Citation for published version:
Erdogan, G & Laporte, G 2013, 'The orienteering problem with variable profits', Networks, vol. 61, no. 2, pp. 104-116. https://doi.org/10.1002/net.21496

DOI:
10.1002/net.21496

Publication date:
2013

Document Version
Early version, also known as pre-print

The Orienteering Problem with Variable Profits

Güneş Erdoğan [*]   Gilbert Laporte [†]

July 10, 2012

Abstract

This paper introduces, models and solves a generalization of the *Orienteering Problem*, called the *The Orienteering Problem with Variable Profits* (OPVP). The OPVP is defined on a complete undirected graph $G = (V, E)$, with a depot at vertex 0. Every vertex $i \in V \setminus \{0\}$ has a profit p_i to be collected, and an associated collection parameter $\alpha_i \in [0, 1]$. The vehicle may make a number of "passes", collecting $100\alpha_i$ percent of the remaining profit at each pass. In an alternative model, the vehicle may spend a continuous amount of time at every vertex, collecting a percentage of the profit given by a function of the time spent. The objective is to determine a maximal profit tour for the vehicle, starting and ending at the depot, and not exceeding a travel time limit.

Keywords: orienteering problem, linearization, interger programming, branch-and-cut.

1 Introduction

In the *Orienteering Problem with Variable Profits* (OPVP), we are given a complete undirected graph $G = (V, E)$, where V is the set of vertices and E is the set of edges. Let T be a subset of *compulsory vertices* of V, including a *depot* 0. Every vertex $i \in V \setminus \{0\}$ has a profit p_i to be collected, and

[*]University of Southampton, School of Management, Highfield, Southampton, SO17 1BJ, United Kingdom G.Erdogan@soton.ac.uk

[†]Canada Research Chair in Distribution Management, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montreal, Canada H3T 2A7 gilbert.laporte@cirrelt.ca

an associated collection parameter $\alpha_i \in [0, 1]$. A vehicle based at the depot may make a number of "passes" at each vertex it visits. In our context, a pass means staying a predefined amount of time at a vertex. When a vehicle makes k passes at a vertex, it stays there k times as long as if it did a single pass, before leaving the vertex. Each pass at vertex i requires r_i units of time and successfully collects $100\alpha_i$ percent of the remaining profit. A travel time t_{ij} satisfying the triangle inequality is associated with every edge $(i, j) \in E$. The aim is to determine a maximal profit tour for the vehicle, starting and ending at the depot, visiting all compulsory vertices, and not exceeding a given travel time limit L. As an alternative model, we also consider the option of spending z_i time units at vertex i, collecting a percentage of the profit given by a function of z_i. Figure 1 depicts a feasible tour for the first model, where the size of the vertices reflect the associated profit and the mandatory vertices are emphasized with solid perimeters. Inner circles of increasing sizes represent larger amounts of collection through multiple passes. The vehicle leaves the depot and visits five vertices. It performs two passes at the first and fourth vertex it visits, and returns to the depot after visiting the fifth vertex.

The OPVP belongs to a broader class of problems known as *Traveling Salesman Problems with Profits* (TSPPs). We refer the interested reader to the comprehensive survey by Feillet et al. (2005). Based on the classification system presented in this survey, TSPPs contain three subclasses, depending on how the objectives of minimizing distance and maximizing profit are handled. The first class consists of problems in which both objectives are combined in a linear fashion. The second class is composed of problems in which the travel cost is a constraint and the objective is to maximize the profits collected. Finally, problems in which the profit is a constraint and the objective is to minimize the travel cost constitute the third class. With respect to this classification scheme, the OPVP belongs to the second class, together with the *Orienteering Problem* (OP) (Golden et al. 1987), the *Maximum Collection Problem* (Kataoka and Morito 1988), and the *Selective Traveling Salesman Problem* (STSP) (Laporte and Martello 1990).

