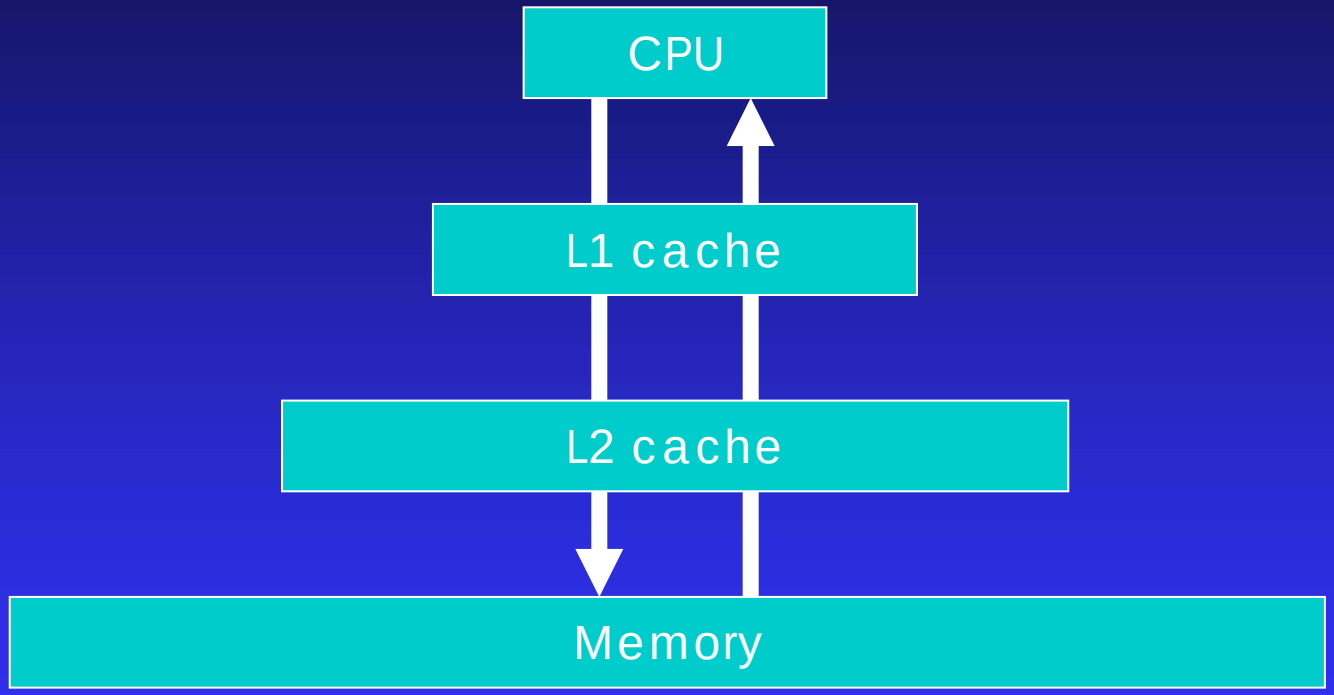
Matrix Multiply Implementation

Kazushige Goto
<kgoto@tacc.utexas.edu>

Contents

- Cache organization
- Packing data
- How MM (DGEMM) kernel works

Cache Organization

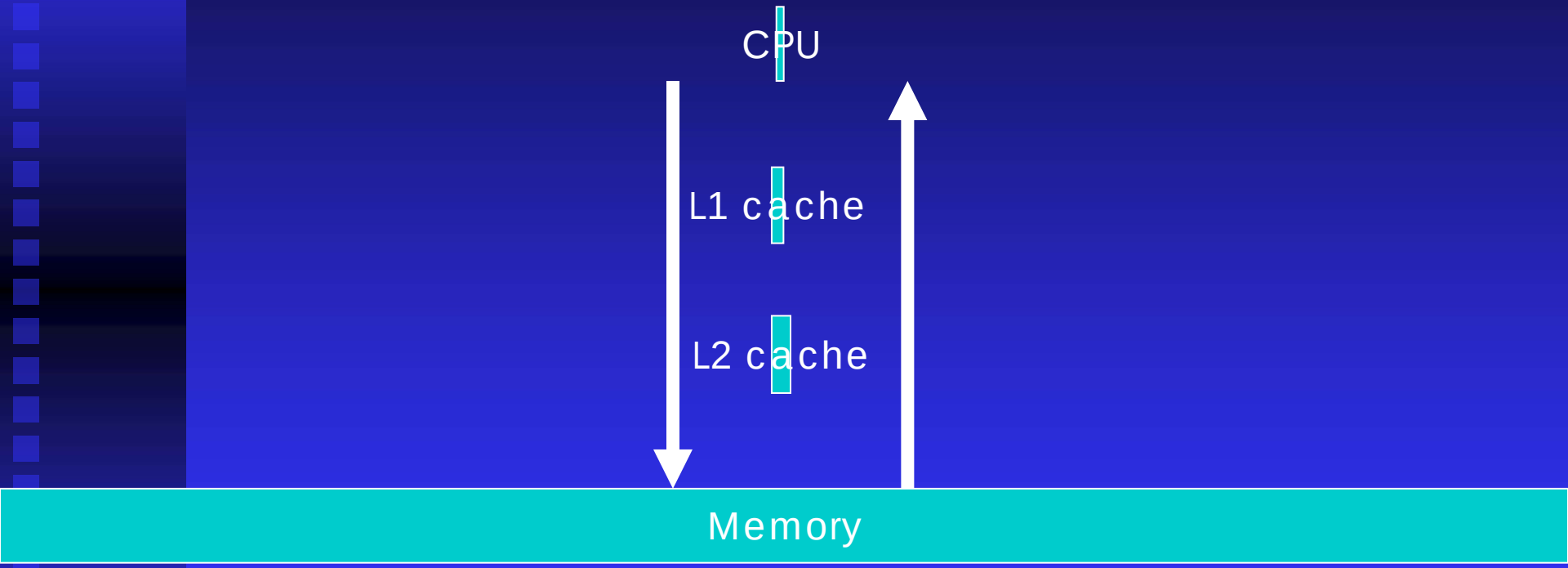


Generally cache improves performance

Three keywords of cache (L1)

