CACHE REPLACEMENT WITH MEMORY ALLOCATION

Shahram Ghandeharizadeh
Sandy Irani
Jenny Lam

ALENEX 2015 — January 5, 2015
THE PAGING PROBLEM

GOAL
minimize number of cache misses
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
PAGING PROBLEM

main memory

GENERALIZED CACHING

webproxy

MANAGED MEMORY CACHING PROBLEM
SIMPLE ALGORITHM

for the managed memory caching problem

COMPETITIVE RATIO

if augmented memory against optimal offline algorithm for the generalized caching problem

 EXPERIMENTAL RESULTS
GDS
Greedy-Dual Size

CAMP
Cost-adaptive multiqueue eviction policy

OUR ALGORITHM
GDS priority

\[
\frac{\text{cost}(p)}{\text{size}(p)} + \text{lowest priority}
\]
CAMP

GDS priority

\[
\frac{\text{cost}(p)}{\text{size}(p)} + \text{lowest priority}
\]

LRU queues

evict min
OUR ALGORITHM

LRU queues
OUR ALGORITHM

FIFO queues
FIFO queue
OUR ALGORITHM

FIFO queue
OUR ALGORITHM

FIFO queue
OUR ALGORITHM

FIFO queue
OUR ALGORITHM

FIFO queue
FIFO queue
OUR ALGORITHM

FIFO queue
OUR ALGORITHM

FIFO queue
OUR ALGORITHM

FIFO queue
FIFO queue
Our Algorithm

First in

FIFO queue
OUR ALGORITHM

FIFO queue

first in
OUR ALGORITHM

FIFO queue

first in
OUR ALGORITHM

FIFO queue

First in
OUR ALGORITHM

FIFO queue

first in
OUR ALGORITHM

FIFO queue

first in
FIFO queue

first in
OUR ALGORITHM

FIFO queue

first in
OUR ALGORITHM

FIFO queue

first in
OUR ALGORITHM

FIFO queue

first in
OUR ALGORITHM

FIFO queue

fragmentation $\leq 2$ (max item size)
OUR ALGORITHM
OUR ALGORITHM
OUR ALGORITHM
OUR ALGORITHM
OUR ALGORITHM
OUR ALGORITHM
OUR ALGORITHM
OUR ALGORITHM
fragmentation $\leq 2 \times \text{(num queues)} \times \text{(block size)} + \text{(num blocks)} \times \text{(max item size)}$
Our algorithm is competitive if memory augmented

\[
\begin{align*}
\text{if} & \quad \text{OPT's cache size} \leq \text{ALG's cache size} - \text{fragmentation bound} \\
\text{then} & \quad \text{cost(ALG)} \leq \frac{\text{ALG's cache size}}{\min \text{item size}} \cdot \text{cost(OPT)}
\end{align*}
\]

\[
\text{fragmentation} \leq 2 \cdot (\text{num queues}) \cdot (\text{block size}) + (\text{num blocks}) \cdot (\text{max item size})
\]
EXPERIMENTS

trace generated by BG, a social networking benchmark
4 million requests
i.i.d. with 70% of requests to 20% of items

20,000 items

costs  sizes
CONCLUSION

biggest performance degradation
going from LRU to FIFO queues

FUTURE RESEARCH

design a better algorithm for
the managed-memory caching problem
allowing moving items once in cache