ELSEVIER European Journal of Operational Research 88 (1996) 475–489

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

A fast and effective heuristic for the orienteering problem

I-Ming Chao [a], Bruce L. Golden [b,*], Edward A. Wasil [c]

[a] Department of Mathematics and Management Sciences, P.O. Box 90602, The Chinese Military Academy, Feng-Shen, Taiwan, ROC
[b] College of Business and Management, University of Maryland, College Park, MD 20742, USA
[c] Kogod College of Business Administration, American University, Washington, DC 20016, USA

Received 1 September 1993; revised 1 January 1995

Abstract

In the orienteering problem, start and end points are specified along with other locations which have associated scores. Given a fixed amount of time, the goal is to determine a path from the start point to the end point through a subset of locations in order to maximize the total path score. In this paper, a fast and extremely effective heuristic is presented and tested on 67 problems taken from the literature and 40 new test problems. The computational results are presented in detail.

Keywords: Vehicle routing problem; Heuristic search

1. Introduction

Orienteering is an outdoor sport usually played in a mountainous or heavily forested area. Armed with compass and map, competitors start at a specified control point, try to visit as many other control points as possible within a prescribed time limit, and return to a specified control point. Each control point has an associated score, so that the objective of orienteering is to maximize the total score. Competitors who arrive at the finish point after time has expired are disqualified, and the eligible competitor with the highest score is declared the winner. Since time is limited, competitors may not be able to visit all control points. The competitors have to select a subset of control points to visit that will maximize their total score subject to the time restriction.

The Orienteering Problem (OP) can be modeled as a multi-level optimization problem. At the first level, we need to choose a subset of control points to visit. At the second level, we need to solve a Traveling Salesman Problem (TSP) or a shortest Hamiltonian path problem over the selected subset of control points. We can think of this as a generalized TSP since competitors may start and finish at different locations. The two levels of this problem are closely related. If the path obtained from solving the second-level TSP is not feasible, then we need to remove some points from the selected subset of points. If the path is feasible, then, in order to maximize the total score, we may need to add points that were not selected at the first level.

We now describe a network optimization formulation of the OP. Let V be the set of control

* Corresponding author. E-mail: bgolden@umdacc.umd.edu

SSDI 0377-2217(95)00035-6

points (vertices) and E be the set of edges between points in V. Then $G = \{V, E\}$ is a complete graph. Each point i in V has a score $s_i \geq 0$ associated with it. The starting point is vertex 1 and the last point is vertex n, and each of these points has a score of zero. Each edge in E has a symmetric, nonnegative cost c_{ij} associated with it, where c_{ij} is the distance between point i and point j, or c_{ij} is the cost of traveling between the two points. For the OP, we need to find a path that starts at point 1 and finishes at point n such that the score amassed from the visited control points is maximized. Each control point can be visited at most once and the total time taken to visit control points cannot exceed the specified limit $T_{\max}$.

We point out that the OP is a variant of the well-known TSP. Suppose that a salesman wants to visit a set of cities in which the starting city and ending city are not the same. Furthermore, suppose that the time available for visiting the cities is limited. All of the cities cannot be visited in the allotted time, so the salesman must select a subset of cities to visit in order to maximize profit (or total score) without violating the time restriction. The TSP is a special case where $T_{\max}$ is very large and the start and end locations coincide.

The OP can be used to model many practical problems. Golden, Assad, and Dahl (1984) use the OP in the solution of a subset selection subproblem for a large inventory/routing problem. In particular, they model the delivery of home heating fuel as an OP. The urgency of a customer's request for fuel is treated as a score and the distribution manager must select a subset of customers to service each day. Keller (1989) treats the OP as a special case of the multi-objective vending problem in which the decision maker must trade off maximizing reward potential by visiting as many points as possible and minimizing travel cost by visiting as few points as possible. Balas (1989) presents another variant of the OP that is known as the Prize Collecting TSP. The objective is to minimize total travel cost and the net penalties for failing to visit some points, while visiting enough points to collect a prescribed amount of prize money. Mittenthal and Noon (1992) and Pillai (1992) examine a specific example of the Traveling Salesman Subset-Tour Problem in which an additional constraint (such as a maximum time constraint) has been included (this problem is denoted by TSSP + 1). Kantor and Rosenwein (1992) add time windows to the OP, so that a point can only be visited within a certain time interval. This version of the OP has potential applications to problems found in bank and postal delivery, industrial refuse collection, dial-a-ride services, and school bus routing. We point out that the OP has been shown to be NP-hard by Golden, Levy, and Vohra (1987).

In the next section, we describe the various solution approaches to the OP that have appeared in the OR literature. In the third section, we develop a new heuristic for solving the OP (without time windows). In the fourth section, we present computational results for 67 test problems taken from the literature and for 40 new problems that we generated. In the last section, we present some conclusions.

2. Review of solution approaches to the OP

Most of the research into solution methods for the OP has occurred over the last 10 years or so. Tsiligirides (1984) develops two heuristic approaches for solving the OP. The first approach is a stochastic method that uses a Monte Carlo technique for generating a large number of solutions. For each point i that is not yet included on the current path, Tsiligirides assigns a desirability measure denoted by A_i. The desirability measure is given by

$$A_i = (s_i / c_{i,\text{last}})^4, \tag{1}$$

where s_i is the score associated with point i and $c_{i,\text{last}}$ is the distance from the last point on the current path to point i. The four points with the largest desirability measures are identified and their measures normalized according to

$$P_i = \frac{A_i}{\sum_{t=1}^{4} A_t}, \quad \text{for } i = 1,2,3,4. \tag{2}$$

Point i is randomly selected with probability P_i as the new last point on the current path. Points continue to be placed on the current path until the remaining time is so small that no points can be feasibly inserted onto the path. Three thousand solutions are generated in this way and the solution with the highest score is selected as the final solution.

