Heuristic Methods Applied to Orienteering

Article *in* Journal of the Operational Research Society · September 1984

DOI: 10.1057/jors.1984.162

Theodore Tsiligiridis
Agricultural University of Athens
79 PUBLICATIONS **1,331** CITATIONS
SEE PROFILE

J. Opl Res. Soc. Vol. 35, No. 9, pp. 797–809, 1984
Printed in Great Britain. All rights reserved

0160-5682/84 $3.00 + 0.00
Copyright © 1984 Operational Research Society Ltd

Heuristic Methods Applied to Orienteering

T. TSILIGIRIDES
Department of Mathematics, Agricultural College of Athens

A general description of the sport of orienteering in its two forms known as orienteering event and score orienteering event is given. The similarities between the latter and the generalized travelling salesman problem are underlined, and a common mathematical model is deduced. A 'good', if not 'optimal' solution of the corresponding problem by means of an efficient and computationally feasible method is derived and discussed. This solution is obtained by the use of either of two algorithms based on approximate methods. The two algorithms are given, and their efficiency is tested on a number of illustrative problems. Finally, a comparison between the two forms of the sport of orienteering is made.

INTRODUCTION

ORIENTEERING is a sport which is a mixture of cross-country running and navigation through a forest, using a map and compass. Competitors start at intervals of, say, a minute and aim to find a number of 'control points' which have been placed in the forest and whose locations are marked on the competitors' maps.

The most usual form of orienteering is the orienteering event (O.E.), in which the competitor must visit every control point in the order given and the winner achieves this in the minimum time. There is also a second form of this sport, known as a score orienteering event (S.O.E.), in which competitors do not have to visit all controls. Each control point has a score or number of points allocated to it, and the competitor's aim is to maximize his total score within a certain prescribed time limit.

In addition to the requirements of physical fitness and good navigational ability for the standard form of event, S.O.E. requires the competitor to perform an optimization problem, to realize his limitations as to which and how many control points he should reach and, from a knowledge of the scores allocated to each control, to plan his route so as to use up almost all the available time. Severe penalty points are charged to competitors who return later than the time limit. Lower scores are usually allocated to those control points near to the start and finish area and larger scores to those further away, although account is also taken of how difficult the control is to find and how distant the control is from other controls.

A good competitor should have a very good idea of the total distance he can run in the time available. Therefore, by carefully studying the control positions and scores, he should plan his route accordingly. His own choice should be to stay close to the start and finish area and pick up a lot of low-scoring controls, or to venture out towards the edge of the map and pick up a smaller number of higher-scoring controls. Having decided on his overall strategy, he may need to revise it as his run progresses by looking at the remaining time, his present position and the positions of the unvisited controls. An orienteer knows the time needed to run between any two control points. Therefore these time intervals are completely known.

The same mathematical model could have been produced from a different physical problem. Consider, for example, the problem in which a travelling salesman has a number of cities which he could visit, but unlike the travelling salesman problem (T.S.P.), he does not have to visit each and every one. He knows the number of sales he can expect to make in each city and therefore he wishes to plan his route so as to maximize his total number of sales, whilst keeping the total length of his route within the distance he can travel in one day (or week). This formulation could be referred to as a generalized travelling salesman problem (G.T.S.P.), in which the cities take the role of the control points and the numbers of sales take the place of the scores. Therefore the objective of the G.T.S.P.

is similar to that of S.O.E. and is stated in terms of maximizing the sales (or scores in S.O.E.), so that the time needed to visit the corresponding nodes is the smallest one which does not exceed the maximum time available.

The above S.O.E. could be considered, alternatively, by placing bounds upon the length of the distance of each route. In this case, the objective is to find the shortest route when the length of the distance which will be covered does not exceed a predetermined value $T_{\max}$, say. Of course, there is a direct proportionality between the distance travelled by a competitor and the time needed to cover this distance, i.e. longer distances require more time. Moreover, assuming the same capacity for all the individuals who may want to achieve this objective, the above proportionality strongly influences their decision, namely, which node they will visit first.

As is obvious from the above, there is no actual difference if we consider $T_{\max}$ as the distance or the time of the route. To avoid any confusion on this point, the value $T_{\max}$ will be used throughout this work as the bound of the length of the distance which may be travelled when a route is given. Of course, the score $S(j)$ attached to each node N_j; $j = 1, 2, \ldots,$ NPTS represents the amount of benefit which will be obtained if this node is visited, whilst the elements of the symmetric cost matrix $C(i, j)$; $i, j = 1, 2, \ldots,$ NPTS represent the corresponding cost from node N_i to N_j. Therefore a measure of the type $S(i)/C(i, j)$ gives a clear idea of what is expected to result if the route from node N_i to node N_j is made.

Let the maximum length distance $T_{\max}$ be temporarily known and fixed. It is possible that the given $T_{\max}$ is not sufficient to complete a tour. In this case, the aim is to find an optimal tour satisfying the above requirements. For this, the use of exact methods is suggested, but since no exact method is still attractive when the number of nodes is large, in practice approximate methods are always used. The G.T.S.P. suggests two methods to be used, and in both cases a 'good' solution which might be the 'optimal' one will be produced. Clearly, the answer will be preferable if the solution achieved is efficient in terms of computational time required.

COMPUTATIONAL ALGORITHMS FOR THE S.O.E.

The stochastic algorithm (S-algorithm)

To describe this, it will be necessary to give the following notations:

NPTS:	the number of nodes including the origin (start) and the end (finish); the start node is always 1 and the finish is always NPTS;
LAST:	the node which has just visited;
NFEAS—f:	indicates the number of feasible nodes which could be visited next, when someone is at a specific node;
$A(j)$:	a measure of 'desirability' for each feasible node $N_j, j = 1, 2, \ldots,$ NPTS;
T:	the time/distance covered up to the LAST node (i.e. the present time/distance);
DEL(j):	an overall measure for location N_j of its 'nearness' to all other locations; actually, $\text{DEL}(j) = \sum_{i=1}^{\text{NPTS}} \frac{S(i)}{C(j, i)}, j = 1, 2, 3, \ldots, \text{NPTS}, i \neq j;$
ALPHA—a:	a weight factor parameter;
POWER—r:	a power factor parameter;
LENGTH—l:	indicates the number of feasible nodes considered.