In the past decade, a number of studies have been conducted in the area of the TSPP, all belonging to the second class. Most have focused on the *Team Orienteering Problem* (TOP), a multi-vehicle variant of the OP. The TOP has been tackled by Tang and Miller-Hooks (2005), Archetti et al. (2007), and Vansteenwegen et al. (2009) using metaheuristics, as well as by Archetti et al. (2009) who have applied both heuristic and exact methods. To the best of our knowledge, the most successful heuristics are due to Ke et al. (2008) and Souffriau et al. (2010), whereas the most successful exact algorithm is that of Archetti et al. (2009). The paper by Archetti et al. (2009) also focuses on the *Profitable Tour Problem*, the single vehicle variant of the TOP.

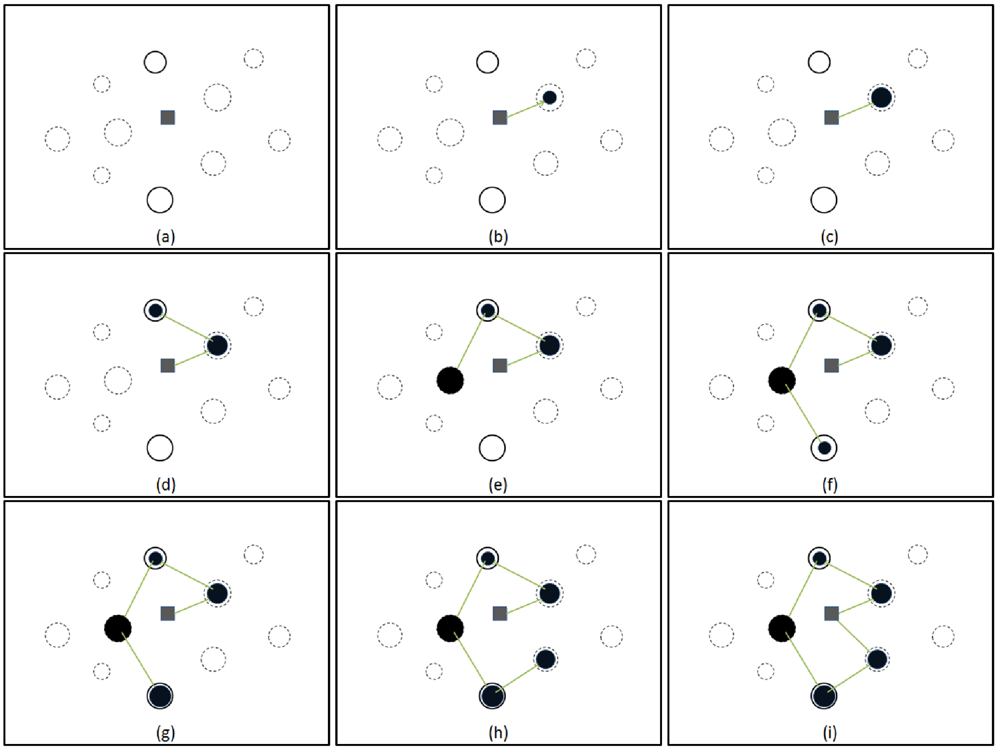

Figure 1: An illustrative example of the OPVP

An interesting application of the OP to the design of circuits for tourists is presented by Souffriau et al. (2008). We refer the reader to the paper by Vansteenwegen et al. (2011) for a recent survey on the OP.

Outside the domain of the OP, two notable studies have focused on a bi-objective variant of the STSP (Bérubé et al. 2009a) and on an undirected formulation for the *Prize Collecting Traveling Salesman Problem* (Bérubé et al. 2009b). Finally, Erdoğan et al. (2010) have presented the *Attractive Traveling Salesman Problem*, where the vertices are partitioned into facility vertices and customer vertices. The vehicle visits a subset of the facility vertices, and every such vertex attracts a portion of the profit from the customers according to their distance from the facility, and on its attractiveness.

A potential application of the OPVP arises in the fishing sector. For example, in North America, there often exists a legal time limit on fishing, the suitable locations for fishing are quite specific, and the amount of fish at each location is variable. Another application can be found in the entertainment sector, where multiple shows or a longer stay at a location may be required to collect more profit. A military application is encountered in the routing of a reconnaissance vehicle, which has the option of staying longer at a location to gather more information. A final application arises in humanitarian logistics, where a helicopter looks for the survivors of a disaster, distributing food and first aid kits whenever survivors are found. Multiple passes or longer stays at target locations should increase the expected number of survivors found.