The second solution approach developed by Tsiligirides is based upon a vehicle routing procedure due to Wren and Holliday (1972). This approach divides the geographic area into sectors that are determined by two concentric circles and an arc of specified length. Routes are built up within each sector in an effort to minimize the total distance traveled. Sectors are changed by varying the two radii of the circles and by rotating the arc. For each problem, Tsiligirides examines 48 cases and selects the solution with the highest score as the final solution.

We point out that, with both heuristics, Tsiligirides improves the final solution by applying a three-step post processor. In the first step, a 2-opt heuristic is applied to reduce the length of the path found previously. Next, Tsiligirides tries to insert new points onto this path. Finally, he tries to increase the total score, while preserving feasibility, by removing a point on the current path and inserting a new point not on the path.

Golden, Levy, and Vohra (1987) develop a procedure for solving the OP that consists of three steps: path construction, path improvement, and center of gravity improvement. In the first step, a 'bang for buck' insertion heuristic is used to construct a path. Each point i is assigned a weighted measure given by

$$W_i = a \cdot s_i + b \cdot C_i + c \cdot E_i, \tag{3}$$

where $a + b + c = 1$, s_i is the score associated with point i, E_i is the summation of distances from point i to the start and finish points, and C_i is the distance from point i to the center of gravity. In the second step, Golden, Levy, and Vohra apply a 2-opt heuristic to improve the current solution. They try to exchange a point on the path with a point not on the path in the hope of decreasing the current path's length. In the third step, a new center of gravity is computed, and the three steps are repeated until a path that is identical to the previous one is produced. The path with the highest score is selected as the final solution.

Golden, Wang, and Liu (1988) incorporate the center of gravity idea and Tsiligirides's randomization concept, along with learning capabilities, into a new procedure for solving the OP. For each point i that is not on the current path, a weighted measure is computed by

$$W_i = \alpha \cdot S_i + \beta \cdot \tilde{C}_i + \gamma \cdot \tilde{E}_i, \tag{4}$$

where $\alpha + \beta + \gamma = 1$, S_i is a combined score that takes into account a point's actual score, the scores of its neighbors, distances to its neighbors, and a learning component that includes information regarding previous solutions, $\tilde{C}_i$ is a scaled measure of the distance from i to the center of gravity, and $\tilde{E}_i$ is a scaled measure of the distance from i to the start and end points. At each step of the procedure, the five points with the largest W_i values are identified and one of these points is selected at random and inserted in a least-cost way onto the current path. Feasibility with respect to the maximum time limit $T_{\max}$ is then checked. If the path is feasible, another point is considered for insertion. If the path is infeasible, then a point is removed from the path to make it feasible. The process of selecting and inserting points is carried out until no new point can be feasibly added to the current path. A new solution is generated by recomputing the center of gravity, updating the W_i's, and repeating the selection and insertion procedures. Initially, five centers of gravity are used and these are positioned at the centers of five squares. For each center, 20 solutions are generated, and the one with the highest score is selected as the final solution.

Keller (1989) modifies his algorithm for the multi-objective vending problem (Keller, 1985) to solve the OP. The algorithm contains a path-construction stage that is followed by an improvement stage. In the construction stage, two different approaches are used to select a point for insertion onto a path. The first approach is a deterministic approach that computes a desirability measure similar to (1) for each point i not on a path, and selects the point with the largest value for insertion onto the current path. The second approach is stochastic in nature. The scores of all points that are not on the current path and can be feasibly inserted are normalized. A random number between 0 and 1 is generated and the point corresponding to this number is inserted onto the current path. After a path has been constructed, an improvement stage that consists of two steps is applied to the solution. The first step inserts one point onto the path and considers removing none, one, or two points to increase the total score. The second step simultaneously removes a cluster of points and inserts a cluster of points to increase the total score.

Ramesh and Brown (1991) integrate four phases – point insertion, cost improvement, point deletion, and maximal insertions – into an algorithm that solves the OP. In the first phase, an insertion method that relaxes the maximum time limit is used to construct a path. The path is then improved by a 2-opt procedure followed by a 3-opt procedure. In the third phase, one point is removed and one point is then inserted in an attempt to decrease the length of the path. Finally, as many unassigned points as possible are inserted onto the current path. The last three phases are applied repeatedly in an attempt to find a high-quality solution.

We point out that all of the heuristic methods that we have described so far have been applied to 49 benchmark problems generated by Tsiligirides (1984). We will apply our new heuristic to these 49 problems and compare our results with the published solutions produced by the five different heuristics.

In the past three years, two exact solution methods for a variant of the OP, in which the start and finish points are the same, have appeared in the literature. Laporte and Martello (1990) use a branch and bound method to solve small, randomly generated test problems that contain as many as 20 points. Ramesh, Yoon, and Karwan (1992) use Lagrangian relaxation along with improvement procedures within a branch and bound method to solve large, randomly generated test problems that contain as many as 150 points for which the T_{max} values are also randomly selected.

