REPORT OF DISSERTATION WORK

ON

A HEURISTIC ALGORITHM FOR NEWSPAPER DISTRIBUTION PROBLEM IN A CITY NETWORK

By

DIBYENDU CHAKRABARTI

MTC9612

Under the supervision of

Dr. BIMAL KUMAR ROY

Applied Statistics Unit

INDIAN STATISTICAL INSTITUTE, CALCUTTA

Year 1997-'98

Acknowledgements

I would like to express my sincere thanks to my guide **Dr. Bimal Kumar Roy** for suggesting me to work in this interesting field. I am grateful for the lot of suggestions and guidance he provided throughout the work. I would also like to thank **Prof. Nilotpal Chakrabarti** of IIM , Calcutta for helping me a lot. **Dr. Palash Sarkar** helped me with many suggestions from time to time. Thanks to my batch mates who also provided help in one form or the other.

Certificate of Approval

This is to certify that the dissertation work entitled **A HEURISTIC ALGORITHM FOR NEWSPAPER DISTRIBUTION PROBLEM IN A CITY NETWORK** submitted by Dibyendu Chakrabarti , in partial fulfilment of the requirements for M.Tech. in *Computer Science* degree of the *Indian Statistical Institute , Calcutta* , is an acceptable work for the award of the degree .

Date : July 29,1998.

(Supervisor)

(External Examiner)

Index

A HEURISTIC ALGORITHM FOR NEWSPAPER DISTRIBUTION PROBLEM IN A CITY NETWORK

ABSTRACT: The problem of constructing an approximately optimal schedule of vehicles for newspaper distribution of a city is considered here. The newspapers are assumed to be dumped at certain dumping points of the city . The requirement is to minimise the total duration for the completion of distribution of the newspapers. The time consumed for the last trip(back to the newspaper office from the last dumping point of any route) is excluded. The optimisation problem of minimisation of total duration is shown to be NP-complete. A heuristic algorithm has been developed to solve the problem.

1.INTRODUCTION: A large area for applications of combinatorial techniques is the area of urban services. Combinatorics has been applied to problems involving the location and staffing of fire and police stations , design of rapid transit systems, assignment of shifts for municipal workers , routing of street sweeping and snow removal vehicles etc. [Refer Beltrami [1] and Helly [4] for a variety of examples.

The problem of newspaper distribution in a city where the newspapers are assumed to be dumped at certain dumping points is slightly different from the problems mentioned above. We are supplied with the map of the city, requirement of each of the dumping points ,time consumption in moving from one dumping point to the other and the vehicle capacity. The resource in terms of the number of vehicles is extremely limited. The requirement is to minimise the total duration of the distribution process, where the last trip(back to the newspaper office from the last dumping point on any route) is excluded from our consideration.

2.DESCRIPTION OF THE PROBLEM : Given the map of a city, requirement of each of the dumping points , time consumption for reaching one point from another and the vehicle capacity, a schedule of vehicles is to be determined such that the total distribution of the newspapers is completed in several trips of vehicles each of which dumps some amount of newspapers without exceeding the vehicle capacity.

An optimal schedule is one that needs the minimum time to complete the distribution process.

The following notations are used :

We consider a graph G(V,E,R,T) where

$V = \{i_1 , i_2 , \ldots\ldots\ldots\ldots i_n \}$ is a set of nodes representing dumping points. Node i_1 is the newspaper office.

E = The set of edges.

R = $(r_1, r_2, \ldots\ldots r_n)$ is a vector of non-negative weights associated with the nodes. It denotes the requirements at different dumping points.

T = $(t_{ij})$is a matrix of time units associated with the edges.

Here, T is symmetrical.

r_j = Requirement of dumping point at location j.

X = Number of delivery vehicles.

W = Capacity of a single delivery vehicle.

t_{ij} = Time taken to reach from dumping point i to dumping point j.

3.THE CLASSES NP HARD AND NP COMPLETE: In measuring the complexity of an algorithm ,we will use the input length as the parameter . An algorithm A is of polynomial complexity if ∃ a polynomial P( ) such that the computing time of A is O(P(n)) ∀ input of size n .

