TRANSNAV
http://www.transnav.eu
the International Journal
on Marine Navigation
and Safety of Sea Transportation
Volume 9
Number 2
June 2015

DOI: 10.12716/1001.09.02.17

Analysis of Graph Searching Algorithms for Route Planning in Inland Navigation

W. Kazimierski & A. Sawczak
Marine Technology Ltd., Szczecin, Poland

N. Wawrzyniak
Maritime University of Szczecin, Szczecin, Poland

ABSTRACT: Route planning is one of the core functionalities of modern navigational systems also in inland waters. There is a possibility of at least partial automation of this process with the use of graph searching algorithms. Main problem here is to create a graph based on nautical spatial data. The paper presents research on examining different graph searching methods for inland waters. The concept of using combined approach for vector and raster data is given, followed by research results for raster data.

1 INTRODUCTION

Inland shipping is one of growing branches of transportation. The impact of this field in Europe is growing, as European Union is paying more and more attention to so called sustainable transport. However inland shipping means not only professional transport of goods but also a wide variety of pleasure boat, leisure craft or other recreational ships used for touristic purposes. All of them need technical and software aids for navigation. The market is offering mostly professional equipment like standardized InlandECDIS, which is not covering all needs of recreational tourists. The alternative for this are applications designed for touristic purposes. One of an example might be MOBINAV, described in this paper. The system is being developed under the project LIDER/039/693/L-4/12/NCBR/2013 financed by Polish National Centre for Research and Development (NCBiR).

The paper describes research on automatic route planning for MOBINAV system. The system itself is presented at first with the focus on spatial analysis. Then a concept of route planning in system is given followed by short introduction of graph searching methods in the aspect of route planning. Finally numerical experiment is described and the research results are presented.

2 MOBILE NAVIGATION IN INLAND WATERS

Every branch of shipping, including inland shipping, is mobile from definition, as mobility is a core attribute of navigation. However mobility is especially important in recreational shipping, where smartphones and tablets are the first choice devices.

Apart of professional InlandECDIS a range of flexible applications dedicated for recreational users is available. An example of this can be MOBINAV.

2.1 *MOBINAV*

Mobile inland navigation is a research project that combines issues of cartography, geoinformatics and navigation and focuses on developing of advanced technology for processing and visualizing spatial data. Measurable effect will be a prototype of MOBINAV system, which will be an innovative mobile inland navigation complying with the requirements of recreational users. According to the research hypothesis, implementation of advanced mobile mapping methodology on modern mobile technology will allow fulfillment of recreational sailors.

Thus, MOBINAV is an example of a geoinformatic system dedicated to the needs of a particular audience. Assumptions of the project include not only the development of mobile mapping presentation model, but also an appropriate set of spatial analysis (Zaniewicz et. all, 2014).

2.2 *Spatial analysis in inland waters*

The basis for planning of every GIS system is determination of user needs according to functional requirements. During this process a set of spatial analysis useful for the users has been stated. The set includes following:

- analysis based on attribute selection such as: finding the desired object, safety isobaths, hazardous areas, depth point, etc.;
- analysis based on spatial selection: finding nearby, locating specific kilometer;
- complex selection (combined) – selection combining attribute and spatial selection, e.g. finding the nearest POI in specified category;
- analysis related to the routes, in particular: manual route planning, automatic route calculation, route validation, route's statistics;
- analysis based on buffering: hazardous areas, reported zones, time buffer;
- bearing and distance, which are the basic measurements on the map;
- alarms – complex functions using selected analysis e.g. alarm of crossing dangerous isobaths, alarm of approaching danger, etc.

3 ROUTE PLANNING IN INLAND NAVIGATION

One of the most important functions for users in navigational system is the possibility of route planning. Apart of classical, manual route planning function, an interesting alternative is automatic route planning. The system proposes the route to follow based on input values. The bases for this are depth and dangerous areas.

3.1 *Specificity of inland waters*

Inland waters are specific areas in the aspect of route planning. Part of them has a form of rivers or canals, which in case of route planning are more or less similar to roads. Vessels are following river direction moving up or down the stream. The possibility of crossing traffic is restricted to the areas of junctions and the routes can be easily divided into legs and waypoints. Convenient way of presenting this kind of waters as spatial data is the use of vector data model - topological vector model would be the best.

