Networks and Vehicle Routing for Municipal Waste Collection

E. J. Beltrami
L. D. Bodin
State University of New York
Stony Brook, New York

ABSTRACT

Vehicle routing for municipal waste collection encompasses a variety of problems. In this paper, we explore the techniques we have developed for solving some of these problems.

INTRODUCTION

One of the persistent questions facing many municipalities is "what do we do with all the garbage?" In operational terms, this means that communities must decide how many disposal facilities to have, where to locate these facilities, and how to get the ever increasing amounts of refuse to them while at the same time satisfying environmental and economic constraints.

In this paper we focus our attention on some operational problems regarding waste collection. Most of what we discuss is a direct outgrowth of our experience as members of the Technical Advisory Group to the New York City Environmental Protection Agency (which includes the Department of Sanitation), and to the Department of Environmental Services of Washington, D.C. Almost all the procedures described in this paper are being implemented in these cities.

Collection activities of the New York City Department of Sanitation include household and bulk collection by sanitation crews working on trucks and barges as well as street cleaning and snow removal. About 80% of the $200 million annual operating budget for the New York City Department of Sanitation is related to wages and benefits for the 11,000 sanitationmen and most of this is in connection with refuse collection. However, little work has been carried out so far on the optimal utilization of this workforce.

Networks, 4: 65-94

In this paper, we consider one aspect of this question; namely, the efficient routing of collection vehicles. Other facets of this problem will not be treated here. These include questions of how to optimally deploy the workforce (this area has been discussed elsewhere by Bodin [1] and Altman et.al. [2]), the effect of innovations in collection technology, and the optimal location of disposal facilities. It is, however, apparent that all of these questions are interrelated. For example, if the disposal facilities are inadequately designed or if there are too few of them available, then delays occur because of the congestion of vehicles waiting to dump. Furthermore, such delays can affect the routing of the vehicles by reducing the amount of time a truck has for collection.

The magnitude of the collection and disposal task in New York City becomes further apparent from the fact that each day about 25,000 tons of solid waste need to be collected and disposed of and about 11,000 miles of streets must be cleaned. Thus, any attempt to render these services more efficient is likely to result in substantial dollar savings. The studies described in this paper have produced noticeable cost benefits to New York City.

In what follows we describe a set of procedures for routing vehicles through a set of nodes in a network or along its branches. Although the task is one of collection with disposal as a terminal operation at the dumpsites, the problem is not too unlike that of the truck delivery problem. In that case, a dump is replaced by a depot and the pickup points become delivery points. For these reasons, the procedures in this paper are of general interest.

One final comment is in order about the methodology to be discussed. Many large scale problems which arise from real-life situations simply do not fit into classical molds and one must be satisfied with near optimal solutions as obtained by combining formal arguments with heuristic reasoning. What is meant by heuristic has been well described by others (e.g., Krone [3]). We will be satisfied to think of heuristic reasoning as meaning that one brings to bear as much intuition and as many plausible arguments as possible on problems which are either computationally intractable or for which an inadequate theory exists. Generally speaking, the kind of problem that we have encountered concerns discrete or combinatorial structures which in one sense or other need to be optimized. In what follows we will try to clarify these remarks by means of some examples.

SOME GENERAL REMARKS ABOUT ROUTING

There are several things to look for in determining the type of algorithm to be used in developing routes for the vehicles. Generally, we are given a map of a region which can be transformed into a network with a node set N and branch set A. The first question that arises is whether we are trying to perform the routing over the nodes in the network (in which case the branches indicate the nodes which are adjacent to each node) or over the branches in the network (in which case the nodes in the network indicate intersections or connection points between the branches). The first class of problems have been called discrete or node routing problems while the second class of problems have been called continuous or branch routing problems. (Marks [4]).

For node routing problems, the objective is to combine the nodes into routes so as to minimize the number of vehicles needed to pick up the refuse at all the vertices in the network while being subject to the capacity constraints of the vehicle and the time constraints of the crew. For branch routing problems, the objective is to minimize the total time the vehicles need for going over the branches in the network more than the required number of times, being subject to the capacity constraints of the vehicle and the time constraints of the crew. The deadheading or deadhitting time of the vehicle is the time the vehicle uses in being routed over the branches in the network more than the required number of times.

If there is but one vehicle to route over the nodes in a network, then we have the Travelling Salesman problem for which both exact and approximate solution procedures exist (Held and Karp [5] and [23], Lin [6] and Little et.al. [7]). If but one vehicle is available to route over the branches of a network, then we have the Chinese Postman problem (Stricker [8]). When all branches in the network are either directed or undirected and one needs but one vehicle, then exact procedures are known for solving this problem (Liebman [9], Liebling [10], Edmonds and Johnson [11], and Christofides [12]). In the case when the network contains both directed and undirected branches, a heuristic procedure has been developed by Edmonds and Johnson [11].

Both the node and branch routing procedures become more complicated when it is necessary to route more than one vehicle over the network. In selecting a proper algorithm, both an analysis of the network and of the results that are expected must be carried out. One has two choices in these routing procedures. First, one can cluster the networks into subgraphs and route one vehicle over each of the subgraphs. Second, one can disregard the time and capacity constraints of the vehicles, form a giant tour through all the nodes (or branches) in the network, and then partition this giant tour into a collection of routes which are feasible with regard to the capacity constraints of the vehicle and the time constraints of the crew.

In the node routing case, the choice between procedures is based on the following considerations - the number of pickup points one can expect on an individual tour for a truck, the number of routes to be created, the time in transit from the pickup points to the dump site and the time to go between pickup points in the network. Generally, if there are few routes to be formed with many pickup points on each route, then it is generally more effective to form a giant tour and then partition this tour into smaller segments. Note that in making a giant tour, the pickup points are ordered without regard to the time to go from the pickup points to the dump. Thus, in the case when there are many tours to be created, with a few pickup points on each route, forming routes first will generally give better results.

In Newton and Thomas [13] an analysis of this problem in the node routing case is carried out. They show that in certain cases breaking up the giant tour works best while in other cases clustering the nodes first gives more effective routes. Heuristic procedures for the node routing case are described in Clarke and Wright [14], Eilon, Watson-Gandy and Christofides [15], Bennett and Gazis [16], Altman et.al. [17] and the IBM VSPX Scheduling Program [18].

In the branch routing case, the choice between competing algorithms is not only based on the parameters mentioned earlier but on other considerations also. Forming a giant tour first results in a better solution (i.e., the total deadheading will be less) but gives tours which may have some undesirable properties from an administrative standpoint. Partitioning first gives non-overlapping regions while not partitioning first gives routes that interact. Since non-overlapping regions are easier to check and administer, they may be more desirable even though the total deadheading time may be greater.

We shall explore these ideas further in this paper as we describe the different routing procedures that we developed. Suffice it to say at this point that in the node routing cases we clustered first and then optimized each of the routes that we derived, while for the branch routing case, we minimized the deadheading first and then partitioned this giant tour into feasible routes for the vehicles.

ROUTING THROUGH NODES

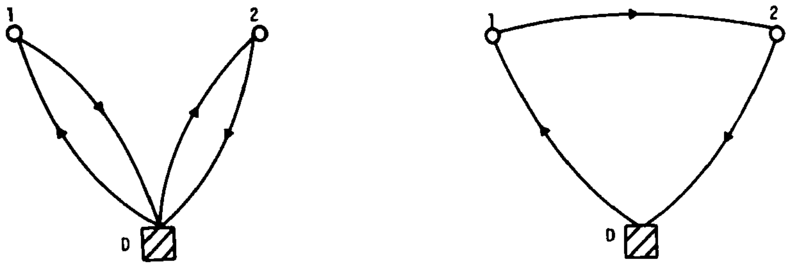
Most refuse collection activities in New York City center around the pickup of household refuse in small bins. However, large institutional sites such as schools, hospitals, and apartment complexes have their refuse stored in large 6-12 cubic yard metal containers (equivalent to 40-80 normal garbage cans). Eventually about 1,000 such sites will need to be serviced by using large capacity trucks which depend on mechanical fork lifts to load the bins into the chassis where it is then compacted (these are called hoist-compactor trucks). Each truck can service several such sites before going to a dump to unload.