Sokkappa (1990) develops two exact methods (one based upon a branch and bound method for the knapsack problem and one based upon a branch and bound method for the TSP) and a heuristic method for solving the OP. The heuristic method is based upon the method of Golden, Wang, and Liu (1988). For each point i not on the current path, a neighborhood score is computed as follows

$$ns_i = s_i + \sum_{j \neq i} s_j e^{-\mu c_{ij}}, \quad \text{for all unvisited } j. \tag{5}$$

The criterion

$$L_i \cdot ns_i / cost_i \tag{6}$$

is used to identify five candidate points for insertion onto the current path, where L_i is the learning factor developed by Golden, Wang, and Liu (1988), and $cost_i$ is the cheapest insertion cost for point i. Each of the five points is selected with the same probability. After one of the five candidates is inserted onto the path, a point that is currently on the path is removed if the time limit T_{max} is violated. The point that has been removed is no longer considered for insertion and this step terminates when no point can be inserted. A 2-opt procedure followed by additional insertions is then used to increase the score. Five centers of gravity are selected initially in the same as way in Golden, Wang, and Liu and 10 solutions are generated for each. The solution with the highest score is selected as the final solution. Sokkappa claims that his heuristic method slightly outperforms Golden, Wang, and Liu's method, although the heuristic was not applied to any of the 49 benchmark problems.

Leifer and Rosenwein (1994) use 0–1 integer programming and a cutting plane method to find a tight upper bound on the optimal objective function value to each of the 49 benchmark test problems. We will use these bounds to assess the effectiveness of our new heuristic.

Pillai (1992) develops an exact procedure to solve the TSSP + 1 problem. Pillai treats the OP as a special case of the TSSP + 1. Her exact procedure is based upon a branching and cutting plane method. A relaxed LP is solved first, the violated constraints are examined and then added to the relaxed LP, and the LP is solved again. The steps are repeated until no violated constraints are detected. If the solution to the relaxed LP is integer, the procedure stops; otherwise, a branching on variables with noninteger values is carried out. Pillai claims that the final solution is the optimal solution. Pillai applies her method to the 49 benchmark problems. We will compare the results produced by our new heuristic to those produced by Pillai's method in Section 4.

3. A new heuristic for solving the OP

In this section, we describe a new heuristic for the OP that is easy to understand and easy to implement, and that produces near-optimal solutions in a short amount of computation time. Our heuristic consists of two steps: initialization and improvement. We initialize the procedure by constructing an ellipse over the entire set of points by using the start and end points as the two foci of the ellipse and the time limit T_{max} as the length of the major axis. In generating a path, we consider only the points that are within the ellipse, since any path that contains a point outside the ellipse will violate the T_{max} constraint. We generate several paths over the points and, in the improvement step, we allow the total score to *decrease* in the hope of ultimately finding a path with a large total score that is nearly optimal.

3.1. Initialization

The initialization step uses a greedy method on the points within the ellipse in order to insert the point with the cheapest insertion cost onto the path while ignoring its score. We construct L solutions for each problem, where L is the minimum of $(10, N)$ and N is the number of points within the ellipse. To get the l-th solution, the first path is constructed by finding the point with the l-th largest distance from the start and end points, forming a path through these three points, and then inserting points in a greedy way (with respect to distance not score) onto this path. When this path is full (that is, inserting a point on the path violates the time limit constraint), we construct a path through the remaining points using the greedy method. We continue to construct paths in this way until all points within the ellipse are on a path. Among these paths, we choose the path with the largest total score as the solution path and its score is the solution score. Among all L solutions, we choose the one with the highest score as the initial solution. We denote the initial solution path as $path_{op}$ and the set of other paths as $paths_{nop}$.

3.2. Two-point exchange

Using the starting solution produced in the initialization step, we try to improve this solution by performing a two-point exchange. A point i is moved from a path in $paths_{nop}$ and inserted onto $path_{op}$ and a point j is moved from $path_{op}$ and inserted onto a path in $paths_{nop}$. The insertions are performed in the cheapest way, and $path_{op}$ and the paths in $paths_{nop}$ are always kept feasible. Point i is inserted between two points on $path_{op}$ so that the increase in distance is minimized. Point j is inserted onto a path in $paths_{nop}$ where the insertion cost is least and the path that results is feasible. If no feasible insertion is possible in $paths_{nop}$, a new path that contains point j must be generated. Such an exchange can cause $path_{op}$ to become a path in $paths_{nop}$ when a path in $paths_{nop}$ has a larger score.

Let $L(p)$ denote the length of path p. The feasibility of the path that results when point i is inserted onto path p and point j is removed from path p can be checked by examining the following expression

$$L(p) - (c_{j,fj} + c_{pj,j} - c_{pj,fj}) + \min_{\substack{k \text{ visited in } p,\\ k \neq 1,j}} \{c_{i,k} + c_{i,pk} - c_{k,pk}\}, \tag{7}$$

where pj is the point that precedes point j on path p, fj is the point that follows point j on path p, and pk is the point that precedes point k

Table 1
Two-point exchange algorithm for the OP

For j = the first to the last point in $path_{op}$	(A loop)
For i = the first to the last point in the first to the last path in $paths_{nop}$	(B loop)
If exchanging i and j is feasible and the total score increases, do the exchange and go to the A loop	
Else	
Set the best exchange = one with the highest score	
End B loop	
If the score of the best exchange $\geq$ *record* − *deviation*, make the best exchange	
End A loop	

on path p after point j has been removed from the path. In (7), the term after the first minus sign is the savings that results from removing point j and the term after the second plus sign is the cost incurred by inserting point i onto path p. If the distance that results from the calculation in (7) is less than or equal to T_{max}, then the insertion is feasible; otherwise, the insertion is infeasible.

For each point in $path_{op}$, candidate exchanges are considered one at a time. Whenever a candidate exchange leads to a higher total score, the exchange is performed immediately, and all other exchanges are ignored. Whenever there is no candidate exchange for a point that increases the total score, we consider exchanges that decrease the total score by a small amount. If the decrease in score is above a threshold value for a specific exchange, then we perform it; otherwise, the point remains in its current position on the path, and we consider exchanging a different point. The score of the best solution obtained during this process is the *record* and the amount of decrease below record that we allow during the process is called the *deviation*. The approach is referred to as record-to-record improvement and is due to Dueck (1990). Our two-point exchange algorithm is given in Table 1.

