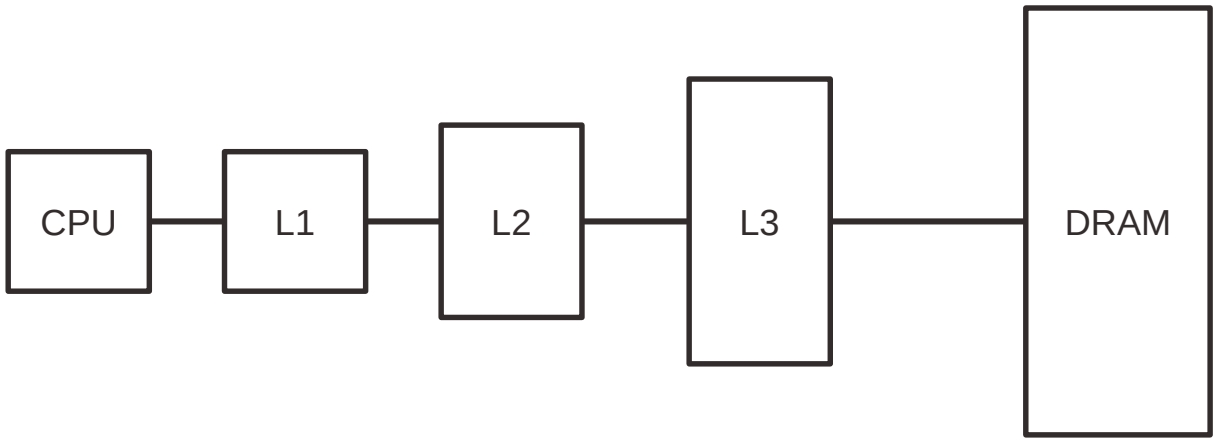
A Study of Perfect Cache Management

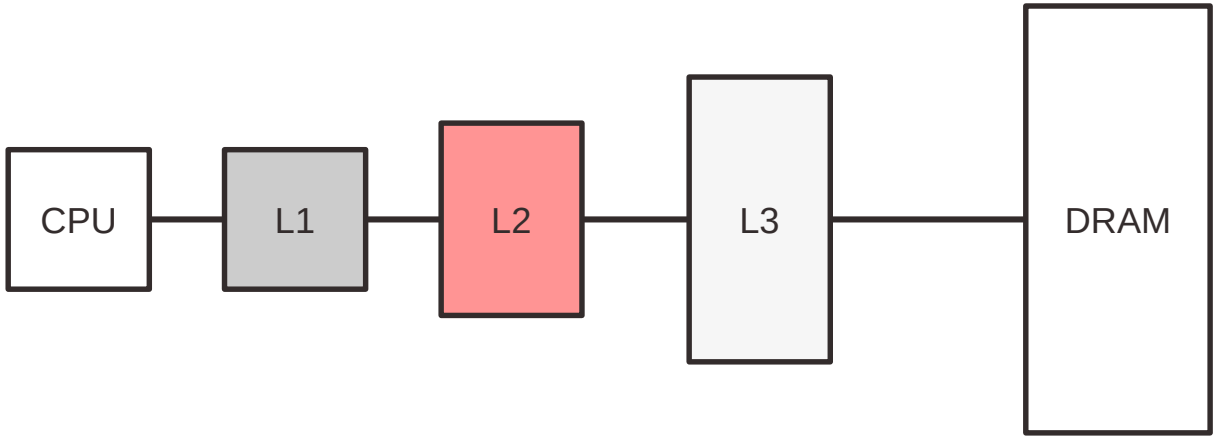
Gino A. Chacon, Daniel Jimenez, and Paul V. Gratz
Texas A&M University


CPU
L1
L2
L3
DRAM

[1978, Smith]

[2004, Nesbit, Dhodapkar, Smith]

[2004, Kim, Chandra, Solihin]



[1978, Smith]

