An adaptive heuristic for the Capacitated Team Orienteering Problem

Asma Ben-Said, Racha El-Hajj, Aziz Moukrim

An adaptive heuristic for the Capacitated Team Orienteering Problem

Asma Ben-Said [*] **Racha El-Hajj** [*,**] **Aziz Moukrim** [*]

[*] *Sorbonne universités, Université de technologie de Compiègne CNRS, Heudiasyc UMR 7253, CS 60 319, 60 203 Compiègne cedex*
[**] *Université Libanaise, Ecole Doctorale des Sciences et de Technologie, Campus Hadath, Beyrouth, Liban*
e-mail: {asma.ben-said,racha.el-hajj,aziz.moukrim}@hds.utc.fr

Abstract: The Capacitated Team Orienteering Problem (CTOP) is a new variant of the well-known Team Orienteering Problem (TOP) where an additional constraint is imposed on the vehicles capacities. By associating a profit and a demand to each customer, the objective of solving CTOP is to select the set of customers to be served in such a way that the total amount of profits collected from the visited customers is maximized, while respecting all the resource limitations, i.e., maximum length limit and maximum capacity of each vehicle. We present in this paper a new adaptive heuristic to solve CTOP. Our method is based on an adaptive iterative destructive constructive heuristic, which adjusts its parameters according to the solution progress. Computational experiments applied on the benchmark of CTOP show the effectiveness of our proposed method, which provided some results of high quality with a competitive computational time. Moreover, an improvement was found in the score of one of the hardest instances of the benchmark.

Keywords: Capacitated Team Orienteering Problem, iterative local search, adaptive destruction/construction heuristic

1. INTRODUCTION

The Capacitated Team Orienteering Problem (CTOP) is a variant of the Vehicle Routing Problems (VRP) where not all customers can be visited due to some resource limitations. In this problem, a profit is associated with each customer, which is collected when the customer is served. A demand is also associated with customers and is provided from the vehicle's capacity. Each vehicle must start its route from a specified point called the departure depot and returns to the arrival one while not exceeding its maximum travel time limit. In addition, each customer must be served at most once by the fleet of vehicles only if the overall provided demands respects the maximum capacity of the vehicle. The objective of solving CTOP is to maximize the total collected profits from the visited customers while respecting all the resource limitations.

CTOP arises in many real life applications. It mainly appears in the truck-load transportation industry, where shippers often look to outsource the transportation of goods to carriers. In some cases, due to the increasing demands of the customers and the limited resources, i.e. number of available vehicles, capacity of each vehicle, the carriers must choose the most convenient customers to serve in order to maximize the collected profit. Therefore, it is recommended to model these problems with CTOP to choose the best itinerary of visits for the available vehicles.

CTOP was first introduced by (Archetti et al., 2009). Despite its importance in many real life applications, few researchers had focused on providing solution techniques for this problem. To the best of our knowledge, only two exact methods have been developed to solve CTOP. The first method was based on the branch-and-price algorithm and was proposed by Archetti et al. (2009). The authors adapted the branch-and-price algorithm proposed by Boussier et al. (2007) for TOP, by taking into consideration the capacity constraints while solving the subproblems. Recently, Archetti et al. (2013) proposed a new exact algorithm based on the branch-and-price technique, where a restricted heuristic is used to provide primal bound values for each node of the enumeration tree. Beside the exact algorithms, some heuristic approaches were developed to provide solutions for CTOP in a short computational time. Three heuristic methods were proposed by Archetti et al. (2009). First, a tabu search algorithm, which just explores feasible solutions was presented as the Tabu Search TSF, then another tabu search algorithm called TSA, which considers feasible and admissible solutions was developed. Finally, a variable neighborhood search (VNS) that iteratively uses the two heuristics TSF and TSA was proposed. Tarantilis et al. (2013) then proposed a Bi-level search method, which seems to be more efficient for CTOP. Many neighborhood solutions are explored at each iteration by applying a local search technique randomly chosen between three local searches that focus on the replacement movement and the insertion. These local searches are followed by an adjustment procedure used to reduce the traveled distance in the best solution found. Moreover, Luo et al. (2013) had recently proposed a new heuristic method named Adaptive Ejection Pool with Toggle-rule diversification ($ADEPT$). Their approach uses a list of unrouted customers sorted according to a certain priority rule chosen between two: The first rule uses a decreasing order of a valuation associated to each customer and the second one uses the first-in-first-out policy. At each iteration, the highest priority customer is randomly inserted in the current solution. Then, a local search is applied to improve the solution, where the feasibility of the obtained solution is restored by removing minimum-valuation customers. So far, $ADEPT$ seems to be the best method for CTOP since it is able to outperform all the existing methods in the literature.

