Computers & Operations Research 37 (2010) 1853 – 1859

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/cor

A Path Relinking approach for the Team Orienteering Problem

Wouter Souffriau[a,b,*], Pieter Vansteenwegen[b], Greet Vanden Berghe[a], Dirk Van Oudheusden[b]

[a]KaHo St.-Lieven, Information Technology, Gebr. Desmetstraat 1, 9000 Ghent, Belgium
[b]Katholieke Universiteit Leuven, Centre for Industrial Management, Celestijnenlaan 300A, 3001 Leuven (Heverlee), Belgium

ARTICLE INFO

Available online 15 May 2009

Keywords:
Team Orienteering Problem
Metaheuristic
Path Relinking

ABSTRACT

This paper introduces a Path Relinking metaheuristic approach for solving the Team Orienteering Problem (TOP), a particular routing problem in which a score is earned for visiting a location. The objective is to maximise the sum of the scores, while not exceeding a time budget T_{max} for travelling to the selected locations. In the case of the simple Orienteering Problem (OP), a single route connecting all selected locations should be followed; in TOP m routes are required and the length of each route is restricted to T_{max}. A fast and a slow variant of the approach are tested using a large set of test instances from the literature; they are compared to other state-of-the-art approaches. The fast variant achieves an average gap of 0.39% to the best known solutions in 5.0 s of calculation time, while the slow variant achieves a 0.04% gap within 272.8 s. Moreover, next to achieving most of the best known solutions for many testproblems, the slow variant improved the best known results in five instances.

1. Introduction

In the Orienteering Problem (OP) a set of n locations i is given, each with a score S_i. The starting point (location 1) and the end point (location n) are fixed. The time t_{ij} needed to travel from location i to j is known for all location pairs. A given T_{max} limits the time to visit locations. The goal of the OP is to determine a single route, limited to T_{max}, that maximises the total collected score. Each location can be visited at most once. The Team Orienteering Problem (TOP) is an OP that maximises the total collected score of m routes, each limited to T_{max}.

The OP [35] is also known as the selective travelling salesperson problem [22], the maximum collection problem [4] and the bank robber problem [2]. It can be formulated as a special case of the resource constrained travelling salesperson problem [26], as a travelling salesperson problem with profits [8] or as a resource constrained elementary shortest path problem [31]. Many (T)OP applications are described in the literature: the sport game of orienteering [7], the home fuel delivery problem [15], athlete recruiting from high schools [4], routing technicians to service customers [33], etc. The TOP cases addressed in this paper are aimed at developing a personalised electronic tourist guide. This is a device that at short notice should suggest a (near) optimal combination of tourist attractions and a route connecting these, taking into account the weather, opening hours, crowded places and personal preferences. The selection problem is called the Tourist Trip Design Problem [36]. The OP can be seen as the simplest form of this problem and has been successfully used as a model for generating personal tourist trips [32].

The remaining of this paper is structured as follows: Section 2 provides an overview of the literature on the (T)OP. Section 3 describes the Path Relinking approach to tackle the TOP. Section 4 presents the results of the proposed approach on different test sets available in the literature. Finally, Section 5 concludes the paper and points out directions for further research.

2. Literature review

Several researchers propose solving the OP with exact algorithms based on a branch-and-bound [23,29] and branch-and-cut approach [10,12]. With the branch-and-cut procedure, instances of up to 500 locations can be optimally solved [10], provided sufficient calculation time is available. Since the OP is NP-hard [15], obviously these exact algorithms are extremely time consuming so most research has focused on heuristic approaches. Tsiligirides [35] proposed a stochastic (S-Algorithm) and a deterministic algorithm (D-Algorithm). Golden et al. [15,16] developed a centre-of-gravity heuristic and later added both learning capabilities and the randomisation idea based on the

* Corresponding author at: KaHo St.-Lieven, Information Technology, Gebr. Desmetstraat 1, 9000 Ghent, Belgium.
E-mail addresses: wouter.souffriau@kahosl.be (W. Souffriau), pieter.vansteenwegen@cib.kuleuven.be (P. Vansteenwegen), greet.vandenberghe@kahosl.be (G. Vanden Berghe), dirk.vanoudheusden@cib.kuleuven.be (D. Van Oudheusden).

0305-0548/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2009.05.002

S-Algorithm. Chao et al. [6] clearly outperform the above-mentioned heuristics with their five step heuristic, which implements an efficient initialisation step, three improvement steps and one diversication step. Gendreau et al. [13] proposed a tabu search heuristic that iteratively inserts clusters of locations in the tour or removes a chain of locations. Compared to the previous methods, this heuristic reduces both the probability of getting trapped in a local optimum and the probability of including locations with a high score that are far from the current tour. More algorithms [11,20,24,28,34,39] were developed to tackle the OP, but none of them significantly improved the state-of-the-art algorithms.

Butt et al. [5] present an exact algorithm based on column generation to solve the TOP. They deal with problems up to 100 locations, provided that the number of selected locations in each tour remains small. Boussier et al. [3] propose a branch-and-price approach to solve problems with up to 100 locations. Only problems in which the number of possible locations in a route is low, namely, up to about 15 per route, can be solved in less than 2 h of calculation.

The first published TOP heuristic was developed by Chao et al. [6] and resembles their five-step heuristic for the OP. Tang and Miller-Hooks [33] developed a tabu search heuristic embedded in an adaptive memory procedure. Archetti et al. [1] came up with two variants of a tabu search heuristic and a slow and fast Variable Neighbourhood Search (VNS) algorithm. One tabu search algorithm only considers feasible solutions while the other also accepts infeasible ones. The VNS applies tabu search with much fewer iterations and only considers feasible solutions. Archetti et al. [1] present results that outperform the above-mentioned algorithms for the TOP. The strength of the algorithm probably lies in the fact that all non-included locations are also grouped in feasible tours, while in other algorithms only included locations are organised in tours. Ke et al. [19] developed four variants of an ant colony optimisation approach for the TOP. Their sequential variant can obtain the best solution quality within less than a minute of calculation time. Vansteenwegen et al. [37,38] implemented a Guided Local Search (GLS) and a skewed Variable Neighbourhood Search (SVNS) algorithm. The results of these algorithms are comparable with the results of Archetti et al. [1] and Ke et al. [19]. The quality is slightly lower but the computational effort is significantly reduced.

