UNIVERSITY of ROCHESTER

Locality Wall: Hardware Limitations and Software Opportunities

Chen Ding

Professor
Rochester, NY

Oct. 14, 2017

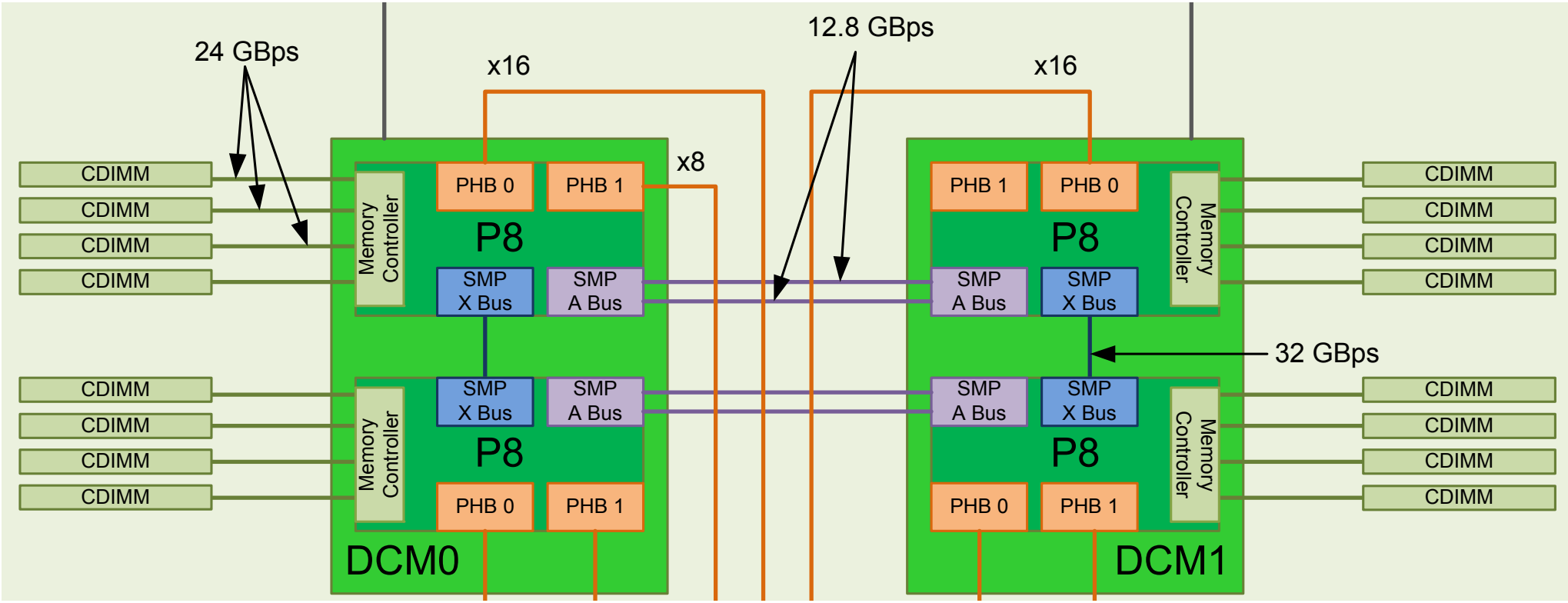
Dawn of A New Era?

- Moore's Law
  - 20,000 times smaller/faster
  - 10μm to 5nm
  - running against the laws of physics and statistics
- Memory Wall
  - Wulf and McKee, 1995
  - solution
    - bigger SRAM cache, prefetching
    - LINPACK has tremendous temporal locality [DK IPDPS'00]
- Power Wall
  - 2004, cooling unrealistic at 120W
  - solution
    - lower clock rate. multicore. SIMD. GPU

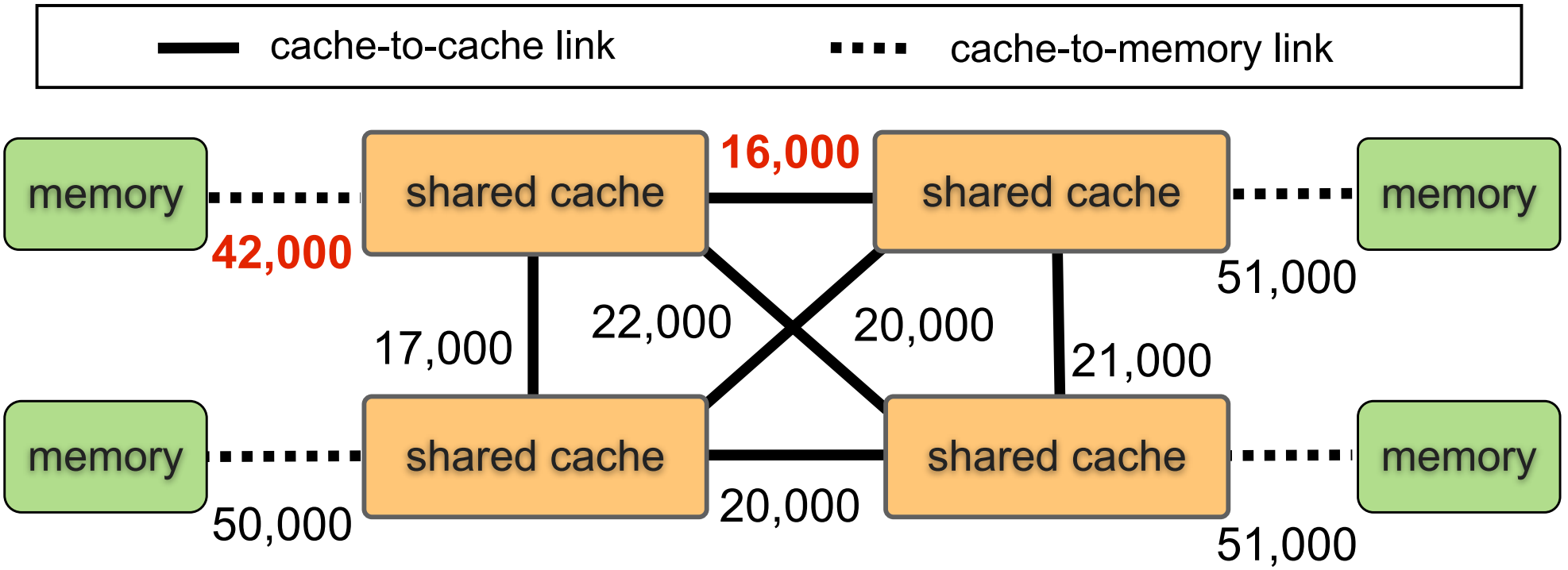
Present Impasse

- Modern applications
  - large data sets
  - no predictable regularity
- Data movement problems take center stage
  - dominant factors in speed and power
- Adding bandwidth
  - observation by David Wang
    - 5GB/s per core by Intel in the last two decades
    - adding memory channels from 1 to 6

