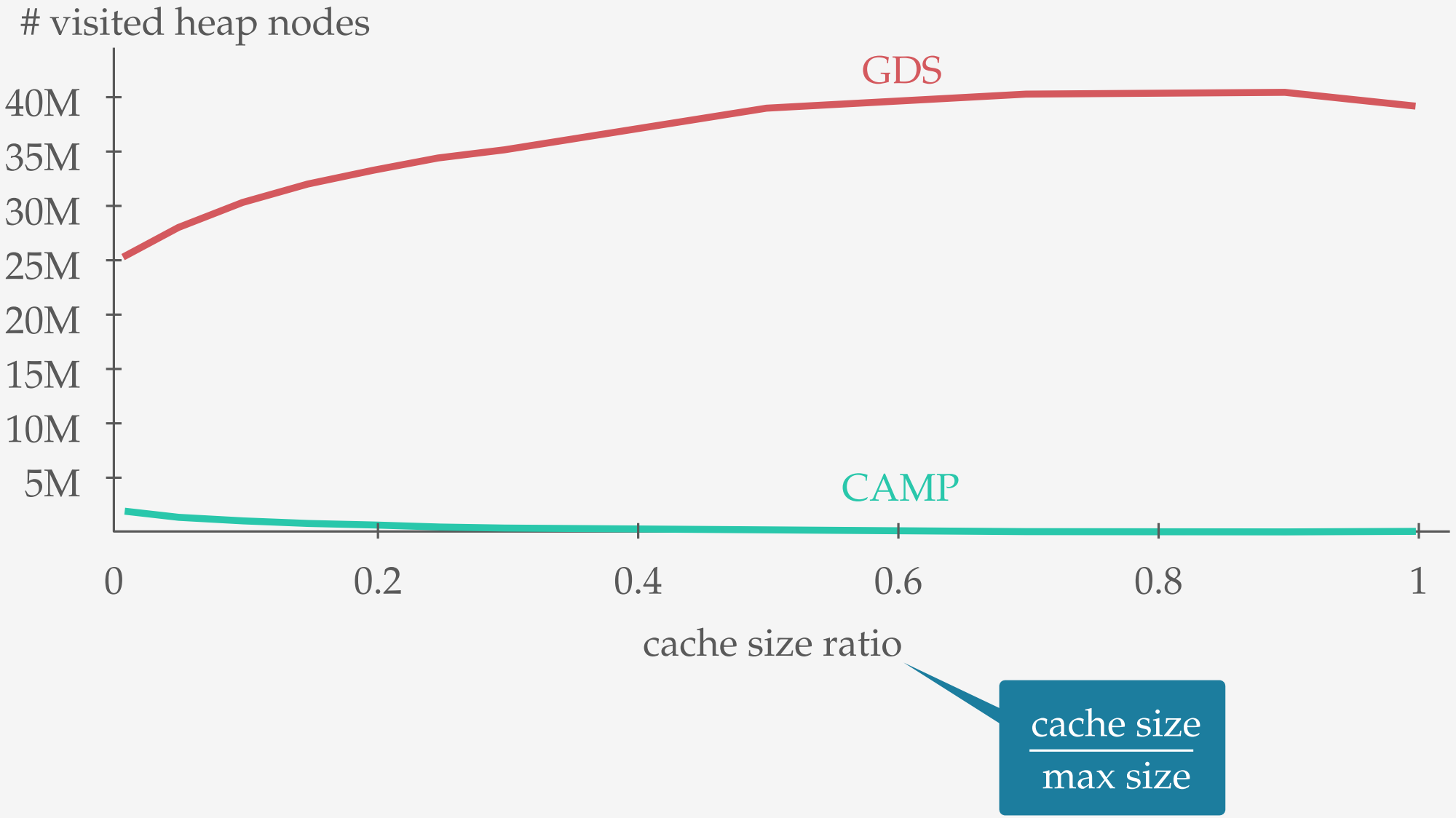
EXPERIMENTS

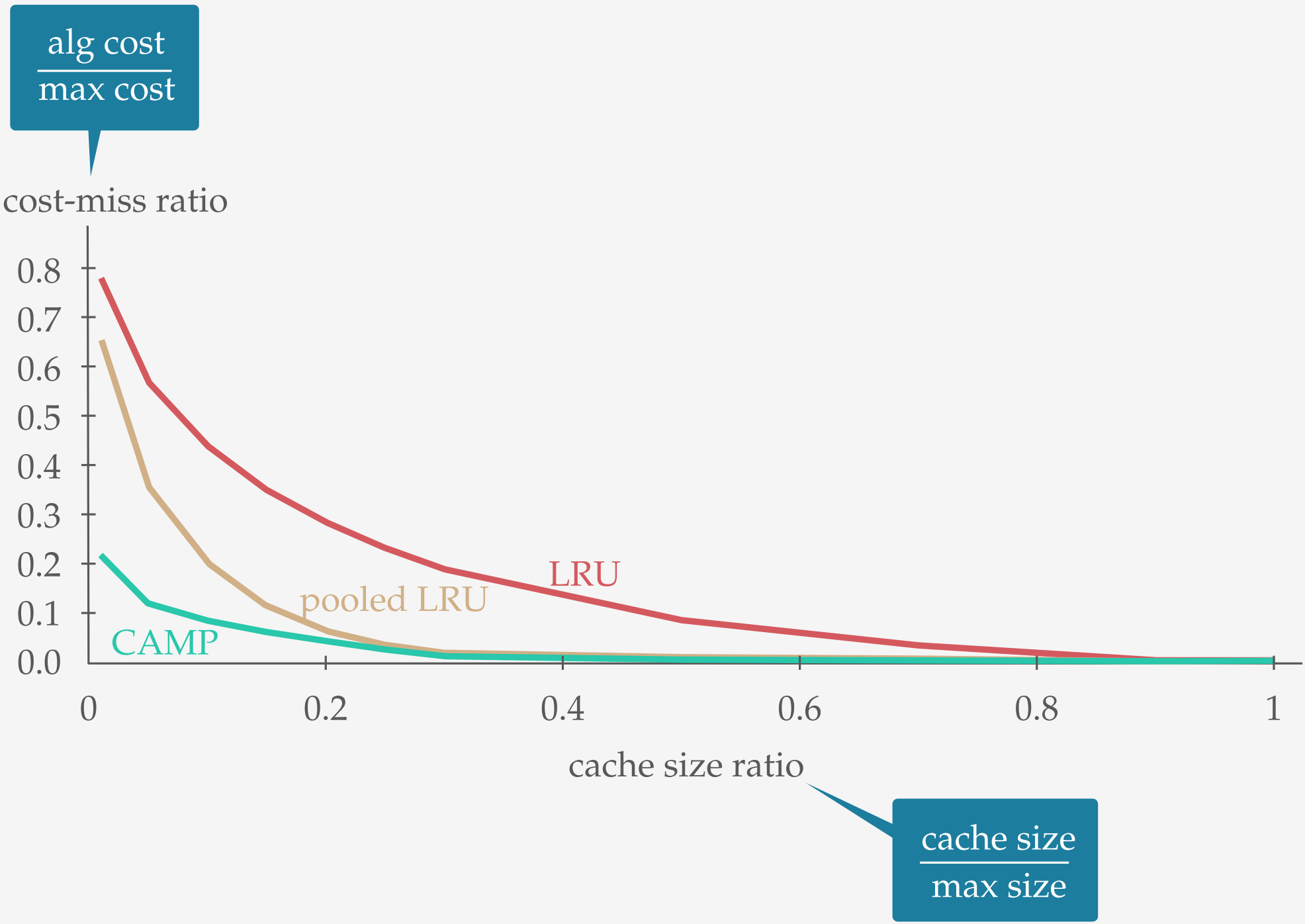
trace generated by BG, a social networking benchmark

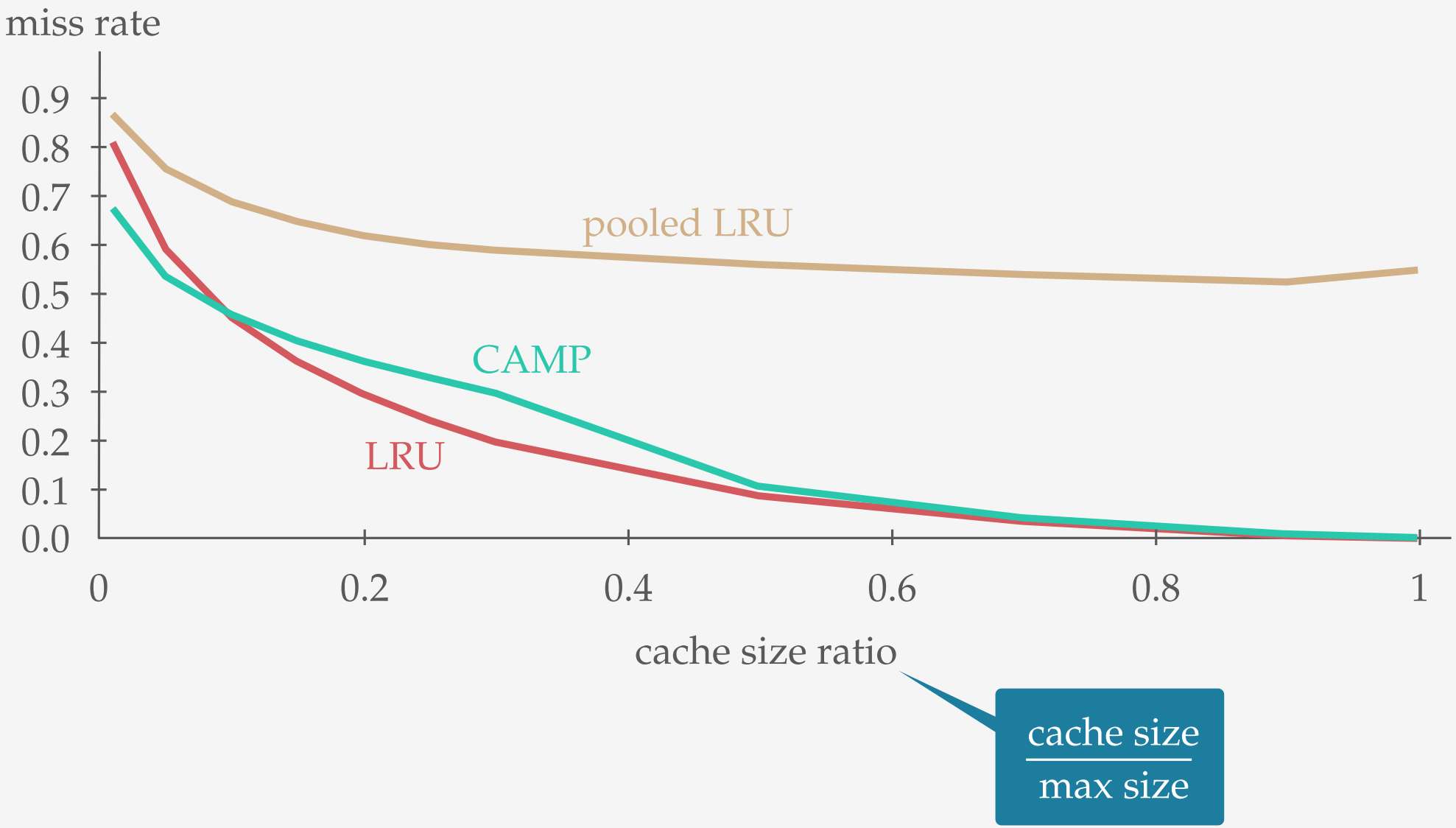
4 million requests

i.i.d. with 70% of requests to 20% of items

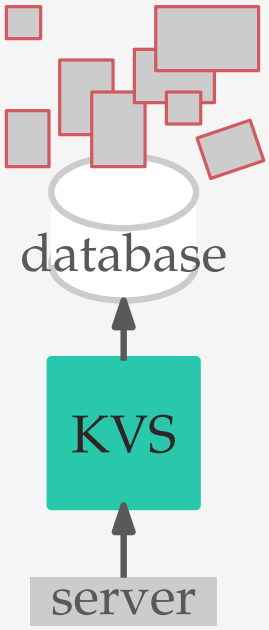

CAMP cost
max cost
cost-miss ratio
0.25
0.20
0.15
0.10
0.05
0.00
cache size ratio = 0.01
cache size ratio = 0.3
∞
11
10
9
8
7
6
5
4
3
2
1
precision
GDS


visited heap nodes
GDS
40M
35M
30M
25M
20M
15M
10M
5M
CAMP
0
0.2
0.4
0.6
0.8
1
cache size ratio
cache size
max size


alg cost
max cost
cost-miss ratio
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
LRU
pooled LRU
CAMP
0
0.2
0.4
0.6
0.8
1
cache size ratio
cache size
max size


miss rate
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
0.2
0.4
0.6
0.8
1
pooled LRU
CAMP
LRU
cache size ratio
cache size
max size

