A Branch-and-Cut Algorithm for Solving the Team Orienteering Problem

Duc-Cuong Dang, Racha El-Hajj, Aziz Moukrim

A Branch-and-Cut Algorithm for Solving the Team Orienteering Problem

Duc-Cuong Dang[1,3], Racha El-Hajj[1,2,⋆], and Aziz Moukrim[1]

[1] Université de Technologie de Compiègne, Département Génie Informatique
Laboratoire Heudiasyc, UMR 7253 CNRS, 60205 Compiègne, France
[2] Université Libanaise, Faculté de Génie, Département Contrôle Industriel
Campus Hadath, Beyrouth, Liban
[3] School of Computer Science, University of Nottingham
Jubilee Campus, Wollaton Road, Nottingham NG8 1BB, United Kingdom
{duc-cuong.dang, racha.el-hajj, aziz.moukrim}@hds.utc.fr

Abstract. This paper describes a branch-and-cut algorithm to solve the Team Orienteering Problem (TOP). TOP is a variant of the Vehicle Routing Problem (VRP) in which the aim is to maximize the total amount of collected profits from visiting customers while not exceeding the predefined travel time limit of each vehicle. In contrast to the exact solving methods in the literature, our algorithm is based on a linear formulation with a polynomial number of binary variables. The algorithm features a new set of useful dominance properties and valid inequalities. The set includes symmetric breaking inequalities, boundaries on profits, generalized subtour eliminations and clique cuts from graphs of incompatibilities. Experiments conducted on the standard benchmark for TOP clearly show that our branch-and-cut is competitive with the other methods in the literature and allows us to close 29 open instances.

Keywords: branch-and-cut, dominance property, incompatibility, clique cut.

Introduction

The Team Orienteering Problem (TOP) [4] is a widely studied Vehicle Routing Problem (VRP) which can be described as follows: a fleet of vehicles is available to visit customers from a potential set and each vehicle is associated with a predefined travel time limit and two particular depots, the so-called departure and arrival. Each customer is associated with an amount of profit that can be collected at most once by the fleet of vehicles. The aim of TOP is to select customers and organize an itinerary of visits so as to maximize the total amount of collected profits. The applications of TOP include athlete recruiting [4], technician routing [1, 9] and tourist trip planning [10].

⋆ Corresponding author.
This work is partially supported by the Regional Council of Picardie and the European Regional Development Fund (ERDF), under PRIMA project.

To the best of our knowledge, three exact methods have been proposed to solve TOP. Butt and Ryan [3] described a set covering formulation and developed a column generation algorithm for solving TOP. In Boussier et al. [2], the authors proposed a branch-and-price algorithm and a dynamic programming approach to deal with the pricing problem. More recently, a pseudo-polynomial linear model for TOP was introduced by Poggi de Aragão et al. [8] and a branch-cut and price algorithm was proposed. These methods are able to solve a large part of the standard benchmark [4] for TOP, however many other instances remain open. Furthermore, a recent effort [6] showed that it is hardly possible to improve the already-known solutions for TOP by heuristics.

In this paper, we propose a branch-and-cut algorithm for the exact solution of TOP based on a set of dominance properties and valid inequalities. This set includes symmetric breaking, generalized subtour eliminations, boundaries on profits/numbers of customers based on dynamic programming, as well as clique cuts based on the graphs of incompatibilities. Experiments conducted on the standard benchmark [4] for TOP clearly show the competitiveness of the approach, especially on the number of instances being solved to optimality. The algorithm also allows us to close 29 open instances.

1 Compact formulation

TOP is modeled with a complete graph $G = (V, E)$. $V = \{1, \ldots, n\} \cup \{d, a\}$ is the set of vertices representing customers and depots. $E = \{(i, j) | i, j \in V\}$ is the set of arcs. Vertices d and a are respectively the departure and the arrival depot for the vehicles. We use V^-, V^d and V^a to denote the sets of customers only, customers with departure depot and customers with arrival depot respectively. Each vertex i is associated with a profit P_i ($P_d = P_a = 0$) and a travel cost C_{ij} is associated with each arc $(i, j) \in E$ ($C_{id} = C_{ai} = \infty, \forall i \in V^-$). The travel costs are assumed to satisfy the triangle inequality. A fleet F is composed of m identical vehicles and available to visit customers without exceeding a travel cost limit L for each vehicle. The problem can be then formulated in Mixed Integer Programming (MIP) using a polynomial number of decision variables x_{ijr} and y_{ir}: $x_{ijr} = 1$ if arc (i, j) is used by vehicle r to serve customer i then customer j and 0 otherwise; $y_{ir} = 1$ if client i is served by vehicle r and 0 otherwise.

