Non-uniform cache architecture (NUCA)

Arkaprava Basu

Acknowledgements

- Several of the slides in the deck are from Luis Ceze (Washington), Nima Horanmand (Stony Brook), Mark Hill, David Wood, Karu Sankaralingam (Wisconsin), Abhishek Bhattacharjee(Rutgers).

- Development of this course is partially supported by Western Digital corporations.

Challenges in designing large LLCs

- Larger the cache: Higher is the access time
  - Large cache → higher hit rate but every access becomes slower

Challenges in designing large LLCs

- Larger the cache: Higher is the access time
  - Large cache → higher hit rate but every access becomes slower
- Solution: Break the large cache in smaller caches (banks/slices)

Challenges in designing large LLCs

- Larger the cache: Higher is the access time
  - Large cache → higher hit rate but every access becomes slower

- Solution: Break the large cache in smaller caches (banks/slices)
- But access times to different banks of caches from a core is non uniform due to wire delay
- **Wire delay $T = rcL^2$**

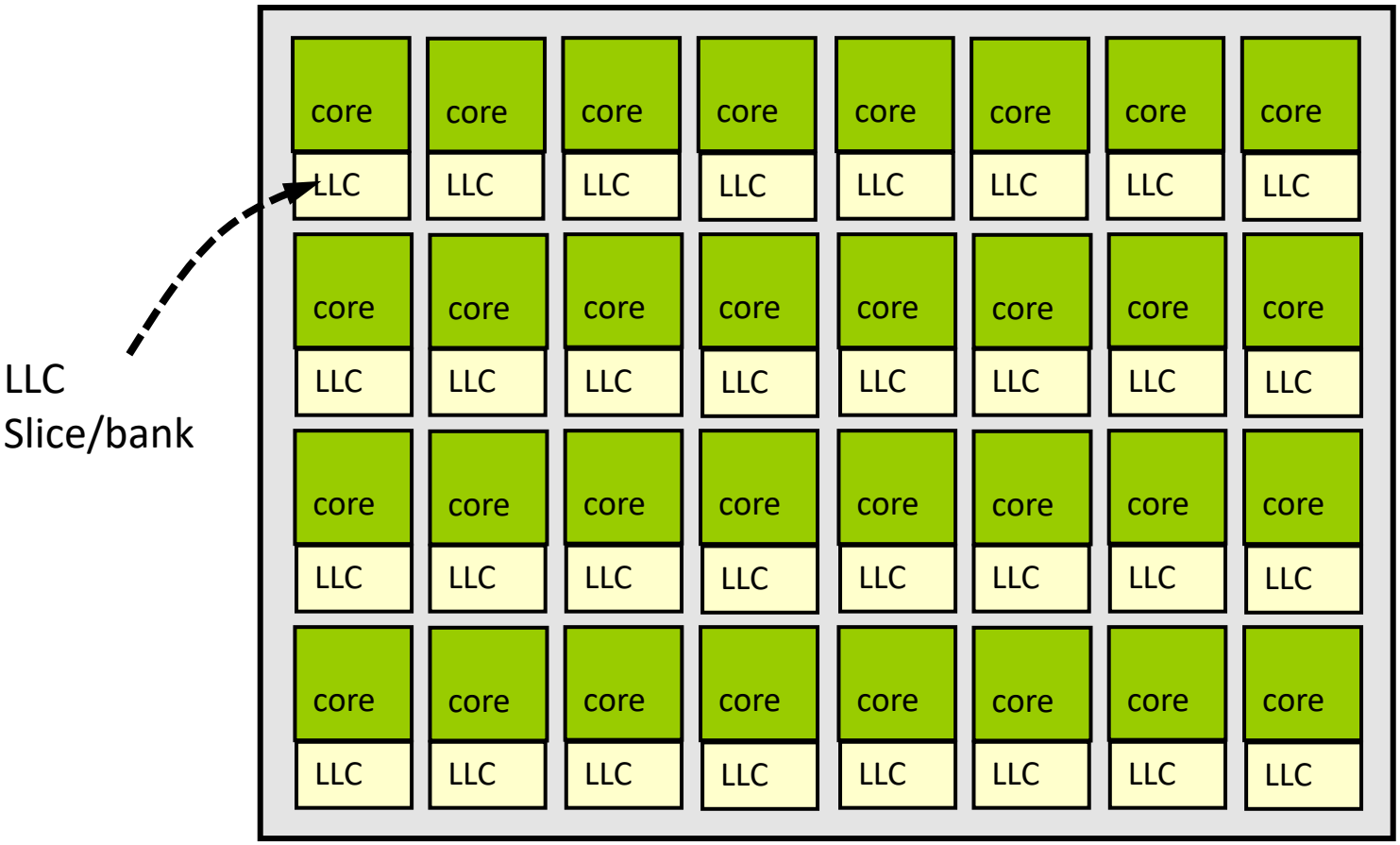
Challenges in designing large LLCs

- Larger the cache: Higher is the access time
  - Large cache → higher hit rate but every access becomes slower
- Solution: Break the large cache in smaller caches (banks/slices)
- But access times to different banks of caches from a core is non uniform due to wire delay
- **Wire delay $T = rcL^2$**
  - Narrower wire → more r and c
  - Bigger chips → longer wires

