CONCORDIA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 6651: Algorithm Design Techniques

Fall 2022 - Quiz # 3

Question 1.

Consider the class scheduling problem described with two arrays $S[1 \cdots n]$ and $F[1 \cdots n]$, where $S[i] < F[i]$ for each i, representing the start and finish times of n classes. Your goal is to find the largest number of classes you can take without ever taking two classes simultaneously. We showed in class that the following greedy algorithm constructs an optimal schedule:

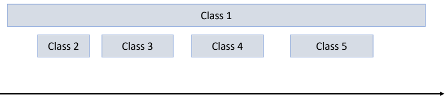
Choose the course that ends first, discard all conflicting classes, and recurse.

But this is not the only greedy strategy we could have tried. For each of the following alternative greedy algorithms, either prove that the algorithm always constructs an optimal schedule, or describe a small input example for which the algorithm does not produce an optimal schedule. Assume that all algorithms break ties arbitrarily (that is, in a manner that is completely out of your control). Exactly three of these greedy strategies actually work.

1. Choose the course x that ends last, discard classes that conflict with x, and recurse.
2. Choose the course x that starts first, discard all classes that conflict with x, and recurse.
3. Choose the course x that starts last, discard all classes that conflict with x, and recurse.
4. Choose the course x with shortest duration, discard all classes that conflict with x, and recurse.
5. Choose a course x that conflicts with the fewest other courses, discard all classes that conflict with x, and recurse.
6. If no classes conflict, choose them all. Otherwise, discard the course with longest duration and recurse.
7. If no classes conflict, choose them all. Otherwise, discard a course that conflicts with the most other courses and recurse.
8. Let x be the class with the earliest start time, and let y be the class with the second earliest start time.
   - If x and y are disjoint, choose x and recurse on everything but x.
   - If x completely contains y, discard x and recurse.
   - Otherwise, discard y and recurse.
9. If any course x completely contains another course, discard x and recurse. Otherwise, choose the course y that ends last, discard all classes that conflict with y, and recurse.

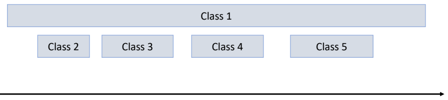
Solution

1. ☒ Choose the course x that ends last, discard classes that conflict with x, and recurse.



- Example:
- This does not work. Given the input, the greedy algorithm chooses the single long course, but the optimal schedule contains all the short courses.

2. ☒ Choose the course x that starts first, discard all classes that conflict with x, and recurse.

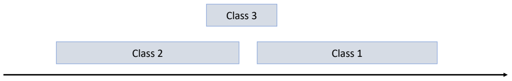


- Example:
- This does not work. Given the input, the greedy algorithm chooses the single long course, but the optimal schedule contains all the short courses.

3. ☑ Choose the course x that starts last, discard all classes that conflict with x, and recurse.

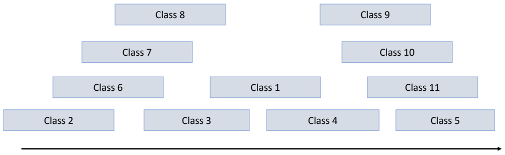
- This greedy strategy works! In fact, this is just a time-reversed version of the greedy algorithm proved correct in class.
- Correctness follows by induction from the following claim:
  - **Claim 1**. There is an optimal schedule S' that includes the course that starts last.
  - **Proof:** Let x be the course that starts last. Let S be any schedule that does not contain x, and let z be the last course in S. Because x starts last, we have $S[z] < S[x]$. Thus $F[i] < S[z] < S[x]$ for every other class in S, which implies that $S = Sz + x$ is still a valid schedule, containing the same number of classes as S. In particular, if S is an optimal schedule, then S' is an optimal schedule containing x. The correctness of this greedy strategy now follows by induction. ■

4. ☒ Choose the course x with shortest duration, discard all classes that conflict with x, and recurse.



- Example:
- This does not work. Given the input, this greedy algorithm chooses the single course in the middle, but the optimal schedule contains the other two courses.

5. ☒ Choose a course x that conflicts with the fewest other courses, discard all classes that conflict with x, and recurse.



- Example:
- This does not work. Given the input, this greedy algorithm would choose the course in the center, which has only two conflicts, and thus would return a schedule containing only three courses. But the optimal schedule contains four courses.

6. ☒ If no classes conflict, choose them all. Otherwise, discard the course with longest duration and recurse.

7. ☒ If no classes conflict, choose them all. Otherwise, discard a course that conflicts with the most other courses and recurse.

8. ☑ Let x be the class with the earliest start time, and let y be the class with the second earliest start time.

   - If x and y are disjoint, choose x and recurse on everything but x.
   - If x completely contains y, discard x and recurse.
   - Otherwise, discard y and recurse.