3.3. One-point movement

We now consider moving one point at a time between paths and we move the points in a greedy way instead of the cheapest-cost way. We first attempt to insert a point i between points in the first edge of path p, then into the second edge of p, and so on. We make the move whenever it is feasible and it increases the total score. If no movement increases the total score, then we consider making the feasible movement that decreases the total score by the least amount. To obtain a candidate movement for a point and

Table 2
Algorithm for one-point movement in the OP

For i = the first to the last point in the T_{max} ellipse (say point i is in path q)	(A loop)
For j = the first to the last point in the first to the last path (p) in both $path_{op}$ and $paths_{nop}$ sets ($p \neq q$)	(B loop)
If inserting i in front of j on path p is feasible and the total score increases, then make the movement and go to the A loop	
Else	
Set the best movement = one with the highest score	
End B loop	
If the score of the best movement $\geq$ *record* − *deviation*, then make the best movement	
End A loop	

then determine which move to make, we apply the steps in Table 2.

We point out that although only one point is moved at a time, this type of movement can still change $path_{op}$. One point can be moved from $path_{op}$ to a path in $paths_{nop}$ and vice versa. We

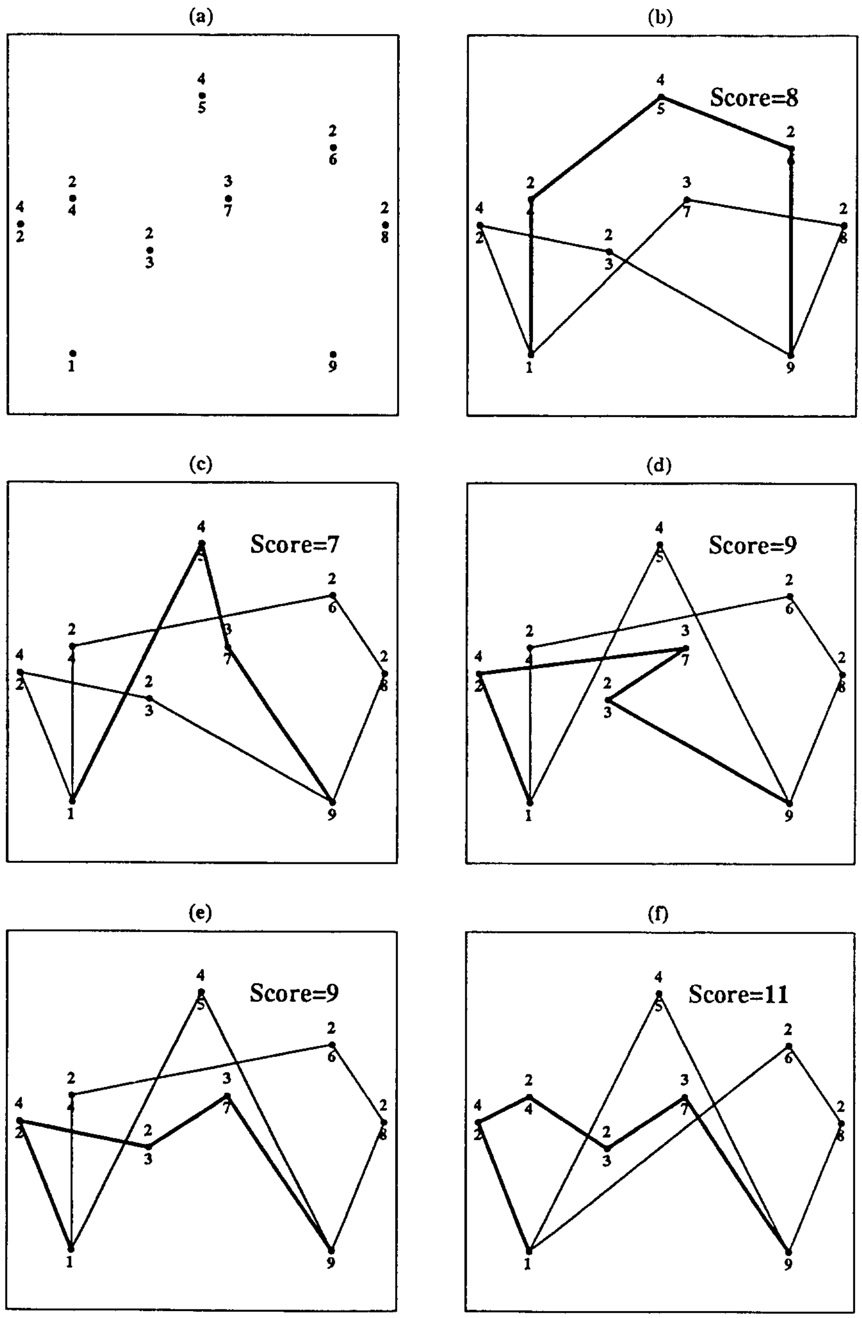

Fig. 1. Two-point exchange, one-point movement, and 2-opt improvement in the OP heuristic.

can also move a point between paths in $paths_{nop}$. We consider all of these movements in an effort to find the path with the largest total score.

