Problem Set 9

Due date: Electronic submission of this homework is due on **11/12/2021** on canvas.

Name: [redacted]

Resources. (All people, books, articles, web pages, etc. that have been consulted when producing your answers to this homework)

On my honor, as an Aggie, I have neither given nor received any unauthorized aid on any portion of the academic work included in this assignment. Furthermore, I have disclosed all resources (people, books, web sites, etc.) that have been used to prepare this homework.

Signature: [redacted]

Read Chapter 34 in our textbook.

Problem 1. (30 points) Let DSAT denote the problem to decide whether a Boolean formula has at least two satisfying assignments. Show that

(a) DSAT is in NP

(b) 3SAT $\leq_p$ DSAT

Conclude that DSAT is NP complete. [Hint: Introduce a new variable]

Solution. a) For this problem, we must show that DSAT is in NP. Considering a boolean input formula $\phi(x_1, x_2, ...x_n)$, we can nondeterministically set two assignments and check if they satisfy ϕ. To show this we can first introduce a new variable, y, and second a output formula $\phi(x_1, x_2, ...x_n, y) = \phi(x_1, x_2, ...x_n) \wedge (y \vee \overline{y})$. So, if $\phi(x_1, x_2, ...x_n)$ is in SAT, then in ϕ there is at least one assignment satisfied. However, to satisfy two assignments, we need to check the new claim, where $\phi(x_1, x_2, ...x_n, y)$ can satisfy at least two assignments by satisfying $(y \vee \overline{y})$ where y can equal either, 0 or 1, to a new y variable, so we can say that $\phi(x_1, x_2, ...x_n, y) \in$ DSAT. However, if $\phi(x_1, x_2, ...x_n) \notin$ SAT, then that claim $\phi(x_1, x_2, ...x_n, y) = \phi(x_1, x_2, ...x_n) \wedge (y \vee \overline{y})$ won't have any satisfying assignments , so $\phi(x_1, x_2, ...x_n, y) \notin$ DSAT. Furthermore, we can conclude that SAT $\leq_p$ DSAT, thus DSAT is NP-complete.

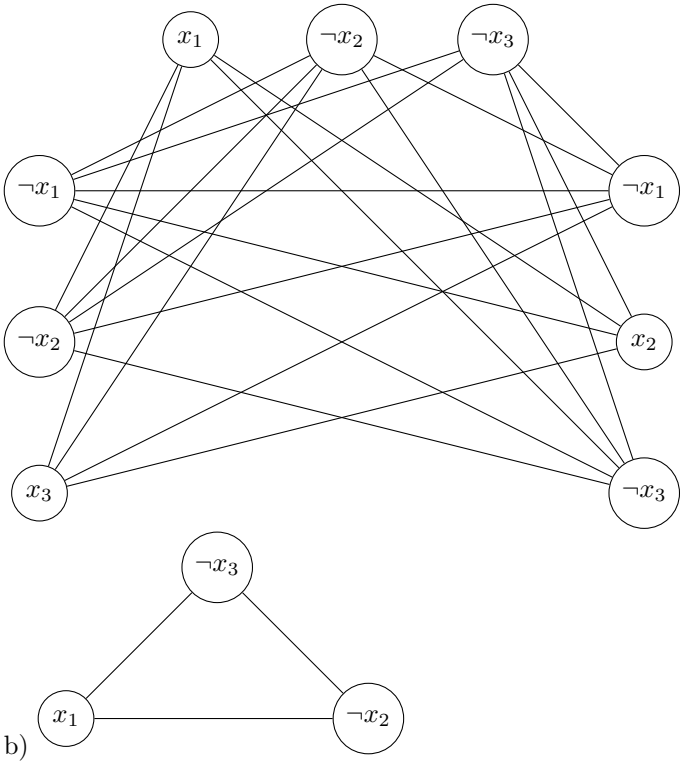
b) To show DSAT $\in$ NP, we just had to show two different assignments for all variables and satisfy them. In order to reduce 3SAT to DSAT, we can create a boolean function x', given a 3cnf-function x. x' contains the clause $(z \vee \overline{z})$ to x, where z is a variable not in x. We can then check x' $\in$ DSAT, and we know that this reduction is in polynomial time. Using this information we can prove that x $\in$ 3SAT if the new boolean function, x' $\in$ DSAT. If x is not satisfied then x' is also not satisfied. If x $\in$ 3SAT, then using the assignment of variables in x, where z= 0 and 1are valid, then theres at least 2 assignments that are satisfied, hence we can further conclude that x' $\in$ DSAT, and that 3SAT $\leq_p$ DSAT.

Problem 2. (20 points) Theorem 34.11 in our textbook shows that CLIQUE is NP-complete using the reduction 3SAT $\leq_p$ CLIQUE. (a) Describe the graph corresponding to the 3SAT instance

$$(x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3).$$

in this reduction. (b) Find a satisfying assignment and the corresponding clique in the graph. [Hint: The package tikz can be helpful in drawing beautiful graphs in LaTeX.]

