Journal of Applied Sciences 13 (6): 876-882, 2013
ISSN 1812-5654 / DOI: 10.3923/jas.2013.876.882
© 2013 Asian Network for Scientific Information

Constructive Heuristics for Team Orienteering Problems

[1]Hamzah Ali Alkhazaleh, [1]Masri Ayob, [1]Zalinda Othman and [2]Zulkifli Ahmad
[1]Center for Artificial Intelligence, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM, Bangi, Selangor, Malaysia
[2]School of Linguistics and Language Studies, Faculty of Social Sciences and Humanities, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

Abstract: This work investigates different constructive heuristics to initial trial solutions for Team Orienteering Problem (TOP). The goal of TOP is to build a particular number of routes that visit some points to maximize the sum of the score, while the route's length does not exceeding the time budget. The initial solutions were constructed by random generation methods, nearest neighbor greedy heuristic, randomly selection method (random generation methods and nearest neighbor greedy heuristic) and two type of Greedy Random Adaptive Search Procedures (GRASP) (cardinality-based criteria (greedy value) and value-based criteria (threshold value)). These constructive heuristics were tested using few instances on benchmark datasets. Results were compared among each other based on solutions' quality value and diversity value. The comparison of results concludes that the nearest neighbor greedy heuristic gain better initial solutions that balanced between solution quality and solution diversity.

Key words: Team orienteering problem, GRASP, nearest neighbor, greedy heuristic, random heuristic

INTRODUCTION

Orienteering Problem (OP) was taken originally from sport skiing game of orienteering (Chao *et al*., 1996) where players will start at a point and need to visit several point which has its own score before reach to the end in a specific time. The objective is to maximize the total collected scores before they reach the end point within the time. Meanwhile, the Team Orienteering Problem (TOP) (Chao *et al*., 1996) or multiple tour maximum collection problems (Butt and Cavalier, 1994) is an Orienteering Problem (OP) where the objective is to determine the route P that brings total high scores, within timeframe of Tmax. Each team consists of several players, where each of them collects scores during the same time span. Golden *et al*. (1988) proved that OP is a Non-deterministic Polynomial-time (NP)-hard problem. Therefore, the exact methods could not solve this problem in a reasonable computation time. To date, very few researchers have attempted to tackle the TOP.

Many researches focused on applying metaheuristic approaches to solve TOP. For example, Ke *et al*. (2008) had proposed the use of sequential, deterministic-concurrent, random-concurrent and simultaneous methods in the ant colony optimization approach to construct candidate solutions. Vansteenwegen *et al*. (2009a) combined guided local search and he used a greedy construction heuristic in the construction phase. Vansteenwegen *et al*. (2009b) published another work that applied skewed variable neighborhood search with greedy construction heuristic to initialize the solutions to solve the TOP. Souffriau *et al*. (2010) introduced path relinking approach that hybridized with greedy random adaptive search procedure to construct the initial solution. On another hand, Bouly *et al*. (2010) introduced a memetic algorithm that use an iterative destruction/construction heuristic and random generation procedure to initialize the population, optimal split procedure for chromosome evaluation and local search techniques for mutation. Recently, Muthuswamy and Lam (2011) applied discrete particle swarm optimization algorithm that combine three methods to generate the population. These are score/distance, cumulative density function and random generation procedure. This approach had shown that it obtained good quality solution within less than a minute of computation time.

Some metaheuristic algorithms start with the construction phase to solve the problems by generating initial solutions. The type of method used in the construction phase in metaheuristic algorithm plays an important role in the effectiveness and efficiency of the

Corresponding Author: Masri Ayob, Center for Artificial Intelligence, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM, Bangi, Selangor, Malaysia

algorithm. Random and greedy approaches are the two main strategies used to generate initial solution. There is a tread-off of using random approach or greedy approach in terms of quality and diversity of the solution and computational time. The answer of this tread-off mainly depends on the effectiveness and efficiency of the random and greedy algorithm itself and the metaheuristic algorithms properties (Talbi, 2009).