   - This greedy strategy works! We need to prove three claims, one for each case.
     - **Claim 2.** If x and y are disjoint, then every optimal schedule contains x.
     - **Proof:** If x and y are disjoint, then $F[x] < S[y]$, which implies that $F[x] < S[i]$ for all i = x. Thus, if S is any valid schedule that does not contain x, then $S + x$ is a larger valid schedule. Thus, no optimal schedule excludes x. □
     - **Claim 3**. If x contains y, there is an optimal schedule that excludes x.
     - **Proof:** If x contains y, then every class that conflicts with y also conflicts with x. Thus, for any valid schedule S that contains x , there is another valid schedule $Sx + y$ of the same size that excludes x. □
     - **Claim 4**. If x and y overlap, but x does not contain y, there is an optimal schedule that excludes y.
     - **Proof:** Suppose x and y overlap, but x does not contain y. Then x must end before y ends, and therefore every class that conflicts with x also conflicts with y. Thus, for any valid schedule S that contains y, there is another valid schedule $Sy + x$ of the same size that excludes y. □

     The correctness of this greedy strategy now follows by induction. ■

9. ☑ If any course x completely contains another course, discard x and recurse. Otherwise, choose the course y that ends last, discard all classes that conflict with y, and recurse.

- This strategy actually works!
  - **Claim 5.** If any course x contains another course y, there is an optimal schedule that does not include x.
  - **Proof:** Let S be any valid schedule that contains x. Then $Sx + y$ is another valid schedule of the same size. □
  - **Claim 6.** Suppose no course contains any other course. Then there is an optimal schedule containing the course that ends last.
  - **Proof:** If no course contains any other course, then the class that ends last is also the class that starts last. This claim now follows from Claim 1 (in question 3). □
- The correctness of this greedy strategy now follows by induction. ■

Question 2

Huffman tree is constructed for the following data: $\{A, B, C, D, E\}$ with frequency $.24, .11, .17, .33 and .15$ respectively. How do you decode 1000001101?

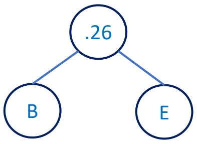
Solution

$$A \equiv .24, B \equiv .11, C \equiv .17, D \equiv .33 \text{ and } E \equiv .15.$$

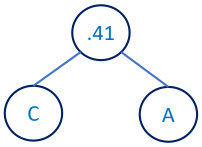
Hence, arranging them in increasing order

Characters	Frequency
B	.11
E	.15
C	.17
A	.24
D	.33

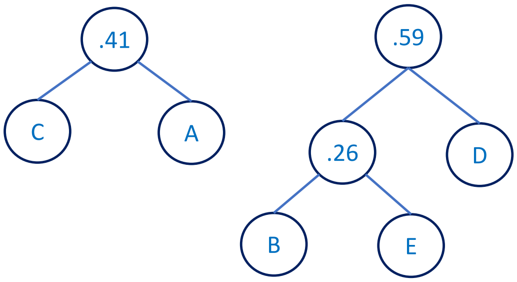
Constructing the tree from bottom up. We will add the ndoes generated in the non decreasing sequence generated above.



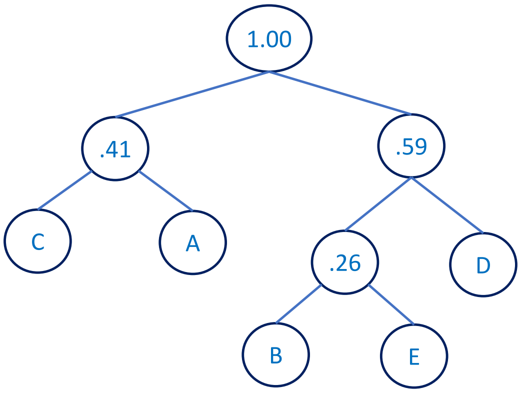
Characters	Frequency
~~B~~	~~.11~~
~~E~~	~~.15~~
C	.17
A	.24
Node 1	.26
D	.33



Characters	Frequency
~~B~~	~~.11~~
~~E~~	~~.15~~
~~C~~	~~.17~~
~~A~~	~~.24~~
Node 1	.26
D	.33
Node 2	.41

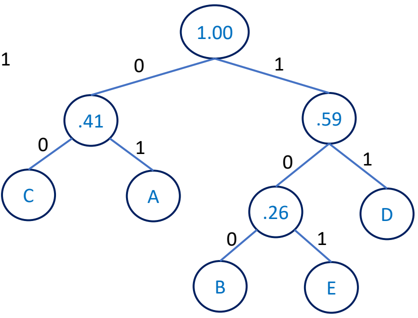


Characters	Frequency
~~B~~	~~.11~~
~~E~~	~~.15~~
~~C~~	~~.17~~
~~A~~	~~.24~~
~~Node 1~~	~~.26~~
D	.33
Node 2	.41
Node 3	.59



Characters	Frequency
~~B~~	~~.11~~
~~E~~	~~.15~~
~~C~~	~~.17~~
~~A~~	~~.24~~
~~Node 1~~	~~.26~~
D	.33
~~Node 2~~	~~.41~~
Node 3	.59

1000001101
= 100 00 01 101
= BCAE



Characters	Code
B	100
E	101
C	00
A	01
D	11