Marketing Simulation – Reducing the Transport Cost for Commercial Agents

BUCEA-MANEA-TONIS, Rocsana
Spiru Haret University
rocsanamanea.mk@spiruharet.ro

BUCEA-MANEA-TONIS, Radu
http://mysqlbrowser.codeplex.com/
radub m@yahoo.com

Abstract

The article is drafted from the perspective of the marketing manager who must minimize carrying costs of a salesperson that share multiple products. We have tackled this simulation problem using a Hamiltonian cycle associated to transport network conceptual reduced to a graph. Identification of optimal solution was achieved through the implementation of Branch and Bound heuristic algorithm in the C++ programming language. The article is developed on two directions: a conceptual approach to clarify the algorithm used and its source code implementation.

Keywords: *critical path, optimal transport problems, Branch and Bound algorithm, Hamiltonian cycle, marketing simulation*

ACM/AMS Classification: C610, M310

1. Introduction

One of the major problems faced by marketing managers is that of transportation of goods. Because carriers ask the round-trip payment, some companies decide to transport the goods through their commercial agents. To determine the route with the lowest cost we use the Branch and Bound method for finding optimal solutions. A branch-and-bound algorithm searches the entire space of candidate solutions, throwing out large parts of the search space by using previous estimates on the quantity being optimized.

2. General description

Assuming a graph G = (V, A) consisting of edges and nodes. On the set of edges is defined a function d which identify the lowest cost $d : A \to Q+$. A

graph can be represented as an $n \times n$ square matrix, where $n = |V|$, having the d_{ij} elements made of the weights associated to the edges and taking values in the set of $Q+\cup\{\infty\}$. For any element d_{vv} with $v \in V$, the relation $d_{vv} = \infty$ [1] is true. Given the costs matrix associated with the graph, where $d_{ij} = d_{ji}, \forall i, j$, we say that the transport problem is symmetric[2] .

To identify the optimum in transport problems using decision trees, it gives an objective function L to be minimized in relation to a set of solutions S, corresponding to the optimal browsing of the graph. Heuristic algorithm implies browsing the possible recursive routes, testing in relation to the simple average of the cheapest pairs of adjacent edges related links to all nodes in the graph, according to: $C = \frac{1}{2} \sum \forall v \in V$

If $L(S)<C$ then you need to explore other possible solutions and identifying the optimal way. In the following example is given a weighted graph (fig. 1) and wishes to establish average cost for any chosen route.

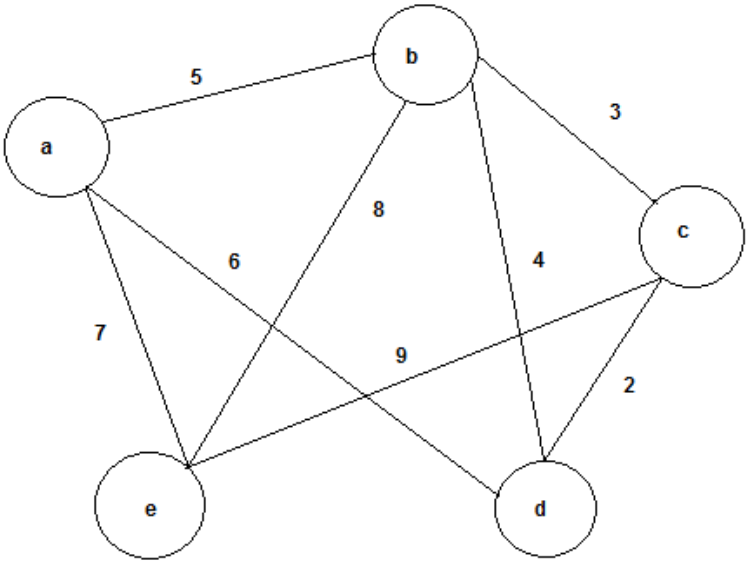

Figure 1: Weighted graph with five nodes

Table 1 shows that the average cost associated with any possible routes is 42/2, this means 21[3]. We will further test the possible routes in relation to this average cost. It results that the number of weighted edges taken into account is of $2^{2n(n \ominus 1)}$ order, so the degree of complexity of binary trees analysis is $O(2^{2n^2} + (n-1)!)$, according to[1] .