In this paper we propose a new Heuristic method to solve CTOP. Our algorithm is mainly based on an Adaptive Iterative Destruction/Construction Heuristic ($AIDCH$), which is composed of an adaptive construction phase based on a Best Insertion algorithm, followed by an adaptive diversification phase with some local search techniques. Our algorithm is initialized with several parameters and then adjusts them by itself according to the solution progress throughout the resolution process. The remainder of the paper is organized as follows. Section 2 provides a formal formulation of CTOP, then our proposed method is described in Section 3. Computational experiments performed on the standard benchmark of CTOP are reported in Section 4 followed by some conclusions and futur works in Section 5.

2. PROBLEM DESCRIPTION AND NOTATION

CTOP is modeled with a complete undirected graph $G = (V \cup \{0\}, E)$, where $V = \{1, \ldots, n\}$ is the set of vertices representing customers with vertex 0 the depot and $E = \{(i, j) : i, j \in V \cup \{0\}\}$ the set of edges interconnecting the vertices. A profit p_i and a demand d_i are associated with each customer $i \in V$. Each edge $(i, j) \in E$ is valued with a cost l_{ij} which is assumed to be symmetric and verifying the triangle inequality. A fleet F of m homogeneous vehicles with a maximum capacity Q is available to serve customers, where each vehicle cannot travel more than L_{max} time units. We define a route r_k as a sequence of q_{r_k} customers assigned to vehicle k in the fleet F. Each route starts and ends with the depot vertex 0: $r_k = (0, r_k[1], \ldots r_k[q_{r_k}], 0), \forall k \in F$. The total demand of all customers served in route r_k, denoted by $D(r_k)$, cannot exceed the vehicle capacity Q and the total length of route r_k, denoted by $L(r_k)$, must respect the travel time limit L_{max}.

A solution S of CTOP represents a set of m routes in which each customer is served at most once by the fleet of vehicles. We denote the total profit of a solution S by $P(S) = \sum_{i \in S} p_i$ and the total demand provided to the served customers by $D(S) = \sum_{i \in S} d_i$. The objective of solving CTOP is to find a solution S such that the total profit collected in all its routes is maximized.

3. ADAPTIVE ITERATIVE DESTRUCTION/CONSTRUCTION HEURISTIC

In order to provide feasible solutions for CTOP with good quality, we propose an adaptive iterative destruction/construction heuristic. The global scheme of our proposed approach is described in Algorithm 1.

Starting with an empty solution, we use an adaptive construction procedure to build an initial solution. Then, at each iteration, a part of the solution is destroyed by removing from the routes a limited random number of customers, bounded by dmax. Therefore, we define $dmax$ as the degree of diversification. This value is initialized to 3 and then incremented after each non-improving iteration. As soon as an improvement is found, we reset $dmax$ to 3 in order to entirely explore the neighborhood of the new solution. A 2-opt operator is next applied to the solution to reduce the length of the routes. Finally, we perform an adaptive construction procedure to complete the solution. This process is reiterated and stops when all customers are served or when a certain number of consecutive iterations have failed to improve the quality of the solution. The final result is the best solution found over all iterations. Based on the previous work of Dang et al. (2013) for the vehicle routing problem with profits, the maximum number of consecutive iterations without improving the best result is set to n^2.