$$\max \sum_{i \in V^-} \sum_{r \in F} y_{ir} P_i \tag{1}$$

$$\sum_{r \in F} y_{ir} \leq 1 \qquad \forall i \in V^- \tag{2}$$

$$\sum_{j \in V^a} x_{djr} = \sum_{j \in V^d} x_{jar} = 1 \qquad \forall r \in F \tag{3}$$

$$\sum_{i \in V^a \setminus \{k\}} x_{kir} = \sum_{j \in V^d \setminus \{k\}} x_{jkr} = y_{kr} \; \forall k \in V^-, \forall r \in F \tag{4}$$

$$\sum_{i \in V^d} \sum_{j \in V^a \setminus \{i\}} C_{ij} x_{ijr} \leq L \qquad \forall r \in F \tag{5}$$

$$\sum_{(i,j)\in U\times U} x_{ijr} \leq |U| - 1 \quad \forall U \subseteq V^-, |U| \geq 2, \forall r \in F \tag{6}$$

$$x_{ijr} \in \{0,1\} \quad \forall i \in V, \forall j \in V, \forall r \in F \tag{7}$$
$$y_{ir} \in \{0,1\} \quad \forall i \in V^-, \forall r \in F$$

The objective function (1) is to maximize the sum of collected profits. Constraints (2) guarantee that each customer is visited at most once. The connectivity of each tour is ensured by constraints (3) and (4). Constraints (5) describe the travel length restriction. Constraints (6) ensure that subtours are forbidden. Finally, the integral requirement on variables is guaranteed by constraints (7).

2 Generalized subtour eliminations

The number of constraints (6) in the formulation is exponential. In practice, these constraints are removed from the formulation and only added when needed. Moreover, we replace these constraints with stronger ones known as the Generalized Subtour Elimination Constraints (GSEC) [7] as follows.

For a given subset S of vertices, we define $\delta(S)$ as the set of arcs connecting vertices of S with vertices of V\S. We also define $\gamma(S)$ as the set of arcs interconnecting vertices of S. The following constraints ensure that each customer served by vehicle r is connected to the depots.

Property 2.1 *GSEC:*

$$\sum_{(u,v)\in\delta(S)} x_{uvr} \geq 2y_{ir}, \forall S \subset V, \{d,a\} \subseteq S, \forall i \in V \setminus S, \forall r \in F \tag{8}$$

Property 2.2 *Equivalent to the GSEC:*

$$\sum_{(u,v)\in\gamma(S)} x_{uvr} \leq \sum_{i\in S} y_{ir} - y_{jr}, \forall S \subset V, \{d,a\} \subseteq S, \forall j \in V \setminus S, \forall r \in F \tag{9}$$

$$\sum_{(u,v)\in\gamma(U)} x_{uvr} \leq \sum_{i\in U} y_{ir} - y_{jr}, \forall U \subseteq V^-, \forall j \in U, \forall r \in F \tag{10}$$

3 Dominance properties

Given an instance X of TOP with m vehicles, we use X_I to denote the same instance for which profit of each customer is changed to 1. We also define X^g as the instance X where the number of vehicles is reduced to g ($g \leq m$). On the other hand, we use $LB(X)$ (resp. $UB(X)$) to denote a lower (resp. an upper) bound of instance X. The following properties hold for the formulation of TOP.