Solution. Graph:
a)



b)
This is the clique (x_1, $\neg x_2$, $\neg x_3$), where there's common nodes and all have edges going to each other.
To satisfy the given statement:
x_1 is True
$\neg x_2$ is True, so x_2 is False
$\neg x_3$ is True, so x_3 is False
$(x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3)$
$(T \vee T \vee T) \wedge (F \vee T \vee F) \wedge (F \vee F \vee T)$
$T \wedge T \wedge T$, which ultimately gives us True.

Problem 3. (30 points) The SUBSET SUM problem asks to decide whether a finite set S of positive integers has a subset T such that the elements of T sum to a positive integer t. (a) Is (S, t) a yes-instance when the set S is given by

$$S = \{2, 3, 5, 7, 8\}$$

and $t = 19$? Prove your result. (b) Why is a brute force algorithm not feasible for larger sets S (c) Explain in your own words why the dynamic programming solution to SUBSET SUM given in

`https://www.cs.dartmouth.edu/~deepc/Courses/S19/lecs/lec6.pdf`

is not a polynomial time algorithm.

Solution. a) Based on the given information, we must determine if (S,t) is a yes-instance, when set S is $\{2, 3, 5, 7, 8\}$ and t= 19. In order to determine this we can first start by adding all the elements in the set, excluding the last one. So it would be 2+3+5+7 = 17. From this we notice that 17 is less than 19 and that in order to reach the target 8 must be included in the set. Repeating this same step, we can now get 2+3+5+8 = 18 which is also less than 19, thus concluding that 7 is also an element that must be part of the set. So now given a set where there is elements 7 and 8, which total up to 15, we know that from the subset $\{2, 3, 5\}$, we much choose elements that equal up to 4. However, 5 is greater than 4, and 2, and 3 are less than 4. Adding 2 and 3 gives us 5, so there is no element or combination of elements that give us 4, to help us reach the target. Hence, there is no subset that exists that sums up to the target value.

b) While brute force algorithms can give optimal solutions, it is easier to use them for small sets. Using a brute force algorithm on a larger set S, would just be too tedious, and would just take a long time. Ideally while solving a problem like this you want the optimal solution along with the fastest time complexity. For a set that is too large, with too many elements, the brute force method is inefficient and time consuming.

c) The dynamic programming solution to SUBSET SUM, first checks for $F[m, 0]-$ 1 for all m, which is true because there can be an empty set. The second check is $F[0, b] - 0$ for all $b > 0$, which isn't necessarily true, since there can't be anything greater than 0 in an empty set. After checking those base cases it then goes through two for loops and sets some conditions to make sure that $F[n, B]$ has an answer, where if its 1 there is a solution else there isn't one. For this dynamic programming solution, the time complexity is $O(nB)$. In $O(nB)$, B could be anything, (eg. it could grow exponentially). In some case, the size of B could even be larger that the number of elements in the set. Because the size of B is unknown and can get very large. We know that the algorithm does not exist for the solution where it is bounded by a polynomial function with some known length n. Thus, this dynamic programming solution is not a polynomial time algorithm.

Problem 4. (20 points) Exercise 34.5-5 on page 1101 using the reduction SUBSET SUM $\leq_p$ SET PARTITION.

Solution. For this problem we must show if the set partition problem is NP complete. So to begin we can consider the factors to show that any problem is NP complete. In an NP complete problem, there is a P time algorithm to solve for A, any problem B can be reduced to A in NP complete, reduction of B to A happens in P time, and finally the original problem A has solution only if B has a solution. To Set Partition $\in$ NP, we must make an original estimate and then later check if they both have equal sums. For reduction, we must start by reducing the subset-sum, where x is a set of integers with some target t and s

is the sum of the elements in x. Assume that there two partitions created have equal sums, if so then the problem is in NP. This reduction from the subset to the set-partition takes P time, and we know that set x is part of the subset sum if it is part of set-partition problem ($x' = x \cup \{s - 2t\}$). Using this, we can say that if there are elements part of the set-partition problem that add up to t, which is the target, then the remaining elements in x add up to s-t. There is a partition part of x' where both partitions equal to s-t. Using this we know that one of the sets have s-2t, however taking this out of the set will lead us to a set that sums up to t. Hence, all elements in the set exist in x, and that the set partition problems is NP complete.

Make sure that you write the solutions in your own words!

Checklist:

☐ Did you add your name?
☐ Did you disclose all resources that you have used?
(This includes all people, books, websites, etc. that you have consulted)
☐ Did you sign that you followed the Aggie honor code?
☐ Did you solve all problems?
☐ Did you submit the pdf file resulting from your latex file of your homework?