Challenges in designing large LLCs

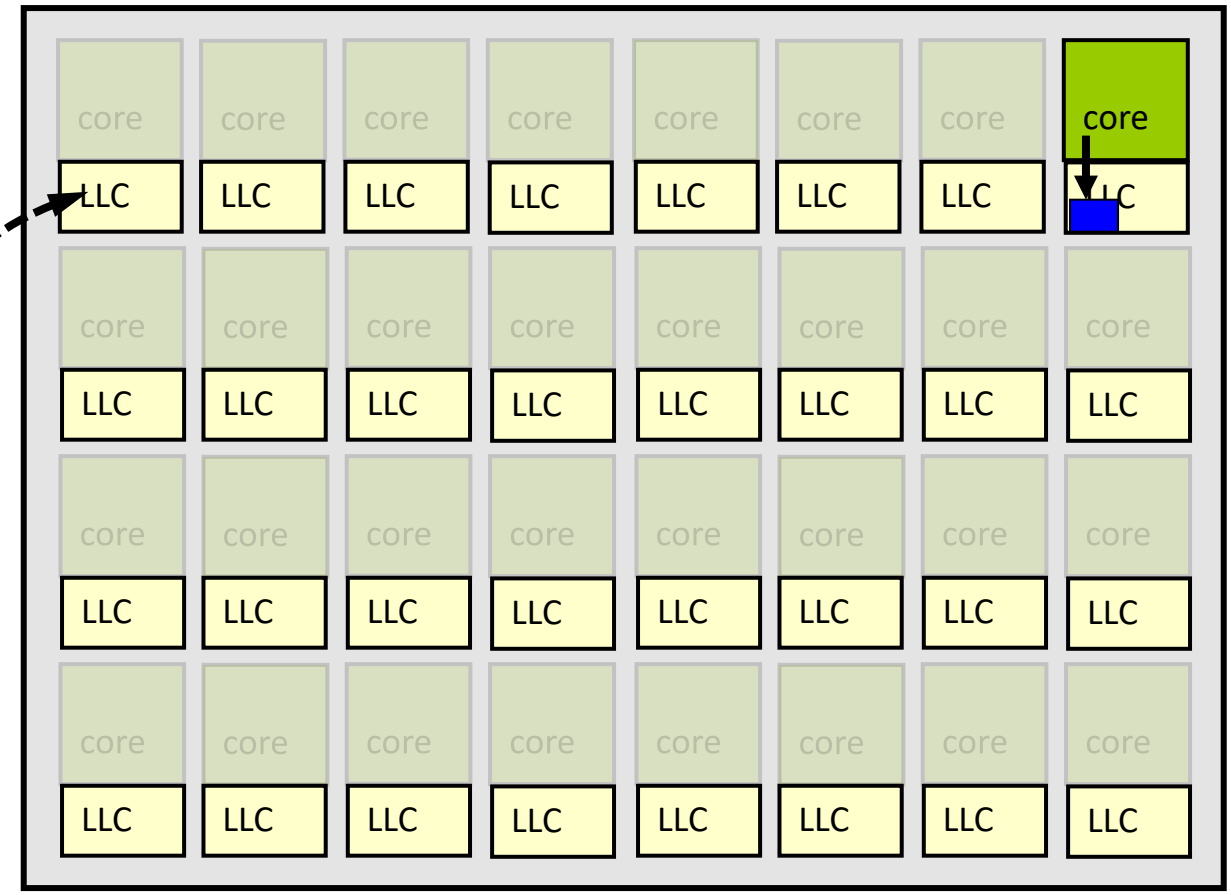
- Larger the cache: Higher is the access time
  - Large cache → higher hit rate but every access becomes slower
- Solution: Break the large cache in smaller caches (banks/slices)
- But access times to different banks of caches from a core is non uniform due to wire delay
- **Wire delay $T = rcL^2$**
  - Narrower wire → more r and c
  - Bigger chips → longer wires
- **Non Uniform Cache Architecture**

Non-Uniform Cache Architecture (NUCA)