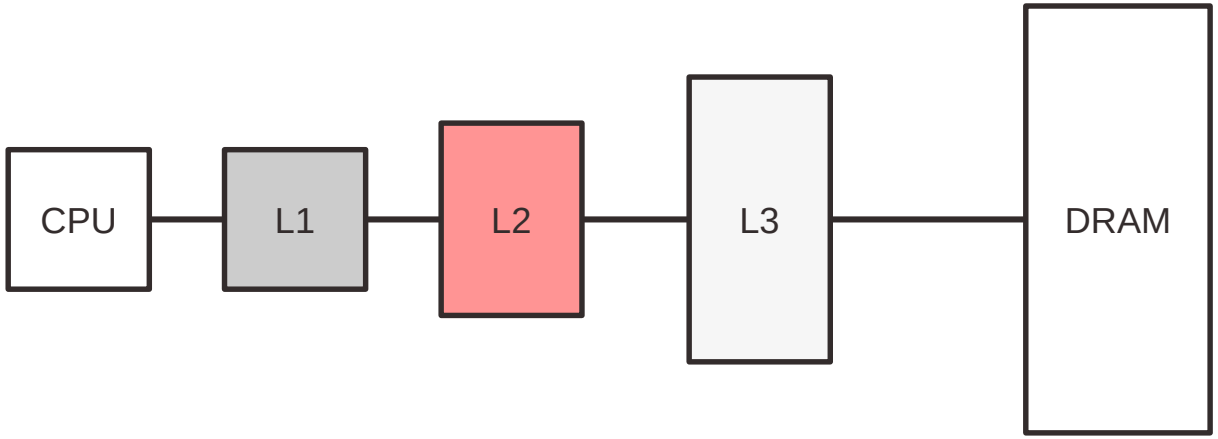
[1985, Smith, Goodman]

[2004, Nesbit, Dhodapkar, Smith]

[2004, Qureshi, Et. Al.]

[2004, Kim, Chandra, Solihin]

[2008, Jaleel Et. Al]



[1978, Smith]

[1985, Smith, Goodman]

[1991, Baer, Chen]

[2004, Nesbit, Dhodapkar, Smith]

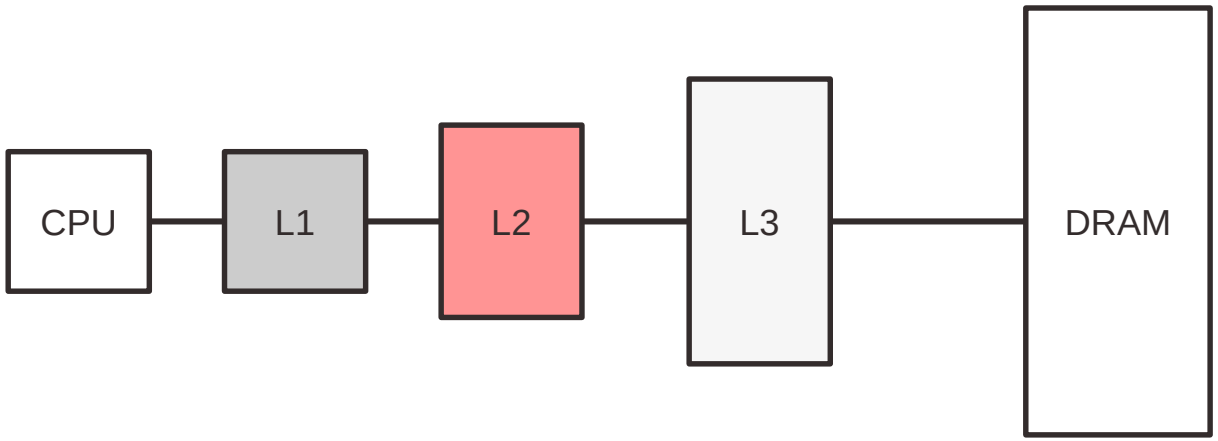
[2004, Qureshi, Et. Al.]

[2010, Khan, Jimenez]

[2004, Kim, Chandra, Solihin]

[2008, Jaleel Et. Al]

[2009, Chaudhur]



[1978, Smith]

[1985, Smith, Goodman]

[1991, Baer, Chen]

[1992, Fu, Patel, Janssens]

[2004, Nesbit, Dhodapkar, Smith]

[2004, Qureshi, Et. Al.]

[2013, Jain, Lin]

