

An abstract network graphic on a dark blue background. It consists of numerous nodes, represented by small circles in shades of blue, purple, and yellow, connected by thin, light blue lines. The nodes are distributed across the frame, with a higher density on the left side and a more sparse arrangement on the right. The lines create a complex web of connections between the nodes.

HW1

pro
ject

OOP

Presentation

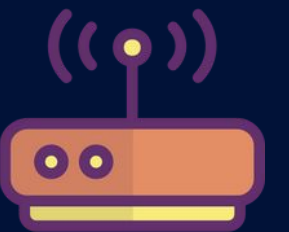
會資四 牟宗立 409526040

The background of the slide features a complex network diagram. It consists of numerous circular nodes of varying sizes, colored in shades of blue, purple, and yellow. These nodes are interconnected by a web of thin, light-colored lines, creating a dense, interconnected structure that resembles a neural network or a complex data graph. The overall aesthetic is modern and technological, set against a dark blue gradient background.

Structure

```
//record the precedence and what else
class precedence{
public:
    ....int gateId;
    ....int logQubitID1;
    ....int logQubitID2;
    ....int precedence;
};

// Custom node structure to store additional data of physical graph
typedef struct Node {
public:
    ....int phy_id;
    ....int logicalId;
    ....int size; // connective size
    ....vector<int> neighbors; // List of neighboring node IDs
}Node;
```

