Exercises - Dynamic Programming 1 - Algoritmiek

Tutorial February 13, 2024

1. **Basic knowledge:**
   (a) Explain in your own words what memoization is.
   (b) What is the difference between memoization and a "classical DP"?

2. **Exact Change: Alternate and Constructive:** In the previous tutorial, you developed an alternate recurrence for the Exact Change problem. Let's revisit this problem.
   (a) Give pseudo-code for a dynamic program using this recurrence, and analyze its running time.
   (b) Your algorithm probably uses $O(nb)$ space. Explain how you would save space so that you only use $O(b)$ space.
   (c) Consider again the original algorithm, which uses $O(nb)$ space. Explain how to find an optimal solution (an optimal set of coins to pay).
   (d) Hard question: Can you combine saving space and finding the solution in a single algorithm? Explain your answer.

3. **Windmills: constructive:** In the previous tutorial, you developed a recurrence to solve the windmills problem.
   (a) Give a dynamic program for your recurrence and analyze its running time.
   (b) Explain how to find an optimal solution (an optimal set of positions for the windmills, such that they are at least K apart).

4. **Mister Animal (revisited):** Consider Mister Animal and the recurrence that you developed for this problem in the previous tutorial.
   (a) Give a dynamic program for your recurrence.
   (b) Analyze the running time and memory space usage of your algorithm.
   (c) Explain how to find a solution (a way to spend exactly B dollars).
   (d) Can you save memory space? If so, explain how.

5. **A2B revisited:** Consider the following recurrence for the A2B problem: for $a \leq c \leq b$, $K(c)$ is the smallest number of operations to get from c to b. Then:
   $K(c) = \min\{1 + K(c+1), 1 + K(2c)\}$ if $2c \leq b$
   $K(c) = 1 + K(c+1)$ otherwise.
   Note that this solves the problem just as well, but from the 'other direction'.
   (a) What is the base case?
   (b) Give pseudocode for a memoization algorithm for this recurrence.
   (c) Give pseudocode for a dynamic program for this recurrence.

6. **Splitting the inheritance:** You are executing a will and need to split an inheritance of n items for value $v_1, \ldots, v_n$ for two brothers (the items themselves are indivisible). To avoid any issues, the split must be done as fairly as possible. How can you find a fairest split of the items, that is, a split of the items such that the total value of items given to brother 1 differs as less as possible from the items given to brother 2? We will design a dynamic programming algorithm for this task.

(a) Consider as a top-choice which brother gets item i. From this, you formulate the following subproblem: what is the fairest split of the first i items. Explain, by way of an example, that this is not a good subproblem (i.e., give a counterexample to the optimality principle).

(b) Someone suggest as a subproblem: is there a split of the first i items such that brother 1 gets exactly c more in total value than brother 2. Prove the optimality principle using this subproblem.

(c) Give a dynamic programming algorithm for this problem. Use the steps as described in class!

(d) Explain to find an optimal solution (a fairest way to split up the inheritance).

(e) Can you save memory space in your algorithm. If so, explain how.

7. **Multiplication target:** Consider a set Σ of symbols on which an operator $* : \Sigma \times \Sigma \to \Sigma$ is given. This operator is neither associative nor commutative; so if $\Sigma = \{a, b\}$, then it is possible that $a * a = b$, $a * b = b$, $b * a = a$ and $b * b = a$. Suppose you are given n symbols $x_1, \ldots, x_n \in \Sigma$, possibly with duplications. You have to place parenthesis such that applying $*$ leads to a target value $d \in \Sigma$. For example, if $\Sigma = \{a, b\}$ and $*$ as before, and given is $x_1x_2x_3 = aba$ en $d = a$, then $(a * b) * a = b * a = a$ leads to a correct solution, but $a * (b * a) = a * a = b$ does not.
Careful: you are not allowed to change the order of the symbols, only place parenthesis.

(a) Give a dynamic programming algorithm for this problem. Use the steps discussed in class!

(b) Analyze the running time and memory space usage of your algorithm.

(c) Explain how to find a solution (a way to place parenthesis such that applying the operator gets you to d).

(d) Can you save memory space? If so, explain how.