Northeastern University
CS5800 – Algorithms
Spring 2023 Semester, Jonathan Mwaura

Problem Set #3

This Problem Set is due at **11:30PM on Wed** 4^{th} Oct, and will be submitted on GRADESCOPE.

This Problem Set will be marked out of 30. There are 2 problems on divide and conquer.

Please type (or neatly handwrite) your solutions on standard 8.5×11 paper, with your name at the top of each solution. Ensure that you submit you solutions in one file PDF file on Gradescope. **each problem sets solution should be on in its own individual page, Gradescope will help ensure you submit each solution under its correct problem number**

While a solution must be absolutely perfect to receive full marks, I will be generous in awarding partial marks for incomplete solutions that demonstrate progress.

So that there is no ambiguity, there are two non-negotiable rules. A violation of either rule constitutes plagiarism and will result in you receiving an F for this course.

(a) If you meet with a classmate to discuss any of the Individual Problems, your submission must be an individual activity, done in your own words, away from others. The process of finding a solution might take 3 - 5 iterations or even more BUT you learn from all these attempts and your confidence grows with each iteration.

(b) These problem sets might seem hard on a first look. They are designed to be so. We learn by attempting problems, struggling through them and coming on top. I encourage you to make this learning exercise worth your while. What do I mean? Open the problem sets as early as you get them, then do not look at hints or answers any where (including on the internet and consulting other students for direct answers), give it the best shot you can. If you get stuck come to Professor or TA's office hour and we shall be glad to listen to your rationale and work with you till you are able to tackle the problem sets.

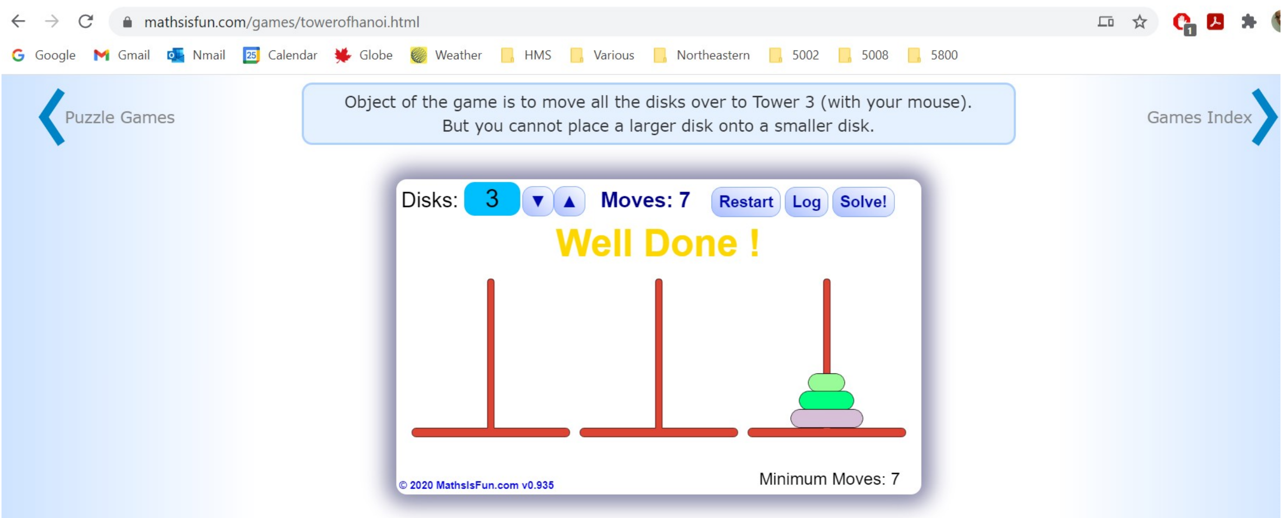
Problem #1 (10 points)

Play one or more games of "Towers of Hanoi", which you can do so on this website:

`https://www.mathsisfun.com/games/towerofhanoi.html`

There are three towers, and n disks. Your goal is to move all n disks from Tower 1 to Tower 3, but you may never place a larger disk on top of a smaller disk.

For example, when $n = 3$, the game can be solved in 7 moves, which is the optimal result.



(a) Attach a screenshot of you winning the $n = 4$ game in exactly 15 moves. (No proof or explanation is necessary – all you need to do is insert a .jpg image, just as I did above.)

(b) Let $T(n)$ be the minimum possible number of moves required to solve the game when there are n disks. For example, $T(2) = 3$ and $T(3) = 7$.

Clearly explain *why* $T(4) = 15$, showing it is possible to solve this game in exactly 15 moves and proving why it is impossible to solve this game in 14 (or fewer) moves.

(c) Find a recurrence relation for $T(n)$ and clearly and carefully explain why that recurrence relation holds. Then solve the recurrence relation using any method of your choice to determine a formula for $T(n)$ that is true for all integers $n \geq 1$.

(d) Substitute $n = \log(m)$ into your recurrence relation for $T(n)$ above, and use the Master Theorem to prove that $T(n) = \Theta(2^n)$. Briefly explain how and why your formula in part (c) is indeed $\Theta(2^n)$.

Problem #2 (20 points)

In this question, you will prove the Master Theorem in the special (and most important) situation when $f(n) = n^z$ for some real number z.

This result enables us to determine tight asymptotic bounds for various recurrence relations, which will help us tremendously in algorithm design and algorithm analysis.

If you can reproduce the proof of this result, then you will understand how the Master Theorem works in all situations – e.g. when $f(n) = 4n^3 + 2n^2 \log n + 5n + 100 \log n + 777$. But for the purposes of this problem, we will assume $f(n) = n^z$.

Let x, y, z be real numbers for which $T(1) = 1$ and $T(n) = xT(n/y) + n^z$.

(a) If $z < log_y(x)$, prove that $T(n) = \Theta(n^{log_y(x)})$.

(b) If $z = log_y(x)$, prove that $T(n) = \Theta(n^{log_y(x)} \log_y n)$.

(c) If $z > log_y(x)$, prove that $T(n) = \Theta(n^z)$.

(d) Some of you have taken a course in Linear Algebra, a core course in an undergraduate mathematics curriculum. In this course, students learn how to multiply two n by n matrices, and one can easily design an algorithm to perform matrix multiplication, running in $\Theta(n^3)$ time.

In 1969, Volker Strassen developed a recursive method to perform matrix multiplication, where the running time $T(n)$ can be given by the recurrence relation $T(n) = 7T(n/2) + n^2$.

Using any of the results in (a), (b), or (c), determine the running time of Strassen's Algorithm and show that it is faster than the standard algorithm that runs in $\Theta(n^3)$ time.

Note: if you are unable to prove (a), (b), (c) in its general form, you are welcome to instead solve the following simplified version of the problem, where you let $y = 2$ and $x = 16$, and prove the result for $z = 3$ in part (a), $z = 4$ in part (b), and $z = 5$ in part (c). If your proof is correct, you will be given 10 points (i.e. instead of the 15 points for the original a,b,c) for solving the simplified version of the problem.