Construct the initial solution using random approach is quick operation and diverse sampling from the entire search space, but the metaheuristic algorithms need a much number of iteration to converge. On another hand, generating the initial solution using greedy heuristic can speed up the improvement search. In many cases, the greedy heuristic can reduce the computational time (at metaheuristic algorithm) to converge to a better quality local optima. Unfortunately, the use of better solution as initial solution does not always guarantee a better local optima result. To enhance the robustness, the combination of random and greedy approaches can be used to initialize a pool of solutions that balanced between quality and diversity of solution (Talbi, 2009). One of the most popular metaheuristic algorithms that focus on creating good quality and diverse solutions is scatter search.

Therefore, this work investigated five constructive heuristics. These are (1) Random generation method, (2) Nearest neighbor greedy heuristic, (3) Random selection method (random generation methods and nearest neighbor greedy heuristic), (4) Greedy random adaptive search procedure (GRASP) with cardinality-based criteria (greedy value) and (5) Greedy random adaptive search procedure (GRASP) with value-based criteria (threshold value). A goal of this study is to study these methods in terms of solution quality and diversity.

PRELIMINARIES

Formulation of the Team Orienteering Problem: TOP can be defined as a set of points, V = {1,..., n} that each point i has specific score S_i. The starting point is 1 and the ending point is n and $S_1 = S_n = 0$. The traveling time/distance between each two points (i,j) is c_{ij}. TOP consist of finding m routes that starts at point 1 and ends at point n such that the total score of visited points is maximized. Each point can be visited at most once. For each route, the total traveling time taken to visit the points can not exceed the fixed limit T_{max}. The evaluation function, f (x_p) for the x_p solution of TOP is formulated as in Eq. 1 (Ke *et al*., 2008):

$$f(x_p) = \mathrm{Max}\sum_{i=2}^{n-1}\sum_{k=1}^{m} S_i y_{ik} \tag{1}$$

$$\textbf{Subject to} \sum_{j=2}^{n}\sum_{k=1}^{m} x_{1jk} = \sum_{i=1}^{n-1}\sum_{k=1}^{m} x_{ink} = m \tag{2}$$

$$\sum_{i<j} x_{ijk} + \sum_{j>i} x_{jik} = 2y_{jk}; \forall lj = 2,\ldots,n-1; \forall lk = 1,\ldots,m \tag{3}$$

$$\sum_{k=1}^{m} y_{ik} \le 1; \forall i = 2,\ldots,n-1 \tag{4}$$

$$\sum_{i=1}^{n-1}\sum_{j>2} c_{ij}x_{ijk} \le T_{max}; \forall k = 1,\ldots,m \tag{5}$$

$$\sum_{\substack{i,j\in U\\ i<j}} x_{ijk} \le |U|-1; U \subset V \setminus \{1,n\}; 2 \le |U| \le n-2; \forall k = 1,\ldots,m \tag{6}$$

$$x_{ijk} 1\{0,1\} (1 \le i \le j \le n; \forall k = 1,\ldots,m) \tag{7}$$

$$y_{1k} = y_{nk} = 1, y_{ik} \in \{0,1\}; \forall i = 2,\ldots,n; \forall k = 1,\ldots,m \tag{8}$$

where, S_i is the score associated with point i, c_{ij} is the traveling time between point i and j, k is the route (k = 1, ..., m). Let y_{ik} = 1 (i = 1, ..., n; k = 1,...,m) if point i is visited in route k, otherwise y_{ik} = 0. Let x_{ijk} = 1 (i, j = 1, ..., n; k = 1, ..., m) if edge (i, j) is visited in path k, otherwise x_{ijk} = 0. Since, $c_{ij} = c_{ji}$, only x_{ijk} (i < j) are defined. Let U be a subset of V (V is a set of points V = 1, ..., n).

Equation 1 is the objective function to maximize the collected score of the visited points, which is calculated in the routes subject to constraint (2)-(8). Constraint (2) ensures that each route must starts at point 1 and ends at point n. Constraint (3) ensures that if a point is visited in a given route, it is outstripped and followed by an exact one other visit in the same route. Constraint (4) ensures that each point is only visited one time at most. Constraint (5) guarantees that each route does not exceed the time budget T_{max}. Constraint (6) ensure the sub-route is prohibited. Constraints (7) and (8) set the integral requirement on each variable.