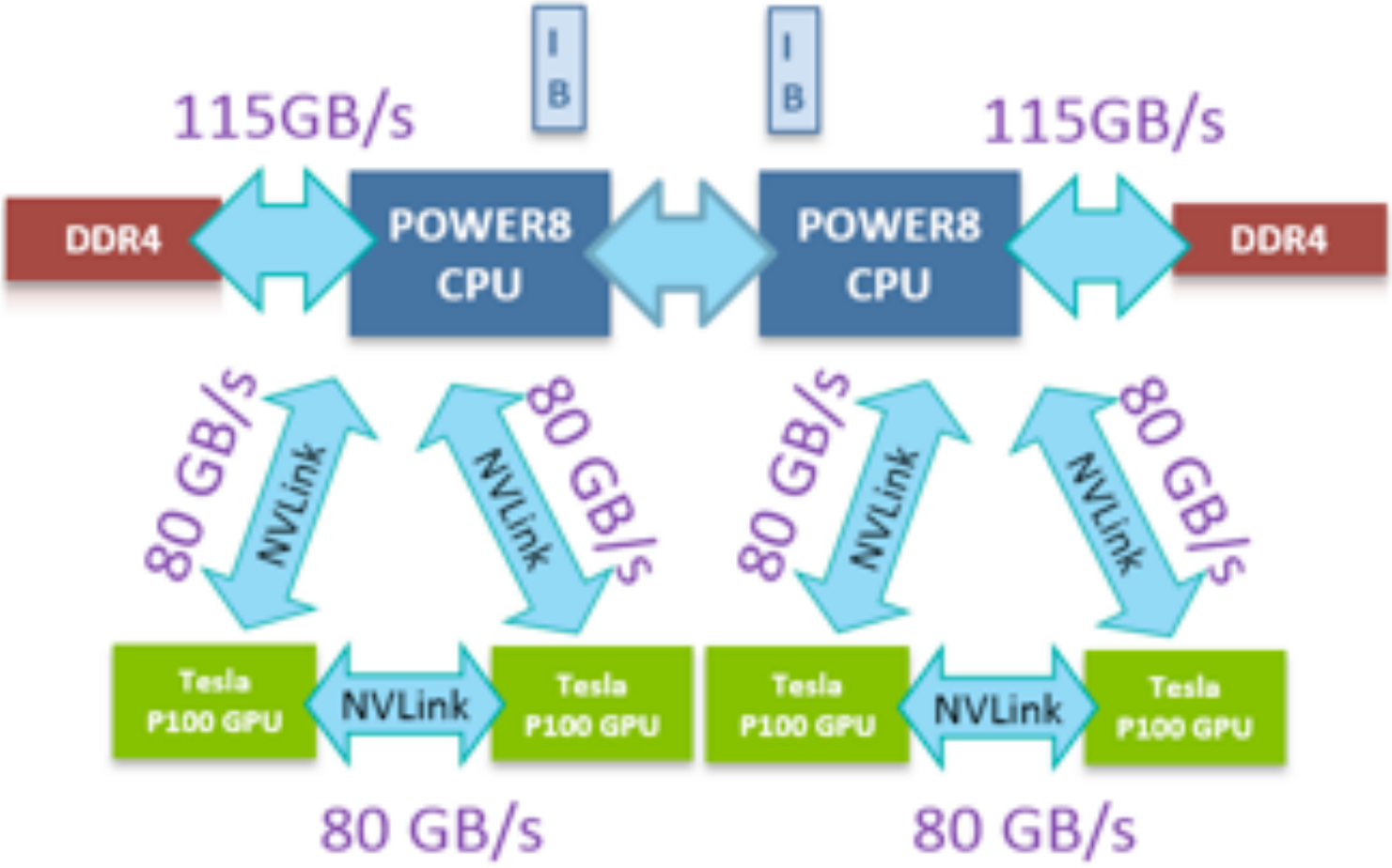
IBM Power8



Chen Ding, University of Rochester, 2016

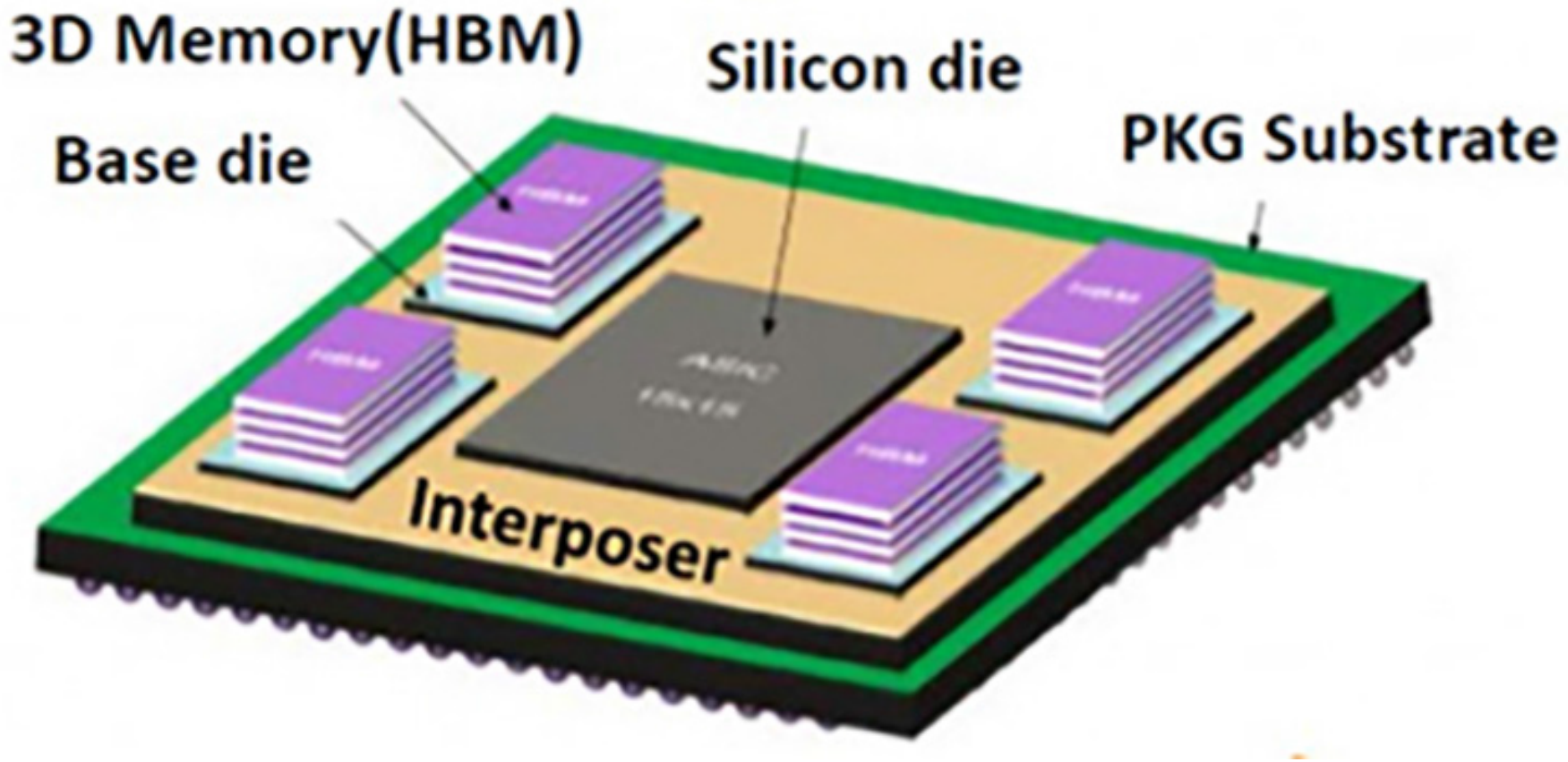

cache-to-cache link
cache-to-memory link
memory
shared cache
16,000
shared cache
memory
42,000
51,000
17,000
22,000
20,000
21,000
memory
shared cache
shared cache
memory
50,000
20,000
51,000

