International Journal of Research, Science, Technology & Management
ISSN Online: 2455-2240, Volume 5 Issue III, December 2016

CNS using restricted space algorithms for finding a shortest path

Shinu Dubey, Aarti Singh

ABSTRACT:

These days, due to the heavy traffic, it is very difficult for drivers in big cities to find an easy and efficient route to a selected destination. Thus, route guidance has become a very important problem for drivers. As a result, car navigation systems are commonly installed in new vehicles. These systems are capable of performing some of the tasks traditionally performed by driver, such as reading the map and determining the best route to the destination. The process of finding optimum path from one point to another is called routing. The suggested algorithm is improved version of dijkstra algorithm that can search for the shortest path at a restricted search space. The search space is restricted by the use of a rectangle or a static and dynamic hexagon. These methods can improve run time and usaged memory because the visited nodes and edges are limited. In order to show the effectiveness of the proposed algorithm, it is tested on the city map of Gurgaon. Following, this result is compared to the result obtained with dijkstra's algorithm.

Keywords: **Shortest path, Geometrical figures, Car Navigation System.**

INTRODUCTION

Technology has evolved so much that life just seems to get easier by the day. The advancement of technology has given birth to a large number of gadgets and devices in the last few years. The Indian automobile industry has witnessed a sudden growth over the last few decades. These days, due to the heavy traffic, it is very difficult for drivers in big cities to find an easy and efficient route to a selected destination. Thus, route guidance has become a very difficult problem for drivers. As a result, car navigation systems are commonly installed in new vehicles. In fact, path finding is the act of obtaining the best route from one location to another. A path finding system, in a city, takes into account various factors, for example: the route with the least traffic, shortest route, most beautiful view route, and etc…. In this work the goal is to find the shortest path. In current scenario, the road distance stored in map data has been used for optimum route calculation purposes. Technically, CNS is navigation system which includes mapping software and its application with remote sensing, land surveying, aerial photography, mathematics, photogrammetric, geography, and tools that can be implemented with CNS software. In the strictest sense, the term describes any navigation system that integrates stores, edits, analyzes, shares, and displays route information. In this work, we study three variants of the dijkstra’s algorithm and apply them to the car navigation system. In the three variants the shortest path is being searched by limiting the search space. This way the time complexity of the algorithm is also reduced. You simply enter your source and destination in the software. All the computer needs to map out the best route is the name of the street and town you are traveling to. All you need to do is follow the path it gives you. The Car Navigation System promises to be a valuable aid for travelers, sales people, delivery personnel, real estate agents, and drivers of emergency vehicle, anyone and everyone who needs to reach a variety of destinations.

Motivation: A fair understanding of the advantages and disadvantages of the different algorithms studied above helped us to focus on areas where the system’s performance can be improved by just reducing the search area. Our aim is to study and implement the various algorithms.

Literature Review

Sara Nazari et.al in [1] proposed some new shortest path finding algorithms that are improved version of Dijkastra algorithm, which states that search space is restricted by the use of rectangle or a static and dynamic hexagon. These methods can improve the run time and usaged memory because the visited nodes and edges are limited. These algorithms are implemented in car navigation system for determining the optimum shortest path from source to destination on a restricted space city map.