Calculations the diversity (difference value) between to solutions: Maquera *et al*. (2011) considered the difference value between the solutions to measure the diversity. The difference value between the two solution x and y is calculated by creating a two dimensional matrix PointCount of size r×r, where r is the maximum number of routes in either x or y (in our case the number of routes in the solution is fixed). Then the matrix will be filled up in

Table 1: Example of PointCount matrix with r = 3

	y_1	y_2	y_3
x_1	1	1	1
x_2	3	1	1
x_3	1	2	1

such a way that PointCount (x_i ,y_j), which includes the number of points that matching route ith in solution x with route jth in solution y. After finish the process above, we compute the total number of points in the solution either x or y, since not all point should be included in the TOP solution, we compute the total of points for both solution then we divide that by 2 (n/2), where n is the total point. Then find the largest PointCount value, sum this value and eliminate the corresponding column and row. The process will finish when PointCount become empty. The sum value will be subtracted from the total number of points and the difference will be a measure of the difference value between two solutions x and y.

The following example illustrates the difference value calculation between two solutions x and y:

x: (6, 4, 3) (2, 5, 13, 8, 10) (12, 7, 1, 9, 11)
y: (3, 5, 13, 2, 11) (4,10, 1, 9) (6, 12, 8, 7)

The PointCount matrix will be filled as in Table 1 and value of r is 3.

The first value (x1, y1) in the matrix in Table 1 is the number of points that mutual between route x1 and route y1. This operation is repeated till all the PointCount matrix is filled. The maximum number in the matrix will be saved, then delete the all values in the same column and raw. This operation is repeated till no more values in the matrix. Then, the sum of the saved values is calculated to detect it from the average number of points in both solutions. For example, after fill up PointCount matrix with number of points which mutual between paths, the value (x2, y1) is selected as maximum value 3 in the matrix, then the corresponding column and row will be deleted. The same procedure is done for value (x3, y2). The only remaining element will be (x1, y3) which has count of 1. The sum of the maximum values in PointCount matrix is 6. After that, we find the average of the total point in both solutions by dividing that by 2, (26/2 = 13). Then the difference value between solution x and y is given by 13-6 = 7.

We calculate the difference value (diversity) for each solution x in the population P with the remaining solutions in the P, then we find the average for each solution corresponding other solution in the P.

CONSTRUCTIVE HEURISTIC

To create some good solutions for an initial population, this work investigated five constructive heuristics and evaluated them based on the solution quality, diversity and computational time. These are (1) Random generation method, (2) Nearest neighbor greedy algorithm, (3) Select insertion methods randomly (random generation methods and nearest neighbor greedy heuristic), (4) Greedy random adaptive search procedure (GRASP) with cardinality-based criteria (greedy value), and (5) Greedy random adaptive search procedure (GRASP) with value-based criteria (threshold value).

Random generation method: Random generation method is the most common heuristic that used to construct the initial solution in many combinatorial optimization problems. This method generates a random permutation with uniform probability, i.e., the generated solution might be not feasible (Talbi, 2009).

We used this heuristic to construct the initial solution for TOP. The member of the team starts with an empty route that only has starting and ending point. The remaining point that unvisited yet will be selected and inserted in the route randomly under some conditions, which it the selected point must be not inserted before starting point, after ending point and not lead the path to exceed the traveling time limit T_{max}. If the selected point passes those conditions, it will be set as a visited point. This procedure will be repeated till no more traveling time in the route (team member). Then, the algorithm starts again with new empty path that only has starting and ending point. The whole procedure will be repeated until all points are visited. Then, the best route will be selected as a team member among initialed routes to complete one solution.

Nearest neighbor greedy algorithm: Nearest neighbor greedy algorithm is a very popular heuristic that was used in many fields since its conception in early fifties (Fix and Hodges, 1952, 1989). For example, nearest neighbor greedy heuristic can be used to construct the solution for traveling salesman problem. Since, the start and end points are fixed in TOP, the new route starts with an empty path that only has start and end points. Then, the heuristic randomly selects point i from unvisited point and insert it in the route without violating the hard constraints. After that, it checks the remaining unvisited points and selects the point j that minimizes the distance between i and j and repeat the same process with point j and unvisited points until the traveling time in the route reach the limit. The heuristic is repeated until all point is visited by the route. Then, the best paths will be selected as a team member among initialed routes to complete one solution.