The method implemented by the S-algorithm is based on generating many routes and picking the best. It uses a stochastic, or so-called Monte Carlo method to take a meaningful sample and to facilitate the selection process itself. This has been achieved by building up a route using a quite simple hierarchy. The selection list is constructed according to a criterion which takes account of the Euclidean distance and the score attached to the nodes, namely, the cost to be payed and the benefit gained, respectively, by visiting a specific node.

The measure of feasible nodes to be visited next when a specific node is visited is given by the formula:

$$A(j) = \left\{ \frac{S(j) + E}{C(\text{LAST}, j)} \right\}^r \quad ; \quad \text{for each unvisited node } N_j, \quad j \neq \text{LAST},$$

where

$$E = a \cdot \{T_{\max} - T - C(\text{LAST}, j) - C(j, \text{NPTS})\} \cdot \text{DEL}(j)$$

is designed to emphasize the remaining time/distance along with the overall score of the location to be visited next. The ordered values of $A(j)$, i.e. the values

$$A(1) > A(2) > A(3) > \ldots > A(k) > \ldots > A(f)$$

with

$$k = \min(l, f),$$

give the hierarchy in accordance with the sample which was taken.

The calculation of the probabilities

$$P(j) = A(j) \bigg/ \sum_{q=1}^{k} A(q)$$

leads to this choice of j which satisfies the inequality $P(j) \geqslant Y$, where Y is the random number generated from the uniform distribution $U(0,1)$. It is clear that this j will show the next location to be inserted in the route. The above process is repeated by reconsidering the new situation (with the new route), until the remaining time becomes so small that it is not possible to visit any other node except the NPTS control.

According to the above, three parameters are specified so that the model will yield the best results. Preliminary thoughts suggest that increasing the range of the links available at any time expands the population, since

$$k = \min(l, f),$$

and hence the potential for generating a near-optimal answer. Therefore large values of the parameter LENGTH seem preferable to small ones, although this causes an increase in the execution time required.

In this paper, three problems are considered, including 32, 21 and 33 locations, respectively. Their input consists of the distances represented by the cost matrices $C(i, j)$ produced from the Cartesian coordinates which appear in Table 1, along with the scores awarded to the nodes. To calibrate the model, the input of the 32-locations problem is used. A series of trials is carried out by varying the length of the ordered selection list $A(j)$ from two to six nodes, and for each run a different set of random numbers Y is generated. The number of routes produced is kept fixed, at about 3000. By comparing the different results obtained from the different values of $T_{\max}$, the most impressive values of the parameters are selected ($l = 4.0$, $r = 4.0$, $a = 0.0$). Of course, this stabilization of the parameters is not unique but it seems the best and most robust. In this case, the formula of $A(j)$ is reduced to

$$A(j) = \left\{ \frac{S(j)}{C(\text{LAST}, j)} \right\}^{4.0} \quad ; \quad \text{for each unvisited node } N_j, \quad j \neq \text{LAST}.$$

Therefore, the 'nearness' $\text{DEL}(j)$ of each location N_j to the rest seems to have no effect on the measure of 'desirability' $A(j)$.

The Deterministic Algorithm (D-Algorithm)

The D-algorithm builds up a route in a similar way to that of Wren and Holiday's

TABLE 1

Node	Problem 1			Problem 2			Problem 3		
I	$X(I)$	$Y(I)$	$S(I)$	$X(I)$	$Y(I)$	$S(I)$	$X(I)$	$Y(I)$	$S(I)$
1	10.50	14.40	0	4.60	7.10	0	19.10	24.30	0
2	18.00	15.90	10	5.70	11.40	20	12.60	24.90	20
3	18.30	13.30	10	4.40	12.30	20	14.40	28.00	20
4	16.50	9.30	10	2.80	14.30	30	16.90	28.10	20
5	15.40	11.00	10	3.20	10.30	15	20.70	28.20	20
6	14.90	13.20	5	3.50	9.80	15	12.50	26.60	20
7	16.30	13.30	5	4.40	8.40	10	21.80	27.30	20
8	16.40	17.80	5	7.80	11.00	20	12.50	22.60	20
9	15.00	17.90	5	8.80	9.80	20	22.50	17.00	30
10	16.10	19.60	10	7.70	8.20	20	19.90	15.00	30
11	15.70	20.60	10	6.30	7.90	15	14.90	15.10	30
12	13.20	20.10	10	5.40	8.20	10	11.50	18.60	30
13	14.30	15.30	5	5.80	6.80	10	12.40	29.80	30
14	14.00	5.10	10	6.70	5.80	25	17.80	28.10	30
15	11.40	6.70	15	13.80	13.10	40	9.10	29.80	40
16	8.30	5.00	15	14.10	14.20	40	10.00	32.60	40
17	7.90	9.80	10	11.20	13.60	30	13.90	33.10	40
18	11.40	12.00	5	9.70	16.40	30	19.95	10.30	40
19	11.20	17.60	5	9.50	18.80	50	15.20	8.00	40
20	10.10	18.70	5	4.70	16.80	30	14.70	31.20	50
21	11.70	20.30	10	5.0	5.60	0	7.40	36.50	50
22	10.20	22.10	10				21.00	25.50	50
23	9.70	23.80	10				18.00	25.30	10
24	10.10	26.40	15				19.50	24.70	10
25	7.40	24.00	15				21.40	21.80	10
26	8.20	19.90	15				16.00	21.40	10
27	8.70	17.70	10				18.65	26.20	10
28	8.90	13.60	10				17.90	28.90	10
29	5.60	11.10	10				14.30	19.90	20
30	4.90	18.90	10				17.00	19.00	20
31	7.30	18.80	10				10.80	21.00	20
32	11.20	14.10	0				15.70	23.70	10
33							18.20	24.00	0

method[1] for a vehicle-scheduling problem. No capacity limit exists, and only one depot is available. It yields a 'near-optimal' or 'good' solution, without using random numbers or probabilities in order to construct the list of feasible locations to be visited next. Routes are constrained on separate sectors of the map according to some predetermined rules. These rules constrain the 'initial' route to be built up in a specified sector and to have a petal-form structure. Clearly, the above structure guarantees that the increase in distance is kept to a minimum.