Definition 3.1 P is the set of all decision problems solvable by a deterministic algorithm in polynomial time . NP is the set of all decision problems solvable by a non-deterministic algorithm in polynomial time.

For a problem to be in NP , we simply require that if x is a yes instance of the problem , then ∃ a concise (i.e. of length bounded by a polynomial in the size of x) certificate for x , which can be checked in polynomial time for validity .

We are now ready to define NP -hard and NP-complete classes of problems. At first , we define the notion of reducibility :

Definition 3.2 Let A_1 and A_2 be recognition (i.e. YES-NO) problems. We say that A_1 reduces in polynomial time to A_2 (denoted as $A_1 \propto^{P} A_2$)iff ∃ a polynomial time algorithm a_1 for A_1 that uses several times as a subroutine at unit cost a (hypothetical) algorithm a_2 for A_2. We call a_1 a polynomial time reduction of A_1 to A_2.

proposition 3.2 If A_1 polynomially reduces to A_2 and there is a polynomial time algorithm for A_2, then there is a polynomial time algorithm for A_1.

Definition 3.3 We say that a recognition problem A_1 polynomially transforms to another recognition problem A_2 if, given any string x, we can construct a string y within polynomial (in $|x|$) time such that x is a YES instance of A_1 iff y is a yes instance of A_2.

Definition 3.4 A problem A is NP-hard iff $\forall\ A' \in NP$, $A' \propto^p A$.

A problem is NP-complete iff A is NP-hard and $A \in NP$.

By proposition 3.1, if a problem A is NP-complete, then it has a formidable property : If there is an efficient algorithm for A, then there is an efficient algorithm for every problem in NP.

Definition 3.5 Two problems A_1 and A_2 are said to be polynomially equivalent iff $A_1 \propto^p A_2$ and $A_2 \propto^p A_1$.

4.COMPUTATIONAL COMPLEXITY OF NEWS : In an attempt to devise an efficient algorithm which gives an optimal solution to NEWS, it is found to be very difficult. So the computational complexity of the problem is explored and the following result is obtained :

The decision version of NEWS is NP-complete

<u>Decision version of NEWS($NEWS_d$) :</u> Given a network G(V,E,R,T), X number of vehicles each with common capacity W, r_j being the requirement at dumping point j, T being the time matrix, does there exist a routing schedule of duration $<$ T ?

PROOF:(I) First, we will show that $NEWS_d$ belongs to NP.

The following non-deterministic algorithm solves the problem in polynomial time :

Step#1 : Guess any path $P = (i_1 = N, \ldots\ldots\ldots, i_k = N')$

Step#2 : Assign r' values to the path non-deterministically such

that $r'(v) = 0$ if $v \in P$.

$0 \leq r'(v) \leq r(v) \quad \forall v \in V$.

$r'(v) \leq W$

Step#3 : $\forall$ v in V, do $r(v) = r(v) - r'(v)$

Step#4 : Include <V,P ,r', T> in the schedule

Step#5 : If $r(v) > 0$ for some $v \in V$,

then GOTO *Step#1*

Step#6 : If total duration of the schedule $\leq$ T,

then output "YES"

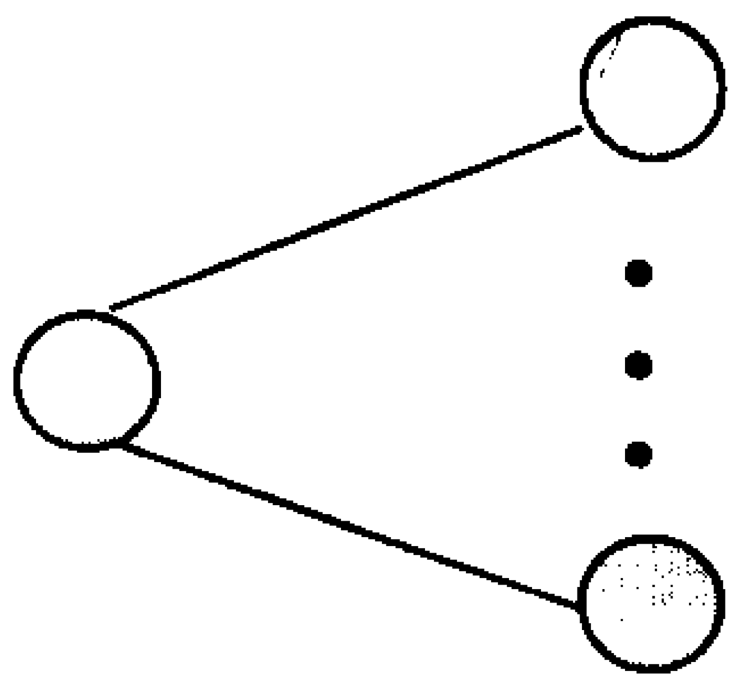
(II) Next, we will show that $NEWS_d$ is NP-hard.

