

Icon

Description automatically generated

Search Algorithm

(Threads)

**FAST - National University of Computer and Emerging Sciences**

**CS2006 - Operating Systems**

**Spring 2023**

**Project Report**

**Group Members**

21K - 4523 Nabiha Rajani

21K - 4899 Asad Tariq

**Problem Description**

*“A variation of Dijkstra's Algorithm; A\* is an informed search algorithm, or a best-first search, meaning that it is formulated in terms of weighted graphs: starting from a specific starting node of a graph, it aims to find a path to the given goal node having the smallest cost (least distance traveled, shortest time, etc.). It does this by maintaining a tree of paths originating at the start node and extending those paths one edge at a time until its termination criterion is satisfied.”*

**Objective**

Apply multithreading to the *A\* Search Algorithm*. By distributing the array to be searched across threads to find the shortest path and lowest cost, efficiency is improved as multiple threads run independently.

**Details**

Although a variation of *Dijkstra's Algorithm* (i.e., it seeks out the shortest path between nodes by selecting edges with the least cost), it is faster since it uses heuristic values for the nodes to estimate whether a path is worth exploring.

A heuristic value is an estimation of the distance between two vertices. In this project we used the standard *Manhattan Distance* strategy, using *Pythagoras’ Theorem* on the horizontal and vertical distances between nodes assumed to be located on a rectangular grid.

We use the formula:

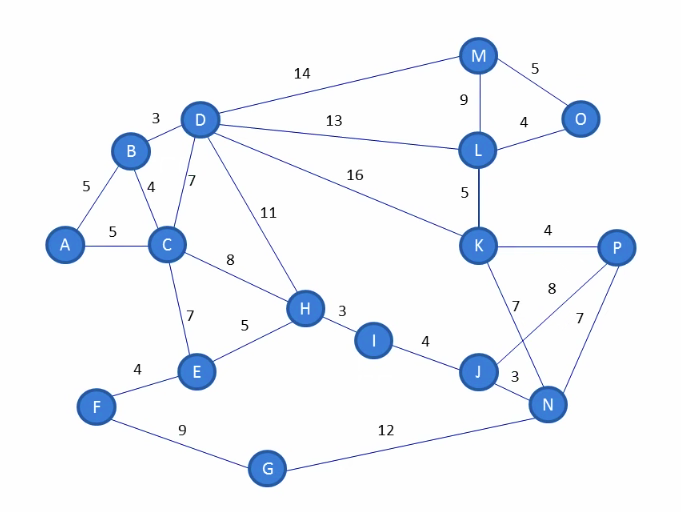
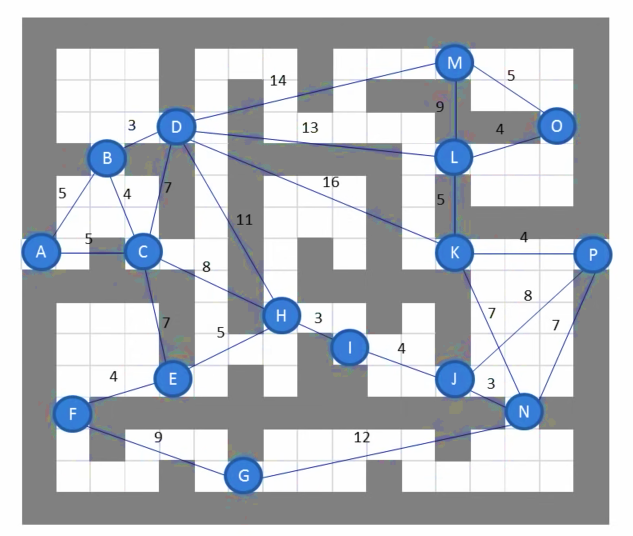
*f = g + h*

g = Dijkstra distance

h = heuristic distance

f = resultant sum

*Shown****:*** *The vertices and edges modeled on a square grid as seen by the algorithm*

 ****

**Multithreading Application**

In the code for the main algorithm, paths are explored sequentially in all directions. A list is continually updated as soon as shorter paths are found. The order in which paths are explored does not impact the success of the algorithm, since the paths are in different directions and different data is being worked upon.

Hence, we took each direction (eight in total) and separated them into functions. Then we simply made different threads for each direction, passing a different function as an argument when creating each thread. Now all directions can be explored in parallel.

**Code Snippet**

pthread\_t T[8];

pthread\_create(&T[0], NULL, firstSuccessor, NULL);

pthread\_create(&T[1], NULL, secondSuccessor, NULL);

pthread\_create(&T[2], NULL, thirdSuccessor, NULL);

pthread\_create(&T[3], NULL, fourthSuccessor, NULL);

pthread\_create(&T[4], NULL, fifthSuccessor, NULL);

pthread\_create(&T[5], NULL, sixthSuccessor, NULL);

pthread\_create(&T[6], NULL, seventhSuccessor, NULL);

pthread\_create(&T[7], NULL, eighthSuccessor, NULL);

for (int i=0 ; i<8 ; i++) pthread\_join(T[i], NULL);

**Results**

Applying multithreading concepts to the A-Star Search algorithm yields quicker results and more efficient utilization of resources, as multiple directions are explored simultaneously.

In addition to multithreading, the code has been amended for clarity and clear output of the grid, intended source and destination cells, and path followed.

**Output**

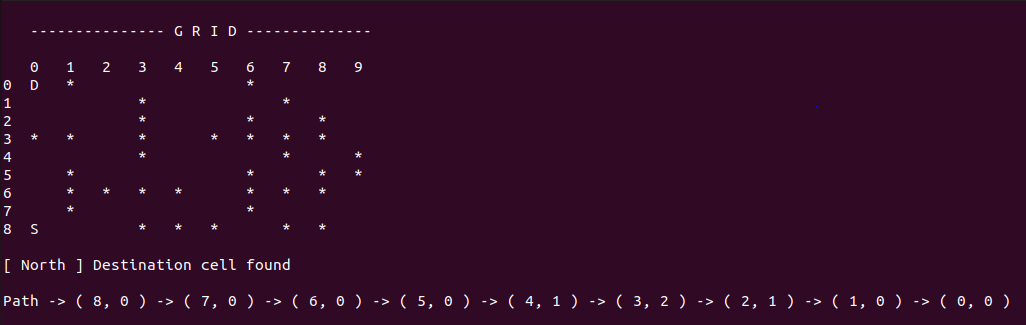
**Windows:**

**A picture containing text, screenshot, black

Description automatically generated**

**Linux(Ubuntu):**

****

****

**Conclusion**

The shortest path from each source vertex to the destination vertex has been successfully obtained by utilizing heuristic values across threads.

**Tools, Technology and Resources**

Learning and references:

<https://youtu.be/eSOJ3ARN5FM>

<https://www.geeksforgeeks.org/a-search-algorithm/> (non-multithreading)

Language: C++

Tools: Visual Studio 2019, Dev C++, Oracle VM VirtualBox (Ubuntu 22.04)