Algorithm 1 General structure of AIDCH

```
Input     : An instance of CTOP
Output    : S best solution found
Variables:  dmax : diversification degree
            S_i : current solution
            UR : unrouted customers
dmax := 3;
iter_max := n^2;
AdaptiveConstruction(S);
S_i := S;
while (iter < iter_max) and (UR not empty) do
    d := rand(1, dmax);
    AdaptiveDestruction(S_i, d);
    Apply2-Opt(S_i);
    AdaptiveConstruction(S_i);
    iter++;
    if (P(S_i) > P(S))
    or (P(S_i) = P(S) and D(S_i) ≤ D(S)) then
        dmax := 3;
        if (P(S_i) > P(S))
        or (P(S_i) = P(S) and D(S_i) < D(S)) then
            S := S_i;
            iter := 0;
        end
    end
    else
        dmax = min(dmax+1, n);
    end
end
```

Our algorithm makes use of an adaptive diversification mechanism to avoid its stuck in a local optimum. The idea is to explore the neighborhood of the new solution as soon as an improvement is found and to explore more distant zones whenever the search is trapped in a local optimum.

The main component of our proposed algorithm is the adaptive construction heuristic based on a Best insertion algorithm (BIA). This algorithm considers a partial solution S, which might be an empty solution, and tries to insert unrouted customers one by one in the solution. The objective of BIA is to evaluate, at each iteration, all feasible insertions that respect the length and the capacity constraints of all the routes in the resulting solution. The best insertion is then performed. This process is iterated until either all customers are served or no further feasible insertions are available.

To evaluate the insertion of customer c between vertices i and i^+ in route r, many experiments pointed out the insertion criterion denoted by CI_{ci}.
Let P_{max} be the profit of the most profitable customer, and $\Delta L_{ci} = l_{i,c} + l_{c,i^+} - l_{i,i^+}$, the difference in length caused by inserting customer c between i and i^+.

Algorithm 2 Best Insertion Algorithm

```
Input     : S_0: partial solution
            UR: unrouted customers
            (α, β, γ): criterion parameters
Output    : Solution S
Variables : FI : list of feasible insertions
            insert : boolean (one customer inserted)
S := S_0;
insert := true;
while (UR not empty) and (insert = true) do
    FI := ∅;
    insert := false;
    foreach customer c do
        foreach Tour r do
            if D(r) + d_c ≤ Q then
                foreach couple of consecutive customers (i, i+) do
                    calculate ΔL_ci;
                    if L(r) + ΔL_ci ≤ L_max then
                        calculate CI_ci;
                        FI := FI ∪ {(c, r, (i, i+))};
                    end
                end
            end
        end
    end
    if FI not empty then
        (c_best, r_best, (i, i+)_best) := best insertion from FI;
        S := S ⊕ (c_best, r_best, (i, i+)_best);
        UR := UR \ {c_best};
        insert := true;
    end
end
```

The insertion criterion is then calculated as follows:

$$CI_{ci} = \frac{(\frac{1+\Delta L_{ci}}{1+L_{max}})^{\beta} * (\frac{d_c}{Q})^{\gamma}}{(\frac{p_c}{P_{max}})^{\alpha}} \qquad (1)$$

Our proposed criterion aims to maximize the profit, and minimize the total traveled distance and the used load. We note that we added a unit to the length factor in order to equally study the insertion criteria, even if the inserted customer c is aligned with the two customers i and i^+ between which it should be inserted or coincide with one of them. A normalized value is considered for each factor in order to evaluate them in the same manner. Beside that, each factor is weighted with a certain parameter, denoted by α, β and γ as shown in Expression (1) in order to control their relative importance.

Algorithm 2 illustrates the Best Insertion Procedure, where we use the notation $(c, r, (i, i^+))$ to denote the insertion of customer c between customers i and i^+ in route r. The insertion operation is denoted by $\oplus$.

At each iteration i of the *AIDCH*, the proposed construction procedure operates as follows. Five constructive heuristics $CM_{j,j\in\{1,...,5\}}$ launch separately BIA with different triplets (α, β, γ) on the current solution. During each launch, α, β and γ are chosen randomly in the cube having the center $(\alpha_{i-1}, \beta_{i-1}, \gamma_{i-1})$ and the side length φ, where α_{i-1}, β_{i-1} and γ_{i-1} represent the best parameters obtained by the method at iteration $i - 1$. At the end of each method applied $CM_{j,j\in\{1,...,5\}}$, the parameters leading to the best solution found by CM_j are saved to be used to calculate the BIA parameters of the next iteration.