GDS ⟶ CAMP

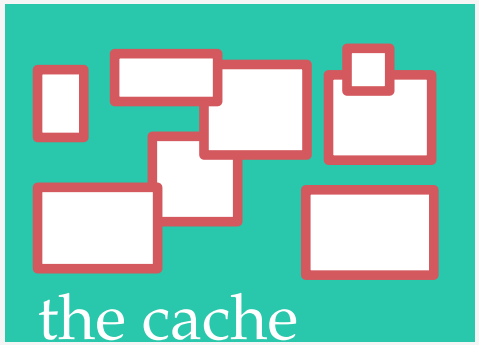


generalized caching ⟶ managed memory caching

2-level cache ⟶ multi-level cache

THE GENERALIZED CACHING PROBLEM

variable size and cost



GOAL

minimize **total cost** of cache misses

SUBJECT TO

total size of items in cache
cannot exceed the cache size

THE MANAGED MEMORY CACHING PROBLEM

variable size and cost

every item must fit in a contiguous segment of memory

CACHE REPLACEMENT

MEMORY ALLOCATION

CAMP-MALLOC

LRU queues

CAMP-MALLOC

FIFO queues

CAMP-MALLOC

FIFO queue

CAMP-MALLOC

FIFO queue

CAMP-MALLOC

FIFO queue

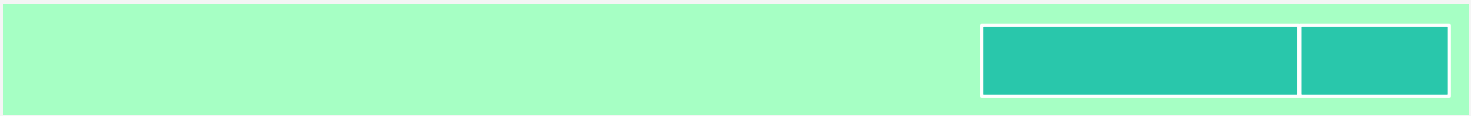
CAMP-MALLOC

FIFO queue

CAMP-MALLOC

FIFO queue

CAMP-MALLOC

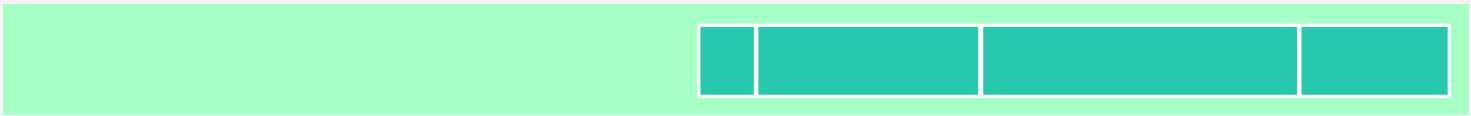


FIFO queue

CAMP-MALLOC

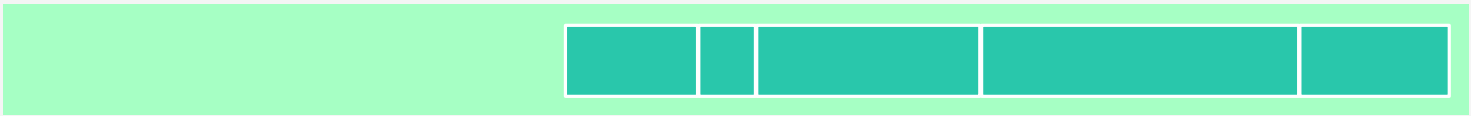
FIFO queue

CAMP-MALLOC



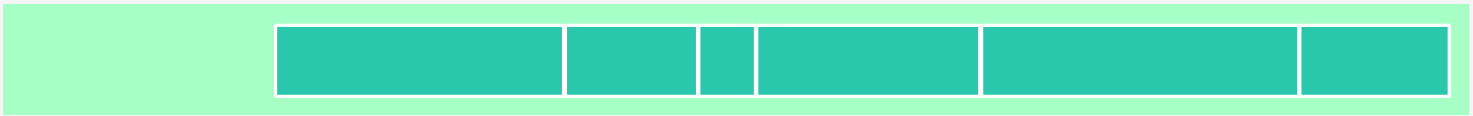
FIFO queue

CAMP-MALLOC



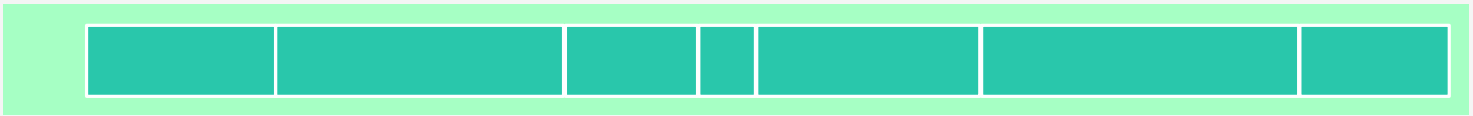
FIFO queue

CAMP-MALLOC



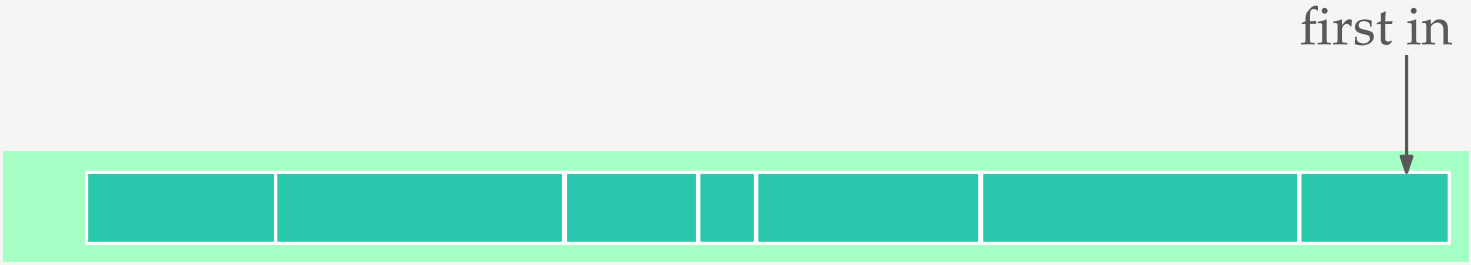
FIFO queue

CAMP-MALLOC



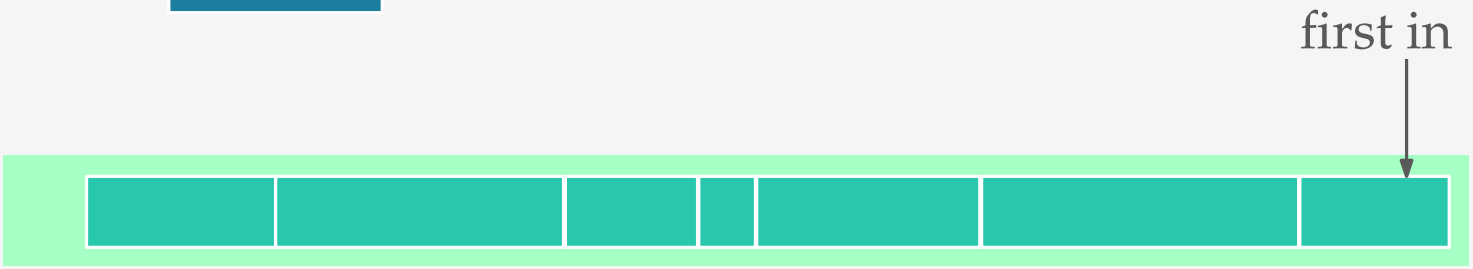
FIFO queue

CAMP-MALLOC



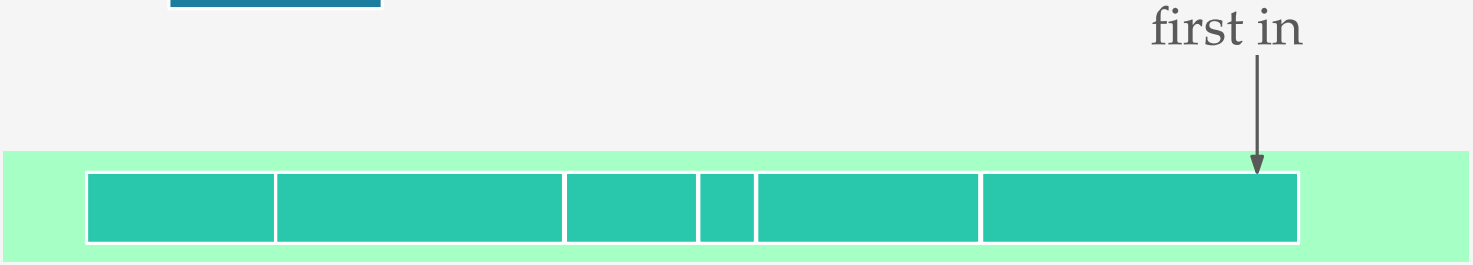
FIFO queue

CAMP-MALLOC



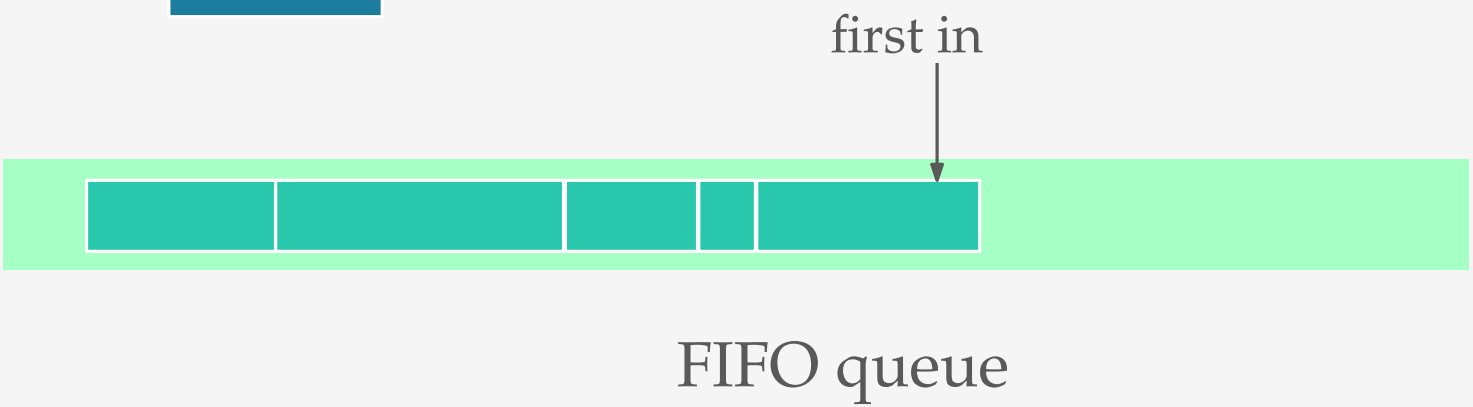
FIFO queue

CAMP-MALLOC



FIFO queue

CAMP-MALLOC



FIFO queue

CAMP-MALLOC

FIFO queue

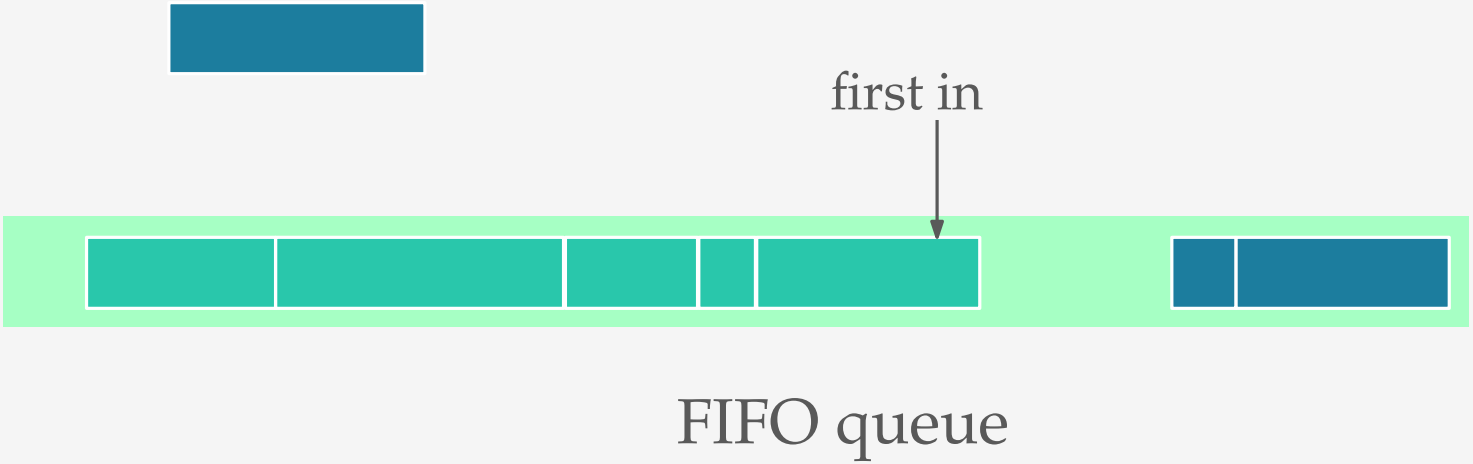
CAMP-MALLOC

FIFO queue

CAMP-MALLOC

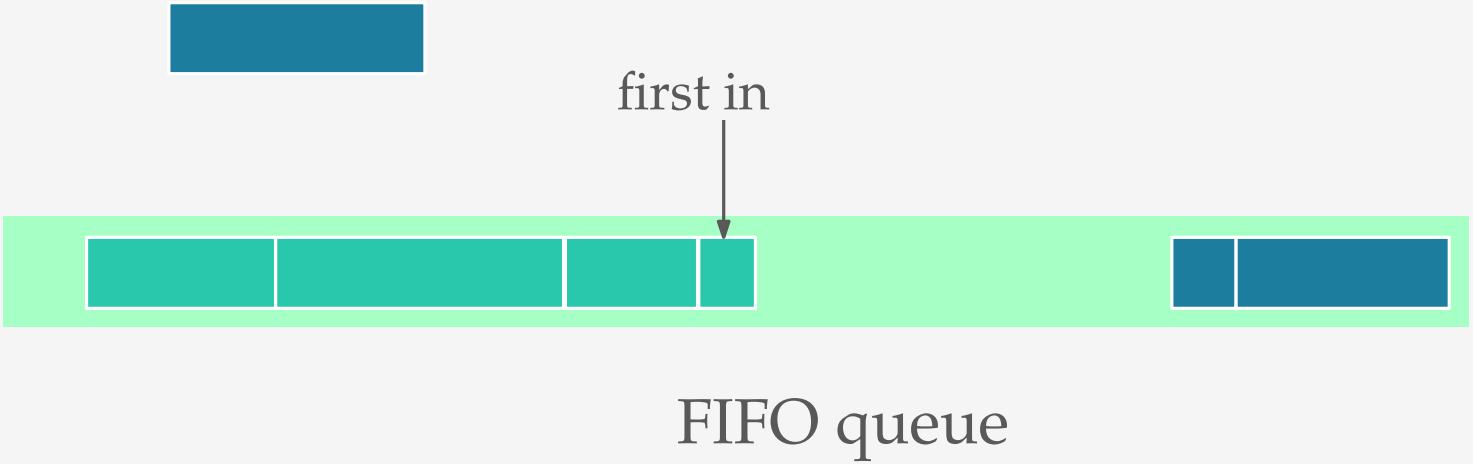
FIFO queue

CAMP-MALLOC

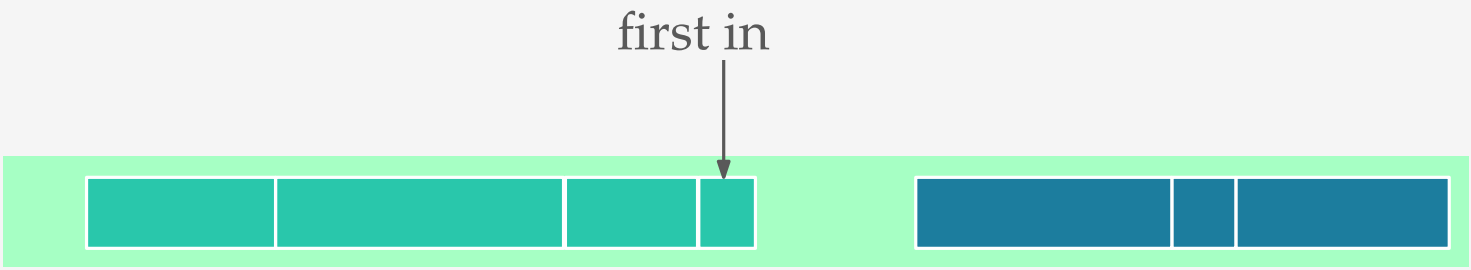


FIFO queue

CAMP-MALLOC



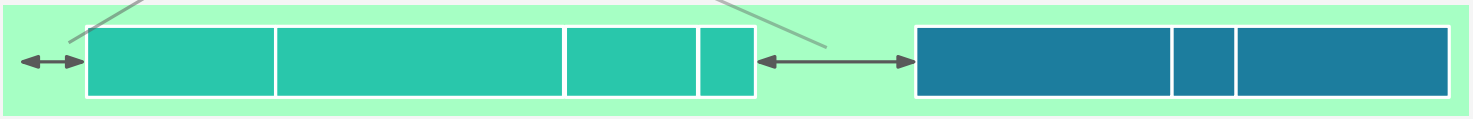
CAMP-MALLOC



FIFO queue

CAMP-MALLOC

fragmentation $\leq$ 2 (max item size)