The other specific areas are lakes. These are larger areas and the vessels have a possibility of moving in many directions. Although there are usually recommended routes, the only restrictions in fact for route planning are safety issues including depth and dangerous areas. In the aspect of spatial data model, lakes can be represented in both ways – with vector data model, but also with raster data model. Both of them have advantages and disadvantages.

3.2 *Vector data approach*

The common approach in route planning which is actually path finding problem (combinatorial optimization) is to use available data as a discrete structures for the purpose of conducting specific search. The goal of such search is to find a sub-structure with a maximum (or minimum) value of some parameter (eg. distance, speed or even comfort of travel). Spatial data for inland waters include layer indicator points of waterways - hectometers, which naturally form a graph structure. Consecutive points are virtually connected with a line of a constant distance value and several other parameters e.g. speed limit or waterway category. The vector data are given in each electronic chart and doesn't need to be post processed to conduct a search. However inland ENCs (Electronic Navigational Charts), besides waterways include other water areas that are available for navigation of leisure craft such as lakes, bays or periodic rivers (often poorly charted). In inland ENC's these areas are not covered by any specific vector data that can form a ready-to-use search structure.

3.3 *Raster data approach*

In MOBINAV one of the given map layer is a raster ortophoto which depicts in detail (depending on an available resolution) all water areas in a given spatial domain. It can't be used directly for a route planning, but it's relatively easy to transform it into a line boundary (by simple image contouring process). Then by determining the size of a regular quadratic mesh (grid) and by overlapping map with it, an undirected graph can be built. Its vertices are represented by grid nodes (or by grid centers) and all created intersection points between water areas boundary and the created mesh (Fig.1).

If the mesh used is regular (quadratic) a created graph will generally have the same length of each edge (if graphs weights are defined as distance, all the fragments of the route equal 1). To simplify the algorithm, one can assume that "boundary" vertices are also connected by edges of such length. Although it is not true in practice, it won't have major impact on finding the optimal solution (fastest route) as long as the defined mesh is relatively dense. It is an example of a so called grid graph, which can be defined as finite, node-induced subgraph of the integer grid. The grid graph is thus represented by a node set completely defined by the set of nodes V and an edge set E, as in equations (1), (2):

$$V = [0...k-1] \times [0...l-1] \tag{1}$$

$$E = \{((i,j),(i,j')) \in V^2 : |j - j'| = 1\} \cup \{((i,j),(i',j)) \in V^2 : |i - i'| = 1\} \quad (2)$$

where k = horizontal number of nodes, l = vertical number of nodes (Mehlhorn and Sanders, 2008).

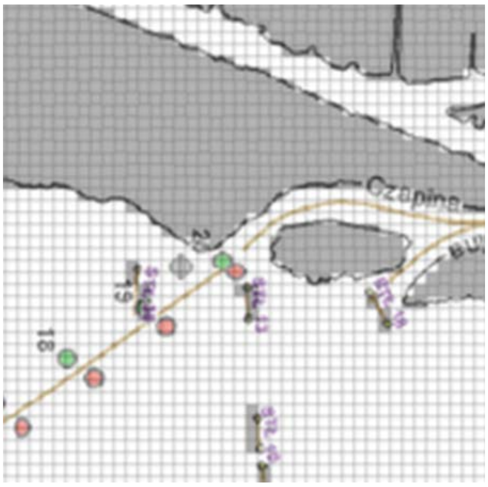

Figure 1. Chart fragment overlaid with exemplary grid.

The ubiquitous undirected uniform-cost grid map is a highly popular method for representing pathfinding environments in many fields, like video games or robotics (Harabor and Grastien, 2011).