IBM Minsky



https://www.hpcwire.com/2016/09/08/ibm-debuts-power8-chip-nvlink-3-new-systems/

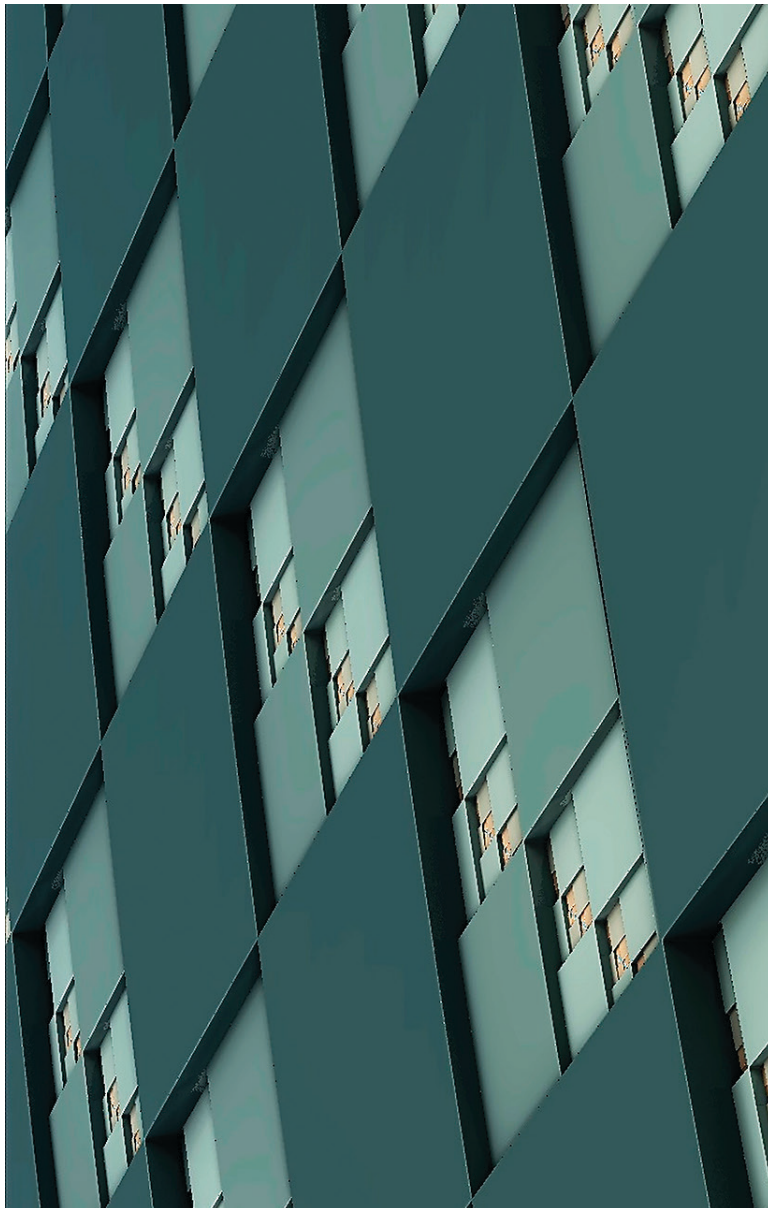
High Bandwidth Memory (HBM)



Implications of the datacenter's shifting center.

BY MIHIR NANAVATI, MALTE SCHWARZKOPF, JAKE WIRES, AND ANDREW WARFIELD

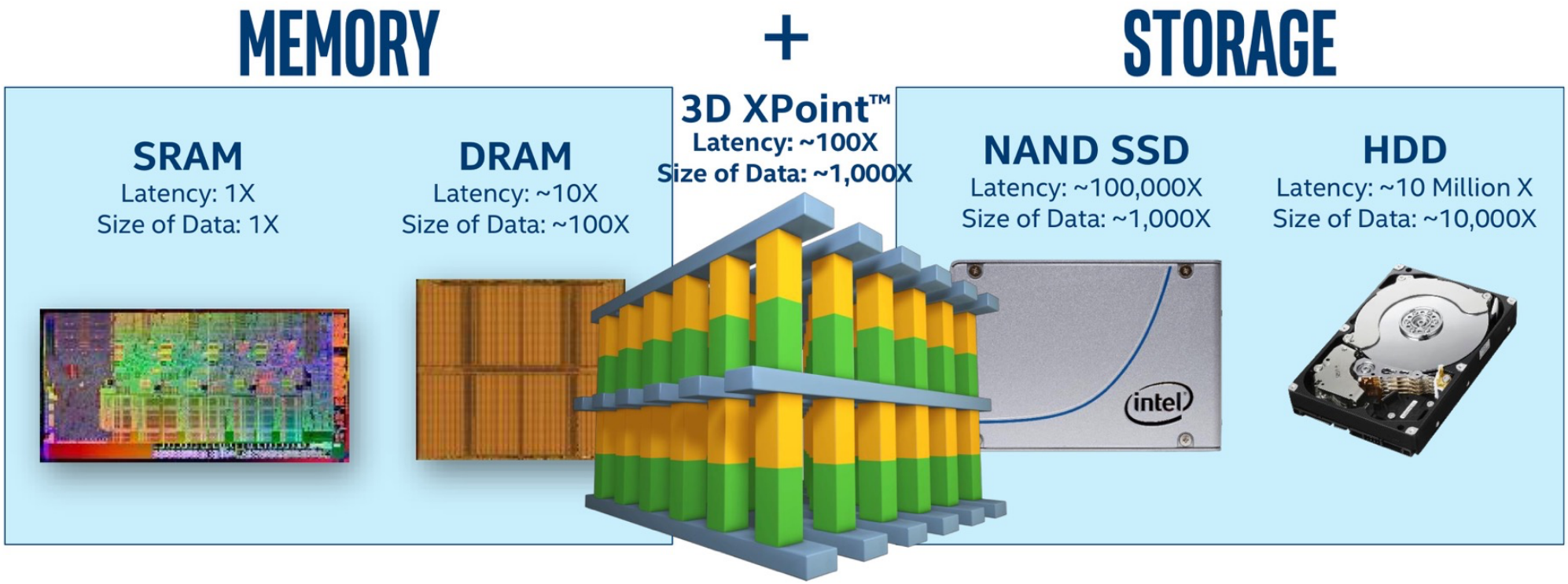
Non-Volatile Storage



"The arrival of high-speed, non-volatile storage … is likely the most significant architectural change that datacenter and software designers will face in the foreseeable future."

3D XPOINT™ MEMORY MEDIA

Breaks the memory/storage barrier



Technology claims are based on comparisons of latency, density and write cycling metrics amongst memory technologies recorded on published specifications of in-market memory products against internal Intel specifications.

https://cdn.arstechnica.net/wp-content/uploads/2017/03/IntelR-OptaneTM-Technology-Workshop-Analyst-and-Press-Slides-3-15...-4.jpeg

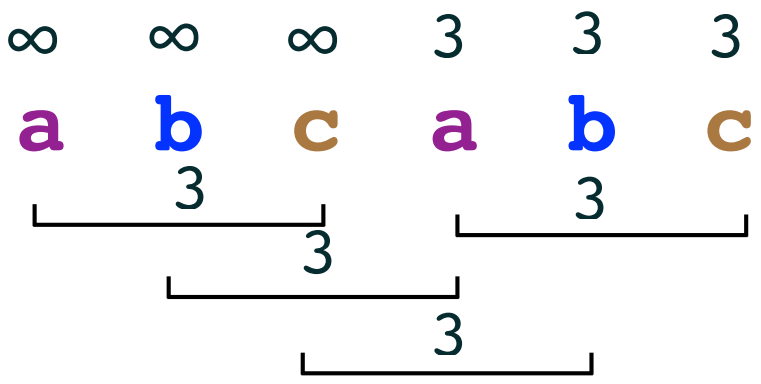
"Locality Wall" — Peter Kogge

- Memory complexity
  - memory domains
  - ports
  - types
  - links/protocols
  - growing block sizes
- Non-locality
  - consumes energy
  - exa-scale, 20 pJ per flop, unrealizable if any access misses in cache
- Sparse problems
  - HPCG, SpMV, BFS/Graph500

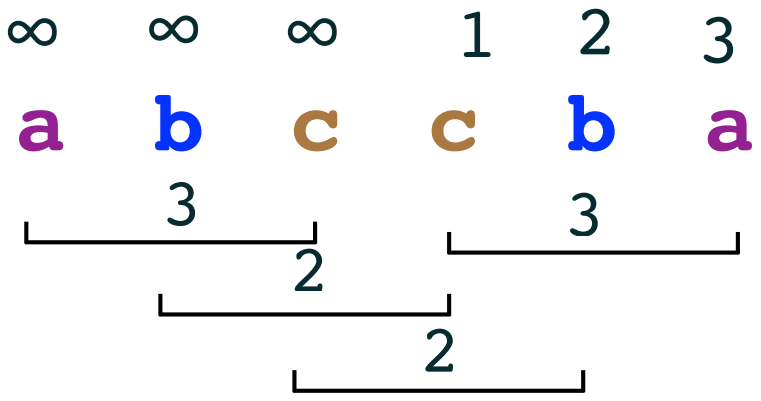
Memory Problems in Software

- Software problems
  - algorithms, data structure / layout, parallelization, scheduling, task/data placement, cache/memory sharing …
- Solution strategies
  - abstractions: let programmers focus on what matters
  - modularity: divide and conquer
- Locality
  - closest access
  - shortest data movement
- What does locality mean for software?
  - assume dynamic allocation/sharing of local memory
  - non-answer: temporal/spatial locality