3. Path Relinking approach

The proposed solution approach is based on a Greedy Randomised Adaptive Search Procedure (GRASP), a metaheuristic first introduced by Feo and Resende [9]. In general, GRASP performs a number of iterations each consisting of a constructive procedure followed by a local search approach. The different iterations are independent and the best solution found is saved and returned as the result. Laguna and Martí [21] successfully combined this approach with Path Relinking, a memory component that visits solutions on the path between two known solutions. Path Relinking itself was originally introduced by Glover [14]. To the best of the authors' knowledge, a pure Path Relinking approach for a vehicle routing problem has never been published in a journal before. Ho and Gendreau [18] were the first to publish a Path Relinking approach for a vehicle routing problem, but complemented it with a tabu search procedure. The fact that Path Relinking has proven to work well for solving knapsack problems [17] and, in combination with GRASP, vehicle routing problems [27,30], motivates the use of this technique to tackle the TOP, which can be seen as a combination of the vehicle routing problem and the knapsack problem. Algorithm Listing 1 overviews the path relinking approach.

Algorithm 1. Path Relinking for the TOP.

while *Nr of iterations without improvement is not exceeded* **do**
  construct;
  local search;
  link to elites;
  update elite pool;
end
Return best found;

Until no further improvements are identified during a fixed number of iterations, four procedures are executed in sequence. First, the Construct procedure generates an initial solution, which is then improved by the Local Search procedure. Construct is based on the GRASP template and provides a new way to construct initial solutions for the TOP. Next, the actual Path Relinking is carried out by the Link To Elites procedure, which is a new and innovative TOP solution recombination operator. Finally, the pool of elite solutions is updated by the Update Elite Pool procedure, which is a solution population management procedure that effectively executes the recombination operator. The algorithm keeps track of the best solution found and returns it as the final result. The following subsections describe the four procedures in detail.

3.1. Construct

The behaviour of the Construct procedure is determined by a single parameter, called greediness. It lies between 0 and 1, and prescribes a precise ratio between greediness and randomness. Every time Construct is executed, a new greediness value is drawn from a uniform distribution. The procedure starts from an empty solution of m identical routes, which only contain the start and end location. In order to construct an initial solution, a candidate list (CL) of feasible insert moves is generated. An insert move IM_{ilj} considers the insertion of an unvisited location l into the current solution between locations i and j. IM_{ilj} is characterised by an additional score S_l, and an increase of the duration, equal to $t_{ilj} = t_{il} + t_{lj} - t_{ij}$. Only feasible insert moves are considered, which means that the time increase of the tour has to be less than or equal to the remaining time budget. All feasible moves are collected into the CL and for each of them, a heuristic value $h_{ilj} = S_l/t_{ilj}$ is calculated. Infeasible insert moves, which lead to a violation of the time budget, are discarded. A threshold is computed by multiplying the difference between the maximum and minimum heuristic values of the candidate list by the greediness parameter. The original CL is filtered and only members with a heuristic value that exceeds the threshold are collected into the restricted CL. A random move from this restricted CL is picked out and applied to the current solution. The Construct procedure ends when no more feasible insertion candidates are found.

3.2. Local Search

The Local Search procedure, illustrated by Algorithm Listing 2, iteratively searches four different neighbourhoods in a best improving way. The local search procedure alternates between reducing the total time of the solution and increasing its total score. The procedure stops when the solution is locally optimal with respect to all the neighbourhoods.

Algorithm 2. Local Search.

while *Improvement* **do**
  2-Opt;
  Swap;
  Replace;
  Insert;
end
Return local optimum;

First, in order to reduce the total time, a standard 2-Opt neighbourhood [25] is explored.

Then, the Swap neighbourhood attempts to exchange locations between tours. For each pair of locations that belong to different tours of the current solution, a Swap move is considered if a reduction in total solution time can be achieved. This neighbourhood is explored in a best improving way: the Swap move that results in the best time reduction is executed, until no improvement in solution time can be found.

Next, Replace tries to exchange locations that belong to the current solution with non-included locations. For every location, its cheapest insertion position is calculated. If enough time budget is available for insertion, a regular Insert move is considered, as described in the Construct procedure. If the remaining budget is insufficient, all included locations with a lower score are considered for exclusion. If excluding one of these locations from the route creates enough time to insert the non-included location under consideration, a replacement will be considered. The Replace neighbourhood is explored in a best improving way, the move with the highest score increase and, in case of a tie, the move with the highest score and lowest time increase is executed.

Finally, the Insert neighbourhood explores the possibility to increase the score of the current solution, as described in the Construct procedure. The different neighbourhoods are explored consecutively until no further improvement is found.

3.3. Link to Elites

The goal of the Path Relinking extension is to avoid the independence of the different iterations of the original GRASP approach by adding a long-term memory component, a pool of elite solutions. Algorithm Listing 3 describes the Link to Elites procedure in detail. This procedures takes two solutions as arguments, the starting solution and the guiding solution, and visits the solutions on the virtual path in the search space that connects both solutions. The best solution found on the path is returned.

Algorithm 3. Link to Elites.

```
Set intersection = common locations of both solutions;
Set locationsToAdd = guiding solution minus intersection;
while locationsToAdd is not empty do
    Insert locations, allow infeasibility;
    Delete locations to restore feasibility;
    Local Search;
end
```

All locations appearing in both the guiding solution and the starting solution, are omitted from the former, resulting in the "set of locations to add". Next, all the non-included locations are considered for insertion in the starting solution. In contrast to the insert moves described above, a location of the set of features to add is also considered for insertion when it violates the time budget, if the route was not infeasible before insertion. When all m routes are infeasible, no more locations are inserted and feasibility is restored for each route by removing locations. For every location l that belongs to an infeasible route, located between i and j, $1/h_{ilj} = t_{ilj}/S_l$ is calculated. The location l with the highest $1/h_{ilj}$ is removed from the route. This removal is repeated until each route is feasible again. Next, Local Search tries to improve the solution. The Link to Elites procedure ends when the set of locations to add is empty.