The problem we considered was how to route the hoist compactor trucks to minimize the total travel time of the vehicles and to determine the minimum number of trucks needed each day. This last condition is important from the point of view of minimizing the capital expenditure needed to outfit a fleet of trucks.

There is also a dual problem to the one mentioned above. That is, for a given number of trucks one seeks the maximum number of locations that can be serviced. We have found several organizations who have either purchased or leased a fleet of trucks interested in this question since they want to know how far they can expand their operations without additional capital expenditure.

The problem is not simply one of finding the shortest path through a network since there are a number of complicating factors. Hence, it becomes quickly apparent that the usual graph theoretic arguments are not directly applicable. First, one must be mindful of capacity and time constraints. Each pickup point can have a different quantity to be picked up and, since the capacity of the truck is limited, the route must be interrupted for travel between pickup points to dumps. Moreover, there are several dump sites. Having saturated the truck, the problem asked is which dump site should be used? Finally, some locations require daily service while others do not. Since there are typically many points to service, the problem is not only to arrange the routes feasibly but to assign each pickup point to days of the week to minimize the number of trucks needed.

An analysis is based on a modification of the heuristic Clarke and Wright idea [14]. The basic Clarke and Wright method, quite simply, is this: Assume, to begin with, that a tour consists of visiting one containerized location and then returning to the depot or a dumpsite. If now two locations are joined together to form one tour then the "savings in total travel time" is illustrated in Figure 1. The savings S_{ij} between pickup points i and j is $S_{ij} = T_{oi} + T_{oj} - T_{ij}$ where T_{ij} is the time to go between point i to j and T_{oi} (or T_{oj}) is the time to go from dumpsite o to pickup point i (point j). One then calculates the "savings" associated with all pairs of locations to be serviced and then sorts in decreasing order the list of pairs with positive savings. Starting at the top of the list, points are combined provided that the resulting tour is feasible from the point of view of time and capacity. In this way increasingly longer and better tours are formed until the list is exhausted.



Initial Routing Procedure: Total travel time is twice travel time from dump to site 1 plus twice travel time between dump and site 2.

a

Modified routing procedure: Total travel time is travel time between dump and site 1 plus travel time between dump and 2 minus travel time between sites 1 and 2.

b

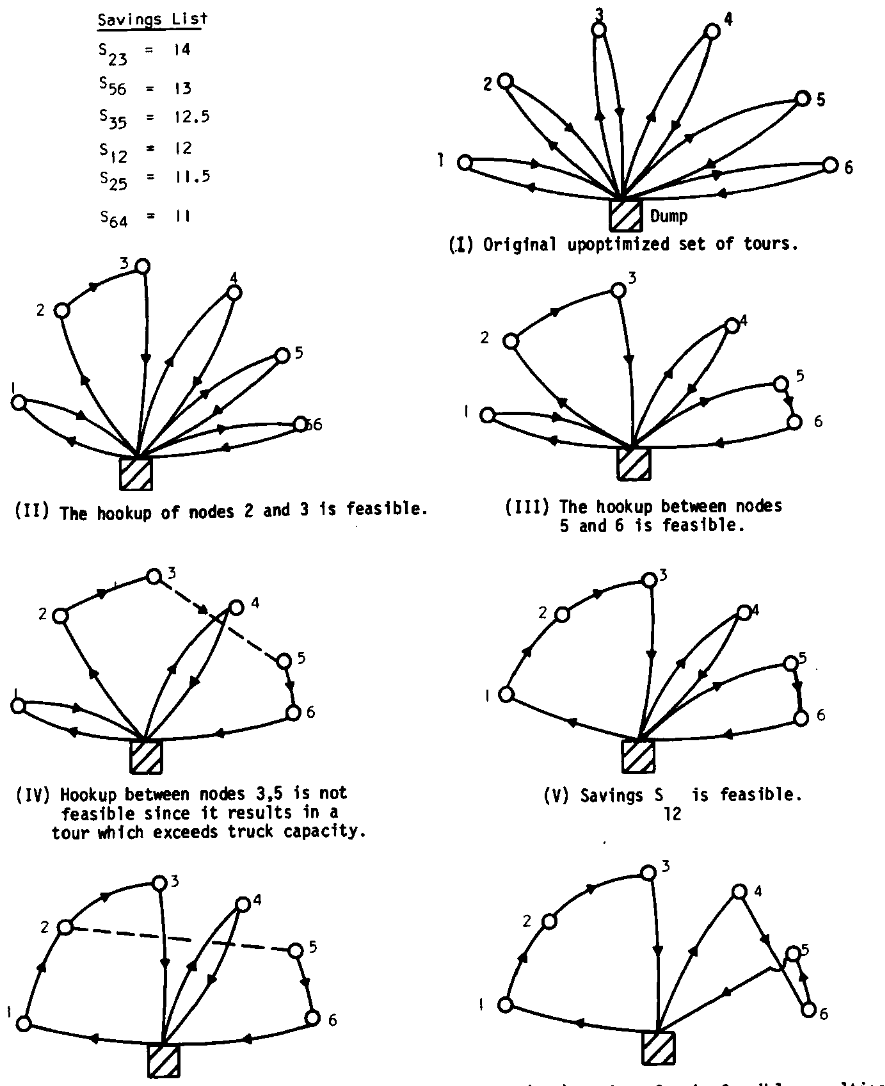
Fig. 1

Example 1

In Figure 2, we illustrate the Clarke and Wright procedure on a simple problem. Each of the sites is to be serviced every day, the demand at each location is 1 unit, the capacity of the vehicle is 3 units, and there is no time constraint. The tours which result from the Clarke and Wright procedure are now constructed step by step. Let S_{ij} be the savings derived by joining sites i and j together. We consider the network to be undirected, and the savings to be defined over those sites which we consider adjacent. In the case when we are studying a complete graph, all nodes are adjacent to each other; in other cases (as exhibited in Figure 2) the graph need not be complete.

Several comments are in order. First the ability to generate tours by this approach does not preclude the possibility that some tours can be improved (total time to cover tour reduced) by altering the order in which the points on the tour are visited.

This is done by solving the traveling salesman problem. Rather than developing a general traveling salesman code, we used the 2-opt and 3-opt heuristics of Lin [6] to approximate the optimal solution to the traveling salesman problem.



(I) Original upoptimized set of tours.

(II) The hookup of nodes 2 and 3 is feasible.

(III) The hookup between nodes 5 and 6 is feasible.

(IV) Hookup between nodes 3,5 is not feasible since it results in a tour which exceeds truck capacity.

(V) Savings S_{12} is feasible.

(VI) This hookup is not feasible since node 2 is already interior to an existing tour and so node 5 cannot be joined to the tour containing node 2.

(VII) Savings S_{64} is feasible, resulting in the final set of tours.

Fig. 2

Second, the simple generation of tours does not completely solve the problem. It happens that some sites need to be serviced 3 times a week while other sites require 6 times a week service (there is no garbage pickup on Sunday except in circumstances which we can ignore here). The question that arises is which 3-time-a-week sites shall be serviced on Monday, Wednesday, Friday and which 3-time-a-week sites shall be handled on Tuesday, Thursday, Saturday. There are several possible ways of solving this problem, two of which are described below.

One approach (first described in [17]) is to form tours and then to assign these tours to either Monday, Wednesday, and Friday or Tuesday, Thursday, and Saturday. The second approach is to preassign sites to specific days of the week and develop routes based on this assignment. The algorithm to be used is based on the decision whether to cluster the nodes first by day assignment and then develop routes based on these clusters (the second approach) or to route first by forming tours and then cluster the resulting tours into day assignments (the first approach). We have implemented both procedures in the study of hoist compactor routing in New York City.

