Dynamic Programming, Greedy Algorithms

University of Colorado Boulder

BY- HARSH GUPTA

WEEK 1:

Max Subarray Problem

1. Consider the array with numbers that is input to the max subarray problem

6 / 6 points

[1, 19, 5, -4, 7, 18, 15, -10]

Select all true facts from the list below making sure that no incorrect choices are selected.

- [x] The output .to the max subarray .problem should be 18 - (-4) = 22

Correct
Correct.

- [ ] The max subarray problem can be solved in linear time by simply taking the difference between the largest and smallest elements in the array.
- [x] The divide and conquer algorithm will compute the result of max subarray problem on the first half of the array, which in this instance yields the value 18

Correct
Correct: 19 - 1 = 18

- [x] The divide and conquer algorithm will compute the result of max subarray problem on the second half of the array, which in this instance yields the value 11

Correct

- [ ] For solving the max subarray problem, it is sufficient to recursively solve the problem for left and right halves of the given array and take the maximum among the two.
- [x] The minimum element of the first half of the array is -4 and maximum element of the second half of the array is 18. These in turn form the result for the max subarray problem which is 22.

Correct
Correct.

2. Consider the recurrence that represents the running time for the max subarray problem:

1 / 1 point

$$T(n) = \begin{cases} \Theta(1) & n \leq 2 \\ 2T(\frac{n}{2}) + \Theta(n) & \text{otherwise} \end{cases}$$

- [x] The case when $n \leq 2$ represents the constant amount of work needed to find the max subarray for input arrays of size 1 or 2.

Correct
Correct: as explained in the lecture.

- [x] The recurrence assumes that n is a power of two, since repeated division by 2 can yield fractional results otherwise.

Correct
Correct: we can always pad an array with dummy elements of $-\infty$ so that its size is a power of two and the result of max-subarray does not change. Doing so will not more than double the number of elements in the worst case.

- [x] The $\Theta(n)$ term in the recurrence for $n > 2$ represents the work to find minimum of first half and maximum of second half.

Correct
Correct

- [ ] The recurrence is identical to that obtained for binary search algorithm.
- [x] The recurrence and the running time are identical to that obtained for the mergesort algorithm covered earlier in course 1 of this specialization.

Correct
Correct

Karatsuba's Multiplication Algorithm

1. The following questions concern the binary representation of numbers and addition. Note that we will write binary numbers as $b_n, \ldots, b_0$ where b_n is the most significant bit whereas b_0 is the least significant bit. However, represented as a list in the computer, the same number would be $[b_0, b_1, \ldots, b_n]$.

4 / 4 points

- [x] The number 6 in decimal is represented by the list [0, 1, 1]

Correct
Correct $6_{10} = (110)_2$ is represented by the list [0, 1, 1]

- [x] Consider two lists [0, 1, 1] and [1, 1]. The sum of these two numbers is given by [1, 0, 0, 1]

Correct
Correct.

- [ ] The addition of a m bit number with a n bit number yields a number with as many as $m + n$ bits.
- [x] The addition of a m bit number with a n bit number yields a number with as many as $\max(m, n) + 1$ bits.

Correct
Correct. The result has one more bit than the larger of the two numbers.

- [x] The algorithm for adding two n bit numbers runs in time $\Theta(n)$.

Correct
Correct.

2. The following questions concern the grade school algorithm we studied in the lecture. Note that we will represent numbers as lists of bits. Select all the correct answers from the list below.

4 / 4 points

- [x] The grade school multiplication algorithms performs as many additions as the number of 1 bits in the second argument.

Correct
Correct since zero bits in the second argument do not contribute to the problem.

- [x] Consider the multiplication of two numbers represented by list a = [1, 0, 1] with b = [1, 1]. The grade school multiplication algorithm performs additions of the number [1, 0, 1,0] with the number [0, 1, 0, 1], yielding the result [1, 1, 1, 1]. Note the number $a_n \ldots a_0$ is represented as a list $[a_0, a_1, \ldots, a_n]$.

Correct
Correct.

- [ ] The shift operation performed at each step of the multiplication algorithm appends a 0 to the end of the list.
- [x] The shift operation performed at each step of the multiplication algorithm appends a 0 to the beginning of the list.

Correct
Correct.

3. The following question concerns the multiplication of two n bit numbers a and b represented by the lists using the divide and conquer scheme presented in the lecture.