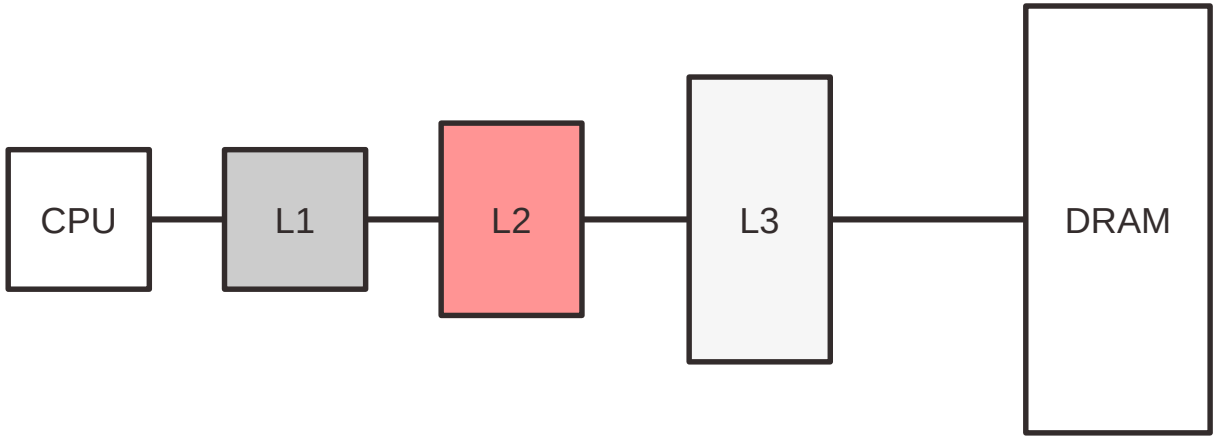
[2004, Kim, Chandra, Solihin]

[2008, Jal

[2010, Khan, Jimenez]

[2010, Jaleel, Et. Al.]

Chaudhur]



[1978, Smith]

[1985, Smith, Goodman]

[1991, Baer, Chen]

[1992, Fu, Patel, Janssens]

[1995, Chen, Baer]

[2004, Nesbit, Dhodapkar, Smith]

[2004, Qureshi, Et. Al.]

[2004, Kim, Chandra, Solihin]

[2008, Jal

Chaudhur]

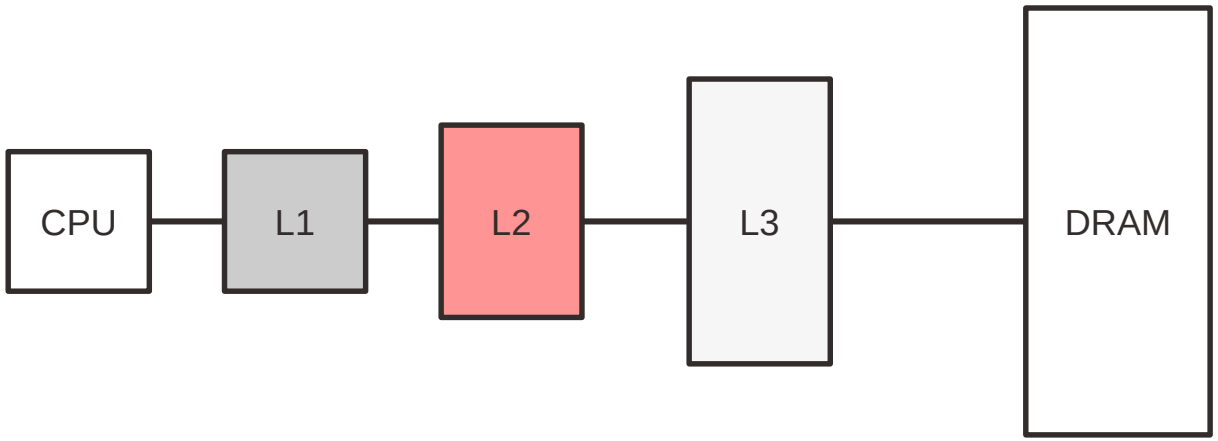
[2010, Jaleel, Et. Al.]

[2010, Khan, Jimenez]

[2011, Gaur, Chadhur, Subramoney]

[2013, Jain, Lin]

[2016, Kim, Et. Al.]



[1978, Smith]

[1985, Smith, Goodman]

[1991, Baer, Chen]

[1992, Fu, Patel, Janssens]

[1995, Che

[2006, Somogyi, Et. Al.]

[2004, Nesbit, Dhodapkar, Smith]

[2004, Qureshi, Et. Al.]

[2013, Jain, Lin]

[2015, Michaud]

[2004, Kim, Chandra, Solihin]

[2008, Jal

Chaudhur]