In the first approach, we used the Clarke and Wright procedure to generate the tours but made all 6 times a week nodes appear twice in the network. However, in forming tours, such a site and its image were not allowed to appear on the same tour. This forced the 6-time-a-week locations to appear on two different tours, each of which is assigned to different days of the week. If there are k six-time-a-week sites in the original network of n vertices then the network with its images consists of $n+k$ sites. In this algorithm, one is sometimes able to join site $i+n$ (the image of site i) with site j even when i cannot be joined to j (because of the subtours that sites i and j are on).

Having completed the first part of this algorithm, one has a collection of tours which are so far unassigned to either Monday, Wednesday and Friday or Tuesday, Thursday, and Saturday. The problem now arises as to which days of the week to assign these tours to. This can pose a difficulty. It is not obvious that from a given collection of tours a feasible day assignment can be found. This point is illustrated in the following example.

Example 2

Consider the three tours shown in Figure 3 which results from use of the algorithm. Here, nodes 1, 2, and 4 are to be serviced 6 times a week and the others 3 days a week. Assign the first route to Monday, Wednesday, and Friday. Then the second tour should be scheduled on Tuesday, Thursday and Saturday since it has point 2 in common with the first. Similarly, the second and third tours have point 4 in common so that the last route should be assigned to Monday, Wednesday, and Friday. This leads to an impasse since the third and first tours also have a point in common.

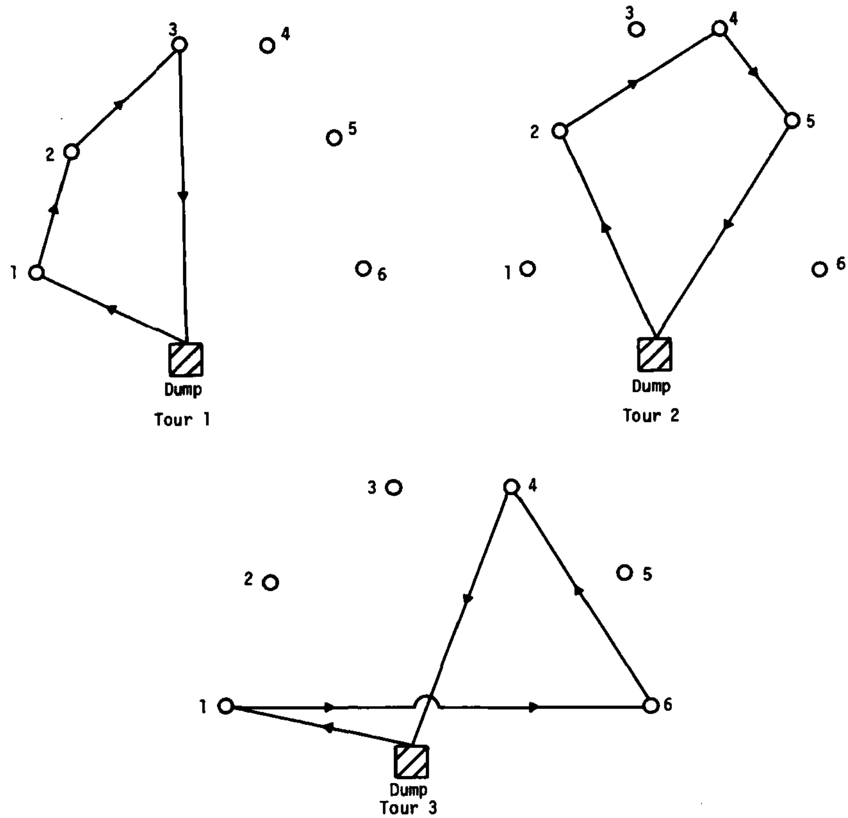


Fig. 3 An infeasible set of tours.

We were able to successfully handle this feasibility question by building into the computer code a routine which prevented the cycle as indicated in the above example to occur. Since there were few six-day-a-week collection points, we did not feel that such a heuristic seriously affected the routes that were generated. If there were many six-day-a-week collection points the second approach imbedded within a simulation would be recommended because it would have been very difficult to find a nearly optimal feasible solution using the first approach. The simulation randomly assigns sites to either Monday, Wednesday, Friday or Tuesday, Thursday and Saturday and solves a standard Clarke and Wright routing problem on the points assigned to Monday, Wednesday, Friday, and the points assigned to Tuesday, Thursday, Saturday.

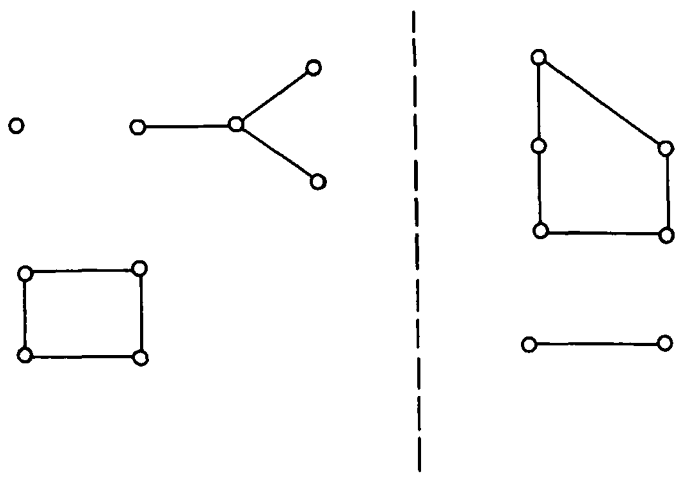
The random assignment of sites to days is repeated several times. The best Clarke and Wright solution from these random assignments is then taken as the optimal set of routes.

It is interesting to note that after experimenting with the first approach quite extensively, the second approach was actually used for routing the hoist compactor trucks for the New York City Department of Sanitation. This was done because the Department of Sanitation decided a priori the day assignment for each site.

Furthermore, this difficulty raised an interesting theoretical question in graph theory. It is perhaps worth a short digression for a glimpse as to what this question entails.

Let us form a graph I whose nodes represent the tours. An edge is inserted between two nodes if and only if the corresponding tours have a pickup site in common. One can prove the theorem that the set of tours is feasible (i.e., does not possess the pathology due to two service frequencies as in the example above) if the graph I contains *no cycles of odd order* or, equivalently, if I is bipartite [17]. We call I an I-graph. This is the same condition one encounters in the two color problem, which is to color adjacent nodes differently using only two colors. Typical I-graphs are displayed in Figure 4.

Typical I-graphs



This graph leads to feasible weekly schedules (it is colorable with two colors).

a

This graph is infeasible since it contains an odd order cycle.

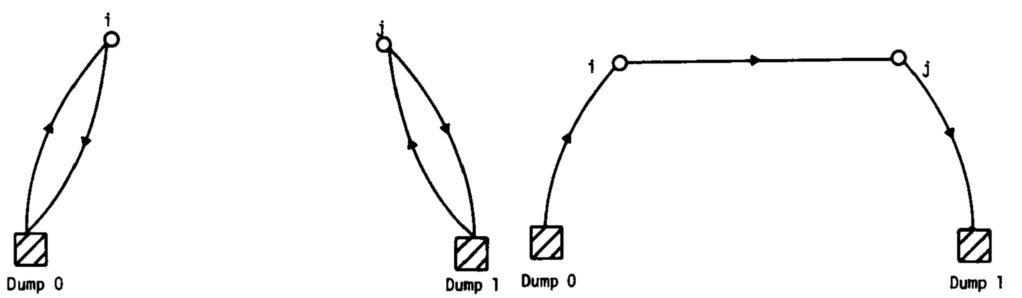
b

Fig. 4

More generally one is interested in conditions for coloring a graph with K colors when K > 2. Only partial results are known here and the problem remains open [20]. This question has application in refuse collection when one wishes to form routes which cover sites that have more than two different service fequency requirements (for example, 2-, 3-, 5-, and 6-day-a-week service).