- Access locality
  - reuse distance of an access: amount of data since previous access
  - shorter reuse distance -> better locality
- Timescale locality
  - footprint of length x: average WSS in all windows of length x
  - smaller footprint -> better locality



fp(3) = 3



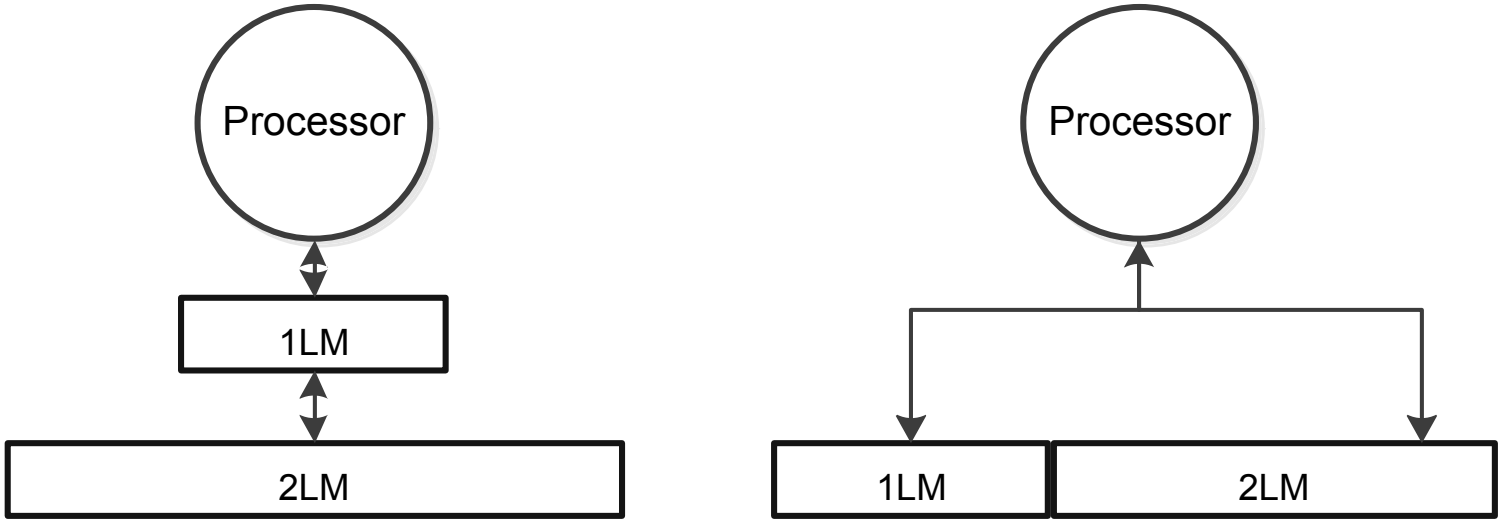
fp(3) = 10/4 = 2.5

Memory Equalizer

Ye et al. MEMSYS 2017

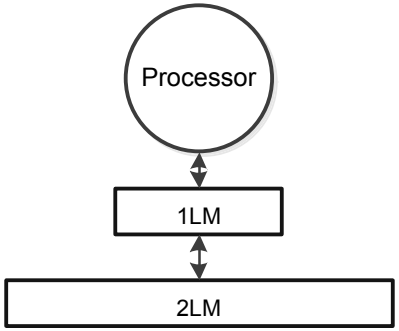
Hybrid Memory Architecture

- HpMC in fist MEMSYS (2015)
  - Su, Roberts, Leon, Cameron, de Supinski, Loh, Nikolopoulos
  - a convincing case study
  - HpMC chooses between PCache and HRank

