SAKSHAM RATHI

22B1003

QUESTION - 2

(a) Let the represent n bits using

$x_1, x_2, x_3, \dots, x_n$

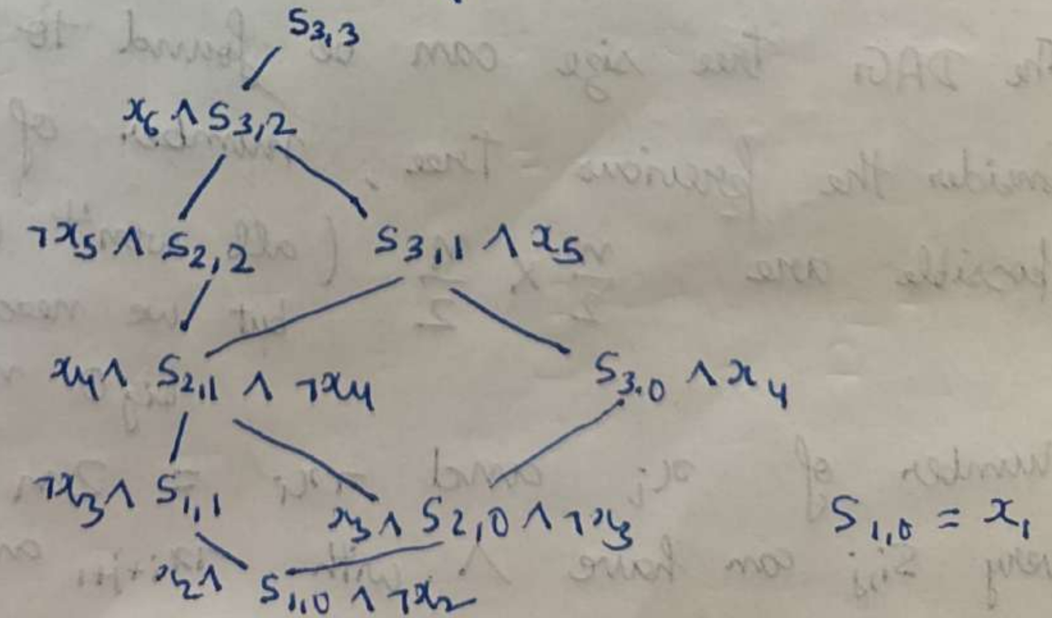
0 encodes '(' and 1 encodes ')'

We will introduce auxiliary variables $s_{i,j}$ to represent binary string of length $i+j$.