It should be mentioned in passing that since the methods used are heuristic it may happen that imposing additional constraints can sometimes yield better results. We found this to be true in one case in which certain tours were blocked to prevent the pathology associated with multiple service frequencies. This somewhat counterintuitive result can be explained by the fact that heuristic algorithms can sometimes lock on to local solutions which are not nearly globally optimal. The use of constraints forcing the tours to be ordered in a different and perhaps better way.

An additional refinement has to do with multiple dumps. The basic Clarke and Wright algorithm assumes that each truck disposes of its load at a single location. If there are several such disposal locations we preassign each collection site to the dump closest to it. The "savings" of combining two points assigned to the same disposal location is computed as before. However, the savings achieved by combining points assigned to two different dumpsites is now $S_{ij} = T_{oi} + T_{1j} - T_{ij}$, where T_{oi} (or T_{1j}) is the time to go between dumpsite o and site i (or between dump 1 and site j) and T_{ij} is travel time between sites i and j (see Figure 5).



Initial routing in case of two dumps.
a

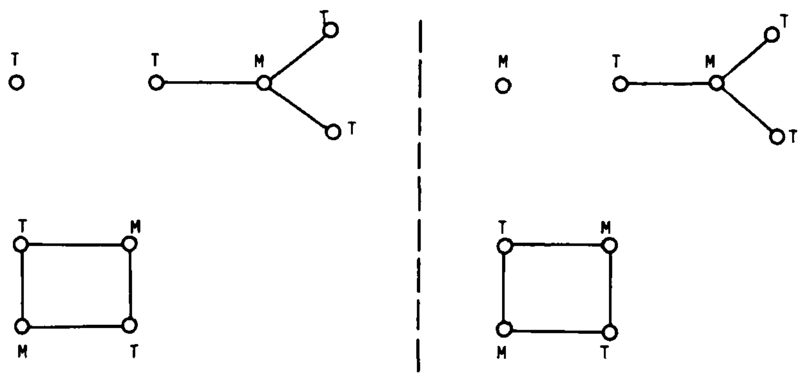
Routing procedure which results in a savings.
b

Fig. 5

A final question to be considered in the routing of the hoist compactor trucks is that of forming day schedules for each of the trucks. It is a typical occurrence that these tours do not exceed the time constraints imposed by union regulations and, hence, several such tours can be hooked together to form a day schedule for the truck. Generally, the capacity of the hoist compactor truck is the binding constraint in generating routes by the algorithms of the Clarke and Wright type. Since one of our objectives is to minimize the total number of trucks needed, some ways of forming day schedules are better than others.

Again, we use heuristic reasoning in concatenating together routes generated by the above procedures to form daily schedules. To see what is involved consider, for example, a set of feasible tours whose I-graph is illustrated in Figure 4a. Suppose, for simplicity, that each tour constitutes a daily route. If M denotes the Monday, Wednesday and Friday tours and T the remaining ones, then two scheduling possibilities are shown in Figure 6. The first option requires 3 trucks on M and 6 on T, so that a total of 6 trucks are needed for the week, whereas the second case involves a maximum of 5 trucks.

Arrangement of Truck Schedules



This option is not optimal.
T represents Tuesday, Thursday, Saturday schedules and M represents Monday, Wednesday, Friday.

a

This option requires the least number of trucks.

b

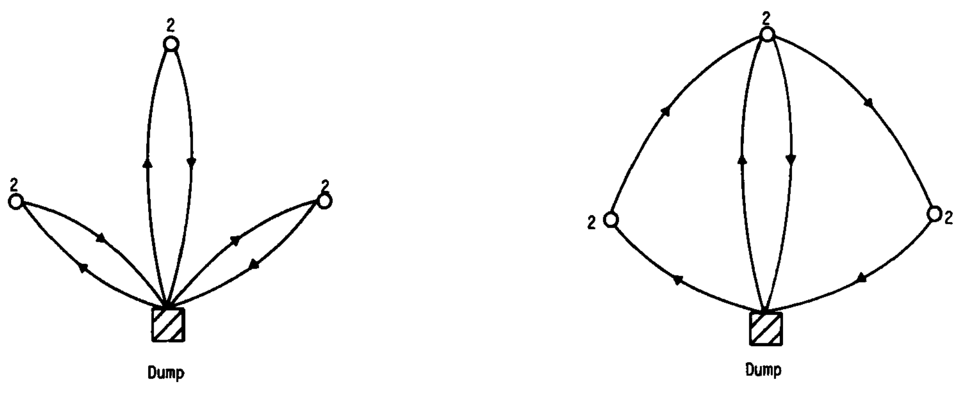
Fig. 6

RANDOMIZED TOURS

A second routing question we encountered was the problem of routing and dispatching of barges and tugboats for the disposal of 10,000 tons of refuse daily at a landfill on Staten Island. The refuse is at 8 marine disposal locations around New York City (brought there by trucks, as discussed above) and is then hauled away by barges. There are two shifts a day with a resulting daily average flow of 16 barges back and forth between the landfill and each of the marine disposal points.

Many of the operational questions here are similar to the ones discussed above but there are some additional complications due to the fact that one has to account for tides (tugs move slowly, if at all, against the tide) and because each tug can haul as little as one barge per trip. Since it is not necessary to pick up a full complement of loads at each site it visits, a given transfer station can be visited on more than one trip each day.

One of the basic restrictions in forming truck tours as described in the previous section is that each node is to be visited at most once on a given day. However, there may be some advantage in visiting some points several times in one day, by different vehicles on different routes. Figure 7a exhibits the only tours possible when each truck has maximum capacity of 3 (and a maximum travel time of 6), if we insist on visiting each node only once. The total travel time is 12. However, in Figure 7b we reduce the travel time to 10 and use only two tours by the simple expedient of allowing site 2 to be serviced twice, where each time half of its daily requirement is taken care of. In general, this kind of split service at certain key nodes can be expected to cut down the number of tours required to satisfy demands, especially when these nodes are such that their requirements are nearly the same as the capacity of the vehicles, which is the case in barge routing.



The numbers indicate the requirements or loads at each of the nodes. Vehicle capacity is 3. Travel time between nodes is 1 and from nodes to dump is 2.

a b

Fig. 7

However, the number of possibilities now are much greater. Suppose, for example, that there are 8 nodes as in the barge problem and that each tour can cover 4 nodes (clearly an approximation). Suppose, in addition, that each node has 3 units to be serviced and that vehicle capacity is 6. Then the total number of possible tours, when splitting is allowed, is $\binom{8}{4}\left(1 + \binom{4}{1} + \binom{4}{1} + \binom{4}{2}\right) = 1050$, and $\binom{8}{1} + \binom{8}{2} = 36$ when splitting is not allowed.

What we suggest is that tours be generated by combining the Clarke and Wright algorithm with a randomized search procedure. The way this is done is now explained.

Suppose we encounter nodes i and j as elements in the savings list where node i has to this point an unassigned requirement of m_i barges and node j has an unassigned requirement of m_j barges. We first check if it is possible to join nodes i and j together without violating the capacity constraints of the tugboats and the time constraints of the crews. If this is so and we randomly decide to form a new tour consisting of nodes i and j alone, then a random integer between 1 and m_i (1 and m_j) is chosen to represent the number of units to be actually serviced at node i (node j). Should the resulting tour through i and j be feasible, the hookup nodes i and j is effected. Otherwise, other load combinations are tried at random until a successful one results.

We then set $m_i = m_i - n_i$ and $m_j = m_j - n_j$ where n_i and n_j are the number of units to be serviced at nodes i and j as found in the above randomization. The above procedure is then repeated until either or both of the nodes i and j have their requirements satisfied. If i and j are simultaneously depleted, we move to the next item in the savings list.