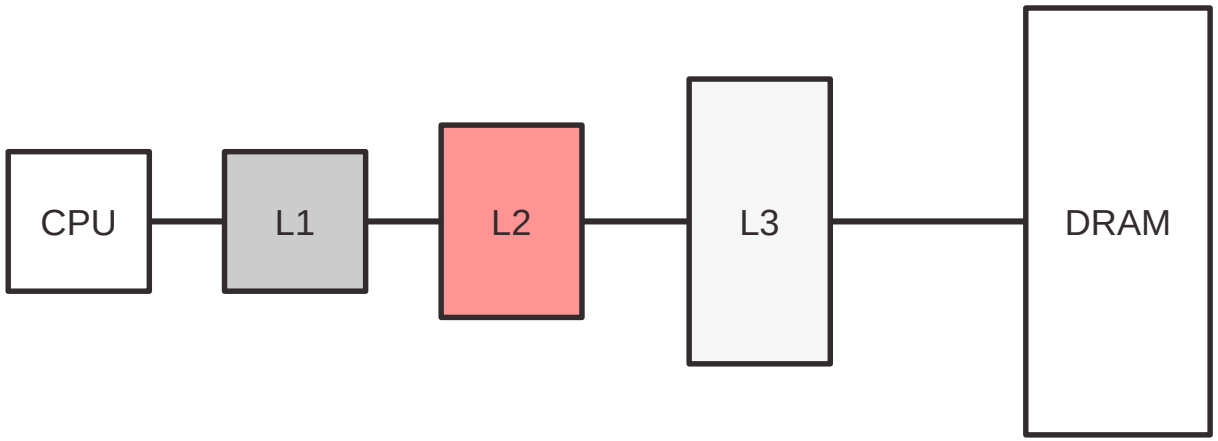
[2010, Jaleel, Et. Al.]

[2010, Khan, Jimenez]

[2016, Kim, Et. Al.]

[2011, Gaur, Chadhur, Subramoney]

[2011, Wu, Et. Al.]



[1978, Smith]

[1985, Smith, Goodman]

[1991, Baer, Chen]

[1992, Fu, Patel, Janssens]

[1995, Che

[2006, Somogyi, Et. Al.]

[2008, Liu, Et. Al.]

[2009, Somogyi]

[2014, Kadjo, Et. Al.]

[2004, Nesbit, Dhodapkar, Smith]

[2004, Qureshi, Et. Al.]

[2013, Jain, Lin]

[2015, Michaud]

[2016, Teran, Et. Al.]

[2017, Kim, Et. Al.]

[2017, Jimenez, Teran]

[2004, Kim, Chandra, Solihin]

[2008, Jal

[2010, Jaleel, Et. Al.]

[2010, Khan, Jimenez]

[2016, Kim, Et. Al.]

[2011, Wu, Et. Al.]

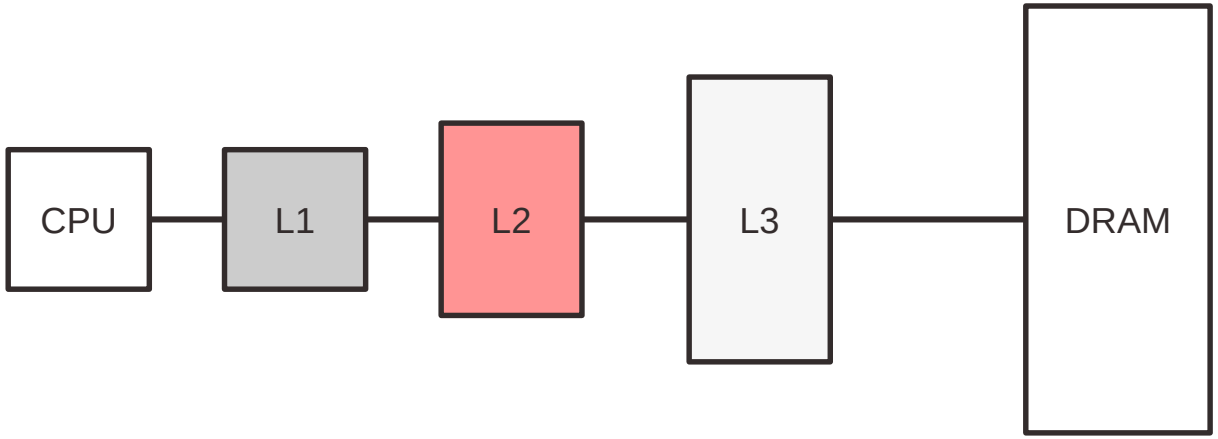
[2012 Duong, Et. Al.]