1 / 1 point

$[a_0, \ldots., a_{n-1}]$ and $[b_0, \ldots, b_{n-1}]$.

For any list of bits l, the number denoted by the list is denoted by $[[l]]$.

Assume that n is a power of 2 and let $m = \frac{n}{2}$.

- [ ] The number denoted by the list $[a_0, \ldots, a_{n-1}]$ can be written as $[[\ [a_0, \ldots, a_{m-1}]\ ]]$ + $[[\ [a_m, \ldots, a_{n-1}]\ ]]$
- [x] The number denoted by the list $[a_0, \ldots, a_{n-1}]$ can be written as $[[\ [a_0, \ldots, a_{m-1}]\ ]]$ + $2^m[[\ [a_m, \ldots, a_{n-1}]\ ]]$

Correct

- [x] Karatsuba multiplication of two n bit numbers makes three recursive calls with two of the calls involving multiplication of m bit numbers and one call involving multiplication of at most $m+1$ bit number.

Correct
Correct.

- [ ] The number of bitwise additions required by Karatsuba's algorithm is identical to that required by the grade school multiplication.
- [ ] Karatsuba's algorithm will multiply two n bit numbers faster than the grade school algorithm for all n.
- [ ] Depending on the implementation details and the computer on which we run, there is a cutoff value of n such that for all inputs with greater than n bits, Karatsuba's algorithm will outperform grade school multiplication.
- [x] Depending on the implementation details and the computer on which we run, there is a cutoff value of n such that for all inputs with greater than n bits, **the worst case running time** of Karatsuba's algorithm will outperform **the worst case running time** of grade school multiplication.

Correct
It is important to note that the claims we make are about worst case complexity.

Master Method

1. Consider a recurrence of the form :

4 / 4 points

$$T(n) = \begin{cases} \Theta(1) & n \leq 1 \\ 3T(n/3) + \Theta(n) & \text{otherwise} \end{cases}$$

Select all the correct options from the list below.

- [x] The recurrence above can be obtained from a divide and conquer scheme that divides inputs of size n into 3 subparts of size n/3 each.

Correct
Correct.

- [ ] The overall complexity of divide + combine steps in the algorithm is $\Theta(1)$.
- [ ] Master method is applied with $a = b = 3$ and 1. Case-1 applies and the overall complexity is $T(n) = \Theta(n^{\log_3(3)}) = \Theta(n)$
- [x] Master method is applied with $a = b = 3$ and 1. Case-2 applies and the overall complexity is $T(n) = \Theta(n^{\log_3(3)} \log(n)) = \Theta(n \log(n))$

Correct

2. Consider a recurrence of the form :

4 / 4 points

$$T(n) = \begin{cases} \Theta(1) & n \leq 1 \\ 2T(n/3) + \Theta(\sqrt{n}) & \text{otherwise} \end{cases}$$

Select all the correct options from the list below.

- [ ] The recurrence could be produced by a divide and conquer algorithm that divides input of size n into three parts of size $n/2$ each.
- [x] The recurrence denotes a divide and conquer scheme wherein the divide and combine steps take $\Theta(\sqrt{n})$ time in total.

Correct
Correct.

- [x] Master method is applied with $a = 2$ and $b = 3$. We have $\log_b(a) = \log_3(2)$ which is a number between 0 and 1 but is greater than half.

Correct
Correct. Note that $\log_3(1.732..) = 0.5$.

- [x] Case-1 of the master method applies and the complexity of the algorithm is $\Theta(n^{\log_3(2)})$.

Correct
Correct.

Complex Numbers and Roots of Unity

1. Select all the correct facts about complex numbers from the list below. 4 / 4 points

- [x] The complex number $3+4j$ has modulus 5.

Correct
$\sqrt{3^2.+4^2}=5$

- [x] The conjugate of the complex number $re^{j\theta}$ is given by $re^{-j\theta}$.

Correct
Indeed: $r\cos(\theta)-jr\sin(\theta)=re^{-j\theta}$ which is the conjugate of $re^{j\theta}$.

- [ ] The conjugate of the complex number $re^{j\theta}$ is given by $\frac{1}{r}e^{j\theta}$.
- [ ] The value of the expression $\exp(j\frac{\pi}{4})$ is j.
- [x] For any complex number z, the numbers $z+\overline{z}$ and $z\times\overline{z}$ are both real numbers.