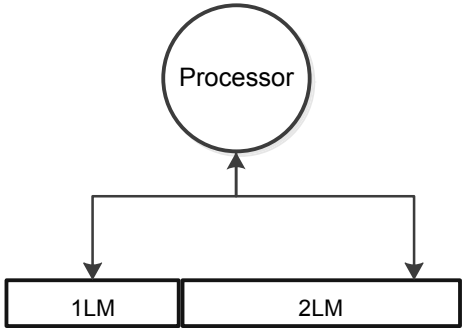


(a) *PCache* (b)*HRank*

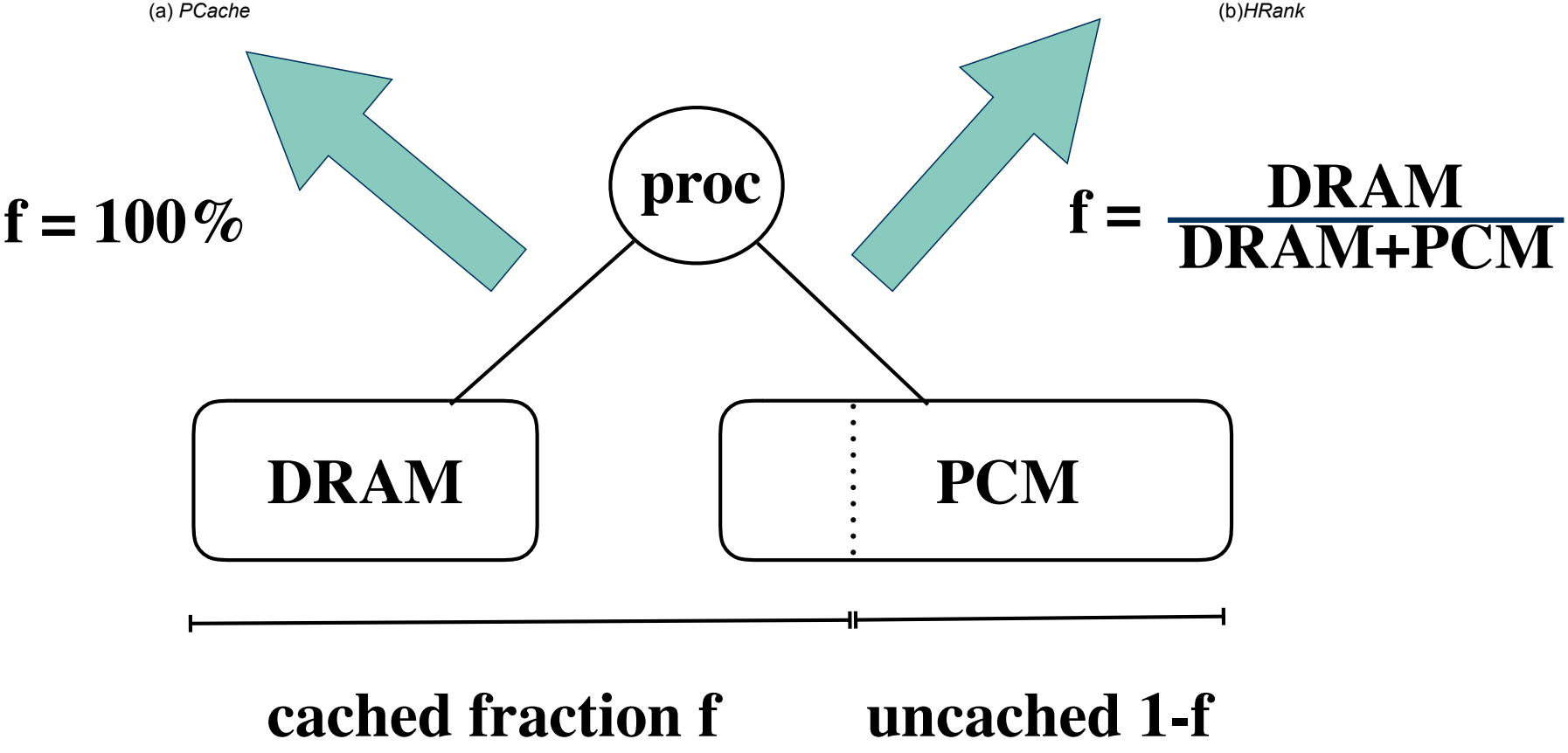
Fraction Cache



(a) *PCache*



(b)*HRank*

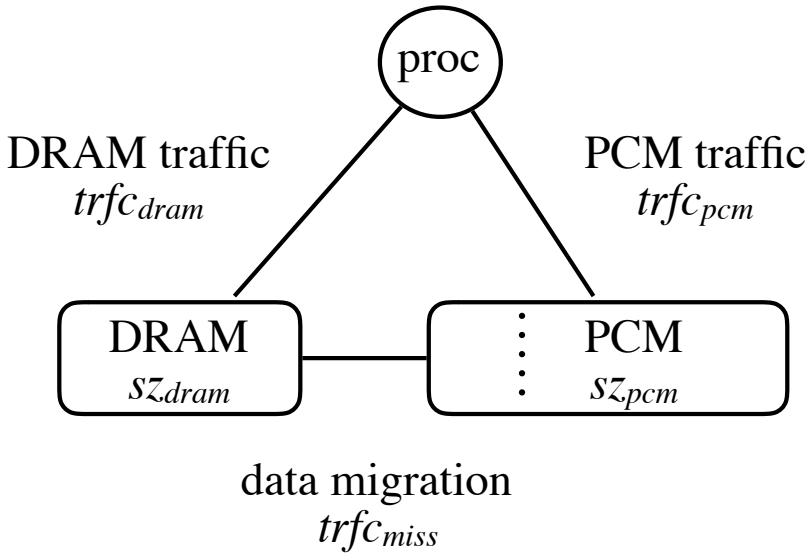


Higher Order Theory of Locality (HOTL)

- Locality is represented by integer functions
  - footprint fp(x)
    - average working-set size for all windows of length x ($x \geqslant 0$)
  - miss ratio mr(c)
    - for fully-associative LRU cache of size c
- Cache modeling becomes function operations