[2014, Volos]

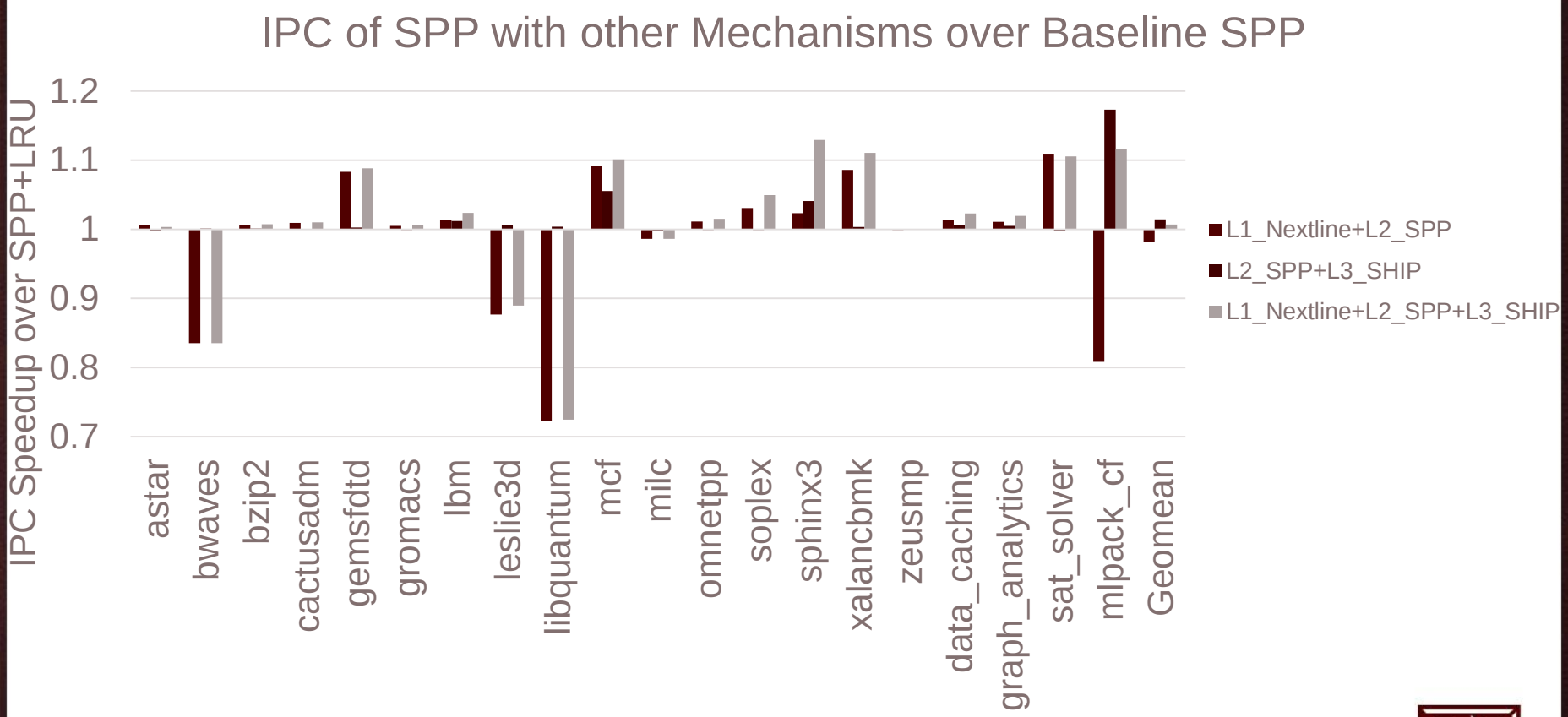
[2015, Shevgoor, Et. Al.]

Chaudhur]