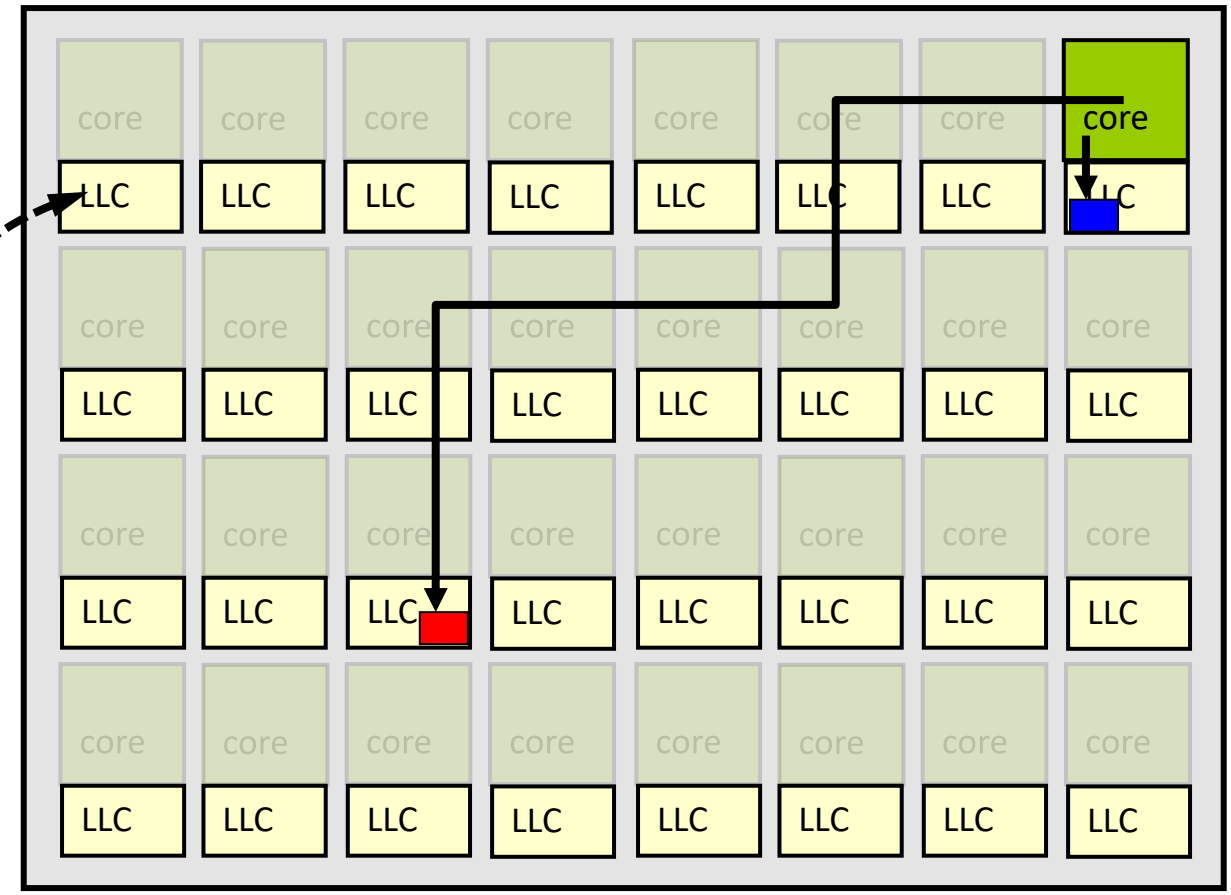
Picture modified from Hardavellas's talk

Non-Uniform Cache Architecture (NUCA)



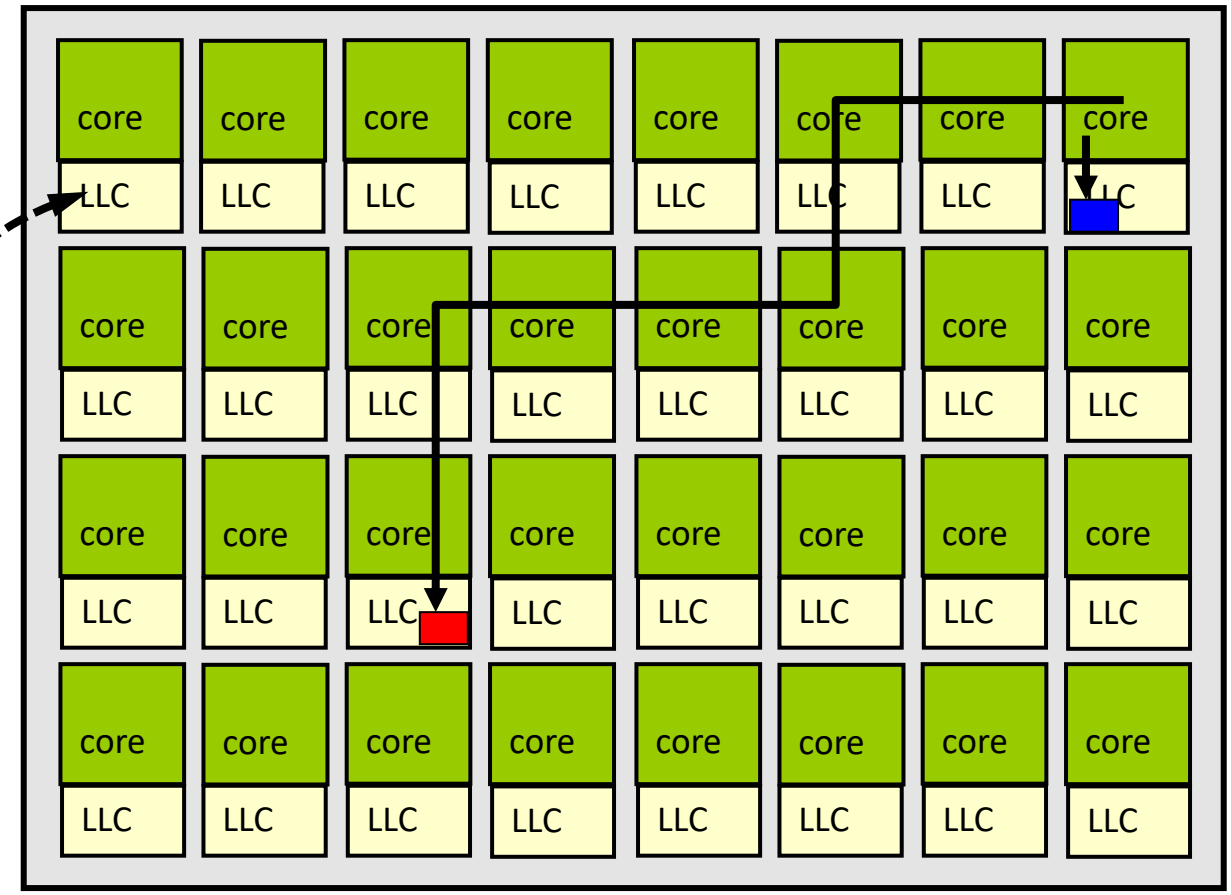
Picture modified from Hardavellas's talk