Our aim is to develop a unified branch-and-cut algorithm for the two versions of the OPVP just described. The remainder of this paper is organized as follows. In Sections 2 and 3, we present the integer programming formulations for the case with discrete passes and for the case with continuous time at each visited vertex, respectively. In Sections 4 and 5, we describe linearization schemes for the cases with concave and convex collection functions, respectively. In Section 6, we provide the details of the valid inequalities we adapt from the literature and associated the branch-and-cut-algorithm. In Section 7, we present the computational results. Finally, we draw some conclusions in Section 8.

2 OPVP with Discrete Passes

We first present a model for the case with discrete passes. We denote the theoretical maximum number of passes at vertex i as m_i ($\leq \lfloor (L - 2t_{0i})/r_i \rfloor$), the value of which may depend on a physical constraint (e.g. fuel), or a managerial constraint (e.g. possibility of overfishing, danger of being noticed). Let x_{ij} be equal to 1 if the vehicle traverses edge $(i,j) \in E$, and 0 otherwise. Furthermore, let y_{ik} be equal to 1 if k or more passes are performed at vertex i, and 0 otherwise.

(OPVP1)

$$\text{maximize} \sum_{i \in V \setminus \{0\}} p_i \sum_{k \in \{1,\dots,m_i\}} \alpha_i (1-\alpha_i)^{k-1} y_{ik} \tag{1}$$

subject to

$$\sum_{j:(i,j)\in E} x_{ij} = 2y_{i1} \quad (i \in V) \tag{2}$$

$$\sum_{\substack{i \in S, j \in V \setminus S \\ \text{or } i \in V \setminus S, j \in V}} x_{ij} \geq 2y_{t1} \quad (S \subset V : 2 \leq |S| \leq |V|-2, T \setminus S \neq \emptyset, t \in S) \tag{3}$$

$$y_{ik} \leq y_{i,k-1} \quad (i \in V, k \in \{2,\dots,m_i\}) \tag{4}$$

$$\sum_{(i,j)\in E} t_{ij}x_{ij} + \sum_{i\in V\setminus\{0\}} r_i \sum_{k\in\{1,\dots,m_i\}} y_{ik} \leq L \tag{5}$$

$$y_{i1} = 1 \quad (i \in T) \tag{6}$$

$$y_{i1} = 0 \text{ or } 1 \quad (i \in V \setminus T) \tag{7}$$

$$y_{ik} = 0 \text{ or } 1 \quad (i \in V, k \in \{2, ..., m_i\}) \tag{8}$$

$$x_{ij} = 0 \text{ or } 1 \quad ((i,j) \in E). \tag{9}$$

The objective function (1) maximizes the profit to be collected. The triangle inequality implies that a vertex need not be visited if no collection must be made there, hence we can use constraints (2) as the degree constraints, and constraints (3) as the connectivity constraints. These inequalities are adapted from the *Covering Tour Problem* (CTP) formulation (Gendreau et al. 1997). Constraints (4) prohibit the vehicle from collecting a profit without visiting the corresponding vertex. Note that in the absence of constraints (4), the formulation allows a solution with $y_{i1} = 0$ and $y_{ik} = 1\,(k > 1)$, which corresponds to making the k^{th} pass without making the first pass that corresponds to the visit. Constraints (5) enforce the time limit, while constraints (6) state that all vertices in T must be visited. The integrality requirements are defined by constraints (7), (8) and (9).

Regarding the generality of OPVP1, the objective function (1) and constraints (4) deserve a closer look. The objective function can be modified to incorporate any collection function (that may be convex, concave, or neither) by replacing the term $\alpha_i(1-\alpha_i)^{k-1}y_{ik}$ in (1) with $f(i,k)$ which denotes the amount of collection for the k^{th} pass at vertex i. The collection function that is used in (1) is almost identical to the objective function of the *Maximum Expected Coverage Location Problem* (MEXCLP) model of Daskin (1983), which was proposed as a model for covering demand points with probabilistically available resources (ambulances). The MEXCLP does not impose constraints of type (4), since the return of the k^{th} resource covering a demand point is always less than that of the $(k+1)^{\text{st}}$ resource. In the case of OPVP1, however, the first pass at a vertex requires the vertex to be visited. In the absence of constraints (4), the model could make further passes at a vertex without actually making a first pass at that vertex. We also point out that the STSP is a special case of the OPVP1, where $\alpha_i = 1$ for all $i \in V \setminus \{0\}$, which proves that the OPVP1 is NP-Hard.

