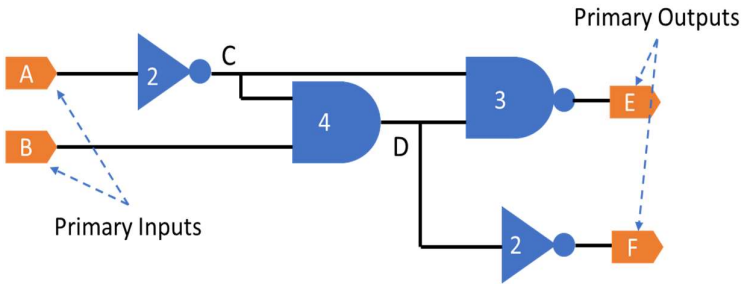
COL215, SUMMER 2023

SOFTWARE ASSIGNMENT 1 REPORT

ANUP LAL NAYAK (2022CS51827) JAKHARIA RISHIT (2022CS11621)

INTRODUCTION

In this assignment, we will try to simulate propagation delay in Python through code.



Problem Description

Each gate implements a Boolean function of its inputs, and there are no cycles in the circuit. The numbers annotated on the gates represent the delay *d* through the gate, in nanoseconds. If the inputs of a gate are ready at time *t*, then the output is ready at time *t+d*.

(a) How much time does such a circuit take to compute its primary outputs? For the example above, the maximum delay *for E is 2+4+3=9, and for F is 2+4+2=8*. Write a program to compute the **delay** of every primary output in a combinational circuit (defined as the earliest time when the output is ready), given the circuit representation and delay information of the individual gates.
Input: Circuit file (see example file *circuit.txt* for diagram above)
Input: Gate Delay file (see example file *gate_delays.txt* for diagram above)
Output: Output Delay file (see example file *output_delays.txt* for diagram above)

(b) Extend your program to answer the converse question: suppose we require the output to be ready at a specific time. When do we require each input to be ready so that the output timing requirement is met?
Input: Circuit file (see example file *circuit.txt* for diagram above)

Input: Required Output Delay file (see example file *required_delays.txt* for diagram above)
Output: Input Delay file (see example file *input_delays.txt* for the diagram and *required_delays.txt* file above)

APPROACH AND LOGIC:

a) First, we create a data structure, "Gate," with the required class variables that help us calculate the delay in the given circuit, then we use the following piece of the algorithm to find the final delay.
    a. Delay of nth signal = Delay of nth gate + Delay till (n-1)th signal
    b. Time complexity is O(n), as we are going over every signal delay once.
b) In part b, we adopt a brute force approach to check for every value of the primary inputs up to the upper bound defined as the max of the required delays.
    a. Here, the time complexity jumps to O(n^3) as we are checking for every value of A and B, so for the values of A and B it requires O(n^2), and checking requires O(n), hence a total of O(n^3)

TEST CASES RUN:

a) Changing the gate time delay to 0 to check if 0-time delay is still working
    a. NAND2 0
    b. AND2 4
    c. NOR2 3.5
    d. OR2 4.5
    e. INV 0
b) Changing the outputs to check if more than three outputs work, in part a
    a. PRIMARY_INPUTS A B
    b. PRIMARY_OUTPUTS E F G
    c. INTERNAL_SIGNALS C D
    d. INV A C
    e. AND2 C B D
    f. NAND2 C D E
    g. INV D F
    h. INV C G
c) With weird floating point numbers that may cause floating point errors, the output is rounded to 3 decimal places to prevent the issue, in part_a
    a. NAND2 3.999999
    b. AND2 4.4563456
    c. NOR2 3.534564
    d. OR2 4.53456
    e. INV 2.3456
d) Testing impossible outputs in part_b to see if the code terminates and prints "not possible", in part b
    a. E 15
    b. F 13

CONCLUSION:

Doing this assignment enhanced our understanding of Propagation delays' basics; we learned in depth about its structure and working.

Reflecting the understood logic into blocks of code was challenging, and we researched and learned a lot about how to manage these by application of correct data structures. Debugging was also a phase where we learned a lot about what conceptual or semantic mistakes we were making with our application of the solution, and it helped to refine our understanding even further.

We also researched the applications of Propagation delay and in which different fields its logic is applied to solve various problems.