# UG Exam Performance Feedback
# Second Year
# 2017/2018 Semester 2

COMP25212 System Architecture

Javier Navaridas-Palma
Antoniu Pop

Comments

Students did very well on problem-solving and logic questions, and less well on bookwork parts. There were few misunderstandings of questions, off topic responses mostly showed a lack of understanding of the topic in general rather than the question, or misreading a 6 as a 5. When this occurred, consistent answers were accepted.

Question 1:
Great to see that the majority has a good grasp on the key notions around cache operation and how programs can take advantage of caches to improve performance. Very satisfactory average performance.

- Q1a: Good understanding of cache miss types and how they occur - with high results for most students.

- Q1b: Good knowledge of the types of locality exploited by caches, but some students misread the question: it asks about program optimizations that can improve locality - not about hardware/architectural optimizations.

- Q1c: The results in question 1c were very polarized, with about a third of students achieving perfect or near-perfect scores (well done!) and a third having low to null scores in this question. The first two groups showed on average a very good grasp on how cache hierarchies operate, while the latter group showed a lack of understanding of the basic notions of cache access hit/miss ratios and how these impact performance.

Question 2:
- Q2a: Very solid performance in this question. The vast majority of students understands the key concepts and tradeoffs in RAID systems, would be able to reason about the provision of storage systems.

- Q2b: Average performance in this question was very satisfactory. A common mistake in this question comes from not understanding the notion of nested RAID or failing to account for all three layers in a RAID 160. This specific configuration has been covered in lectures with a very similar example. Part 2 has been misunderstood in some cases as asking what is the storage efficiency of this system, and correct answers have been accepted. Part 2 and 3 have been accepted whenever they are consistent with the RAID configuration chosen in part 1, even if that was not itself correct. Part 4 is largely independent as the answer is the same regardless of any of the previous parts. A common mistake has been to overlook the fact that disk operations can proceed in parallel, so the recovery time does not depend on the number of failed drives calculated in part 3.

- Q2c&d: Knowledge about virtual machine operations checkpointing and restoring was shallow and a substantial number of students were not aware of what these operations are or what this can be used for. In 2d, all reasonable uses of the technique were accepted.

Q3a. was answered appropriately by the students and I'm generally happy with their performance. There were many minor issues related with this question. Many students did not identify data dependencies that were not affecting performance.
Many students identified WAR and WAW dependencies when they only affect out-of-order processors and not the standard, in-order pipeline of the question. I did not penalise this but this was common enough to be worth mentioning here.
A few students considered that as all forms of forwarding were implemented data dependencies were never an issue. This is not actually the case; for instance MEM->EX cannot be solved only by forwarding. Many students forgot to take into account in their calculations the extra cycles to go through the pipeline or did not use the correct number (4 in the case of a 5-stage pipeline). Some students calculated the execution time as if it was a non-pipelined processor, but added the penalties as if it was pipelined. Some students draw an execution diagram, rather than calculating the execution time from the number of instructions, the penalties and the tail of the pipeline. This has not been penalised, as it has already put these students in a disadvantage as drawing the diagram has probably taken much more time than required for this exercise. In the cases were dependencies were not explicitly discussed but were respected in the execution diagram, I marked as if only the ones respected were identified. CPI was generally calculated correctly, but some students calculated IPC instead, and a few others used incorrect functions.

Q3b. Good performance in this question. Most students were able to find a better instruction ordering and most issues were from calculating the execution time and CPI (as above).

Q3c. The performance for this question was rather poor, with most students not being able to identify most of the main aspects of the two architectures. Centralised versus decentralised and register renaming, the two main aspects to remember from the architectures, were only discussed by a subset of the students. I'm under the impression that most students disregarded this part of the course as it hadn't appear in the exams before.

# UG Exam Performance Feedback
# Second Year
# 2017/2018 Semester 2

Q4a. Very good performance in this question. I'm glad to see that this important aspect of multicore computing has been grasped by students. There were some common minor issues in this question. The most important is mixing up consistency and coherence.
It was also relatively common for students to assume that broadcasting/updating values is the only way to keep coherence, but invalidate protocols also exist and are, indeed, more common.
Also some believed that updating main memory with every write was required, which is not the case.
Finally, some students said the main reason for cache coherence in multicores is performance; but not having incorrect results is much more important.

Q4b. Rather unsatisfactory performance for this question. Most students were able to mention only a couple of differences and most of the time were not even justified properly (e.g. directory is more scalable, without any reason for that).
Many students mentioned the Common Data Bus, which is part of Tomasulo architecture, as being part of snoopy protocols. Snoopy protocols use a regular bus for that.
Also some students said that Directory-based protocols rely on the use of broadcast, which is not normally the case.

Q4c. Good performance overall with this question. The most common mistakes, which I decided not to penalise, were not writing the units and/or orders of magnitude. Also many students did not understood correctly the effects of threads on peak IPC and Throughput. Finally, it was also relatively common to have students believing that a non-multithreaded multicore processors has only one (or none) hardware thread, instead of one per core.

Q4d. The performance for this question was much worse than I anticipated as I expected this question to be easy for students. Indeed, one fourth of the students skipped it.
Many students used the provided IPCs straight away. This is not enough to actually model the performance since the clock frequency and the number of threads have a great impact on the performance. Students needed to realise that phase i) uses a single thread, so the performance for that phase was IPC*clock, whereas phase ii) is fully parallel and so can use all available threads in parallel, so performance here is threads*IPC*clock.
It was also rather common to just pick some values from the above exercise as justification. Only in very few cases, if the justification was reasonable enough, I gave some marks to such answers.
Finally, I appreciate some students tried to apply Amdahl's law to solve this exercise, alas, it wasn't really the way to go as Amdahl's law does not consider the clock frequency