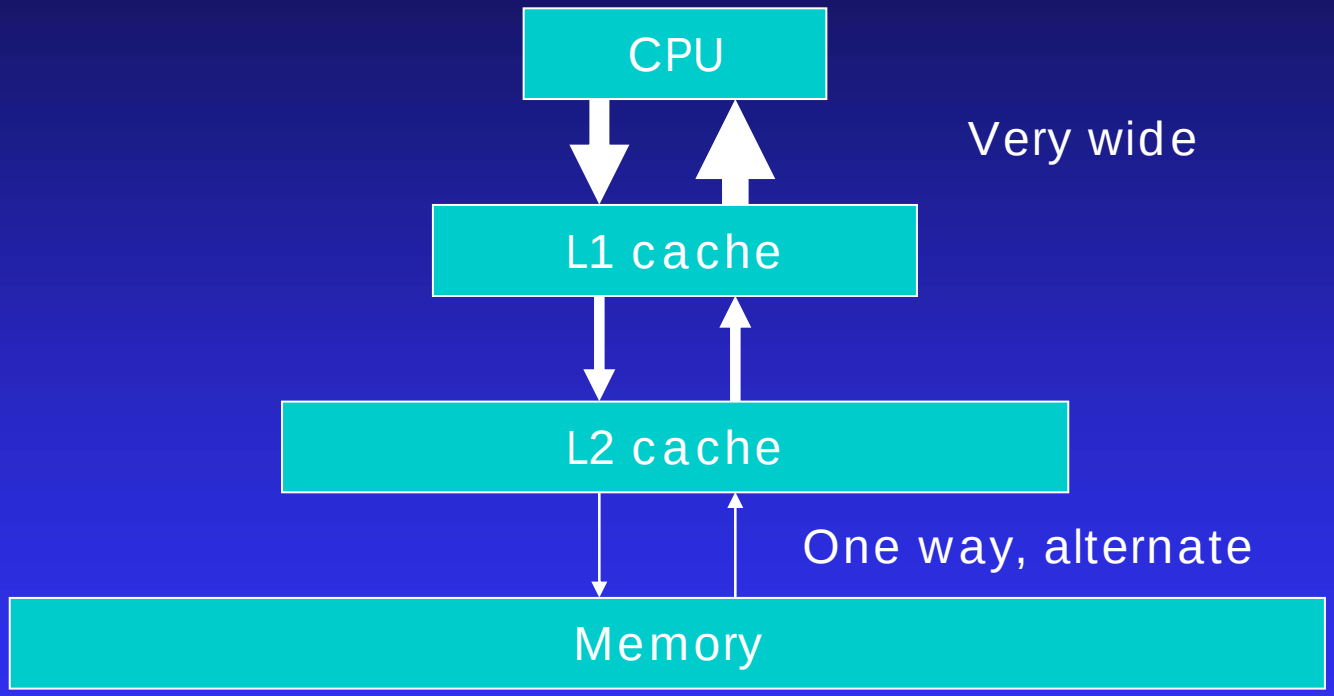
- Size
  - Pretty small (8kB – 64kB)
- Bandwidth
  - How much it can move data per cycle
  - Very wide
- Latency
  - Response time to get data
  - Relatively low

Size



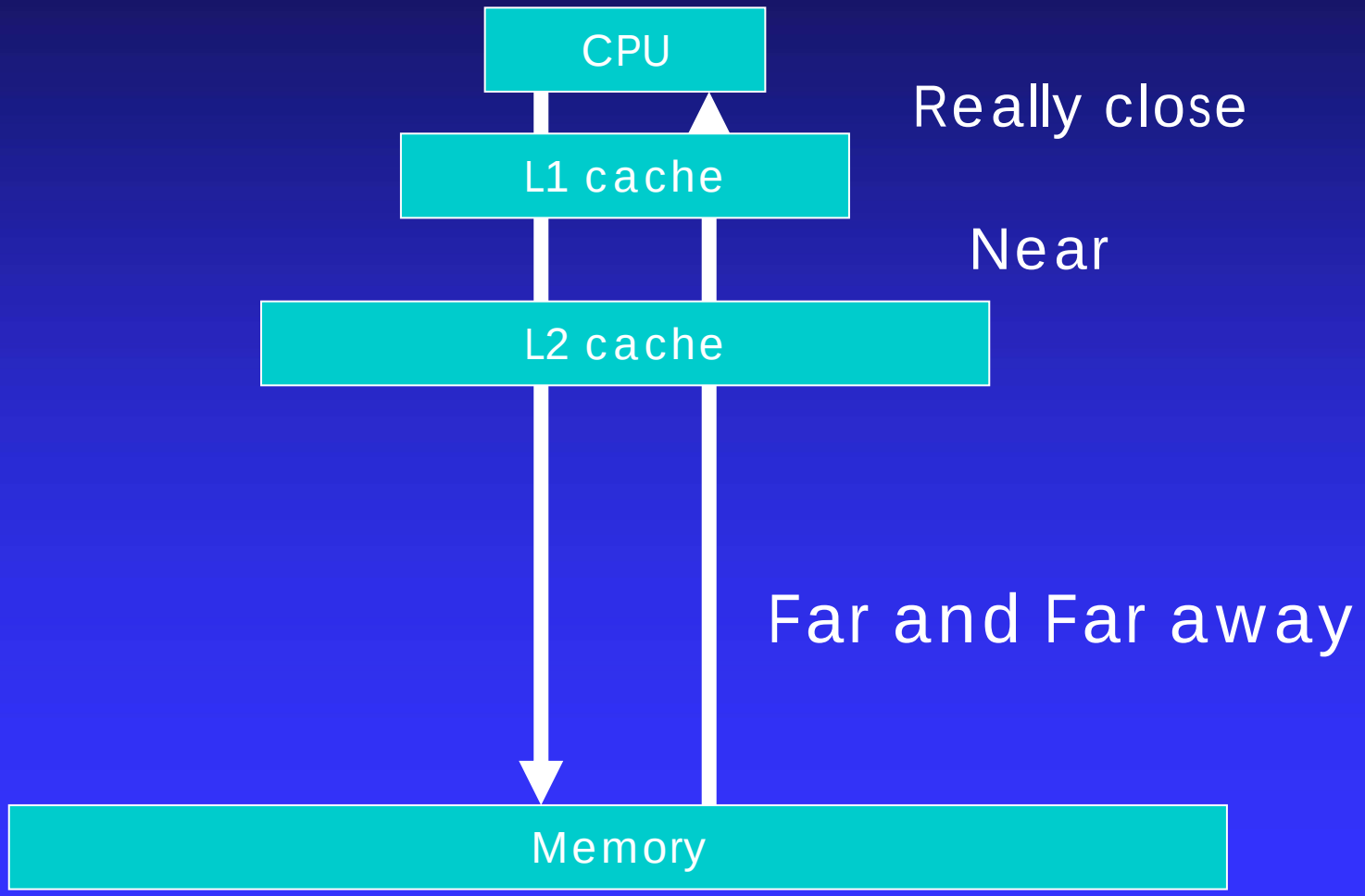
Cache size is very small!

Bandwidth



L1 Bandwidth is much wider than memory bandwidth (over 20 times?)