3. **Case study**

A courier has to deliver many packages in four cities. It is desirable to optimize transportation costs so the salesperson can visit every town once

Table 1: The cheapest two adjacent edges

Node	The cheapest two adjacent edges	Total cost
a	(a,b), (a,d)	11
b	(b,c), (b,d)	7
c	(c,b), (c,d)	5
d	(d,c), (d,b)	6
e	(e,b), (e,a)	13
-	-	$Total = 42$

on the shortest route. The cities and possible routes between them can be treated as a weighted graph. Associated transport costs are listed in the cost matrix. The situation comes down to a classic transportation problem, the optimal route forming a Hamiltonian graph which will be identified using the algorithm presented above.

Assume the following cost matrix – array of roads – relating to the transport problem for the four cities where the symbol ∞ it is the lack of links between the cities to be visited.

$$D = \begin{pmatrix} \infty & 6 & 8 & 5 \\ 10 & \infty & 4 & 9 \\ 9 & 3 & \infty & 2 \\ 7 & 7 & 5 & \infty \end{pmatrix}$$

A first element of the solution of the transport problem is to define an objective function that allows calculating the reduced cost matrix. Low cost matrix gives the additional costs associated with identifying the link between the two cities for which the cost is minimal. This minimal cost results from the summation of the lower cost of departure from each town, plus the lowest entry cost in every city. Optimal function returns the minimum cost obtained by running a subset of d_{ij}, for any route must aim at entry and exit of each city exactly once. First, it reduces the line 1. The smallest value of line 1 is the minimum cost to leave the city 1. Line 1 is reduced with its smallest value, in this case 5, reducing it in every element of the line. The smallest value (5) is added at the lower limit. After cutting the line 1, we keep in mind the cost 5 and get the following matrix:

$$D = \begin{pmatrix} \infty & 1 & 3 & 0 \\ 10 & \infty & 4 & 9 \\ 9 & 3 & \infty & 2 \\ 7 & 7 & 5 & \infty \end{pmatrix}$$

Then, we cut the line 2 with its smallest value, 4 in this case, subtracting 4 from each element of the line 2. We keep in mind the cost 4 and get the following matrix:

$$D = \begin{pmatrix} \infty & 1 & 3 & 0 \\ 6 & \infty & 0 & 5 \\ 9 & 3 & \infty & 2 \\ 7 & 7 & 5 & \infty \end{pmatrix}$$

Then, we cut the line 3 with its smallest value, 2 in this case, subtracting 2 from each element of the line 3. We keep in mind the cost 2 and get the following matrix:

$$D = \begin{pmatrix} \infty & 1 & 3 & 0 \\ 6 & \infty & 0 & 5 \\ 7 & 1 & \infty & 0 \\ 7 & 7 & 5 & \infty \end{pmatrix}$$

Then, we cut the line 4 with its smallest value, 5 in this case, subtracting 5 from each element of the line 4. We keep in mind the cost 5. Minimum intermediate cost is $5 + 4 + 2 + 5 = 16$, and the resulting reduced matrix is:

$$D = \begin{pmatrix} \infty & 1 & 3 & 0 \\ 6 & \infty & 0 & 5 \\ 7 & 1 & \infty & 0 \\ 2 & 2 & 0 & \infty \end{pmatrix}$$

Reducing rows takes into account only the cost of leaving each city. Reduction columns include the lowest cost to get in every city. Reduction of columns is similar to reduction of lines. A column is reduced with its smallest value, by subtracting this value from each element of the column. The smallest value of the first column is 2, so we subtract 2 from each element of the first column. We keep in mind the cost 2, which is added to the costs calculated above $16 + 2 = 18$. We get the following matrix:

$$D = \begin{pmatrix} \infty & 1 & 3 & 0 \\ 4 & \infty & 0 & 5 \\ 5 & 1 & \infty & 0 \\ 0 & 2 & 0 & \infty \end{pmatrix}$$

Then we reduce the column 2 with its smallest value, 1 in this case, subtracting 1 from each element of the column 2. We keep in mind the cost 1, which is added to the costs calculated above $18 + 1 = 19$. We get the following matrix:

$$D = \begin{pmatrix} \infty & 0 & 3 & 0 \\ 4 & \infty & 0 & 5 \\ 5 & 0 & \infty & 0 \\ 0 & 1 & 0 & \infty \end{pmatrix}$$

The remaining columns are already reduced, because they already contain a 0. Therefore we obtain the optimal route of transportation – the cost being 19 currency units – it remains to determine the order of cities.