3.4 *Combined approach*

Because MOBINAV system uses both kinds of spatial data (vector and raster) coming originally from many sources with different resolution, reliability and validity (Wawrzyniak and Hyla, 2014) it seems reasonable to use the best one available. To be able to do that, one needs to combine both search techniques in order to find the optimal (fastest, easiest or the most pleasurable) route. Such approach requires a few stages:

- Spatial division and data hierarchy assignment
- Raster data processing
- Structure parameters definition
- Graph building
- Graph search

Firstly it's crucial to define which available data will be chosen for route planning from the sources that are currently available for each area. Areas may be divided according to existing spatial data allotment (e.g. chart cells) or concerning the type of water area where the route will be planned (rivers, waterways separately from lakes and bays). Secondly all parts of raster data, if chosen, must be converted into graph substructures as described in 3.3. Then, taking into consideration all the type of data that is going to be used, the weighting parameters of future whole search structure must be defined. The main stage is to build the graph which will be a combination of solutions described in 3.2 and 3.3 depending on a data chosen in first stage of the approach. This task mostly consists of joining the substructures into one graph representing all of the area that is going to be searched. The last and the most important stage is conducting a search which may use many available algorithms widely studied and discussed (Biggs et. all, 1976; Gross and Yellen, 2004).

4 GRAPH SEARCHING ALGORITHMS

Graph searching is an important topic in many areas, particularly in path planning used in computer games and logistics. There are two main approaches: non-heuristic and heuristic algorithms (Cormen et. all, 2009).

4.1 *Non-heuristic algorithms*

Non-heuristic algorithms search graph around starting point until they found goal, always returning an optimal solution. No approximation rules are used, so a lot of operations is required to determine a path, looking for locally optimal solution (Ortega et. all, 2015).

4.1.1 *Breadth First Search*

Breadth First Search is a simple algorithm for searching a graph. It begins with a starting point and searches all adjacent vertices and marking them as visited. In the second stage, vertices reachable at distance of two edges are visited and the process repeats increasing the distance, until ending point is reached. As a result, algorithm outputs a tree, with a root being a starting point, containing all vertices accessible from this point.

4.1.2 *Dijkstra's algorithm*

Dijkstra's algorithm determines the shortest path between starting point and every other vertex in the searching area until it founds a goal. It begins from starting point, then the algorithm visits the closest vertices which are not yet examined. It expands outwards until it reaches the goal. That means high computational complexity of this algorithm but the shortest route will always be found.

4.2 *Heuristic algorithms*

Heuristic is a set of rules helping in finding the best solution. It is a technique used for solving problems quicker than classical methods. Heuristic algorithms find approximate solution, so they can be used when classical methods are failing in finding exact path or have too big computational load. The risk is however that the optimal solution will not be found at all. Thus it can be said that heuristic algorithms approximate optimal solution (Koenig et. all, 2004).

4.2.1 *Best First Search algorithm*

Best First Search algorithm is similar to Dijkstra's algorithm - the difference is in the use of heuristic. It provides estimated information about how far from a goal any vertex is. It chooses the closest point to the goal, instead of choosing vertex closest to the starting point. For example, if a goal is to the north of the starting point, algorithm will be leading the path to the north. Best First Searching works much quicker than Dijkstra's algorithm but determined path doesn't have to be the shortest one.

4.2.2 *A* algorithm*

A* algorithm is the most popular choice for pathfinding because it is quite flexible and can be used in many contexts. It can be treated as a combination of the Dijkstra's and Best First Search algorithms. Algorithm is favoring vertices that are close to the starting point (like in Dijkstra's algorithm) and vertices that are close to the goal (like in Best First Search). By using heuristic, the algorithm searches only a limited number of fields.

Usually A* algorithm is defined by two factors g(n) and h(n) and is resulting in finding least value of the cost function f(n) as in equation (3):

$$f(n) = g(n) + h(n) \tag{3}$$

where:

g(n) - represents exact cost of route from starting point to any vertex n;

h(n) - is a parameter specifying heuristic estimated cost of path from any vertex n to the goal.

4.2.3 *Jump Point Search*

Jump Point Search is an optimized version of the A* algorithm. This algorithm reduces symmetries in the search procedure by using the graph pruning. This action eliminates certain vertices in searching process. As a result, algorithm can consider long „jump" along straight lines: horizontal, vertical and diagonal. That makes the Jump Point Search work faster than A* algorithm (Harabor and Grastien, 2011).