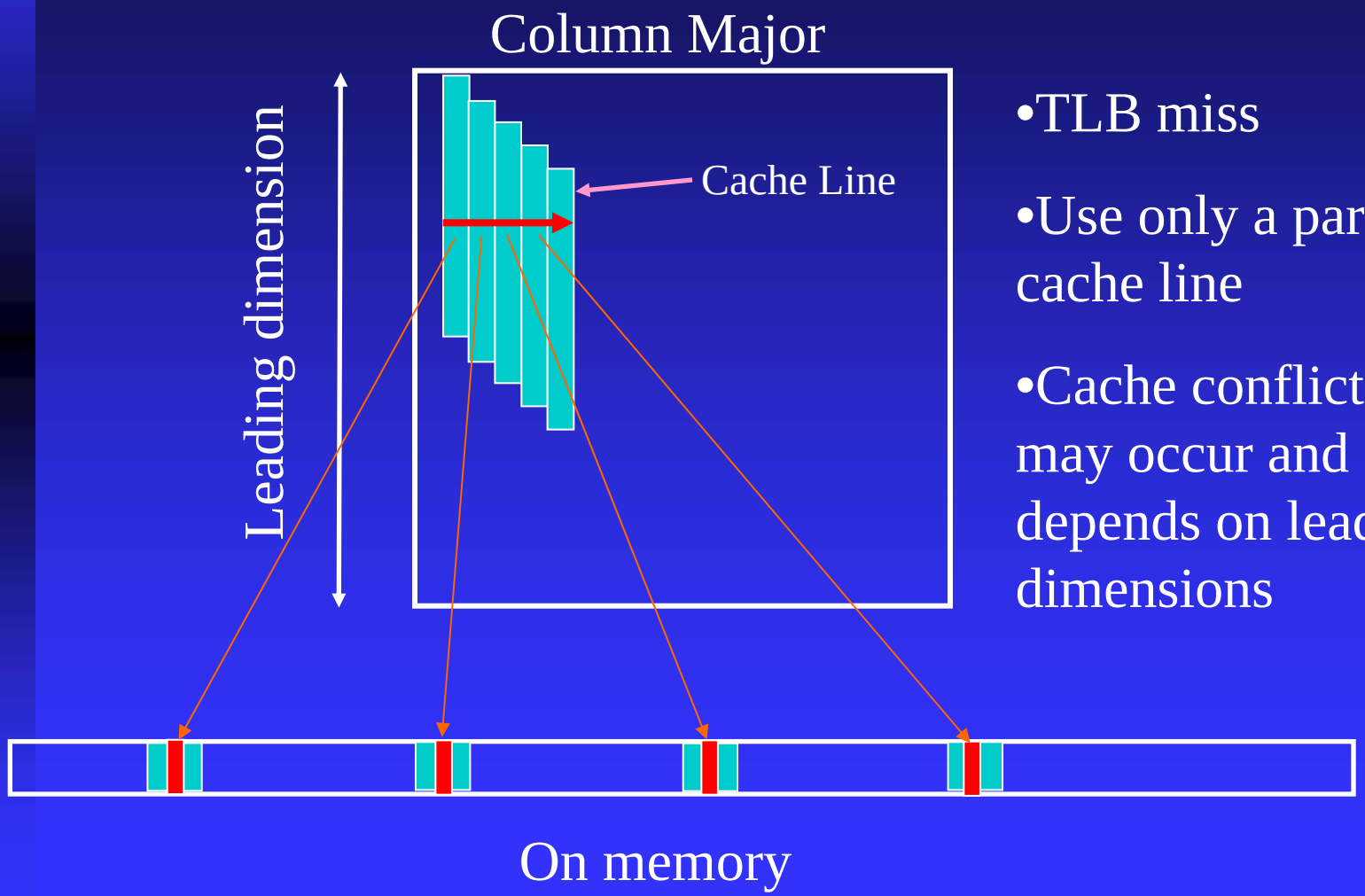
Latency


CPU
L1 cache
L2 cache
Memory
Really close
Near
Far and Far away

Why do we pack data?

- Actual memory location is not contiguous
  - Virtual memory mapping
  - Row or column major, leading dimension
  - Cache line size / associative
- Packing will solve above problem
- Copy (packing) overhead is a headache

Matrix in memory



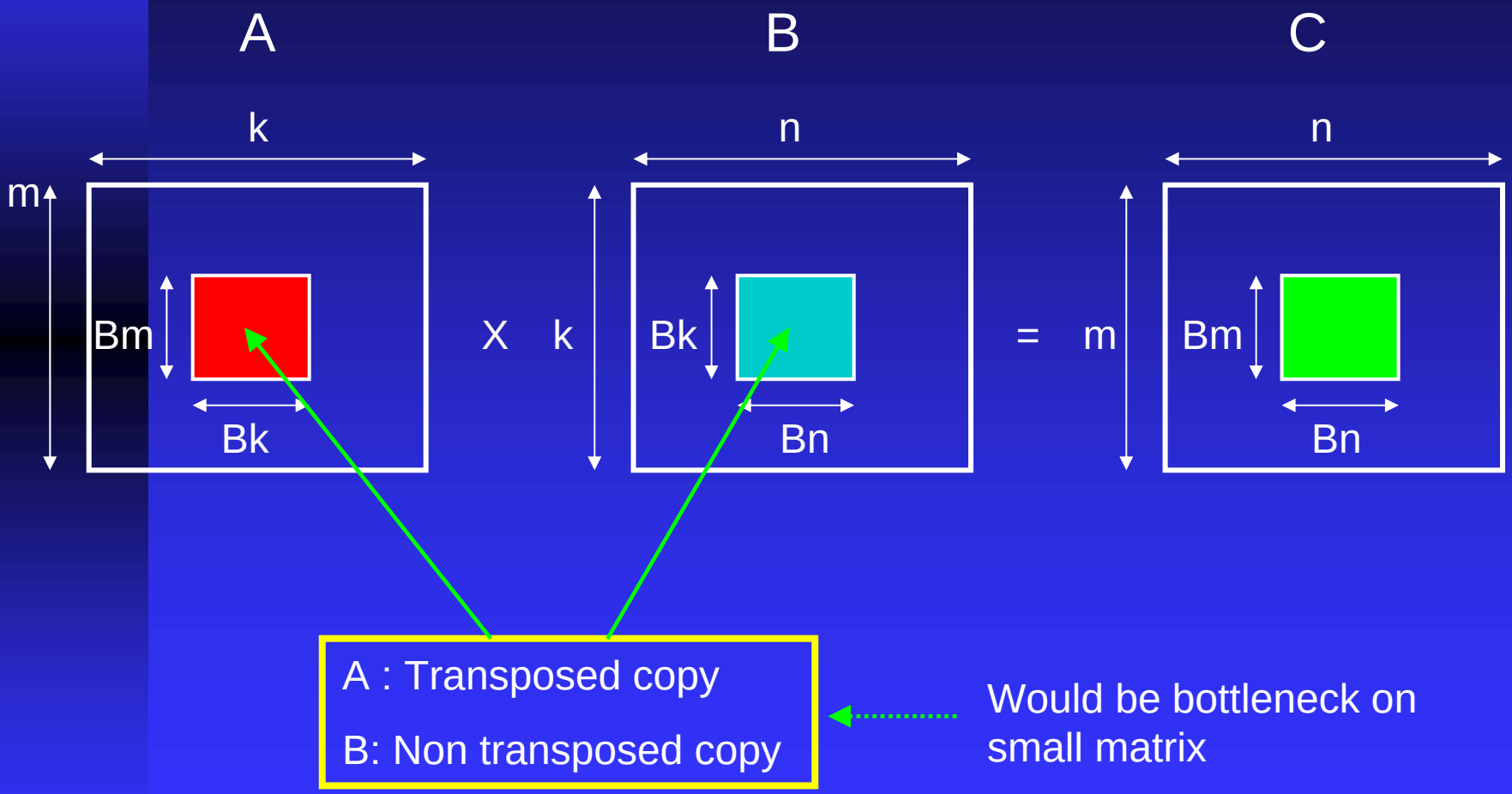
- TLB miss
- Use only a part of cache line
- Cache conflict may occur and depends on leading dimensions