FIFO queue

CAMP-MALLOC

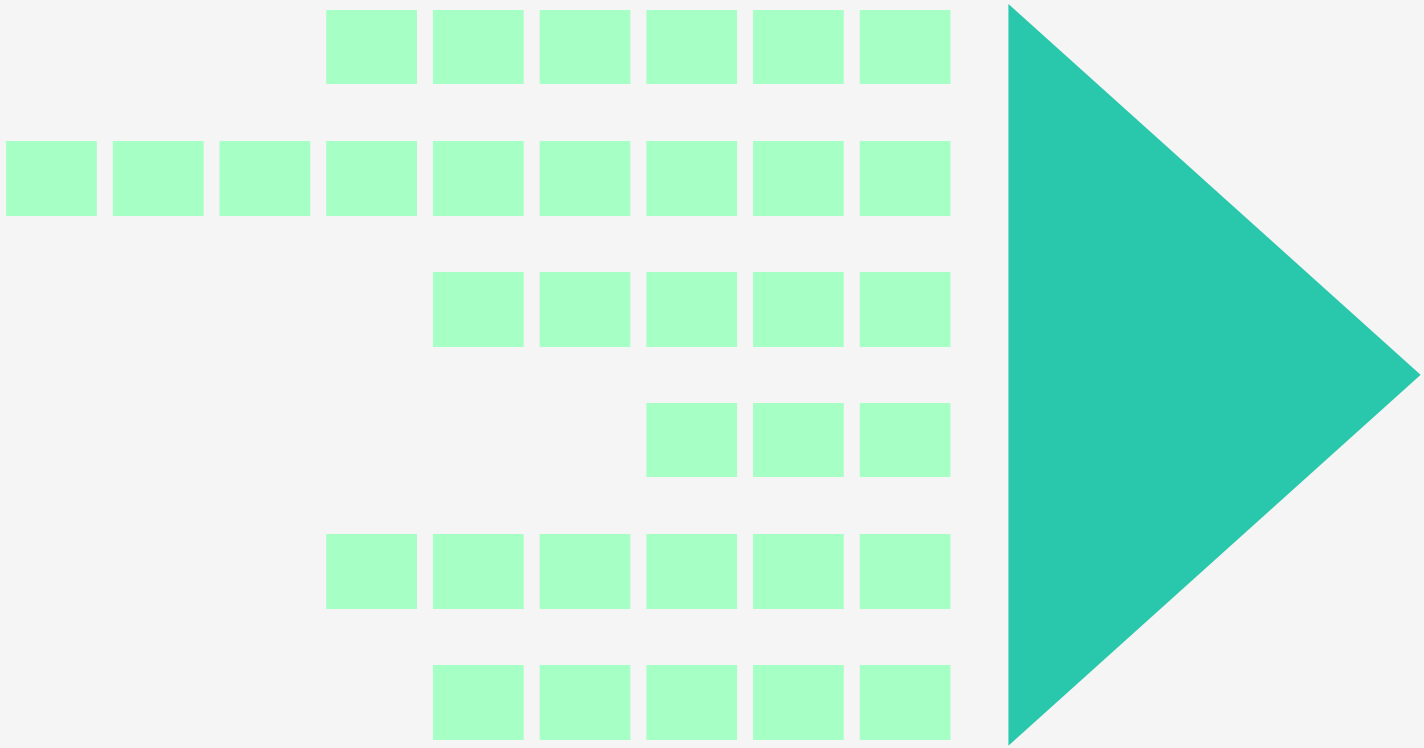
CAMP-MALLOC

CAMP-MALLOC

CAMP-MALLOC

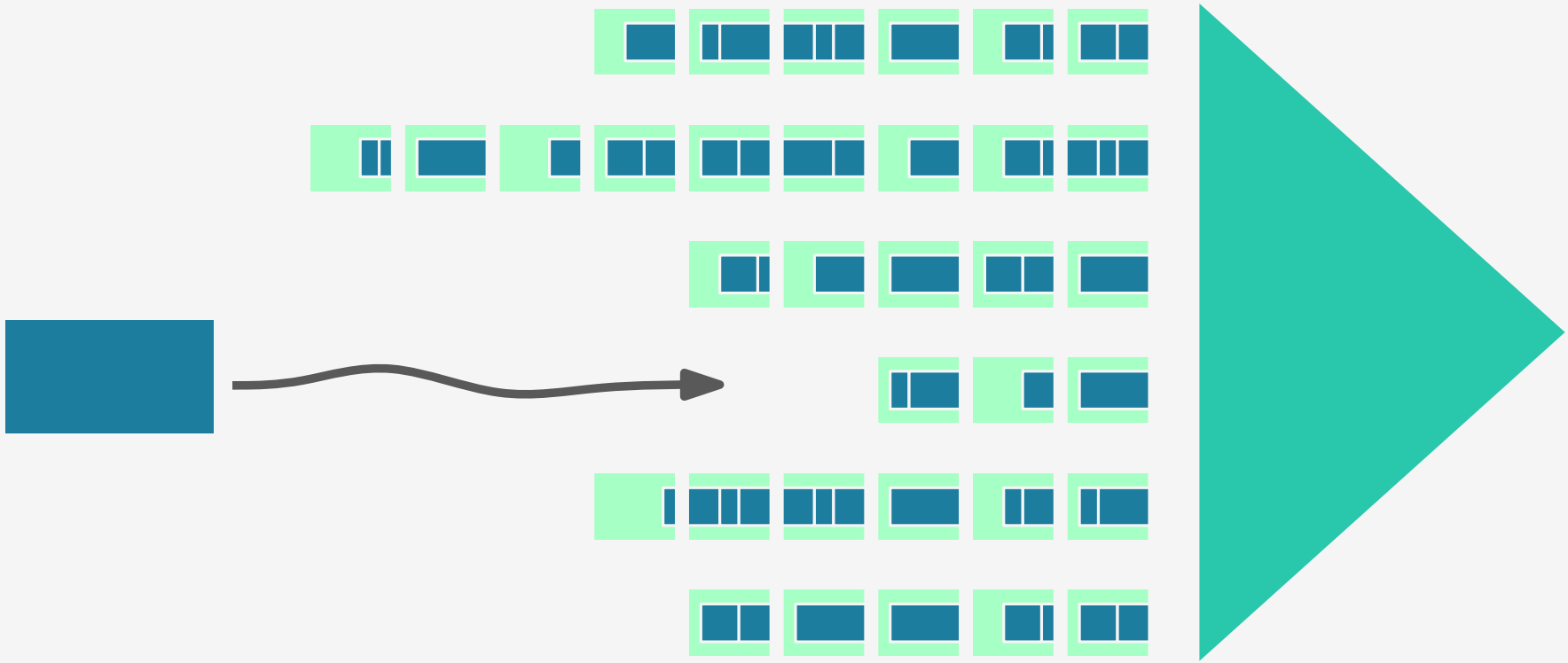
CAMP-MALLOC

CAMP-MALLOC

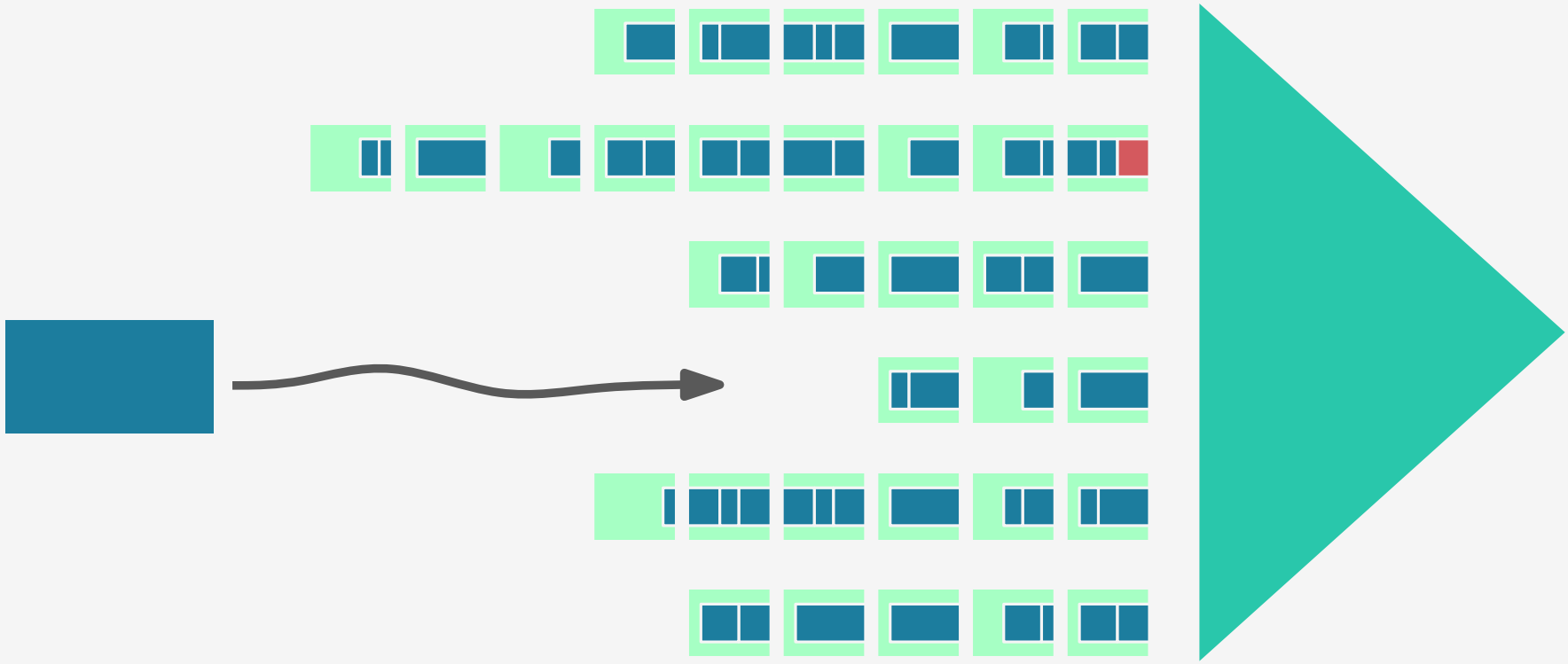


CAMP-MALLOC

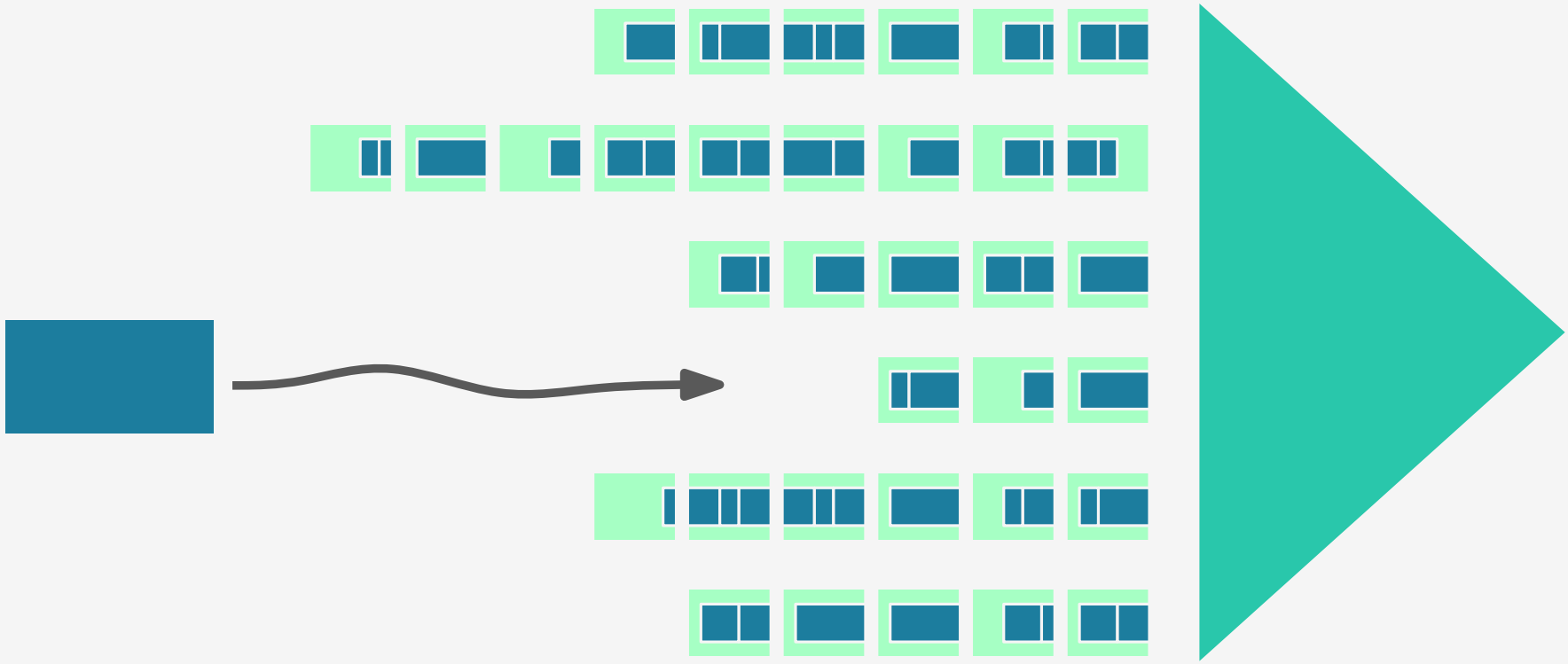
CAMP-MALLOC



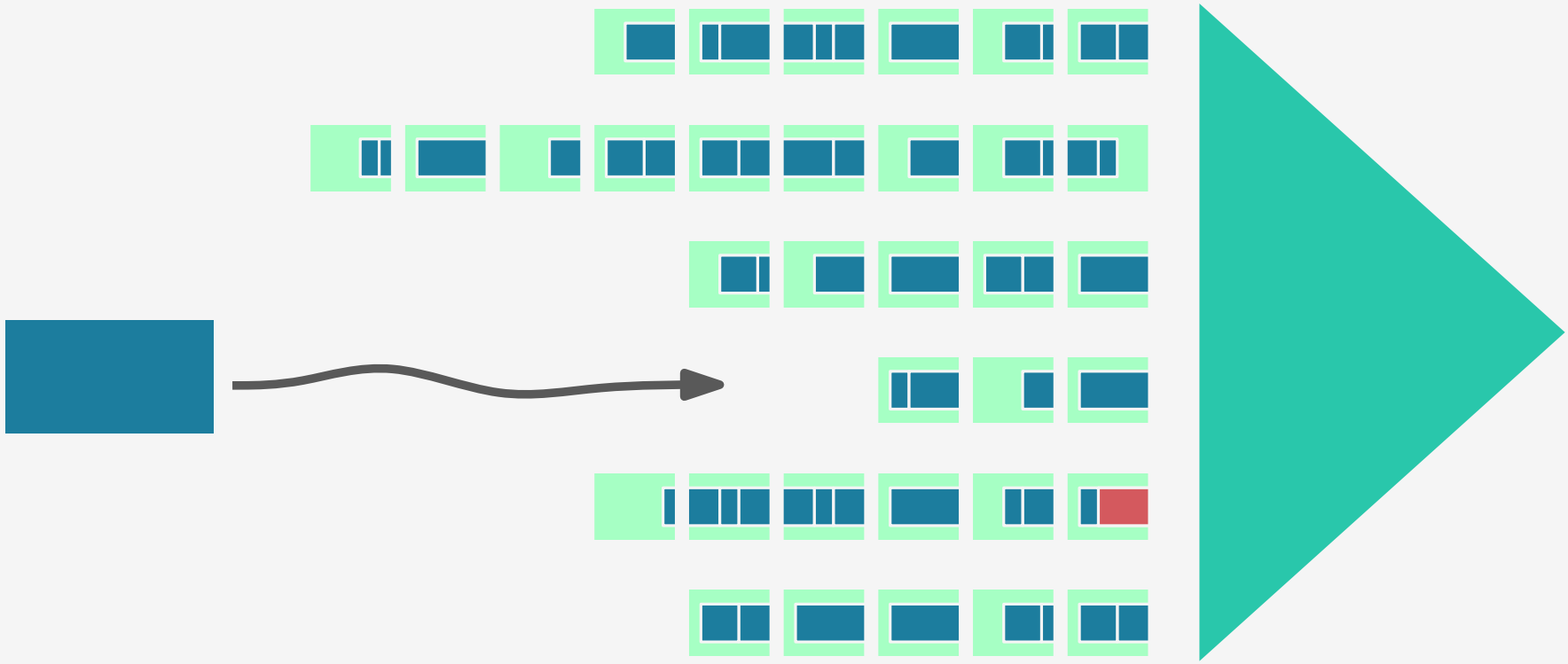
CAMP-MALLOC