For this part, we will show that PARTITION problem, which is already proved to be NP-complete(See Garey and Johnson[3]) reduces to $NEWS_d$ in polynomial time.

The PARTITION problem is defined as :

Given a set $S = \{a_1, \ldots\ldots\ldots, a_n\}$ of n integers such that $\sum_{i=1}^{n} a_i = 2Y$, Y being an integer. Does there exist a set $R \subset S$ such that $\sum_{a_i \in R} a_i = \sum_{a_i \notin R} a_i = Y$?

We will produce an instance of $NEWS_d$ from an instance of PARTITION in linear time as follows : Let's consider the following graph:



(Each edge is of unit weight)

$r(i_1) = 0$

$r(i_j) = a_j$

Capacity = Y

The question is :

"Does there exist a schedule of duration $\leq 2(n-1)$?"

The equivalence between the two instances is shown regarding the answers.

Let the PARTITION instance has an "YES" answer. So, there exists a set $R \subset S$ such that

$$\sum_{a_i \in R} a_i = \sum_{a_i \notin R} a_i = Y \ ?$$

Without loss of generality, we can assume that

$R = \{a_1, \ldots\ldots\ldots\ldots, a_k\}$

and

$S = \{ a_1, \ldots\ldots\ldots\ldots, a_n \}$

Let's consider the schedule with two paths :

$P_1 = < i_1 , i_2 , i_1 , i_3 \ldots\ldots\ldots\ldots\ldots , i_1 , i_k >$

$P_2 = < i_1 , i_{k+2} , i_1 , i_{k+3} , \ldots\ldots\ldots\ldots\ldots\ldots , i_1 , i_n >$

Also,

$$r_i(v) = \begin{cases} r(v) & \text{if } v \in P_i \\ 0 & \text{otherwise} \end{cases}$$

for $i = 1,2$;

So,

$$\sum_v r_1(v) = \sum_{v \in P_1} r(v) = \sum_{j=1}^{k} r(i_j) = \sum_{j=1}^{k} a_j = Y$$

Similarly,

$$\sum_v r_2(v) = \sum_{v \in P_2} r(v) = \sum_{j=k+1}^{n} r(i_j) = \sum_{j=k+1}^{n} a_j = Y$$

Hence the conditions are satisfied and the total time = 2(n-1).

Thus the NEWS $_d$ instance also has an "YES" answer.

Again, if the NEWS(d) instance has an "YES" answer ,then there exists a schedule of duration $\leq 2(n-1)$

Since total requirement = $\sum_{j=1}^{n} r(i_j) = 2Y \geq 2W$ (assumed)

So ,there must be at least two trips.

At least one of these two uses the edge (i_0,i_j)(where i_j is of degree one) twice while the other does not.

Newspaper distributed in each trip must exactly be equal to Y as there are two trips only and total demand = $\sum_{j=1}^{n} r(i_j) = 2Y$

We may note that the two trips partition the set of edges (i_0,i_j) in two parts.