Every time, the sector is determined by two concentric circles and an arc of known length (see Figure 1). By varying the two radii of the circles, many rings are obtained. To examine different possibilities, the rotation of the axes four times with $\pi/2$ difference is made. In particular, 12 possibilities of radii, in each rotation, are examined, and since only four rotations with difference $\pi/2$ can be made, the total examination includes 48 cases in each run.

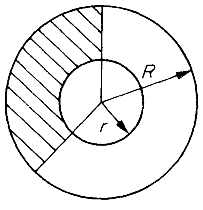


FIG. 1

The process of building up the route in a specified ring will stop either if all the nodes in that ring have been visited, or if it is not possible to visit any other node of the same ring, without causing a violation of the $T_{\max}$ constrain. The method provides an opportunity of examining many sectors of the map in a quite sophisticated way and to be more sensitive in cases where there are clusters of points in some parts of the map. Thus overlaps are avoided.

The route-improvement algorithm [(*R-I*)-*algorithm*]

The main purpose of this algorithm is to improve a given route by using three different approaches. Any 'initial' route is realigned, either by using a tour-to-tour improvement method or by considering the introduction of new nodes in such a way that there is no violation on the $T_{\max}$ constraint.

The algorithm tries to improve the 'initial' route by making some interchanges between pairs of nodes on the route. Pairs of nodes on the route are interchanged in two ways. To explain it further, suppose that the list array which gives the route is

1–6–8–3–9–30–32,

where 1 is the 'start' and 32 is the 'finish' node (see Figure 2). If the interchange of two nodes, say 6 and 9, is implemented, then there are two possible routes for the algorithm.

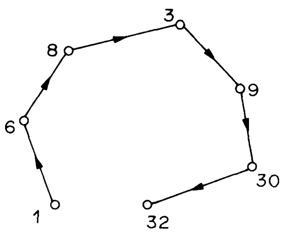


Fig. 2

These routes are

1–9–8–3–6–30–32 (see Figure 3)

and

1–9–3–8–6–30–32 (see Figure 4).

That is, the order of those nodes between 6 and 9 either remains the same or is reversed. Clearly, the total score remains the same, but there will be a change in the total travelled time/distance T, say. If both types of interchange would result in an increase in T, the interchange of that pair of the nodes is not made; if either, or both, would result in a decrease in T, the better interchange is implemented.

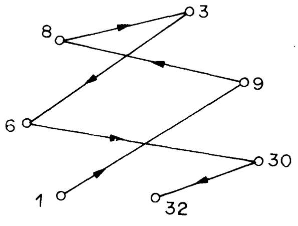


Fig. 3

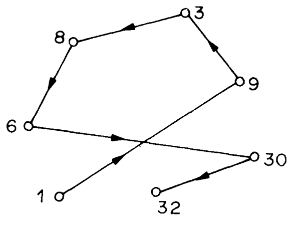


Fig. 4

Next, new unvisited nodes are considered for inclusion in the 'initial' route. Obviously, the total score will increase if the score attached to a node is positive, as is usually the case. The insertion will not be made if the total time/distance T would be increased beyond $T_{\max}$. If there is a choice of positions to insert the new node without exceeding the bound $T_{\max}$, the best position is selected, namely, the one which produces a minimal increase in T.

The final approach tries to improve a given route by inserting a new unvisited node whilst, at the same time, excluding one presently visited node. Nodes are included and excluded on the route in two ways and in accordance with their order in the list array which gives the route. If, for example, the list array is

1–14–18–22–6–8–3–9–4–30–32

and the decision is to insert the node 7 between those of 6 and 8 (see Figure 5), then the route will become either

1–14–22–6–7–8–3–9–4–30–32

if the excluded node 18, say, stands on the left side of the node 6 (see Figure 6), or

1–14–18–22–6–7–8–3–9–30–32

if the excluded node 4, say, stands on the right side of the node 8 (see Figure 7).

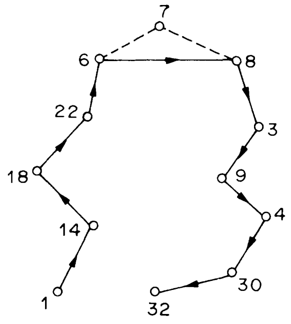


Fig. 5

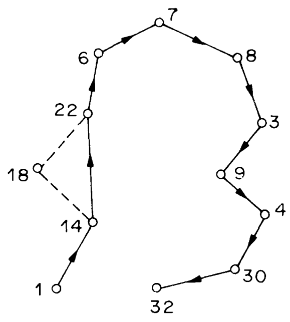


Fig. 6

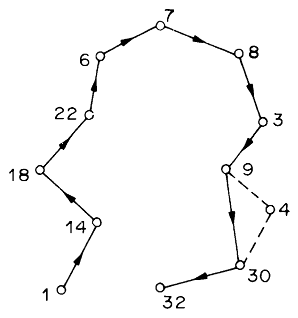


Fig. 7

The total score will increase if the score attached to the included node is greater than the score attached to the excluded one. The insertion will not be made if the total score is not greater than the previous one, or if the total distance/time T, produced from the changes in the nodes, would be increased beyond $T_{\max}$. Again, if there is a choice of positions to insert or exclude, the best pair of positions is selected, i.e. the pair which produces a minimal increase in T.