5 NUMERICAL EXPERIMENT

The goal of the research was to examine graph searching algorithms in the scope of implementing them for inland waters. For this purpose a numerical experiment comparing different algorithms in lake environment was proposed.

5.1 *Experiment overview*

The experiment was based on raster chart of Dąbie Lake. It has been modified in order to improve the presentation. The summary cost raster was created by adding raster charts containing information about land, depth, fishing nets and buoyage.

The map has been introduced into on-line pathfinding software available at https://github.com/qiao/PathFinding.js, which allowed testing of algorithms and theirs parameters. On the basis of raster map the grid graph has been created. The approach placing grid nodes in mesh centers has been used. This allowed using of diagonal edges, as well as horizontal and vertical. One scenario has been examined and five presented algorithms have been tested:

- A*
- Dijkstra's
- Best First Search
- Breadth First Search
- Jump Point Search.

For heuristic algorithms Manhattan heuristic has been used.

5.2 *Research scenario*

The research scenario is presented in (Fig. 2).

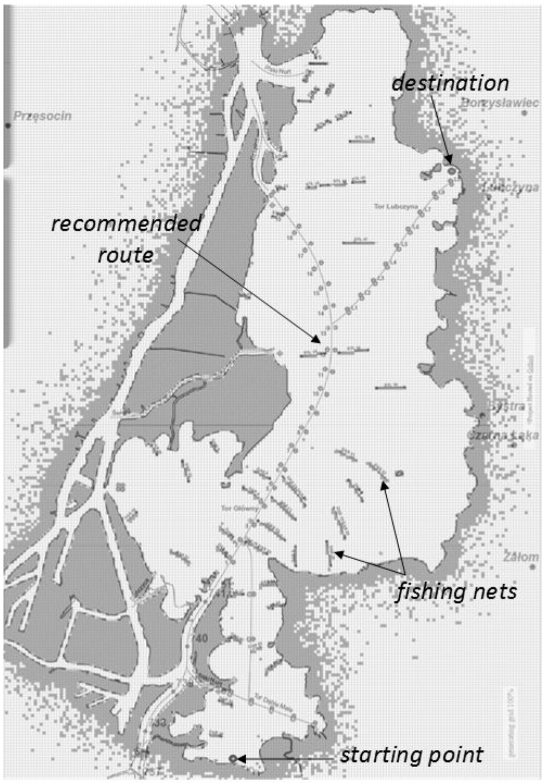

Figure 2. Research scenario.

It assumed finding the shortest path from Szczecin yacht harbor in Small Dąbie Lake to Lubczyna harbor. Created raster is in fact a binary representation of available and not available areas. The land and the obstacles that are on it are marked in dark grey, while available waters are in light grey.

The algorithms have been used to find the best route with the criterion of distance. Thus the route found by algorithm is the shortest path calculated according the algorithm used.

5.3 *Results of experiment*

Results of experiment are presented in two ways – graphically and numerically. Firstly the picture presenting path calculated for each algorithm is given. This allows comparing calculated routes from navigational point of view. Secondly the table with numerical results is presented - distance of calculated route (rounded to 1 meter) and number of operations needed to find the route. This allows comparing effectiveness of algorithms and computational load.

5.3.1 *Graphical comparison of algorithms*

In (Fig. 3) the map of Dąbie Lake with determined by various algorithms paths along the lake is presented. It can be noticed that all of the paths avoid successfully obstacles, however different solutions were chosen.

In case of Dijkstra's algorithm all lake area has been searched for finding optimal solution. After passing Small Dabie Lake, the path follows eastward between leaving the nets on starboard side and that the route goes near to east coast. Route has been determined after 34 246 operations and the length of route is 13 890 m.

Breadth First Search algorithm is also a non-heuristic approach, so the path found by it is similar to Dijkstra's. Major difference is that the center part of the path is going straight across the lake, between the fishing nets. Determined route has a length of 13 890 m at the performance of the 34 713 operation by the algorithm. The distance is the same as in Dijkstra's but the route is much more useful from navigational point of view as it has less way points, so less course alterations.