It can occur through the above randomization or when we encounter nodes i and j in the savings list that either, but not both, $m_i = 0$ or $m_j = 0$. We now try to link the unexhausted node, say j, to a tour which already has i as an endpoint and which does not already contain the node j. We randomly determine how much of m_j to assign to this tour. We then search through the remaining tours to determine if there are any other tours with node i as an endpoint. We then repeat the procedure described in this paragraph until j has no loads left, or until there exists no feasible connection possible between j and each of the tours that have i as an endpoint. At this juncture we move to the next pair of elements on the savings list, thereby continuing to enlarge the tours. This means that at each move along the list one generally has to store several feasible paths through the points which appear there, unlike the usual savings approach.

There are several other refinements in the algorithm which are best illustrated graphically. For this reason Example 3 will examine in detail how the algorithm works in a quite typical problem. It should be noted however that if the random selection is not a fortuitous one then the resulting set may require too many round trips and more travel time than is necessary. For this reason the entire procedure is repeated a number of times. Each such repetition is likely to result in a different set of tours which is then compared with the previous one and the better set retained. How large a sample to take is problem dependent but as a rule the iterations would continue either until they become too costly or until substantial improvements cease to occur. Example 3 develops only one set of possible tours. This will suffice for the purpose of illustration.

Example 3

To illustrate the potential effectiveness of the randomized method against the non-randomized Clarke and Wright algorithm, consider a six node network in which the vehicle capacity is 5 loads and the maximum allowable time is 8 units. The demand requirements at the sites is given by 2, 5, 4, 1, 1, 1 respectively for nodes A, B, C, D, E, F. The ordinary approach does not allow for splitting results in 5 tours for a total tour time of 22 units, as one can easily verify, whereas a particular randomized set of 3 tours can be generated for a total tour time of 18 units as illustrated in a step by step fashion. The savings list and distance matrix is exhibited in Figure 8.

Distance Matrix

	A	B	C	D	E	F
Dump	4	2	1	2	1	2
A		1	4	6	5	7
B			2	1	3	5
C				5	1	3
D					4	4
E						1

Savings list

$S_{A,B} = 5$

$S_{B,D} = 3$

$S_{E,F} = 2$

$S_{B,C} = 1$

$S_{A,C} = 1$

$S_{C,E} = 1$

Fig. 8 Distance matrix and savings list for Example 4.

I: The A, B, pair is first in the savings list. The first random sample gives loads 2 and 4 which exceeds truck capacity and so it is rejected.

II: The second random sample on the A, B, pair gives loads 2, 2. Since both time and capacity constraints are met, A can be joined to B to form tour 1. The load requirements at A, B are now 0, 3. See Figure 9.

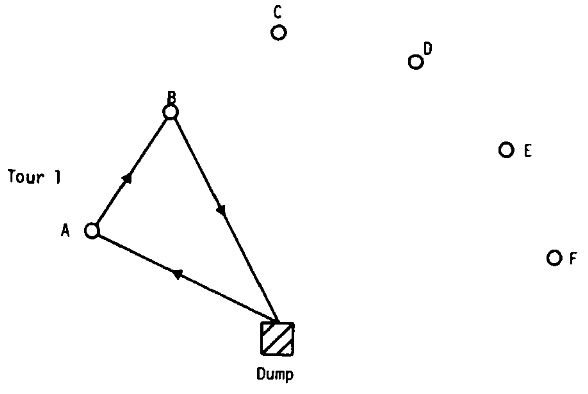


Fig. 9

III: Now try the B, D pair. Since B can be linked to D either as a member of tour 1 or by itself, the randomization says to link D to 1. There is only one possibility here since D has only one load. Note that we do not sample for the load at B since we are attempting to join D to an already formed tour and not to an isolated point. Since both time and capacity constraints are met, tour 2 results as shown in Figure 10. Note that we could have tried to join D to B as isolated points first, in which case a different tour would have resulted. However, D no longer has any load requirements left to it and so such a hookup is now meaningless.

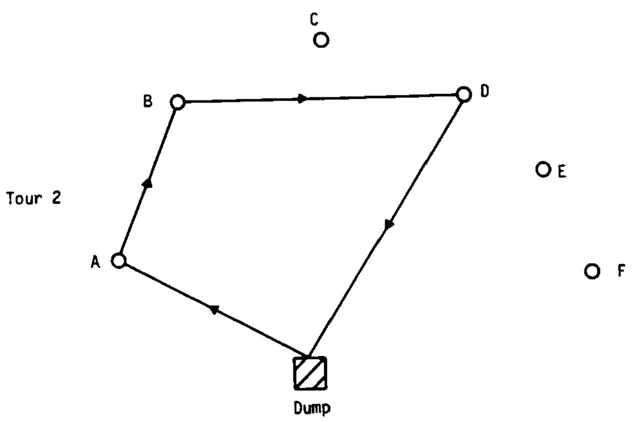


Fig. 10

IV: Now try the E, F pair. Here each has only one load and so there is only one possibility. Since no constraint is violated we joined them to form tour 3. Nodes E and F now no longer have any demands. See Figure 11.

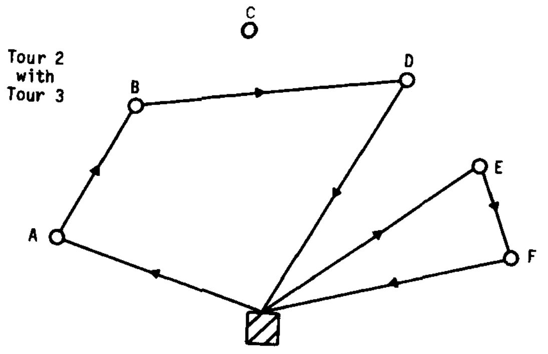


Fig. 11

V: Try the B, C pair which is next on the savings list. Since B is an interior point of tour 2, C cannot be attached to this tour. However, B can be considered by itself here. A random sampling for B, C gives loads 3, 2 which is acceptable and so we form tour 4, as shown in dotted lines in Figure 12. This move exhausts the requirements at B.

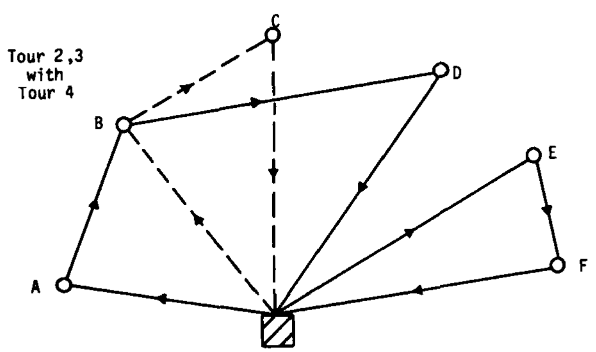


Fig. 12

Note that a different set of possibilities would have occurred if, for example, the load sample had been 2, 2 instead of 3, 2 which would give tour 4'. In this case, since B, C both still have demands to satisfy, one takes another random sample to obtain 1, 1. In this case B and C can again be joined to form tour 4" which is identical in travel to tours 4 and 4'. We will examine the consequence of this possibility in the Step VII below keeping in mind that node C still has one unit of demand to satisfy in this case.

VI: In the A, C pair A cannot be considered by itself since it has no load. C can only hook onto A if we consider A as an endpoint of tour 2. But this is not feasible since the total time for the new tour would be 9 > 8. Therefore, this possibility is rejected.

VII: Now try the last savings entry, the C, E pair. Both E and C are on tours already (tours 3 and 4) and so we try to hook these tours together. However, this is not feasible since the total loads of the combined tours would be 7 > 5. Therefore, we reject this possibility. However, even though no other node has any load demands remaining, node C has 2 loads to deliver up. Therefore we try to join tour 3 with the isolated point C. A random sample for C gives a load of 2 which results in the feasible tour 5 shown below together with the other already existing tours (Figure 13). We have now reached the end of the savings list and all nodes have had their demands satisfied. The final set of tours are shown in Figure 13 (tours 2, 4, 5) for a total travel time of 18.