The Link to Elites procedure is performed at each iteration for all the pairs of combinations of (1) the solution constructed and improved in the current iteration and (2) the members of the elite pool as starting and guiding solution, and vice versa. The similarity measure between starting and guiding solution is calculated. The formula $2n_{X\cap Y}/(n_X + n_Y)$ denotes the similarity between solutions X and Y, where n_X and n_Y are the number of locations of solutions X and Y, and $n_{X\cap Y}$ the number of locations that the two solutions have in common. If both solutions are too similar, i.e. the similarity measure is below a threshold, the path between them is not explored.

3.4. Update elite pool

The best solution found by Link to Elites is considered for insertion into the pool of elite solutions. If the elite pool has not reached its maximum number of members, then the solution is added to the pool. If the elite pool is full and the solution under consideration is better than the worst elite solution, the worst elite solution is replaced.

Each member of the elite pool is given an age, which is initialised to 0 and incremented by 1 each time it is used by the Link to Elites procedure. When a member's age achieves the threshold of max (10; Nr of iterations without improvement/10), it is removed from the pool.

4. Experimental results

This section presents the experimental results of the path relinking approach compared to the following algorithms from the literature:

- TMH, tabu search by Tang and Miller-Hooks [33];
- GTP, tabu search with penalty strategy by Archetti et al. [1];
- GTF, tabu search with feasible strategy by Archetti et al. [1];
- FVF, fast Variable Neighbourhood Search by Archetti et al. [1];
- SVF, slow Variable Neighbourhood Search by Archetti et al. [1];
- GLS, Guided Local Search by Vansteenwegen et al. [38];
- ASe, sequential ant colony optimisation by Ke et al. [19];
- ADC, deterministic concurrent ant colony optimisation by Ke et al. [19];
- ARC, random concurrent ant colony optimisation by Ke et al. [19];
- ASi, simultaneous ant colony optimisation by Ke et al. [19];
- SVNS, Skewed Variable Neighbourhood Search by Vansteenwegen et al. [37].

The results of Chao et al. [7] are excluded from this comparison as they are obtained using a different rounding precision and are outperformed by the above-mentioned algorithms.

Data sets 4–7 from Chao et al. [7] are used to benchmark the different approaches. These sets contain 100, 66, 64 and 102 (n) vertices, respectively, start and end vertex included. Each set contains instances with m equal to 2, 3 and 4 routes. Instances from the same set with the same number of routes differ in the value T_{max}. Instances for which the same result is obtained by all above-mentioned algorithms are excluded from the comparison, resulting in 157 relevant instances [1].

The Path Relinking algorithm is coded using JAVA 1.6. All experiments are performed on an Intel Xeon 2.5 GHz. Preliminary tests identify a pool size of five and a similarity threshold of 0.9 as well performing parameters. As the stopping criterion 10 and 300 iterations without improvement are used, which is called Fast Path Relinking (FPR) and Slow Path Relinking (SPR), respectively. Ten runs of the algorithm are performed for each instance. Tables 1–4 compare for each test set the best scores of both approaches out of 10 independent runs to the best scores found by the approaches from the literature. The results are summarised in Table 5. For each approach, the number of times the best known solution is found and the average gap to the best known solution are given, as well as

Table 1
Results test set 4.