Select insertion methods randomly (random generation methods and nearest neighbor greedy heuristic): This heuristic starts with an empty route that only has starting and ending points. Then, the heuristic select point i from unvisited point using random selection or nearest point to the previous point. Then, insert it in the route without violating the hard constraints. This process is repeated until end of traveling time in the route. The heuristic is repeated until all points are visited or reach the time limit. Then, the best paths will be selected as team members among initialed paths to complete one solution.

Greedy random adaptive search procedure (GRASP): GRASP is a meta-heuristic approach to optimize combinatorial problem. It is usually implemented in multi-start procedure, where each iteration consists of construction phase and local search phase. In the former phase, a random greedy solution is constructed. Then, the latter phase starts with the constructed solution, will be iteratively applied to improve the solution until it found the local optimal solution (Feo and Resende, 1995). However, this work concentrated on the construction phase.

The construction phase in general, consists of creating a list that includes the candidate points that can be inserted in the partial solution with keeping the feasibility. Each candidate point in the list has been evaluated by a greedy function. Then, from the first list, a sub-list is generated based on greedy value of the candidate point. This sub-list is representing the Restricted Candidate List (RCL). The RCL list play an important role in GRASP metaheuristic, because it represents the probabilistic aspect of the algorithm. The restricted criteria depend on two way, which it cardinality-based criteria and value-based criteria. At each iteration, a random point is selected from RCL and inserts it to partial solution (this step denote to the probabilistic aspect of the heuristic). Once the point inserted to partial solution, the RCL list is updated. To update RCL list, the greedy value of the unvisited candidate points must be reevaluated and update RCL list with new candidate points (this is the adaptive aspect of the heuristic) (Hart and Shogan, 1987; Feo and Resende, 1989).

In the TOP, the heuristic starts with an empty path that only has starting point and end point. Then, the heuristic check all unvisited point and determine the feasible point and add it to candidate list. The greedy value is computed for all points in the candidate list based greedy function which it: $g\ (l)_{ilj} = S_l/t_{ilj}$, where, g (l) is the greedy value of the candidate point l. S_l is the point's score which it already fixed in data set. t_{ilj} is the traveling time between previous point i and inserted point l and next point j which it the ending point in the path. From the candidate list, the heuristic create a RCL. The RCL is created based on two conditions:

- **Cardinality-based criteria (greedy value):** In this case, the RCL list is built of the p best points in terms of greedy value from the candidate list. Where, the parameter p denote to maximum number of points in the RCL. The heuristic select the best 5 points in terms of greedy value from the candidate list to build the RCL. The parameter p is equal 5 based in the preliminary study
- **Value-based criteria (threshold value):** In this case, the RCL is built of the points that have greedy value greater or equal to the threshold value. The threshold value is $g_{min}+\alpha(g_{max}-g_{min})$, where, g_{min} is the minimum greedy value in the candidate list. g_{max} is the maximum greedy value in the candidate list. α is random value less than 1 and greater than 0 i.e., (0001, 0.999). After the threshold is computed, the point's greedy values in the candidate list are compared with the threshold value. All the points that have greedy value greater or equal to the threshold value, it added to RCL. i.e., the point i is selected from the candidate list, if $g\ (I) = g_{min}+\alpha\ (g_{max}-g_{min})$ the point i will be added to RCL. Where the g (i) is greedy value of the point i

After RCL list is created, the heuristic chose points randomly from RCL list to insert it to the path. Once the point is incorporated to the path, all the unvisited points are evaluated by greedy function and create new candidate list. Then, the RCL list is updated based on the new candidate list. This process repeated until no more traveling time in the path. The heuristic is repeated until all point is visited by the paths. Then, the best paths will be selected as team members among initialed paths to complete one solution.