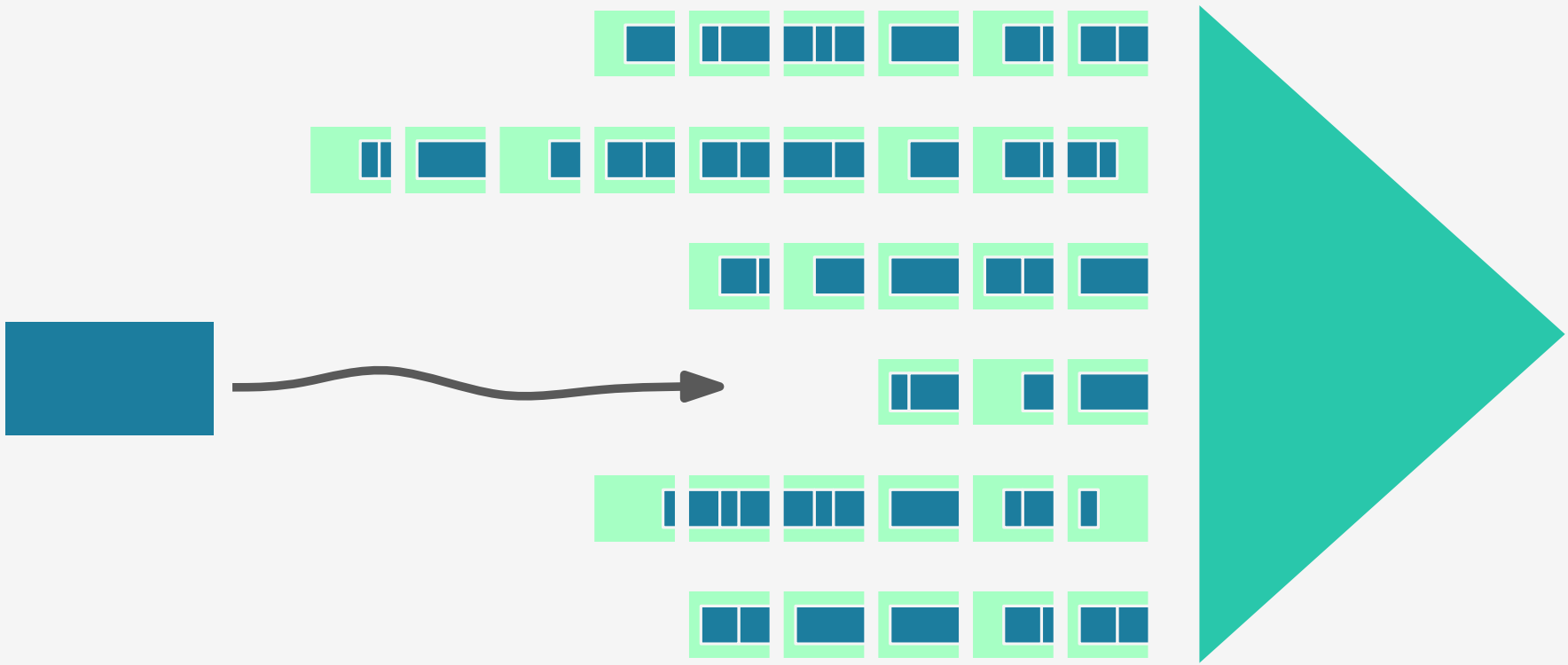
CAMP-MALLOC



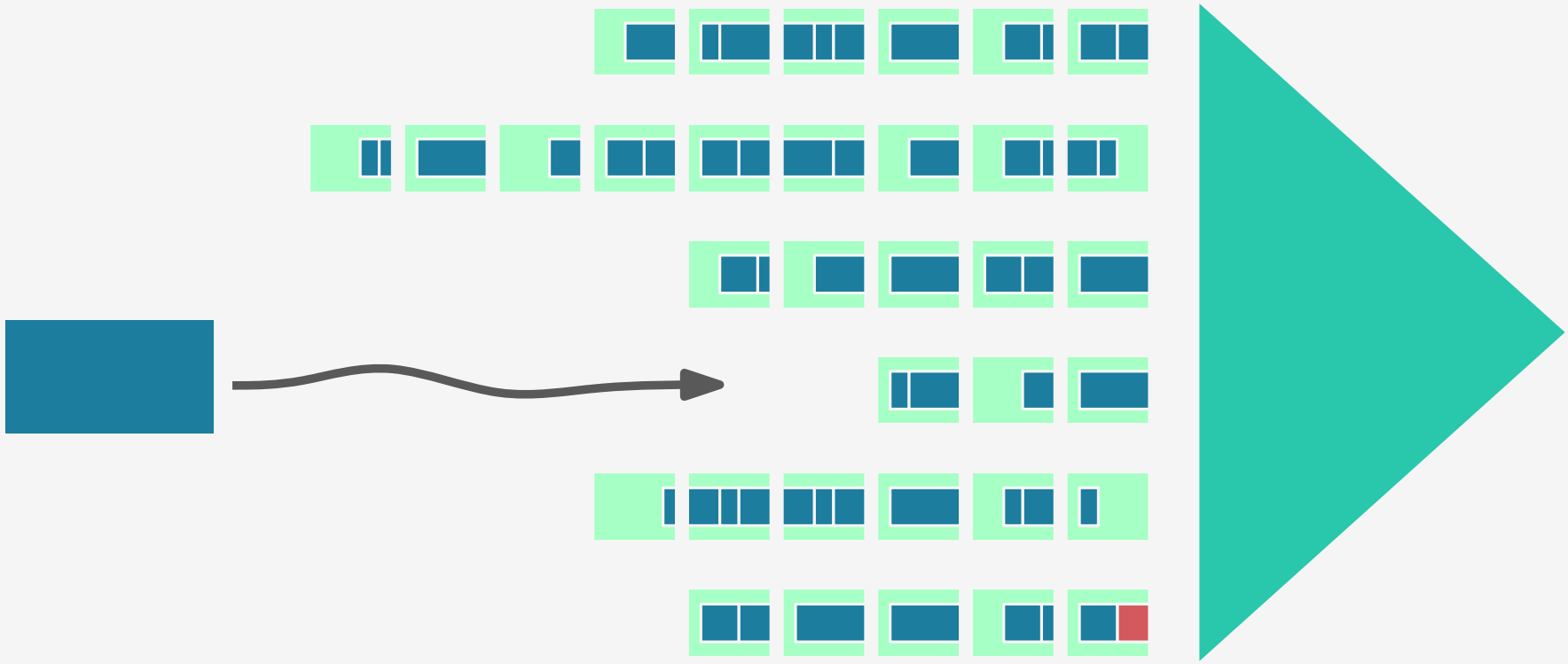
CAMP-MALLOC



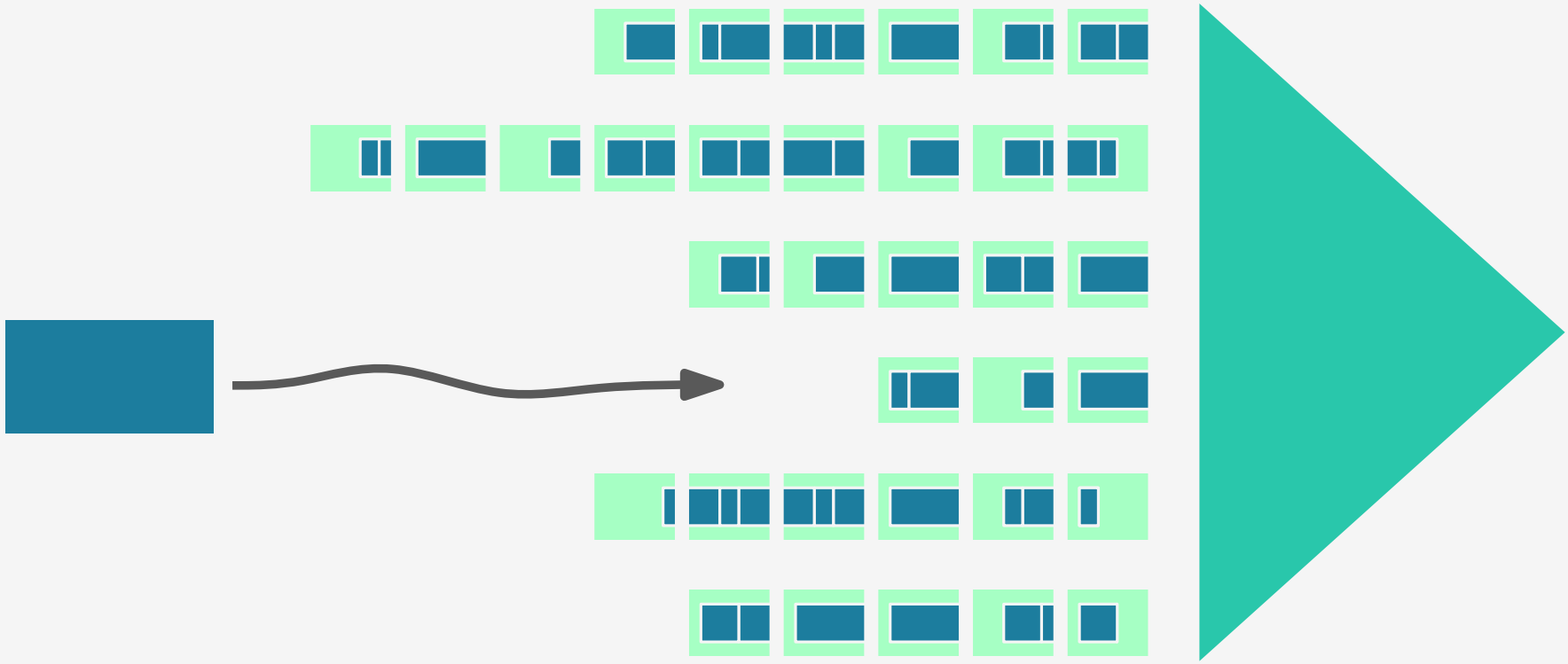
CAMP-MALLOC



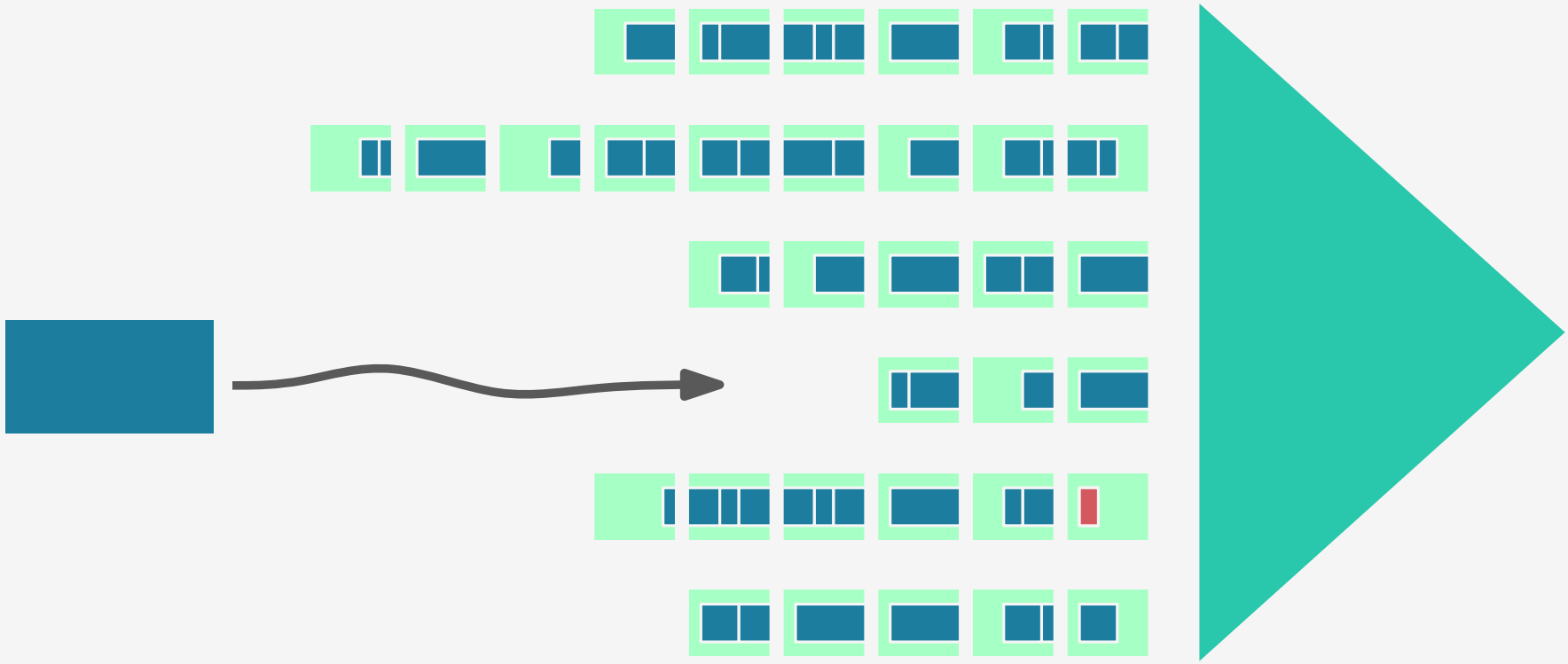
CAMP-MALLOC



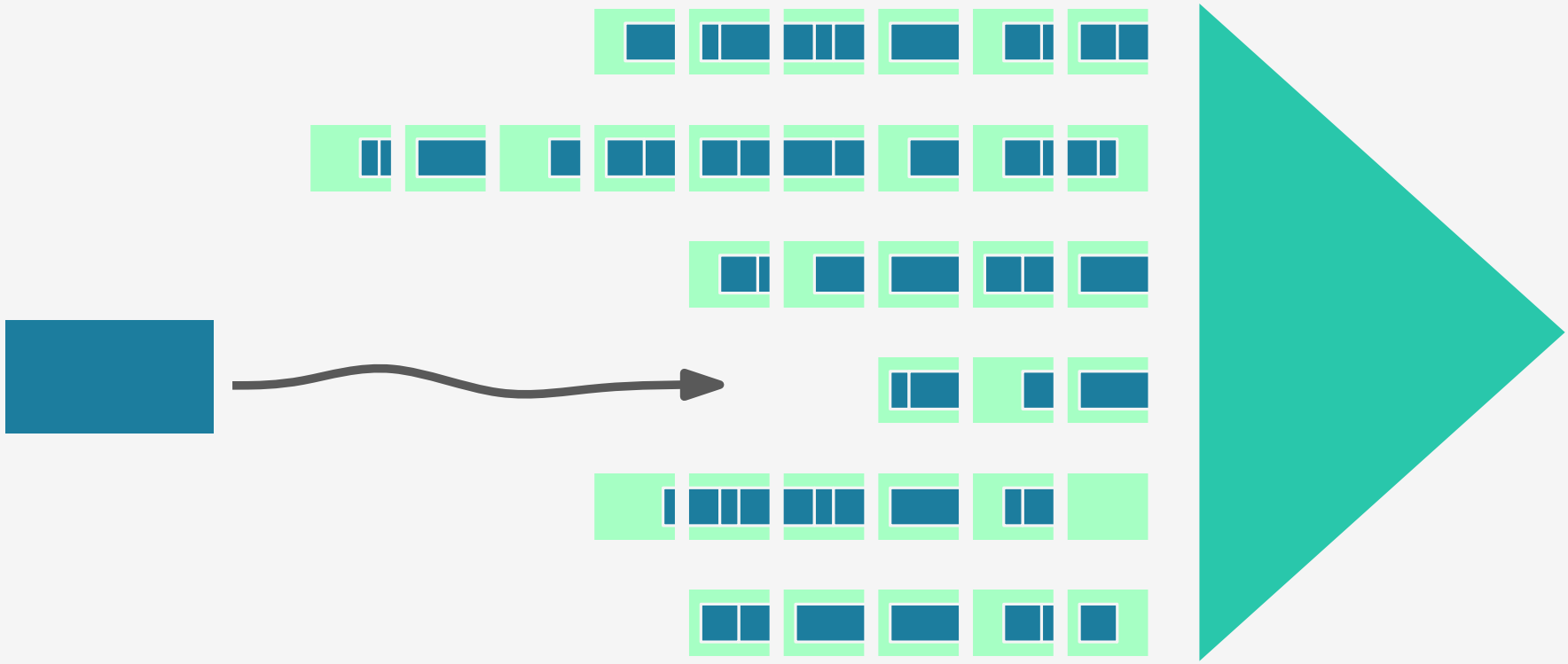
CAMP-MALLOC



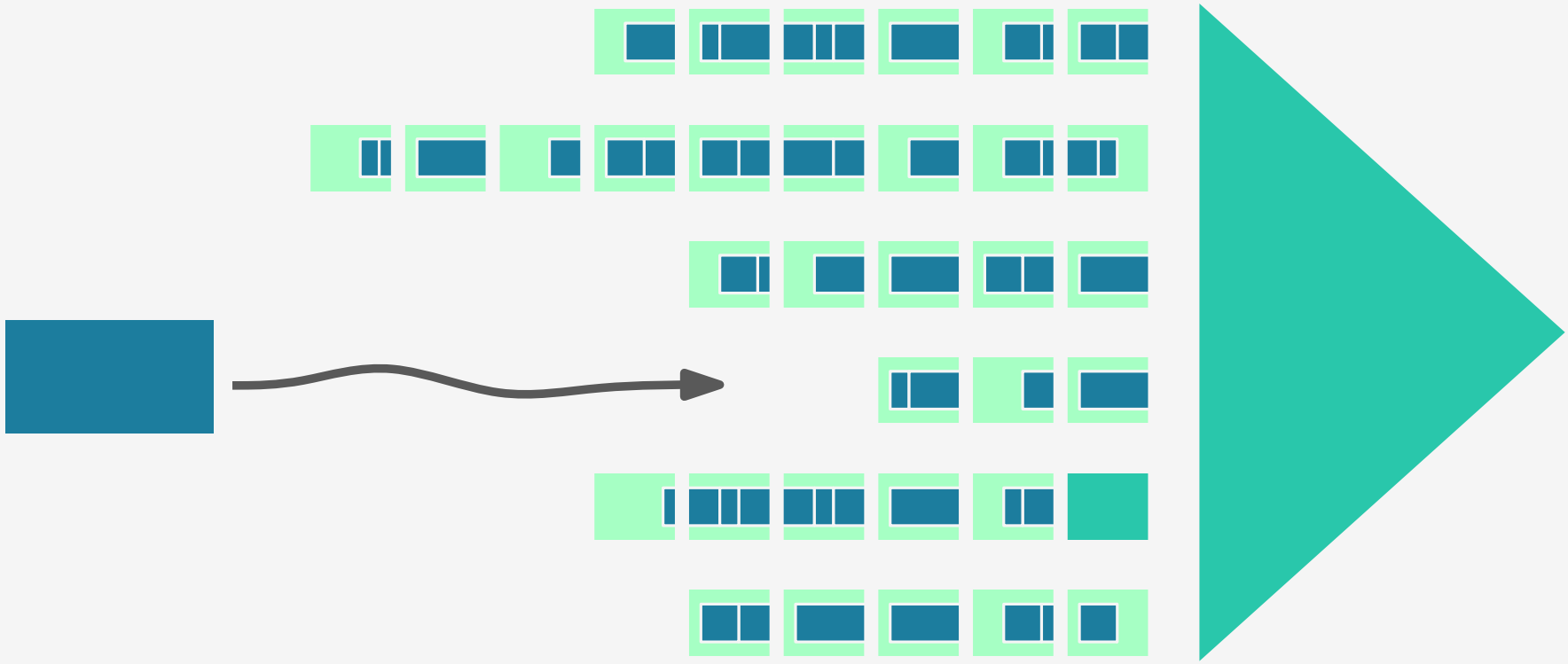
CAMP-MALLOC



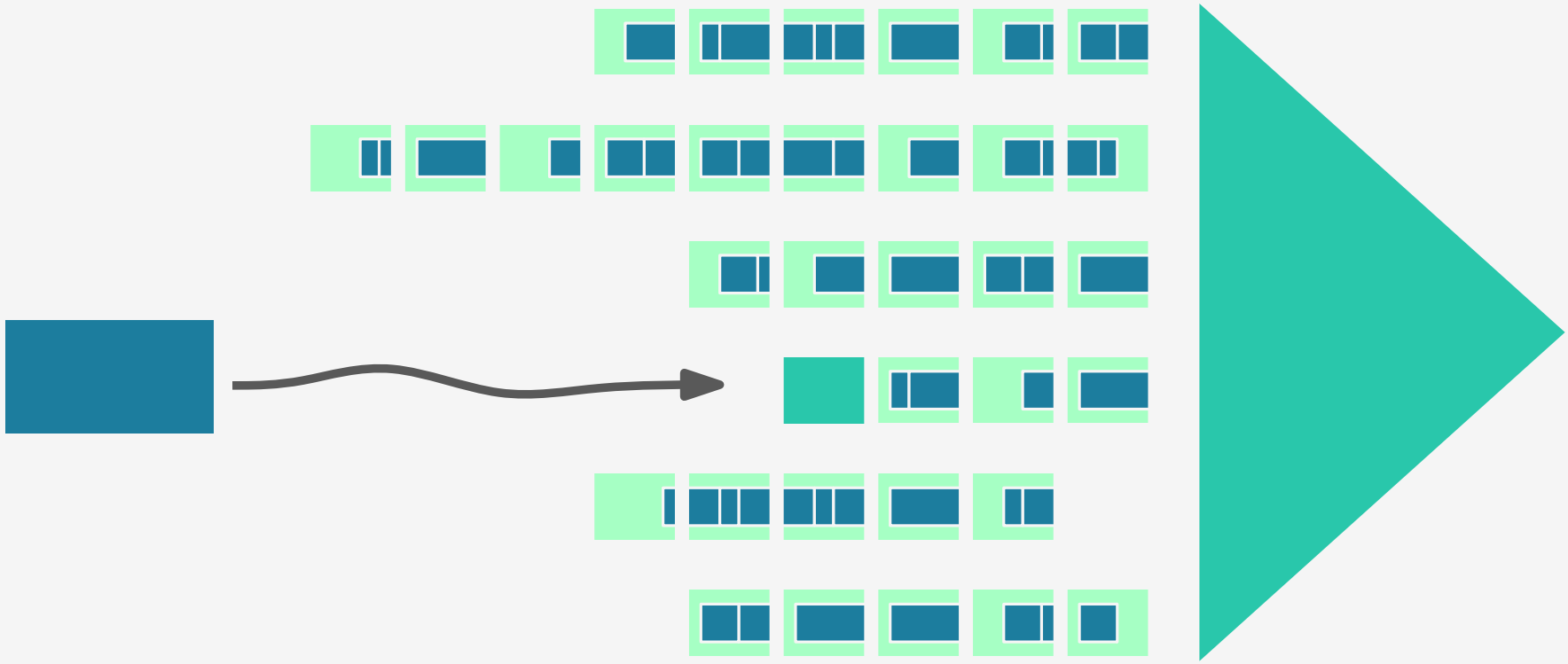