In case of Best Fit Search, as well as in other heuristic algorithms, the area searched is much smaller than in case of non-heuristic algorithm. This results also in proposed path, as the algorithm is locally driven towards suggested solution. The path itself goes very near western coast passing all fishing nets far on starboard. Then the route is turning to starboard directly towards destination. Determined route has a length of 14 871 m at the performance of 1 180 operations by the algorithm.

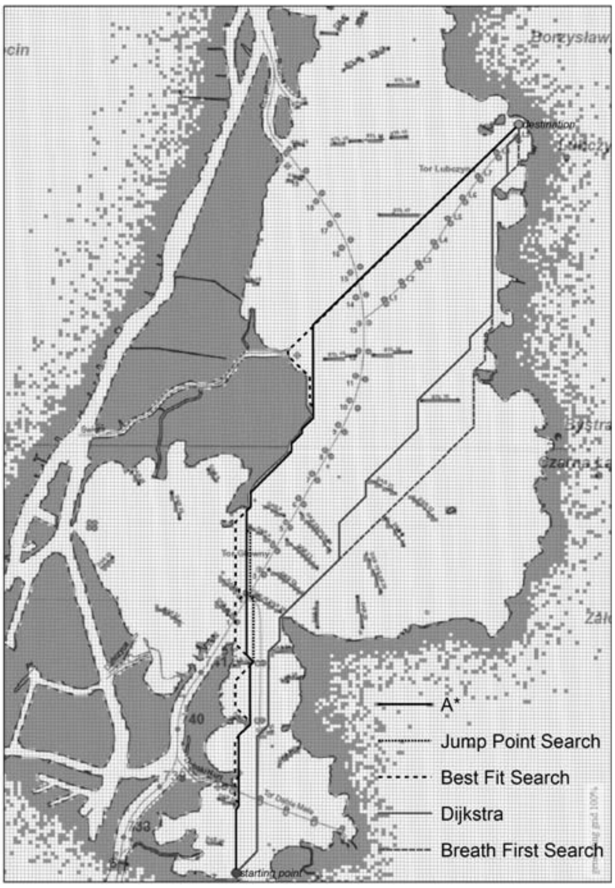

Figure 3. Graphical presentation of results

Results obtained by A* algorithm are almost the same as in case of Best First Search, but it is a bit closer as it moves from the west coast earlier. The route goes close to western coast until it finds clear straight path to destination. Determined route has a length of 13 988 m at the performance of the 1 983 operation by the algorithm.

Jump Point Search algorithm gives also very similar results to other heuristic algorithms and is almost the same as A* path. Only a small part of the route, when leaving Small Dąbie Lake is different. Route has been determined by with the implementation of 7 442 operations. The length of route is 13 988 m.

5.3.2 *Numerical comparison of algorithms*

Graphical comparison of algorithms presented above allows stating some general findings about the results and comparing them based on navigational experience. However some numbers allowing comparison of methods are given below. Table 1 provides a summary of methods performance. Number of operations and the length of the route computed by each algorithm are included. The length of route is given both in pixels and in meters.

Table 1. Summary of the results.

Type of algorithm	Route length [pixels]	[meters]	Number of operations
Non- heuristic algorithms			
Dijkstra's	234,48	13 889	34246
Breadth First Search	234,48	13 889	34713
Heuristic algorithms			
Best First Search	251,05	14 871	1180
A*	236,14	13 988	1983
Jump Point Search	236,14	13 988	7442

In conclusion, the shortest route is given by Dijkstra's and Breadth First Search algorithms. These are both non-heuristic algorithms so they are meant to find optimal solution. The cost however for this is a large computational load - about 35 000 in both cases. The entire area of lake is searched to find the best solution. On the contrary the heuristic algorithms focus only on the very small part of area to be searched. Thanks to the use of heuristics, planned route is determined at a significantly reduced number of performed operations. In this situation however the route is not the shortest path. The fastest search was performed with Best First Search algorithms, but the route was the longest – almost 1 000 meters longer than optimal route. The best compromise seems to be A* algorithm, which performed almost as fast as Best First Search and path distance is reasonably close to optimal one determined by non-heuristic algorithms. The computational load in this situation is 1 983 operations, almost 18 times less than the shortest path algorithms defining and the route is only 100 meters longer. The route has also reasonable number of waypoints. The same distance was obtained by Jump Point Search algorithm but with the use of almost 7 500 operations.