EXPERIMENTAL RESULTS AND COMPARISON

To estimate the benefit of using the constructive heuristic in terms of solution quality and diversity, this work compared five constrictive heuristics between each other. These constructive heuristics is labelled as follow:

H1: Random generation method
H2: Nearest neighbor greedy algorithm

Table 2: Results based on solution quality value (51 independent runs)

	H2				H4				H5				H3				H1			
	Min.	Max.	Avg.	SD	Min.	Max.	Avg.	SD	Min.	Max.	Avg.	SD	Min.	Max.	Avg.	SD	Min.	Max.	Avg.	SD
p4.2.l (Q)	519	827	685.98	60.69	218	314	260.03	22.12	257	395	320.04	30.24	177	283	215.88	25.02	146	239	188.05	18.67
p4.3.l (Q)	518	717	606.36	52.16	256	380	298.58	25.39	285	409	346.04	28.31	151	266	211.98	26.27	167	242	200.88	16.36
p4.4.l (Q)	453	647	526.78	38.39	270	369	319.92	24.09	301	458	363.82	29.92	189	290	225.58	23.57	174	245	206.62	19.57
p5.2.r (Q)	925	1145	1040.6	55.48	305	465	390.08	39.19	390	615	495.07	56.18	280	420	333.09	36.97	260	410	315.04	30.65
p5.2.r (Q)	715	985	871.50	53.87	355	520	433.08	38.70	425	645	525.05	49.32	290	440	350.08	34.84	265	360	308.08	21.77
p5.4.r (Q)	575	735	669.00	38.20	405	580	472.05	39.77	435	595	518.09	33.42	280	450	343.09	37.95	250	360	304.07	23.74
p6.2.l (Q)	666	948	823.44	74.57	258	444	343.32	36.55	348	600	441.84	46.61	228	384	286.02	29.55	210	330	261.36	25.63
p6.3.l (Q)	426	852	641.64	101.23	288	432	371.28	34.66	366	522	445.92	31.99	198	336	258.84	33.92	198	324	254.16	25.82
p6.4.l (Q)	288	504	397.56	46.94	270	372	320.76	23.45	300	432	350.04	32.25	162	252	209.04	21.86	174	282	216.00	23.88
p7.2.n (Q)	481	663	576.66	41.17	192	355	251.74	29.51	234	357	301.24	31.66	143	220	185.94	18.19	152	217	182.38	16.80
p7.3.n (Q)	451	609	516.96	36.37	257	370	300.48	24.60	254	426	343.00	33.16	162	239	197.24	19.74	144	255	195.00	22.81
p7.4.n (Q)	429	555	495.48	29.82	280	403	336.48	32.33	314	437	369.96	25.12	172	267	213.76	21.81	166	253	205.04	19.32

Values in Max., Min. and Avg. columns are the quality of the solution

Table 3: Results based on solution diversity value (51 independent runs)

	H2				H4				H5				H3				H1			
	Min.	Max.	Avg.	SD	Min.	Max.	Avg.	SD	Min	Max.	Avg.	SD	Min.	Max.	Avg.	SD	Min.	Max.	Avg.	SD
p4.2.l (D)	1.08	8.94	4.80	1.93	0.36	0.98	0.62	0.13	0.03	1.46	0.79	0.25	0.01	0.52	0.31	0.11	0.08	0.48	0.21	0.09
p4.3.l (D)	1.05	7.76	4.16	1.30	0.54	1.78	1.09	0.24	0.64	2.04	1.52	0.35	0.08	0.56	0.33	0.13	0.08	0.44	0.24	0.08
p4.4.l (D)	1.16	6.09	3.22	0.97	1.00	2.26	1.64	0.28	1.22	4.16	2.63	0.72	0.16	1.08	0.51	0.20	0.08	0.68	0.34	0.14
p5.2.r (D)	3.32	6.06	4.93	0.77	0.38	0.92	0.62	0.12	0.58	1.06	1.06	0.27	0.16	0.58	0.30	0.10	0.18	0.06	0.34	0.10
p5.2.r (D)	1.84	3.56	2.72	0.41	0.38	1.06	0.73	0.16	0.05	1.48	0.98	0.25	0.06	0.62	0.36	0.12	0.01	0.52	0.26	0.09
p5.4.r (D)	0.56	1.88	1.32	0.33	0.48	1.18	0.74	0.15	0.52	1.46	0.93	0.21	0.06	0.52	0.28	0.11	0.04	0.04	0.19	0.07
p6.2.l (D)	0.32	3.46	1.12	0.73	0.52	1.32	0.79	0.16	0.54	1.07	1.14	0.27	0.14	0.62	0.40	0.12	0.16	0.52	0.33	0.08
p6.3.l (D)	1.00	3.42	2.19	0.55	0.56	1.42	0.94	0.23	0.07	1.09	1.20	0.25	0.08	0.56	0.27	0.11	0.04	0.52	0.26	0.10
p6.4.l (D)	0.32	2.00	1.02	0.30	0.32	1.36	0.83	0.23	0.64	2.28	1.34	0.28	0.06	0.34	0.17	0.06	0.02	0.36	0.18	0.08
p7.2.n (D)	0.36	5.64	1.44	0.98	0.02	1.02	0.54	0.20	0.03	1.86	1.11	0.37	0.02	0.36	0.13	0.06	0.02	0.28	0.13	0.06
p7.3.n (D)	0.92	4.86	2.16	0.65	0.05	2.06	1.20	0.36	0.66	3.04	2.03	0.53	0.02	0.34	0.15	0.09	0.02	0.32	0.17	0.09
p7.4.n (D)	1.62	4.82	3.25	0.81	0.54	2.22	1.54	0.46	1.09	4.52	3.06	0.66	0.02	0.66	0.29	0.14	0.02	0.06	0.26	0.13