Correct
Correct: Let $z=x+jy$. We have $z+\overline{z}=2x$ and $z\times\overline{z}=x^2+y^2$.

- [ ] The phase (angle) of a complex number $1+j$ in radians is $\frac{\pi}{2}$.
- [x] The complex number j is one of the fourth roots of unity.

Correct
Correct. $j^4=-1^2=1$

- [x] The value of $\frac{1}{1+j}$ is $\frac{1}{2}(1-j)$.

Correct
Correct.

2. Let w_n denote the generator of the n^{th} roots of unity for $n\geq 1$. Select all the correct options from the list below. 4 / 4 points

- [x] $w_n=\cos(\frac{2\pi}{n})+j\sin(\frac{2\pi}{n})$

Correct
Correct

- [ ] The set of n^{th} roots of unity is $\{w_n, w_n^2, \cdots, w_n^{n-1}\}$.
- [x] $w_n^{n-1}=\overline{w_n}=\frac{1}{w_n}$.

Correct
Correct. Because $w_n^{n-1}=w_n^n\times w_n^{-1}=\frac{1}{w_n}=\overline{w_n}$

- [ ] If n is even and $n\geq 2$ then $w_n^{n/2}=1$
- [x] If n is even and $n\geq 2$ then $w_n^2=w_{n/2}$.

Correct
Correct: $w_n^2=e^{j\frac{2\pi}{n}\times 2}=e^{j\frac{2\pi}{n/2}}$

- [x] For any $0\leq k<n$ we have $w_n^k=\overline{w_n}^{n-k}=\frac{1}{\overline{w_n}^k}$.

Correct
Correct

- [x] $1+w_n+w_n^2+\cdots+w_n^{n-1}=0$ for all $n\geq 2$

Correct
$1+w_n+w_n^2+\cdots+w_n^{n-1}=\frac{w_n^n-1}{w_n-1}=0$ since $w_n^n=1$.

- [ ] $w_n^n=0$

FFT Algorithm and Applications

1. Consider a sequence $[a_0, a_1, a_2, a_3]$. Let $[A_0, A_1, A_2, A_3]$ be the discrete fourier transform of this sequence. Select all the correct options below.

4 / 4 points

- [x] The 4th roots of unity are $\{1, j, -1, -j\}$.

Correct
Correct.

- [x] $A_0 = a_0 + a_1 + a_2 + a_3$.

Correct
Correct

- [x] The DFT can be viewed as evaluating the polynomial $a(x) = a_0 + a_1x + a_2x^2 + a_3x^3$ for $x = 1, j, -1$ and $-j$ respectively.

Correct
Correct.

- [ ] $A_1 = a(j) = a_0 + a_1j + a_2 - a_3j$
- [x] $A_2 = a(-1) = a_0 - a_1 + a_2 - a_3$

Correct
Correct.

- [ ] $A_3 = a(-j) = a_0 - a_1j + a_2 - a_3j$
- [x] A_1 and A_3 must be complex conjugates of each other as long as the sequence a_0, a_1, a_2, a_3 consist of real numbers.

Correct
Correct.

- [x] A_0 and A_2 must always be real numbers as long as the sequence a_0, a_1, a_2, a_3 consists of real numbers.

Correct
Correct.

2. Let $a_0, \ldots, a_{511}$ be a sequence of real numbers obtained from sampling wind velocity with 8 samples per minute over 64 minutes (roughly 1 hour).

6 / 6 points

Suppose we compute the DFT and obtain the sequence $[A_0, \ldots, A_{n-1}]$ as the DFT coefficients.

- [x] $A_0 = \sum_{j=0}^{511} a_j$.

Correct
Correct.

- [x] $A_{256} = a_0 - a_1 + a_2 - a_3 + \cdots + a_{510} - a_{511}$.

Correct
Correct. Note that $\omega^{256} = -1$ where ω is the root $\exp \frac{2\pi j}{512}$. Thus plugging in $x = -1$ in the polynomial $a_0 + a_1 x + a_2 x^2 + \cdots + a_{511} x^{511}$.

- [ ] A_{12} and A_{499} are always complex conjugates for all real values $a_0, \ldots, a_{511}$.
- [x] A_{128} corresponds to the frequency: $8 \times \frac{128}{512} =$ per minute = 2/minute.

Correct
Correct.

- [ ] The highest frequency component is A_{511}.
- [x] The highest frequency component is A_{256} which corresponds to a frequency of 4/minute.