Generally speaking, there is some restriction on the number of nodes available in order to use the (R-I)-technique. For instance, in the first approach, interchanges cannot be made if the number of nodes in the route is less than four, whilst the third approach requires no less than three nodes. Obviously, the second approach does not require a specific number of nodes since it actually builds up the route.

RESULTS

Any resulting route coming from either the S- or the D-algorithm can be considered as an 'initial' route for the (R-I)-algorithm. The new realigned route is a 'good' solution for either the stochastic or the deterministic approach (see Figure 8). In this case, the results are superior to the corresponding initial ones. If the (R-I)-algorithm cannot make any

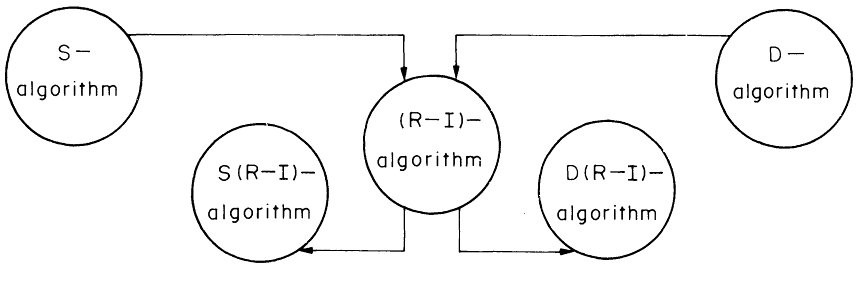


Fig. 8

improvement in the 'initial' route, the program outputs the same route. To verify the above, a comparison of the results obtained from (S-, S(R-I)-) and from (D-, D(R-I)-) algorithms is made. Numerical results are presented in Tables 2–4. The $T_{\max}$, the maximum score and the distance T, have been presented in such a way that a comparison between the different algorithms used can be easily made. The resulting routes are not reported, but indicative examples are given for the 32-locations problem in Figures 9 and 10.

A comparison between the D- and S-algorithms shows the main difference in their framework. The D-algorithm builds up the routes in pre-specified sectors of the map using some pre-determined rules, which constrain the routes in order to have a petal-form structure. The S-algorithm builds up the routes covering all the map area and using random numbers or probabilities in order to construct the list of feasible nodes. As expected, the results obtained from the S-algorithm are superior to those of the D-algorithm (see Tables 2–4). There is no appreciable difference in their execution time, although this time is increasing with the number of nodes.

TABLE 2. 32-LOCATIONS PROBLEM

	D-algorithm		S-algorithm		D(R-I)-algorithm		S(R-I)-algorithm	
$T_{\max}$	Score	Distance	Score	Distance	Score	Distance	Score	Distance
5	—	—	10	4.2	—	—	—	—
10	—	—	15	7.0	—	—	—	—
15	—	—	45	14.3	—	—	—	—
20	40	19.99	65	19.6	65	19.60	65	19.60
25	65	24.33	90	24.7	90	24.82	90	24.65
30	80	28.94	110	28.8	110	28.80	110	28.80
35	105	34.23	135	34.1	135	34.08	135	34.08
40	105	36.37	150	38.0	150	38.02	150	38.02
46	130	45.33	175	44.5	175	44.51	175	44.51
50	140	48.60	190	49.9	190	44.53	190	49.78
55	160	51.96	205	54.8	200	52.86	205	54.08
60	175	56.78	220	58.9	220	59.65	220	58.93
65	200	63.91	240	63.9	240	63.82	240	63.82
70	200	69.84	255	68.8	260	69.13	260	69.13
73	205	71.07	260	72.7	265	70.73	265	70.73
75	210	73.35	270	74.7	275	74.66	275	74.66
80	220	78.17	275	79.0	280	77.73	280	78.34
85	235	82.41	280	82.2	285	81.33	285	81.82

TABLE 3. 21-LOCATIONS PROBLEM

	D-algorithm		S-algorithm		D(R-I)-algorithm		S(R-I)-algorithm	
$T_{\max}$	Score	Distance	Score	Distance	Score	Distance	Score	Distance
15	100	14.84	120	14.88	120	14.25	120	14.88
20	130	18.79	190	18.81	200	19.88	200	19.88
23	130	18.79	205	22.75	210	22.65	210	22.65
25	145	24.84	230	24.89	230	24.13	230	24.26
27	160	26.00	230	24.89	230	24.13	230	24.26
30	190	29.74	250	29.09	265	29.85	260	29.49
32	195	31.54	275	30.59	300	31.63	300	31.63
35	200	34.17	315	34.25	320	34.51	320	34.51
38	210	36.81	355	37.62	355	37.62	355	37.62
40	240	39.91	395	39.78	385	39.56	395	39.78
45	270	44.37	430	44.28	450	44.44	450	44.44

TABLE 4. 33-LOCATIONS PROBLEM

	D-algorithms		S-algorithms		D(R-I)-algorithm		S(R-I)-algorithm	
$T_{\max}$	Score	Distance	Score	Distance	Score	Distance	Score	Distance
15	70	14.50	100	13.82	100	13.82	100	13.82
20	120	17.91	140	19.25	140	19.25	140	19.25
25	140	24.65	190	24.66	190	24.66	190	24.66
30	180	27.50	240	29.60	240	29.60	240	29.60
35	220	32.11	290	34.93	280	34.15	290	34.93
40	240	37.32	330	39.65	340	39.70	330	39.65
45	280	44.39	370	44.04	370	44.04	370	44.04
50	310	49.03	410	48.35	420	49.58	420	49.58
55	340	54.73	450	54.31	440	54.61	460	53.97
60	350	57.05	500	59.07	500	59.07	500	59.07
65	410	64.51	530	64.65	530	63.81	530	64.65
70	460	66.68	560	69.39	560	68.75	560	69.39
75	490	72.75	590	74.78	600	74.32	590	74.78
80	510	75.47	640	79.80	640	79.42	640	79.80
85	520	82.95	670	83.61	670	83.61	670	83.61
90	580	86.30	690	89.07	700	89.14	700	89.13
95	610	92.42	720	94.40	740	94.96	730	94.67
100	640	98.35	760	99.67	770	99.09	760	99.10
105	660	103.48	770	104.55	790	103.65	790	104.89
110	680	109.72	790	107.97	800	106.19	800	108.31

