Javier Navaridas-Palma
Antoniu Pop

Comments

Questions 1 and 2 - Antoniu Pop

Q1 (Caches):

b & c. A large majority of students had a very good grasp on the concept of cache locality and understood well where temporal and spatial locality is found in programs, but only a few caught the relation between each type of locality and cache characteristics (i.e., replacement policy, associativity, cache size & line size, ...). A common mistake was the assumption that each form of locality was only related to one type of cache organisation (e.g., temporal locality for fully associative cache).

d. Many students misunderstood the question and described HOW a multi-level cache is organised instead of answering WHY multi-level caches are necessary.

e & f. Question (1e) was, by far, the most misunderstood question in part A. It was only correctly answered by 3 or 4 students. This question was not intended to be difficult, but instead to help students, to point them in the right direction for the following questions. Strangely enough quite a few students found the right answer to question (1f) while failing to find the answer to (1e) although the answer was ultimately the same (4 cache lines per set / 1 set used).

Q2 (Virtualization and Storage):

b. This question was often misunderstood as asking for the list of uses of system virtualization.

c. A frequent error was to confuse live migration with freezing/copying/resuming a VM, which is of course not "live" migration.

Overall Question 2 was better apprehended and had better results (on
average) than Question 1. In particular, Storage was very well treated by most students.

Questions 3 and 4 - Javier Navaridas-Palma

I'm generally happy with students' performance in Q3 and Q4, overall high marks were achieved.

From Q3 it seems that most students seem to understand dependencies and the concept of pipelining. Some common errors were dumping the knowledge about pipelines in part a without actually referring to the code, but I've been rather lenient with that. Also not many students remembered that instruction reordering can help to solve some dependencies and even less did give examples that would help with the provided code. Another common issue was to consider structural hazards a problem in in-order pipelines.

The dataflow diagram for the code was acceptable in many cases. The most common mistakes were putting a dependence 13à14 which are completely independent and a dependence 10à11, where the actual dependence is 9à11.

With regards to Q4, there seems to be more common than I'd like for students to mix up the terms 'concurrent' and 'parallel/simultaneous'
within the discussion for part a. For part b, most students understood the benefits of reordering the code but the pros/cons of doing it in SW or HW were generally a bit weaker.

In part c, a significant amount of students forgot the wasted cycle due to ICMs in coarse grain MT. A few of them have given the answer (for coarse and fine grain) as it if was a scalar, rather than superscalar, processor, which I have penalized (1 mark rather than 2, when correct).
With SMT, a few students issued instructions without complying with the original dependencies.