Correct
Correct: The frequency would be $8 \times 256/512$

- [ ] The component A_{511} is always the complex conjugate of A_1 and corresponds to a frequency of -8/minute
- [x] The reason we assign negative frequencies to components A_j for $j > \frac{n}{2}$ is because they correspond to roots of unity w_n^j which can be seen as "rotating" in clockwise direction (opposite direction to roots w_n^j for $j \leq n/2$).

Correct
Yes that is why!!

WEEK 2:

Rod Cutting Problem and Recurrence

1. Consider the rod cutting problem with the following price table:

1 / 1 point

Length	Price
2	1.9
3	2.2
5	4.2

Assume that "wasting" earns no revenue.

We have a rod of length L = 11. Choose all the correct facts below.

- [ ] The solution where we cut the rod into 5 portions of size 2 while wasting length 1 yields the highest revenue
- [ ] The optimal solution involves one cut of length 5 and 2 cuts of length 3 with no waste
- [ ] The optimal solution involves four cuts of length 2 and one cut of length 3 yielding no waste.
- [x] The optimal solution involves 1 cut of length 5 and 3 cuts of length 2.

Correct
This yields 5.7 + 4.2 = 9.9 revenue

2. Consider the rod cutting problem with the following price table:

1 / 1 point

Length	Price
2	1.9
3	2.2
5	4.2

Assume that "wasting" earns a penalty that is $-1 \times \text{waste length}$

Let $maxRev(L)$ denote the maximum revenue obtainable from a rod of length L with the price table fixed above. Fill in the missing portions of the recurrence below:

$$maxRev(L) = \begin{cases} ?_1 & L < 0 \\ ?_2 & L = 0 \\ \max\left(?_3, maxRev(L-2) + ?_4, maxRev(L-3) + ?_5, maxRev(L-5) + ?_6\right) & \text{otherwise} \end{cases}$$

- [ ] $?_1$ should be 0
- [x] $?_2$ should be 0

Correct
Correct: it represents the fact that we have cut the rod so that there is no waste.

- [ ] $?_3$ should be 0
- [x] $?_4$ should be 1.9

Correct
Correct: it is the revenue for a cut of length 2

- [ ] $?_5$ should be 1.9
- [x] $?_6$ should equal the revenue of a cut of length 5.

Correct
Correct.

Memoization

1. Consider the rod cutting problem with the following price table:

1 / 1 point

Length	Price
2	1.9
3	2.2
5	4.2

Assume that "wasting" earns no revenue.

We have a rod of length L = 9.

Using the recurrence, we will fill out the memo table T below for the rod cutting problem for length L = 9

0	1	2	3	4	5	6	7	8	9
0									

Choose all the correct facts below.

- [x] T[1] = 0

Correct
Correct

- [x] T[2] = 1.9

Correct
Correct

- [x] $T[3] = \max(0, T[0] + 2.2, T[1] + 1.9, T[-2] + 4.2)$

Correct
Correct

- [ ] T[4] = 2.2
- [ ] T[5] = 4.1
- [x] $T[j] = \max(0, T[j-2] + 1.9, T[j-3] + 2.2, T[j-5] + 4.2)$With $T[j] = -\infty$ for $j < 0$

Correct
Correct

Coinchanging Problem

1. Consider we wish to compute change for 48 cents given coins of denomination

4 / 4 points

{ 2, 5, 10, 20} cents. We design two tables

Tbl[j] that records the best solution in terms of number of coins for j cents.

S[j] that records the "first" coin denomination that we need to use for obtaining the solution for Tbl[j]

Select all the correct facts from the list below.

- [ ] Tbl[j] = $-\infty$ for $j < 0$
- [x] $Tbl[j] = \min(1 + Tbl[j-2], 1 + Tbl[j-5], 1 + Tbl[j-10], 1 + Tbl[j-20])$

Correct
Correct

- [x] $Tbl[3] = \infty$ denoting that we cannot make change for 3 cents using the given denominations.

Correct

- [x] Suppose we wish to recover the solution, let $S[48]$ have the value 20 after we finish implementing the memoization. The solution recovery will add a 20 cent coin to our solution list and look up $S[28]$

Correct
Correct

- [ ] Following the previous option, suppose we look up $S[28]$ and encounter $S[28] = 2$, we will look up $S[20]$ next.