3.4. Clean up

In order to shorten the length of $path_{op}$ we apply a 2-opt improvement procedure (Lin, 1965). The hope is that by decreasing the length of the path we have more opportunities to insert points from paths in $paths_{nop}$ onto $path_{op}$. We illustrate our two-point exchange, one-point movement, and 2-opt improvement in Fig. 1. In Fig. 1(a), we provide the scores and locations of the points, where the number above a point is the associated score. Fig. 1(b) shows the initial solution, where the thick, bold lines denote $path_{op}$ (that is, the path 1–4–5–6–9 with a total score of eight), and the thin lines denote the paths in $paths_{nop}$ (that is, the two paths 1–2–3–9 and 1–7–8–9). In Fig. 1(c), a two-point exchange is performed by inserting point 5 between points 1 and 7, and inserting point 8 between points 6 and 9 at the same time. This exchange results in a path with a total score of seven, which is lower than the initial solution and $path_{op}$ is now 1–5–7–9. In Fig. 1(d), we perform a one-point movement: point 7 moves from $path_{op}$ (that is, from 1–5–7–9) to a path in $paths_{nop}$ (that is, to 1–2–3–9). The new path (that is, 1–2–7–3–9) has a score of nine and becomes $path_{op}$. We now perform a 2-opt procedure on $path_{op}$ in Fig. 1(d). $path_{op}$ becomes 1–2–3–7–9 with a score of nine but its length has been reduced. To see this, compare the path 1–2–7–3–9 in Fig. 1(d) and the reduced-length path 1–2–3–7–9 in Fig. 1(e). We now move point 4 from a path in $paths_{nop}$ to $path_{op}$. This yields the solution shown in Fig. 1(f), which has the highest score of all solutions produced by our heuristic.

3.5. Reinitialization

In the hope of finding a path that yields a larger total score, we remove k points from $path_{op}$ that have the smallest ratio

$$s_i/cost_i, \tag{8}$$

where $cost_i$ is the current insertion cost of i, and insert them onto paths in $paths_{nop}$. As the iteration count increases in our procedure, we increase the value of k and remove more points from $path_{op}$. Our complete, new heuristic for the OP is shown in Table 3.

Table 3
A new heuristic for the TOP

```
Step 1. Initialization
    Perform initialization
    Set record = team score of the initial solution
    Set p = 10
    Set deviation = p% × record
Step 2. Improvement
    For k = 1,2,...,K                                              (K loop)
        For i = 1,2,...,I                                          (I loop)
            Perform two-point exchange
            Perform one-point movement
            Perform clean up
            If no movement has been made above, end I loop
            If a new better solution has been obtained, then
                set record = score of new best solution
                set deviation = p% × record
        End I loop
        Perform reinitialization
    End K loop
Step 3. Reset p = 5, and redo Step 2 once more
```

Table 4
Abbreviations and symbols for the OP

Abbreviations	
P_{no}	Test problem number
T_{max}	Maximum distance permitted
N_i	Number of points in problem set i
OH	New heuristic for the OP
CPU	Solution time in seconds on a SUN 4/370
UB	Upper bound on score from Leifer and Rosenwein (1991)
TS	Tsiligirides (1984) heuristic
GV	Golden, Levy, and Vohra (1987) heuristic
GL	Golden, Wang, and Liu (1988) heuristic
KL	Keller (1989) heuristic
RB	Ramesh and Brown (1991) heuristic
PL	Pillai (1992) exact method
TA	Our implementation of Tsiligirides's stochastic algorithm
Symbols	
+	OH produced a higher score
–	OH produced a lower score
– –	Solution of problem not attempted
?	No comparison listed since previous heuristic produced a score that exceeded the reported upper bound
Empty cell	OH produced the same score

Table 5
Dimensions for problems taken from the OP literature

Set 1 $N_1 = 31$		Set 2 $N_2 = 21$		Set 3 $N_3 = 33$		Set 4 $N_4 = 32$	
P_{no}	T_{max}	P_{no}	T_{max}	P_{no}	T_{max}	P_{no}	T_{max}
1.a	5	2.a	15	3.a	15	4.a	5
1.b	10	2.b	20	3.b	20	4.b	10
1.c	15	2.c	23	3.c	25	4.c	15
1.d	20	2.d	25	3.d	30	4.d	20
1.e	25	2.e	27	3.e	35	4.e	25
1.f	30	2.f	30	3.f	40	4.f	30
1.g	35	2.g	32	3.g	45	4.g	35
1.h	40	2.h	35	3.h	50	4.h	40
1.i	46	2.i	38	3.i	55	4.i	46
1.j	50	2.j	40	3.j	60	4.j	50
1.k	55	2.k	45	3.k	65	4.k	55
1.l	60			3.l	70	4.l	60
1.m	65			3.m	75	4.m	65
1.n	70			3.n	80	4.n	70
1.o	73			3.o	85	4.o	73
1.p	75			3.p	90	4.p	75
1.q	80			3.q	95	4.q	80
1.r	85			3.r	100	4.r	85
				3.s	105		
				3.t	110		

4. Computational testing

We test the performance of our new heuristic for the OP in two ways. We apply our heuristic to 67 problems taken from the literature and compare our solutions with the published results. We then generate 40 new test problems and apply our heuristic to these new problems. Our heuristic is coded in FORTRAN and executed on a SUN 4/370 workstation. We perform all computations using real precision (we do not round or truncate values), and round the length of the final path to one decimal place. In Table 4, we list abbreviations and symbols that we use throughout the remainder of this section.

4.1. Results on problems from the literature

Data for three sets of test problems have been published by Tsiligirides (1984). We present information about the test problems in Table 5. In examining Table 5, we see that in Set 1 there are 32 points. For each of the 18 problems in Set 1, the locations and scores of the points are the same, but the maximum distance value (T_{max}) is increased as we move from problem 1.a to problem 1.r. For each of the 11 problems in Set 2 and 20 problems in Set 3, the locations and scores of the points are the same, but the maximum distance value (T_{max}) is increased from 15 to as much as 110.

