Reading Quiz #6

ⓘ This is a preview of the published version of the quiz

Started: Feb 24 at 5:10pm

Quiz Instructions

To prepare for this quiz, please read sections 6.1-6.2 (inclusive) in the Kleinberg + Tardos textbook.

The goal of this quiz is to lightly assess a first quick reading of these resources to prepare for class. You should definitely return to this material for a more thorough read to solidify your learning and prepare for assignments and exams.

Note that you are limited to ***3 attempts*** for this quiz.

Best of luck! :-)

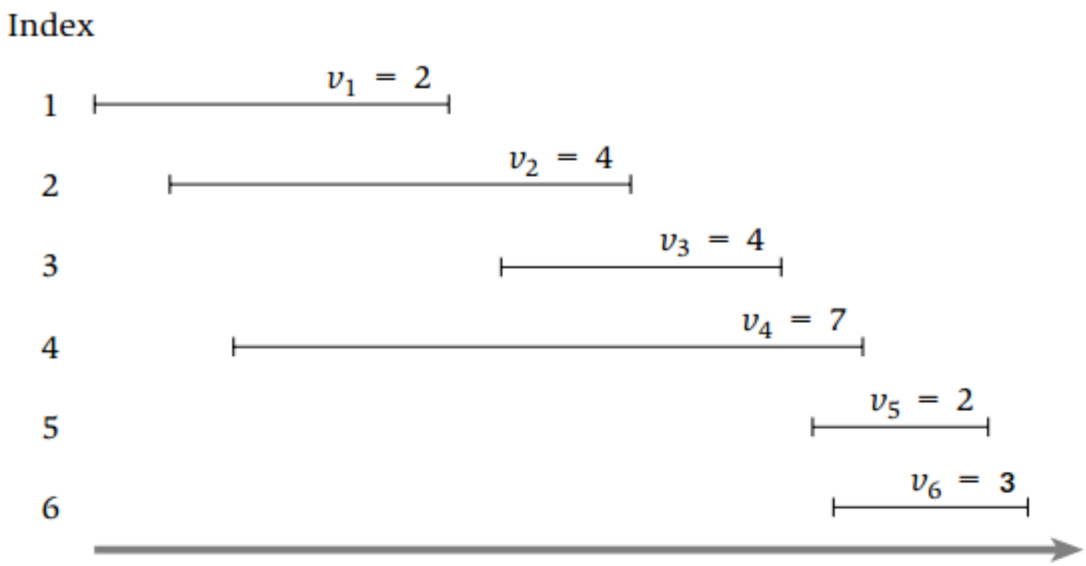
Question 1

1 pts

Consider the Weighted Interval Scheduling Problem (WISP) as defined in section 6.1 of the textbook. Consider two potential greedy algorithms that a CPSC 320 student might be tempted to apply to solve it:

- **GreedyEarliestFinish** is our name for exactly the algorithm you learned about in section 4.1 of the textbook; it chooses first the interval with the earliest finish time, discards all intervals that conflict with it, and then repeats for the interval with the next earliest finish time, etc.
- **GreedyHighestValue** is one of the natural greedy algorithms you might expect for the Weighted Interval Scheduling problem: first choose the interval with the greatest weight/value, discard all intervals conflicting with it, and then iterate on the remaining intervals.

Now, consider the following non-trivial, but still manageable, instance of WISP, taken directly from the same textbook section (with a single minor change, to make it a little more interesting):



For this question, please match each algorithm on the left to the value of the solution it would find for this instance. (For the "**Optimal**" choice, it doesn't matter which algorithm generated the solution; we're only interested in the greatest value possibly achievable on this particular instance.)

GreedyEarliestFinish	[Choose]
GreedyHighestValue	[Choose]
Optimal	[Choose]

Question 2

1 pts

Which of the following insights are used in determining the recurrence relation for the Weighted Interval Scheduling Problem (WISP)? Choose all that apply.

In the choices below, assume that n is the total number of intervals in an instance of WISP, and that the intervals have already been sorted in order of non-decreasing finish time, and j is the index of some arbitrary interval, with

$1 \leq j \leq n$.

p(j) is defined as in Section 6.1 of the textbook **with a typo correction** to the book's English description: it gives the index of the **rightmost** interval that ends before *j* begins.

- [ ] For any j between 1 and n, if we know the optimal scheduling for the first j-1 intervals, finding the optimal solution for the first j intervals amounts to making a binary choice.
- [ ] The optimal schedule that includes the nth interval adds its value to the optimal solution for the first p(n) intervals.
- [ ] All intervals in the optimal solution for the first j intervals must be included in the optimal solution for the first j+1 intervals
- [ ] If there are exactly two choices for whether the jth interval is included in the optimal solution, the optimal solution must include the lowest in value of these choices
- [ ] The original problem instance can be expressed as a subproblem of itself

Question 3 **1 pts**

Below is a table containing all pertinent information from the execution of the Dynamic Programming (DP) algorithm for WISP on a particular instance containing 6 intervals. The columns of the table are labelled according to the following legend:

- The **j** column lists the indices of each interval. The intervals have been numbered in order of non-decreasing finish time. (The zeroth index is a placeholder to make the recurrence relation nicer.)
- **v_j** gives the weight/value of the *j*th interval.
- **p(j)** is defined as in Section 6.1 of the textbook **with a typo correction** to the book's English description: it gives the index of the **rightmost** interval that ends before *j* begins.
- **M[j]** is the memo table populated by the DP algorithm; it contains the value of the optimal solution considering only the first *j* intervals.

j	v_j	p(j)	M[j]
0			0
1	2	0	2
2	1	0	2
3	1	2	3
4	2	0	3
5	1	3	4
6	3	2	5

Using this table, along with the recurrence relation for WISP given in Section 6.1 of the textbook (Equation 6.1), **reconstruct the solution found** by the DP algorithm, by selecting the index ("j") of EACH interval that gets scheduled in the optimal (highest-weight) scheduling.

- ☐ 1
- ☐ 2
- ☐ 3
- ☐ 4
- ☐ 5
- ☐ 6

Question 4

1 pts

Which of these differs between a *recursive, memoized* implementation of DP, vs a *bottom-up, iterative* implementation of DP?

(Elmer Fudd is not the right answer, but you can **[Google Seawch (https://www.google.ca/search?hl=xx-elmer)](https://www.google.ca/search?hl=xx-elmer)** for him.)

- ○ By the time we consider a particular problem, whether the solutions to its subproblems have already been computed.
- ○ Whether we compute the solution to a particular subproblem more than once.
- ○ Whether we ask for the solution to a particular subproblem more than once.
- ○ Whethuh Elmuh Fudd was taught it in grade school.

Not saved Submit Quiz