Instance	Best	TMH	GTP	GTF	FVF	SVF	GLS	ASe	ADC	ARC	ASi	SVNS	FPR	SPR
p4.2.a	**206**	202	**206**	**206**	**206**	**206**	**206**	**206**	**206**	**206**	**206**	202	**206**	**206**
p4.2.b	**341**	**341**	**341**	**341**	**341**	**341**	303	**341**	**341**	**341**	**341**	**341**	**341**	**341**
p4.2.c	**452**	438	**452**	**452**	**452**	**452**	447	**452**	**452**	**452**	**452**	**452**	**452**	**452**
p4.2.d	**531**	517	530	**531**	**531**	**531**	526	**531**	**531**	530	**531**	528	**531**	**531**
p4.2.e	**618**	593	**618**	613	**618**	**618**	602	**618**	600	600	613	593	612	**618**
p4.2.f	**687**	666	**687**	676	684	**687**	651	**687**	672	672	672	675	**687**	**687**
p4.2.g	**757**	749	751	756	750	753	734	**757**	756	756	756	750	**757**	**757**
p4.2.h	**835**	827	795	820	827	**835**	797	827	819	819	820	819	**835**	**835**
p4.2.i	**918**	915	882	899	916	**918**	826	**918**	900	**918**	**918**	916	**918**	**918**
p4.2.j	**965**	914	946	962	962	962	939	**965**	962	962	962	962	962	**965**
p4.2.k	**1022**	963	1013	1013	1019	**1022**	994	**1022**	1016	1016	1016	1007	1013	**1022**
p4.2.l	**1074**	1022	1061	1058	1073	**1074**	1051	1071	1070	1071	1069	1051	1064	**1074**
p4.2.m	**1132**	1089	1106	1098	**1132**	**1132**	1051	1130	1115	1119	1113	1051	1130	**1132**
p4.2.n	**1173**	1150	1169	1171	1159	1171	1117	1168	1149	1158	1169	1124	1161	**1173**
p4.2.o	**1218**	1175	1180	1192	1216	**1218**	1191	1215	1209	1198	1210	1195	1206	**1218**
p4.2.p	**1242**	1208	1226	1239	1239	1241	1214	**1242**	1229	1233	1239	1237	1240	**1242**
p4.2.q	**1265**	1255	1252	1255	**1265**	1263	1248	1263	1253	1252	1260	1239	1257	1263
p4.2.r	**1288**	1277	1281	1283	1283	1286	1267	**1288**	1278	1278	1279	1279	1278	1286
p4.2.s	**1304**	1294	1296	1299	1300	1301	1286	**1304**	**1304**	1303	**1304**	1295	1293	1296
p4.2.t	**1306**	**1306**	**1306**	**1306**	**1306**	**1306**	1294	**1306**	**1306**	**1306**	**1306**	1305	1299	**1306**
p4.3.c	**193**	192	**193**	**193**	**193**	**193**	**193**	**193**	**193**	**193**	**193**	**193**	**193**	**193**
p4.3.d	**335**	333	**335**	**335**	**335**	**335**	**335**	**335**	333	333	**335**	331	333	**335**
p4.3.e	**468**	465	**468**	**468**	**468**	**468**	444	**468**	**468**	**468**	**468**	460	**468**	**468**
p4.3.f	**579**	**579**	**579**	**579**	**579**	**579**	564	**579**	**579**	**579**	**579**	556	**579**	**579**
p4.3.g	**653**	646	651	652	**653**	**653**	644	**653**	652	**653**	652	651	**653**	**653**
p4.3.h	**729**	709	722	727	724	**729**	706	720	713	713	713	718	725	**729**
p4.3.i	**809**	785	806	806	806	807	806	796	793	793	786	807	797	**809**
p4.3.j	**861**	860	858	858	**861**	**861**	826	**861**	857	855	858	854	858	**861**
p4.3.k	**919**	906	**919**	918	**919**	**919**	864	918	913	910	910	902	918	918
p4.3.l	**979**	951	976	973	975	978	960	**979**	958	976	966	969	968	**979**
p4.3.m	**1063**	1005	1034	1049	1056	**1063**	1030	1053	1039	1028	1046	1047	1043	**1063**
p4.3.n	**1121**	1119	1108	1115	1111	**1121**	1113	**1121**	1109	1112	1103	1106	1108	1120
p4.3.o	**1172**	1151	1156	1157	**1172**	1170	1121	1170	1163	1167	1165	1136	1165	1170
p4.3.p	**1222**	1218	1207	1221	1208	**1222**	1190	1221	1202	1207	1207	1200	1209	1220
p4.3.q	**1253**	1249	1237	1241	1250	1251	1210	1252	1239	1239	1238	1236	1246	**1253**
p4.3.r	**1272**	1265	1224	1269	**1272**	**1272**	1239	1267	1263	1263	1263	1250	1257	**1272**
p4.3.s	**1294**	1282	1250	**1294**	1289	1293	1279	1293	1291	1289	1291	1280	1276	1287
p4.3.t	**1305**	1288	1303	1304	1298	1304	1290	**1305**	1304	1303	1304	1299	1294	1299
p4.4.e	**183**	182	**183**	**183**	**183**	**183**	**183**	**183**	**183**	**183**	**183**	**183**	**183**	**183**
p4.4.f	**324**	315	**324**	**324**	**324**	**324**	312	**324**	**324**	**324**	**324**	319	**324**	**324**
p4.4.g	**461**	453	**461**	**461**	**461**	**461**	**461**	**461**	**461**	**461**	460	**461**	**461**	**461**
p4.4.h	**571**	554	**571**	**571**	**571**	**571**	565	**571**	556	556	556	553	**571**	**571**
p4.4.i	**657**	627	655	**657**	**657**	**657**	**657**	**657**	653	652	653	**657**	653	**657**
p4.4.j	**732**	**732**	731	731	**732**	**732**	691	**732**	731	711	731	723	**732**	**732**
p4.4.k	**821**	819	**821**	816	**821**	**821**	815	**821**	820	818	818	**821**	820	**821**
p4.4.l	**880**	875	878	878	879	**880**	852	**880**	877	875	875	876	875	879
p4.4.m	**919**	910	916	918	916	**919**	910	918	911	906	911	903	914	**919**
p4.4.n	**977**	**977**	972	976	968	968	942	961	956	956	956	948	953	969
p4.4.o	**1061**	1014	1057	1057	1051	**1061**	937	1036	1030	1021	1029	1030	1033	1057
p4.4.p	**1122**	1056	1120	1120	1120	1120	1091	1111	1108	1088	1110	1120	1098	**1122**
p4.4.q	**1161**	1124	1148	1157	1160	**1161**	1106	1145	1150	1137	1148	1149	1139	1160
p4.4.r	**1213**	1165	1203	1211	1207	1203	1148	1200	1195	1195	1194	1193	1196	**1213**
p4.4.s	**1259**	1243	1245	1256	**1259**	1255	1242	1249	1256	1249	1252	1213	1231	1250
p4.4.t	**1285**	1255	1279	**1285**	1282	1279	1250	1281	1281	1283	1281	1281	1256	1280

the average CPU times in seconds. Execution times per test set are provided in Table 6. For the ant colony approaches of Ke et al. [19], the reported average execution times are multiplied by 10, which equals the number of runs it took to find their best solution. In order to make a fair comparison, the execution times reported for the FPR and SPR approaches are also the sum of the execution times of the 10 independent runs. The FPR approach is able to find the best known solution 78 times and achieves an average gap of 0.39%. Table 7 lists the number of instances for which FPR achieved a better, equal and worse solution compared to the approaches from the literature. Based on these criteria, the quality of the results of GTF, FVF and SVF of Archetti et al. [1] and ASe, ARC and ASi of Ke et al. [19] are better. However, the average execution time of FPR, 5.0 s, is much faster. The SPR approach managed to find the best known solution 131 times and achieves a 0.04% gap. Tables 5 and 8 show that SPR achieves comparable results to SVF and is slightly better than ASe, using less computation effort. Moreover, next to achieving most of the best known solutions for many testproblems, SPR managed to find a new best solution for instances p4.2.n, p4.3.i, p4.3.q, p4.4.p and p4.4.r.

The strength of the new approach of this paper lies in the usage of the Link to Elites procedure. In order to assess the contribution of the path relinking extension to the basic GRASP framework, a test is performed of 10 iterations without the path relinking. This basic GRASP approach achieves an average gap of 3.05% in 0.7 s. The PR extension can be seen as an evolutionary population based method

Table 2
Results test set 5.