Liang Dai in [2] suggested the A* algorithm that can achieve better running time by using Euclidean heuristic function although it theoretical time complexity is still the same as Dijkstra’s. It can also guarantee to find the shortest path. The restricted algorithm can find the optimal path within linear time but the restricted area has to be carefully selected. The selection actually depends on the graph itself. The smaller selected area can get less search time but the tradeoff is that it may not find the shortest path or, it may not find any path. This algorithm can be used in a way that allowing search again by increasing the factor if the first search fails, Syarifah Teh Nadhiah in [3] suggested many different operations that can be done on graphs to finding a minimum path. Usually methods such as Kruskal's algorithm and Prim's algorithm find the most efficient way to solve that problem. However, if the distance between two given vertices needed to be calculated, an alternate method would be required. As we know, Dijkstra's algorithm may be useful to determine alternatives in decision making in finding a good route between the nodes. The huge and complicated road network in a modem country makes it difficult to find a best route with minimum path for traveling from one place to another. Dijkstra's Algorithm will be used to solve this problem effectively . Car navigation systems are being offered as a special feature of new cars for an increasing number of car-brands. A navigation system offers the driver the possibility to be guided to his destination, by means of spoken and/or visual advices. The key components of a car navigation system are positioning, route planning and guidance. J.H Verriet in [4] suggested the route planning functionality of a car navigation system. He expects that the system provides him an optimum route according to a particular cost function Planning optimum routes fast on very large road networks still poses a significant challenge to companies developing car navigation systems. At the same time, taking information on daily congestion patterns into account is getting more important. A car navigation system uses road networks that may contain millions of road segments. However, a driver expects a route to be calculated quickly, within a few seconds typically.

SOFTWARE REQUIREMENT SPECIFICATION (CNS Gurgaon)

Introduction

Finding an efficient route is difficult problem for many drivers. Car Navigation Systems are sometimes offered as a special feature on new cars. These systems are capable of performing some of the tasks traditionally performed by driver, such as determining the best route to the destination. The software should have all the basic functionalities like searching, getting site information, getting shortest distance between a source and destination etc.

Purpose

The purpose of Software Requirements Specification (SRS) document is to describe the behavior of the software “CNS Gurgaon”. Requirements Specification defines and describes the operations, interfaces, performance, and quality assurance requirements of the software. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate. The intended users of this software are the primarily the car drivers, tourists, conservationist of the historic sites.

Scope

This software will be used for finding the shortest possible path in the map of Gurgaon. The features that are described in this document are used in the future phases of the software development cycle. The features described here meet the needs of all the users. The success criterion for the system is based in the level up to which the features described in this document are implemented in the system.

Document Overview

The SRS will provide a detail description of the CNS software. This document will provide the outline of the requirements, overview of the characteristics and constraints of the system. This section provides the full description of the project for the user of the software. This section also provides the general factors that affect the product and its requirements. The items such as product perspective, minimum system requirements, assumptions and dependencies are described in this section. This section of SRS contains all the software requirements mentioned in section in detail sufficient enough to enable designers to design the system to satisfy the requirements and testers to test if the system satisfies those requirements.

Assumption and dependencies

- The user should have sufficient knowledge of computers. The user must know English language, as the user interface and installation guide will be provided in English. Browser must be installed on the target system. The user interface should be self explanatory so that even novice users with little knowledge can use it.

DATA FLOW DIAGRAM

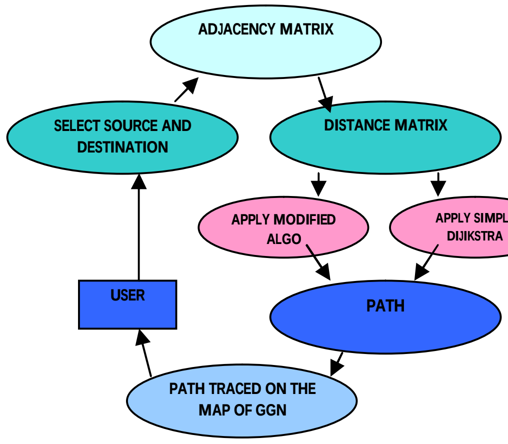
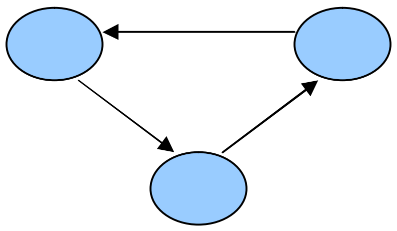


Fig2.2: Data Flow Diagram

DIJKSTRA'S ALGORITHM

Dijkstra's algorithm is called the single-source shortest path problem. It computes length of the shortest path from the source to each of the remaining vertices in the graph. The single source shortest path problem can be described as follows: Let G= {V, E} be a directed weighted graph with V having the set of vertices. The special vertex s in V, where s is the source and let for any edge e in E, Edge Cost(e) be the length of edge e. All the weights in the graph should be non-negative. Before going in depth about Dijkstra's algorithm let's talk in detail about directed-weighted graph. Directed graph can be defined as an ordered pair G: = (V, E) where V is a set, whose elements are called vertices or nodes and E is a set of ordered pairs of vertices, called directed edges, arcs, or arrows. Directed graphs are also known as digraph.