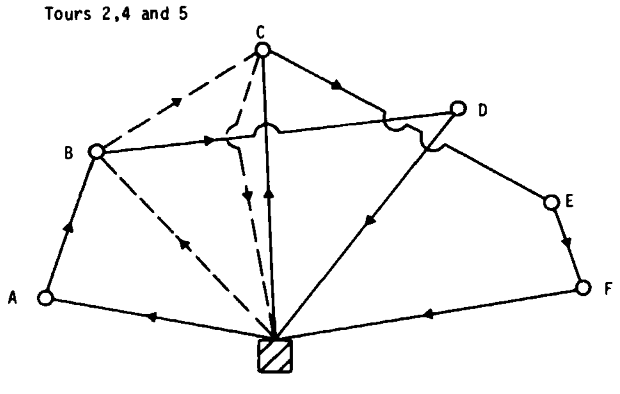


Fig. 13

Note that if the second sampling possibility of Step V had occurred then one would have tried to hook up tours 4' and 4" with tour 3. The hookup of 4' with 3 is not feasible but 4" with 3 is acceptable giving a tour 5'. This leaves point C isolated with a single demand. This results in a final tour 5" consisting of a round trip between the dump and C. The final tours would now be 2, 4', 5', 5" for a total time greater than 22. This illustrates that some random sets of tours can be disadvantageous.

It is possible to find other sets of tours by the randomized approach whose total time is also less than 22 units. We can, therefore, conclude that the randomized approach is reasonably robust in comparison with the ordinary Clarke and Wright.

To reiterate, the randomized approach is most effective when the demand on many of the nodes is near the capacity of the vehicle. In a general case when there are many nodes, randomizing a few of the large capacity nodes allow these demand points to act as slack variables in generating tours; the unused capacity of any tour being filled by part of the demand at these points. Such a procedure can lead to considerable savings in the number of vehicles needed.

Finally, there have been other attempts to extend the Clarke and Wright idea to solve problems for which core procedure is inadequate. Some of the more interesting references are Eilon, Watson-Gandy and Christofides [15], Bennett and Gazis [16] and Krolak et.al. [21]. In the appendix there are some additional comments and examples on the nature of the Clarke and Wright heuristic which may offer insight on the limitations of this method.

BRANCH ROUTING

A third class of problems which we were asked to investigate by the New York City Department of Sanitation was the routing of the street sweepers or, as they are sometimes called, the mechanical brooms. These vehicles, which are trucks having rotating brooms attached to their front, sweep the streets of a road network by moving along these streets in a continuous path. This continuous path is a sequence of streets that have to be covered and streets that are deadheaded.

For a given area of New York City, we know, because of the City's alternate side of the street parking regulations, which side of the street has to be swept for each time period. Furthermore, each time period is separated from the other time periods by at least two hours; therefore, we can solve the routing problems for each time period separately. For each time period and a street network representing a given area, there is a subset of streets that have to be covered. The other streets in the original street network are used for deadheading the vehicles between the streets that are to be covered.

The starting and ending times of the shift for the driver are known and there are no restraints on how much refuse the brooms can pick up. The brooms are to be at the starting point of the tour when the tour is scheduled to commence, and the brooms are to return to the garage at the end of the day. However, we did not have to account for these times in the routing of the vehicle because the hours that the alternate side of the street parking regulations are in effect are well contained within the starting and ending times of the shift for the driver.

The problem we have, therefore, is to service the street network that has to be covered in each time period so that the total amount of deadheading by all the brooms assigned to the area is minimized. This deadheading time can occur from either a vehicle traversing a street that has to be covered more than the required number of times or travelling at least once over a street that does not have to be swept. By reducing the deadheading as much as possible, we can either cut down on the number of brooms that are needed to service a given area or provide a better service to this area with the brooms we now have available. Better service in this case means traversing a street at a slower speed or going over a street more than once during a time period.

Because the sweepers have to travel in the direction of traffic and because of the alternate side of the street parking regulations, we consider all the branches in the network to be directed. Thus, for a two-way street, this allows us to distinguish between the side that is to be covered and the side which can be used for deadheading. There are other classes of branch routing problems whose networks are either undirected or a mixture of undirected and directed. Two examples of this in the public domain are the routing of snow plows and the routing of trucks for household refuse collection.

As cited earlier, variants of this problem have been of interest to mathematicians and operations researchers for many years. One version of this problem was mentioned by Euler in 1736 in a celebrated paper in which he showed that when all the nodes of the network are of even degree and all the branches in the network are undirected, then a tour through all the branches without deadheading in the network can be found [22]. A tour through all the branches in the network, regardless of whether the network has directed branches, undirected branches, or both, is called an Euler tour (Edmonds and Johnson [11]). For directed networks, a sufficient condition for the existence of an Euler tour is that the number of branches directed into each node of the network is equal to the number of branches directed out of that node (Edmonds and Johnson [11]). In all cases the network is assumed to be connected.

The basic premise in all the procedures designed for solving branch routing problems with one vehicle is to add branches to the subgraph containing only the branches which have to be covered to guarantee that when all the branches are added, an Euler tour exists in the network. The branches added to the original network are selected to insure that the sum of the distances of all the branches added to the network is minimized (Edmonds and

Johnson [11], Liebman [9], Stricker [8], and Liebling [10]). We shall now discuss two variants of the single vehicle routing procedures for solving the mechanical broom routing problem involving more than one vehicle.

The first procedure finds a giant tour and busts this tour into a set of routes which are feasible with regard to the time capacity of the vehicle. The second method breaks the geographic area over which the routes are to be formed into a collection of contiguous regions, the number of such regions equal to the number of brooms to be routed, and then solves the branch routing problem over each of these subregions.

The first procedure can be broken down into the following steps. In the ensuing discussion let $G = (N,A)$ by the network representing the entire area where A includes the branches which need not be covered and $G' = (N,A')$ be the subgraph of G such that A' is exclusively the set of branches which have to be covered in the time period. It is assumed that G' is connected.

Step 1: In the subgraph G', find those nodes n_j such that the number of branches directed into n_j is greater than the number of branches out of this node. Call the difference s_j. Similarly, find those nodes m_k in G' such that the number of branches directed out of m_k is greater than the number of branches into this node. Call this difference d_k. It can easily be shown that $\sum s = \sum d$.

Step 2: In the total network $G = (N,A)$, find the shortest distance from each node n_j to each node m_k. Call this distance c_{jk}. For the problem to be feasible, each node n_j must have at least one branch directed out of it and each node m_k must have at least one branch directed into it in the total network G.

Step 3: We now have the ingredients for the following transportation problem:

$$\text{Min } z_0 = \sum c_{jk}\, x_{jk}$$

subject to

$$\sum_k x_{jk} = s_j \quad \text{for all } j$$

$$\sum_j x_{jk} = d_k \quad \text{for all } k$$

$$x_{jk} \geq 0.$$

Each node n_j can be thought of as a supply node of the transportation problem with supply s_j and each node m_k can be considered a demand node of the transportation problem with demand d_k. The travel time between supply node n_j and demand node m_k is c_{jk}.

(Having solved the transportation problem, Z_0 tells the minimum deadheading that is needed to find an Euler tour over those branches which have to be covered in the network. We now augment subgraph G' = (N,A') with a collection of branches (as found from the solution of the transportation problem) to derive a new network G" = (N,A") so that within G" there exists a feasible Euler tour. We shall find the Euler tour in G" and then bust this tour into feasible schedules for each of the mechanical brooms. This is carried out in Steps 4 to 6.)

Step 4: For each pair of nodes (n_j, m_k) such that $x_{jk} > 0$, we know the shortest path from n_j to m_k. Let this shortest path be (n_j, p_1), (p_1, p_2), ..., (p_{r-1}, p_r), (p_r, m_k). Add these branches on this shortest path from n_j to m_k to the branch set A', x_{jk} times. When all such branches are added we have the network G" = (N,A"). We note that in G", every node has the same number of branches coming into it as leaves it and the Euler tour in G" must cover all the branches in A".