Instance	Best	TMH	GTP	GTF	FVF	SVF	GLS	ASe	ADC	ARC	ASi	SVNS	FPR	SPR
p5.2.h	**410**	**410**	**410**	**410**	**410**	**410**	385	**410**	**410**	**410**	**410**	395	**410**	**410**
p5.2.j	**580**	560	**580**	**580**	**580**	**580**	**580**	**580**	**580**	**580**	**580**	**580**	**580**	**580**
p5.2.k	**670**	**670**	**670**	**670**	**670**	**670**	665	**670**	**670**	**670**	**670**	**670**	**670**	**670**
p5.2.l	**800**	770	**800**	**800**	**800**	**800**	760	**800**	**800**	**800**	**800**	770	**800**	**800**
p5.2.m	**860**	**860**	**860**	**860**	**860**	**860**	830	**860**	**860**	**860**	**860**	**860**	**860**	**860**
p5.2.n	**925**	920	**925**	**925**	**925**	**925**	920	**925**	920	920	**925**	920	**925**	**925**
p5.2.o	**1020**	975	**1020**	**1020**	**1020**	**1020**	1010	**1020**	**1020**	1010	1010	**1020**	**1020**	**1020**
p5.2.p	**1150**	1090	1130	**1150**	**1150**	**1150**	1030	**1150**	**1150**	**1150**	**1150**	**1150**	**1150**	**1150**
p5.2.q	**1195**	1185	**1195**	**1195**	**1195**	**1195**	1145	**1195**	**1195**	**1195**	**1195**	**1195**	**1195**	**1195**
p5.2.r	**1260**	**1260**	**1260**	**1260**	**1260**	**1260**	1225	**1260**	**1260**	**1260**	**1260**	**1260**	**1260**	**1260**
p5.2.s	**1340**	1310	1330	**1340**	**1340**	**1340**	1325	**1340**	1330	1330	1330	1325	**1340**	**1340**
p5.2.t	**1400**	1380	1380	**1400**	**1400**	**1400**	1360	**1400**	**1400**	**1400**	**1400**	1380	**1400**	**1400**
p5.2.u	**1460**	1445	1440	**1460**	**1460**	**1460**	**1460**	**1460**	**1460**	**1460**	**1460**	1450	**1460**	**1460**
p5.2.v	**1505**	1500	1490	**1505**	1500	**1505**	1500	**1505**	1495	1500	1495	1500	**1505**	**1505**
p5.2.w	**1565**	1560	1555	**1565**	1560	1560	1560	1560	1555	1555	1555	1560	1560	1560
p5.2.x	**1610**	**1610**	1595	**1610**	1590	**1610**	**1610**	**1610**	**1610**	**1610**	**1610**	1600	**1610**	**1610**
p5.2.y	**1645**	1630	1635	1635	1635	1635	1630	**1645**	**1645**	**1645**	**1645**	1630	**1645**	**1645**
p5.2.z	**1680**	1665	1670	**1680**	1670	1670	**1680**	**1680**	**1680**	**1680**	**1680**	1665	1670	**1680**
p5.3.k	**495**	**495**	**495**	**495**	**495**	**495**	470	**495**	**495**	**495**	**495**	**495**	**495**	**495**
p5.3.l	**595**	575	**595**	**595**	**595**	**595**	545	**595**	**595**	**595**	**595**	**595**	**595**	**595**
p5.3.n	**755**	**755**	**755**	**755**	**755**	**755**	720	**755**	**755**	**755**	**755**	**755**	**755**	**755**
p5.3.o	**870**	835	**870**	**870**	**870**	**870**	**870**	**870**	**870**	**870**	**870**	**870**	**870**	**870**
p5.3.q	**1070**	1065	**1070**	**1070**	**1070**	**1070**	1045	**1070**	1065	1065	1065	1065	**1070**	**1070**
p5.3.r	**1125**	1115	1110	**1125**	**1125**	**1125**	1090	**1125**	1120	**1125**	**1125**	**1125**	**1125**	**1125**
p5.3.s	**1190**	1175	1185	**1190**	**1190**	**1190**	1145	**1190**	**1190**	**1190**	1185	1185	1185	**1190**
p5.3.t	**1260**	1240	1250	**1260**	**1260**	**1260**	1240	**1260**	1250	1255	**1260**	**1260**	**1260**	**1260**
p5.3.u	**1345**	1330	1340	**1345**	**1345**	**1345**	1305	**1345**	1330	1335	1335	**1345**	1335	**1345**
p5.3.v	**1425**	1410	1420	**1425**	**1425**	**1425**	**1425**	**1425**	**1425**	**1425**	1420	**1425**	1420	**1425**
p5.3.w	**1485**	1465	**1485**	**1485**	**1485**	**1485**	1460	**1485**	1465	1465	1465	1475	1465	**1485**
p5.3.x	**1555**	1530	**1555**	**1555**	**1555**	**1555**	1520	1540	1535	1540	1540	1535	1540	1550
p5.3.y	**1595**	1580	1590	**1595**	**1595**	**1595**	1590	1590	1590	1590	1590	1580	1590	1590
p5.3.z	**1635**	**1635**	1625	**1635**	**1635**	**1635**	**1635**	**1635**	**1635**	**1635**	**1635**	**1635**	**1635**	**1635**
p5.4.m	**555**	**555**	**555**	**555**	**555**	**555**	550	**555**	**555**	**555**	**555**	550	**555**	**555**
p5.4.o	**690**	680	**690**	**690**	**690**	**690**	680	**690**	**690**	**690**	**690**	**690**	**690**	**690**
p5.4.p	**765**	760	**765**	**765**	**765**	**765**	760	**765**	760	760	760	760	760	760
p5.4.q	**860**	**860**	**860**	**860**	**860**	**860**	830	**860**	**860**	**860**	**860**	835	**860**	**860**
p5.4.r	**960**	**960**	**960**	**960**	**960**	**960**	890	**960**	**960**	**960**	**960**	**960**	**960**	**960**
p5.4.s	**1030**	1000	1025	**1030**	**1030**	**1030**	1020	**1030**	**1030**	**1030**	**1030**	1020	1005	1025
p5.4.t	**1160**	1100	**1160**	**1160**	**1160**	**1160**	**1160**	**1160**	**1160**	**1160**	**1160**	**1160**	**1160**	**1160**
p5.4.u	**1300**	1275	**1300**	**1300**	**1300**	**1300**	**1300**	**1300**	**1300**	**1300**	**1300**	**1300**	**1300**	**1300**
p5.4.v	**1320**	1310	**1320**	**1320**	**1320**	**1320**	1245	**1320**	**1320**	**1320**	**1320**	**1320**	**1320**	**1320**
p5.4.w	**1390**	1380	1375	**1390**	**1390**	**1390**	1330	**1390**	1380	**1390**	1380	1380	1380	**1390**
p5.4.x	**1450**	1410	1440	**1450**	**1450**	**1450**	1410	**1450**	**1450**	**1450**	**1450**	1440	1430	**1450**
p5.4.y	**1520**	**1520**	**1520**	**1520**	**1520**	**1520**	1485	**1520**	1510	1510	1500	1500	**1520**	**1520**
p5.4.z	**1620**	1575	**1620**	**1620**	**1620**	**1620**	1590	**1620**	**1620**	1575	1580	1600	**1620**	**1620**

Table 3
Results test set 6.