Non-Uniform Cache Architecture (NUCA)

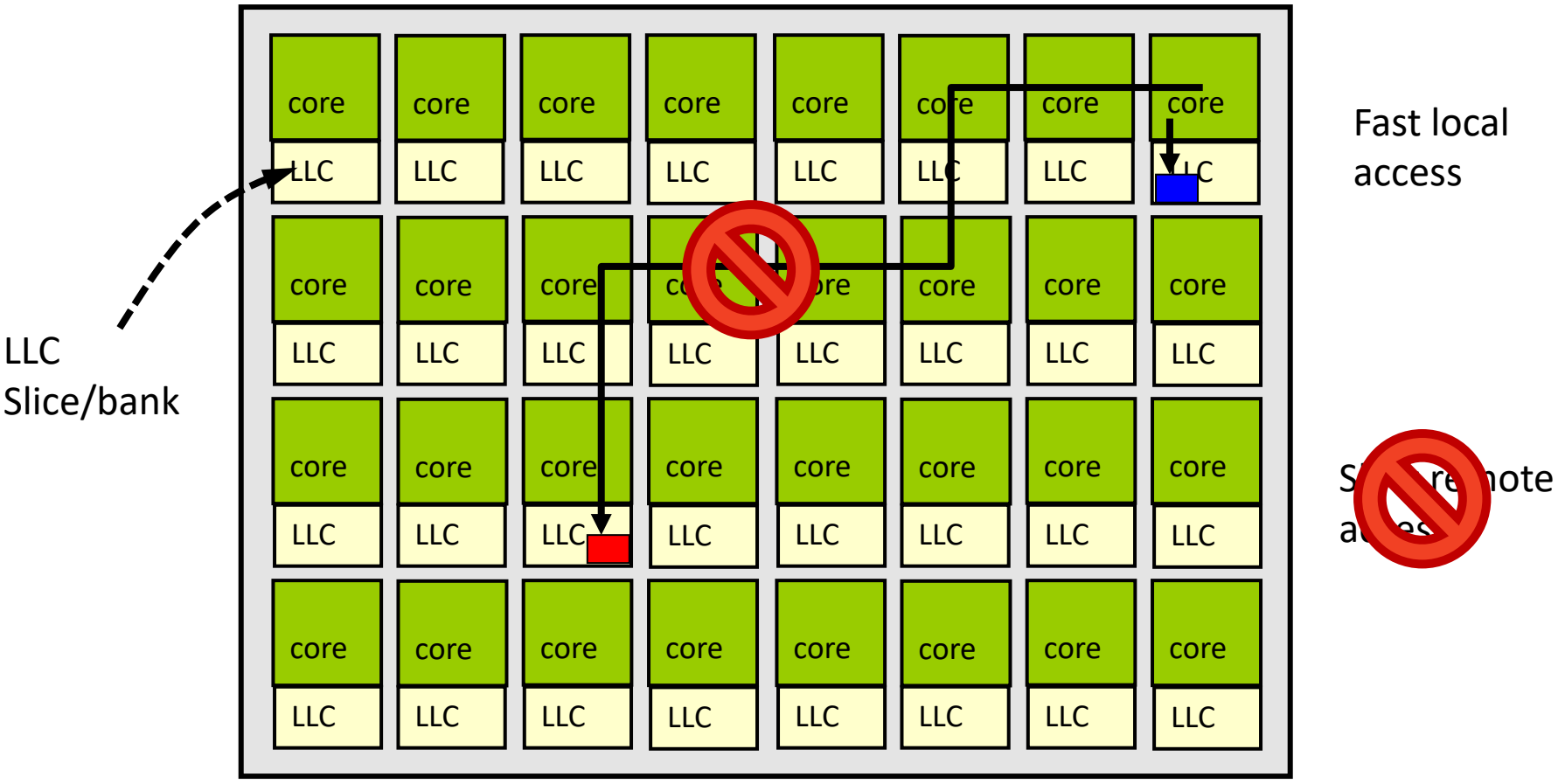


Picture modified from Hardavellas's talk

NUCA as Private LLC



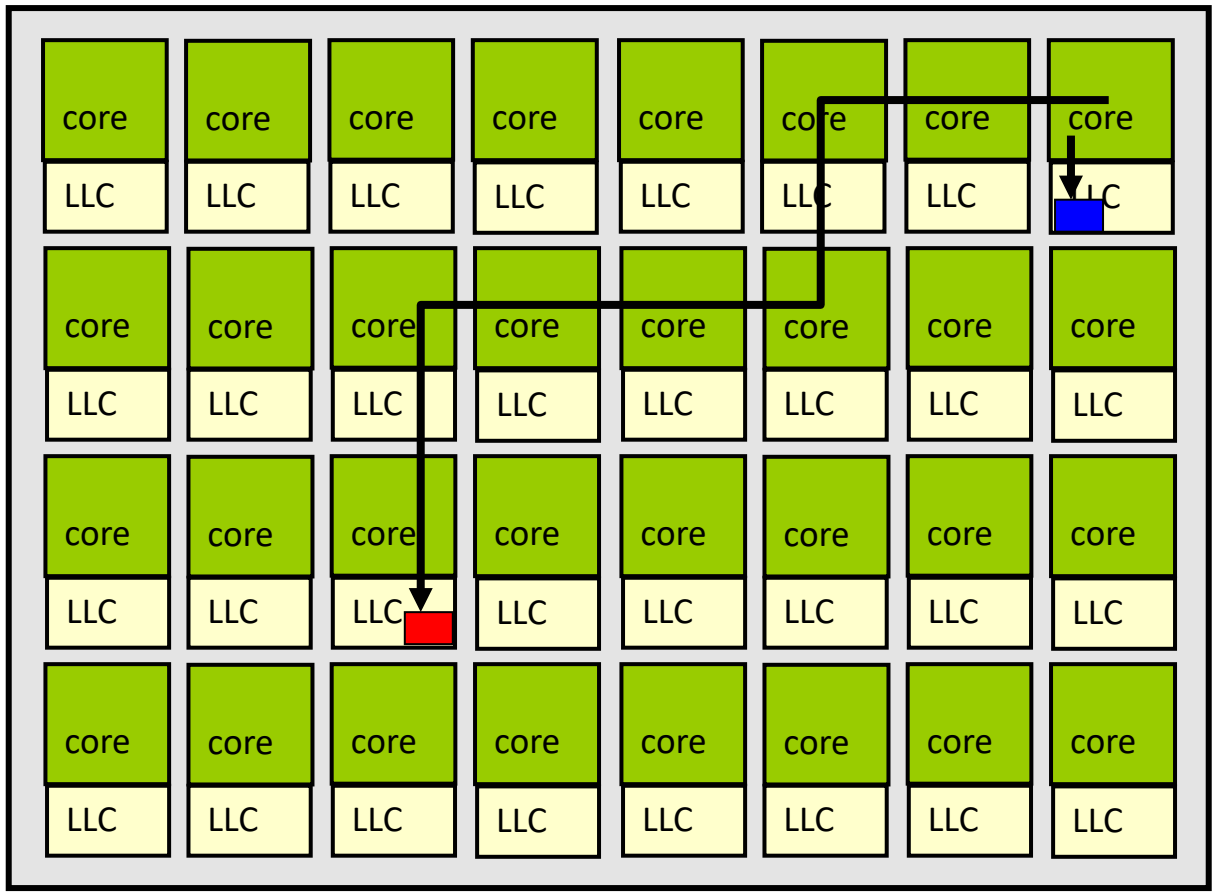
NUCA as Private LLC



- (+) Low access latency, less interconnect traffic
- (-) Duplication, possible under-utilization of capacity, coherence

NUCA as shared LLC (Static NUCA)

Physical address bit decides "home" slice/bank



- (+) No duplication, no coherence, full capacity utilization
- (-) High remote access latency, interconnect traffic

Can we get the best of both worlds?