Packing will

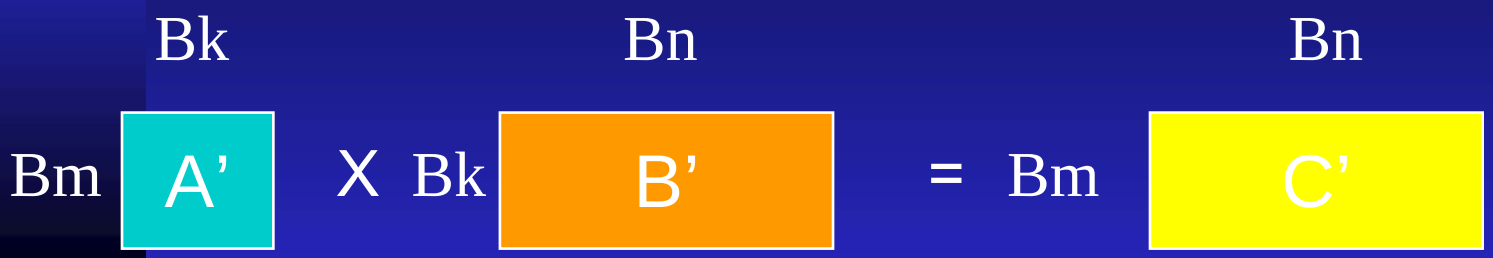
- Reduce TLB misses
- Increase effective cache size
- Reduce required bandwidth
- Help hardware/software prefetch to work effectively

- Need extra buffer space
- Copy overhead

Packing Algorithm



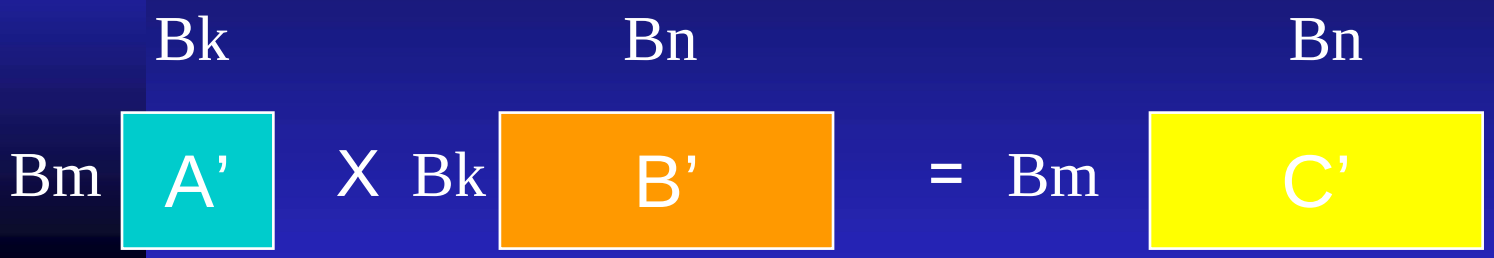
Blocking on L1 cache



What's the problem?

- Kernel may perform 100%
- Blocking size is Bm = Bk = 64 ~ 80
- If blocking size is too small, copy overhead is heavy
- Copy overhead is 20% of total computation time. Total performance will be 80% of peak at most.

