Airport distribution efficiency in an undirected graph.

Keertan Dakarapu - kxd160830

Abstract

Public services are always pushed to improve their efficiency. For location based services, such as air transport, its productivity greatly depends on efficient distribution of its resources. The placement of airports must be decided after great analysis as it determines the survival and business model of the system. We formulate a deterministic algorithm to find the most efficient number of the airports to build along with its distribution. we rely on A star algorithm as a base model, manipulating the heuristic function and cost function accordingly. We use this algorithm recursively for all possible traversals in the graph and get a cumulative cost. We generate a cost matrix and select the most efficient arraignment. We vary and experiment around the parameters and analyse the results.

1. Introduction

Heuristic search increases the intelligence factor of searching by which drastically reduces our search space. It is particularly helpful in large search spaces in finding the efficient path of traversals. We can use its concept to find the most efficient distribution of airports in given graph considering the aggregate costs of traversal for all paths via the most efficient path.

Let us assume that we have an undirected weighted graph with weight being the distance factor, and all the straight-line distances from each node to another. We must find the most efficient number of airports, and their most efficient placement, given static parameters such as the cost of travel by air, cost of travel by road, and cost of building an airport, speed of travel by air, speed of travel by road.

The solution requires us to use A-Star algorithm recursively. We first create our heuristic function, based on cost values provided. Then we iterate upon the number of airports to be build starting from 2, and all the way till n/2, where n is the number of nodes. We calculate and compare the sum of least cost incurred by all possible paths for travelling, following the most efficient placement of airports in each respective case. Then we decide upon the no of airports comparing the least cost estimation or each case. Internally, we should use A-star for determining the shortest path between two nodes, where in we must consider the air routes also.

2. Related Work

i. Optimization of airport ground operations integrating genetic and dynamic flow management algorithms – *García Jesús, Berlanga Antonio and co.*
ii. Empirical analysis of airport network and critical airports – *Wei Cong, Minghua Hu and co.*

3. Approach

To find the most efficient number of airports for a given graph, we consider all the possible number of airports (say *Airport_count*) starting from a minimum of two airports to N-1, where N is the number of nodes in the graph. We generate all the possible combinations of airport locations for each airport_count. We find the aggregate cost for each arrangement, which is attained by the summation of individual cost of all the possible n^2 paths in the graph with respect to the arrangement. For each airport_count, we find the arrangement with the least aggregate cost and store it in an array. After we find the respective

least cost arrangements for each airport_count, we return the most minimum of these as the most efficient solution with the airports.

$$Airport\_count = 2, 3, \ldots, N-1$$

We know that if a path from one point to another is the shortest path between them, then any two points on the path must have their shortest path as a subset of the original path. Using the above axiom, to find the shortest path to a goal node, we must proceed towards the node which is nearest to the goal node. The meaning of nearest is a relative term and is domain specific. Thus, we define a function *H*, called heuristic, which evaluates the value pertaining to the quantitative measure of nearness to the goal state. Searching the search space by estimating the next best step as above is called Heuristic search. We follow A star algorithms approach of using a combination of a heuristic function *H*, and a cost function *G*, which gives us the cost of path from current position to the desired node.

$$\boldsymbol{F = H + G}$$

There are many generic heuristic functions for finding the most efficient path in a graph. They could be very simple or complicated. The typical A star algorithm uses a simple heuristic of straight line distance from the goal state. But we cannot consider only the straight-line distance as a measure of 'nearness to goal' as there are many factors such as availability of an airport at a node in search space and traffic from the current node to that node, which cause a cost difference when G is calculated. In the heuristic function, we must consider the traffic flow between the nodes to approximate the airfare better. We, introduce $TrafficWeight_{XY}$ which is a constant derived by scaling the traffic flow between node X and Y to a number, between 0 and 1, with respect to all the NxN traffic flows in the graph.

$TrafficWeight_{xy}$ = 0, then the traffic is 0 or least.

$TrafficWeight_{xy}$ = 1, then traffic between x and y is very high, or maximum.