CAMP-MALLOC



CAMP-MALLOC

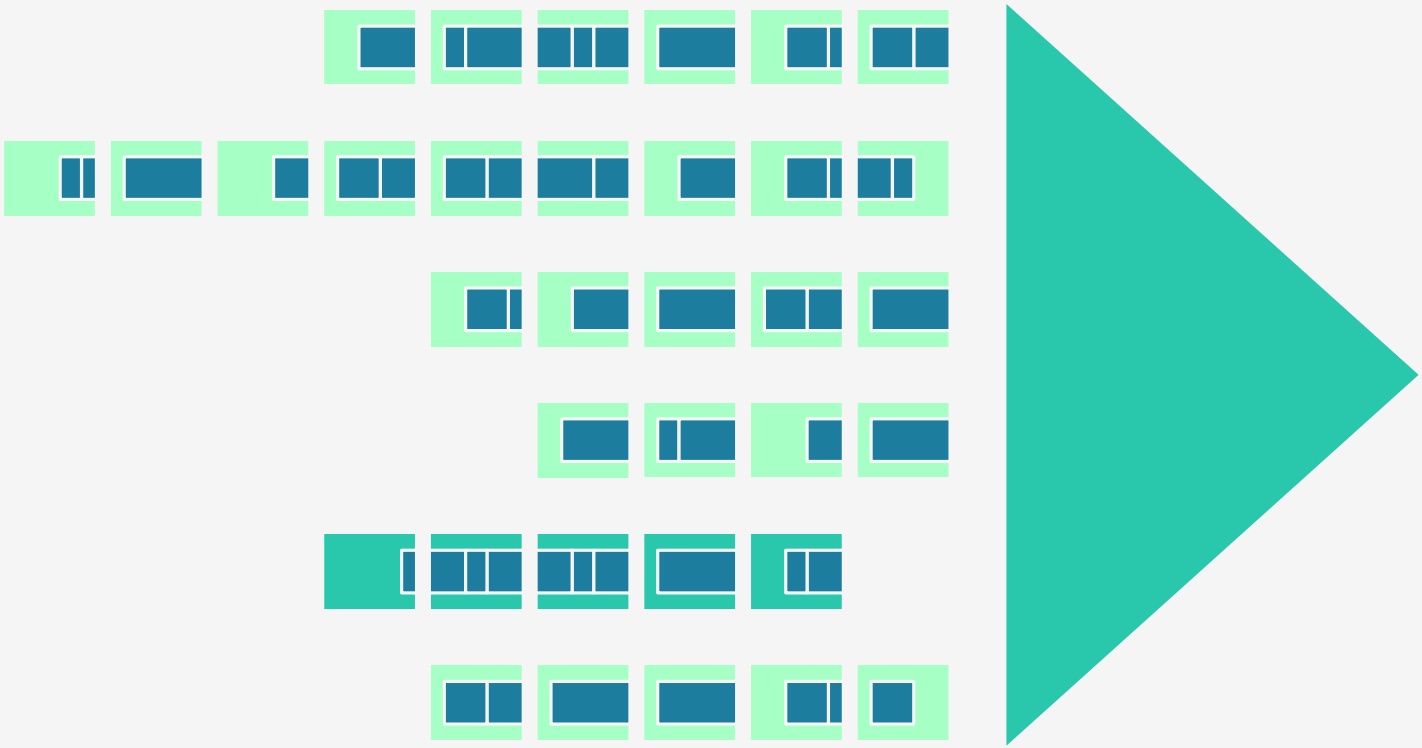


CAMP-MALLOC

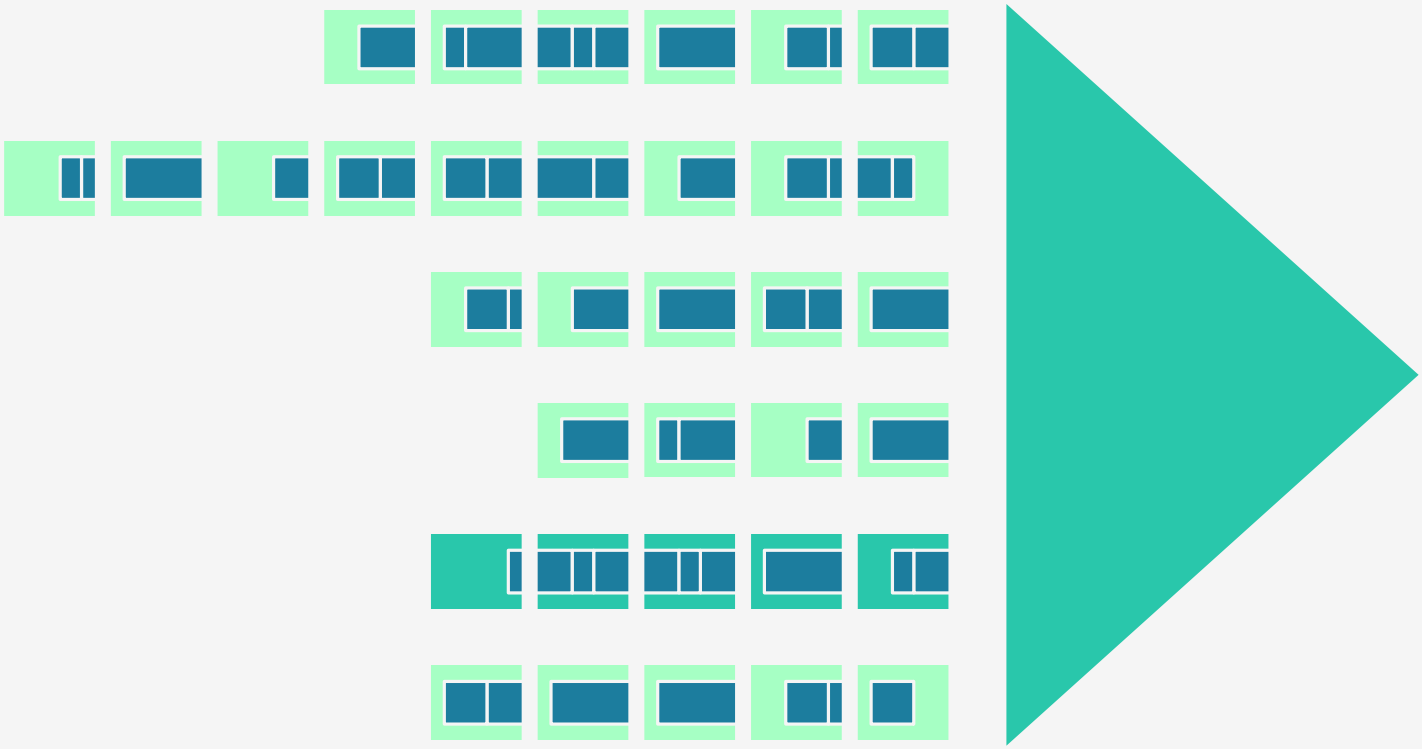


CAMP-MALLOC

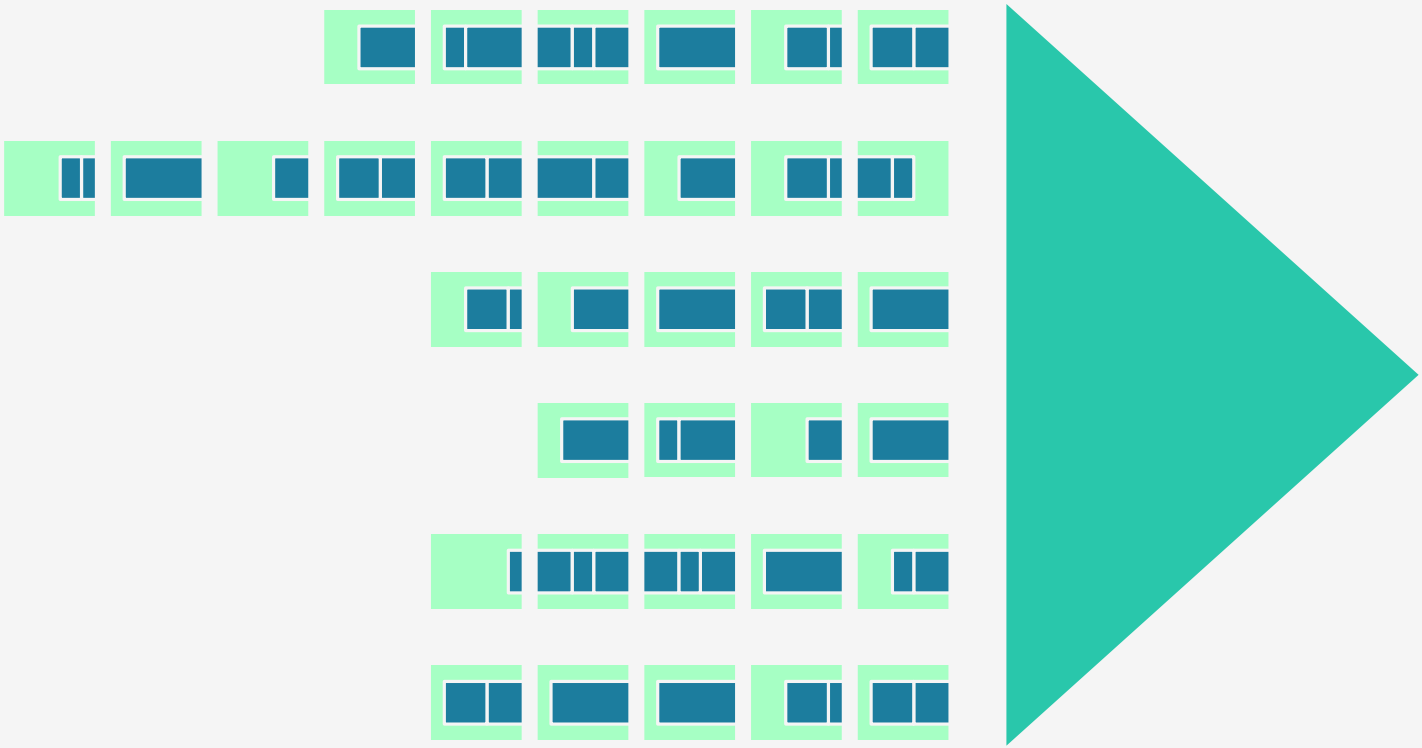
CAMP-MALLOC



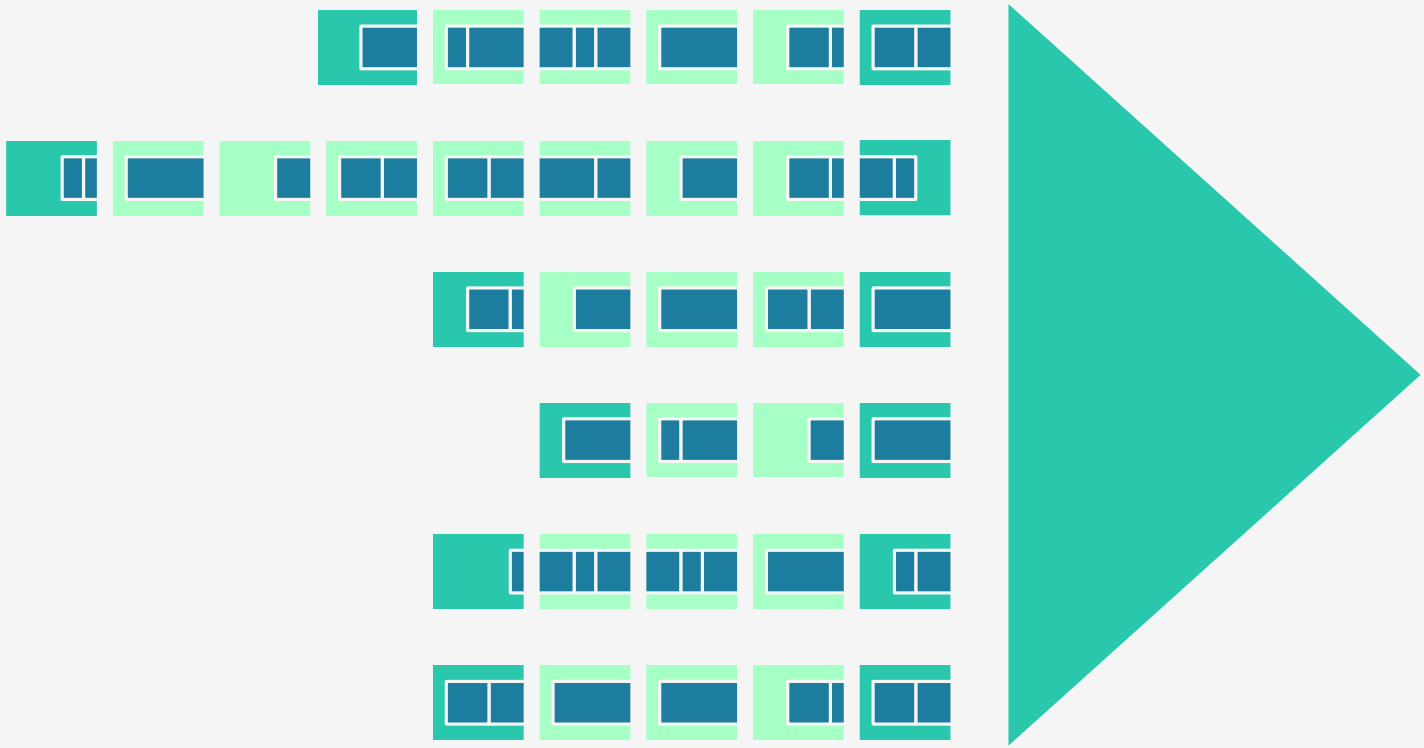
CAMP-MALLOC



CAMP-MALLOC



CAMP-MALLOC



fragmentation $\leq$ 2 (num queues) (block size) + (num blocks) (max item size)