Property 3.1 *Symmetric breaking on profits, (without loss of generality) we focus on solutions in which profits of routes are sorted in a particular order:*

$$\sum_{i\in V^-} y_{i(r+1)} P_i - \sum_{i\in V^-} y_{ir} P_i \leq 0, \forall r \in F \setminus \{m\} \tag{11}$$

Property 3.2 *Boundaries on profits:*

$$\sum_{r\in H}\sum_{i\in V^-} y_{ir}P_i \leq UB(X^{|H|}), \forall H \subset F \tag{12}$$

$$\sum_{r\in H}\sum_{i\in V^-} y_{ir}P_i + UB(X^{m-|H|}) \geq LB(X), \forall H \subseteq F \tag{13}$$

Property 3.3 *Boundaries on numbers of customers:*

$$\sum_{r\in H}\sum_{i\in V^-} y_{ir} \leq UB(X_I^{|H|}), \forall H \subset F \tag{14}$$

$$\sum_{i\in V^-} y_{ir} \geq LB(\bar{X}_I^1),\ \forall r \in F \tag{15}$$

Values of LB in (13) are calculated using an efficient heuristic for TOP, such as the one used in [6]. Similarly to dynamic programming, values of UB in (12), (13) and (14) are computed as follows. We start our resolution by computing these values for the smallest instance ($|H| = 1$) using a stopping condition, i.e. computational time or number of branch-and-bound nodes, then we use the obtained values to solve larger instances ($|H| \leq m$). In (15), we use $LB(\bar{X}_I^1)$ to denote a lower bound obtained from solving the derived MIP model of X_I^1 for which the objective function is reversed to minimization and the constraints (12) and (13) with $|H| = 1$ are added.

4 Incompatibilities and clique cuts

Let S be a small subset of vertices of V^- (or arcs of E), we use $MinLength(S)$ to denote the length of the shortest path from d to a containing all vertices (or all arcs) of S. The graph of incompatibilities between customers is defined as: $G_{V^-}^{Inc} = (V^-, E_{V^-}^{Inc})$ with $E_{V^-}^{Inc} = \{[i,j] | i \in V^-, j \in V^-, MinLength(\{i,j\}) > L\}$. The graph of incompatibilities between arcs is defined as: $G_E^{Inc} = (E, E_E^{Inc})$ with $E_E^{Inc} = \{[i,j] | i = (u,v) \in E, j = (w,s) \in E, MinLength(\{(u,v),(w,s)\}) > L\}$. The following inequalities hold for the formulation of TOP.

Property 4.1 *(Clique) Let K (resp. Q) be a clique of $G_{V^-}^{Inc}$ (resp. G_E^{Inc}):*

$$\sum_{i\in K} y_{ir} \leq 1, \forall r \in F \tag{16}$$

$$\sum_{(u,v)\in Q} x_{uvr} \leq 1, \forall r \in F \tag{17}$$

The two graphs $G_{V^-}^{Inc}$ and G_E^{Inc} can be computed beforehand and archived for each instance. Maximal cliques are preferred in inequalities (16) and (17) since they provide tighter formulation. In practice a greedy decomposition of the two graphs into maximal cliques is used and the details are given in the next section.

5 Branch-and-cut algorithm

Branching rule: Since the objective function is to maximize the collected profits, the selection of correct customers appears to be crucial for TOP. Therefore, our branching rule is prioritized on y_{ir} first then x_{ijr} [2, 8].

Presolving steps: By definition, a customer is said to be *inaccessible* if the travel cost of the tour containing only that customer excesses the cost limit. An *inaccessible* arc can be similarly defined. So in order to make a proper linear formulation, we first eliminate all *inaccessible* customers and arcs. Additionally, during a limited computational time at the beginning (e.g. limited to 5% of the total solving time), the values required for inequalities (12)-(15) are computed using the method mentioned in Section 3. Then a greedy decomposition of $G^{Inc}_{V^-}$ into maximal cliques is generated using [5] and the associated inequalities (16) are added to the formulation.

Complete algorithm: The MIP solver is initialized with a feasible solution generated from an heuristic of [6]. This initialization accelerates the resolution by eliminating portions of the search space composed of solutions with lower profits. In the first iteration, the linear model containing constraints (2)-(5), (7), (11)-(15) and (16) is solved and a solution is obtained. The solution is then checked for subtours. If the solution does not contain any subtour, then it is optimal and the resolution is terminated. Otherwise, the associated constraints (8), (9) and (10) are added into the linear model. Additionally, based on the sets of vertices and arcs from the solution, we extract the associated subgraphs from $G^{Inc}_{V^-}$ and G^{Inc}_{E}, then generate their greedy clique decompositions in order to add the corresponding constraints (16) and (17) to the linear model. In the next iteration, the same solving process is repeated with the new model.