We point out that the data for Set 1 as published by Tsiligirides (1984) contains an error. In his paper, Tsiligirides presents two solutions to problem 1.r with $T_{max} = 85$. One solution has a total score of 285 and a distance of 81.82 and the other solution has a score of 285 and a distance of 81.33. When checking these solutions using the published locations and scores for problem 1.r, we obtain scores of 285 for both problems, but the distances are 88.23 and 87.74, which exceed the T_{max} value. The cause of this violation is the coordinate of point 31. The published coordinate is (4.90, 18.90) and this coordinate yields the two infeasible solutions just cited. Changing the coordinate to (4.90, 14.90) yields the feasible solutions published by Tsiligirides. We point out that all of the computational testing on Set 1 reported in

Table 6
Comparison of results on test problem Set 1

New heuristic			Previous methods							OH vs. previous					
P_{no}	OH	CPU	UB	TA	GV	GL	KL	RB	PL	TA	GV	GL	KL	RB	PL
1.a	10	0.67	10	10	10	10	10	10	10						
1.b	15	0.80	20	15	15	15	15	15	15						
1.c	45	2.28	45	45	45	45	45	45	45						
1.d	65	17.49	70	65	65	65	65	65	65						
1.e	90	9.01	95	90	90	90	90	90	90						
1.f	110	31.92	120	110	110	110	110	110	110						
1.g	135	25.25	140	135	125	135	130	135	135		+		+		
1.h	155	16.76	160	150	140	155	155	155	155	+	+				
1.i	175	21.58	180	170	165	175	175	175	175	+	+				
1.j	190	24.91	195	185	180	190	185	180	190	+	+		+	+	
1.k	205	24.67	210	195	200	205	200	205	205	+	+		+		
1.l	225	24.28	230	220	205	225	225	225	225	+	+				
1.m	240	23.26	245	235	220	240	240	240	240	+	+				
1.n	260	25.09	260	255	240	260	260	260	260	+	+				
1.o	265	25.24	270	260	255	265	265	265	265	+	+				
1.p	270	28.53	270	265	260	270	270	275	270	+	+			?	
1.q	280	26.84	285	270	275	280	280	280	280	+	+				
1.r	285	21.71	285	280	285	285	285	285	280	+					+
						Summary of OH vs. previous				11 +	11 +	0 +	3 +	1 +	1 +
										0 −	0 −	0 −	0 −	0 −	0 −

Table 7
Comparison of results on test problem Set 2

New heuristic			Previous methods							OH vs. previous					
P_{no}	OH	CPU	UB	TS	GV	GL	KL	RB	PL	TS	GV	GL	KL	RB	PL
2.a	120	1.29	145	120	120	120	120	120	120						
2.b	200	2.24	200	190	200	200	200	200	200	+					
2.c	210	4.45	215	205	210	205	210	210	210	+		+			
2.d	230	5.65	240	230	230	230	230	230	230						
2.e	230	6.37	265	230	230	230	230	230	230						
2.f	265	6.18	275	250	260	265	260	260	275	+	+		+	+	–
2.g	300	7.21	305	275	260	300	300	300	300	+	+				
2.h	320	7.81	350	315	300	320	320	320	320	+	+				
2.i	360	6.84	375	355	355	360	360	385	360	+	+			?	
2.j	395	7.14	400	395	380	395	380	395	395		+		+		
2.k	450	0.61	450	430	450	450	450	450	450	+					
						Summary of OH vs. previous				7 +	5 +	1 +	2 +	1 +	0 +
										0 –	0 –	0 –	0 –	0 –	1 –

the OP literature relies on Tsiligirides's published data with the (4.90, 18.90) coordinate. We will compare the performance of our new heuristic against previous results using the coordinates of the problem in Set 1 as published by Tsiligirides. Of course, we cannot include Tsiligirides's pub-

Table 8
Comparison of results on test problem Set 3

New heuristic			Previous methods							OH vs. previous					
P_{no}	OH	CPU	UB	TS	GV	GL	KL	RB	PL	TS	GV	GL	KL	RB	PL
3.a	170	4.37	175	100	170	170	170	170	170	+					
3.b	200	5.16	200	140	200	200	200	200	200	+					
3.c	260	9.40	260	190	250	260	260	260	260	+	+				
3.d	320	9.96	320	240	320	320	320	320	320	+					
3.e	390	15.38	390	290	380	390	370	390	390	+	+		+		
3.f	430	18.65	430	330	420	430	430	430	430	+	+				
3.g	470	26.84	470	370	450	470	460	470	470	+	+		+		
3.h	520	28.74	520	410	500	520	520	520	520	+	+				
3.i	550	30.27	550	450	520	550	550	550	550	+	+				
3.j	580	27.68	580	500	580	580	570	580	580	+			+		
3.k	610	25.02	610	530	600	610	610	610	610	+	+				
3.1	640	29.82	640	560	640	640	640	640	640	+					
3.m	670	29.25	670	590	650	670	670	670	670	+	+				
3.n	710	30.14	710	640	690	710	700	710	710	+	+		+		
3.o	740	28.30	740	670	720	740	740	740	740	+	+				
3.p	770	24.43	770	690	770	770	760	770	770	+			+		
3.q	790	22.33	790	720	790	790	790	790	790	+					
3.r	800	0.67	800	760	800	800	800	800	–	+					
3.5	800	0.60	800	770	800	800	800	800	–	+					
3.t	800	0.72	800	790	800	800	800	800	–	+					
						Summary of OH vs. previous				20 +	10 +	0 +	5 +	0 +	0 +
										0 –	0 –	0 –	0 –	0 +	0 –

lished results on this problem set in our comparisons, since his results are for the problems that use the (4.90, 14.90) coordinate. To overcome this deficiency, we coded a version of Tsiligirides's stochastic algorithm in FORTRAN (we denote our version by TA) and ran TA on all 18 problems in Set 1 using the (4.90, 18.90) coordinate. We also applied our new heuristic to a corrected data set (with the coordinate (4.90, 14.90)), denoted by Set 4 in Table 5, and will compare its results against those of Tsiligirides's stochastic algorithm (that is, against TA) and against his published results.