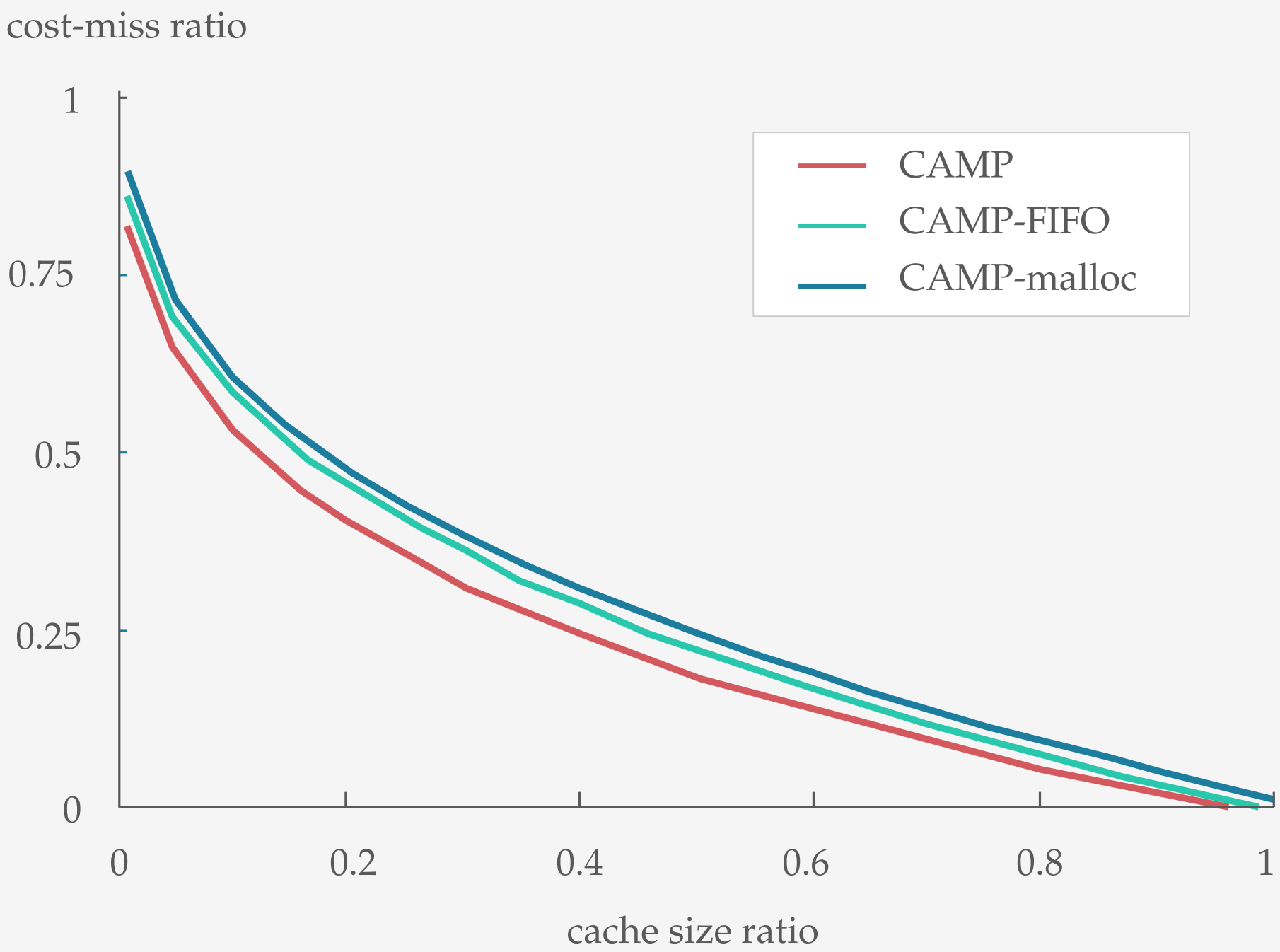
CAMP-MALLOC

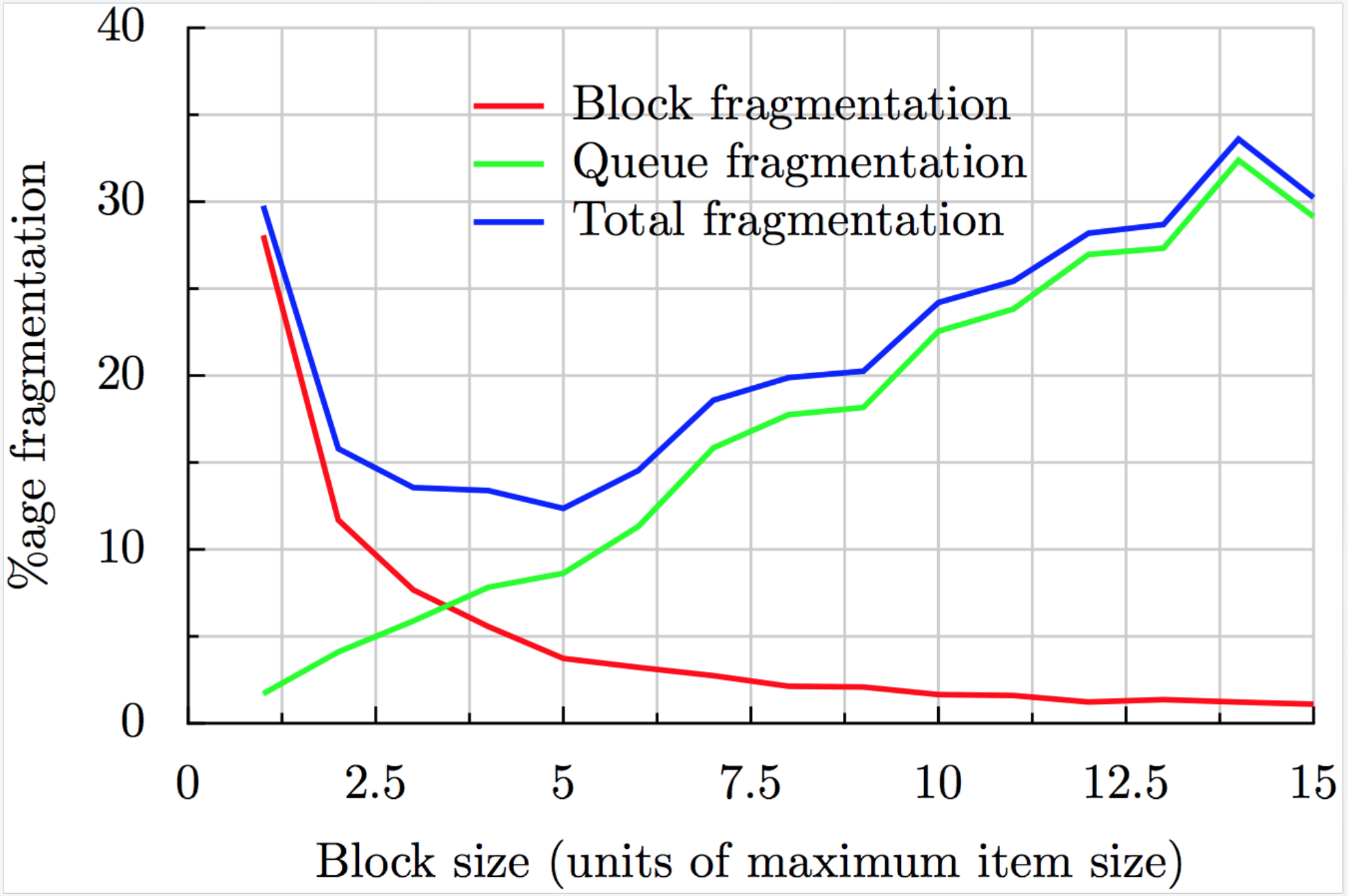
is competitive if memory augmented

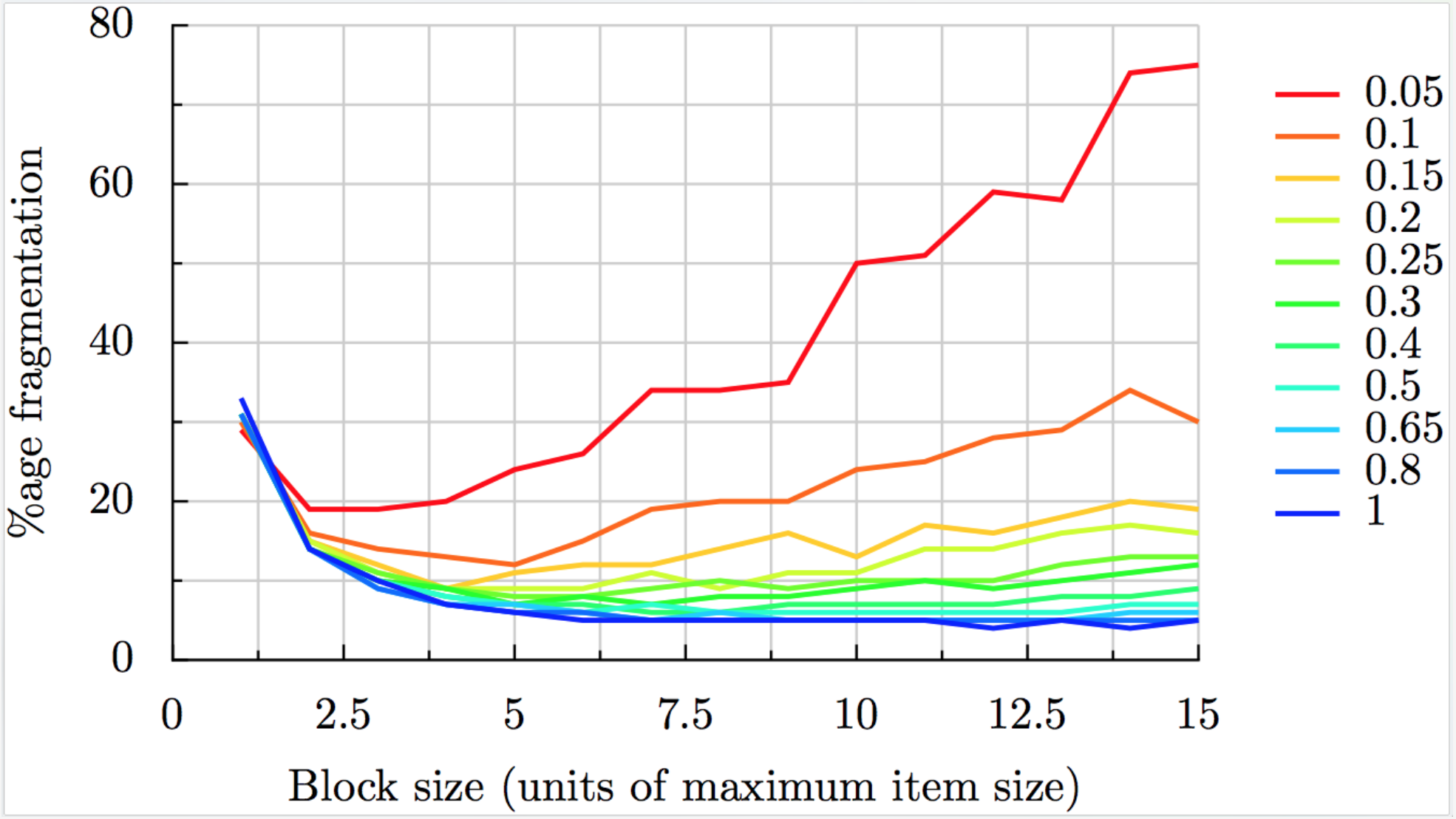
if $\text{OPT's cache size} \leq \text{ALG's cache size} - \text{fragmentation bound}$

then $\text{cost(ALG)} \leq \frac{\text{ALG's cache size}}{\text{min item size}} \text{cost(OPT)}$

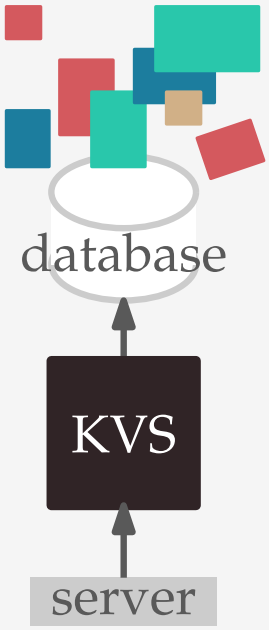
fragmentation $\leq$ 2 (num queues) (block size) + (num blocks) (max item size)


cost-miss ratio
1
0.75
0.5
0.25
0
0
0.2
0.4
0.6
0.8
1
cache size ratio
CAMP
CAMP-FIFO
CAMP-malloc


Block fragmentation
Queue fragmentation
Total fragmentation
%age fragmentation
Block size (units of maximum item size)
0
10
20
30
40
0
2.5
5
7.5
10
12.5
15

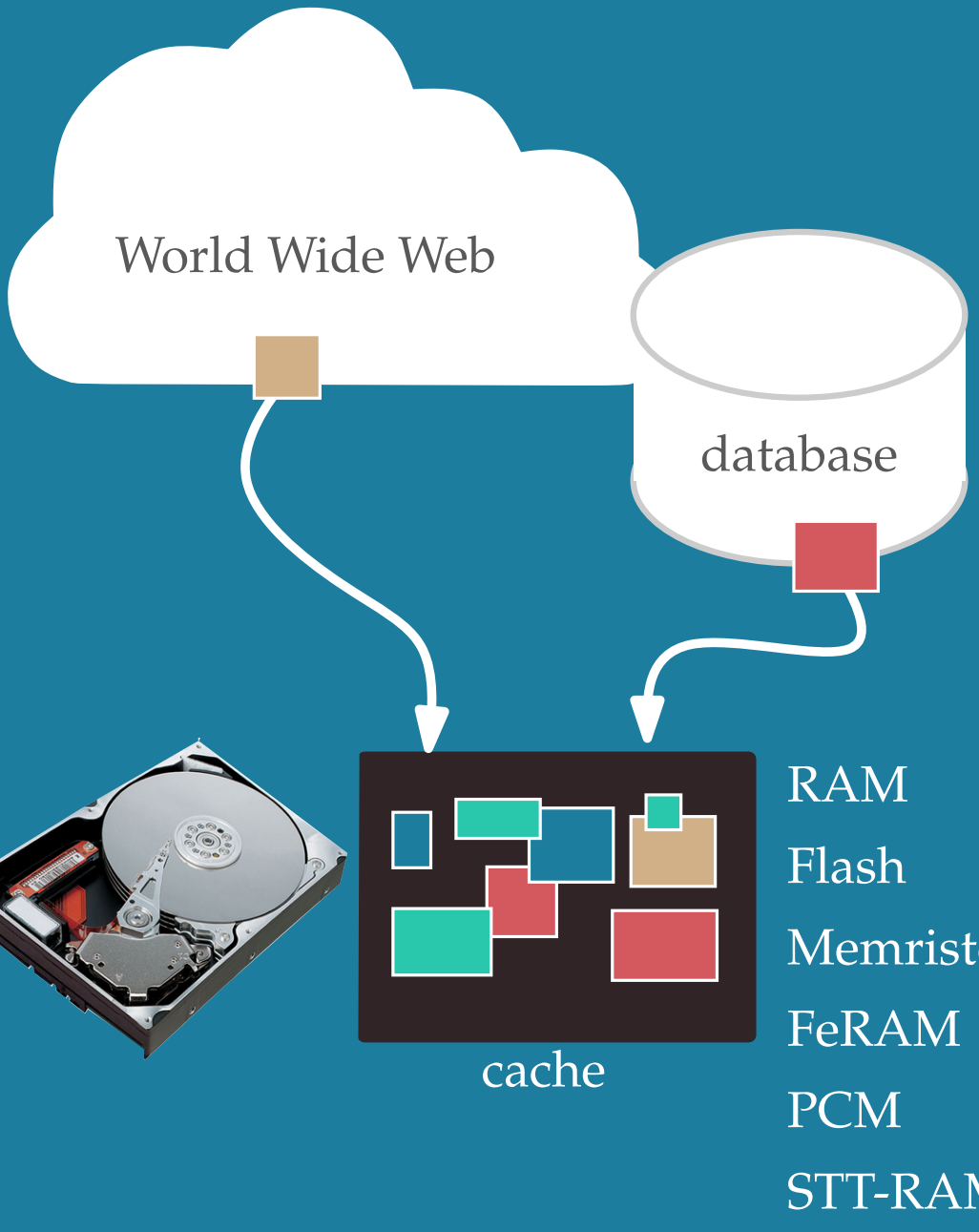

80
60
40
20
0
%age fragmentation
0
2.5
5
7.5
10
12.5
15
Block size (units of maximum item size)
0.05
0.1
0.15
0.2
0.25
0.3
0.4
0.5
0.65
0.8
1