Note: *Traffic is a directional property, hence, $TrafficWeight_{xy}$ is not equal to $TrafficWeight_{yx}$.*

$$\boldsymbol{H} = \frac{\left(\frac{S_{fg}}{TrafficWeight_{fg}}\right)}{Avg\_node\_weight_g}$$

Every heuristic algorithm takes the initial position and the goal state as inputs and tries to find the next best step by finding the step with the least F value.

$$Avg\_node\_weight_g = \frac{1}{N}\sum_{x=1}^{N} TrafficWeightgx$$

$$\boldsymbol{G} = \begin{cases} \frac{S_{fg} * (\, aircost \,/ TrafficWeight_{fg})}{airspeed}, & if\ airpath\ exists \\ S_{fg} * \left(\frac{roadcost}{roadspeed}\right), & if\ only\ road\ path\ exists \end{cases}$$

where,

S_{fg} = Straight line distance from f to g
$TrafficWeight_{fg}$ = Scaled traffic volume from f to g
Airspeed = Speed via air route.
Aircost = Average unit base fare for air travel.
Roadspeed = Speed via road route.
Roadcost = Average unit cost of road travel

Notice that cost of air traversal is not a constant but is a function of traffic weight and base air fare. The fare could be more if less traffic is less and it could significantly decrease if high traffic weight is given.

Final cost =

$$\left(\sum_{N\times N} pathcost\right) + Airport\_count \times Airport\_cost$$

The final cost of the arrangement would be the summation of all NxN path costs and the cost of building the airport_count number of airports.

4. Implementation

We would be using the following directed graph of 10 nodes for the implementation. (figure 1.2) The graph is weighted with the distance factor.

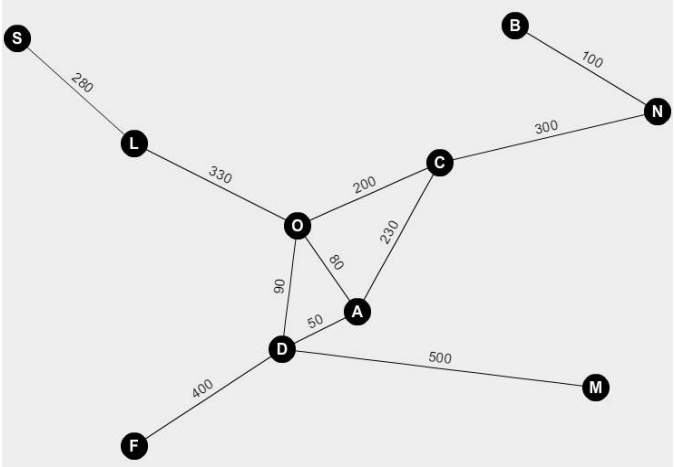

Figure 1.1: The weighted graph considered for implementation.

We are given an NxN Array matrix of straight line distances from each other. We convert the given graph into appropriate representation, in our case a python List. We take the names of the nodes as a python set. We initialise all the required parameters and set them as global variables for accessing them in various functions.

We define a basic driver function to check which value of *Airport_count* gives us the most minimal aggregate cost. Internally the flow of program goes from this main function to the *get_min_cost* function where, calling the *get_all_placements* function returns all the possible airport arrangements for the given count, thereby allowing us to compare the aggregate cost for traversal by calling the *get_cost* function for each combination.

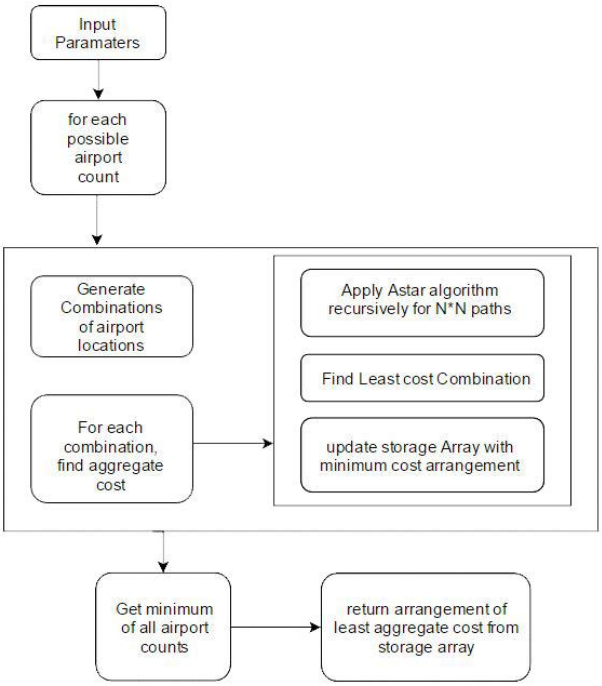

Figure 1.2: Software Architecture Diagram

Get_cost - function internals

```
for i in range(0, N):
    for j in range(i, N):
        cost = cost +Astar(i, j, placement)
cost = (cost) + (count * airport_cost)
```

It in turn calls the *Astar* function recursively over all possible paths and returns us the aggregate cost. The control finally comes back to the original driver function, which returns the solution.