Step 5: Utilize the procedure on page 44 of Edmonds and Johnson [11] to find the Euler tour in the network G".

Step 6: Take the Euler tour found in Step 5, start at any arbitrary starting point and partition it into feasible schedules for each of the mechanical brooms.

The basic procedure given in Steps 1 through 5 above is known although we are not acquainted with an English reference. We have, however, discussed this procedure in depth with Michael Held and Ellis Johnson of IBM, Jon Liebman of the University of Illinois and David Marks of M.I.T. We gratefully acknowledge their cooperation in this study.

The above method is now illustrated in the following example.

Example 4

We want to find an Euler tour in the following network (Figure 14). The networks G and G' are here the same.

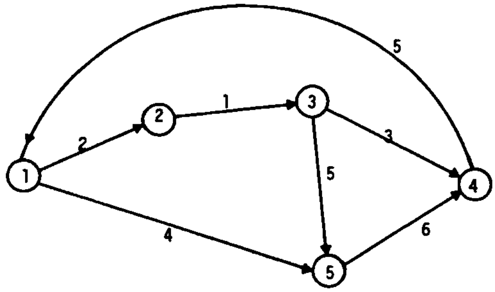


Fig. 14

Step 1: n_1 = node 4 $\quad s_1 = 1$
n_2 = node 5 $\quad s_2 = 1$
m_1 = node 3 $\quad d_1 = 1$
m_2 = node 1 $\quad d_2 = 1$

Step 2: $c_{11} = 8 \quad c_{12} = 5$
$c_{21} = 14 \quad c_{22} = 11$

Step 3: Solve the following transportation problem

$$\text{Min } Z_0 = 8x_{11} + 5x_{12} + 14x_{21} + 11x_{22}$$

subject to

$$\begin{aligned} x_{11} + x_{12} \qquad\qquad &= 1 \\ x_{21} + x_{22} &= 1 \\ x_{11} \qquad + x_{21} \qquad &= 1 \\ x_{12} \qquad + x_{22} &= 1 \\ x_{ij} \geq 0,\ i = 1,2,\ j &= 1,2 \end{aligned}$$

One optimal solution to the transportation problem is $Z_0 = 19$, $x_{11} = 1$, $x_{22} = 1$.

Step 4: The network G" becomes the following (Figure 15). Note that in G" the number of branches into each node equals the number of branches out of this node.

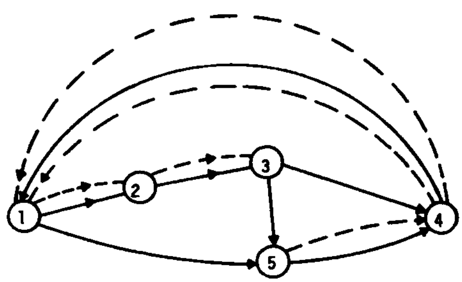


Fig. 15

Step 5: An Euler tour in this network is given in Figure 16.

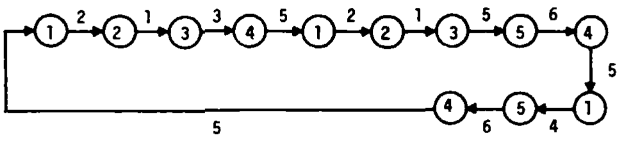


Fig. 16

Step 6: Suppose that we have two vehicles and the time that each vehicle has to cover its route is 25 units, then we end up with the following two routes (Figure 17).

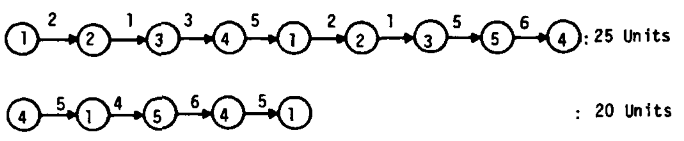


Fig. 17

Note above that on route 2 one of the two branches 4 → 1 can be left out since this branch need only be covered once. Furthermore, note that if the branch 5 → 4 on route 2 is to be cleaned then the branch 5 → 4 on route 1 can be deadheaded and, hence, left off. These end point conditions must be checked since alterations as above can reduce the deadheading time.

The above example shows that the two routes interact. It may be desirable to form routes that do not intersect except along the boundary. The second approach is formulated with this objective in mind. In the second approach, we first partition the network G' = (N,A') into contiguous regions such that the total required workload within each cluster is about the same and can be handled by a mechanical broom within its time restraints. One then carrys out Steps 1 to 5 of approach 1 on each of these subgraphs of G' to derive the Euler tours such that the deadheading time within each subgraph is minimized.

We close by biefly discussing a question that is not solved at this time. Approach 2 partitions the network G' = (N,A') into a set of contiguous subregions. However, in evaluating this partitioning we may encounter two problems:

(1) The Euler tour formed for a subregion after the deadheading branches are added may not be feasible in the sense that the time capacity constraints of the mechanical brooms may be violated.
(2) If all the Euler tours are feasible, then the total deadheading may not be minimized.

We, therefore, wish to develop a set of procedures which will allow us to go from one set of partitions to a second set of clusters in such a way that in the second set of clusters, the total deadheading time is less than in the first set of partitions. When this transformation can no longer be carried out, we then say that we have converged to what can be called a strong local optimum. How to design this transformation is currently under investigation.

APPENDIX

It is perhaps worth noting here several somewhat pathological features of the Clarke and Wright method. Because the technique is heuristic it is possible for it to lock onto locally optimal solutions which in some cases can be inferior to solutions obtained by even simpler and more naive methods. However, in practice the Clarke and Wright method has shown itself to be robust.

One could, for example, try to link nodes by a nearest neighbor rule which is, quite simply, to move from the dumpsite or depot to the closest node and from there to the next closest node, and so on, from node to node, until either the capacity or the time constraints are violated. In Figure 18, 19 and 20, we compare this nearest neighbor rule with that of Clarke and Wright and we see that the latter method comes out second best in this one case at least.

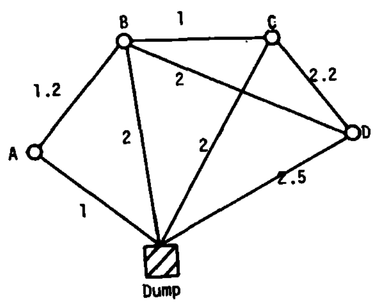
Savings List

$S_{B,C} = 3$

$S_{B,D} = 2.5$

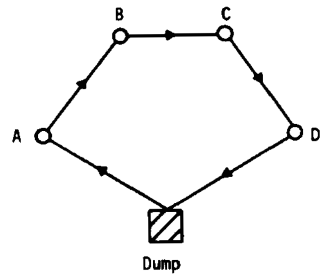
$S_{C,D} = 2.3$

$S_{A,B} = 1.8$



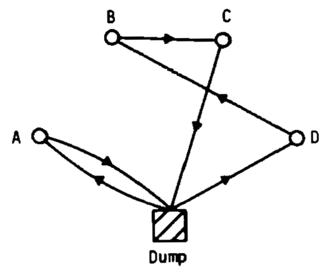
A network problem.
All numbers refer to travel times.
Vehicle travel is restricted to a maximum of 8 units.
There are no capacity constraints.

Fig. 18



Minimum time tour
using Nearest Neighbor Heuristic.
Total travel time is 7.9 units.

Fig. 19



Minimum time tour
using Clarke & Wright heuristic.
Total travel time is 9.5 units.
Note that hookups between C,D and A,B are not feasible.

Fig. 20

In order to coax the Clarke and Wright algorithm into finding better local optima it is possible to perturb the ordering of node pairs in the savings list. This is accomplished by increasing the distance from the depot or dump to one or more of the nodes and then resolving the problem subject to the original constraints. If the resulting solution is better we restore the perturbed distance to its original value and then compute the resulting payoff. It may appear strange that this procedure can be at all effective but sometimes it can be used to advantage as is illustrated by the following example.