Digraph

Dijkstra's –A Greedy Algorithm: Greedy algorithms use problem solving methods based on actions to see if there's a better long term strategy. Dijkstra's algorithm uses the greedy approach to solve the single source shortest problem. It repeatedly selects from the unselected vertices, vertex v nearest to source s and declares the distance to be the actual shortest distance from s to v. The edges of v are then checked to see if their destination can be reached by v followed by the relevant outgoing edges. Before going into details of the project it is important to know how the algorithm works. Dijkstra's algorithm works by solving the sub-problem k, which computes the shortest path from the source to vertices among the k closest vertices to the source. For the dijkstra's algorithm to work it should be directed-weighted graph and the edges should be non-negative. If the edges are negative then the actual shortest path cannot be obtained.At the kth round, there will be a set called Frontier of k vertices that will consist of the vertices closest to the source and the vertices that lie outside frontier are computed and put into New Frontier. The shortest distance obtained is maintained in sDist[w]. It holds the estimate of the distance from s to w. Dijkstra's algorithm finds the next closest vertex by maintaining the New Frontier vertices in a priority-min queue. The algorithm works by keeping the shortest distance of vertex v from the source in an array, sDist. The shortest distance of the source to itself is zero. sDist for all other vertices is set to infinity to indicate that those vertices are not yet processed. After the algorithm finishes the processing of the vertices sDist will have the shortest distance of vertex w to s. two sets are maintained Frontier and New Frontier which helps in the processing of the algorithm. Frontier has k vertices which are closest to the source, will have already computed shortest distances to these vertices, for paths restricted upto k vertices. The vertex that resides outside of Frontier is put in a set called New Frontier.

USING RECTANGLE TO RESTRICT THE SEARCH SPACE:

In this method, the search space is restricted with a rectangle which covers a rectangular space with two nodes of source, s and destination, d.

It is observed, in fig, that the maximum length of edges connected to the source and destination nodes are equal to the maximum edges present in the considered graph. Hence, the maximum amount of parameter L is equal to the maximum edges present in the graph.

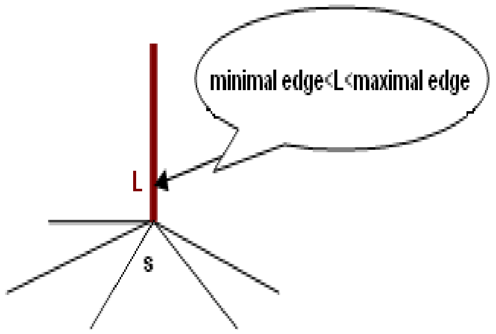


Fig Determination of L parameter

The Algorithm: In order to consider the presence of one node in a rectangle search space, we can use the function check (v, rectangle). This function is explained as below:

1. V is a node that should be considered in Rectangular space.
2. Rectangular space can be calculated on the Basis of L parameter. (L is the coefficient of maximum edges situated in the graph. According to experimental results, this Coefficient is recorded as 0.6). Add L to the source and destination nodes, and then create a rectangular shape as in fig 1.
3. The function checks v node in the Rectangular space as below:

If (ys-l < yv < yd+l) and (xs-l <xv < xd+l)
Node v is in a rectangle

USING STATIC HEXAGON TO RESTRICT SEARCH AREA

In this method, we restrict the search space with a hexagon. As shown in fig 3, it is most probable that the shortest path between two nodes is situated in a space around a direct line between these two nodes. We can show this space using a hexagon. As it is clearly shown below, the hexagonal space is obviously smaller than the rectangular space.