In order to test our heuristic, we must set the values of K and I in the improvement step of our heuristic (see Step 2 in Table 3). We set K equal to $\min(10, 0.75P)$, where P is the number of points on $path_{\mathrm{op}}$. After some experimenting, we set the value of I at 10. Using these parameter settings, we compare the results produced by our heuristic (OH) to results produced by six other methods (TA, GV, GL, KL, RB, and PL) on problem Sets 1, 2, and 3 in Tables 6–8. We note that the first five methods are heuristics, while the sixth method is an exact solution procedure.

In examining Tables 6 and 7, we point out that for two problems (that is, 1.p and 2.i) the solutions produced by Ramesh and Brown's heuristic exceeded the upper bounds on the scores given by Leifer and Rosenwein. We cannot check the feasibility of these solutions since their paths were not given by Ramesh and Brown. Because of this discrepancy, we do not compare OP against RB on these two problems.

For the 49 test problems listed in Tables 6–8, OH produces scores for 38 problems that are better than the scores produced by TA, 26 scores that are better than those produced by GV, 1 score that is better than that produced by GL, 10 scores that are better than those produced by KL, 2 scores that are better than those produced by RB, and 1 score that is better and 1 score that is worse than those produced by PL. We point out that on problem 1.r, OH produces a score of 285 and PL produces a score of 280. This should not happen since PL is supposed to be an exact method. Our solution is feasible, but we cannot verify the solution produced by PL since it has not been published. For the 49 test problems, OH

Table 9
Comparison of results on test problem Set 4

New heuristic			Previous methods		OH vs. previous	
P_{no}	OH	CPU	TS	TA	TS	TA
4.a	10	0.22	10	10		
4.b	15	0.27	15	15		
4.c	45	0.72	45	45		
4.d	65	4.76	65	65		
4.e	90	2.47	90	85		+
4.f	110	10.86	110	110		
4.g	135	14.11	135	135		
4.h	155	21.81	150	150	+	+
4.i	175	21.62	175	175		
4.j	190	22.76	190	185		+
4.k	205	24.81	205	200		+
4.1	220	20.39	220	220		
4.m	240	26.78	240	240		
4.n	260	25.51	255	250	+	+
4.o	265	27.04	260	265	+	
4.p	275	27.47	270	265	+	+
4.q	280	28.17	275	270	+	+
4.r	285	21.64	280	285	+	
			Summary of OH vs. previous		6 +	7 +
					0 −	0 −

produced only one score that was worse than the score produced by one of the other six methods. The complete set of results, including paths and locations of points, is given by Chao (1993).

The results in Tables 6–8 reveal that OH, indeed, produces high-quality results. One might argue, however, that other heuristics (e.g., GL) perform nearly as well. Furthermore, the GL computer code is, roughly speaking, comparable to OH in terms of running time. Nonetheless, the new heuristic stands out for several reasons. First, it is, we think, a bit smarter than the GL procedure. The GL procedure relies heavily upon brute force repetition from five initial centers of gravity, whereas the new heuristic is based primarily on the clever notion of record-to-record improvement. Second, this application is one of the first to successfully test the record-to-record improvement idea on a complex discrete optimization problem. Third, unlike the GL heuristic, the new heuristic generalizes in a natural way to solve related combinatorial problems (e.g., Chao, Golden, and Wasil, 1996).

In Table 9, we compare results produced by OH to results produced by TS and TA on problems in Set 4 (recall, this is the data set with the correct coordinate to point 31 and TA is our version of Tsiligirides's stochastic algorithm). In examining Table 9, we point out that the column of results under the heading TS are the values that Tsiligirides reported in his paper. The results produced by TS don't always coincide with those of TA since there is a stochastic component in both algorithms. For the 18 problems listed in Table 9, OH produces scores for six problems that are better than those produced by TS and 7 scores that are better than those produced by TA.

Table 10
Comparison of results on squared-shaped test problems

New heuristic			Previous method		OH vs. TA
P_{no}	OH	CPU	TA	CPU	
5.a	10	1.05	10	18.10	
5.b	40	0.46	40	34.20	
5.c	120	4.33	100	68.20	+
5.d	195	6.17	190	151.30	+
5.e	290	73.42	290	144.30	
5.f	400	54.82	400	188.90	
5.g	460	32.42	460	237.20	
5.h	575	98.92	575	288.50	
5.i	650	58.13	645	329.30	+
5.j	730	68.05	730	373.50	
5.k	825	65.23	820	414.90	
5.1	915	84.59	915	461.30	
5.m	980	82.18	980	495.20	
5.n	1070	119.00	1070	532.40	
5.o	1140	116.70	1140	566.70	
5.p	1215	108.93	1215	598.80	
5.q	1270	132.45	1265	629.10	+
5.r	1340	502.41	1340	655.50	
5.s	1380	467.13	1390	682.40	–
5.t	1435	128.56	1455	711.10	–
5.u	1510	316.30	1515	736.40	–
5.v	1550	469.94	1550	761.40	
5.w	1595	474.64	1590	783.50	+
5.x	1635	357.98	1635	807.90	
5.y	1655	268.86	1655	826.20	
5.z	1680	32.05	1670	847.30	+
		Summary of OH vs. TA			7+ 3–

Table 11
Comparison of results on diamond-shaped test problems

New heuristic			Previous method		OH vs. TA
P_{no}	OH	CPU	TA	CPU	
6.a	96	13.01	90	25.10	+
6.b	294	27.86	258	107.30	+
6.c	390	238.90	354	183.90	+
6.d	474	74.48	432	180.30	+
6.e	570	139.78	516	248.90	+
6.f	714	137.90	642	316.90	+
6.g	816	204.98	732	372.90	+
6.h	900	231.57	828	423.90	+
6.i	984	246.18	906	482.90	+
6.j	1044	264.77	978	527.90	+
6.k	1116	232.57	1020	568.50	+
6.1	1176	230.95	1110	608.40	+
6.m	1224	223.12	1152	645.30	+
6.n	1272	212.27	1200	678.90	+
			Summary of OH vs. TA		14+ 0–

4.2. Results on new test problems

In this section, we generate two new sets of test problems that contain a total of 40 problems (see Chao (1993) for new problem data), apply OH and TA to solve each problem, and then compare the results produced by the two heuristics.