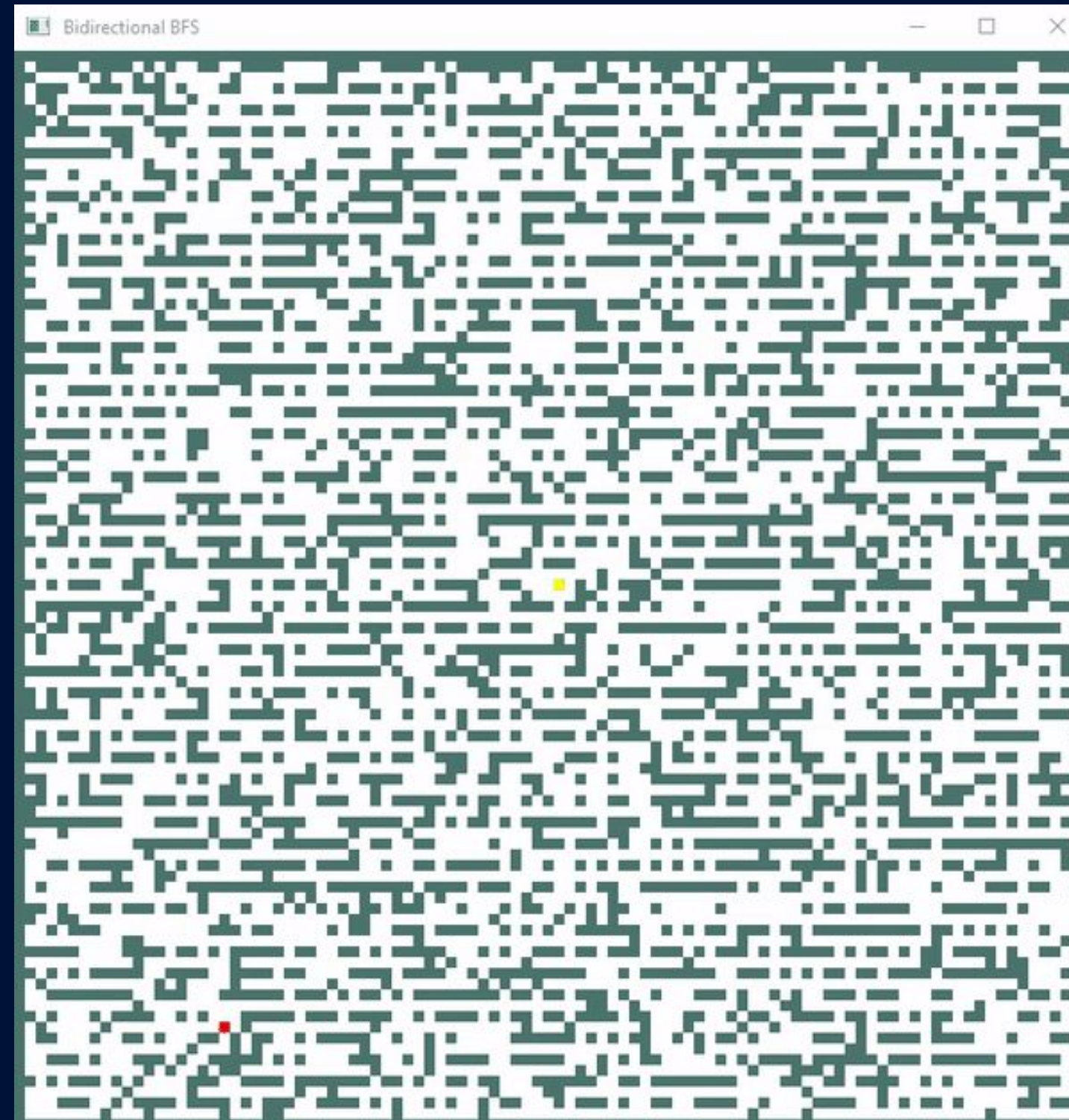




What

I Do


Bidirectional Bfs





Submission details

Problem: HW1 - OOP24HW1 Submitted: 3/25 10:14 Language: C++ Compilation: **successful**

Result: **TIMELIMIT** Testcase runs: 

#18, name: 18, runtime: 5.993s, result: timelimit


Compilation output

There were no compiler errors or warnings.

Close

Submission details

Problem: HW1 - OOP24HW1 Submitted: 3/25 11:45 Language: C++ Compilation: **successful**

Result: **CORRECT** Testcase runs: 

#18, name: 18, runtime: 4.961s, result: correct

Compilation output

There were no compiler errors or warnings.

Close

After implementing
Bidirectional Bfs



Result: **CORRECT** Testcase runs: 

#18, name: 18, runtime: 3.117s, result: correct

After changing endline to “\n”

Why



ChatGPT

In C++, `std::endl` is more than just a newline character (`\n`). It not only inserts a newline character into the output stream but also flushes the stream. Flushing the stream means that it forces any buffered output to be written to the output device immediately.

When you use `'\n'`, it only inserts a newline character into the output stream without flushing the stream. Flushing the stream can be a relatively expensive operation, especially if you're performing many output operations.

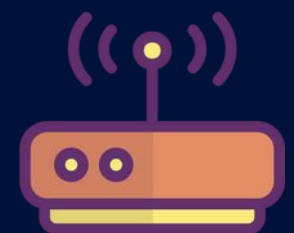
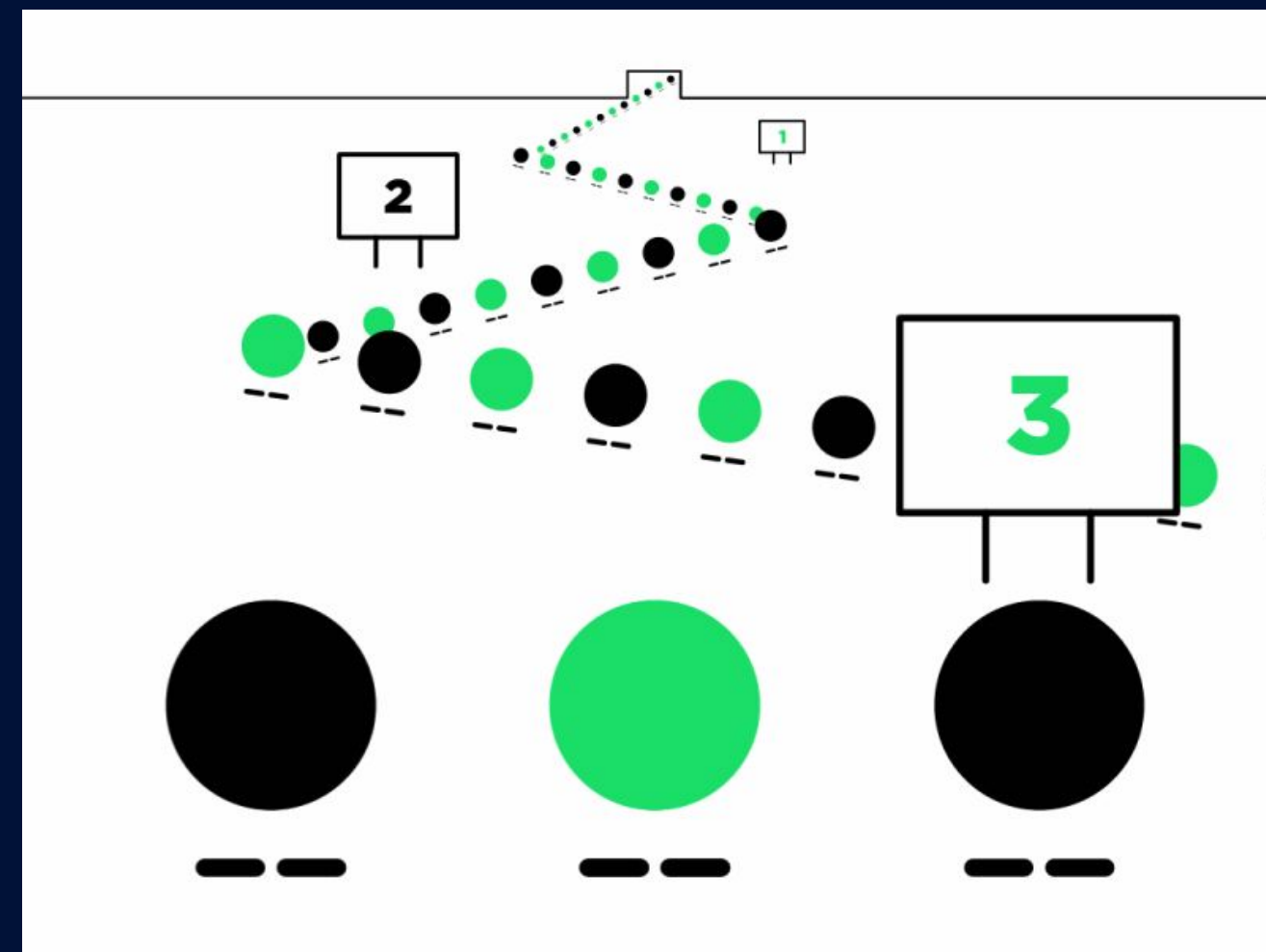
So, if you're repeatedly using `std::endl` in a loop or in a performance-critical section of your code, it can significantly slow down the program compared to using `'\n'` because of the additional overhead of flushing the output stream.