In order to check the presence of one node in a hexagonal search space, we can use the function **check(v, hexagon)**. This function is explained a below:
1. v is a node that should be considered in hexagonal space.
2. Hexagonal space can be calculated on the basis of L parameter.(L is the coefficient of maximum edge situated in the graph. According to experimental results, this coefficient is recorded as 0.6). Add L to source and destination nodes and then create hexagonal shape as in figure.
3. The function checks v node in the hexagonal space as below: (m is line slope)

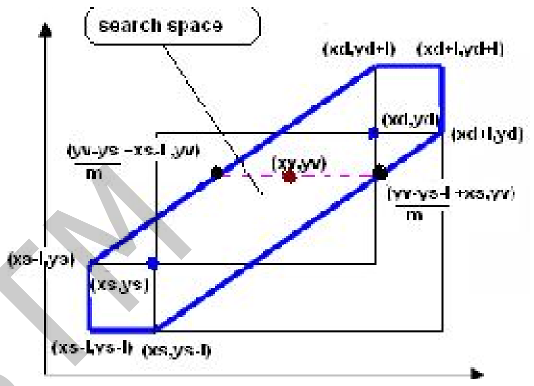


Fig Find node v in hexagonal restricted space

If (xv< xs-l) nor (xv>xd+l)

Node v is out of area

If (yv==ys-l,xs-l<xv<xs) or (yv==yd+l,xd< xv <xd+l)

Node v is out of area

If((yv-ys)(m+xs-l)>xv)or((yv-ys-l)/m+xs<xv)

Node v is out of area

RESTRICTING THE SEARCH SPACE USING DYNAMIC HEXAGON

In this method due to absence of path, search space increases. Increasing space would continue till finally an optimum path between two nodes (source and destination) found or in case of total graph considered, no path is presented. According to figure 4, increasing space is on the basis of constant parameter lm1, in which according to experimental assessments, the initial amount of lm1 is equal to 0.33. This parameter can be increased continuously by a constant amount 0.01 till finally an optimum path between two nodes is found. The stages of suggested algorithm are mentioned as below:

1. Set lm1 according to a constant amount 0.33.
2. Set m=lm1 * (max edge of graph) and add m to source and destination nodes and then create search space through source and destination nodes and m parameter.
3. Call dijkstra algorithm with new search space.
4. If path is found in the search space, display the path and end performing process. Otherwise increase LM1 to the size 0.01 and then go to second stage.

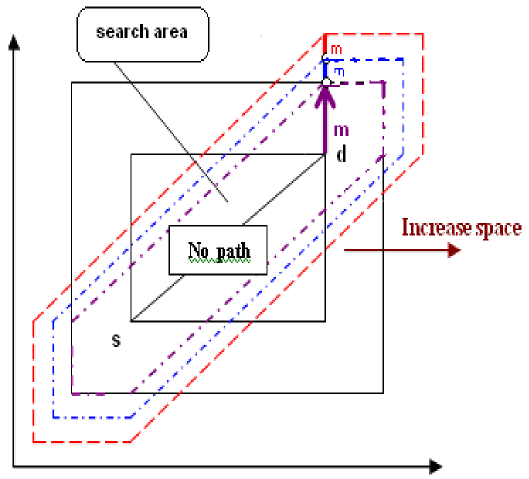


Fig 3.5 Restricted search area by dynamic hexagon

FEATURES: Search space increased if path not found until all nodes are traversed

RESULTS:

COMPARISON OF THE ALGORITHMS

CONSIDERED GRAPH

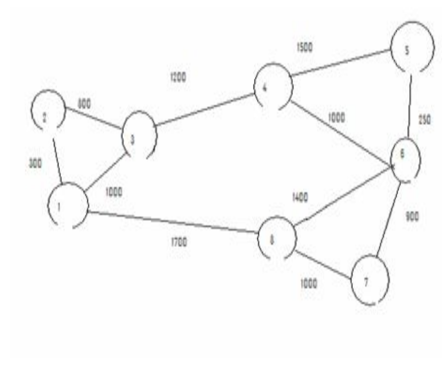
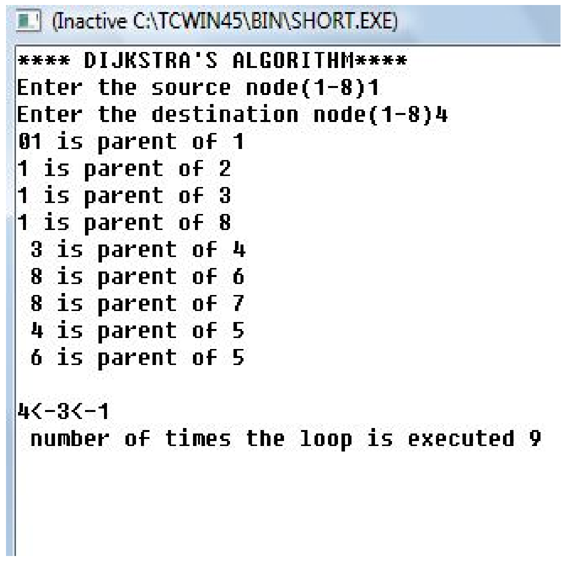


Fig Considered graph

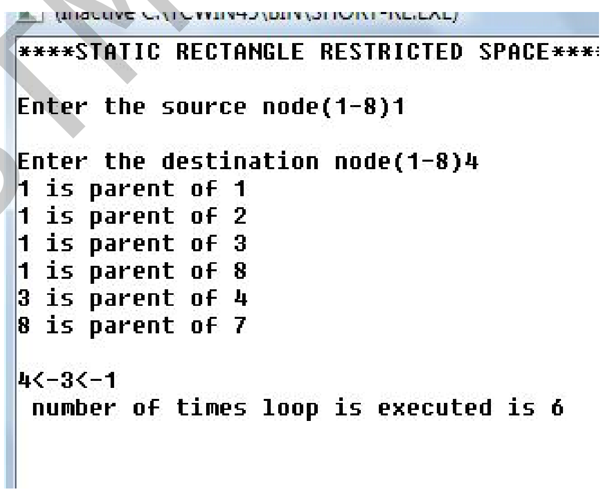
OUTPUT:



```
(Inactive C:\TCWIN45\BIN\SHORT.EXE)
**** DIJKSTRA'S ALGORITHM****
Enter the source node(1-8)1
Enter the destination node(1-8)4
01 is parent of 1
1 is parent of 2
1 is parent of 3
1 is parent of 8
 3 is parent of 4
 8 is parent of 6
 8 is parent of 7
 4 is parent of 5
 6 is parent of 5

4<-3<-1
 number of times the loop is executed 9
```

OUTPUT:



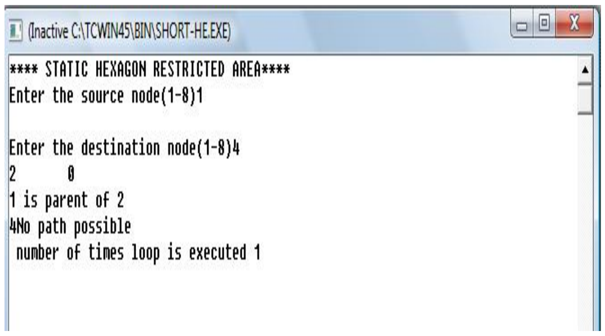
```
****STATIC RECTANGLE RESTRICTED SPACE***

Enter the source node(1-8)1

Enter the destination node(1-8)4
1 is parent of 1
1 is parent of 2
1 is parent of 3
1 is parent of 8
3 is parent of 4
8 is parent of 7

4<-3<-1
 number of times loop is executed is 6
```