3 OPVP with Continuous Time

We now move on to the second model in which the vehicle may spend a continuous amount of time at a vertex it visits. We denote the maximum amount of time that can be spent at vertex i as u_i ($\leq L - 2t_{0i}$). Similar to the parameter m_i in Section 2, the value of which may depend on a physical constraint (e.g. fuel), or a managerial constraint (e.g. possibility of overfishing, danger of being noticed). Let y_i be equal to 1 if vertex i is visited, and 0 otherwise. Furthermore, let z_i be the time spent at vertex i, and let $f_i(z_i)$ be the corresponding amount collected. The resulting model is

(OPVP2)

$$\text{maximize} \sum_{i \in V \setminus \{0\}} p_i f_i(z_i) \tag{10}$$

subject to

$$\sum_{j:(i,j) \in E} x_{ij} = 2y_i \quad (i \in V) \tag{11}$$

$$\sum_{\substack{i \in S, j \in V \setminus S \\ \text{or } i \in V \setminus S, j \in V}} x_{ij} \geq 2y_k \quad (S \subset V : 2 \leq |S| \leq |V| - 2, T \setminus S \neq \emptyset, k \in S) \tag{12}$$

$$z_i \geq r_i y_i \quad (i \in V \setminus \{0\}) \tag{13}$$

$$z_i \leq u_i y_i \quad (i \in V \setminus \{0\}) \tag{14}$$

$$\sum_{(i,j) \in E} t_{ij} x_{ij} + \sum_{i \in V \setminus \{0\}} z_i \leq L \tag{15}$$

$$y_i = 1 \quad (i \in T) \tag{16}$$

$$z_i \geq 0 \quad (i \in V \setminus \{0\}) \tag{17}$$

$$y_i = 0 \text{ or } 1 \quad (i \in V \setminus \{0\}) \tag{18}$$

$$x_{ij} = 0 \text{ or } 1 \quad ((i,j) \in E). \tag{19}$$

Objective function (10), just as that of OPVP1, maximizes the collected profit. Constraints (11) are degree constraints, and constraints (12) are the connectivity constraints, stated using the new variable definition. Constraints (13) force the vehicle to spend the minimum collection time at a vertex if it is visited. Constraints (14) prohibit the vehicle from collecting a profit without visiting the vertex. Constraints (15) enforce the time limit, while constraints (16) state that all vertices in T must be visited. The nonnegativity constraints are defined by (17) and the integrality constraints by (18) and (19). As for OPVP1, the STSP is a special case of the OPVP2, where $f_i(z_i) = 1$ for $z_i \geq r_i$, and $f_i(z_i) = 0$ for $z_i < r_i$, for all $i \in V \setminus \{0\}$. This relationship also proves that the OPVP2 is NP-Hard.

4 Linearization Scheme for Concave Collection Functions

We now focus on the case where the collection functions $f_i(z_i)$, $i \in V \setminus \{0\}$ are concave, which we call OPVP2-CV. Even though the resulting model can theoretically be solved through nonlinear optimization algorithms, we opt to linearize it since most nonlinear solvers work on a static problem object, and do not accept the addition of cuts. The presence of constraints (12), which have to be identified and added in the course of the algorithm, makes linearization a more suitable solution approach. As part of the linearization scheme, we define auxiliary variables w_i to denote the fraction of profit collected at vertex i. We define the collection function as $f_i(z_i) = 1 - e^{-\beta_i z_i}$, where $\beta_i > 0$ is a control parameter. Applying the linearization approach of Erdoğan et al. (2010), we construct the following valid inequalities:

$$w_i \leq \beta_i e^{-\beta_i z_i^*} z_i + 1 - e^{-\beta_i z_i^*} - z_i^* \beta_i e^{-\beta_i z_i^*} \quad (i \in V \setminus \{0\}, z_i^* \in [0, u_i]). \quad (20)$$