Example 5

Consider a six node problem in which $d_{i,\text{dump}} = 6$ for $i = 1, \ldots, 6$ and for which each node has a unit demand. Vehicle capacity is 3 units and there is no time restriction. The distances between nodes is given by

$$d_{1,2} = 2$$
$$d_{3,4} = 2.3$$
$$d_{5,6} = 2.6$$
$$d_{2,3} = 3$$
$$d_{4,5} = 3.5$$

In this case the savings list and the resulting Clarke and Wright tours are exhibited in Figure 21a for a total travel distance of 42.9 units. We now let $d_{2,dump} = d_{3,dump} = 10$. The savings list is then reordered to give

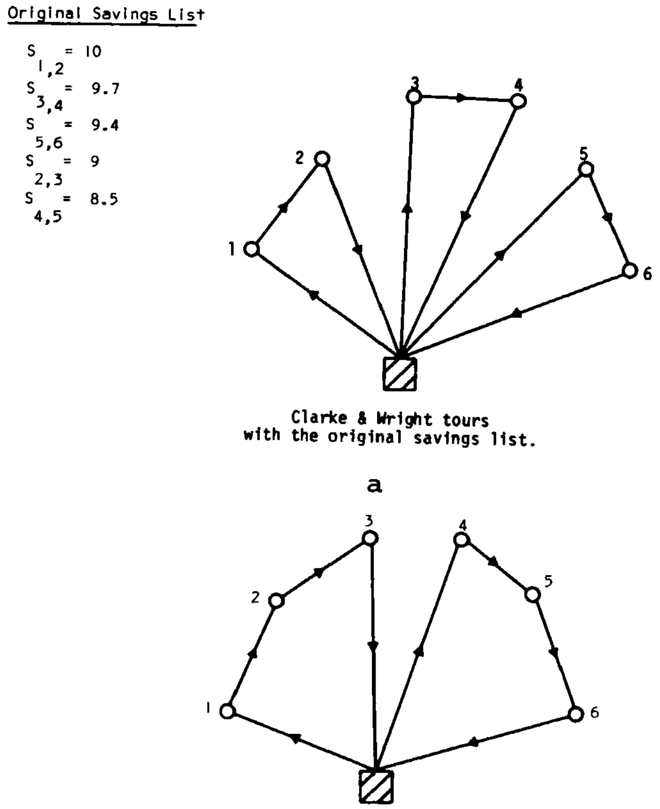
$$s_{2,3} = 17$$
$$s_{1,2} = 14$$
$$s_{3,4} = 13.7$$
$$s_{5,6} = 9.4$$
$$s_{4,5} = 8.5$$

In this case the resulting tours are displayed in Figure 21b. If we now reassign the value of 6 to $d_{2,dump}$ and $d_{3,dump}$ the total tour time is 35.1, a considerable improvement over the previous set of tours. In fact, 35.1 is the global optimum in this case.

Another way to improve the Clarke and Wright idea is to reapply the savings concept to the tours which result from a first pass through the original savings list. However, the savings idea is now generalized to apply between tours rather than to just nodes. For instance, in the example of Figure 21 one lets

$$s = d_{3,4} + d_{dump,2} + d_{dump,5} - d_{2,3} - d_{4,5}$$

and, since s is positive, the three tours of 21a are compressed into the two tours of 21b. Our experience has shown, however, that even though this procedure can result in improved tours, the amount of additional computation involved is large compared to the expected payoff and for this reason the extended savings idea has not been formalized.



Clarke & Wright tours
with the original savings list.

a

Clarke & Wright tours
with perturbed savings list.

b

Fig. 21

ACKNOWLEDGMENT

A number of our colleagues at Stony Brook have contributed substantially to portions of this work, notably Stanley Altman and Naresh Bhagat. Their involvement has been an essential ingredient to the success of the Stony Brook routing studies.

REFERENCES

1. Bodin. L., "Towards a General Theory of Manpower Scheduling - Parts 1 and 2," *Journal of Urban Analysis*, Vol. 1, No. 2, 1973.

2. Altman, S., E. Beltrami, S. Rappaport, and G. Schoepfle, "A Nonlinear Programming Model for Household Refuse Collection," *IEEE Systems, Man and Cybernetics*, Vol. SMC-1, July 1971, pp. 289-291.

3. Krone, M. J., "Heuristic Programming Applied to Scheduling Problems," unpublished doctoral dissertation, Princeton University, 1970.

4. Marks, D., J. Cohon, H. Moore, and R. Stricker, "Routing for Municipal Services," *Proc. of the 6th Annual ACM Urban Symposium,* 1971.

5. Held, M. and R. Karp, "The Travelling Salesman Problem and Minimum Spanning Trees," *Operations Research,* Vol. 18, No. 6, pp. 1138-1162.

6. Lin, S., "Computer Solutions of the Travelling Salesman Problem," *Bell System Technical Journal 44,* 1965, pp. 2245-2269.

7. Little, J., K. Murty, D. Sweeney, and C. Karel, "An Algorithm for the Travelling Salesman Problem," *Operations Research,* Vol. 11, No. 6, 1963, pp. 972-989.

8. Stricker, R., "Public Sector Vehicle Routing: The Chinese Postman Problem," doctoral thesis at M.I.T. (Department of Electrical Engineering), 1970.

9. Liebman, J., "Notes on Edmonds' Matching and White's K. Matching," unpublished class notes, June 1972.

10. Liebling, T., "Graphen Theorie in Planungs-und Touren Problemen," lecture notes in *Operations Research,* Vol. 21, Springer-Verlag, 1970.

11. Edmonds, J. and E. L. Johnson, "Matching, Euler Tours and the Chinese Postman," *Mathematical Programming,* Vol. 5, No. 1, 1973, pp. 88-124.

12. Christofides, N., "The Optimum Traversal of a Graph," *Report 71/16* of the Imperial College of Science (London), November 1971.

13. Newton, R. and W. Thomas, "Developing a Computer Program for Bus Routing: Final Report," State University of New York Research Foundation, Albany, New York, July 1970.

14. Clarke, G. and J. W. Wright, "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points," *Operations Research,* 12, 1964, pp. 568-581.

15. Eilon, S., Watson-Gandy, and N. Christofides, *Distribution Management*, Hafner Publishing Co., New York, 1971.

16. Bennett, B. and D. Gazis, "School Bus Routing by Computer," to appear in *Transportation Research*.

17. Altman, S., N. Bhagat and L. Bodin, "Extension of the Clarke and Wright Algorithm for Routing Garbage Trucks," presented at the TIMS 8th International Meeting, Washington, D.C., March 1971.

18. Benzer, S., "On the Topology of the Genetic Fine Structure," *Proc. Nat. Academy of Science USA*, 45, 1959, pp. 1607-1620.

19. Kendall, D. G., "Incidence Matrices, Interval Graphs and Seriation in Archeology," *Pacific J. Math*, 28, 1969, pp. 565-570.

20. Tucker, A., "Perfect Graphs and an Application to Optimizing Municipal Services," submitted for publication.

21. Krolak, P., W. Felts, and J. Nelson, "A Man-Machine Approach Toward Solving the Generalized Truck-Dispatching Problem," *Transportation Science*, Vol. 6, No. 2, 1972, pp. 149-170.

22. Ore, O., *Graphs and Their Uses*, Random House, New York, 1963.

23. Held, M. and R. Karp, "The Travelling Salesman Problem and Minimum Spanning Trees: Part II," *Mathematical Programming*, Vol. 1, 1971, pp. 6-25.

24. Orloff, C., "Routing and Scheduling a Fleet of Vehicles to/from Central Facilities - The School Bus Problem," Ph.D. dissertation, Cornell University, 1972.

Research was in part supported under the NSF Grant GI-5.

Paper received September 19, 1972.