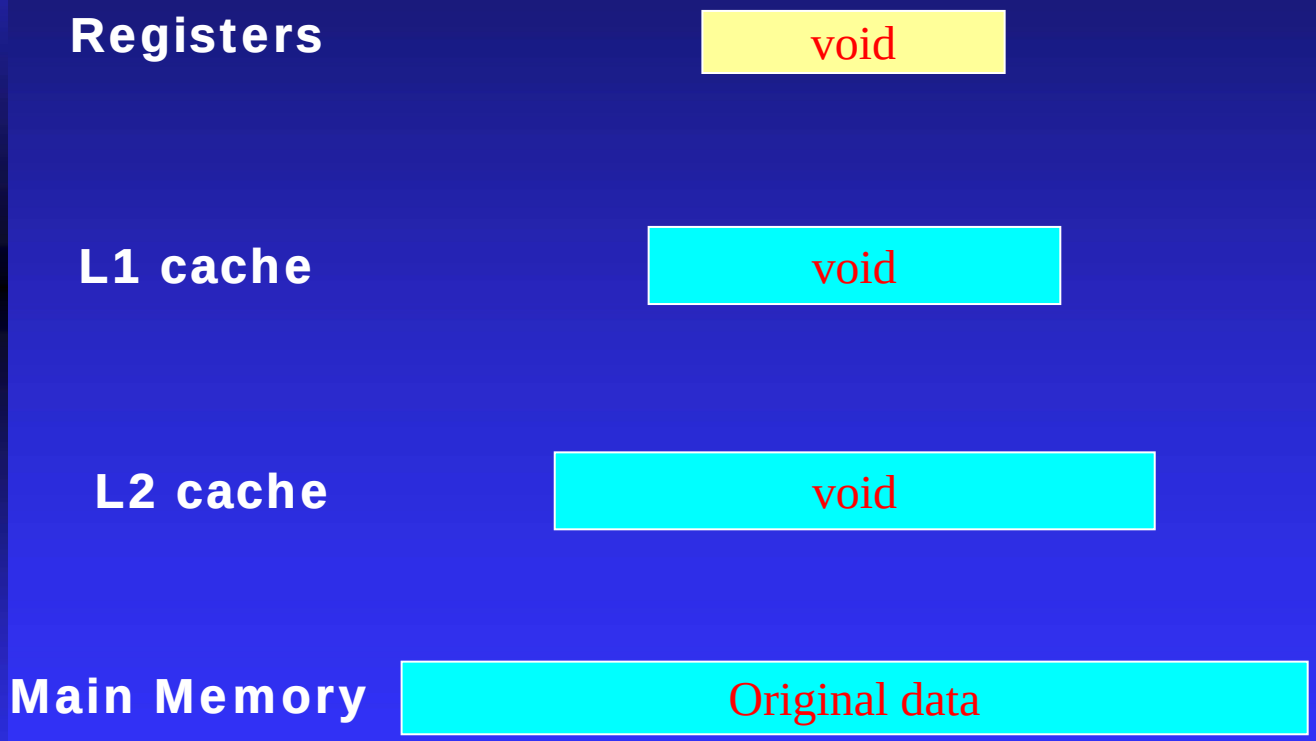
Increasing Blocking Size



Solutions

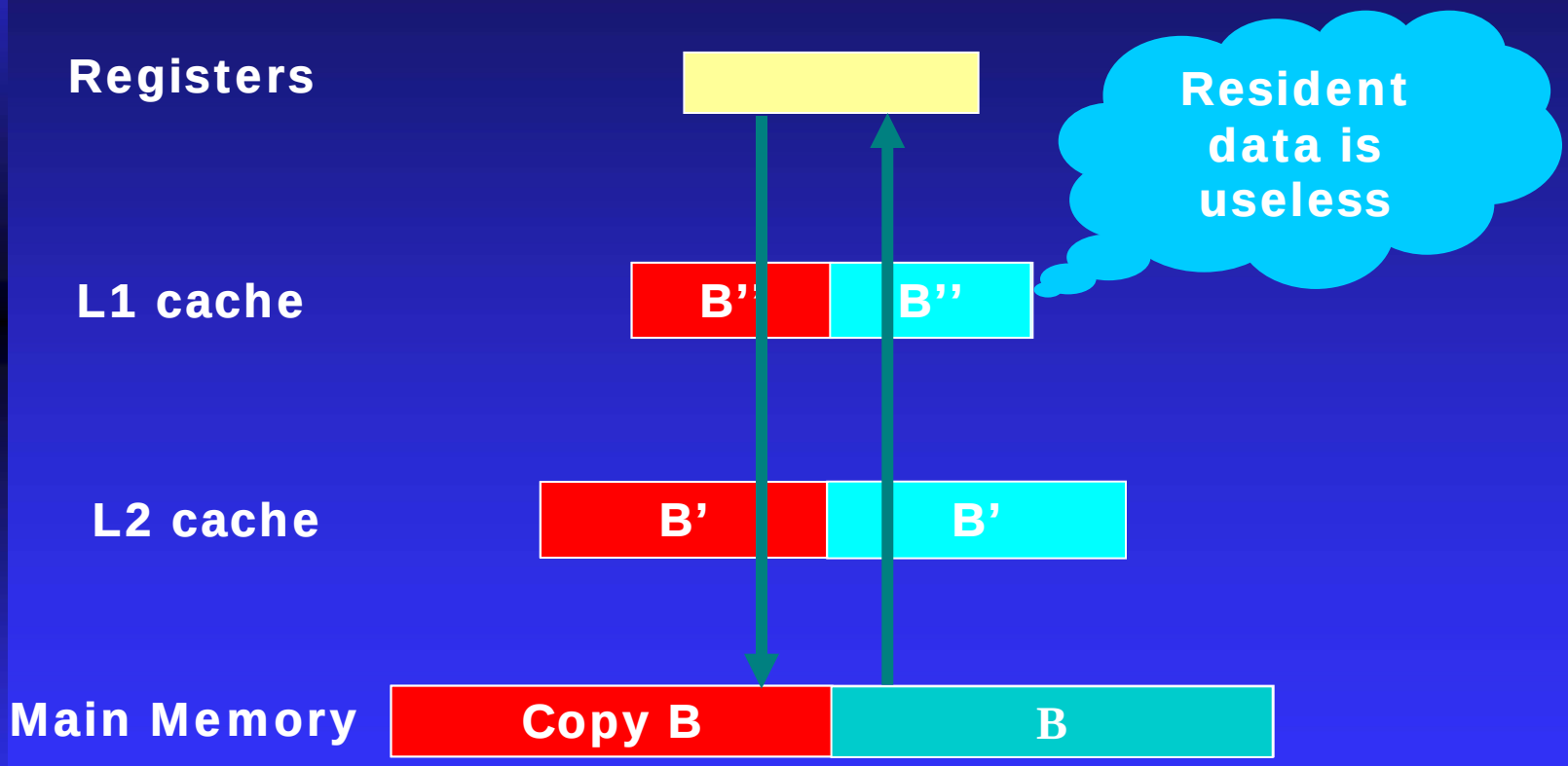
- Bm, Bk >= 256
- Copy overhead is less than 1% of total computation
- Streaming data from A (on L2) is a key

DGEMM kernel (1)
-- START --



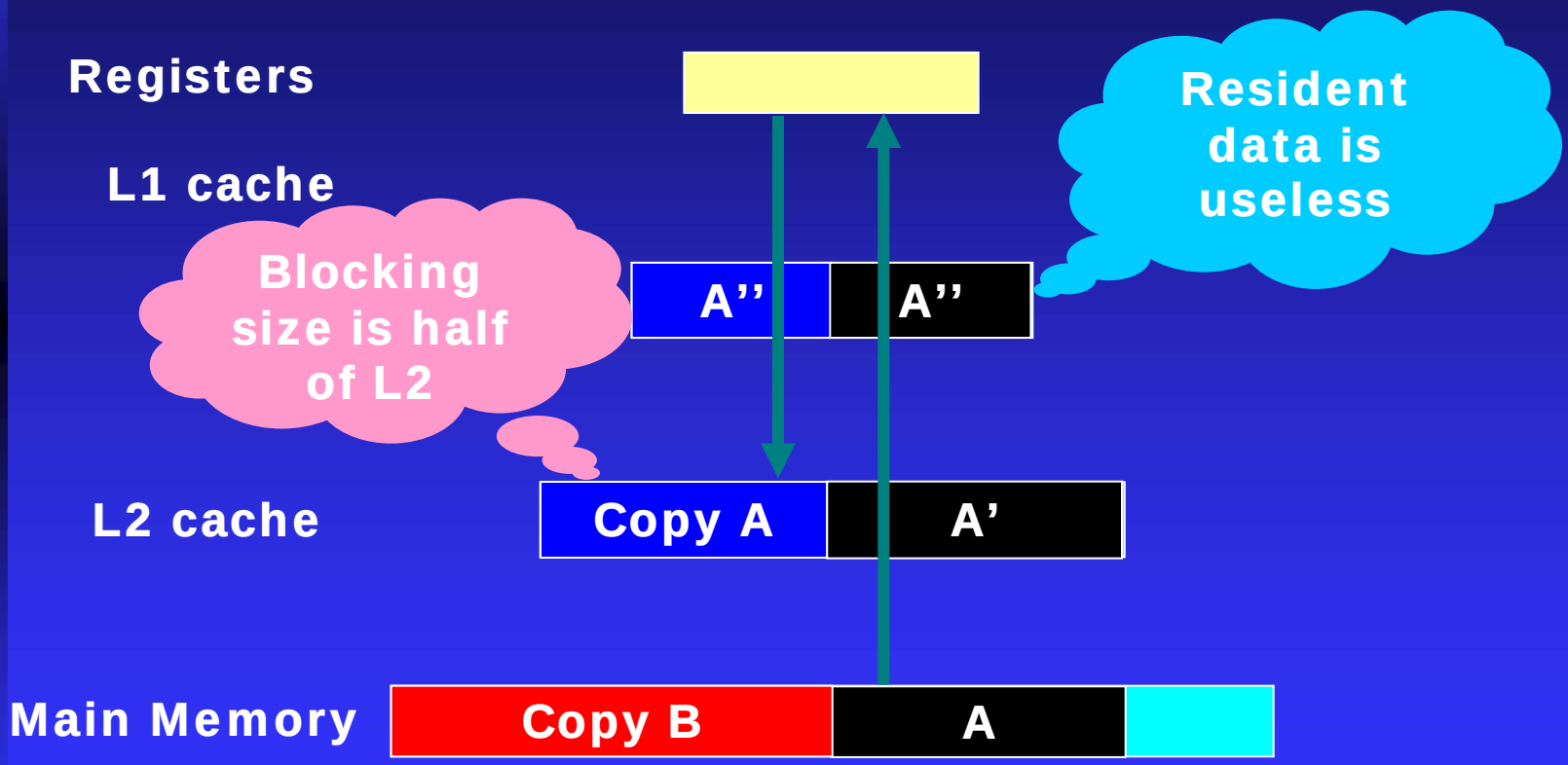
DGEMM kernel (2)