5.4 *Conclusions*

The results presented in the preceding chapter allows stating a few interesting conclusions. Firstly, it can be noticed that in the case present in the paper more than one shortest path is available. Both non-heuristic algorithms obtained the same distance of path, however indicating slightly different route. The common approach for them was however to move eastwards after passing Small Dabie Lake. Different way was chosen by heuristic algorithms. All of them move close to western coast. This was probably the result of dense fishing net location at the entrance to Big Dabie Lake. Instead of moving between the nets and close to eastern shore, the algorithms "decided" to go near the western coast but far from the fishing gears.

Another important conclusion is that non - heuristic algorithms makes much more operation to determine a path than algorithms which use heuristic. In some algorithms it is even more than 30 times more operations. This is a cost of looking for optimal solution. In such a case it seems to be reasonable to use heuristic algorithms for real-time applications and find an acceptable compromise between length of path and computational load.

It can be also noticed that some of the algorithms result in finding more useful paths for navigational purposes. In this case the number of course alteration is important as the sailors do not want to be involved in constant altering the course.

6 SUMMARY

The paper presents research on automated route planning in mobile inland navigation system. The research are based on graph created as grid on raster data. Various graph searching methods were analysed and compared.

The results of experiment lead to a few major findings. Heuristic algorithms, especially A*, seems to be reasonable compromise between length of planned route and computational load for a real-time system. Non-heuristic algorithms, like Dijkstra's can find optimal solution, but their computational load is much bigger. They can be used in systems, which are not tending to work on-line, for example for earlier planning of routes in fixed areas. Other findings relates to kind of data used. Raster data are commonly available for larger areas like lakes, but graph created based on them is much more complicated than typically vector-based graph. Thus, in real environments combined systems can be used. First, some major routes can be found on raster chart and converted to vector and then common graph for all area can be implemented. It would be also important in future works to introduce also additional information to weight in graph (not only distance) for better performance of system.

The problem itself and the results achieved are interesting and the authors will continue works on this subject. In future works other graph parameters should be considered like – construction of grid graph based on mesh nodes, implementing additional information to weights or using other heuristics.

REFERENCES

Biggs Norman L., Lloyd Keith E., Wilson Robin J., Graph theory 1736-1936, Oxford [Eng.] : Clarendon Press, 1976

Cormen T. H., Leiserson C. E., Rivest R. L., Stein C.: Introduction to Algorithms . The MIT Press, third edition, 2009

Gross, Jonathan L.; Yellen, Jay Handbook of graph theory. CRC Press, 2004

Harabor D., Grastien A.: Online Graph Pruning for Pathfinding on Grid Maps . 25th National Conference on Artificial Intelligence. AAAI, 2005

Kazimierski, W., Wawrzyniak, N.: Exchange of Navigational Information between VTS and RIS for Inland Shipping User Needs, in Mikulski J.(ed.) Telematics in the Transport Environment, Book Series: CCIS 471, , Ustron, 2014

Koenig S., Likhachev M., Furcy D., Planning A*, Artificial Intelligence, 155(1-2): 93-146, 2004

Mehlhorn K., Sanders P.: Data Structures and Algorithms: The Basic Toolbox, Springer Verlag, Berlin Heidelberg, 2008

Ortega-Arranz H., Llanos Diego R., Gonzalez-Escribano Arturo, The Shortest-Path Problem: Analysis and Comparison of Methods, Morgan & Claypool, 2015

Wawrzyniak, N., Hyla, T.: Managing Depth Information Uncertainty in Inland Mobile Navigation Systems. Book Editor(s): Kryszkiewicz et al., Joint Rough Set Symposium, LNAI, pp. 343-350, Granada-Madrit, 2014

Zaniewicz G., Włodarczyk-Sielicka M., Kazimierski W., Problems of integration of spatial data from various sources in inland mobile navigation, Annals of Geomatics vol. XII. 3(65), Warsaw, 2014 (in polish)