Instance	Best	TMH	GTP	GTF	FVF	SVF	GLS	ASe	ADC	ARC	ASi	SVNS	FPR	SPR
p6.2.d	**192**	**192**	**192**	**192**	**192**	**192**	180	**192**	**192**	**192**	**192**	**192**	**192**	**192**
p6.2.j	**948**	936	**948**	**948**	**948**	**948**	**948**	**948**	**948**	**948**	**948**	**948**	942	**948**
p6.2.l	**1116**	**1116**	1098	1110	**1116**	**1116**	1104	**1116**	1110	**1116**	**1116**	**1116**	1110	**1116**
p6.2.m	**1188**	**1188**	1164	**1188**	**1188**	**1188**	1164	**1188**	**1188**	**1188**	**1188**	**1188**	**1188**	**1188**
p6.2.n	**1260**	**1260**	1242	**1260**	**1260**	**1260**	1254	**1260**	**1260**	1254	**1260**	1248	**1260**	**1260**
p6.3.g	**282**	**282**	**282**	**282**	**282**	**282**	264	**282**	**282**	**282**	**282**	276	**282**	**282**
p6.3.h	**444**	**444**	**444**	**444**	**444**	**444**	**444**	**444**	**444**	438	438	**444**	**444**	**444**
p6.3.i	**642**	612	**642**	**642**	**642**	**642**	**642**	**642**	**642**	**642**	**642**	**642**	**642**	**642**
p6.3.k	**894**	876	**894**	**894**	**894**	**894**	882	**894**	888	888	**894**	**894**	**894**	**894**
p6.3.l	**1002**	990	**1002**	**1002**	**1002**	**1002**	990	**1002**	**1002**	**1002**	**1002**	996	**1002**	**1002**
p6.3.m	**1080**	**1080**	**1080**	**1080**	**1080**	**1080**	1068	**1080**	1074	**1080**	**1080**	**1080**	**1080**	**1080**
p6.3.n	**1170**	1152	**1170**	**1170**	**1170**	**1170**	1140	**1170**	1164	1164	1164	1152	1164	**1170**
p6.4.j	**366**	**366**	**366**	**366**	**366**	**366**	360	**366**	**366**	**366**	**366**	**366**	**366**	**366**
p6.4.k	**528**	522	**528**	**528**	**528**	**528**	**528**	**528**	**528**	**528**	**528**	**528**	**528**	**528**
p6.4.l	**696**	**696**	**696**	**696**	**696**	**696**	678	**696**	**696**	**696**	**696**	678	**696**	**696**

that recombines newly generated solutions with a pool of existing elite solutions. Its population based design inherently reduces the probability of getting trapped in local optima. The method is capable of both generating "just good" solutions very fast and, by adapting only one parameter, generating results that are comparable to other state-of-the-art algorithms.

Table 4
Results test set 7.

Instance	Best	TMH	GTP	GTF	FVF	SVF	GLS	ASe	ADC	ARC	ASi	SVNS	FPR	SPR
p7.2.d	**190**	**190**	**190**	**190**	**190**	**190**	**190**	**190**	**190**	**190**	**190**	182	**190**	**190**
p7.2.e	**290**	**290**	**290**	**290**	289	**290**	279	**290**	**290**	**290**	**290**	289	**290**	**290**
p7.2.f	**387**	382	**387**	**387**	**387**	**387**	340	**387**	**387**	**387**	**387**	**387**	**387**	**387**
p7.2.g	**459**	**459**	456	**459**	**459**	**459**	440	**459**	**459**	**459**	**459**	457	**459**	**459**
p7.2.h	**521**	**521**	520	520	**521**	**521**	517	**521**	**521**	**521**	**521**	**521**	**521**	**521**
p7.2.i	**580**	578	579	579	575	579	568	**580**	579	579	579	579	578	**580**
p7.2.j	**646**	638	643	644	643	644	633	**646**	**646**	**646**	**646**	632	**646**	**646**
p7.2.k	**705**	702	702	**705**	704	**705**	691	**705**	704	704	704	700	702	**705**
p7.2.l	**767**	**767**	758	**767**	759	**767**	748	**767**	**767**	**767**	**767**	758	759	**767**
p7.2.m	**827**	817	**827**	824	824	**827**	798	**827**	**827**	**827**	**827**	**827**	816	**827**
p7.2.n	**888**	864	884	**888**	883	**888**	861	**888**	878	878	878	866	**888**	**888**
p7.2.o	**945**	914	933	**945**	**945**	**945**	897	**945**	**945**	940	941	928	932	**945**
p7.2.p	**1002**	987	1000	**1002**	**1002**	**1002**	954	**1002**	991	993	993	955	993	**1002**
p7.2.q	**1044**	1017	1041	1043	1038	**1044**	1031	1043	1042	1043	1043	1029	1043	**1044**
p7.2.r	**1094**	1067	1091	1088	**1094**	**1094**	1075	**1094**	1093	1088	**1094**	1069	1076	**1094**
p7.2.s	**1136**	1116	1123	1128	**1136**	**1136**	1102	**1136**	**1136**	1134	1131	1118	1125	**1136**
p7.2.t	**1179**	1165	1172	1174	1168	**1179**	1142	**1179**	**1179**	**1179**	**1179**	1154	1168	1175
p7.3.h	**425**	416	**425**	**425**	**425**	**425**	418	**425**	**425**	**425**	**425**	**425**	**425**	**425**
p7.3.i	**487**	481	**487**	**487**	**487**	**487**	480	**487**	**487**	486	**487**	480	485	**487**
p7.3.j	**564**	563	**564**	**564**	562	**564**	539	**564**	**564**	**564**	**564**	543	560	**564**
p7.3.k	**633**	632	**633**	**633**	632	**633**	586	**633**	632	**633**	**633**	**633**	**633**	**633**
p7.3.l	**684**	681	683	679	681	681	668	**684**	683	**684**	**684**	681	**684**	**684**
p7.3.m	**762**	756	749	755	745	**762**	735	**762**	**762**	**762**	**762**	743	**762**	**762**
p7.3.n	**820**	789	810	811	814	**820**	789	**820**	819	819	**820**	804	813	**820**
p7.3.o	**874**	**874**	873	865	871	**874**	833	**874**	**874**	**874**	**874**	841	859	**874**
p7.3.p	**929**	922	917	923	926	927	912	**929**	925	926	925	918	925	927
p7.3.q	**987**	966	976	**987**	978	**987**	945	**987**	**987**	**987**	**987**	966	970	**987**
p7.3.r	**1026**	1011	1018	1022	1024	1022	1015	**1026**	1024	1021	1022	1009	1017	1021
p7.3.s	**1081**	1061	**1081**	**1081**	1079	1079	1054	**1081**	**1081**	**1081**	1077	1070	1076	**1081**
p7.3.t	**1118**	1098	1114	1116	1112	1115	1080	**1118**	1117	1103	1117	1109	1111	**1118**
p7.4.g	**217**	**217**	**217**	**217**	**217**	**217**	209	**217**	**217**	**217**	**217**	**217**	**217**	**217**
p7.4.h	**285**	**285**	**285**	**285**	**285**	**285**	**285**	**285**	**285**	**285**	**285**	283	**285**	**285**
p7.4.i	**366**	359	**366**	**366**	**366**	**366**	359	**366**	**366**	**366**	**366**	364	**366**	**366**
p7.4.k	**520**	503	**520**	**520**	518	**520**	511	**520**	**520**	**520**	**520**	518	518	518
p7.4.l	**590**	576	**590**	588	588	**590**	573	**590**	**590**	**590**	**590**	575	581	**590**
p7.4.m	**646**	643	644	**646**	**646**	**646**	638	**646**	644	**646**	**646**	639	**646**	**646**
p7.4.n	**730**	726	723	721	715	**730**	698	**730**	725	725	726	723	723	**730**
p7.4.o	**781**	776	772	778	770	**781**	761	**781**	778	**781**	778	778	780	780
p7.4.p	**846**	832	841	839	**846**	**846**	803	**846**	**846**	838	842	841	842	**846**
p7.4.q	**909**	905	902	898	899	906	899	**909**	**909**	**909**	**909**	896	902	907
p7.4.r	**970**	966	**970**	969	**970**	**970**	937	**970**	**970**	**970**	**970**	964	961	**970**
p7.4.s	**1022**	1019	1021	1020	1021	**1022**	1005	**1022**	1019	1021	1019	1019	**1022**	**1022**
p7.4.t	**1077**	1067	1071	1071	**1077**	**1077**	1020	**1077**	1072	**1077**	**1077**	1073	1066	**1077**