2. Consider the usual solution that people use to make change for target T.

3 / 3 points

- Take the largest denomination coin that is $\leq$ T let it be c_j.

- Give c_j and recursively make change for $T - c_j$.

- Stop when the remaining change is 0.

Consider the denominations $\{1, 2, 5, 10, 20, 25\}$ cents.

- [x] Suppose we wish to make change for 50 cents, our approach will provide two 25 cent coins as change, which is optimal.

Correct
Correct

- [ ] Suppose we wish to make change for 40 cents, our approach will use two 20 c coins, which is optimal.
- [ ] The algorithm presented above always produces the optimal solution for any coin denominations and target.
- [x] The algorithm makes optimal decision for T = 49, using four coins: one 25 cent, one 20 cent and two 2 cent coins.

Correct
Correct

Knapsack Problem

1. Throughout this quiz consider a knapsack problem with three items

1 / 1 point

Item	Value	Weight
I_1	v_1	w_1
I_2	v_2	w_2
I_3	v_3	w_3

Once again we wish to maximize the total value of our steal while keeping weights under limit W. However, for each item we can steal arbitrarily many copies of that item. For instance, if we steal item I_2 5 times, we have a value of $5v_2$ and weight $5w_2$. There is no limit on the number of times an item can be stolen.

Assume $w_j > 0$ for each item: otherwise, we can take infinitely many copies of the items and the problem becomes undefined.

- (●) This sort of situation can happen if I_j is a stock where we can invest in 0 or more units of the stock I_j.
- ( ) This sort of situation is purely imaginary and not based on any sort of reality.

Correct
Correct

2. Refer to the problem introduced in the previous question.

1 / 1 point

Let $maxValue(j, W)$ be the maximum value obtained for considering items $I_j, \ldots, I_3$ and weight limit W. Note that $1 \leq j \leq 4$. In particular for $j = 4$, we obtain the empty list of items.

Select all the correct facts from the choices below.

Notation $\lfloor \frac{a}{b} \rfloor$ is the value by computing $\frac{a}{b}$ and rounding it down when $a, b > 0$.

- [x] The minimum number of times we can choose item I_j is 0 and maximum number of times is $\lfloor \frac{W}{w_j} \rfloor$.

Correct
Correct.

- [x] $maxValue(4, W) = 0$ whenever $W \geq 0$.

Correct
Correct.

- [x] If the thief chose to steal item I_j, $k \geq 0$ times, the remaining weight budget is $W - kw_j$ and value obtained is kv_j

Correct

- [ ] If $j < 4$ and $W \geq 0$ then

$$(\; 0 + maxValue(j + 1, W)$$

Longest Common Subsequence

1. Consider two strings: **4 / 4 points**

`s1 = "ATTCCGGAC" and s2 = "TTACGG"`

We wish to find the longest common substring between these two strings.

- [ ] The longest common substring is of length 4: `"TTGG"`
- [ ] The longest common substring is of length 5: `"ATCGG"`
- [x] The longest common substring is of length 5: `"TTCGG"`

Correct
Correct. The only longer possibility is `"TTACG"` which is not a substring of s1

- [x] Suppose we have committed to matching the second character `"T"` in s1 to the first character s2, the remaining decision is to optimally find the LCS for the suffix: `TCCGGAC` and `TACGG`

Correct
Correct

2. Consider strings s1: `ATCCG` and s2: `CACGC` **7 / 7 points**

We construct a memo table which for your convenience is labeled with the characters in strings s1 and s2

	C	A	C	G	C	
A						0
T						0
C			??7	??5	??6	0
C				??4	??3	0
G				??2	??1	0
	0	0	0	0	0	0

Write down the values that will be filled in for the positions labeled with ??1 - ??7 in the memo table above and use those to select the correct answers below.

- [x] ??1 = 0

Correct

- [x] ??2 = 1

Correct

- [ ] ??3 = max(0, 0) = 0
- [x] ??3 = 1

Correct
Correct

- [x] ??4 =max(??3, ??2)

Correct
Correct

- [ ] ??4 = ??1 + 1
- [ ] ??6 = ??3 + 1
- [x] ??7 = ??4 + 1

Correct
Correct: the characters corr. to the cell labeled ??7 are the same.

- [x] ??7 = 2

WEEK 3:

Greedy Algorithms

1. Consider once again the coin changing problem with denominations: {1, 2, 5, 10} cents. **3 / 3 points**

- [x] The greedy coin changing algorithm when applied to make change for 18 cents will utilize 4 coins.