-- Copying for B --



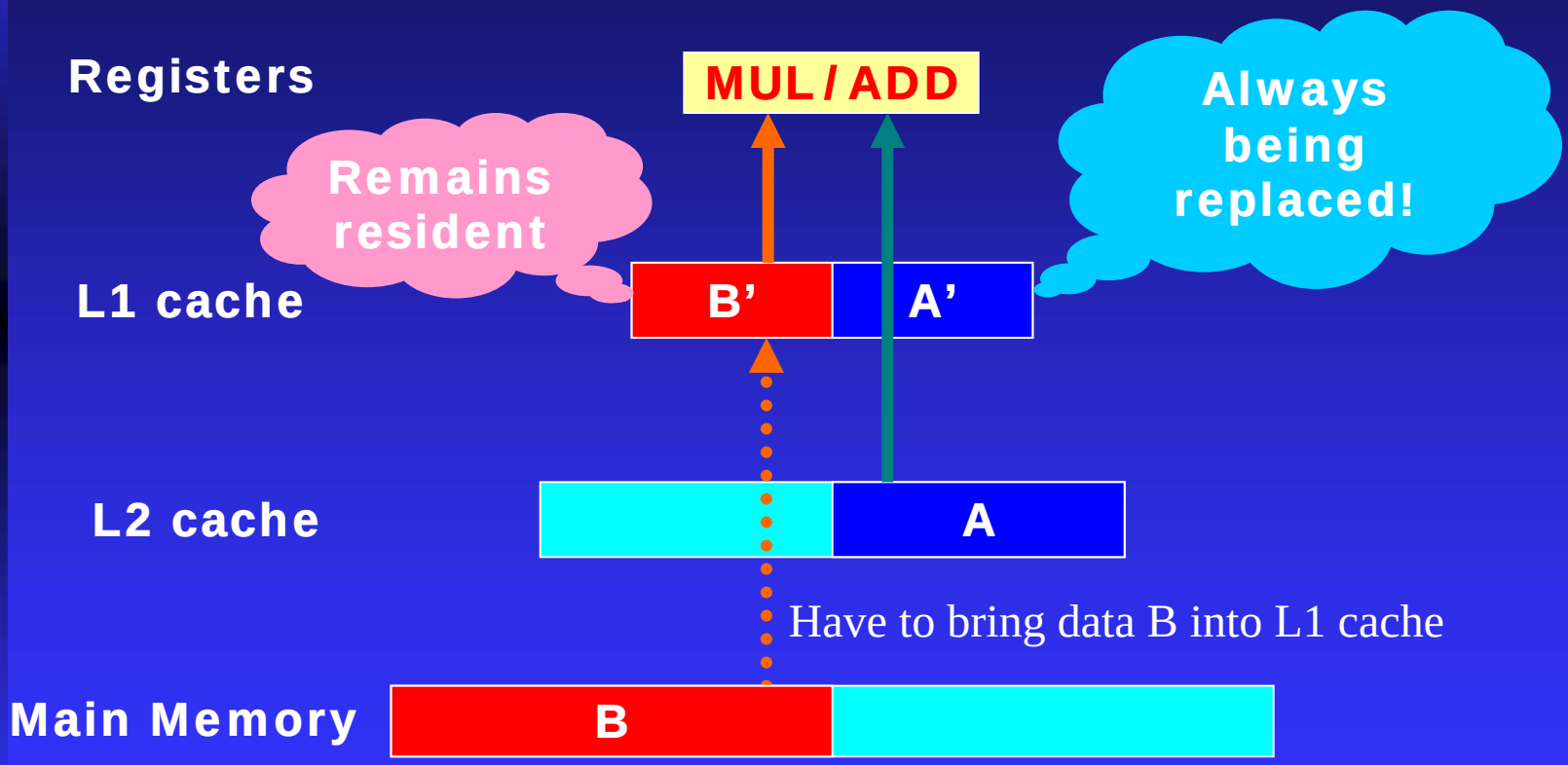
DGEMM kernel (3)