Finally, the best solution obtained among the five methods is retained as the current solution. The purpose of this procedure is to perform parallel independent searches in the solutions space and choose the best track to follow in order to converge faster toward a good solution.

4. COMPUTATIONAL EXPERIMENTS

To evaluate our *AIDCH*, we used the standard benchmark of CTOP generated by Archetti et al. (2009). Our algorithm is coded in C++ using the Standard Template Library (STL) for data structures, where we used the GNU GCC to compile our program. All tests are carried out on a linux server with an Intel Xeon X7542 CPU clocked at 2.66 GHz.

In order to evaluate the proposed methods, Archetti et al. (2009) proposed a new benchmark of CTOP composed of 130 instances which are classified into 13 sets according to the number of vehicles available, to the vehicle capacity and to the maximum length.

In what follows, we analyze the impact of various components in our *AIDCH* and we provide some experiments to tune the needed parameters.

We use the best known score in the literature, denoted by BK, which represents the best score obtained by the heuristic and the exact methods proposed in the literature for CTOP. We tested our approach with the same protocol used in Dang et al. (2013) and Ke et al. (2008). For each instance, our algorithm is executed 10 times from which we recorded the relative percentage error (RPE), which is defined as the relative error between the best score obtained by the heuristic and the exact methods proposed in the literature for CTOP (BK) and the maximal score obtained among the 10 runs.

4.1 Impact of the adaptive construction mechanism

In order to evaluate the performance of our adaptive mechanism, we carried out several experiments to choose the best value of the step φ and to compare the behavior of our algorithm with and without the adaptive mechanism.

The value of the step φ affects significantly the resulting performance of our method. Various experiments with different values of φ are tested. For these tests, we performed a fast version of our algorithm where $iter_{max}$ is set to n

instead of n^2. Figure 1 shows the evolution of RPE in terms of φ.

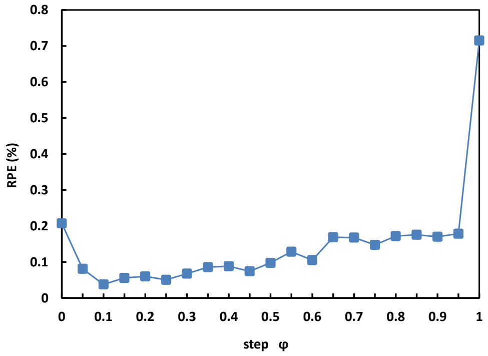

Fig. 1. Performance of $AIDCH$ in terms of φ

Based on the graphic of Figure 1, we notice that our algorithm provides the best RPE with $\varphi = 0.1$. Hence, we have chosen the parameter φ to be equal to 0.1 in our next experiments.

On the other hand, we examined the ability of the adaptive construction mechanism to guide the search toward finding, at each time, the best trade-off between the profit, the distance and the demand. Therefore, we test two versions of our algorithm, the first one by considering the overall approach and the second one where the adaptive mechanism is disabled and the parameters (β, γ) are chosen randomly in [0,1] at each iteration. In these experiments, for each instance, the algorithm is launched only once and the best solution for the first 15000 iterations is recorded. The RPE values obtained in each iteration are reported in Figure 2.

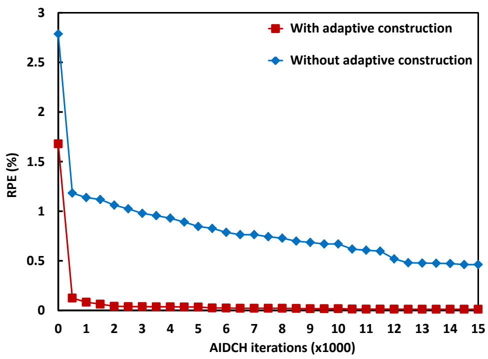

Fig. 2. Impact of the adaptive construction mechanism

These experiments show that with the use of the adaptive construction mechanism, the search procedure converges quickly toward good solutions. In contrast, the convergence to the best solution is very slow in the case treated without adaptive construction mechanism. Therefore, based on the obtained results, it is recommended to fix parameter α to 1, and to vary β and γ in [0, 1].