Table 5
Summary of results.

	TMH	GTP	GTF	FVF	SVF	GLS	ASe	ADC	ARC	ASi	SVNS	FPR	SPR
# Best	34	69	94	97	128	21	130	80	81	84	44	**78**	**131**
Avg. gap (%)	1.32	0.49	0.20	0.18	0.05	2.53	0.08	0.35	0.40	0.32	0.97	**0.39**	**0.04**
Avg. CPU (s)	336.6	100.0	146.6	18.9	268.6	7.8	252.3	213.8	204.8	215.0	3.8	**5.0**	**212.4**

Table 6
Execution times (s).

Test set	TMH	GTP	GTF	FVF	SVF	GLS	ASe	ADC	ARC	ASi	SVNS	FPR	SPR
set4	796.7	105.29	282.92	22.52	457.89	11.4	370.9	317.9	307.4	320.4	7.4	**8.6**	**367.4**
set5	71.3	69.45	26.55	34.17	158.93	3.5	173.6	150.6	143.3	151.3	1.5	**2.9**	**119.9**
set6	45.7	66.29	20.19	8.74	147.88	4.3	161.1	140.8	135.2	141.7	1.9	**2.1**	**89.6**
set7	432.6	158.97	256.76	10.34	309.87	12.1	303.5	245.9	233.1	246.5	4.3	**6.3**	**272.8**

5. Conclusions

This paper presents the application of the Path Relinking metaheuristic to the Team Orienteering Problem, in combination with a Greedy Randomised Adaptive Search Procedure. The Path Relinking extension significantly improves the results of the basic GRASP algorithm. A fast and a slow variant of the approach, which differ only in one parameter setting, are presented and their results are compared to other state-of-the-art approaches. The fast variant achieves high solution quality in a small amount of calculation time: an average gap of 0.39% to the best known solutions in 5.0 s. The slow variant achieves solutions that deviate on average only 0.04% from

Table 7
FPR compared to the literature.

Approach	Better	Equal	Worse
TMH	**101**	37	19
GTP	**49**	62	46
GTF	26	72	**59**
FVF	23	70	**64**
SVF	4	77	**76**
GLS	**128**	19	10
ASe	3	83	**71**
ADC	44	69	44
ARC	42	71	**44**
ASi	31	80	**46**
SVNS	**88**	44	25

Table 8
SPR compared to the literature.

Approach	Better	Equal	Worse
TMH	**121**	35	1
GTP	**82**	69	6
GTF	**55**	88	14
FVF	**52**	93	12
SVF	**19**	120	18
GLS	**133**	24	0
ASe	**19**	122	16
ADC	**70**	77	10
ARC	**69**	79	9
ASi	**66**	81	10
SVNS	**108**	48	1

the best known results in a reasonable computation time of 272.8 s. Next to achieving most of the best known solutions for many test-problems, the slow variant improved the best known results in five instances. The presented metaheuristic cumulates the advantages of being efficient, effective and simple: it achieves top class results in a reasonable computation time, it is simple to understand and to implement, and moreover, it is almost parameter free.

The comparison shows very small gaps for the different approaches, which suggests that the test sets are (almost) solved to optimality. Future research on the TOP should focus on solving larger instances.

Acknowledgements

Pieter Vansteenwegen is a post doctoral research fellow of the Fonds Wetenschappelijk Onderzoek - Vlaanderen (FWO).