Inequalities (20) simply define the tangents of each collection function. We emphasize the fact that this approach is applicable for every choice of concave collection function, and is quite similar to that of Quesada and Grossman (1992) for convex MINLP optimization problems. The resulting linearized formulation for OPVP2-CV is

$$\text{maximize} \sum_{i \in V \setminus \{0\}} p_i w_i \quad (21)$$

subject to

$$0 \leq w_i \leq 1 \quad (i \in V \setminus \{0\}), \quad (22)$$

and (11) – (20).

The auxiliary variables allow us to impose the following set of valid inequalities without using nonlinear functions:

$$w_i \leq (1 - e^{-\beta_i u_i}) y_i \quad (i \in V \setminus \{0\}). \quad (23)$$

Denote the linearized formulation including valid inequalities (23) as OPVP2-CV-L. Furthermore, denote the continuous relaxation of an integer programming formulation F as F^R, and its optimal objective value as $v(F^R)$. We now state the relationship between $v(\text{OPVP2-CV-L}^R)$ and $v(\text{OPVP2-CV}^R)$, and show that the relaxation of OPVP2-CV-L is stronger than OPVP2-CV.

Proposition 1. $v(\text{OPVP2-CV-L}^R) \leq v(\text{OPVP2-CV}^R)$.

Proof: Clearly, any solution for OPVP2-CV-L^R is feasible for OPVP2-CV^R. We now prove that certain feasible solutions of OPVP2-CV^R are not feasible for OPVP2-CV-L^R. The objective function (10) and constraints (14) set the upper bound of the profit to be collected at vertex i as $1 - e^{-\beta_i u_i y_i}$ for OPVP2-CV^R. However, valid inequalities (23) set the upper bound as $(1 - e^{-\beta_i u_i})y_i$ for OPVP2-CV-L^R. We need to show that

$$g(y_i) = (1 - e^{-\beta_i u_i})y_i - (1 - e^{-\beta_i u_i y_i}) \leq 0 \text{ for } 0 \leq y_i \leq 1. \tag{24}$$

Observe that $g(y_i)$ is a continuous and differentiable function in the interval $0 \leq y_i \leq 1$ and $g(0) = g(1) = 0$. These properties allow us to apply Rolle's theorem, and to conclude that $g'(y_i)$ has at least one root in this interval. The first derivative of $g(y_i)$ can be computed as $g'(y_i) = 1 - e^{-\beta_i u_i} - \beta_i u_i e^{-\beta_i u_i y_i}$ and the second derivative as $g''(y_i) = (\beta_i u_i)^2 e^{-\beta_i u_i y_i}$. The only root of $g'(y_i)$ is $y_i^* = -\ln((1 - e^{-\beta_i u_i})/\beta_i u_i)/\beta_i u_i$, which we know to be in the interval $(0, 1)$ through Rolle's theorem. Since $g''(y_i)$ is strictly positive, we can conclude that it is a local minimum. Consequently, $g(y_i)$ attains its maximum at the endpoints with a maximal value of 0. □

5 Linearization Scheme for Convex Collection Functions

In this section, we focus on the case where the collection functions $f_i(z_i), i \in V \setminus \{0\}$ are convex, which we call OPVP2-CX. We define the collection function as $f_i(z_i) = (e^{(z_i/u_i)} - 1)/(e - 1)$, although the following analyses apply to any convex collection function. For the case of convex collection functions, the valid inequality approach given in the previous section fails to set the exact value of the auxiliary variable. We need linear functions $h_i(z_i)$ that will provide an upper bound for w_i, such that $\int_0^{u_i}(h_i(z_i) - f_i(z_i))\,\mathrm{d}z_i$ is minimized. As depicted in Figure 2a, the linear function satisfying this condition is the line connecting the points $(0, 0)$ and $(u_i, f_i(u_i))$, i.e. $h_i(z_i) = z_i(f_i(u_i)/u_i)$. The resulting valid inequalities that provide an imperfect linearization are

$$w_i \leq z_i \frac{f_i(u_i)}{u_i}. \tag{25}$$

Let us create a linearization for OPVP2-CX by replacing (20) by (25) in OPVP2-CV-L. We denote this linearization as OPVP2-CX-L, its relaxation as OPVP2-CX-L^R, and a feasible solution for OPVP2-CX-L^R as w_i^*, z_i^*. If w_i^*