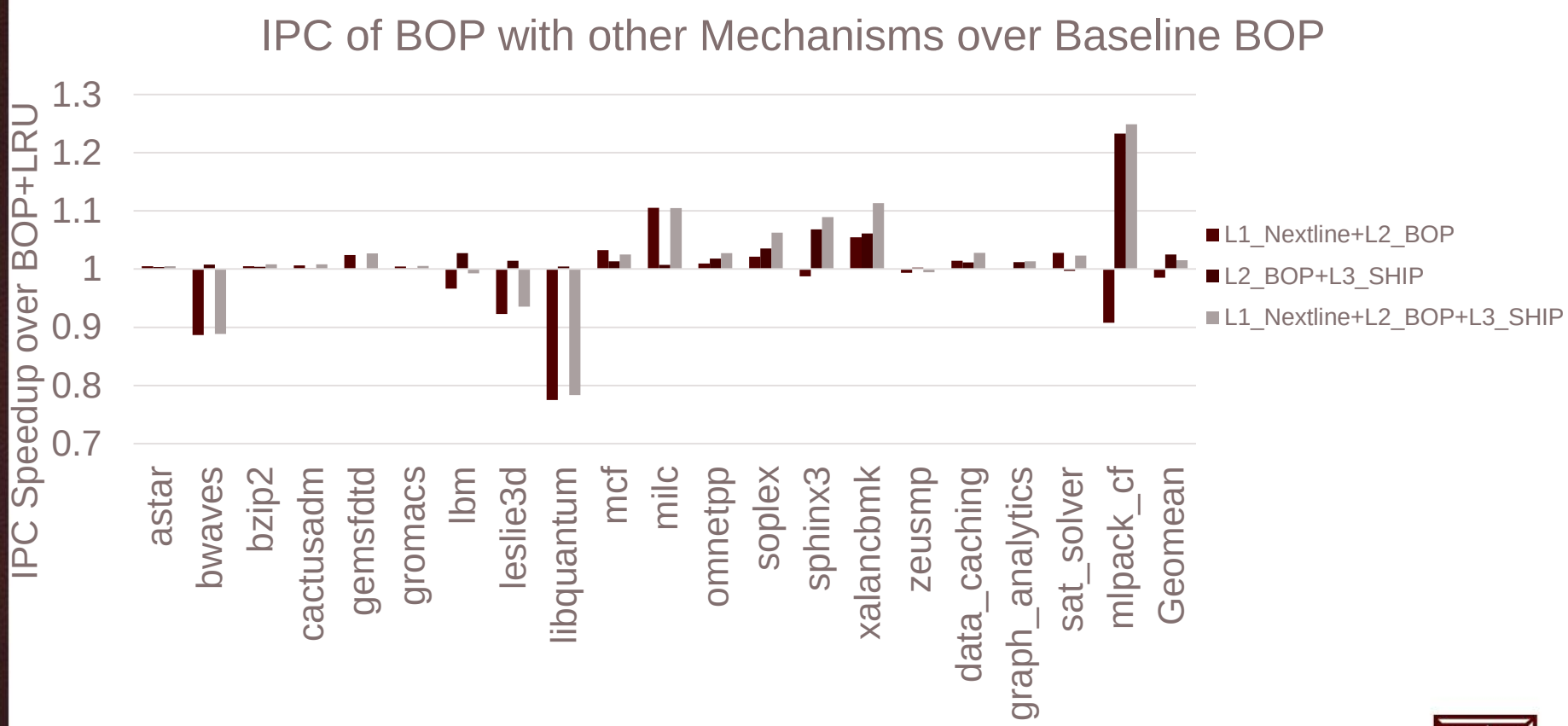
[2011, Gaur, Chadhur, Subramoney]



Multiple Mechanisms in One System



Multiple Mechanisms in One System



Coordinated Cache Management

- Independent management impedes performance improvements
- Create a theoretical framework for perfect management
  - Belady's provides perfect replacement without prefetching
  - Mattson's OPT shows perfect replacement and cache size
  - Recent works do not cover the full hierarchy [2018, Nori, Et Al.][2018, Jain, Lin]
- Create a theoretical framework that treats the hierarchy as a single unit
  - Start with ground-level principles of memory management
  - Purely theoretical
  - Observe behavior and apply it to realistic designs