Values in Max., Min. and Avg. columns are the diversity values

H3: Select insertion methods randomly (random generation methods and nearest neighbor greedy heuristic)

H4: Greedy random adaptive search procedure (GRASP) based on cardinality-based criteria (greedy value)

H5: Greedy random adaptive search procedure (GRASP) based on value-based criteria (threshold value)

The proposed constructive heuristics were tested on TOP. There are four datasets used to benchmark different approaches from Chao *et al.* (1996). Each data sets contain different numbers of locations: n = 100 (date set 4), n = 66 (data set 5), n = 64 (data set 6) and n = 102 (date set 7); including start and end locations. Each set contains instances with r equal to 2, 3 and 4 routes. The time budget T_{max} differs for each data set. The constructive heuristics were coded in Java 1.7 and performed on Intel Pentium (R) Daul-Core 3.0 GHz CPU personal computer with 2 gigabyte RAM, running on Windows 7 operating system (32-bit).

Three instances from each data set were selected, which it distributed equally between each particular data set based on the number of route, to test the constructive heuristic. We report the minimum, maximum, average and standard deviation over 51 independent runs in Table 2 and 3 for the results with regards to quality and diversity of solution, respectively. The reason of executed 51 run as odd number is to calculate the median easily. Table 2 clearly shows that the H2 got best average quality value of the solutions comparing with other heuristic across all instances. Moreover, Table 3 shows that H2 also got the best average diversity value for most instances of the solution diversity comparing with other heuristics. H5 heuristic achieved average results better than other heuristics in two instances which it P6.2.l and P6.4.l.

The statistical analysis for all constructive heuristic has been done based on Wilcoxon test. The P. Sig value is reported for each two heuristic with the same instances. Whereas, P. Sig value is to show the performance of each heuristic with other heuristic is statistically significant difference. These two values are measured for the quality value of the solutions in Table 4 and for the diversity value of the solutions in Table 5. We can see in Table 4 that almost all the P. Sig is less than 0.05 that is mean it's significant, unless H1 with H3 in some instances in dataset 6 with p6.3.l = 0.44 and p6.4.l = 0.592 and dataset

Table 4: Statistical analysis of the solution quality value based Wilcoxon test (51 independent runs)

Instances		H4-H2	H5-H2	H3-H2	H1-H2	H5-H4	H3-H4	H1-H4	H3-H5	H1-H5	H1-H3
P4.2.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P4.3.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010
P4.4.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001
P5.2.r	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.011
P5.3.r	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P5.4.r	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P6.2.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P6.3.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.440
P6.4.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.592
P7.2.n	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.302
P7.3.n	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.521
P7.4.n	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.021

Table 5: Statistical analysis of the solution diversity value based Wilcoxon test (51 independent runs)

Instances		H4-H2	H5-H2	H3-H2	H1-H2	H5-H4	H3-H4	H1-H4	H3-H5	H1-H5	H1-H3
P4.2.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P4.3.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P4.4.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P5.2.r	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P5.3.r	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P5.4.r	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P6.2.l	P. Sig	0.002	0.085	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P6.3.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.014
P6.4.l	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.009
P7.2.n	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.513
P7.3.n	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
P7.4.n	P. Sig	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

7 with p7.2.n = 0.302 and p7.3.n = 0.521. On other hand, Table 5 shows that just two non-significant of *P. Sig* which it H5 with H2 with instance p6.2.l = 0.085 in dataset 6 and H1 with H3 with instance p7.2.n = 0.513 in dataset 7.

