CS3210 Tutorial 3

AY 25/26 Sem 2 — github/omgeta

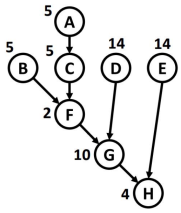
Q1. Instruction-Level Parallelism: pipeline, branch prediction, superscalar
Thread-Level Parallelism: Simultaneous Multi-Threading (SMT)
Processor-Level Parallelism: multi-core running in parallel, UPI shared memory, Omni-Path distributed memory
IO Parallelism: Multiple DDR4 channels and PCIe lanes

Q2.
(a) SISD: single core with single instruction stream on each memory address
(b) MIMD: parallel instruction streams on different cores can act on different data
(c) SIMD: vector instructions act over multiple memory locations
(d) SISD: single core with single instruction stream, pipelined instructions still act on a single memory address at a specific time step
(e) MISD: same paper (data) operated on by different students (instructions)

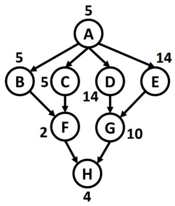
Q3.
(a) False; Shared-Memory can be UMA or NUMA
(b) False; data locality affects latency
(c) True

Q4.
(a) False/True; binary semaphores emulate mutual exclusion but code depending on mutex ownership and semantics may be affected
(b) False; it depends on overheads, synchronisation and workload

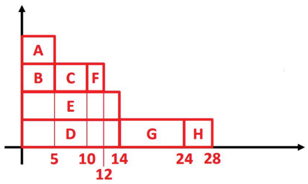
Q5. (a) Fragment 1:



Fragment 2:

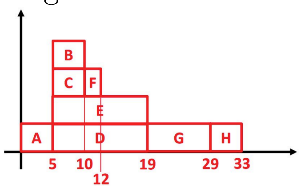


(b) Fragment 1:



Max Concurrency $= 4$, Average Concurrency $= \frac{59}{28} = 2.1$

Fragment 2:

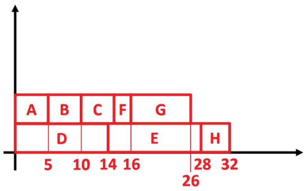


Max Concurrency $= 4$, Average Concurrency $= \frac{59}{33} = 1.78$

(c) Fragment 1 Speedup $= \frac{59}{28} = 2.1$

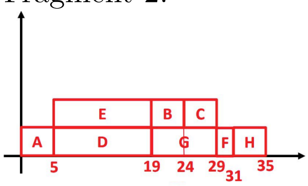
Fragment 2 Speedup $= \frac{59}{33} = 1.78$

(d) Fragment 1:



Speedup $= \frac{59}{32} = 1.84 < 2.1$

Fragment 2:



Speedup $= \frac{59}{35} = 1.69 < 1.78$

Q6. (a) Data Parallelism; SIMD; each independent output element c_i is an independent dot product of the vector b with row a_i

(b) Parbegin-Parend

Q7. (a) Task Parallelism: each stage is a task
Pipelining: best for cases where each stage takes a similar time

(b) Task Parallelism: each stage is a task
Producer–Consumer: producer reads from socket and processes, consumer writes back to socket and there are equal number of producers as consumers

(c) Task Parallelism: each stage is a task
Producer–Consumer: same as (b) but more producers than consumers

(d) Task Parallelism: each stage is a task
Task Pool: pool of tasks are ready for jobs and assigned as necessary

(e) Same; but need explicit communication

Q8. Average CPI (Translation 1) $= \frac{10}{5} = 2$
Average CPI (Translation 1) $= \frac{9}{6} = 1.5$

Q9. Execution Time $(A_1) = \frac{(5+1\cdot 2+1\cdot 3)\times 10^9}{2\times 10^9} = 5s$
Execution Time $(A_1) = \frac{(10+1\cdot 2+1\cdot 3)\times 10^9}{2\times 10^9} = 7.5s$

Conclusion: MIPS is not an accurate measurement of performance

Q10. (a) Sequential Time: $\frac{(100+100^2)\times 2}{10^9} = 20200ns$
Parallel Time $(p = 10)$: $\frac{(100+\frac{100^2}{10})\times 2}{10^9} = 2200ns$ (9.18× speedup)
Parallel Time $(p = 100)$: $\frac{(100+\frac{100^2}{100})\times 2}{10^9} = 400ns$ (50.6× speedup)
Parallel Time $(p = \infty)$: $\frac{(100+0)\times 2}{10^9} = 200ns$ (101× speedup)

(b) From (a), $T_1 = 20200ns$
For $p = 10$,
$$\frac{(N + \frac{N_2}{10}) \times 2}{10^9} = 20200ns \implies N_{p=10} = 312 \implies S_{10}(312) = 9.669\times$$
For $p = 100$,
$$\frac{(N + \frac{N_2}{100}) \times 2}{10^9} = 20200ns \implies N_{p=100} = 956 \implies S_{100}(312) = 90.58\times$$