-- Copying for A --



DGEMM kernel

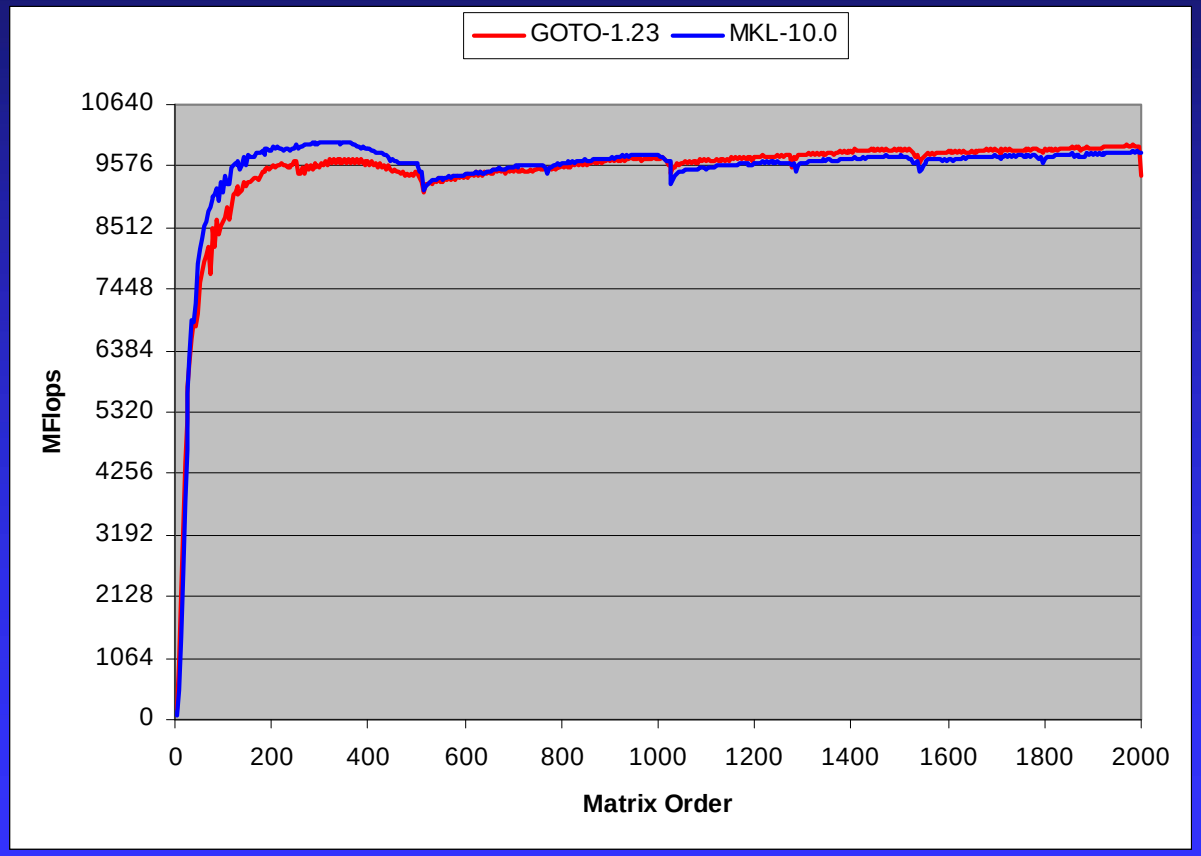
– two streams operation -



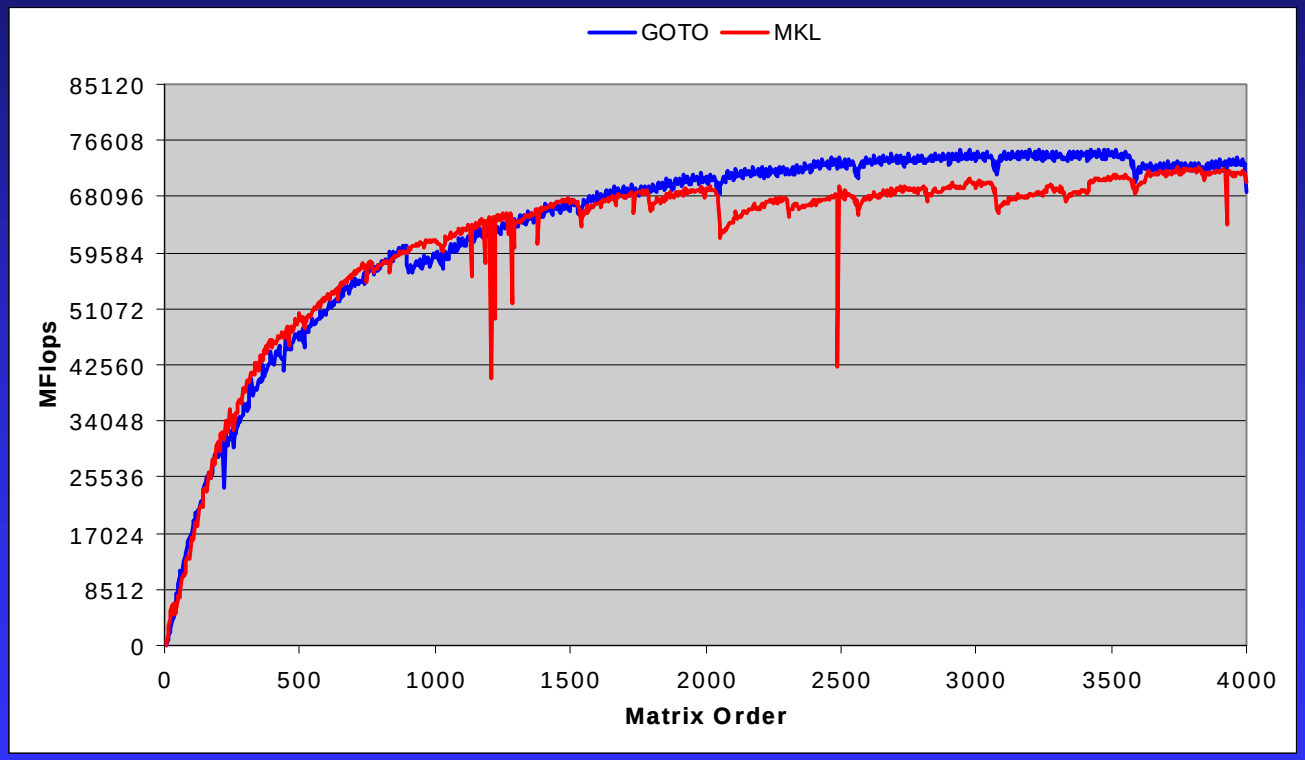
Problem and Solution

- Operation is against cache policy (LRU)
  - B' is on L1 cache, A' is always replaced
  - Use prefetch instruction to give a hint
- Bandwidth from L2 is relatively narrow
  - Control it by changing unrolling type
- Latency from L2 is pretty long
  - Use software prefetch