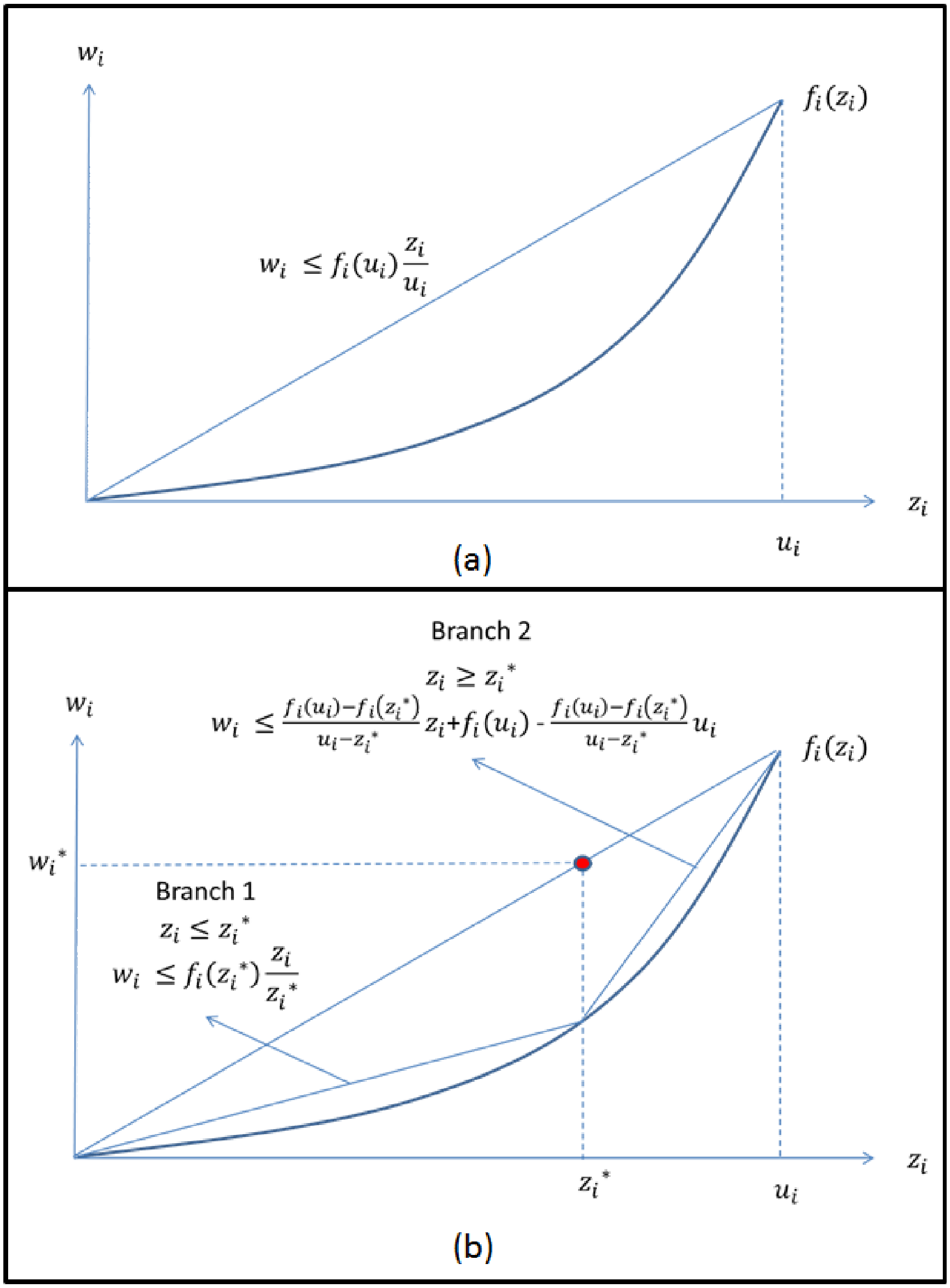

Figure 2: Branching scheme for the linearization of convex collection functions

assumes a value that is greater than the collection function evaluated at z_i^*, i.e. $f_i(z_i^*) < w_i^*$ (as depicted in Figure 2b), then this solution is infeasible for the OPVP2-CX. The remedy is to construct two subproblems by branching on the bounds of z_i, and adding new linearization constraints for the two convex subparts of $f_i(z_i)$, also depicted in Figure 2b. We now formally state this result.

Proposition 2. For a solution of OPVP2-CX-L^R with $w_i^*, z_i^* : f_i(z_i^*) < w_i^*$ where $z_i \in [a_i, b_i]$ and $z_i^* \in (a_i, b_i)$, a branching scheme that constructs two subproblems, the first one with the constraints

$$z_i \geq z_i^*, \tag{26}$$
$$w_i \leq z_i \tfrac{f_i(b_i)-f_i(z_i^*)}{b_i-z_i^*} + f_i(b_i) - b_i \tfrac{f_i(b_i)-f_i(z_i^*)}{b_i-z_i^*}, \tag{27}$$

and the second one with the constraints

$$z_i \leq z_i^*, \tag{28}$$
$$w_i \leq z_i \tfrac{f_i(z_i^*)-f_i(a_i)}{z_i^*-a_i} + f_i(a_i) - a_i \tfrac{f_i(z_i^*)-f_i(a_i)}{z_i^*-a_i} \tag{29}$$

finds the optimal solution.

Proof: The result follows from the fact that the branching scheme successfully discards the infeasible point and partitions the problem into two subproblems whose feasible regions contain all feasible solutions of the corresponding partition of the original problem. □

Although the branching scheme will result in an optimal solution, the implementation may prove to have a high computational cost since almost all the variables w_i will need to be branched on. We now analyze the similarity of a subproblem of OPVP2-CX with another problem from the supply chain literature, which will help us construct a stronger formulation. The *Concave Cost Supply Problem* (CCSP), is the problem of choosing among n suppliers to purchase a given quantity A of a single item or service type. Each supplier $i \in \{1, ..., n\}$ can provide the items subject to the conditions that 1) there is a minimum amount m_i to be purchased if the supplier is chosen to provide the item, 2) the supplier cannot provide more than M_i units, and 3) the price per item monotonically decreases as the amount purchased increases (hence, the concavity). The aim of the CCSP is to choose a subset of suppliers as well as the quantities to be purchased from the chosen suppliers, so as to satisfy the demand requirement and to minimize the total cost. Chauhan and Proth (2003) have studied the CCSP and have provided the following formulation:

$$\text{minimize} \sum_{i\in\{1,...,n\}} k_i(x_i) \tag{30}$$

subject to

$$\sum_{i\in\{1,...,n\}} x_i = A, \tag{31}$$

$$x_i \in \{0\} \cup [m_i, M_i] \quad (i \in \{1, ..., n\}). \tag{32}$$

where $k_i(x_i)$ are concave functions of $x_i, \forall i \in \{1, ..., n\}$. The authors have also proved that there exists at least one optimal CCSP solution in which there is at most one variable $x_i^* \in (m_i, M_i)$, and the remaining variables are equal to zero or one of their associated bounds, i.e. $x_j^* \in \{0, m_j, M_j\}, \forall j \in \{1, ..., n\} \setminus \{i\}$. Now consider the subproblem of OPVP2-CX in which the routing decisions are fixed, i.e. $x_{ij} = x_{ij}^*, \forall (i,j) \in E$ and $y_i = y_i^*, \forall i \in V$, and denote it OPVP2-CX-S. Clearly, OPVP2-CX-S consists of a maximization of separable convex functions with respect to a time limit constraint, and upper and lower bounds for every visit duration. We now show that OPVP2-CX-S has the same property as CCSP.