  - mr(c) is the derivative of fp(x), or in Leibniz's notation

$$mr(c) = \frac{\mathrm{d}}{\mathrm{d}x} fp(x) \Big|_{fp(x)=c}$$

Traffic Modeling



$$h(x) = rfp(x)$$

$$mr(sz_{dram}) = \frac{\mathrm{d}}{\mathrm{d}x} h(x) \Big|_{h(x)=sz_{dram}}$$

$$trfc_{miss} = mr(sz_{dram}) * n$$

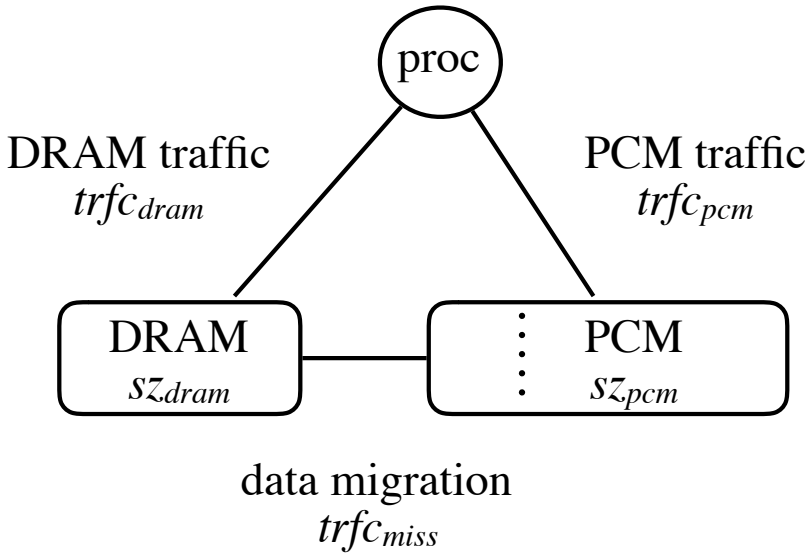
$$trfc_{dram} = r * n - trfc_{miss}$$

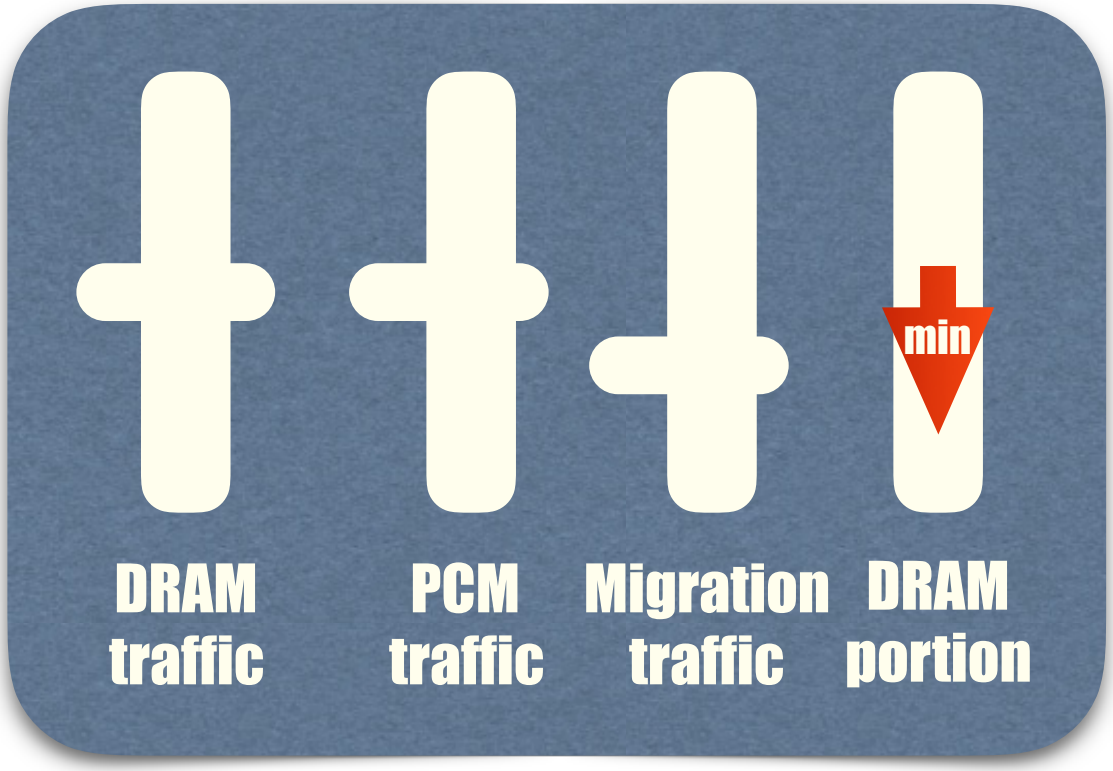
$$trfc_{pcm} = (1 - r) * n + trfc_{miss}$$

$$= n - trfc_{dram}$$

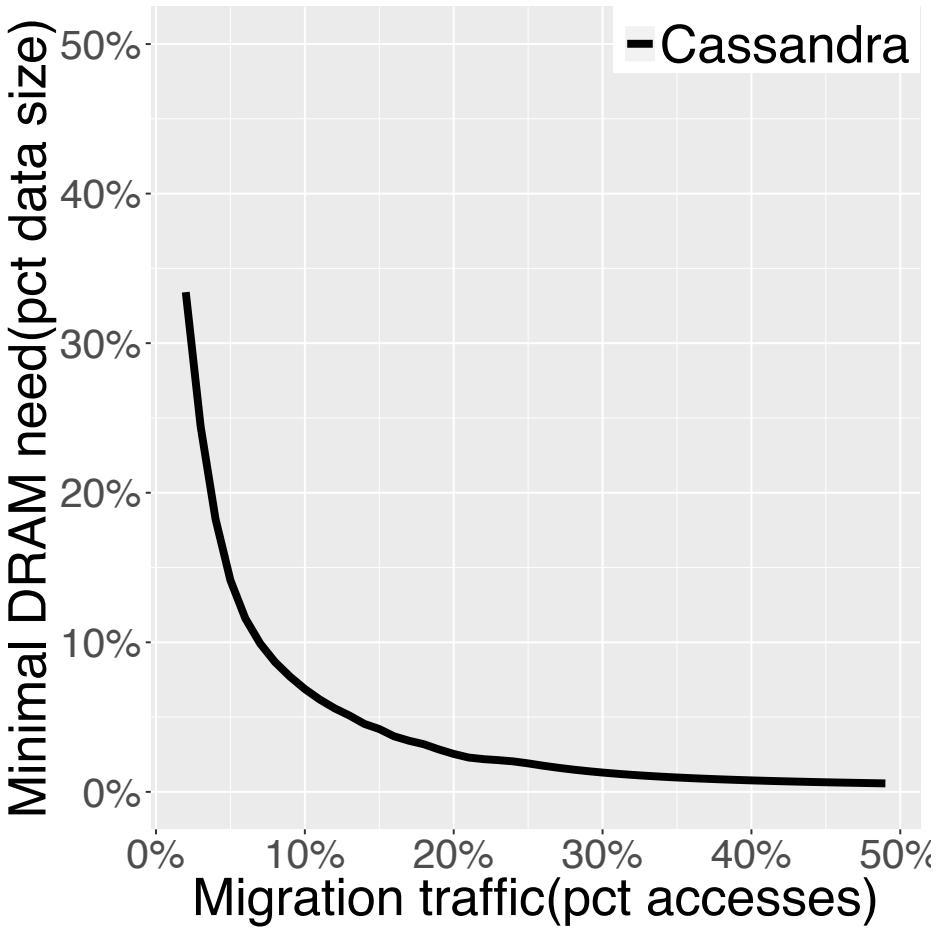
$$r = \frac{trfc_{dram} + trfc_{miss}}{n}$$