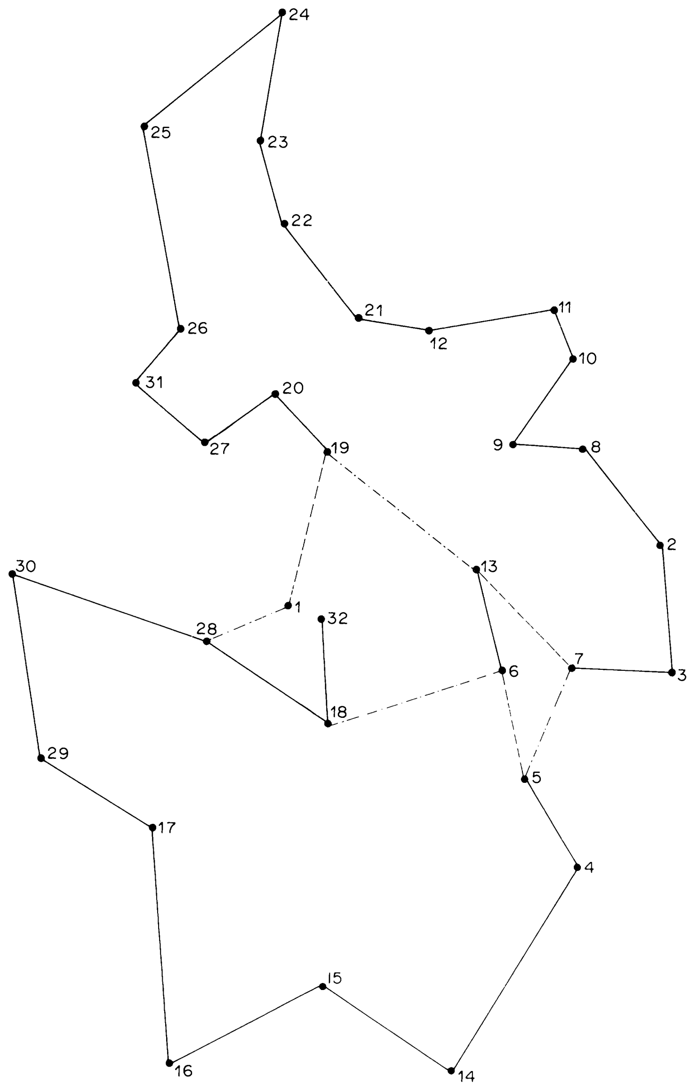


FIG. 9. $T_{\max}$: 85/score: 285/T: 81.82
S(R-I): 1–28–30–29–17–16–15–14–4–5–7–3–2–8–9–10–11–12–21–22–23–24–25–26–31–27–20–19–13–6–18–32.

$T_{\max}$: 85/score: 285/T: 81.33
D(R-I): 1–19–20–27–31–26–25–24–23–22–21–12–11–10–9–8–2–3–7–13–6–5–4–14–15–16–17–29–30–28–18–32.

A comparison between S(R-I)- and D(R-I)-algorithms shows that there is some similarity in their framework. Both algorithms use the (R-I)-technique to improve their results. The main difference is the usage of different initial routes. The first builds up a route in accordance with the most likely to be visited next node, while the second takes account of the position of the nodes on the map. Despite the fact that different initial routes are used, the algorithms seem to give nearly the same results (see Tables 2–4). Therefore, these algorithms complement each other; there is not much difference in their execution time. More precisely, the D(R-I)-algorithm seems more tractable than the S(R-I)-algorithm in terms of computational time. Of course, the execution time increases with the number of nodes. The programs have been written in Fortran IV and executed on an ICL 1906 S machine.

SUMMARY AND CONCLUSIONS

From an appraisal of the work cited above, it is clear that the S.O.E. is closely related to the G.T.S.P. and the heuristic methods it uses. As is well known, the 'savings' heuristic

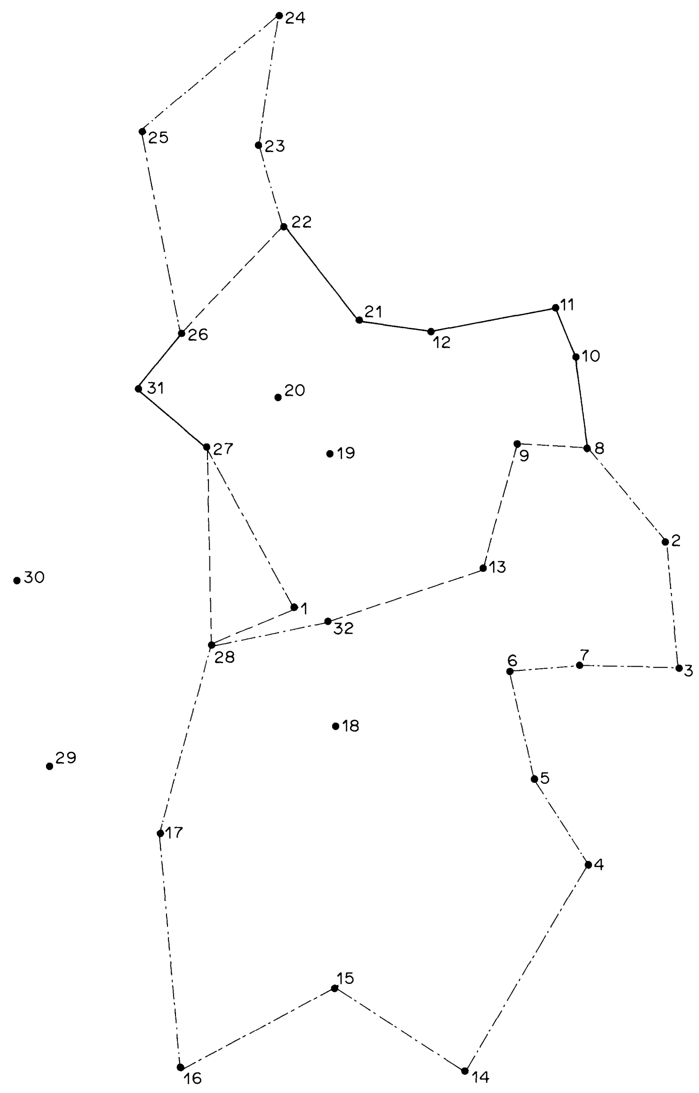


FIG. 10. $T_{\max}$: 30/score: 110/T: 28.80

S, S(R-I), D(R-I): 1–28–27–31–26–22–21–12–11–10–8–9–13–32.