Correct
Correct: it will utilize all the coins provided

- [ ] The greedy coin changing algorithm will not be able to make exact change for some amounts (assume that we are making change for 1 cent or higher).
- [x] If we added a 8 cent coin to our set of denominations, the greedy algorithm will not be an optimal algorithm for making change with the least number of coins.

Correct
Correct: think of making change for 24 cents. Optimally, we can do it with 3 x 8 cents. The greedy approach would use 2 x 10 cents, 2 x 2 cents = 4 coins.

2. Consider a version of knapsack problem for a perfumer who has an empty perfume bottle with volume V = 10 ml and would like to mix ingredients in various proportions to maximize the total profit which is simply calculated by summing up the profit/unit volume of each ingredient times the volume of the ingredient used. **1 / 1 point**

Liquid	Profit/Unit Volume	Available Amount
A	1.5 $/ml	10 ml
B	2.2 $/ml	4 ml
C	3.3 $/ml	3 ml
D	2.1$/ml	5 ml
E	1.7 $/ml	2 ml

- [ ] The optimal solution is to fill the bottle with liquid A earning a total profit of 15 dollars.
- [x] The optimal solution would be to fill up 3 units of C, 4 units of B, and 3 units of D yielding a total profit of 3 x 3.3 + 4 x 2.2 + 3 x 2.1 = 25 dollars.

Correct
Correct

- [x] For solving problems like this, a greedy strategy would be to first take as many units of ingredient C as possible since that has the most profit per unit volume.

Correct
Correct

- [ ] It is possible to obtain an optimal solution that uses strictly less than 3 ml of ingredient C.

Greedy Interval Scheduling

1. In this lecture, we saw how greedy algorithms were optimal for interval scheduling when our goal is to maximize the number of meetings held. Suppose we have the same problem setup of meetings with start/end times specified and our goal was to improve the overall room utilization : i.e, the amount of time the room is occupied in a meeting rather than the number of meetings.

5 / 5 points

Answer the following questions below by selecting the correct facts.

- [x] Suppose we design a greedy algorithm that first schedules the longest meeting, deletes all the conflicting meetings and recursively solves the remaining sub problem, this algorithm is not optimal in terms of total occupancy.

Correct
Correct: as a counter example, let us take three meetings [6AM - 12 noon], [12 noon - 6 PM] and [8:59 AM - 3:01 PM]. Scheduling the third meeting first will remove the first two meetings from consideration. However the optimal solution was to schedules the first two meetings.

- [x] The greedy algorithm used to maximize the number of meetings held will not necessarily maximize the total time the room is occupied by a meeting.

Correct
Correct: a simple example is to have one meeting that spans the entire day and a bunch of smaller meetings. Maximizing the number of meetings in this kind of a situation will not maximize occupancy.

- [x] Suppose all meetings are of the same duration then maximizing the number of meetings is equivalent to maximizing the total time utilized by the meetings.

Correct
Correct since, the total time will be an integer multiple of the common meeting duration.

Huffman Codes

1.

3 / 3 points

Consider a large document with 5 letters {A, B, C, D, E} and the following percentages of occurrence for each of the five letters:

letter	fraction of occurrence
A	35%
B	25%
C	20%
D	15%
E	5%

Consider how many bits each character gets allocated by a Huffman code:

letter	# bits
A	b_A
B	b_B
C	b_C
D	b_D
E	b_E

Select all the correct answers from the list below about the Huffman code generated for this example.

- [ ] b_A = 1 bit since A is the most frequent character.
- [x] The construction of Huffman code will first merge D and E into a subtree.

Correct
Correct

- [x] $b_A = b_B = 2$

Correct
Correct

- [x] $b_C = 2$

Correct

- [x] The average number of bits per character for the Huffman code is 2.2 bits/character

- [x] The average number of bits per character for the Huffman code is 2.2 bits/character

Correct
Correct: 2 * 0.35 + 2 * 0.25 + 2 * 0.2 + 3 * 0.15 + 3 * 0.05 = 2.2

- [ ] D and E are assigned 4 bits each in the prefix code.

2. Select all the correct facts about the behavior of the Huffman coding algorithm given a set of characters $A_1, \ldots, A_n$ and their frequencies $f_1, \ldots, f_n$.