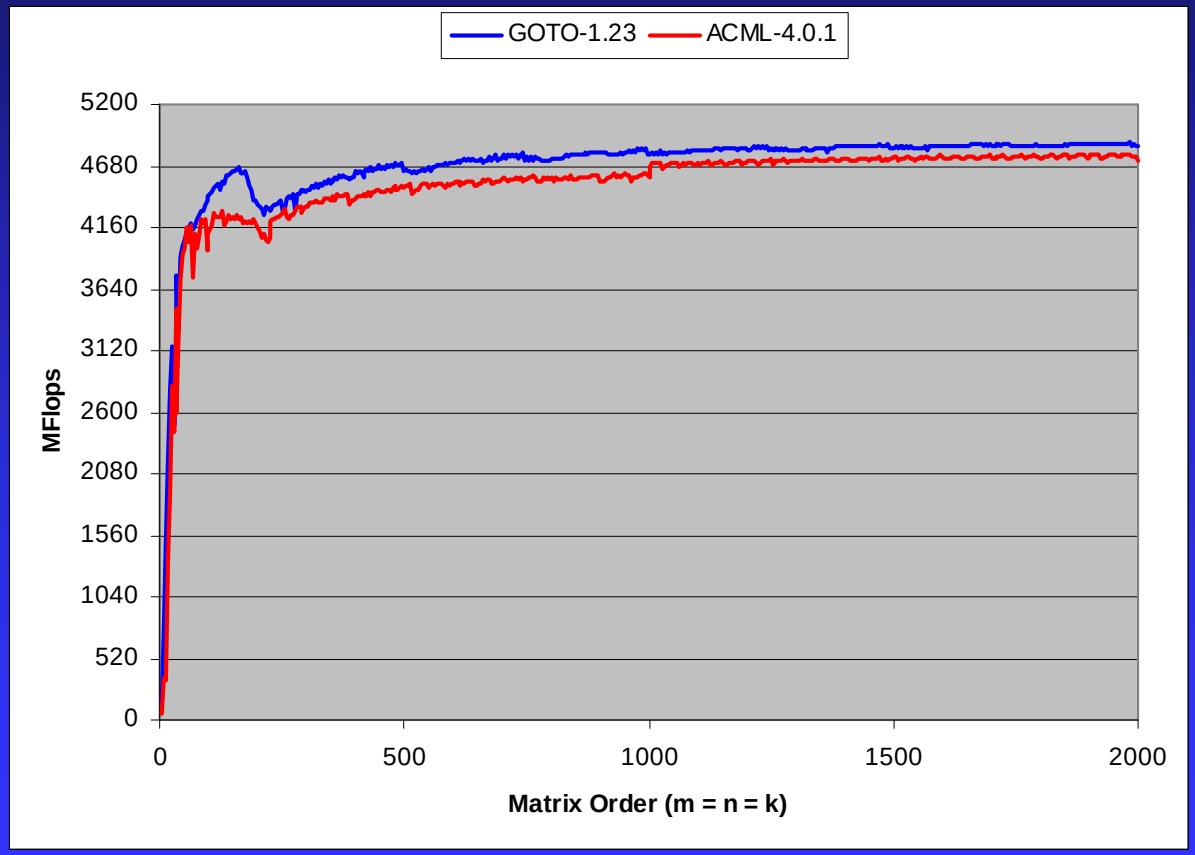
Intel Core2(2.66GHz) performance



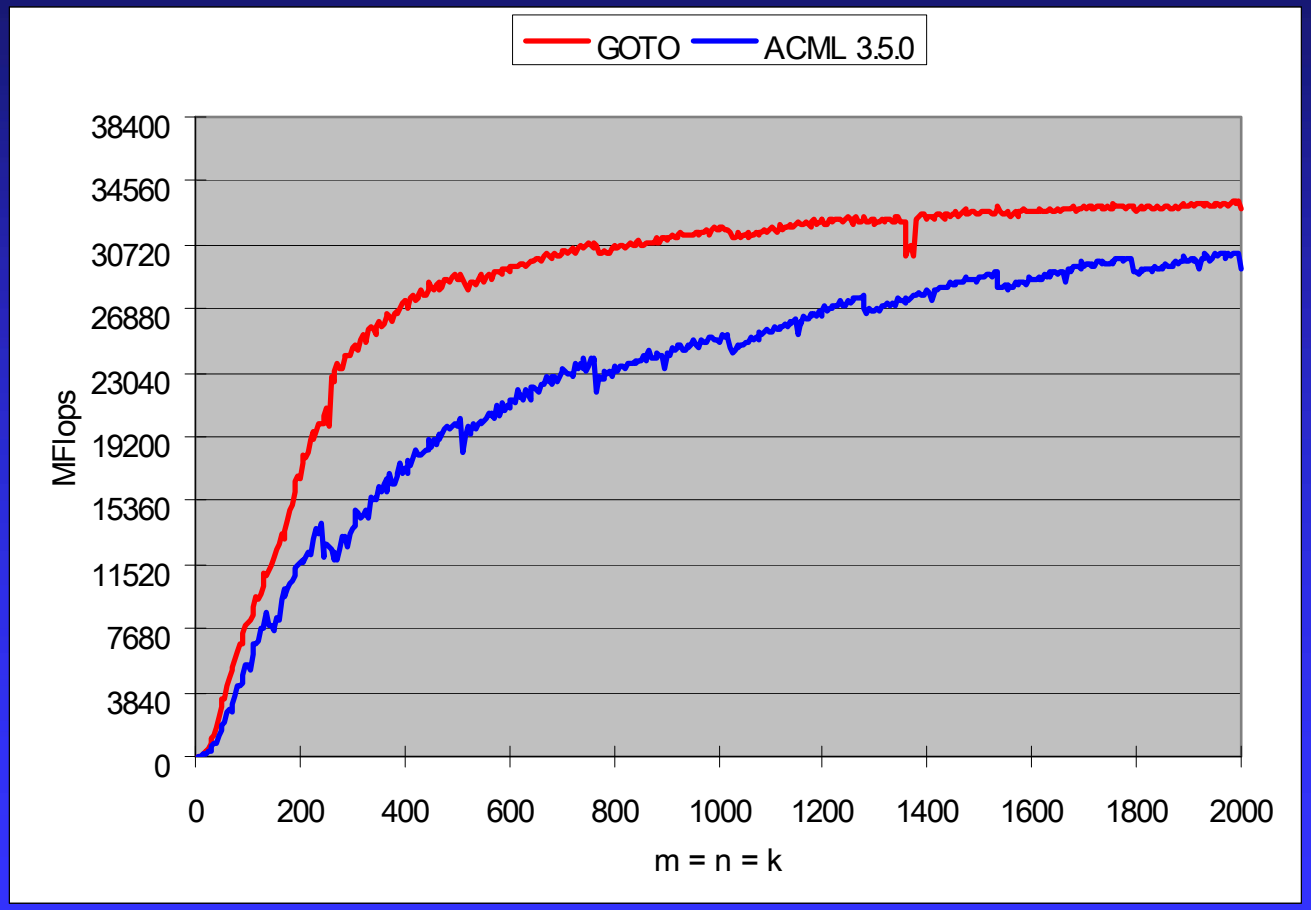
Core2 x 8(2.66GHz) Performance



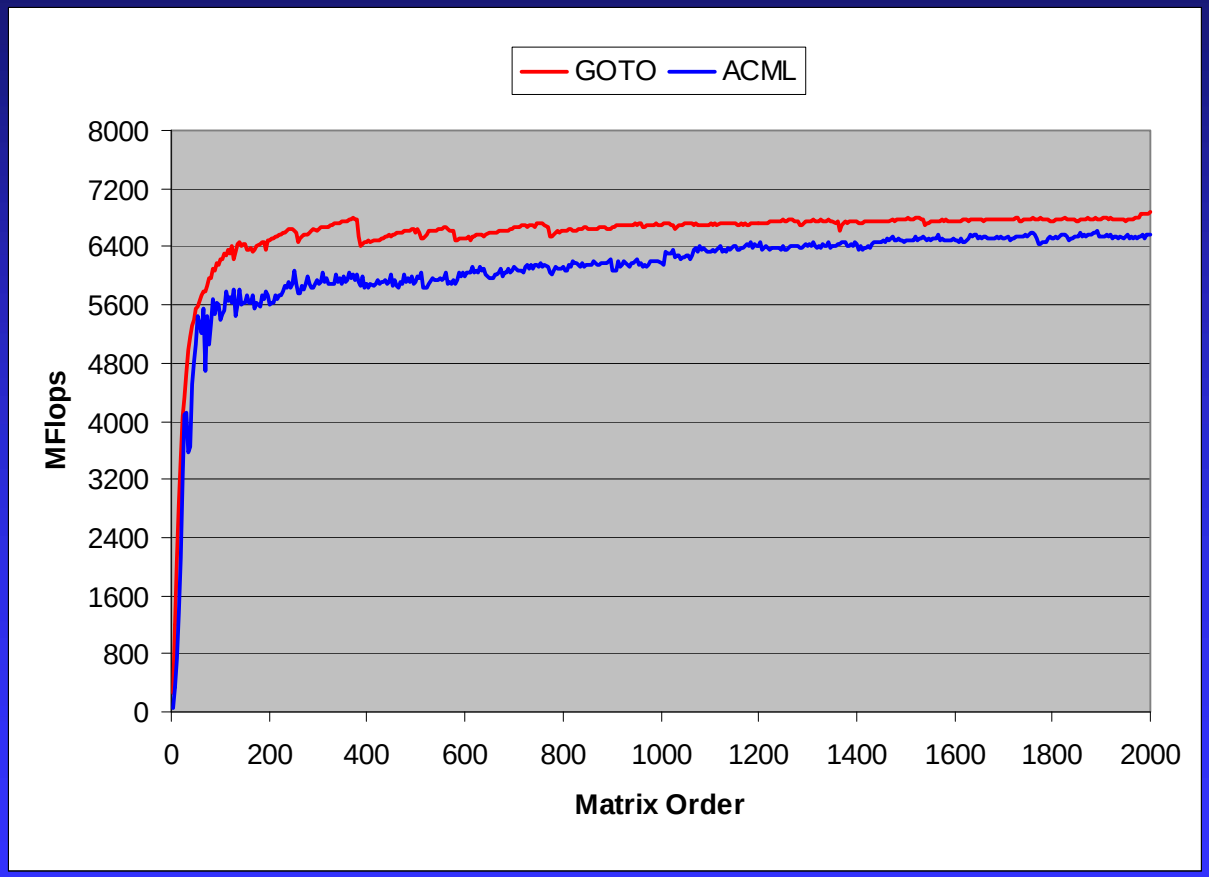
AMD Opteron(2.2GHz) Performance



Opteron x 8 (2.2GHz) Performance



AMD Barcelona(2GHz) Performance



Barcelona x 16 (2GHz) Performance