Let $R = \{a_j : \{ i_0,i_j \}$ is cleared in first trip$\}$

Clearly $\sum_{a_i \in R} a_i$ = Total newspaper distributed in the first trip

$= Y$

5.PROPOSED HEURISTIC ALGORITHM : We first give some definitions:

Let u , v ∈ V be the vertices of the graph $G = (V,E)$

Definition ***5.0*** A u-v walk in G is $(u = u_0,e_1,u_1,e_2,u_2,\ldots\ldots\ldots,e_n ,u_n = v)$ where $u_i \in V$, $0 \le i \le n$, are vertices ; $e_i \in E$, $1 \le i \le n$, are edges and

$e_i = \{u_{(i-1)},u_i\}$.

u-v walk is closed if u = v ;otherwise open .

Definition 5.1 A u-v walk in which all the edges are distinct is called a TRAIL . A trail that traverses every edge of G is called Euler trail of G . A Euler tour is a closed Euler trail. A graph is Eulerian if it contains an Euler tour.

Theorem 5.1 A non-empty connected graph is Eulerian iff it has no vertices of odd degree.

Corollary 5.1 A connected graph has an Euler trail iff it has two vertices of odd degree.

Our heuristic algorithm **DISTRIBUTE** works for any general graph . In the very first step , one has to solve the Chinese Postman Problem (CPP) for the given graph . Since the CPP reduces to finding an Eulerian tour when Eulerian graphs are considered , we implemented the algorithm for Eulerian graphs , although the rest of the steps remain the same for general graphs .

At first , we suggest some algorithm for finding out an Euler tour in an Eulerian graph . The following algorithm **EULERTOUR** does the same:

Algorithm **EULERTOUR :**

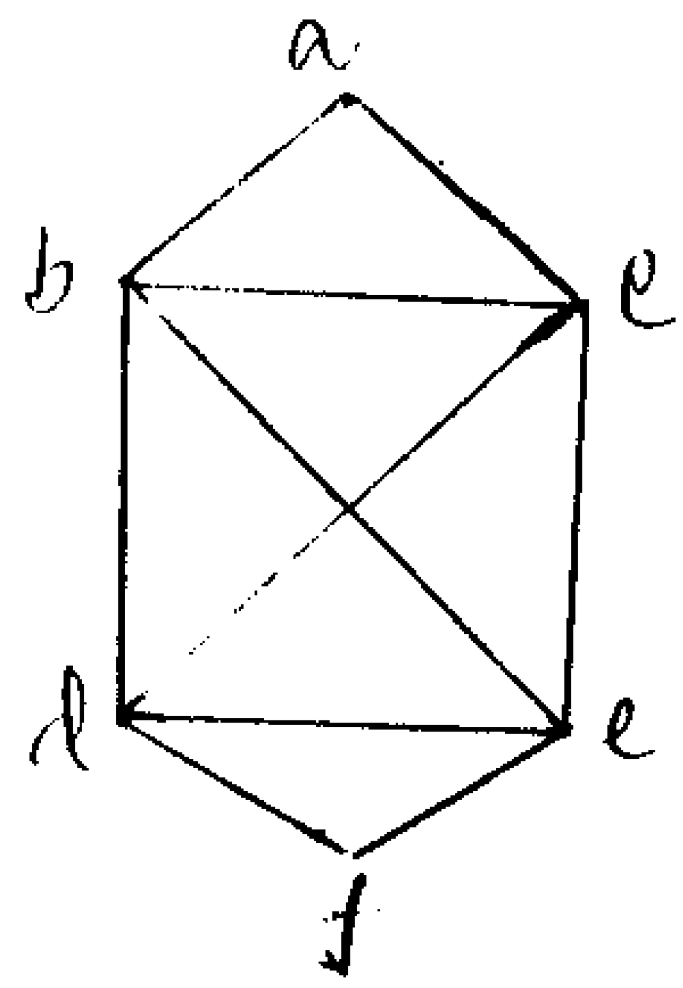
Step #1 Choose any vertex in the given graph and visit the adjacent unsaturated vertex . Repeat this until you come back to the starting vertex . Mark the edges already traversed . Store this trail in a list .

Step#2 Select the first unsaturated vertex from the list obtained in

Step#1 and repeat Step#1 . Continue until all the vertices are saturated

Step#3 Now you are left with several lists . Start from the first list . Replace the node which is the starting node of the second list by the contents of the second list . Repeat this until you are left with a single list.

