UG Exam Performance Feedback
Second Year
2017/2018 Semester 2

COMP26120 Algorithms and Imperative Programming

David Rydeheard
Ian Pratt-Hartmann
Milan Mihajlovic

Comments

Q1: This question is closely related to the examinable lab that the students do over the three week period in the second semester. The first part of the question is the standard book work and was attempted and successfully answered by almost all students. The second part required the transformation of the weights in the graph before a systematic application of Dijkstra's algorithm to obtain the optimal travel time. The transformation was got right by most of the students, but in the second half a number of students took shortcuts, resulting in lost marks. In the final part of the question asymptotic complexity of Dijkstra's algorithm was sought for various implementations of the priority queue and when applied to different types of graphs (dense/sparse). A solid number of students got this part right, which is a good news, as complexity is the material they usually struggle with. The overall performance was very good, with the average mark probably above 70%.

Q 2 a) Far too many candidates were incapable of giving pseudocode for backtracking though all subsets of a set. The most obvious solutions are to use recursion or to set up a queue (popping off partial solutions anmd pushing their two extensions). Either was fine. Quite a few students used a counter to 2^n. This is a viable solution, but anyone who helped himself to a function to convert numbers to binary strings (with no explanation) lost a couple of marks. Candidates who referred to the "next permutation" or to "stepping through all permutations" without explanation or proper pseudocode obtained only minimal credit. b) Most people got this right: O(2^n). Candidates who gave a wrong algorithm for a) but the correct complexity of that algorithm still got one mark. c) Most candidates had revised the DP solution to the 0/1 knapsack problem, and promptly regurgitated it. But that was not what was required: the 0/1 knapsack problem is an optimization problem, while the present problem is a YES/NO problem. So you have to be careful what you store in the DP matrix: it should really be a Boolean. More importantly, you have to get the conditions on the branch in the middle of the DP loop right: there is no notion of a "best answer so far" here. Actually, when you think about it, the main DP matrix is the same as the keep-matrix; but that is inessential. A few Candidates also wrote pseudocode which did not produce the output specified. But the question was marked leniently, and most Candidates did okay. d) Almost all Candidates knew that this was O(s.n).

Q 3. Hashing and quadratic probing.
The average for this question was just over 50%.

Those who knew how quadratic probing worked tended to get very high marks (>70%). Many others guessed and often used an incorrect algorithm, either not understanding the formula or not rehashing on expansion of the hash table. This rehashing is necessary to allow quick look-up.

The final part is about patterns of entries using quadratic probing - many gave rough answers, sometimes poorly expressed. The answer is that a form of clustering can occur with quadratic probing - details of this were required.