How did i do mapping?

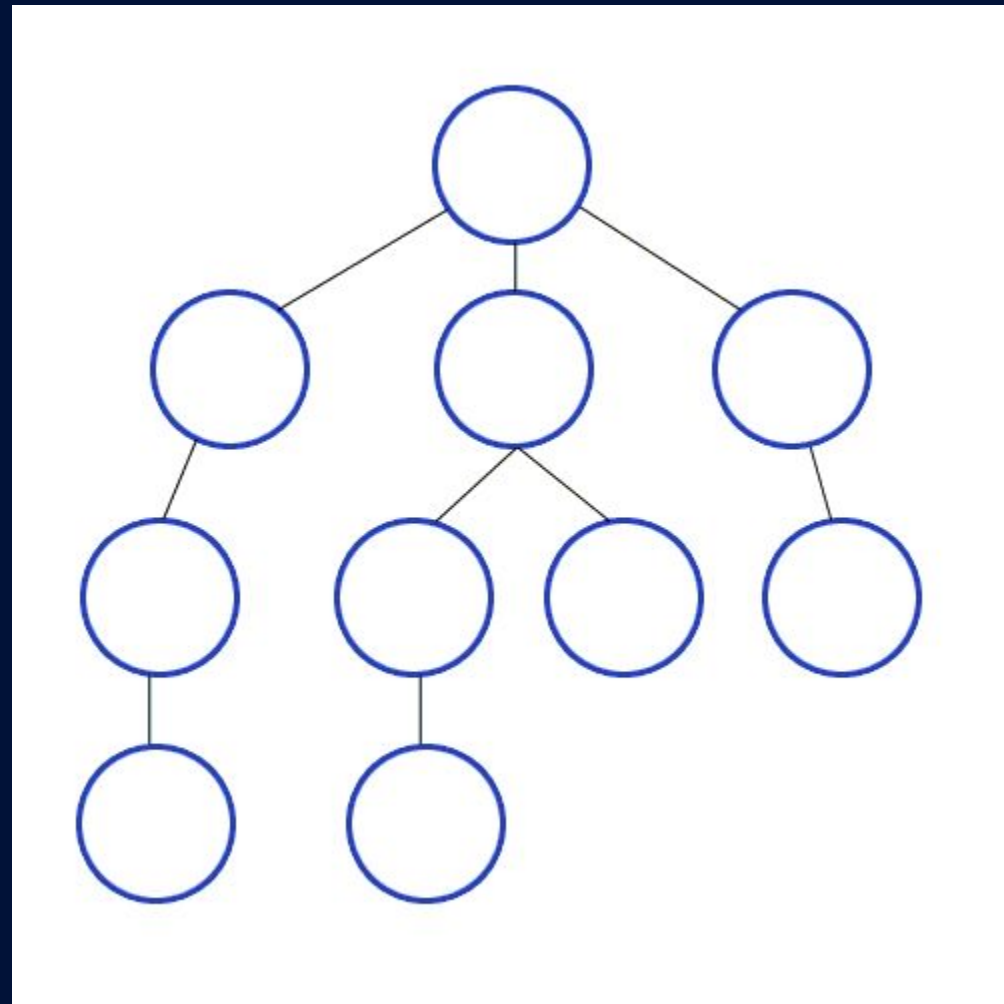
```
//record the precedence and what else  
class precedence{  
public:  
    int gateId;  
    int logQubitID1;  
    int logQubitID2;  
    int precedence;  
};
```

```
int logQubitID1;  
int logQubitID2;
```