OUTPUT:



```
(Inactive C:\TCWIN45\BIN\SHORT-HE.EXE)
**** STATIC HEXAGON RESTRICTED AREA****
Enter the source node(1-8)1

Enter the destination node(1-8)4
2       0
1 is parent of 2
4No path possible
 number of times loop is executed 1
```

Comaprision Table

	EXAMPLE1(8)	GURGAON MAP(51)
DIJKSTRA	9	51
RECTANGLE	6	32
HEXAGON	0	27

TABLE 1: Result Comparisons

DISCUSSION: It is a graphical user interface element which has been developed to be used in conjunction with the mouse pointer. The user however the cursor over an item on the map, without clicking it and an image & brief information will appear in a small hover box. All the major tourist places of Gurgaon are covered in this feature. It consists of moving images of important tourist places of Gurgaon giving a good visual effect to the software application. A slideshow is a display of a series of chosen pictures, which is done for artistic or instructional purposes. This feature allows the user to see the enlarged version of the map for better viewing. Usually maps contains various details which makes it difficult to show all the details in the map but using zoom feature, these can be easily shown.

RESULTS
See different figure for algorithm

Result Analysis

GURGAON MAP(52) ALGORITHMS	EXAMPLE1	EXAMPLE2
SOURCE	Udyog Vihar	Udyog Vihar
DESTINATION	Palam Vihar	ITM University
SIMPLE DIJIKSTRA	55	57
STATIC RECTANGLE	25	25
DYNAMIC RECTANGLE	-	-
STATIC HEXAGON	16	0
DYNAMIC HEXAGON	-	27

TABLE 2: Result Analysis

Applications :
CNS application can be used for investigations, resource management, and asset management. For example, CNS might allow emergency planners to easily calculate shortest distance to reach a destination in the event of a natural disaster. Officials concerned with tourism industries including the private sector, will have an important role in improving and planning the future of their industries using CNS capabilities. Moreover, CNS will enable us to have up-to-date information, recommendation and orientation to tourists to guarantee their security in case of emergency which is a great concern to them. Tourism these days is an important national resource and industry, which all nations are trying to enjoy and exploit. Countries are racing into improving their tourism industries through all its shapes

and faces. Some countries are even going to tourism education as a start, then continue to include all other related aspects and supporting means such as transportation, food, hotel and lodging, natural parks, shopping etc. Some of them are creating new types of tourism (business, medical, relaxation, and education tourism) and others are reviving and improving their assets. In this work we are developing a car navigation system for day-to-day services and improvement purposes in addition to a website that has something to offer to the drivers as a beginning. The CNS algorithm is concentrated in analyzing, improving, monitoring, decision-making and optimal planning. It is a computer system capable of assembling, storing, manipulating, and displaying geographically referenced information, i.e. data is identified. Then, CNS is often defined not for what it is, but for what it can do. In addition to its capabilities in the analysis and it is an interactive software; it allows easy modification of any data, and fast up-to-date informing of the change by posting on the web if needed.

CONCLUSION

The primary goal of this work is to study the three variants of dijkstra's algorithm and implement them in improving the car navigation systems' time complexity. Two algorithms, namely rectangle restricted space and hexagon restricted space have been implemented. Applying the restricted space algorithm reduces the time complexity and is really a boon for any path finding algorithm. This project provided us with an opportunity to analyze and practice all the phases of the Software Development Life Cycle.

REFERENCES

[1] Sara Nazari, M. Reza Meybodi, M.ali salehiGh, Sara taghipour, "An Advanced Algorithm for Finding Shortest Path in Car Navigation System," IEEE/ACS International Conference on Computer Systems and Applications, 2008

[2] Liang Dai,"Fast Shortest Path Algorithm for Road Network and Implementation", Carleton Honours PROJECT,2005.

[3] Syarifah Teh Nadhiah bt Syed Mohamad Nor, "Finding The Minimum Path Using Dijkastra's Algorithm, Universiti Teknologi MARA , 2007.

[4] Aartsen-prof.dr. J. van Leeuwen Copromotor, dr. J.H. Verriet," Route planning Algorithm for Car navigation",I.C.M 2004.