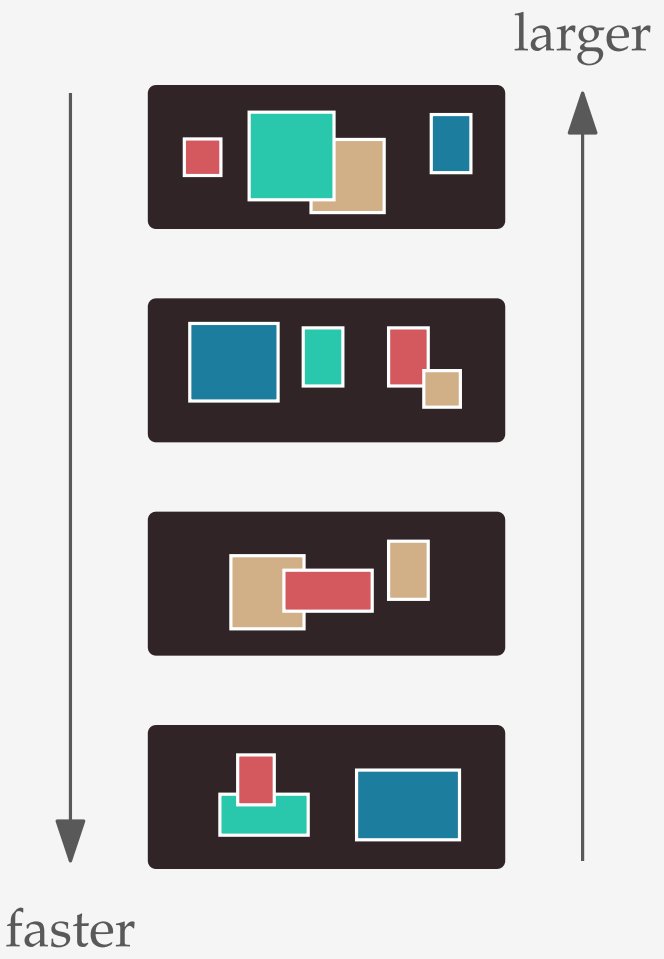
GDS → CAMP



generalized caching → managed memory caching

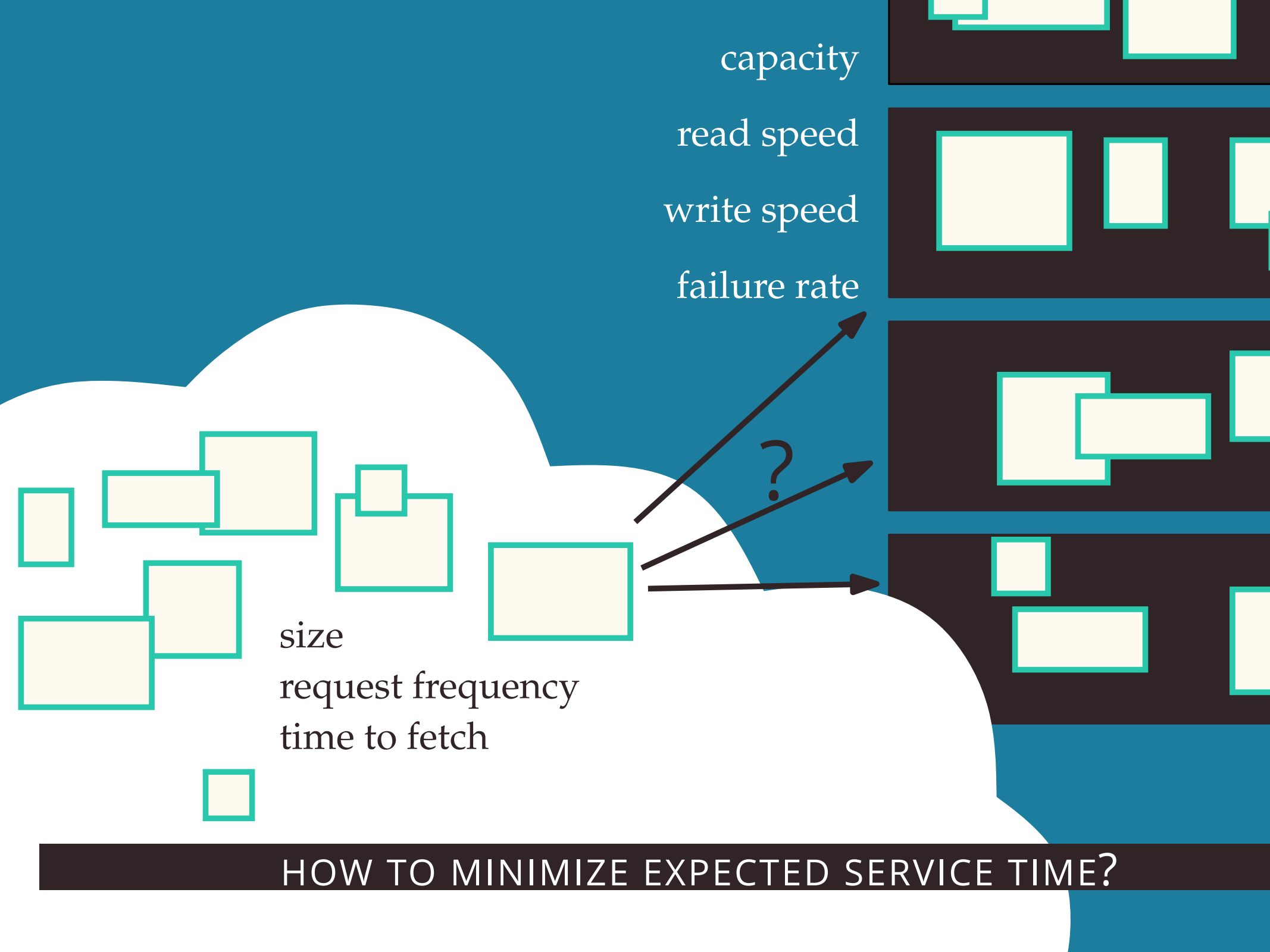
2-level cache → multi-level cache



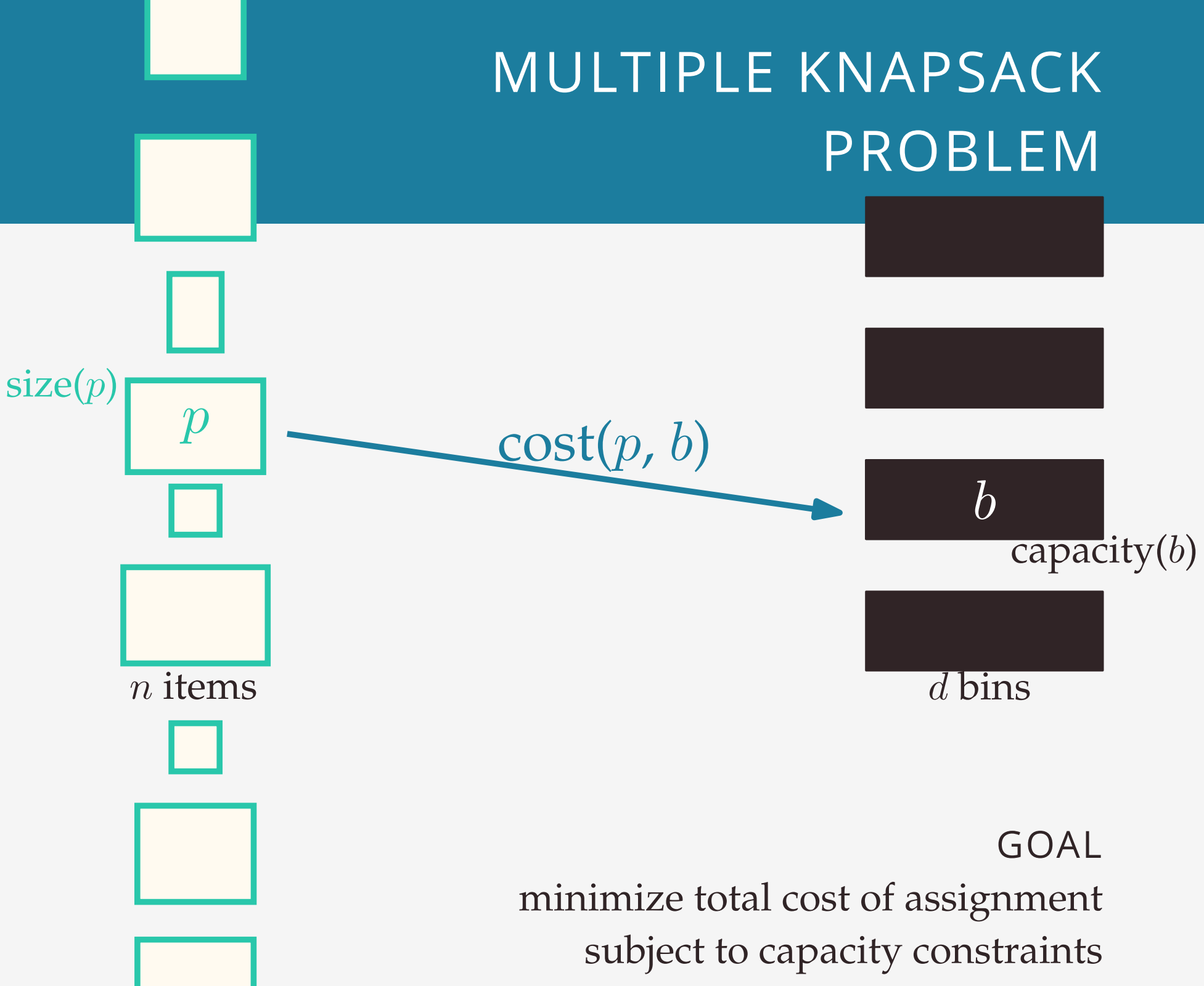


TIERED MEMORY ARCHITECTURE

GENERALIZED CACHING


capacity
read speed
write speed
failure rate
?
size
request frequency
time to fetch
HOW TO MINIMIZE EXPECTED SERVICE TIME?

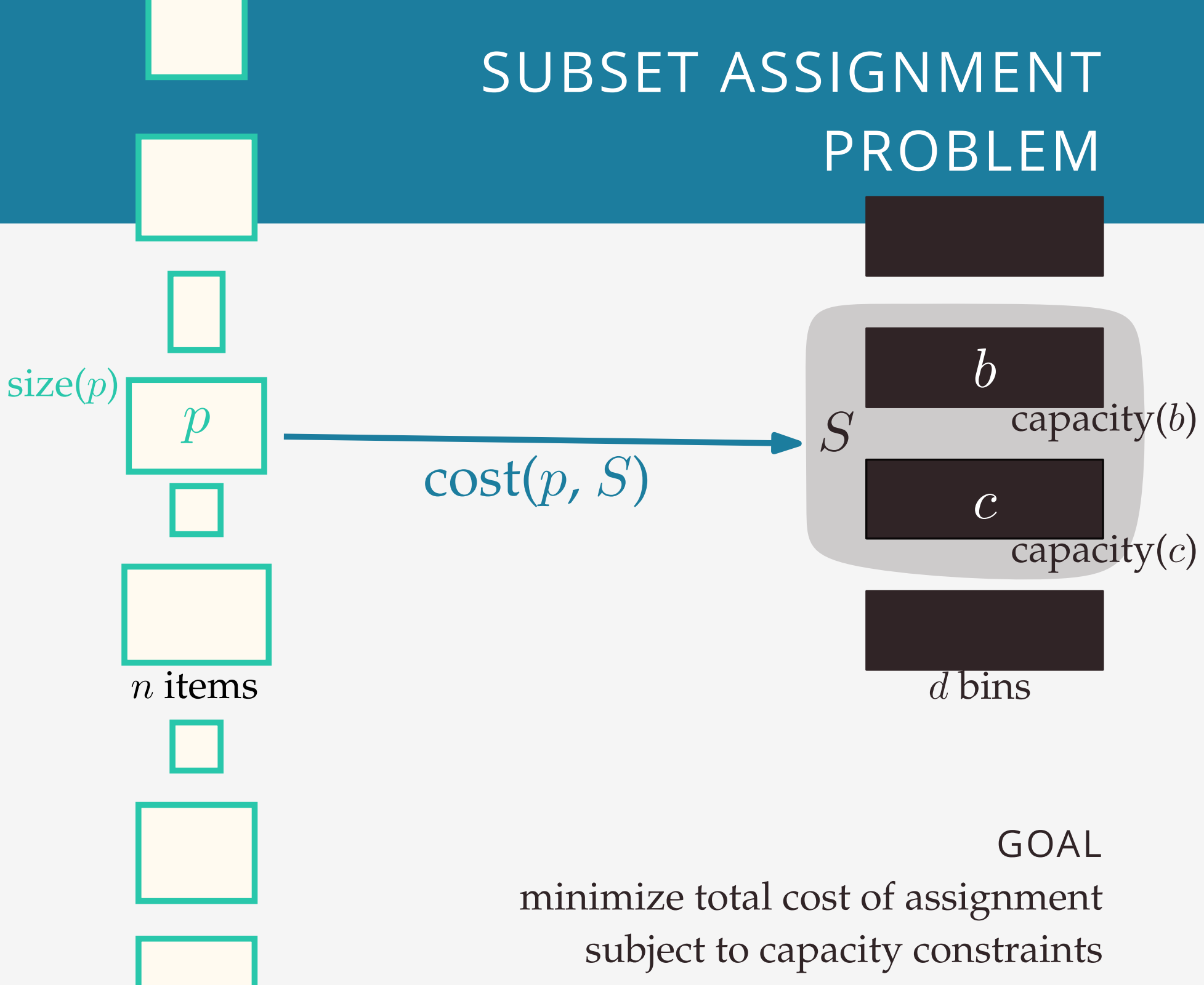
MULTIPLE KNAPSACK PROBLEM



GOAL

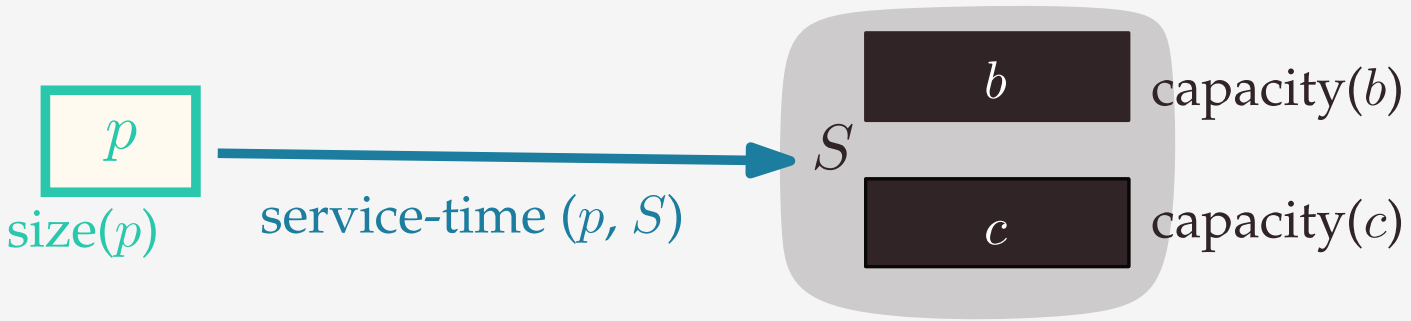
minimize total cost of assignment
subject to capacity constraints

SUBSET ASSIGNMENT PROBLEM

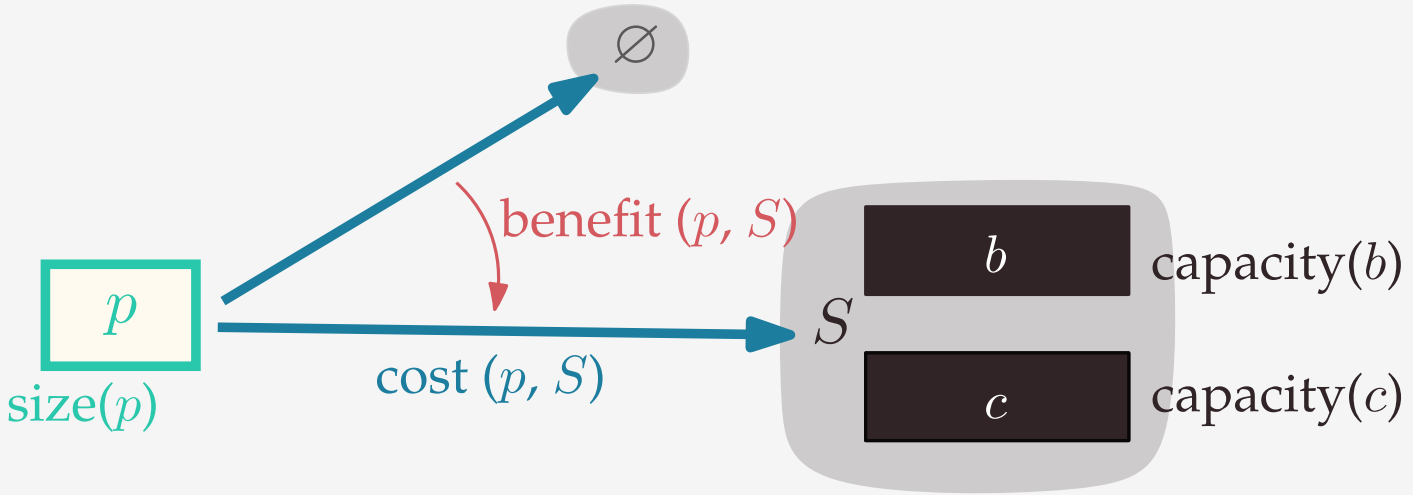


GOAL

minimize total cost of assignment
subject to capacity constraints



$$
\begin{aligned}
\text{service-time}(p, S) \;=\;& \text{read-frequency}(p) \;\; \text{read-time}(p, S) \\
&+ \;\; \text{write-frequency}(p) \;\; \text{write-time}(p, S) \\
&+ \sum_{F \subseteq S} \text{fail-freq}(F) \;\; \Big(\text{read-time}(p, S \setminus F) \\
&\qquad\qquad + \;\; \text{write-time}(p, S \cap F) \Big)
\end{aligned}
$$



cache configuration

$$\text{maximize} \sum_{p,S} \text{benefit}(p,S)\, x(p,S)$$

$$\sum_{S} x(p,S) = 1$$

$$\sum_{p,S} \text{price}(p,S) x(p,S) \leq \text{budget}$$

$$x = 0, 1$$

subset assignment

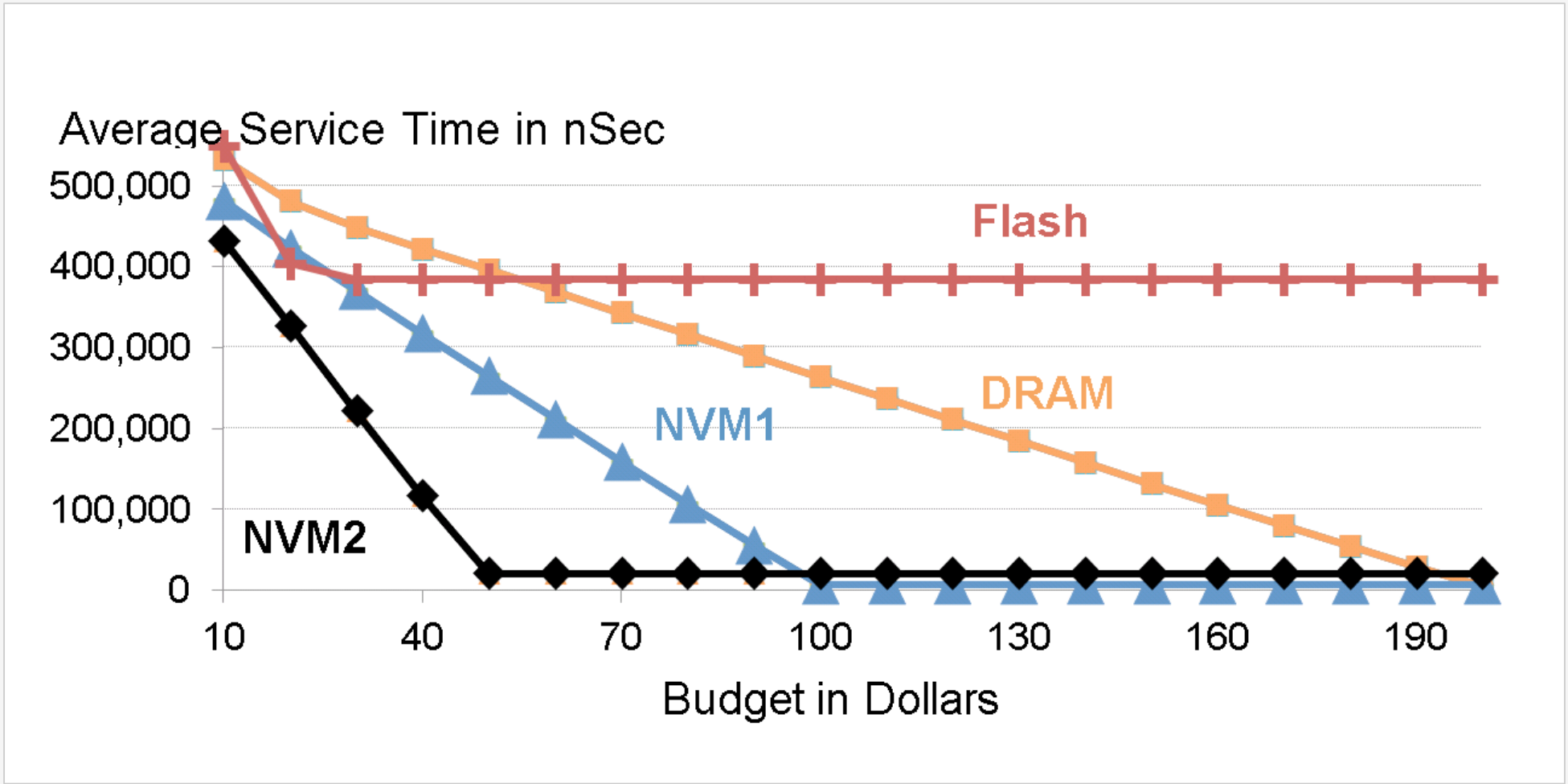
$$\text{minimize} \sum_{p,S} \text{cost}(p,S)\, x(p,S)$$

