LUT University

LUT Computational Engineering 2022-03-30

BM40A0202 Foundations of Computer Science

Olli-Pekka Hämäläinen

Exercise 11 (week 14): Algorithms and complexity.

Tasks (1 p/task)

1. If the complexity of the module `algo` is linear, i.e., upper bound is O(n), then what is the complexity of the module below? Justify your answer.

```
MODULE rithm(n) RETURNS multi
  multi := 1
  FOR i := 1, ..., n/2 DO
    multi := multi * algo(i)
  ENDFOR
  RETURN multi
ENDMODULE
```

   Hint: The $O(n)$ order of magnitude means that the module `algo` requires an order of magnitude n steps (typically $constant * n$). Based on this, calculate the order of magnitude of the total number of steps required by the `rithm` module.

2. Let's examine the following complexity functions:

$$
\begin{array}{cccc}
(n-1)^3 & \log_{10} n & n^{1/2} & n! \\
\log_2(\log_2 n) & 3n & 2^{n+1} & n \log_2 n \\
n(n+1)/2 & \log_2 n^2 & n^2 & n^2/(2n+1)
\end{array}
$$

   a) Group the functions $T(n)$ into groups, i.e. magnitude classes according to their time complexity exponential, polynomial, linear, logarithmic and constant time. Arrange the functions within the groups from largest to smallest when n is large. (The results of b) can be used as a help.)

   b) Calculate the values of all complexity functions for three input sizes $n = 10$, $n = 100$ and $n = 1000$. Present the results in tabular form with two or three decimals, using decimal powers for larger numbers.

   Hint: www.wolframalpha.com can calculate even the largest numbers needed, and you can round them to the nearest significant digit, i.e. to the power of ten. Other tools are also recommended, e.g. formula input into Excel/MATLAB and Wolfram calculation only for functions above the range of their values.

3. a) What difference does it make whether we're talking about complexity of a problem or complexity of an algorithm? Or are they the same thing?

PO BOX 20, FI-53851 LAPPEENRANTA, FINLAND
E-mail: Olli-Pekka.Hamalainen@lut.fi

b) Let's say that you should find a route that travels through 24 cities; this kind of problem is called a traveling salesperson problem (TSP). You have a computer and data of the roads, but since this problem is NP-hard and hence infeasible, you probably don't have time to find the exactly most optimal solution. Present some method how you could get a solution that is at least somewhat reasonable in relatively short time. (Note: this doesn't have to be a fancy, highly mathematical algorithm; keep it simple and think practically!)

c) The size of a TSP is considered to be the number of cities. In reality, what other factor has an effect on the complexity of the solution?

4. a) Define an algorithm that finds the integers x and y such that $x^2 + y^2 = n$, where n is a given positive integer. What is the complexity of the algorithm you found?

   b) Could there be an algorithm with complexity lower than $O(n^i), i = 2, 3, ...$? Or even lower than $O(n)$?

5. In a "subset sum problem", one tries to find a set of elements of a vector such that the sum of the set (of elements) is a given number n. For example, if the numbers of a given vector are $\{1, 2, 3, 4, 5\}$ and the given sum $n = 10$, then the solutions are the sets $\{1, 4, 5\}$ and $\{2, 3, 5\}$, $\{1, 2, 3, 4\}$.

   a) Write a program that implements the solution to the problem - that is, to find the set(s) of elements in a given vector whose sum is the given number n. Test your program with different vectors (vectors of different sizes) and different sums n.

   b) What is the complexity of your solution?