- Low avg. hit latency of private caches
- Low LLC miss rate of shared caches

Cooperative Caching (Chang et. al. 2006)

- Start with private cache organization
- Aggregate cache capacity by cooperation

Cooperative Caching (Chang et. al. 2006)

- Start with private cache organization

- Aggregate cache capacity by cooperation
  - Before evicting a cache block to memory send it to another private slice → spilled into another cache slice
    - On a miss at local private cache slice, coherence protocol determines if there exists another copy

Cooperative Caching (Chang et. al. 2006)

- Start with private cache organization

- Aggregate cache capacity by cooperation
  - Before evicting a cache block to memory send it to another private slice → spilled into another cache slice
    - On a miss at local private cache slice, coherence protocol determines if there exists another copy
  - Randomly choose which cache slice to spill to with higher probably of choosing the neighboring slice
    - Spill probability to decide whether to cooperate or not

Cooperative Caching (Chang et. al. 2006)

- Controlling replication:
  - Singlet: Only copy on chip
  - Cache replacement policy avoids victimizing singlet, if possible
  - A cache block is given only once chance to spill

Cooperative Caching (Chang et. al. 2006)

- Controlling replication:
  - Singlet: Only copy on chip
  - Cache replacement policy avoids victimizing singlet, if possible
  - A cache block is given only once chance to spill

- Problem with Cooperative caching:

Cooperative Caching (Chang et. al. 2006)

- Controlling replication:
  - Singlet: Only copy on chip
  - Cache replacement policy avoids victimizing singlet, if possible
  - A cache block is given only once chance to spill

- Problem with Cooperative caching:
  - Coherence among the private LLC slices!

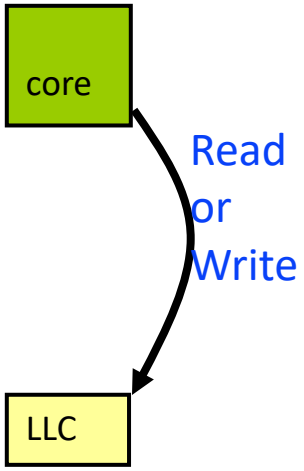
Reactive NUCA (ISCA'09)

- Observation: Not all cache blocks are equal

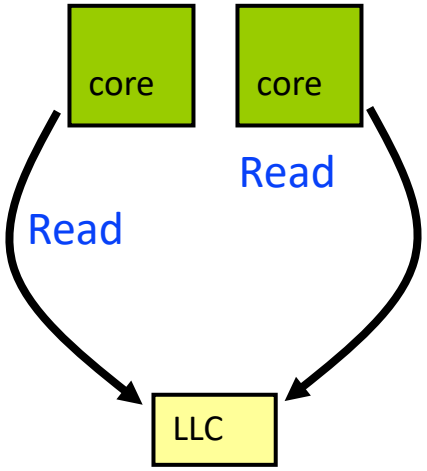
Picture modified from Hardavellas's talk

Reactive NUCA (ISCA'09)

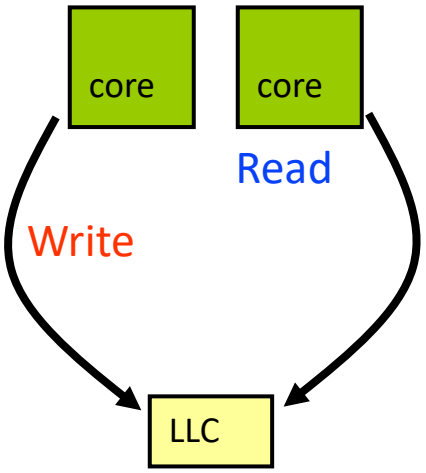
- Observation: Not all cache blocks are equal



Private
(Read-only or
Read-Write)



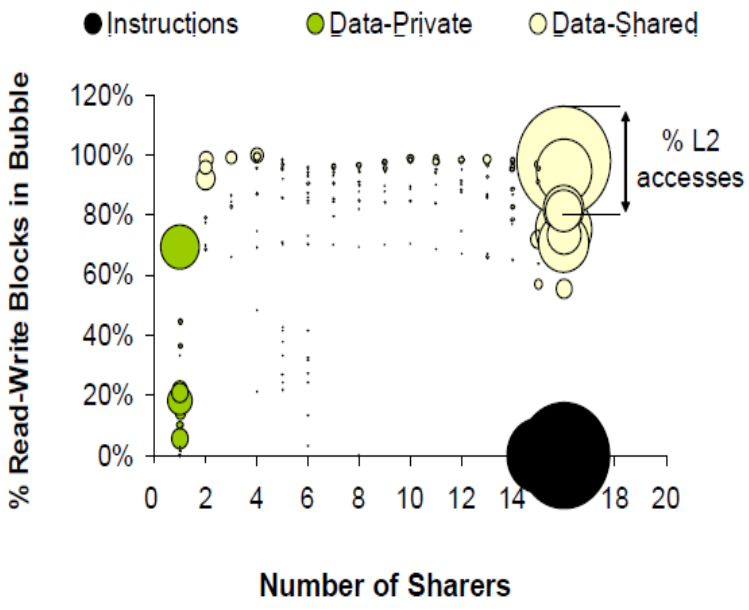
Shared
Read-Only
(Often instructions)