$$\sum_{S} x(p,S) = \text{size}(p)$$

$$\sum_{p,S \ni b} x(p,S) \leq \text{capacity}(b)$$

$$x(p,S) = 0, \text{size}(p)$$

CACHE CONFIGURATION



SUBSET ASSIGNMENT

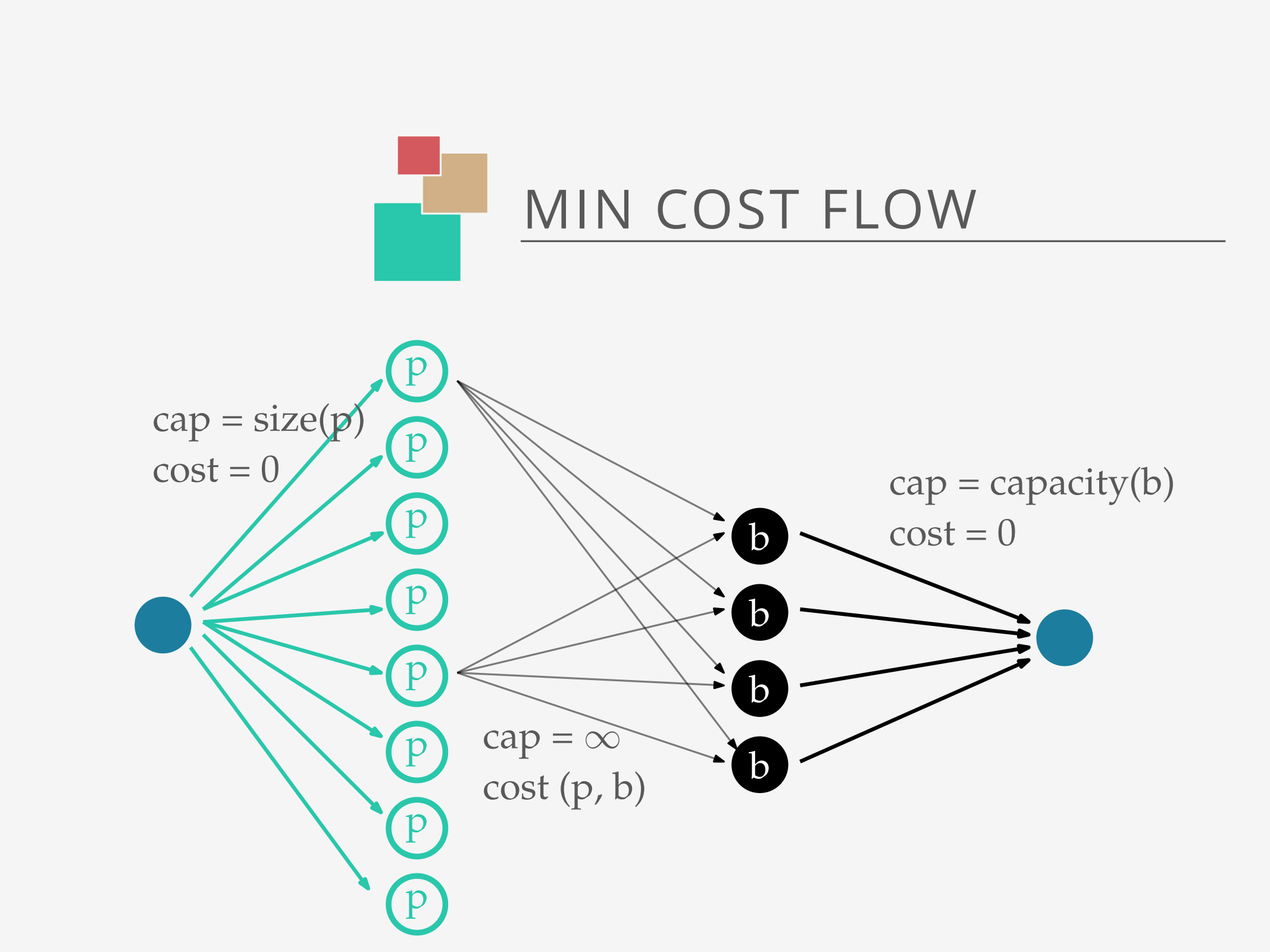
HAVE $d \ll n$

sol to LP relaxation has few fractional assignments

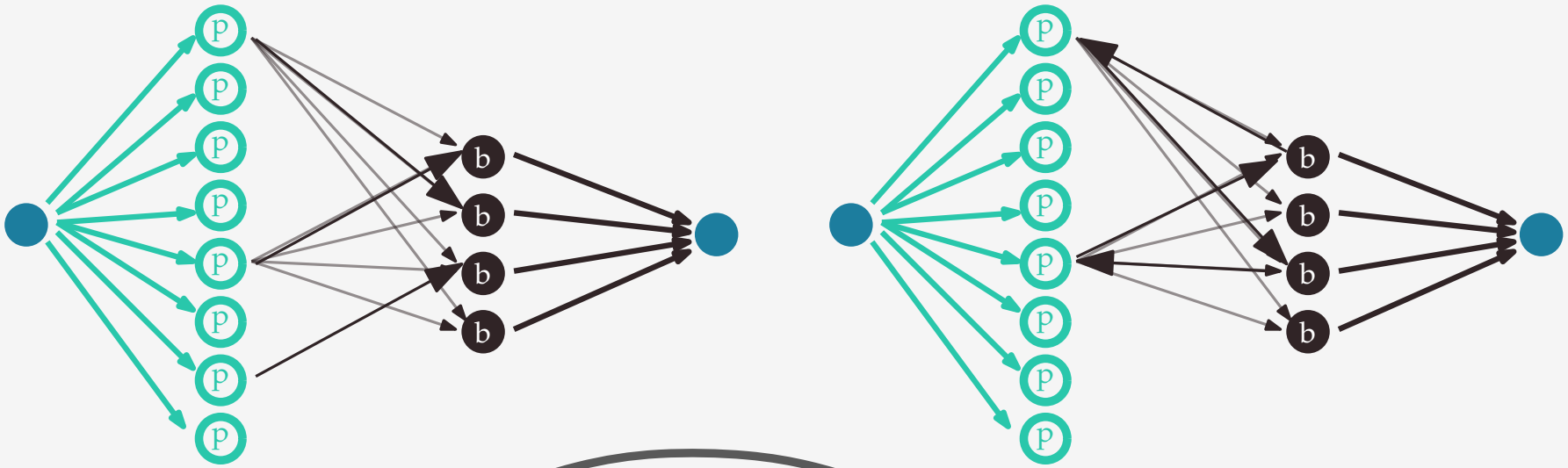
GOAL solve LP relaxation in f(d) poly(n)

1. cycle canceling algorithm

2. simplex algorithm


MIN COST FLOW
cap = size(p)
cost = 0
p
p
p
p
p
p
p
p
cap = ∞
cost (p, b)
b
b
b
b
cap = capacity(b)
cost = 0

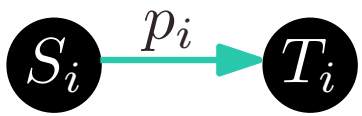
1. cycle canceling algorithm



feasible flow

negative cycle in residual graph

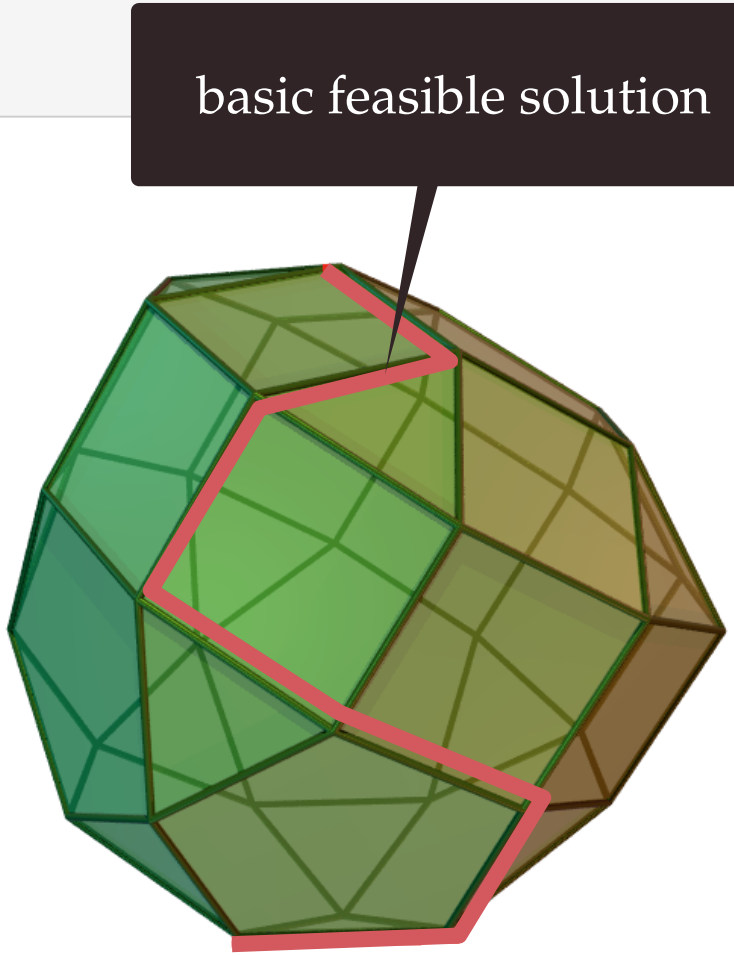
"cycle" in subset assignment problem

augmentation 

such that $\sum_i \alpha_i \overrightarrow{S_i T_i} = \vec{0}$

cost difference (negative) $\sum_i \alpha_i \ (\text{cost}(p_i, T_i) - \text{cost}(p_i, S_i))$

2. simplex algorithm



BASIC FEASIBLE ASSIGNMENT

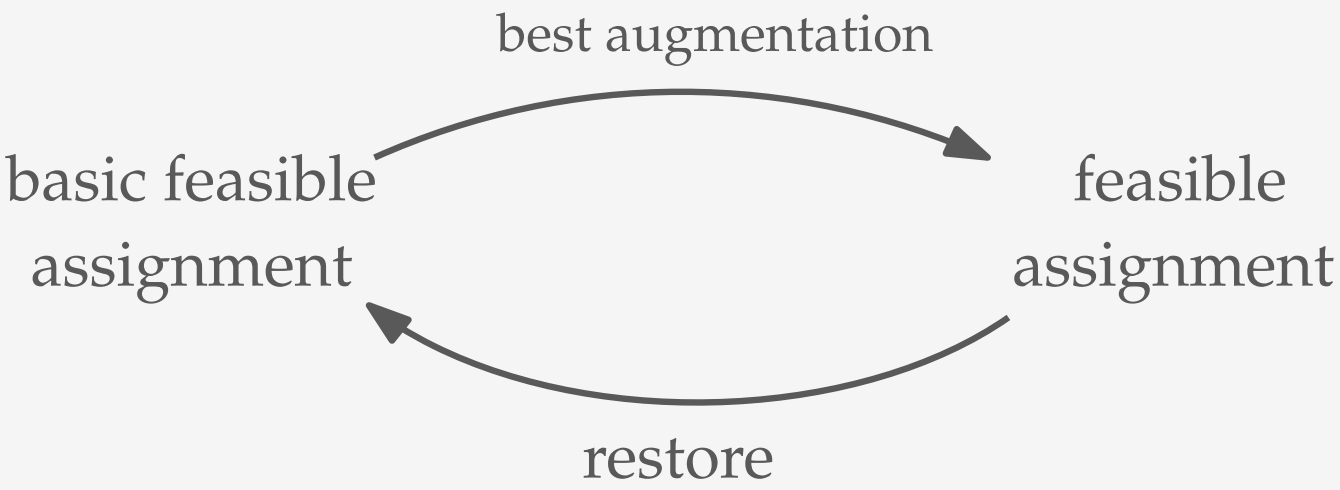
$< 2d$ fractional assignments

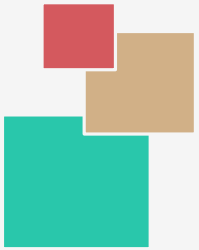
bound granularity of vars

$$x(p, S) = \frac{k}{l}$$

$< d^{d/2}$

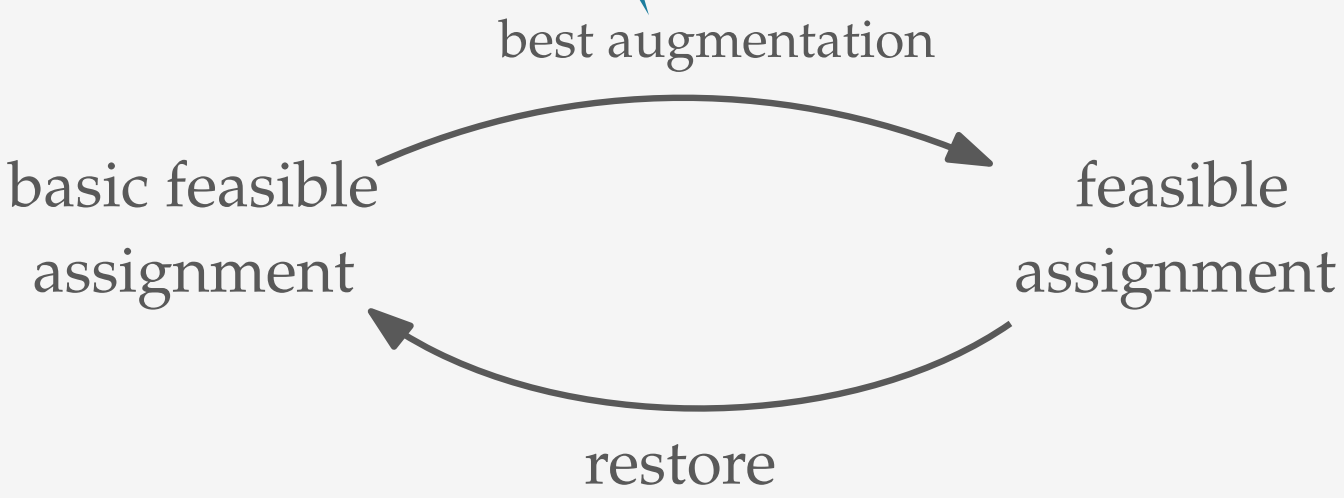
ALGORITHM


best augmentation
basic feasible assignment
feasible assignment
restore

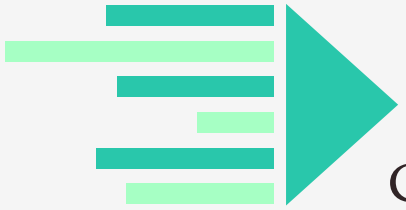


ALGORITHM

preprocessing $\sum_i \alpha_i \overrightarrow{S_i T_i} = \vec{0}$

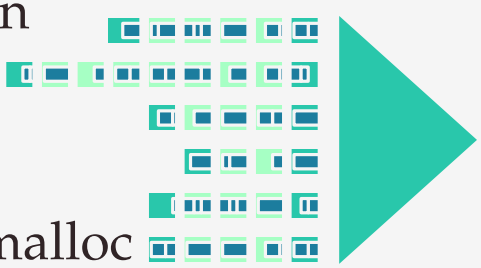


time $O(\exp(d(d+1)\text{poly}(d)\ \ n\log(n)\log(nC)\log(S))$

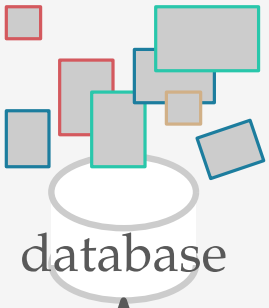


CAMP

caching with memory allocation



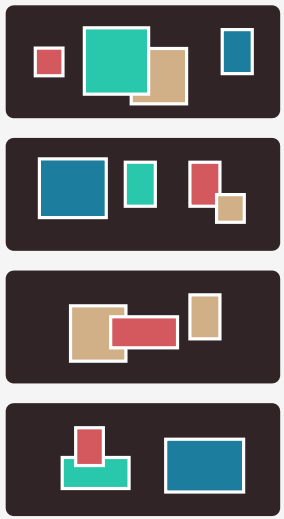
CAMP-malloc



database

KVS

server



multi-level memory

cache configuration

subset asssignment