4.2 Impact of the diversification mechanism

To evaluate the effectiveness of our adaptive perturbation component, we implemented an alternative version of $AIDCH$ where the perturbation procedure is used as in the approach of Dang et al. (2013), which was developed to solve the vehicle routing problems with profits. In their studies, they limited the maximum number of removed customers $dmax$ to 3. Figure 3 illustrates the average RPE recorded against the number of iterations to show the evolution of the two versions: with and without the adaptive diversification mechanism.

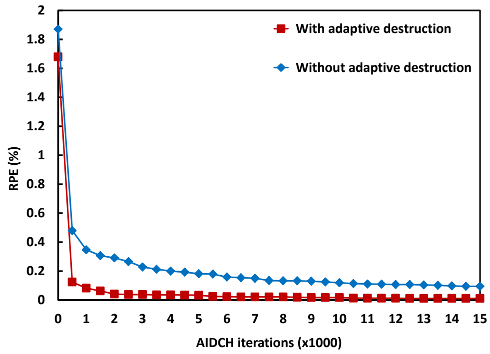

Fig. 3. Impact of the adaptive diversification mechanism

Based on these two graphs, we can easily notice that the average RPE with the adaptive perturbation component is always below the average RPE with the standard perturbation of Dang et al. (2013) at each iteration, which shows the effectiveness of our proposed technique.

4.3 Comparison with the literature

We compared the results of our $AIDCH$ on the benchmark of CTOP with the state-of-the-art heuristic methods available in the literature:

- TSF, TSA and VNS proposed by Archetti et al. (2009), tested on a Pentium 4 2.80 GHz CPU.
- BiF proposed by Tarantilis et al. (2013), tested on an Intel Core2 Quad 2.83 GHz CPU.
- $ADEPT$ proposed by Luo et al. (2013), tested on an Intel Xeon E5430 2.66 GHz CPU.

To compare the cpu times between different machines, it is recommended to use the Super PI protocol, which estimates the CPU speed. By performing these tests to the four machines, we noticed that our machine and those used by Tarantilis et al. (2013) and Luo et al. (2013) have almost similar computational power and are about three times faster than the machine used by Archetti et al. (2009).

The performance of all methods is evaluated according to the quality of the solutions and the computational times. Table 1 reports the averages RPE obtained for the 13 sets of the benchmark, while the average TTB is summarized in Table 2. The dash mark '-' indicates that the algorithm is not tested on the corresponding set of instances. In the line *best*, we count the number of best solutions found, while in the bottom of each table we report the average RPE and TTB on all CTOP instances.

The results obtained in these tables show that our proposed algorithm was able to find the best solutions for 127 instances of the benchmark of CTOP with a much lower computational time compared to TSF, TSA and VNS methods which have solved, respectively, 87, 69 and 101 instances. Moreover, these results show that $AIDCH$ and BiF methods are very competitive in terms of computation time. In contrast, we obtained best solutions for

Set	TSF	TSA	VNS	BiF	$ADEPT$	$AIDCH$
g1	0.029	0.018	0.005	0	-	-0.003
g2	0	0.075	0	0.081	0	0.075
g3	0	0.05	0	0	0	0
g4	0	0.079	0	0	0	0
g5	0	0	0.047	0	0	0
g6	0.096	0.096	0.096	0	0	0
g7	0.102	0.299	0.196	0	0	0
g8	0.107	0.253	0.035	0	0	0
g9	0.143	0.540	0.240	0.049	0	0
g10	0.348	0.739	0.332	0	0	0.037
g11	0.266	0.212	0.018	0	0	0
g12	0.469	0.278	0.049	0	0	0
g13	0.189	0.226	0	0	0	0
best	87	69	101	126	120	127
average	0.135	0.220	0.078	0.010	0	0.008

Table 1. Comparison of the average RPE between our AIDCH and the other heuristic methods in the literature