2 / 2 points

- [x] The character with lowest frequency will always have the highest number of bits assigned.

Correct
Correct: this is needed for optimality since if this were not the case, we can always swap the code for the lowest freq. character with the character that got the highest number of bits and get a code that achieves better # bits/char.

- [ ] The highest frequency character will always be assigned 1 bit in the Huffman code.
- [x] The character with second lowest frequency will also have the highest number of bits assigned.

Correct
Correct.

- [x] Suppose $n = 32$ and we assign 5 bits to each character. A Huffman code will always assign 5 or fewer bits per character, on average.

Correct
Correct: since assigning 5 bits per character is also a prefix-code.

WEEK 4:

Decision Problems and Languages

1. Select all the correct facts from the list be low.

10 / 10 points

- [ ] Suppose we are given a graph G and asked to return a cycle involving two or more vertices OR return `None` if there are no such cycles. There is no decision version of this problem.
- [x] Consider the language L = { 0, 10, 100, 110, ... } of binary encodings of all even numbers. An algorithm that recognizes L is also an algorithm that given a number returns true if even and false if odd.

Correct
Correct

- [x] Consider the problem of finding if a graph G is strongly connected (i.e, entire graph is a single SCC). The corresponding language is $L = \{< G > \mid \text{G is a graph that is strongly connected}\}$

Correct
Correct

- [x] It is possible to encode graphs as binary strings of 0s and 1s such that every graph G corresponds to a unique binary string.

Correct
Correct: arguably that is what we do when we represent a graph data structure in the memory of a computer

- [ ] The problem of given a number n, checking whether or not it is prime is undecidable.

Polynomial Time and Certificates

1. Select all the problems for which we know of an efficient polynomial time algorithm through techniques studied thus far in this specialization.

4 / 4 points

- [x] Given a weighted graph G, does there exist a negative weight cycle?

Correct
We studied the Bellman Ford algorithm that can solve this problem.

- [x] Given a graph is there a spanning tree whose weight is less than K?

Correct
We can use the MST algorithm to compute the minimum weight spanning tree and using the weight of this spanning tree, we can easily answer the question in polynomial time

- [x] Given an array `a` of size `n` and a number `k`, are there more than `n/4` elements which are `>=` k?

Correct
This is polynomial time solvable: simply use k as a pivot and partition the array. You can answer the question based on the sizes of the two partitions thus created.

- [ ] Given a graph G and two vertices s, t, is the longest simple path (a path that does not visit any vertex more than once) of length more than k?

---

2. Select all the correct answers regarding certificates from the list below.

1 / 1 point

- [x] Given a number n that is known to be a product of two large prime numbers (such numbers are used extensively in cryptography), we wish to find out the kth bit of its smallest prime factor. The certificate is the prime factor p itself.

Correct
Correct: we can check that the certificate is a prime factor: we can check that it is a factor of n, we can check that $n/p \geq p$ and finally, we can extract the kth bit of p, to check the answer.

- [x] Let G be a weighted directed graph and s,t be two vertices of G. We wish to know if there a path from s to t of length $\leq W$. The empty string can be a certificate for this problem that can be checked in polynomial time.

Correct
Correct: simply run Dijkstra's algorithm and it will give us the shortest path weight from s to t. We do not need a certificate since Dijkstra's algorithm can run in polynomial time.

- [x] Let G be an undirected graph. We wish to know if there is a cycle that visits all vertices of G exactly once. The certificate for a yes answer is given by the cycle itself

Correct
Correct: the cycle can be verified in polynomial time in the size of the graph: check that it is a cycle, check that it visits each vertex exactly once.

- [ ] We are given an instance of the knapsack problem weights $W_1, \ldots, W_n$ and values $V_1, \ldots, V_n$. We wish to know if we can select items with total weight $\leq W$ and value $\geq v$. Suppose the algorithm comes back with the answer "no". The certificate for this answer is just a selection of items whose weight is $\leq W$ and value $< v$

NP Completeness Reductions

1. Suppose problem A reduces in polynomial time to problem B. Let us suppose that problem B cannot be solved in polynomial time. Does it mean that A cannot also be solved in polynomial time?

1 / 1 point

- ○ Yes, because we have reduced any input of problem A to an input of problem B, it follows that if B cannot be solved in polynomial time, then A cannot be solved either.
- ◉ No, it is possible that there is some other "more ingenious" way of solving A in polynomial time that does not involve a reduction to B.