$T_{\max}$: 65/score: 240/T: 63.82

S, S(R-I), D(R-I): 1–27–31–26–25–24–23–22–21–12–11–10–8–2–3–7–6–5–4–14–15–16–17–28–32.

rule yields acceptable results, which can be improved by systematic deviations.[2–4] Moreover, ‘good’ solutions cannot be obtained without explicitly constructing many alternatives which deal with the time/distance consuming, ‘trial and error’ evaluation procedures.[1,5–7] The points above suggest that the Monte Carlo simulation can be used as a powerful alternative to the studied problem by means of the S(R-I)-algorithm. This has been achieved by adopting in a similar manner the ‘savings’ principle, in so far as nodes become available for inclusion in a route and in accordance with their choice and standing in the ‘savings’ hierarchy.

It has been shown in previous works[1,2,8–10] that tours found on a ‘savings’ basis have a propensity to originate at the peripheral locations and to grow in a predominantly ‘circumferential’ manner with respect to the depot. Therefore the ideal route structure is based alternatively upon a ‘petal’ formation by means of the D(R-I)-algorithm. The directionality of each tour is determined by a random change, and the regulation of the number of solutions produced during the simulation run presumably constitutes a very simple and direct mechanism for controlling cost effectiveness.

A RELATED PROBLEM

Description of the problem

Consideration has so far been given to the efficiency of algorithms for solving an S.O.E. with the scores allocated to the control points and the value of $T_{\max}$ fixed. Both the Monte Carlo and the deterministic route-generation programs have been shown to work satisfactorily, especially when used in conjunction with the (R-I)-algorithm. Therefore the objective of this work, that of finding an efficient and computationally feasible method of finding a 'good', if not 'optimal' solution to the S.O.E., has been successfully completed. Next, a study of the influence of the node scores on the routes found will be conducted.

In any particular problem, the node scores are specified. If the real problem is a G.T.S.P. in which a certain number of sales to be made in each city has been fixed, then these nodes could conceivably take any values. However, in the original formulation of S.O.E., the control points furthest away from the start/finish area generally have the highest scores, and those near the start/finish tend to have lower scores.

In the more common form of orienteering, the sequence of control points to be visited is fixed, and the winner is the one who completes the course in the minimum time $t_{\min}$, say. A competitor who successfully completes the course in a time

$$t \leqslant 1.25 \cdot t_{\min}$$

is said to have achieved a gold standard; if

$$1.25 \cdot t_{\min} < t \leqslant 1.5 \cdot t_{\min},$$

he is said to have achieved a silver standard; and if

$$1.5 \cdot t_{\min} < t \leqslant 2.0 \cdot t_{\min},$$

he is said to have achieved a bronze standard.

In an S.O.E., the allowed $T_{\max}$ is fixed and the route is variable. Suppose the winner accumulates a total of $S'_{\max}$ points. Then the same gold, silver and bronze standards apply, but in this case based on the winner's score $S'_{\max}$. If a competitor scores a total of S points, he is awarded:

a gold standard if	$0.8 \cdot S'_{\max} \leqslant S,$
a silver standard if	$(2/3) \cdot S'_{\max} \leqslant S < 0.8 \cdot S'_{\max}$, and
a bronze standard if	$(1/2) \cdot S'_{\max} \leqslant S < (2/3) \cdot S'_{\max}.$

It should be noted that the coefficients of 0.8, 2/3 and 1/2 are the reciprocals of the coefficients 1.25, 1.5 and 2.0 of the other type of event, respectively.

Orienteering is a sport which prides itself on being logical, having an internal set of rules and standards. There has recently been some debate about whether the various standards are easier to achieve in one type of event than the other. If the standards are consistent, this implies that the score S which the competitor achieves increases linearly and proportionally with the distance/time he can cover in the allowed $T_{\max}$. To express the same idea in an equivalent way: if the variation of $T_{\max}$ is allowed, the best score achievable within that time/distance is proportional to $T_{\max}$ (see Figure 11).

Solution of the problem

As has been pointed out above, when the allowed time/distance $T_{\max}$ is sufficiently great, a competitor will be able to visit all the control points and to achieve the full score $S^*_{\max}$, say, in the shortest time/distance $T^*_{\max}$, say. In most events the winner almost manages this. Thus the requirement for comparability of the standards in the two types of event can be summarized in the following manner. Over the range $(1/2) \cdot T^*_{\max}$ to $T^*_{\max}$ the best achievable score S should be proportional to $T_{\max}$. Therefore it is sufficient to study the form of this relationship for the three cases and then try different schemes allocating scores to control points in order to see what effect (if any) this has upon the relationship.