4. **Source code implementation**

Program code associated with reducing the cost matrix rows and columns below was procedural, structured as follows:

Stage 1 . Initializing array values of roads:

```
for(int i=1;i<=n;i++)
for(int j=1;j<=n;j++)
{
printf("d[%d][%d]=",i,j);
scanf("%d",&c);
d[i][j]=c;
}
```

Stage 2. Determination of the minimum link costs per rows:

```
for(int i=1;i<=n;i++)
{
min[i]=d[i][1];
for(int j=2;j<=n;j++)
{
if (d[i][j]<min[i]) min[i]=d[i][j];
}
}
```

Stage 3. Reduced matrix per rows after minimum values obtained in the previous step:

```
for(int i=1;i<=n;i++)
for(int j=1;j<=n;j++)
d[i][j]-=min[i];
```

Stage 4. Testing dij pairs against null in order to reduce the matrix on columns:

```
for(int i=1;i<=n;i++)
for(int j=1;j<=n;j++)
if (d[j][i]==0) break;
else if (j==n)
col[++k]=i;
```

Stage 5. Reducing matrix on columns after determining the minimum link cost:

```
for(int l=1;l<=k;l++)

min[l]=d[1][col[l]];
for(int i=2;i<=n;i++)
if (d[i][col[l]]<min[l]) min[l]=d[i][col[l]];

for(int l=1;l<=k;l++)
for(int i=1;i<=n;i++)
d[i][col[l]]-=min[l];
```

Step 6. Display the final result:

```
for(int i=1;i<=n;i++)
{
printf("|");
for(int j=1;j<=n;j++)
{
printf("%d ",d[i][j]);
if (j==n) printf("|\n");
} }
```

where:
n - matrix dimension
d[10][10] - matrix vector
c - the cost beetwen two nodes
min[10] - minimum values per rows/columns
col[10] - reduced matrix columns that doesn't contain any null value
k=0 - index variable.

The result of the program execution for a cost matrix

$$C = \begin{pmatrix} \infty & 2 & 5 & 1 \\ 4 & \infty & 9 & 4 \\ 2 & 8 & \infty & 9 \\ 2 & 3 & 6 & \infty \end{pmatrix}$$

is shown in Figure 2:

Figure 2: The result of Hamilton.exe

We observe that it is advisable to avoid the inclusion of the links that have different values from zero in the reduced matrix. Basically, this will remove

the redundant links between cities, once chosen the optimal in-out route of the city. There are 7 records containing 0 in the reduced matrix above. It starts with the link (2,3) having minimum weight 0. If the route from the second city to the third is included in the optimal solution, then the rest of the links between them can be deleted, because we leave the second city once and we will enter once into the third one. We get the following matrix:

$$D = \begin{pmatrix} \infty & 0 & \infty & 0 \\ \infty & \infty & \infty & \infty \\ 5 & 0 & \infty & 0 \\ 0 & 1 & \infty & \infty \end{pmatrix}$$

Once removed pairs d_{2j} and d_{i3} , $\forall i, j$ we can also eliminate the possibility of circular link (3,2) or d_{32}, as follows:

$$D = \begin{pmatrix} \infty & 0 & \infty & 0 \\ \infty & \infty & \infty & \infty \\ 5 & \infty & \infty & 0 \\ 0 & 1 & \infty & \infty \end{pmatrix}$$

There is only one optimal route for departure from town 3 and entering town 4 so we identify the optimal route for exit from the fourth city. This is determined only by the link d_{41} so remains only to eliminate all unefficient pairs d_{4j} and $d_{i4}, \forall i, j$ coresponding for leaving the fourth city and entering the first.

$$D = \begin{pmatrix} \infty & 0 & \infty & 0 \\ \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 \\ \infty & \infty & \infty & \infty \end{pmatrix}$$

We obtain the optimal path so the company will go through four towns in the following order: City 1 → City 2 → City 3 → City 4 → City 1.

Conclusions

This paper provides the marketing manager a methodology and necessary tools to reduce transport costs for carried goods. This solution addressed a classic transport problem, defining an optimal function based on reducing the distance between two places and a restriction which prevents multiple transports through the same city.

References

1. Leo Liberti, *Branch-and-Bound for the Travelling Salesman Problem*, "Teaching", http://www.lix.polytechnique.fr/ liberti/teaching/dix/inf431-11/bb-notes.pdf, 2011.
2. Katta G. Murtyi, *Branch and Bound Examples*, "Case studies of realistic applications of optimum decision making", http://www-personal.umich.edu/ murty/614/614slides5.pdf , 2011.
3. Allan Borodin, Ran El-Yaniv, *ONLINE COMPUTATION AND COMPETITIVE ANALYSIS*, "Cambridge University Press", UK, 2005.