4.2.1. Square-shaped test problems

We generate 26 test problems in which the locations of the points take on a square shape. Each test problem has 66 points. The start point and end point are near each other, and both are located at the center of four concentric squares. The distance between a point and its adjacent neighbor is either 2 or $2\sqrt{2}$. Points further from the start and finish have larger scores than points that are closer in. Points that comprise the innermost, second, third, and outermost concentric squares have scores of 5, 15, 25, and 35, respectively. We obtain 26 different problem instances by varying T_{max} from 5 to 130 in increments of 5.

4.2.2. Diamond-shaped test problems

We generate 14 problems in which the locations of the points take on a diamond shape. Each test problem has 64 points. The start point and end point are located far apart at the top and bottom of the diamond. The distance from a point to its adjacent neighbor is either 2 or $\sqrt{2}$. Points further from the start and finish have larger scores than points that are closer in. The scores are multiples of 6 and range from 6 to 42. We obtain 14 different problem instances by varying T_{max} from 15 to 80 in increments of 5.

We apply OH and TA to the square-shaped and diamond-shaped test problems and show the results produced by each heuristic in Tables 10 and 11. For the 26 square-shaped problems, OH produces scores for 7 problems that are better than the scores produced by TA, while TA produces scores for 3 problems that are better than the scores produced by OH. For the 14 diamond-shaped problems, OH produces scores on all 14 problems that are better than the scores produced by TA. We also point out that for both sets of problems the CPU time for OH is much shorter than the time required by TA.

5. Conclusions

In this paper, we have presented a new heuristic for the orienteering problem. Our heuristic is based on the notion of record-to-record improvement. We have applied our heuristic to 67 problems from the literature as well as to 40 new test problems. Our new heuristic has been shown to be computationally efficient and it performs well on all test problems.

References

Balas, E. (1989), "The prize collecting traveling salesman problem", *Networks* 19, 621–636.

Chao, I-M., (1993), "Algorithms and solutions to multi-level vehicle routing problems", Ph.D. Dissertation, Applied Mathematics Program, University of Maryland, College Park, MD.

Chao, I-M., Golden, B, and Wasil E. (1996), "The team orienteering problem", *European Journal of Operational Research* 88, 464–474.

Dueck, G. (1990), "New optimization heuristics, the great deluge algorithm and the record-to-record travel", Scientific Center Technical Report, IBM Germany, Heidelberg Scientific Center, Heidelberg, Germany, December.

Golden, B., Assad, A., and Dahl, R. (1984), "Analysis of a large-scale vehicle routing problem with an inventory component", *Large Scale Systems* 7, 181–190.

Golden, B., Levy, L., and Vohra, R. (1987), "The orienteering problem", *Naval Research Logistics* 34, 307–318.

Golden, B., Wang, Q., and Liu, L. (1988), "A multifaceted heuristic for the orienteering problem", *Naval Research Logistics* 35, 359–366.

Kantor, M., and Rosenwein, M. (1992), "The orienteering problem with time windows", *Journal of the Operational Research Society* 43/6, 629–635.

Keller, P.C. (1985), "Multiobjective routing through space and time: The MVP and TDVP problems", Ph.D. Dissertation, The University of Western Ontario, London, Ontario, Canada.

Keller, P.C. (1989), "Algorithms to solve the orienteering problem: A comparison", *European Journal of Operational Research* 41, 224–231.

Laporte, G., and Martello, S. (1990), "The selective traveling salesman problem", *Discrete Applied Mathematics* 26, 193–207.

Leifer, A.C., and Rosenwein, M.S. (1994), "Strong linear programming relaxations for the orienteering problem", *European Journal of Operational Research* 73, 517–523.

Lin, S. (1965), "Computer solutions of the traveling salesman problem", *Bell System Technical Journal* 44, 2245–2269.

Mittenthal, J., and Noon, C.E. (1992), "An insert/delete heuristic for the traveling salesman subset-tour problem with one additional constraint", *Journal of the Operational Research Society* 43/3, 277–283.

Pillai, R.S. (1992), "The traveling salesman subset-tour problem with one additional constraint (TSSP + 1)", Ph.D. Dissertation, The University of Tennessee, Knoxville, TN.

Ramesh, R., and Brown, K.M. (1991), "An efficient four-phase heuristic for the generalized orienteering problem", *Computers & Operations Research* 18/2, 151–165.

Ramesh, R., Yoon, Y., and Karwan, M. (1992), "An optimal algorithm for the orienteering tour problem", *ORSA Journal on Computing* 4/2, 155–165.

Sokkappa, P.R. (1990), "The cost-constrained traveling salesman problem", Ph.D. Dissertation, The University of California, Livermore, CA.

Tsiligirides, T. (1984), "Heuristic methods applied to orienteering", *Journal of the Operational Research Society* 35/9, 797–809.

Wren, A., and Holliday, A. (1972), "Computer scheduling of vehicles for one or more depots to a number of delivery points", *Operational Research Quarterly* 23, 333–344.