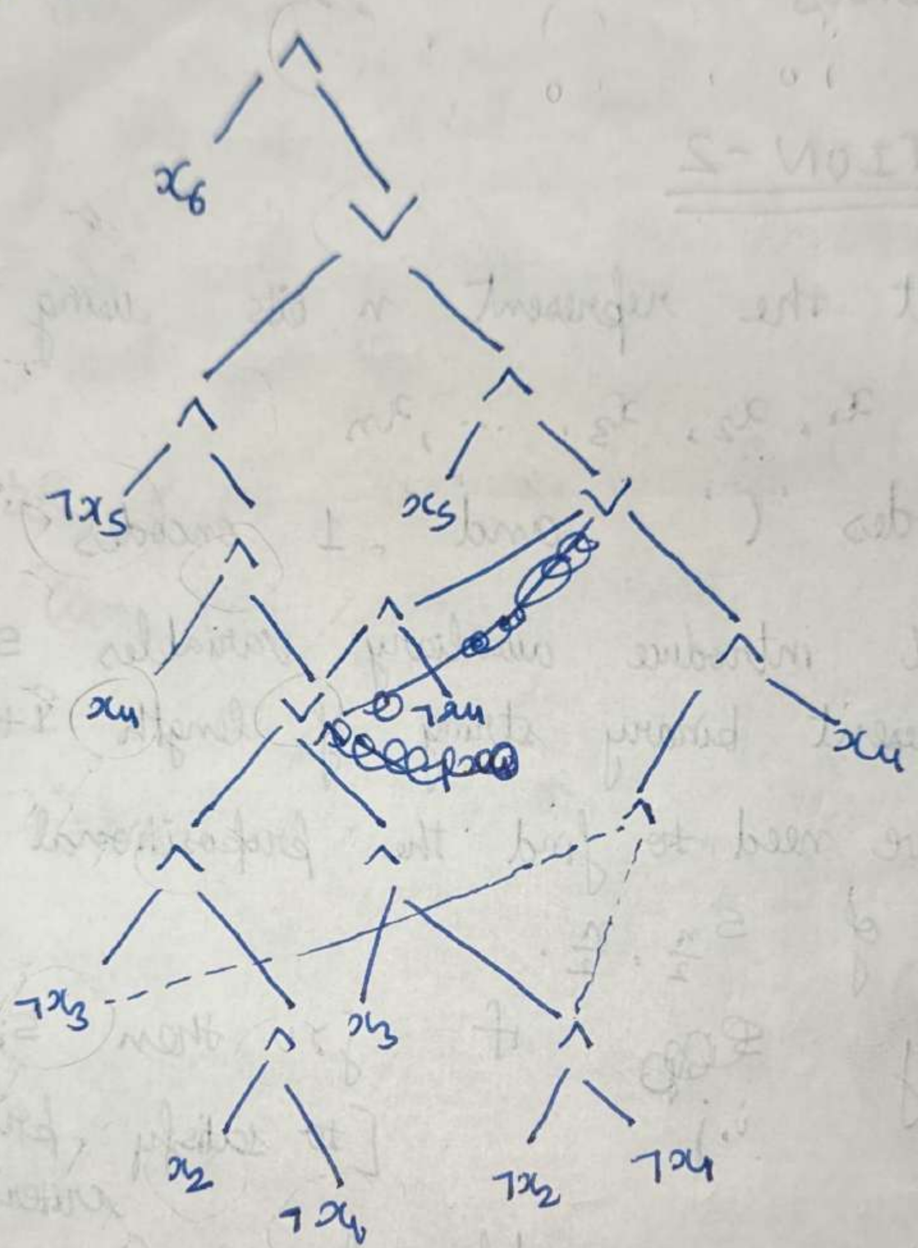
Now we need to find the propositional logic formula of $s_{\frac{n}{2}, \frac{n}{2}}$.

For every i, j if $j > i$ then $s_{i,j} = \text{false}$ [to satisfy proper prefix criteria]

Let us take the example of $n = 6$ and construct the DAG:



Let us construct the DAG tree only using x_i and $\wedge, \vee, \neg$.



The DAG tree size can be found to be $O(n^2)$.
Consider the previous Tree, number of $s_{i,j}$ possible are $\frac{n}{2} \times \frac{n}{2}$ (all won't occur, but we need size in big O notation)

number of x_i and $\neg x_i = 2n$
Every $s_{i,j}$ can have $\wedge$ with $\neg x_{i+j+1}$ and x_{i+j+1}

So, total number of $\wedge$ $\leq \left(\frac{n}{2} \times \frac{n}{2}\right) \times 2$

Total DAG size $\leq 2n + \frac{n^2}{2} + \frac{n^2}{4}$

$= 2n + \frac{3n^2}{4} = O(n^2)$

Hence, we have proved that DAG size is atmost $O(n^3)$ $\left[O(n^2) \leq O(n^3)\right]$

(b) Our end goal is to reach $s_{\frac{n}{2}, \frac{n}{2}}$ i.e. the number of '(' and ')' should be equal.

Clearly, if n is odd, we won't be able to assign 0 and 1 to x_i equally.

This means, that our formula won't be satisfiable.