6 Numerical results

Our approach was tested on the benchmark of TOP instances proposed by Chao et al. [4]. It comprises 387 instances divided into 7 sets. The numbers of customers and vehicles are up to 100 and 4 respectively. Our algorithm is coded in C++. Experiments were conducted on an AMD Opteron 2.60 GHz and CPLEX 12.4 was used as MIP solver. We used the same 2h limit of solving time as in [2, 8].

In order to evaluate the usefulness of the proposed components, we activated them one by one so that the complete algorithm is obtained in the last activation. Table 1 shows the number of instances being solved to optimality for each activation. The average computational times on the subset of instances being solved by all configurations are also given. We notice that each component contributes to the improvement of the number of instances being solved, as well as to the reduction of the computational times. Table 2 shows a small comparison between our results and those of Boussier et al. [2]. Each instance is chosen so that only one of the two methods is able to prove the optimality. Columns *instance*,

n, m, and L indicate respectively name of the instance, number of customers, number of vehicles and the cost limit. The main columns report the results of the branch-and-price method of Boussier et al. [2] in B-P and our results in B-C. Columns LB, UB and CPU report respectively the lower bound, upper bound and solving time in seconds for each instance and method. To summarize, our algorithm is able to prove the optimality of all the instances in the sets 1, 2, 3, and 6 and most of the instances in the other three sets. In total, we are able to solve 278 instances. Compared to [2] which solved 270 instances, our approach is clearly competitive. Moreover, it allows us to close 29 open instances.

Table 1: Performance of our branch-and-cut.

Accumulated components	Solved instances	CPU (in seconds)
Standard model	177/387	198.9
+ Generalized subtour eliminations	233/387	61.75
+ Symmetric breaking	243/387	14.32
+ Boundaries on profit/customers	265/387	13.16
+ Clique cuts	278/387	3.24

Table 2: Comparison between the branch-and-price [2] and ours.

Instance	n	m	L	B-P [2]			B-C		
				UB	LB	CPU	UB	LB	CPU
$p1.2.p$	30	2	37.5	250	–	2926	250	250	8.85
$p1.2.q$	30	2	40	–	–	–	265	265	10.96
$p1.2.r$	30	2	42.5	–	–	–	280	280	10.15
$p3.2.l$	31	2	35	605	–	4737	590	590	31.33
$p3.2.m$	31	2	37.5	–	–	–	620	620	58.41
$p3.2.n$	31	2	40	–	–	–	660	660	26.72
$p3.2.o$	31	2	42.5	–	–	–	690	690	34.64
$p3.2.p$	31	2	45	–	–	–	720	720	39.39
$p3.2.q$	31	2	47.5	–	–	–	760	760	16.55
$p3.2.r$	31	2	50	–	–	–	790	790	14.25
$p3.2.s$	31	2	52.5	–	–	–	800	800	0.11
$p3.3.s$	31	3	35	738.91	–	416	720	720	188.04
$p3.3.t$	31	3	36.7	763.69	–	4181	760	760	93.71
$p4.2.h$	98	2	60	–	–	–	835	835	2783.76
$p4.2.i$	98	2	65	–	–	–	918	918	1511.7
$p4.2.t$	98	2	120	–	–	–	1306	1306	1.29
$p4.3.g$	81	3	36.7	656.38	653	52	665	653	–
$p4.3.h$	90	3	40	735.38	729	801	761	729	–
$p4.3.i$	94	3	43.3	813.63	809	4920	830	809	–
$p4.4.i$	68	4	32.5	665.4	657	23	660	657	–
$p4.4.j$	76	4	35	741.47	732	141	784	732	–
$p4.4.k$	83	4	37.5	831.95	821	558	860	821	–