LRU Without Oracle

Access #	1	2	3	4
Address	D	B	C	D

Access: 0

L2	
Address	LRU
F	3
E	2
D	1
C	0

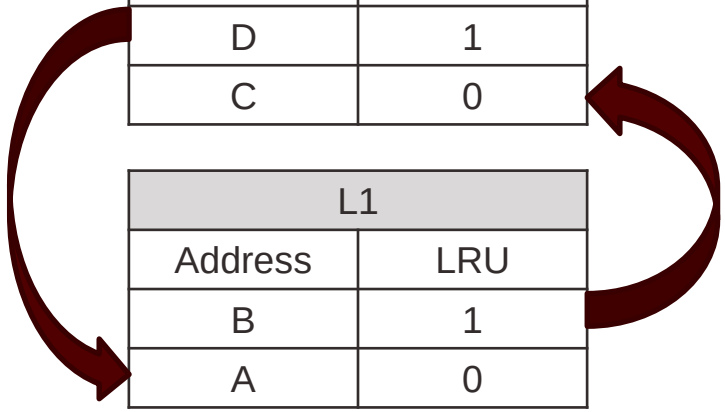
L1	
Address	LRU
B	1
A	0

LRU Without Oracle

Access #	1	2	3	4
Address	D	B	C	D

L2	
Address	LRU
F	3
E	2
D	1
C	0

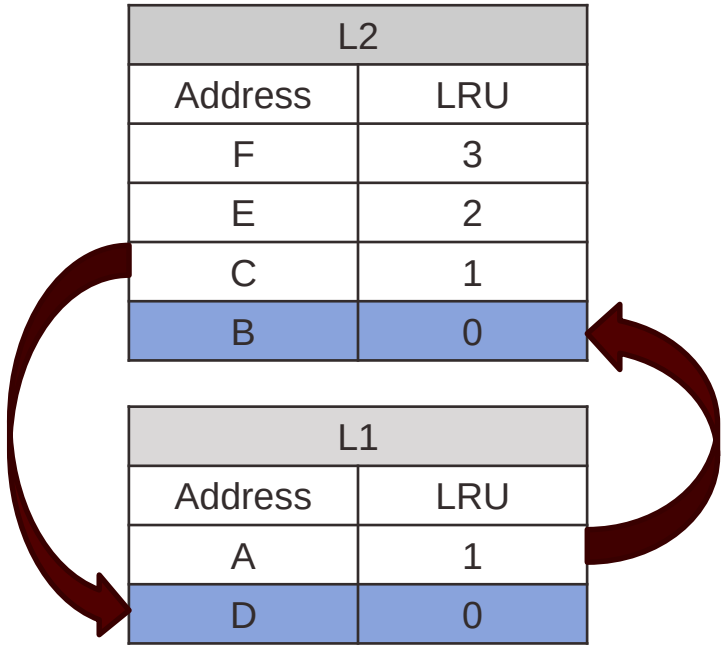
L1	
Address	LRU
B	1
A	0



LRU Without Oracle

Access #	1	2	3	4
Address	D	B	C	D

B is evicted and D is promoted, even though B is accessed sooner.



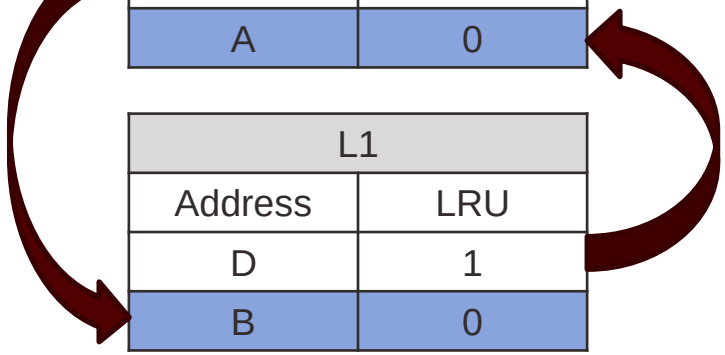
LRU Without Oracle

Access #	1	2	3	4
Address	D	B	C	D

A is evicted to the L2 and B is promoted back into the L1.

L2	
Address	LRU
F	3
E	2
C	1
A	0

L1	
Address	LRU
D	1
B	0



LRU Without Oracle

Access #	1	2	3	4
Address	D	B	C	D

D is evicted back to the L2 and C is brought into the L1.

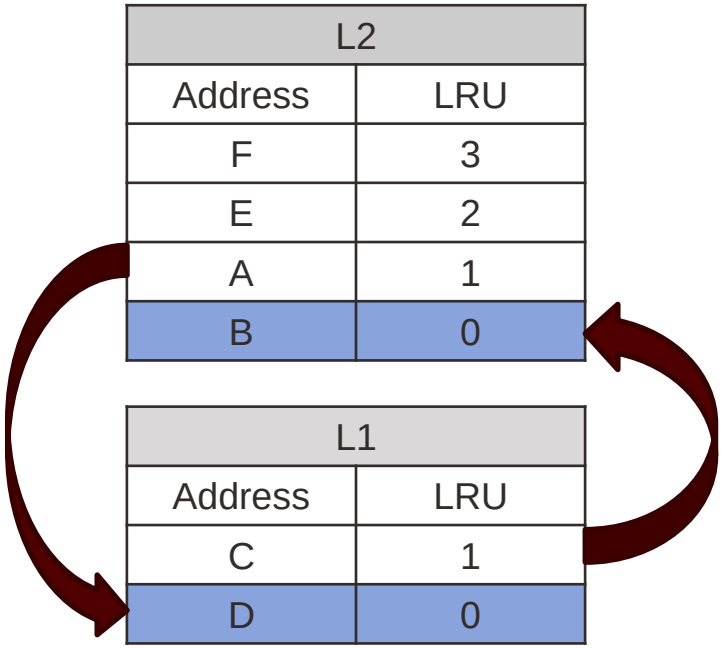
L2	
Address	LRU
F	3
E	2
A	1
D	0

L1	
Address	LRU
B	1
C	0

LRU Without Oracle

Access #	1	2	3	4
Address	D	B	C	D

Again, D is placed back to the L1 and C is brought into the L1.