(c) For every node at height i

If $x_i = 1$ then we move to sub-DAG with x_i

If $x_i = 0$ then we move to sub-DAG with $\neg x_i$

So, for every i, we need to evaluate only one node. ~~is~~ ~~[illegible]~~

Also, since we are moving along a particular subtree, whenever we encounter $\vee$, that automatically evaluates to true. Let us take $n=6$ and input string to be ( ) ( ) ( )

$x_1 = 0$

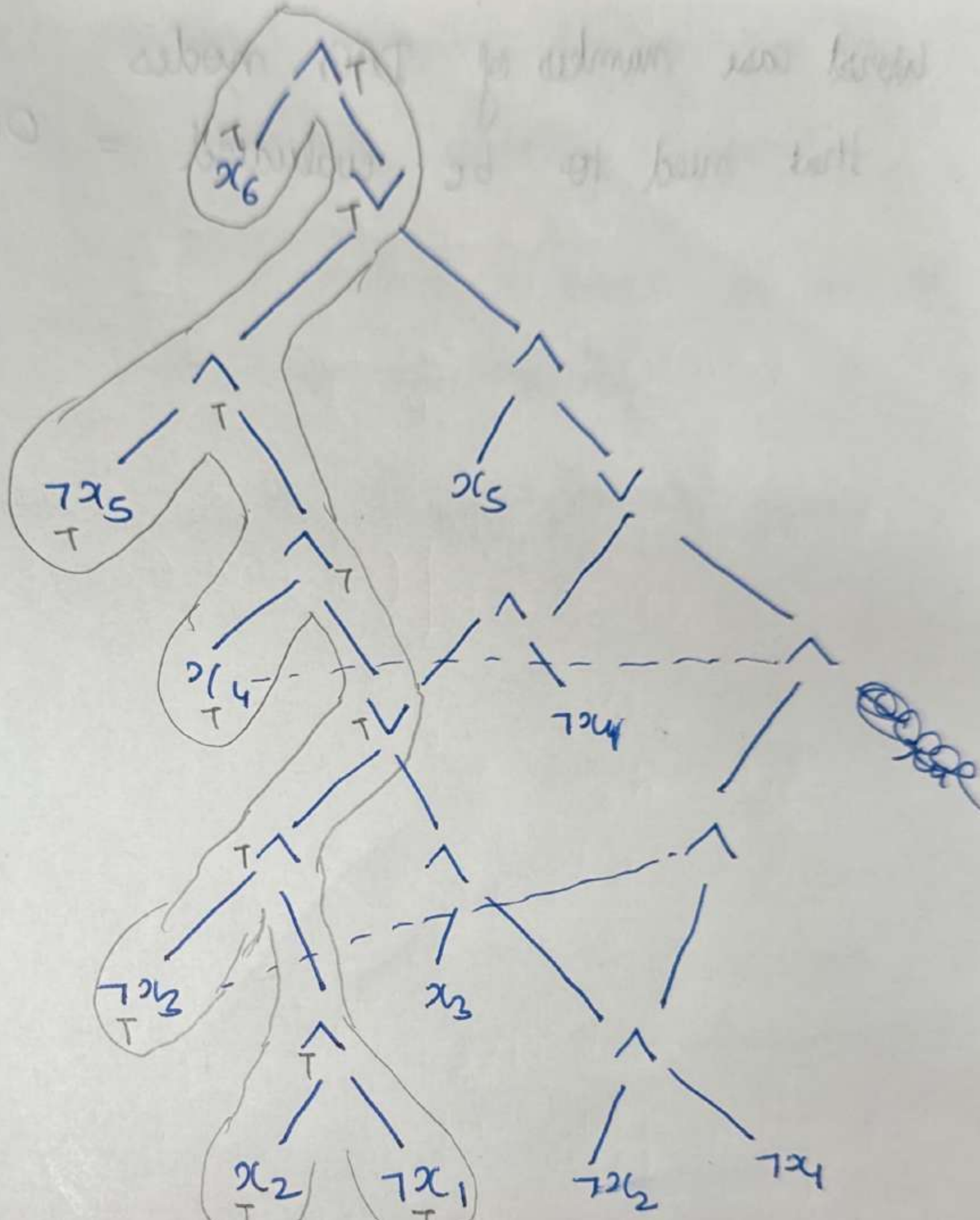
$x_2 = 1$

$x_3 = 0$

$x_4 = 1$

$x_5 = 0$

$x_6 = 1$



Number of x_i variables evaluated $\leq n$

$\wedge$ encountered $\leq n$

$\vee$ encountered $\leq n$

Total number of operations performed $\leq 3n$

Worst case number of DAG nodes that need to be evaluated $= O(n)$