Memory Portion Reduction

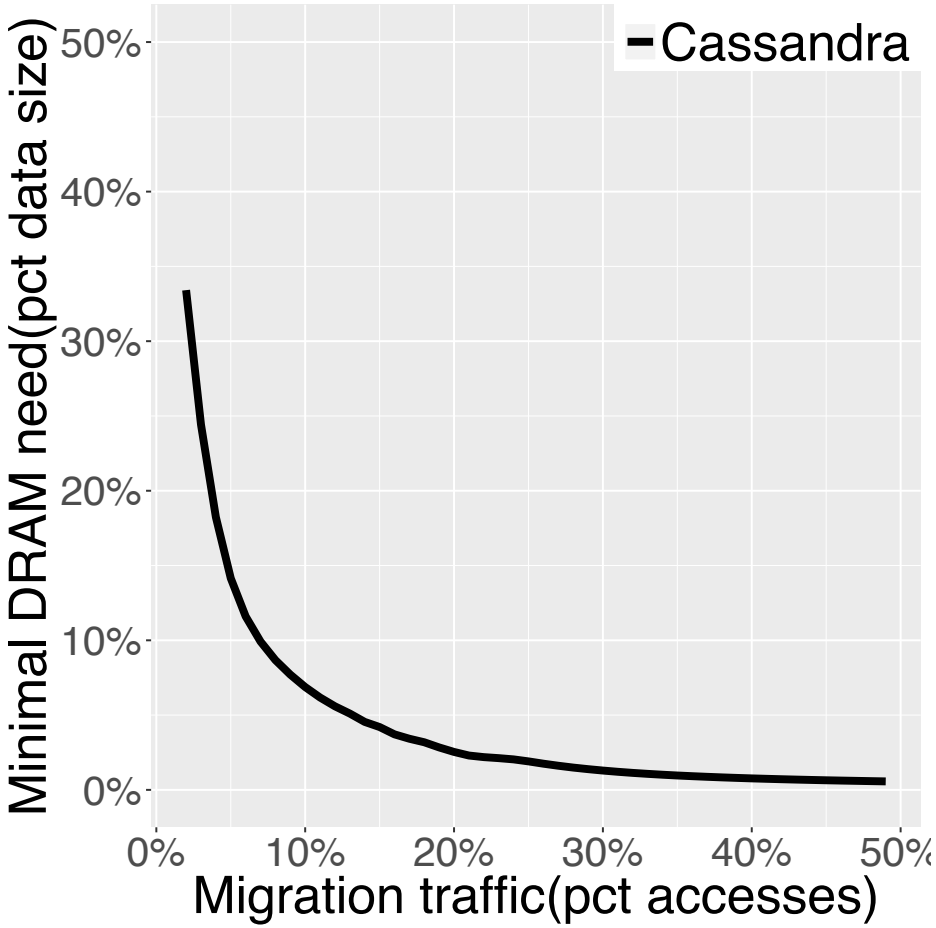


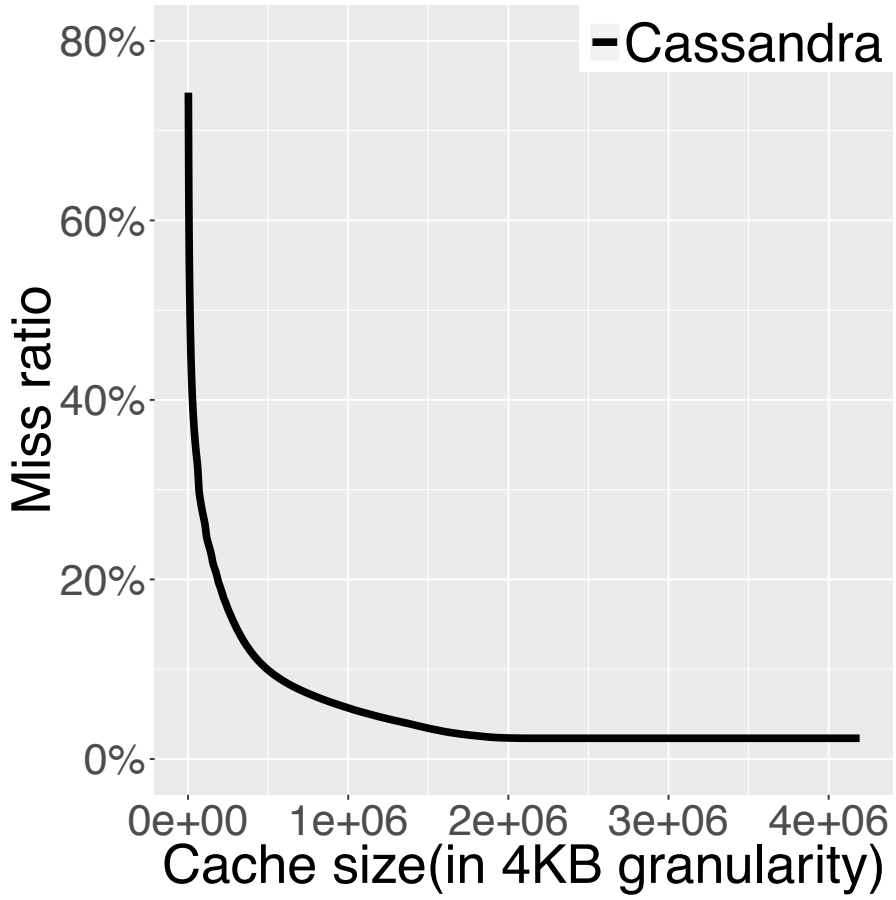


Evaluation

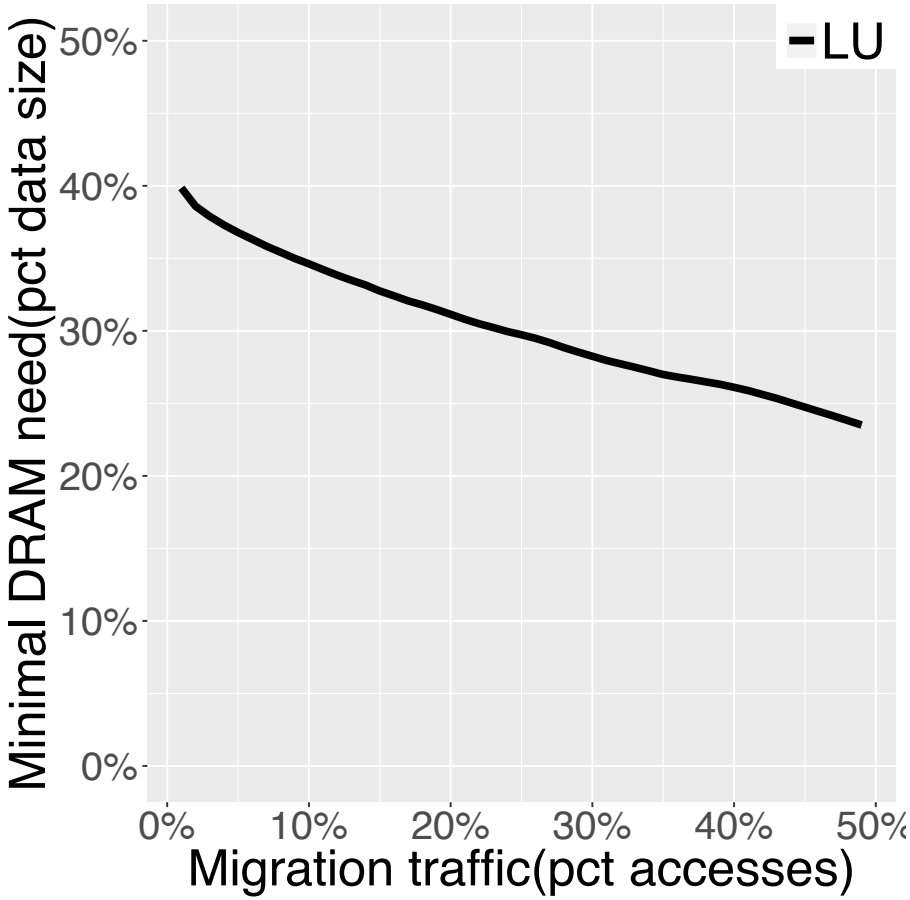


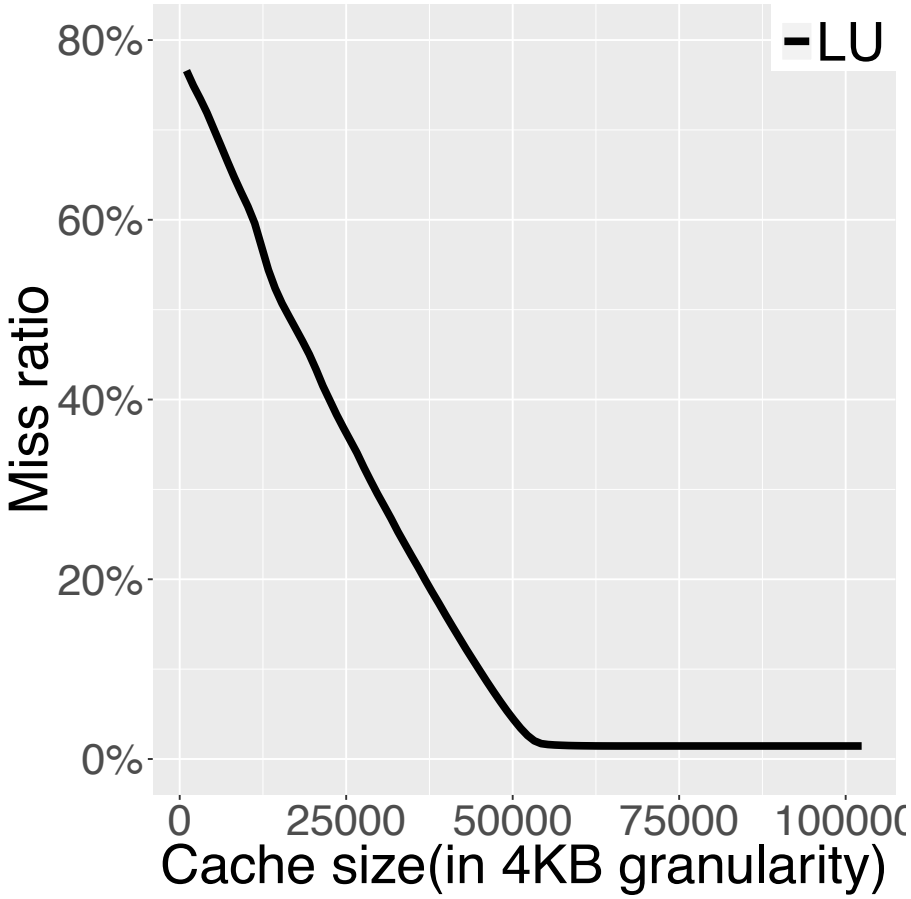
- Apache Cassandra
  - open-source NoSQL db
  - 17 GB data
  - 7% DRAM portion at 10% migration traffic
    - 60% fraction cached
  - 0.6% DRAM portion at near 50%
    - 100% fraction cached
- Other tests
  - 3 PARSEC, 3 NPB
  - at least 500MB data
- All run with 8 threads