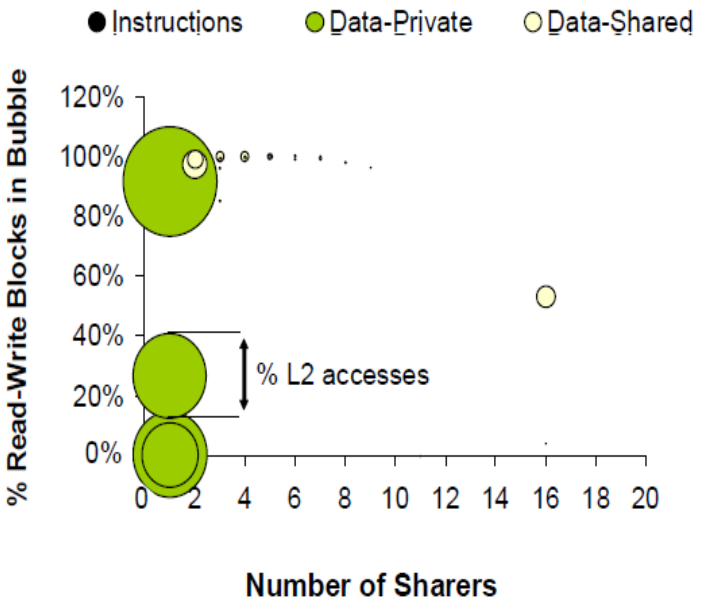
Shared
Read-Write

Picture modified from Hardavellas's talk

Characterization of LLC accesses



(a) Server Workloads

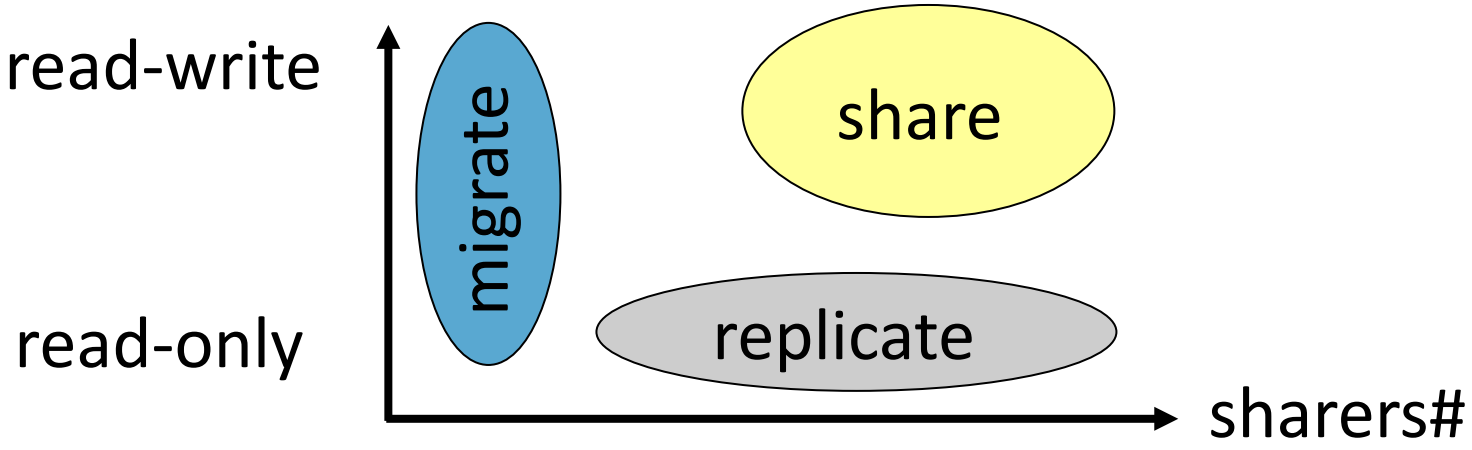


(b) Scientific and Multi-programmed Workloads

- Not all types of workloads are equal either !