Common Cache Behavior

- Unnecessary evictions degrade performance
  - Increases with prefetching and multiple mechanisms
- Short example, but occurs with larger caches
  - Streaming
  - Single-touch accesses

Optimal Cache Management

- Each cache level is a sorted list based on future accesses
  - Furthest access reference – Least Valuable Position
  - Nearest access reference – Most Valuable Position
- New data from main memory starts in the L3
  - Moves through cache as its access time approaches
- All memory requests hit in the L1
- Replacement and prefetching operate per-access

Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Access: 0

Reorder the cache

L2	
Address	Next Access
F	8
E	6
D	1
C	3
L1	
Address	Next Access
B	2
A	16

Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Access: 0

Reorder the cache

L2	
Address	Next Access
A	16
F	8
E	6
C	3
L1	
Address	Next Access
B	2
D	1

Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Access D in the L1

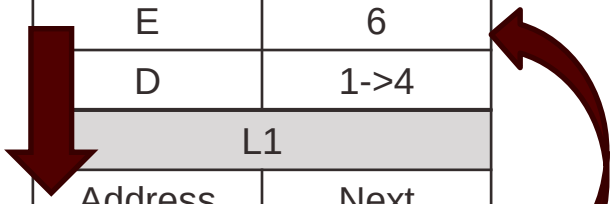
L2	
Address	Next Access
A	16
F	8
E	6
C	3
L1	
Address	Next Access
B	2
D	1

Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Update D's next access and reorder cache

L2	
Address	Next Access
A	16
F	8
E	6
D	1->4
L1	
Address	Next Access
C	3
B	2



Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Access B, update its next access, and reorder the cache

L2	
Address	Next Access
A	16
F	8
E	6
D	4
L1	
Address	Next Access
C	3
B	2 -> 10

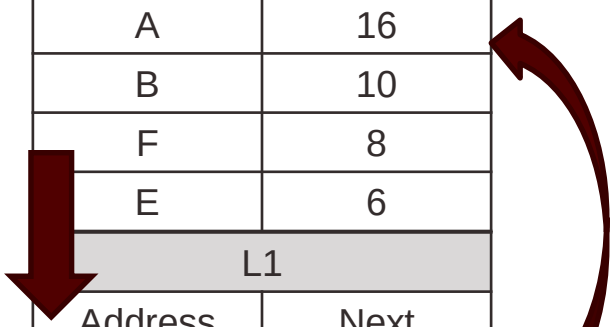
Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Access B, update its next access, and reorder the cache

L2	
Address	Next Access
A	16
B	10
F	8
E	6

L1	
Address	Next Access
D	4
C	3



Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Access C, update its next access, and reorder the cache

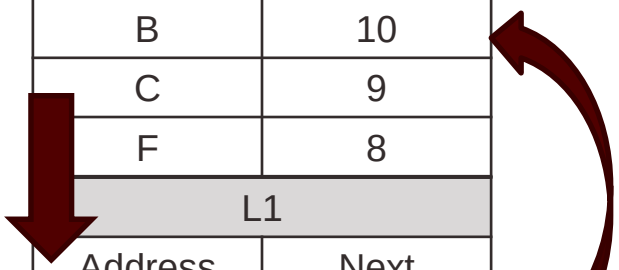
L2	
Address	Next Access
A	16
B	10
F	8
E	6
L1	
Address	Next Access
D	4
C	3 -> 9

Oracle Management

Access #	1	2	3	4
Address	D	B	C	D

Access C, update its next access, and reorder the cache

L2	
Address	Next Access
A	16
B	10
C	9
F	8
L1	
Address	Next Access
E	6
D	4



Observations

- With no latency
  - The MVP is the only replaceable entry in the L1
  - Caches behave as a buffer
  - If number of sets between levels match, they behave as a stack
- With latency
  - Movement between levels still determined by LVP and MVP
  - Causes late migration
  - Delaying requests to allow for migration still increases performance
  - Cache latency is sometimes necessary for performance
- Scheduling
  - Minimize the latency to complete a request
  - Not only a L1 hit, but delivers data on time

Methodology and Future Work

- In-house simulator for limit study
  - Focus on bandwidth and latency modelling
  - Framework for implementing oracle mechanisms
- Next steps
  - Adapt to simulator with accurate CPU model – Champsim
  - Develop scheduling mechanism
- Increased complexity when applied to shared L3

Summary

- Growing complexity requires coordination
  - Cache improvements are beneficial alone, but not always constructive together
- Studying perfect cache management to provide a foundation for future realistic work