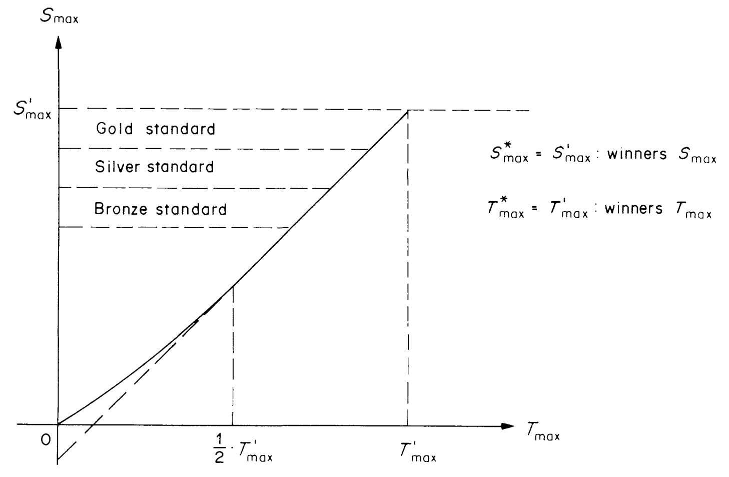

S_{max}
S'_{max}
Gold standard
Silver standard
Bronze standard
$S^*_{max} = S'_{max}$: winners S_{max}
$T^*_{max} = T'_{max}$: winners T_{max}
0
$\frac{1}{2} \cdot T'_{max}$
T'_{max}
T_{max}

FIG. 11

The main indicator of a desirable relationship will be an intercept close to zero, when a linear regression line has been fitted for S on T_{max} over the range $T^*_{max}/2$ to T^*_{max}. It is required that the intercept of the fitted regression line lies inside an interval $(-k, +k)$, with an error $\lambda\%$. By calculating the three intervals of the corresponding cases, it will be possible to examine whether the scores attached to the nodes give satisfactory results or whether it is necessary to produce new score shemes. To obtain the values of the intercepts, the best-fitted lines over the interval $[T^*_{max}/2, T^*_{max}]$ are derived using the least-square criterion. If T'_{max} is the shortest time/distance travelled by the winner who achieves a total score of S'_{max}, then the intercepts α of the regression lines must satisfy the inequality:

$$|\alpha| \leqslant 0.05 \cdot S'_{max}$$

by admitting an error of $\lambda\% = 10\%$.

Results obtained from the allocation and reallocation of the scores

The values of S'_{max} for the 32, 21 and 33 locations problems have been found to be 285, 450 and 800, respectively. Therefore the corresponding intervals are:

$$(-14.25, +14.25), (-22.5, +22.5) \text{ and } (-40.0, +40.0).$$

Notice that in the above problems $S^*_{max} = S'_{max}$, i.e. the corresponding winner has visited all the nodes. The least-square equations are presented along with their graphs of regression lines in Figures 12–14. The plotting values are the best found between all those produced from the different algorithms used. None of the intercepts seems to lie inside the corresponding interval, and therefore modification of the scores is attempted in order to see if any improvement, in the sense described above, can be made.

There are several score schemes to reallocate the scores. Of course, the more appropriate are used, i.e.

$$S(i) = \beta_1 \cdot S_1(i) + \beta_2 \cdot S_2(i),$$

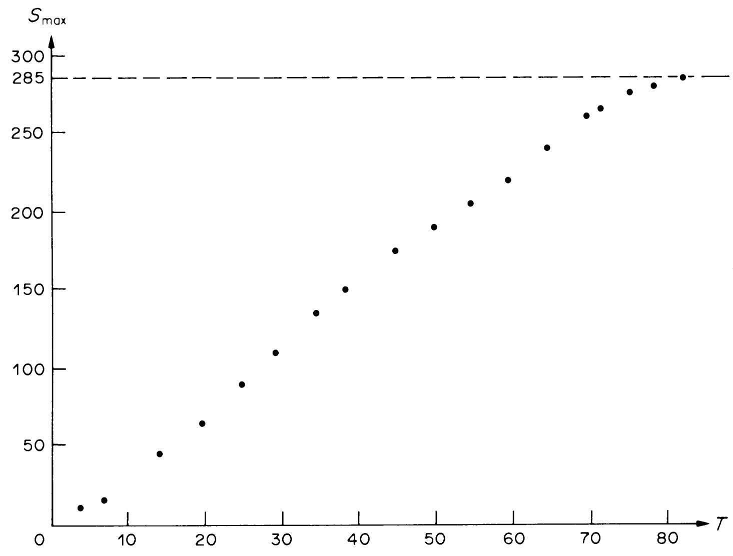
where

$$S_1(i) = h_1 \cdot \min\{C(i,j)\}; \quad i,j = 1,2,\ldots, \text{NPTS}, \quad i \neq j$$

and

$$S_2(i) = h_2 \cdot \sum_{j=1}^{\text{NPTS}} \{C(i,j)/(\text{NPTS} - 3)\}.$$

Notice that the new scores have been approximated to the nearest five integer. In order to simplify the calculations, the operator factors h_1 and h_2 were given the value 10.0 for all the cases. Four modifications are made for the 21-locations problem by letting the



PROBLEM 1

(32 — locations)

T_{max}	S_{max}	T
5	10	4.20
10	15	7.00
15	45	14.20
20	65	19.60
25	90	24.70
30	110	28.80
35	135	34.08
40	150	38.02
45	175	44.51
50	190	49.53
55	205	54.80
60	220	58.93
65	240	63.82
70	260	69.13
73	265	70.73
75	275	74.61
80	280	77.73
85	285	81.33

Total score: 285

$|a| \leqslant 0.05 \cdot 285 = 14.25$

$S = 34.13 + 3.17 \cdot T$

Fit: 99.5%

FIG. 12

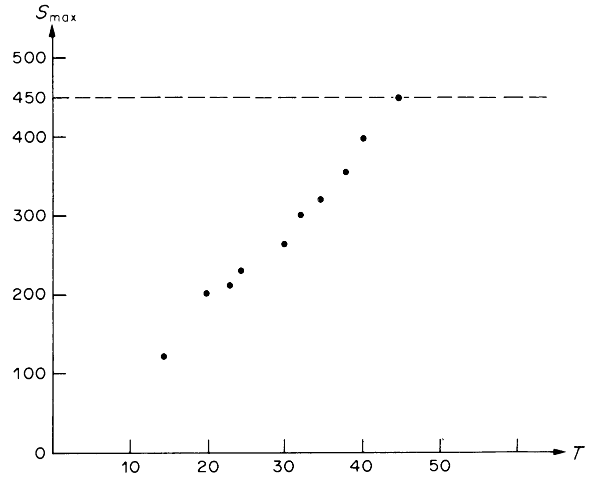
operator factors β_1 and β_2 take the following values:

$$(\beta_1, \beta_2);\ (0.0,\ 1.0),\ (1.0,\ 0.5),\ (0.5,\ 0.5),\ (0.5,\ 1.0).$$

The modifications of the 33-locations problem is also made by letting the same operator factors take the values (0.5, 1.0).

