Assignment 1

Varun SM - EE18BTECH11030

Download all files from

https://github.com/Elonian/C_and_DataStructures/tree/main/Assignment_1

1 Problem

(Q 17) The following C program is executed on a Unix/Linux system

```
#include <unistd.h>
int main(){
    int i;
    for(i = 0; i < 10; i++){
        if(i%2 == 0){
            fork();
        }
    }
}
```

The total number of child processes created are.

2 Solution

Answer: Number of child process created are 2^n-1 = 2^5-1 = 31. Where n is the number of times fork() call occurred.

Explanation:

- **fork()** is used for process creation on Unix-like operating systems. It wont take any arguments and returns process ID.
- The purpose of fork() is to create a new process, which becomes the child process of the caller(Parent process).
- After a new child process is created, both processes will execute the next instruction following the fork() system call.

In the **for** loop **if** condition is satisfied only for even values of i;

First fork method is called at i = 0 and child process is created with variable i = 0 in child process (c1). In level 1 one child process(c1) and parent process(p) is running.

In parent process(p) when i = 2 . it calls fork method then a new child process(c2) is created with variable i = 2.

similarly for the child process(c1) at i = 2 another child process(c3) is created.

At each level in row all the child processes along with parent process running at given instant are present.

In the Level 2 total number of child process = 3 = 2^L -1.

where L is the highest level of tree or number of times fork() call occurred.

This process creation continuous till level 5 because variable **i** takes even values.

Number of child process at level 5 = 2^5 -1 = 31.

Graphically it is depicted as below.

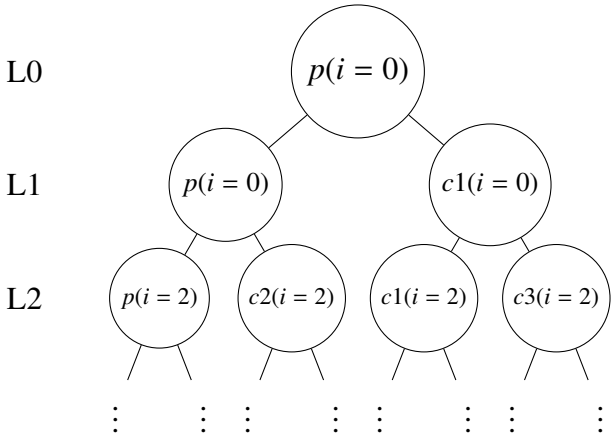

Fig. 0: Tree representation