Set	TSF	TSA	VNS	BiF	$ADEPT$	$AIDCH$
g1	43.3	50.4	720.0	0.2	-	121.2
g2	98.7	150.9	138.4	0.2	3.5	0.0
g3	97.9	168.8	182.4	0.2	9.4	0.1
g4	97.4	163.6	243.7	1.0	11.6	0.1
g5	594.1	723.3	892.9	5.7	13.2	0.1
g6	630.2	692.0	1151.2	6.0	33.1	0.4
g7	740.2	426.1	1325.0	29.3	39.4	4.7
g8	706.7	765.1	1147.1	13.7	35.7	0.6
g9	848.6	747.4	1353.1	54.3	125.9	9.7
g10	738.9	999.3	1621.4	65.3	98.5	13.3
g11	382.1	275.1	714.7	1.3	2.0	1.9
g12	381.7	264.2	1244.2	8.2	10.2	35.0
g13	400.1	241.7	1640.8	1.3	15.4	14.0
average	443.1	436.0	951.9	14.4	33.2	15.5

Table 2. Comparison of the average RPE between our AIDCH and the other heuristic methods in the literature

127 instances while BiF obtained 126 best solutions. In addition, $ADEPT$ was not tested on the instances of the first set of instances. In their survey Luo et al. (2013), the authors didn't give the result of their $ADEPT$ algorithm in the first set of instances. By comparing the average computational time of the common treated sets i.e. from g2 to g13, we found that our $AIDCH$ consumed 3.3 *sec* which is much smaller than the TTB of $ADEPT$, which reached 33.2 *sec*. We note that our $AIDCH$ makes a greater computational time in the first set which contains large scale instances. This result pointed out the ability of our method to deal with difficult instances.

On the other hand, our $AIDCH$ was able to improve one new best solution for an instance from the first set with $m = 15$, $Q = 200$ and $L_{max} = 200$. It is worth to mention that Tarantilis et al. (2013) showed that their BiF algorithm was able to serve all customers in this instance which is impossible in terms of capacity constraint with the use of only 15 vehicles, where at least 16 vehicles are needed to cover the demand of all customers.

5. CONCLUSION AND FUTURE WORK

We presented in this paper an adaptive iterative destruction construction heuristic for the Capacitated Team Orienteering Problem. Two main adaptive mechanisms highlight our approach and had proven to be very efficient for the case of CTOP. The adaptive construction procedure based on the best insertion algorithm manages the insertion of customers in the solution according to the progress of our algorithm, while the adaptive perturbation mechanism controls the number of removed customers according to the search efficiency and to the status of the solution obtained so far.

Computational experiments performed on the benchmark of CTOP show the effectiveness of our algorithm compared to the other heuristic methods performed for CTOP. Our $AIDCH$ reached a relative error of 0.008% and improved the quality of one of the hardest instances with a small computational time.

As for future work, we will perform several ameliorations for our $AIDCH$ to improve its performance in terms of RPE and computational time. By applying some modifications to the problem, we will be able to solve other variants of CTOP in order to respond to new needs, as CTOP with time windows and/or resource synchronization.

ACKNOWLEDGEMENTS

This work is carried out in the framework of the Labex MS2T, which was funded by the French Government, through the program "Investments for the future" managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02). It is also partially supported by the TCDU project (Collaborative Transportation in Urban Distribution, ANR-14-CE22-0017).

REFERENCES

Archetti, C., Bianchessi, N., and Speranza, M.G. (2013). Optimal solutions for routing problems with profits. *Discrete Applied Mathematics*, 161(4), 547–557.

Archetti, C., Feillet, D., Hertz, A., and Speranza, M.G. (2009). The capacitated team orienteering and profitable tour problems. *Journal of the Operational Research Society*, 60(6), 831–842.

Boussier, S., Feillet, D., and Gendreau, M. (2007). An exact algorithm for team orienteering problems. *4or*, 5(3), 211–230.

Dang, D.C., Guibadj, R.N., and Moukrim, A. (2013). An effective pso-inspired algorithm for the team orienteering problem. *European Journal of Operational Research*, 229(2), 332–344.

Ke, L., Archetti, C., and Feng, Z. (2008). Ants can solve the team orienteering problem. *Computers & Industrial Engineering*, 54(3), 648–665.

Luo, Z., Cheang, B., Lim, A., and Zhu, W. (2013). An adaptive ejection pool with toggle-rule diversification approach for the capacitated team orienteering problem. *European Journal of Operational Research*, 229(3), 673–682.

Tarantilis, C.D., Stavropoulou, F., and Repoussis, P.P. (2013). The capacitated team orienteering problem: a bi-level filter-and-fan method. *European Journal of Operational Research*, 224(1), 65–78.