Concluding remarks

Unfortunately, none of the above modifications seems to satisfy the imposed requirement for the intercept. Therefore the above form of scheme seems not to have any effect on the relationship between S'_{max} and T'_{max}, as was hoped and expected. This suggests that other score schemes may be more appropriate. On the other hand, one may examine the effect of different values for the operator coefficients, although there is no guarantee that this will produce better results.



PROBLEM 2

(21 — locations)

T_{max}	S_{max}	T
15	120	14.25
20	200	19.88
23	210	22.65
25	230	24.13
27	230	24.13
30	265	29.85
32	300	31.63
35	320	34.51
38	355	37.62
40	395	39.78
45	450	44.44

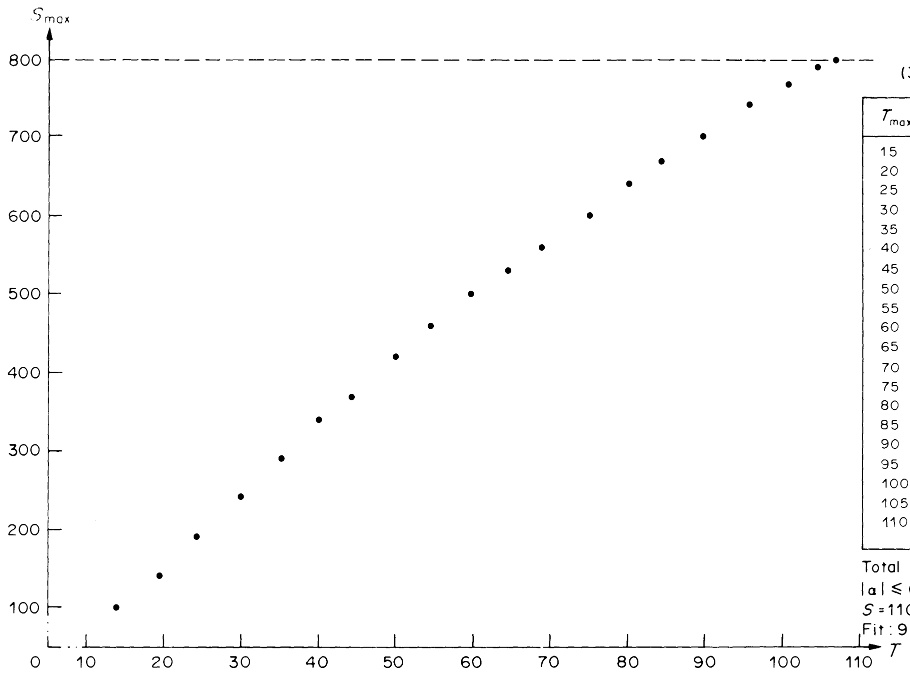
Total score : 450

$|a| \leqslant 0.05 \cdot 450 = 22.5$

$S = -34.32 + 10.62 \cdot T$

Fit: 99.15%

FIG. 13



PROBLEM 3

(33 — locations)

T_{max}	S_{max}	T
15	100	13.82
20	140	19.25
25	190	24.66
30	240	29.60
35	290	34.93
40	340	39.70
45	370	44.04
50	420	49.58
55	460	53.97
60	500	59.07
65	530	63.81
70	560	68.75
75	600	74.32
80	640	79.42
85	670	83.61
90	700	89.14
95	740	94.96
100	770	99.90
105	790	103.65
110	800	106.19

Total score : 800

$|a| \leqslant 0.05 \times 800 = 400$

$S = 110.51 + 6.59 \cdot T$

Fit : 99.90 %

FIG. 14

ACKNOWLEDGEMENTS

The main part of this work has been carried out in the Department of Probability and Statistics of Sheffield University. The author wishes to express his gratitude to Dr M. J. Maher of this Department for his valuable assistance during the preparation of this work. The author is also indebted to the referees for their valuable comments.

REFERENCES

A. WREN and A. HOLIDAY (1972) Computer scheduling of vehicles from one or more depots to a number of delivery points. *Opl Res. Q.* **23,** 333–344.

[2]G. CLARK and J. W. WRIGHT (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Opns Res.* **12,** 568–581.

R. L., KARG and G. L. THOMPSON (1964) A heuristic approach to solving travelling salesman problems. *Mgmt Sci.* **10,** 225–247.

R. A. HOLMES and R. G. PARKER (1976) A vehicle scheduling procedure based upon savings and a solution perturbation scheme. *Opl Res. Q.* **27,** 83–92.

[5]S. LIN (1965) Computer solution of the travelling salesman problem. *Bell Syst. Tech. J.* **44,** 2245–2269.

N. CHRISTOFIDES and S. EILON (1969) An algorithm for the vehicle dispatching problem. *Opl Res. Q.* **20,** 309–318.

B. E. GILLETT and L. R. MILLER (1974) A heuristic algorithm for the vehicle-dispatch problem. *Opl Res.* **22,** 340–349.

[8]G. A. GROES (1958) A method for solving travelling salesman problems. *Opns Res.* **6,** 791–812.

M. BELLMORE and G. L. NEMHAUSER (1968) The travelling salesman problem: a survey. *Opns Res.* **16,** 538–558.

[10]M. FLOOD (1956) A method of solving travelling salesman problems. *Opns Res.* **5,** 791.