References

[1] Archetti C, Hertz A, Speranza M. Metaheuristics for the team orienteering problem. Journal of Heuristics 2007;13:49–76.
[2] Arkin E, Mitchell J, Narasimhan G. Resource-constrained geometric network optimization. In: Proceedings of the 14th ACM symposium on computational geometry, 1998. p. 307–316.
[3] Boussier S, Feillet D, Gendreau M. An exact algorithm for the team orienteering problem. 4OR 2007;5:211–30.
[4] Butt S, Cavalier T. A heuristic for the multiple tour maximum collection problem. Computers and Operations Research 1994;21:101–11.
[5] Butt S, Ryan D. An optimal solution procedure for the multiple tour maximum collection problem using column generation. Computer and Operations Research 1999;26:427–41.
[6] Chao I, Golden B, Wasil E. Theory and methodology—the team orienteering problem. European Journal of Operational Research 1996;88:464–74.
[7] Chao I-M, Golden B, Wasil E. A fast and effective heuristic for the orienteering problem. European Journal of Operational Research 1996;88(3):475–89.
[8] Feillet D, Dejax P, Gendreau M. Traveling salesman problems with profits. Transportation Science 2005;39:188–205.
[9] Feo TA, Resende MGC. A probabilistic heuristic for a computationally difficult set covering problem. Operations Research Letters 1989;8:67–71.
[10] Fischetti M, Salazar J, Toth P. Solving the orienteering problem through branch-and-cut. INFORMS Journal on Computing 1998;10:133–48.
[11] Gendreau M, Laporte G, Semet F. A branch-and-cut algorithm for the undirected selective travelling salesman problem. Technical Report CRT-95-80, Centre de recherche sur les transports, Montreal, Canada; 1995.
[12] Gendreau M, Laporte G, Semet F. A branch-and-cut algorithm for the undirected selective travelling salesman problem. Networks 1998;32:263–73.
[13] Gendreau M, Laporte G, Semet F. A tabu search heuristic for the undirected selective travelling salesman problem. European Journal of Operational Research 1998;106:539–45.
[14] Glover F. Tabu search and adaptive memory programming. Interfaces in Computer Science and Operations Research. Dordrecht: Kluwer; 1996 p. 1–75.
[15] Golden B, Levy L, Vohra R. The orienteering problem. Naval Research Logistics 1987;34:307–18.
[16] Golden B, Wang Q, Liu L. A multifaceted heuristic for the orienteering problem. Naval Research Logistics 1988;35:359–66.
[17] Hillier FS. Efficient heuristic procedures for integer linear programming with an interior. Operations Research 1969;17:600–37.
[18] Ho SC, Gendreau M. Path relinking for the vehicle routing problem. Journal of Heuristics 2006;12:55–72.
[19] Ke L, Archetti C, Feng Z. Ants can solve the team orienteering problem. Computers & Industrial Engineering 2008;54:648–65.
[20] Keller P. Algorithms to solve the orienteering problem: a comparison. European Journal of Operational Research 1989;41:224–31.
[21] Laguna M, Martí R. Grasp and path relinking for 2-layer straight line crossing minimization. INFORMS Journal on Computing 1999;11:44–52.
[22] Laporte G. The vehicle routing problem: an overview of exact and approximate algorithms. European Journal of Operational Research 1992;59:345–58.
[23] Laporte G, Martello S. The selective travelling salesman problem. Discrete Applied Mathematics 1990;26:193–207.
[24] Liang Y, Kulturel-Konak S, Smith A. Meta heuristics for the orienteering problem. In: Proceedings of the 2002 congress on evolutionary computation, Honolulu, Hawai, 2002. p. 384–9.
[25] Lin S. Computer solutions of the traveling salesman problem. Bell System Technical Journal 1965;44:2245–69.
[26] Pekny J, Miller D. An exact parallel algorithm for the resource constrained travelling salesman problem with application to scheduling with an aggregate deadline. In: Proceedings of the ACM annual conference on cooperation, 1990.
[27] Prins C, Prodhon C, Wolfler Calvo R. Solving the capacitated location-routing problem by a grasp complemented by a learning process and a path relinking. 4OR 2006;4:221–38.
[28] Ramesh R, Brown K. An efficient four-phase heuristic for the generalized orienteering problem. Computers and Operations Research 1991;18:151–65.
[29] Ramesh R, Yoon Y, Karwan M. An optimal algorithm for the orienteering tour problem. ORSA Journal on Computing 1992;4:155–65.
[30] Reghioui M, Prins C, Labadi N. Grasp with path relinking for the capacitated arc routing problem with time windows. In: EvoWorkshops, 2007. p. 722–31.
[31] Righini G, Salani M. Dynamic programming for the orienteering problem with time windows. Technical Report 91, Dipartimento di Tecnologie dellInformazione, Universita degli Studi Milano, Crema, Italy, 2006.
[32] Souffriau W, Vansteenwegen P, Vertommen J, Vanden Berghe G, Van Oudheusden D. A personalised tourist trip design algorithm for mobile tourist guides. Applied Artificial Intelligence 2008;22(10):964–85.
[33] Tang H, Miller-Hooks E. A tabu search heuristic for the team orienteering problem. Computer and Operations Research 2005;32:1379–407.
[34] Tasgetiren M. A genetic algorithm with an adaptive penalty function for the orienteering problem. Journal of Economic and Social Research 2001;4(2):1–26.
[35] Tsiligirides T. Heuristic methods applied to orienteering. Journal of the Operational Research Society 1984;35(9):797–809.
[36] Vansteenwegen P, Van Oudheusden D. The mobile tourist guide: an OR opportunity. OR Insight 2007;20(3):21–7.
[37] Vansteenwegen P, Souffriau W, Vanden Berghe G, Van Oudheusden D. Metaheuristics for tourist trip planning. Metaheuristics in the Service Industry. In: Lecture notes in economics and mathematical systems, vol. 624. Berlin: Springer; to appear (ISBN: 978-3-642-00938-9).
[38] Vansteenwegen P, Souffriau W, Vanden Berghe G, Van Oudheusden D. A guided local search metaheuristic for the team orienteering problem. European Journal of Operational Research 2009;196(1):118–27.
[39] Wang Q, Sun X, Golden B, Jia J. Using artificial neural networks to solve the orienteering problem. Annals of Operations Research 1995;61:111–20.