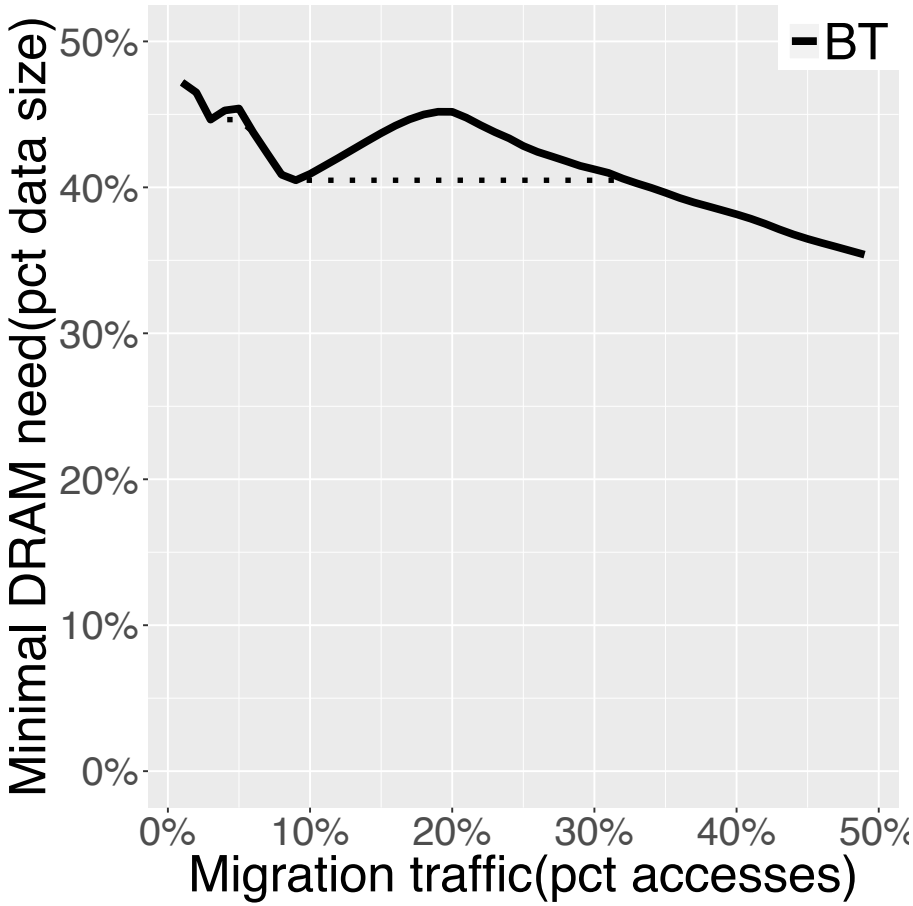


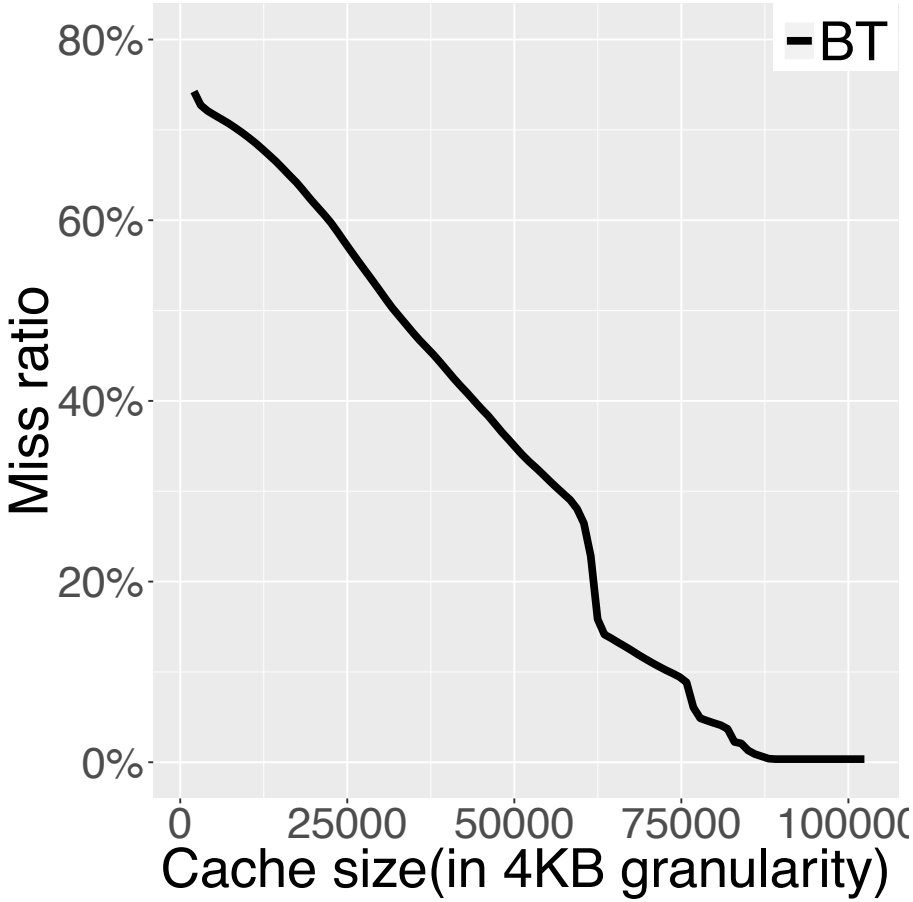
- DRAM reduction vs. miss ratio
  - good locality leads to effective reduction





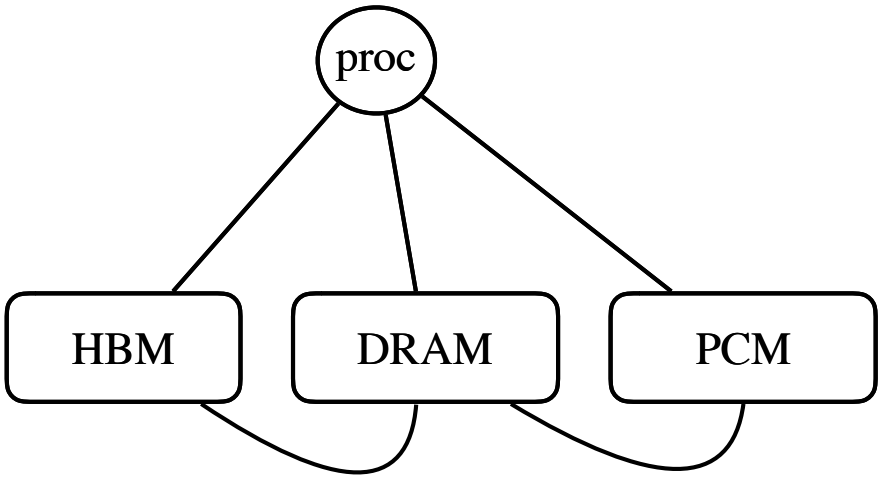
- Poor locality means ineffective reduction

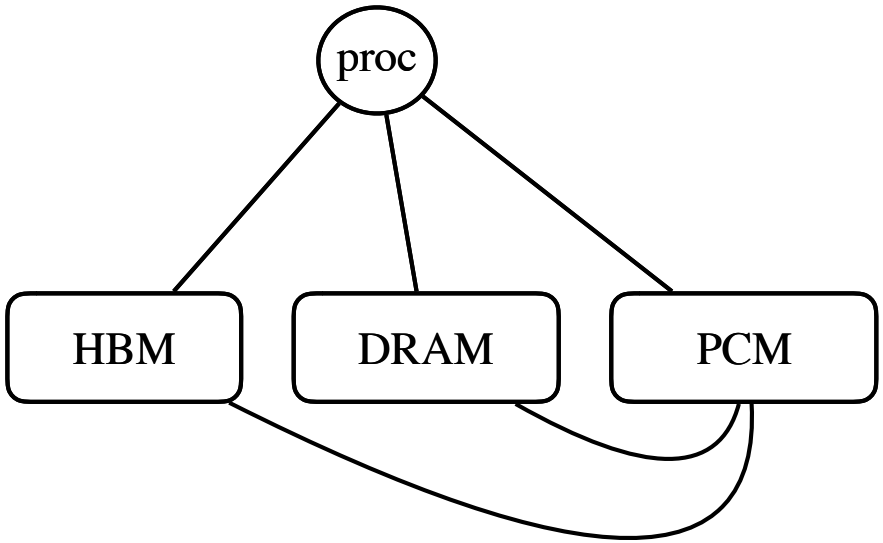




- Non-convexity loses monotonicity

More Fractions





- Multiple solutions
- Multiple objectives
  - size, traffic, endurance

- Useful models
  - write locality [MEMSYS'16]
  - cache exclusivity [TACO'17]

Cache and Memory Optimization

- Higher order theory of locality (HOTL)
  - locality is represented by integer functions
  - cache modeling becomes mathematical, w/ provable properties
    - concavity [ASPLOS'13]
    - composition invariance [USENIX'16]
    - cache exclusivity: correctness/uniqueness [TACO'17]
  - so does cache optimization
    - LAMA: memory allocation in Memcached [USENIX'16]
      - monotonicity implied by optimality
- Higher order theory of memory demand (HOTM)
  - liveness metrics [ISMM'14]
  - concurrent memory allocation [ISMM'16]
    - symmetry implied by optimality

Concurrent Collections

- CnC
  - separation of what and how
    - domain and tuning specification
    - collections and steps
  - communication centric
    - explicit dependences
    - graph programming model
  - compiled
    - static binding between collections and tuners

Freedom to Optimize

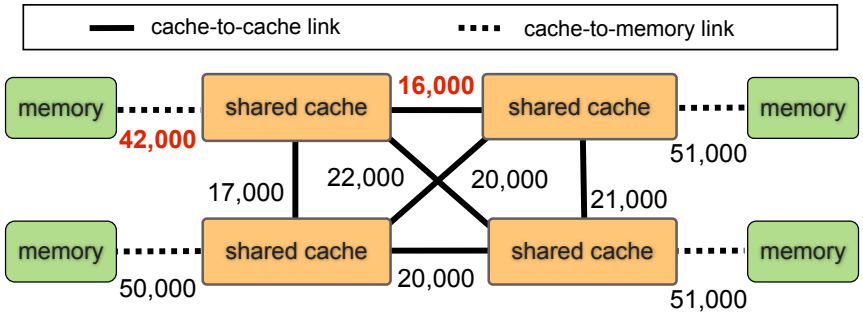
- Existing parallel languages
  - fixed parallelization and data layout
    - OpenMP, CUDA, MPI+X
    - Cilk family
  - fixed data layout
    - Jade and recent DSLs
- CnC
  - "future proof"
  - only the essential elements
  - e.g. Nick's hand-on tutorial
    - stencil can be 1D/2D arrays or a hash table

Locality Optimization in CnC

- Static locality analysis
  - footprint and miss ratio curves
    - per step per item
- Selecting/composing CnC Tuners
  - locality ranking of alternative implementations of each step
    - with parameters
  - combined effect in all steps
- Optimization
  - formalization of the implementation space
  - search for the best solution
    - lower bound work by Luis et al.
    - performance synthesis

AI? What about AM?

- Locality theory + deep learning
  - LT to produce traffic numbers
  - DL to map traffic numbers to performance
- Memory and intelligence
  - meta-cognition is how confident you are in what you think you know
  - it means fluency
  - fluency means efficiency
    - speed of computation
    - speed of data access



Summary

- Locality Wall
  - parallel, distributed and heterogeneous processing and memory
  - data movement is most critical
- Locality
  - access and timescale locality
  - mathematical relations/properties
- Memory equalizer
  - utilization, traffic, power, endurance
  - constrained optimization
- CnC
  - freedom to optimize
  - locality tuners