Project 2

Algorithms and Data Structures

25th November, 2020

1 The maximum-subarray problem

Suppose that you been offered the opportunity to invest in the Volatile Chemical Corporation. Like the chemicals the company produces, the stock price is rather volatile (see Figure 1). You are allowed to buy one unit of stock only one time and then sell it at a later date, buying and selling after the close of trading for the day. To compensate for this restriction, you are allowed to learn what the price of the stock will be in the future. Your goal is to maximize your profit.

Figure 1. Information about the price of stock in the Volatile Chemical Corporation after the close of trading over a period of 18 days.

Day	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Price	100	113	110	85	105	102	86	63	81
Change		13	-3	-25	20	-3	-16	-23	18
Day	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
Price	101	94	106	101	79	94	90	97	86
Change	20	-7	12	-5	-22	15	-4	7	-11

1. Implement both the brute-force (just try every possible pair of buy and sell dates in which the buy date precedes the sell day) and recursive algorithms for maximum –subarray problem on your own computer. What problem size n_0 gives the crossover point at which the recursive algorithm beats the brute-force algorithm? Then, change the base case of the recursive algorithm to use the brute-force algorithm whenever the problem size is less than n_0. Does that change the crossover point?
2. Suppose we change the definition of the maximum-subarray problem to allow the result to be an empty subarray, where the sum of the values of an empty subarray is 0. How would you change any of the algorithms that do not allow empty subarrays to permit an empty subarray to be the result?

2 Interval-graph coloring problem

Suppose that we have a set of activities to schedule among a large number of lecture halls, where any activity can take place in any lecture hall. We wish to schedule all the activities using as few lecture halls as possible. Give an efficient greedy algorithm to determine which activity should use which lecture hall. (This problem is also known as the ***interval-graph coloring problem***. We can create an interval graph whose vertices are the given activities and whose edges connect incompatible activities. The smallest number of colors required to color every vertex so that no two adjacent vertices have the same color corresponds to finding the fewest lecture halls needed to schedule all of the given activities.)

1. Try to use greedy algorithm to solve the interval-graph coloring problem and give the algorithm complexity analysis

2. Provide test cases to verify the correctness of your algorithm and Explain it.

3. Beam Search

Decisions about sequence searching strategies lie on a spectrum, with easy questions at either extreme. What if only accuracy matters? Obviously, brute-force search. What if only computational cost matters? Clearly, greedy search. A real-world applications usually asks a complicated question, somewhere in between those two extremes.

Beam search is an improved version of greedy search. It has a hyperparameter named beam size **k**. At time step 1, we select **k** tokens with the highest conditional probabilities. Each of them will be the first token of **k** candidate output sequences, respectively. At each subsequent time step, based on the **k** candidate output sequences at the previous time step, we continue to select **k** candidate output sequences with the highest conditional probabilities from **k**|**Y**| possible choices.

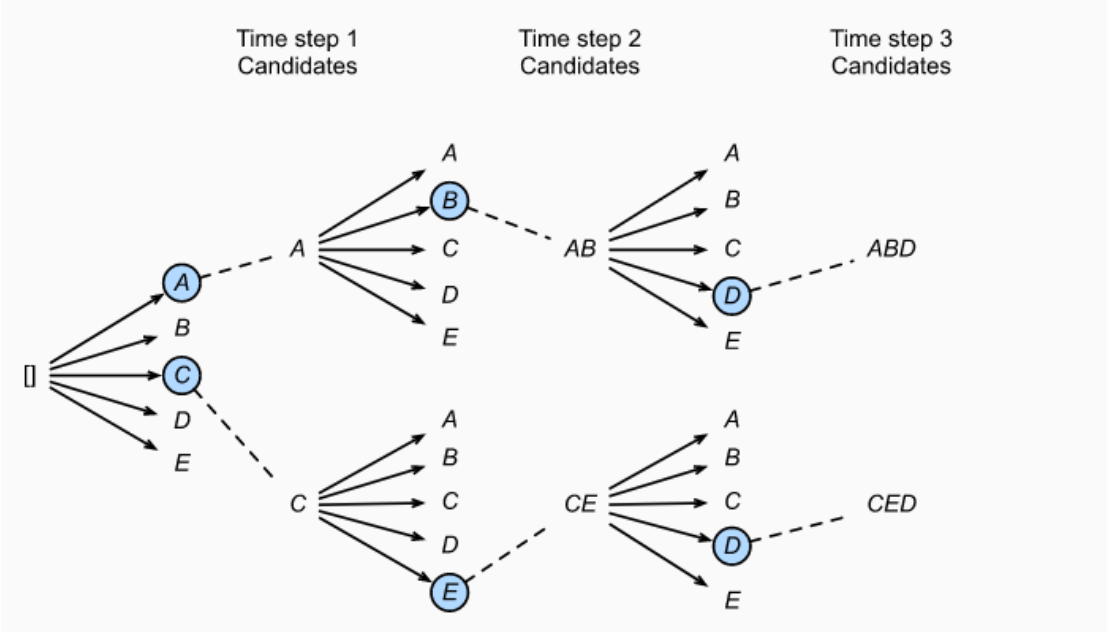

Figure 2 The process of beam search (beam size: 2, maximum length of an output sequence: 3). The candidate output sequences are A, C, AB, CE, ABD, and CED.

Figure 2 demonstrates the process of beam search with an example. Suppose that the output contains only five elements: $\mathcal{Y} = \{A, B, C, D, E\}$,Let the beam size be 2 and the maximum length of an output sequence be 3.

(If you still have some questions about beam search, you can visit https://zhuanlan.zhihu.com/p/82829880)

1. Implement beam search algorithm and provide test cases to verify the correctness.

2. Analysis the results of your beam search algorithm with different beam size **k** and talk about your ideas about the choice of brute force search, greedy algorithm and beam search algorithm

4 Deadline

23:59:59 12th December, 2020