Correct

2. True of False: Every problem in P can be reduced in polynomial time to the k-clique problem of checking if a given graph G has a clique of size at least k.

1 / 1 point

- ◉ True: every problem P is trivially in NP and by Cook-Levin theorem all problems in NP can be reduced to 3-SAT and we saw in the lecture that 3-SAT was reduced in polynomial time to the k-clique problem.
- ○ False: problems in NP can be reduced to k-clique but problems in P can already be solved in polynomial time.

Correct
Exactly what the question itself says.

3. Suppose we have a polynomial time reduction from problem A to problem B, select all the true facts from the list below.

1 / 1 point

- [ ] If A can be solved in polynomial time then B can be solved in polynomial time as well.
- [x] If there is no algorithm that can solve A in polynomial time, then there is no algorithm that can solve B in polynomial time.

Correct
Correct: for the sake of contradiction if there were an algorithm to solve B in polynomial time, the reduction provides an algorithm for A in polynomial time.

- [x] If A is NP complete and B is in NP, then B is also NP complete.

Correct
Correct

- [ ] If B is NP complete and A is in NP then A is NP complete.

4. Suppose we wish to prove the travelling salesperson problem (TSP) discussed in the lecture is NP complete. Which of the following steps will be needed to do so?

1 / 1 point

- [x] We will need to show that TSP is in NP by showing that the yes answer comes with a certificate that can be checked in polynomial time.

Correct
Correct

- [x] We need to reduce 3-SAT or a previously known NP complete problem to the TSP in polynomial time.

Correct

- [ ] We need to reduce TSP to a known NP complete problem in polynomial time.
- [x] A reduction from 3-SAT to TSP, will take a boolean 3-CNF-SAT formula and create a weighted graph G that is an instance of TSP.

Correct
Correct

- [ ] To prove TSP to be NP Complete , the reduction from 3-SAT will take a weighted graph G that is an instance of TSP and produce a boolean formula that is an instance of the 3-SAT problem.

NP Completeness Problems

1. Consider a 3-CNF-SAT problem with n variables denoted $x_1, \ldots, x_m$ and m clauses. We wish to reduce it to a 0-1 ILP problem:

3 / 3 points

Find $z_1 \in \{0,1\}, \ldots, z_n \in \{0,1\}$ such that a set of m linear inequality constraints $c_1 z_1 + c_2 z_2 + \cdots + c_n z_n \geq c_0$ are all satisfied.

Select all the true facts about the reduction.

- [x] We use a 0-1 variable z_i corresponding to each variable x_i in the original 3-CNF-SAT problem.

Correct
This is quite natural since we can always map false to the value 0 and true to the value 1.

- [x] A clause of the form $x_i \vee x_j \vee x_k$ translates into an inequality $z_i + z_j + z_k \geq 1$

Correct
Correct

- [ ] The logical negation of a variable x_i can be modeled as the negation $-z_i$
- [x] The logical negation of a variable x_i can be modeled as the arithmetic operation $1 - z_i$

Correct
Correct

- [x] The clause $\overline{x_i} \vee x_j \vee \overline{x_k}$ is translated to the inequality $-z_i + z_j - z_k \geq -1$

Correct
Correct: $(1 - z_i) + z_j. + (1 - z_k)$ is equivalent to $2 - z_i + z_j - z_k$ which in turn gives us the inequallity shown above.

- [x] The reduction yields as many inequalities as the number of clauses in the 3-SAT formula

Correct
Correct

2. An independent set in a graph is a subset of vertices such that no two vertices in the independent set have an edge between them.

3 / 3 points

k-Indpendent-Set Problem

Given a graph G and a number k, we wish to know if there is an independent set of size at least k in G.

- [x] The k Independent-Set problem is in NP since the certificate can involve just the set of k vertices that we claim to belong to an independent set.

Correct
Correct: we can verify the certificate by checking that there are no edges between any two vertices in our claimed independent set.

- [ ] To show that k-independent-set is NP complete we can reduce from the problem to k-clique problem which is already shown to be NP complete
- [x] A graph G has an independent set of size k if and only if its complement has a clique of size k.

Correct
Correct: two vertices have an edge in completement iff they do not have an edge in the original graph.

- [x] We can reduce the problem of finding k-clique in a graph G to that of finding a k-independent-set in its complement $\overline{G}$.

Correct
Correct