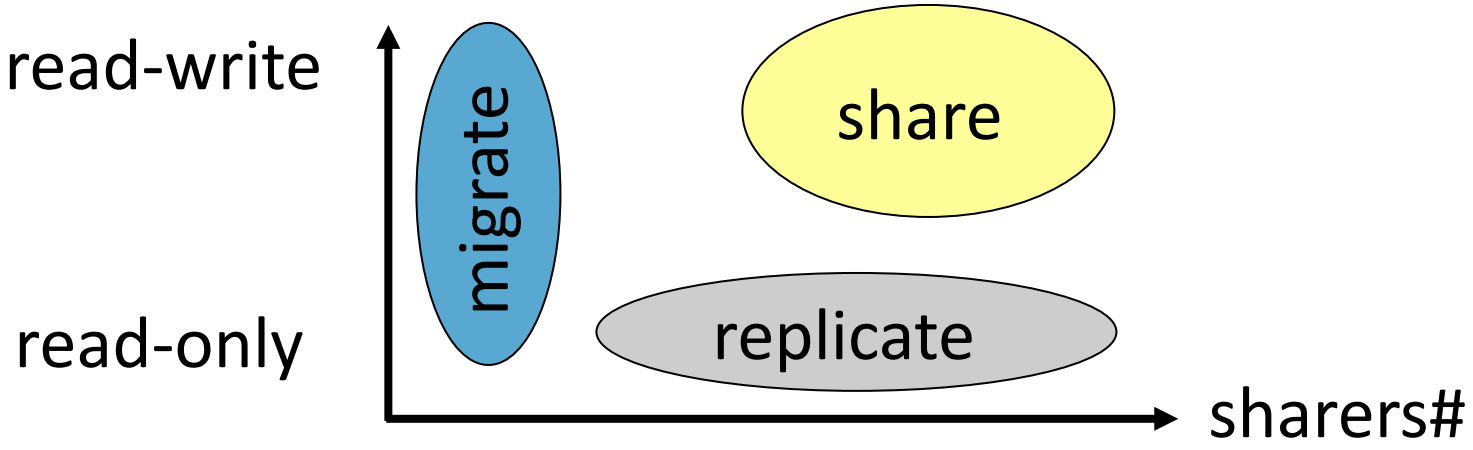
Picture modified from R-NUCA paper

Where to place what type of content?



Picture modified from R-NUCA paper

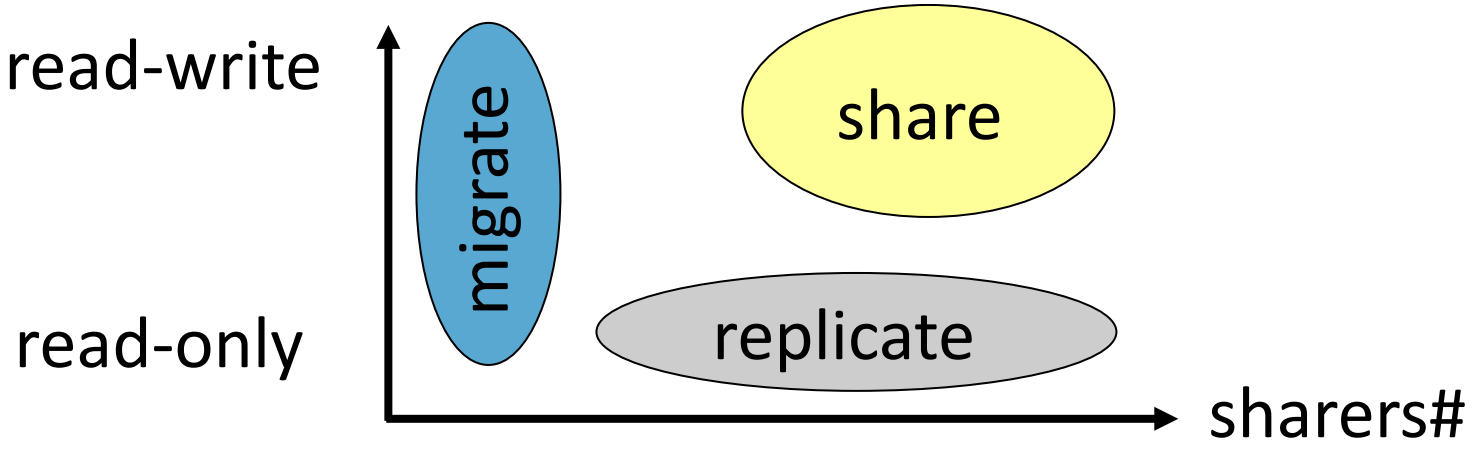
Where to place what type of content?



- Private and replicated read-only content no need to worry about coherence

Picture modified from R-NUCA paper

Where to place what type of content?



- Private and replicated read-only content no need to worry about coherence
- Shared read-write data can be placed in "home" bank/slice based on address (like S-NUCA) → no coherence

Picture modified from R-NUCA paper

How to classify cache content?

- How do you know a priori the behavior of a cache block?

How to classify cache content?

- How do you know a priori the behavior of a cache block?

- Done by the OS at page granularity:
  - PTEs are extended to keep the classification information
  - On a page walk, TLB entry are populated with classification information and core-ID that accessed it last time

How to classify cache content?

- How do you know a priori the behavior of a cache block?

- Done by the OS at page granularity:
  - PTEs are extended to keep the classification information
  - On a page walk, TLB entry are populated with classification information and core-ID that accessed it last time
- Before any cache access TLB lookup happens
  - Get the cache block classification from TLB

How to classify cache content?

- How do you know a priori the behavior of a cache block?

- Done by the OS at page granularity:
  - PTEs are extended to keep the classification information
  - On a page walk, TLB entry are populated with classification information and core-ID that accessed it last time
- Before any cache access TLB lookup happens
  - Get the cache block classification from TLB
- What happens on a miss-classification?
  - e.g., a page classified read-only is being written to

Putting it together: R-NUCA

- OS helps classify type of content at page granularity
- TLB entries keep this classification information
- A cache block is replicated (shared read-only) migrated (private read-only, read-write) or kept at "home" bank/slice (shared read-write), based on the classification