5. Experiments and results

The Table 1.1 gives us the rendered results of the algorithm with average traffic weight of 0.5 for each node and varying other parameters. The top portion of the table is considering the cost function type 1, which considers time and cost as equal factors. The bottom part of the table considers the cost function type 2, which only looks at the cost

Table 1.1: 'Air_cost' parameter and the reflected change in cost efficiency with average weight of 0.5 for each path.

Cost function type	Weight type	Air_cost	Airspeed	Airport_cost	Roadspeed	Roadcost	No of Airports	% cost efficiency increase from Roadways
1	0.5 for all	100	1000	3000	80	100	5	18.92%
1	0.5 for all	200	1000	3000	80	100	5	16.43%
1	0.5 for all	300	1000	3000	80	100	5	13.94%
1	0.5 for all	400	1000	3000	80	100	5	11.46%
1	0.5 for all	500	1000	3000	80	100	4	8.15%
1	0.5 for all	600	1000	3000	80	100	4	6.05%
1	0.5 for all	700	1000	3000	80	100	5	2.53%
1	0.5 for all	800	1000	3000	80	100	4	0.53%
2	0.5 for all	100	1000	3000	80	100	6	8.01%
2	0.5 for all	200	1000	3000	80	100	3	5.81%
2	0.5 for all	300	1000	3000	80	100	3	4.52%
2	0.5 for all	400	1000	3000	80	100	3	3.23%
2	0.5 for all	500	1000	3000	80	100	3	1.94%
2	0.5 for all	600	1000	3000	80	100	3	0.65%
2	0.5 for all	700	1000	3000	80	100	3	-0.63%
2	0.5 for all	800	1000	3000	80	100	3	-1.92%

Table 1.2: $TrafficWeight_{Dallas,x}$ is changed and the reflected change in airport distribution and cost efficiency with average weight of 0.5 for other nodes.

Weight for others	Experimental weights	Air_cost	No. of Airports	Airport Locations	% cost efficiency increase from Roadways
0.5	0.5 for Dallas	500	4	Dallas, Las vegas, Miami, NewYork	8.15%
0.5	0.4 for dallas	500	5	Austin, Florida, Las vegas, Miami, New York	7.44%
0.5	0.3 for Dallas	500	5	Austin, Florida, Las vegas, Miami, New York	9.98%
0.5	0.8 for Oklahoma	500	5	Chicago, Florida, Las vegas, Miami, New York	1.19%

without considering time. It is evident that if time is not considered, then road ways always have better cost efficiency.

Let us begin with a standard average weight of 0.5 for all the nodes, and look at the airport distribution. We see that there is an airport located at Dallas. Let us change the weight of Dallas to 0.4 and look at the result.

We might predict that since the traffic to Dallas is lower than surrounding nodes yet, Dallas was a part of the most efficient arrangement, so, the airport location would mostly be shifted to some nearby node. If you look at Table 1.2, we can see that the number of airports increased and the airport location shifted from Dallas to Austin, and a new airport came up at Florida.

Let us change the weight of Dallas to 0.3. If you look at the table, you would notice that the airport count as well as the airport locations did not change but the percentage efficiency has increased from 2% from the above case. This could be because, in the prior case, we had traffic to Dallas, but no airport, hence the increased cost. But in the latter case, we further reduced the traffic to Dallas, and hence reduced the cost increase.

Let us do the reverse process by increasing the weight of Oklahoma to 0.8. We notice that Oklahoma was not a part of original airport locations, but we may predict that this change might include Oklahoma as an airport location. But the results have been different. Interestingly Chicago showed up as a new airport location replacing Austin. Chicago is connected to both Austin and Oklahoma, and hence the increased weight of traffic at Oklahoma shifted the airport from Austin to Chicago balancing the traffic in both routes.

6. Conclusion

By analysing the percentage of cost improvement with changes to the parameters, we conclude that efficiency is not a simple function to estimate humanly but an efficiency algorithm can give us a solution. Our experiments show us the working of our algorithm and how it interdepends on various factors. we cannot account for all the factors in real-life situations and neither are all of them quantitative. Hence, we get a probabilistic model. This does not mean that our solution is incorrect, but the cases with very little increase in cost must be closely re-evaluated with more factors. Yet practically, our solution set is the distribution of the airports which is mostly correct, but the increase in cost efficiency is a real-time variable.

7. References

1. http://ai.stanford.edu/~nilsson/OnlinePubs-Nils/PublishedPapers/astar.pdf
2. http://paper.ijcsns.org/07_book/201101/20110119.pdf
3. http://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1128&context=csetechreports
4. Artificial Intelligence: A Modern Approach (Text Book)- Dr. Peter Norvig.