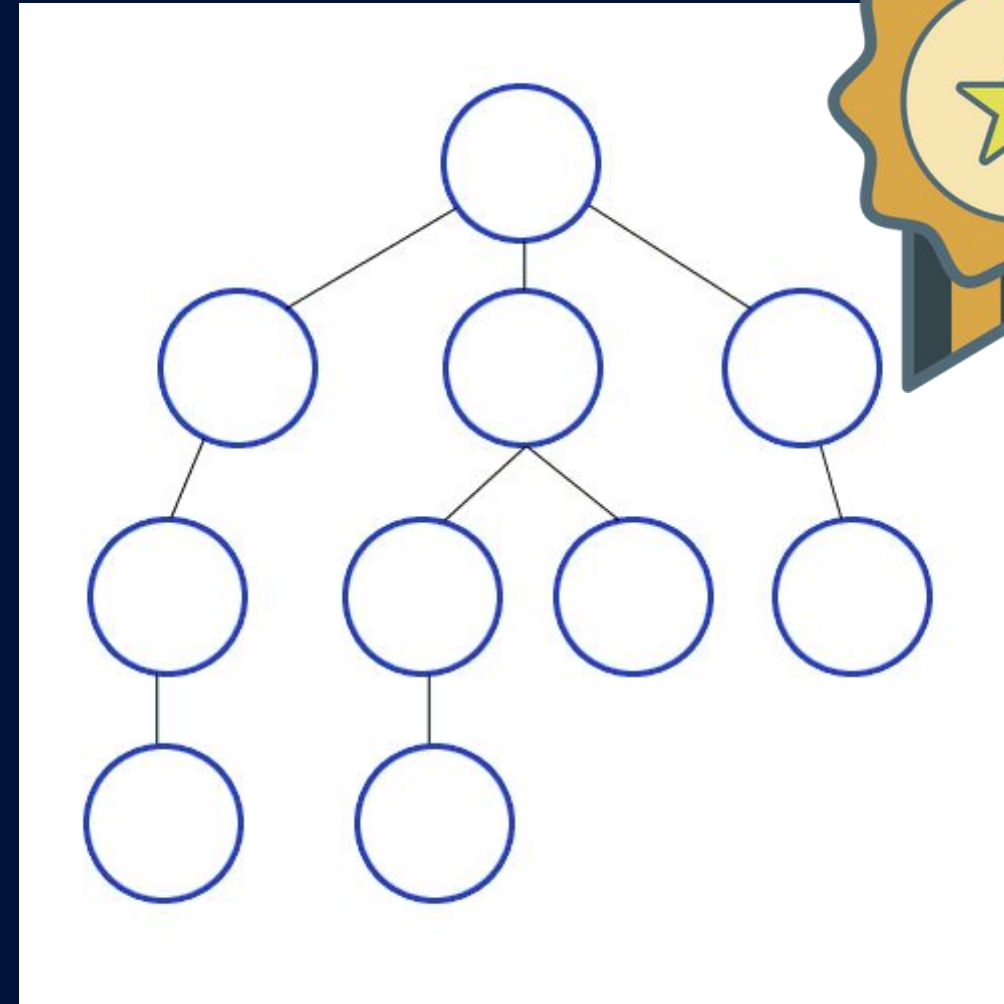
put logQuibit to the queue



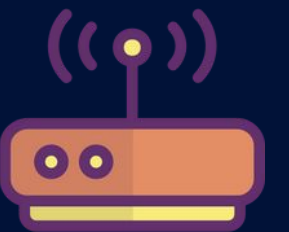
Choosing Bfs or Dfs to initialize?



BFS



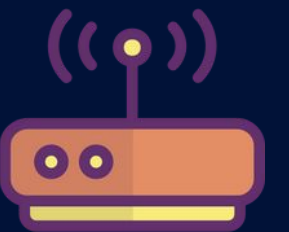
DFS





Be Better

1. Do the easy connection first
2. Put those busy node together
3. Improve topological sort precedence



An abstract graphic featuring a network of nodes and connecting lines. The nodes are represented by small circles in various colors, including purple, blue, and yellow. The lines are thin and connect the nodes in a complex, web-like pattern. The background is a solid dark blue. The text "Thanks For Attention" is written in a large, bold, pink font, positioned on the right side of the image.

Thanks For
Attention