ILLUSTRATION : consider the following graph :



Suppose a is chosen .

a->b->c->a is the closed trail .

b is the next unsaturated vertex in the above trail.

So we start from b .

b->d->e->b is the next trail .

The next unsaturated vertex is d .

So we start from d .

d->f->e->c->d is the next trail .

Now we find that all the vertices are saturated .

So we start merging the lists as follows:

a->b->c->a

b->d->e->b

are merged to a->b->d->e->b->c->a

Next , the above list is merged with the remaining list d->f->e->c->d

to finally give a->b-> d->f->e->c->d->e->b->c->a

Algorithm **DISTRIBUTE :**

INPUT : Adjacency matrix t_{ij} (edge time) of a weighted Eulerian graph having n number of nodes{1,...........,n}of which (n-1) are dumping nodes, node 1 being the newspaper office.

OUTPUT: Approximate route of minimum duration where each dumping node is visited at least once (excluding the last trip back to the newspaper office).

Step#1 : Find an Eulerian path in the given graph by the algorithm EULERTOUR..

Step#2 : Find the shortest path from node 1 to all other nodes(using Dijkstra's algorithm).

Step#3 : Let r(1),...................,r(m) be the sequence of requirements of the dumping points on the Eulerian path. Some of the vertices are repeated .When any particular vertex (say the i-th) is encountered for the second time , we put the corresponding requirement value equal to zero (i.e. r(i) = 0) .

Consider the first vehicle :

Compare floor(Σr(j)/W) and floor(Σ r(j)/W).

If they differ, assign the first vehicle to satisfy the requirements of all the nodes from the newspaper office up to the k-th. The remaining newspapers, i.e. (W - r(j))are dumped at node (k+1).

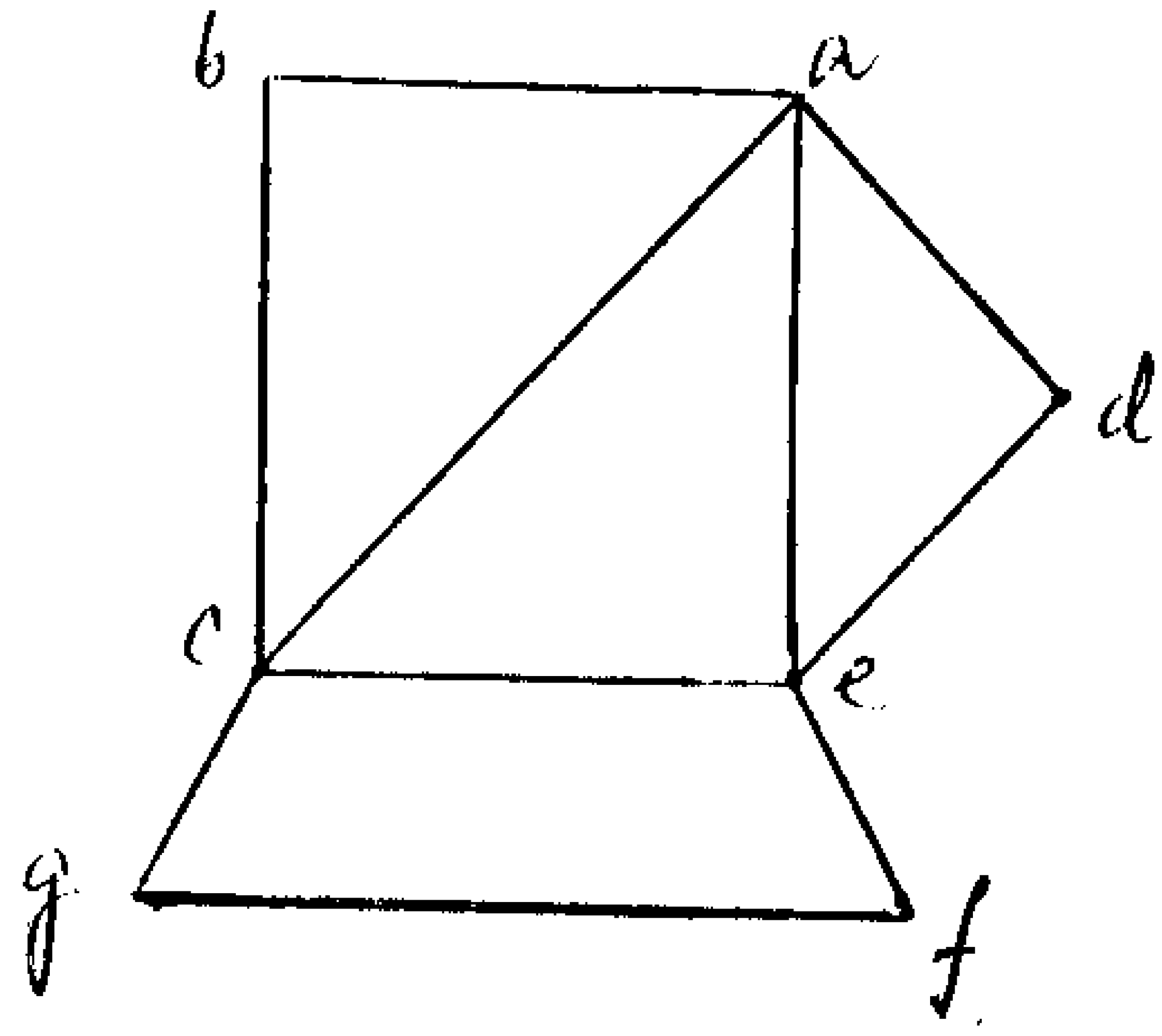
The requirements are updated.

