Computer Architecture HW3

Fabian Wüthrich

November 3, 2020

1 Critical Paper Reviews [500 points]

see here

2 Genome Analysis I [150 points]

2.1 Edit Distance

a) `M O N - - - - T A G`
`s - - i i i i - - -`
`D O N N E R S T A G`
Edit Distance: 5

b) `F R E I T A G`
`s s s s - - -`
`S A M S T A G`
Edit Distance: 4

c) `D O N N E R S T A G`
`d d d d d d d d d d`
`- - - - - - - - - -`
Edit Distance: 10

2.2 Read Mapping

a) read1: 6 + 3 + 3 = 12
read2: 2 + 5 + 6 = 13
read3: 2 + 0 + 0 = 2
read4: 3 + 2 + 5 = 10
read5: 6 + 5 + 3 = 14
51 invocations

b) read1: 10, 15, 20 → 1
read2: 50, 55, 60 & 615, 620, 625 → 2
read3: - → 0
read4: 610, 615, 620 → 1
read5: 710, 715, 720 → 1
5 invocations

c) AAAAA and GGCCT are above threshold.

read1: $0 + 3 + 3 = 6$

read2: $2 + 0 + 0 = 2$

read3: $2 + 0 + 0 = 2$

read4: $0 + 3 + 2 = 5$

read5: $0 + 0 + 3 = 3$

18 invocations

3 Genome Analysis II [150 points]

a) {AAAA} → {1}
{ATAC} → {5, 17, 41}
{TGAT} → {9, 43}
{CCTT} → {13}
{GTTG} → {21, 37}
{TAAG} → {25, 45}
{GTTT} → {29}
{CAAA} → {33}

b) {AAAA} → {1}
{ATAC} → {5, 17, 41}
{GTTG} → {21}
{CAAA} → {33}

c) non-overlapping 4-mers: $2^{log_2(8)} + 13 = 21$ bytes

non-overlapping 4-mers minimizers: $2^{log_2(2)} + 6 = 10$ bytes

d) **non-overlapping 4-mers**

3 cycles for hash table query

144 cycles for edit distance calculation

total cycles: $3 + 2 \times 144 = 291$ cycles

wasted cycles: $0.9 \times 291 = 262$ cycles

non-overlapping 4-mers minimizers

2 cycles for hash table query

144 cycles for edit distance calculation

total cycles: $2 + 144 = 146$ cycles

wasted cycles: $0.8 \times 146 = 117$ cycles

4 RowClone [150 points]

a) Step 1:

The memory controller writes back any dirty cache line from the source region.

Step 2:

The memory controller invalidates any cache line (clean or dirty) from the destination region that is cached in the on-chip caches.

b) To perform the copy operation without RowClone, the memory controller transfers the source page (physical page 10) over the data bus to the caches. Then, the destination page (physical page 12) is written back to main memory. A page has 8KB and the memory controller moves two pages i.e. 16KB over the memory bus.

With RowClone the copy is performed inside the DRAM, so now data is transferred over the memory bus.

Amount of data eliminated: 16KB

5 Tiered-difficulty [150 points]

a) row 0 and row 2

b) row 1 and row 3

c) Place rows that support other requests in near segment instead of frequently used rows.

d) **Exclusive Design**

+ almost full DRAM capacity available

– swapping requires 3 operations (increased latency)

Inclusive Design

+ smaller tag storage

– DRAM capacity is reduced by the size of the near segment

e) A: 5 rows B: 3 rows C: 0 rows

6 Low-Latency DRAM [150 points]

a) In the near segment, the bitlines to a row are shorter so that a smaller electrical load is required, which reduces the access latency.

b) In the far segment, the bitlines to a row are longer so that a higher electrical load is required, which increases access latency.

c) $\frac{16}{512}$

d) Each entry in the tag store has a $\lceil \log_2(496) \rceil = 9$ bit tag, 1 valid bit and 1 dirty bit. With 16 rows in the near segment, 4 bits are used to track the MRU entry. Thus, the total tag store size is $16 \times 11 + 4 = 180$ bits.

e) $\frac{1}{512}$ (swap operation requires one row)

f) Now each row requires the tag of the physical page whose data is stored in that row. Thus, each row requires a 9-bit tag store. An exclusive cache doesn't require a valid or dirty bit so the near segment needs just 4 bits to track MRU. Total Size: $511 \times 9 + 4 = 4603$ bits (swap row excluded).

g) *True* The virtual address translation doesn't work with chunks smaller than the page size.

False The OS controls the virtual-to-physical mapping so it can also model an inclusive cache (map frequently used page in near and far segment).

False An inclusive cache has obviously a capacity loss and even the exclusive cache requires one row for the swap operation.

True This is only true if the cache is exclusively managed by the OS and now support from the memory controller (e.g. store tags efficiently) is provided.