CONCLUSION

The team orienteering problem is a difficult combinatorial optimization problem and a suitable platform to study the effectiveness of the search mechanism. This paper presents five heuristics to construct the initial solutions for TOP. These constructive heuristics were compared between each other based on the quality and diversity, in order to select the one that balances between the quality and diversity of solution. The performance of each heuristic with other heuristic is statistically significant difference was proofed by statistical analysis using Wilcoxon test. The advantages of the proposed heuristics are efficient, effective and easy to implement.

The nearest neighbor greedy algorithm got the better average value of quality and diversity. For the future work, scatter search algorithm that give freedom to select the constructive heuristic, nearest neighbor greedy algorithm can be used as diversification generation method to initialize the trial solution.

ACKNOWLEDGMENTS

The authors wish to thank Ministry of Higher Education for supporting this work under the FRGS Research Grant Scheme (FRGS/1/2012/SG05/UKM/02/11).

REFERENCES

Bouly, H., D.C. Dang and A. Moukrim, 2010. A memetic algorithm for the team orienteering problem. Q. J. Oper, Res., 8: 49-70.

Butt, S.E. and T.M. Cavalier, 1994. A heuristic for the multiple tour maximum collection problem. Comput. Oper. Res., 21: 101-111.

Chao, I., B.L. Golden and E.A. Wasil, 1996. A fast and effective heuristic for the orienteering problem. Eur. J. Oper. Res., 88: 475-489.

Feo, T.A. and M.G.C. Resende, 1989. A probabilistic heuristic for a computationally difficult set covering problem. Oper. Res. Lett., 8: 67-71.

Feo, T.A. and M.G.C. Resende, 1995. Greedy randomized adaptive search procedures. J. Global Optim., 6: 109-133.

Fix, E. and J.L. Hodges, 1952. Discriminatory analysis-nonparametric discrimination: Small sample performance. http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA800391

Fix, E. and J.L. Hodges, 1989. Discriminatory analysis: Nonparametric discrimination: Consistency properties. Int. Stat. Rev./Revue Int. Stat., 57: 238-247.

Golden, B.L., Q. Wang and L. Liu, 1988. A multifaceted heuristic for the orienteering problem. Naval Res. Logistics, 35: 359-366.

Hart, J.P. and A.W. Shogan, 1987. Semi-greedy heuristics: An empirical study. Oper. Res. Lett., 6: 107-114.

Ke, L., C. Archetti and Z. Feng, 2008. Ants can solve the team orienteering problem. Comput. Ind. Eng., 54: 648-665.

Maquera, G., M. Laguna, D.A. Gandelman and A.P. Sant'Anna, 2011. Scatter search applied to the vehicle routing problem with simultaneous delivery and pickup. Int. J. Applied Metahe. Comput., 2: 1-20.

Muthuswamy, S. and S.S. Lam, 2011. Discrete particle swarm optimization for the team orienteering problem. Memetic Comput., 3: 287-303.

Souffriau, W., P. Vansteenwegen, G.V. Berghe and D. Van Oudheusden, 2010. A path relinking approach for the team orienteering problem. Comput. Oper. Res., 37: 1853-1859.

Talbi, E.G., 2009. Metaheuristics: From Design to Implementation. John Wiley and Sons, Hoboken, New Jersey.

Vansteenwegen, P., W. Souffriau, G.V. Berghe and D.V. Oudheusden, 2009. A guided local search metaheuristic for the team orienteering problem. Eur. J. Oper. Res., 196: 118-127.

Vansteenwegen, P., W. Souffriau, G.V. Berghe and D.V. Oudheusden, 2009. Metaheuristics for tourist trip planning. Metaheuristics Serv. Ind., 624: 15-31.