The second vehicle also follows the same sequence of actions ,the only difference is that it starts from the (k+1)-th node. It goes to the (k+1)-th dumping point from node 1 following the shortest path.

The requirements are updated again.

This continues till all the requirements are met.

ILLUSTRATION :



Let

r(a) = 0 ;

r(b) = 2 ;

r(c) = 2 ;

r(d) = 2;

r(e) = 3;

r(f) = 5;

r(g) = 5;

and

Vehicle capacity W = 7

Output of EULERTOUR on the above graph :

a->b->c->a->d->e->f->g->c->e->a

0 2 2 0 2 3 5 5 0 0 0

First vehicle travels from a to d through b and c . At last , it dumps one unit of newspaper at e . So the updated picture is

a->b->c->a->d->e->f->g->c->e->a

0 0 0 0 0 2 5 5 0 0 0

The second vehicle directly goes to e from a following the shortest path . The updated picture is :

a->b->c->a->d->e->f->g->c->e->a

0 0 0 0 0 0 0 5 0 0 0

So the third (i.e. the last in this particular case) goes to g from a following the shortest path . The updated picture is

a->b->c->a->d->e->f->g->c->e->a

0 0 0 0 0 0 0 0 0 0 0

REMARK : Since all the vehicles actually move in parallel ,the total duration of the distribution process will be much less than the apparent sequential behaviour of the algorithm.

IMPLEMENTATION & RESULTS : A program in C language has been written to simulate the above heuristic . The results obtained are optimistic and in most of the cases , near optimal solution is obtained .

References

[1]Beltrami ,E.J.(1977) , Models for Public Systems Analysis , Academic Press , New York .

[2]Edmonds , J and Johnson , E.L. (1973) , Matching Euler tours and the Chinese Postman , Mathematical Programming , 5.

[3]Garey , M.R. and Johnson , D.S. (1978) , Computers and Intractability , A guide to the theory of NP-Completeness , W.N.Freeman and Company , San Fransisco.

[4]Helly , W(1975) ,Urban Systems Models , Academic Press , New York.

[5]Papadimitriou , C.H. and Steiglitz , K.(1982),Combinatorial Optimization : Algorithms and Complexity , Prentice Hall , New Jersey .

[6]Roberts , F.S.(1984) , Applied Combinatorics , Prentice Hall , New Jersey .

[7]Bondy,A. and Murty,U.S.R. (1976) , Graph theory with applications , The Macmillan Press Ltd.