continued on next page

Table 2 – continued from previous page

Instance	*n*	*m*	*L*	B-P [2]			B-C		
				UB	*LB*	*CPU*	*UB*	*LB*	*CPU*
*p*5.2.*l*	64	2	30	–	–	–	800	800	0.49
*p*5.2.*m*	64	2	32.5	–	–	–	860	860	0.97
*p*5.2.*p*	64	2	40	–	–	–	1150	1150	0.79
*p*5.2.*t*	64	2	50	–	–	–	1400	1400	3162.41
*p*5.2.*x*	64	2	60	–	–	–	1610	1610	38.81
*p*5.2.*z*	64	2	65	–	–	–	1680	1680	0.82
*p*5.3.*l*	64	3	20	605	595	33	615	595	–
*p*5.3.*m*	64	3	21.7	650	650	2	660	650	–
*p*5.3.*n*	64	3	23.3	755	755	42	765	755	–
*p*5.4.*l*	44	4	15	430	430	1	445	430	–
*p*5.4.*m*	52	4	16.2	555	555	0	560	555	–
*p*5.4.*n*	60	4	17.5	620	620	0	640	620	–
*p*5.4.*o*	60	4	18.8	690	690	1	720	690	–
*p*5.4.*p*	64	4	20	790	765	729	820	765	–
*p*5.4.*q*	64	4	21.2	860	860	1	880	860	–
*p*5.4.*v*	64	4	27.5	1320	1320	446	1340	1320	–
*p*6.2.*j*	62	2	30	–	–	–	948	948	0.46
*p*6.2.*k*	62	2	32.5	–	–	–	1032	1032	137.85
*p*6.2.*l*	62	2	35	–	–	–	1116	1116	13.92
*p*6.2.*m*	62	2	37.5	–	–	–	1188	1188	5.48
*p*6.2.*n*	62	2	40	–	–	–	1260	1260	1.03
*p*6.3.*m*	62	3	25	1104	–	33	1080	1080	574.45
*p*7.2.*g*	87	2	70	–	–	–	459	459	520.18
*p*7.3.*h*	59	3	53.3	429	425	8	436	425	–
*p*7.3.*i*	70	3	60	496.98	487	3407	535	487	–
*p*7.4.*j*	51	4	50	462	462	1	481	462	–
*p*7.4.*k*	61	4	55	524.61	520	73	586	520	–
*p*7.4.*l*	70	4	60	593.63	590	778	667	590	–

Conclusion and future work

In this article, we presented a branch-and-cut algorithm to solve TOP. Several cuts that strengthen the classical linear formulation were proposed. They include symmetric breaking, generalized subtour eliminations, boundaries on profits/numbers of customers and clique cuts. Experiments conducted on the standard benchmark show that our algorithm has the ability to solve a large number and a variety of instances. The algorithm permits to close several new instances. The obtained results clearly show the competitiveness and the robustness of our method on the classical linear TOP formulation. For future work, we plan to extend the approach to solve other combinatorial optimization problems.

Bibliography

[1] H. Bouly, A. Moukrim, D. Chanteur, and L. Simon. Un algorithme de destruction/construction itératif pour la résolution d'un problème de tournées de véhicules spécifique. In *MOSIM'08*, 2008.

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

[3] S. E. Butt and D. M. Ryan. An optimal solution procedure for the multiple tour maximum collection problem using column generation. *Computers & Operations Research*, 26:427–441, 1999.

[4] I.-M. Chao, B. Golden, and E. Wasil. The team orienteering problem. *European Journal of Operational Research*, 88:464–474, 1996.

[5] D.-C. Dang and A. Moukrim. Subgraph extraction and metaheuristics for the maximum clique problem. *Journal of Heuristics*, 18:767–794, 2012.

[6] D.-C. Dang, R. N. Guibadj, and A. Moukrim. A PSO-inspired algorithm for the team orienteering problem. Submitted preprint, 2012.

[7] M. Fischetti, J. J. Salazar González, and P. Toth. Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10: 133–148, 1998.

[8] M. Poggi de Aragão, H. Viana, and E. Uchoa. The team orienteering problem: Formulations and branch-cut and price. In *ATMOS*, pages 142–155, 2010.

[9] H. Tang and E. Miller-Hooks. A tabu search heuristic for the team orienteering problem. *Computer & Operations Research*, 32:1379–1407, 2005.

[10] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden. Metaheuristics for tourist trip planning. In *Metaheuristics in the Service Industry*, volume 624 of *Lecture Notes in Economics and Mathematical Systems*, pages 15–31. Springer Berlin Heidelberg, 2009.