Proposition 3. OPVP-CX-S has at least one optimal solution in which at most one vertex i is visited with an intermediate collection time ($r_i < z_i^* < u_i$, $y_i^* = 1$), and the rest of the vertices are visited with either maximal or minimal collection times ($z_j^* \in \{r_i, u_i\}, j \in V \setminus \{0, i\} : y_j^* = 1$).

Proof: Take any feasible solution for OPVP-CX-S, $(\bar{z})$. Assume that there exist two vertices $i, j \in V \setminus \{0\}$ such that $r_i < \bar{z}_i < u_i$, $r_j < \bar{z}_j < u_j$, and $f_i'(\bar{z}_i) \geq f_j'(\bar{z}_j)$. Define $\delta = \min\{u_i - \bar{z}_i, \bar{z}_j - r_j\}$ and construct another feasible solution $(\hat{z})$ as $\hat{z}_i = \bar{z}_i + \delta, \hat{z}_j = \bar{z}_j - \delta, \hat{z}_k = \bar{z}_k \, \forall k \in V \setminus \{0, i, j\}$. Since the collection functions $f_i(z_i)$ and $f_j(z_j)$ are convex, their tangents constructed at $\bar{z}_i$ and $\bar{z}_j$ will provide lower bounds for $f_i(\hat{z}_i)$ and $f_j(\hat{z}_j)$, i.e.

$$f_i(\hat{z}_i) \geq (\bar{z}_i + \delta) f_i'(\bar{z}_i) + f_i(\bar{z}_i) - \bar{z}_i f_i'(\bar{z}_i) \tag{33}$$

and

$$f_j(\hat{z}_j) \geq (\bar{z}_j - \delta) f_j'(\bar{z}_j) + f_j(\bar{z}_j) - \bar{z}_j f_j'(\bar{z}_j). \tag{34}$$

Summing up the inequalities (33) and (34) gives

$$f_i(\hat{z}_i) + f_j(\hat{z}_j) \geq f_i(\bar{z}_i) + f_j(\bar{z}_j) + \delta(f_i'(\bar{z}_i) - f_j'(\bar{z}_j)). \tag{35}$$

Since $\delta > 0$ and $f_i'(\bar{z}_i) \geq f_j'(\bar{z}_j)$, we can conclude that $f_i(\hat{z}_i) + f_j(\hat{z}_j) \geq f_i(\bar{z}_i) + f_j(\bar{z}_j)$. Repeating this process with different pairs of variables will lead to a solution with at most one intermediate collection time. □

Because Proposition 3 holds for any instance of OPVP2-CX-S, it carries over to OPVP2-CX. We now construct an improved linearization for OPVP2-CX based on this property. Let y_i^1 be 1 if vertex i is visited with the minimum collection time r_i, and 0 otherwise. Let y_i^2 be 1 if vertex i is visited with an intermediate collection time $z_i \in [r_i, u_i]$, and 0 otherwise. Finally, let y_i^3 be 1 if vertex i is visited with the maximum collection time u_i, and 0 otherwise. The formulation is then

(OPVP2-CX-L2)

$$\text{maximize} \sum_{i \in V \setminus \{0\}} p_i(f_i(r_i)y_i^1 + w_i + f_i(u_i)y_i^3) \tag{36}$$

subject to

$$\sum_{j:(i,j)\in E} x_{ij} = 2(y_i^1 + y_i^2 + y_i^3) \quad (i \in V) \tag{37}$$

$$\sum_{\substack{i \in S, j \in V \setminus S \\ \text{or } i \in V \setminus S, j \in V}} x_{ij} \geq 2(y_k^1 + y_k^2 + y_k^3) \quad (S \subset V : 2 \leq |S| \leq |V| - 2, T \setminus S \neq \emptyset, k \in S) \tag{38}$$

$$y_i^1 + y_i^2 + y_i^3 = 1 \quad (i \in T) \tag{39}$$

$$y_i^1 + y_i^2 + y_i^3 \leq 1 \quad (i \in V \setminus T) \tag{40}$$

$$\sum_{i \in V \setminus \{0\}} y_i^2 \leq 1 \tag{41}$$

$$z_i \geq r_i y_i^2 \quad (i \in V \setminus \{0\}) \tag{42}$$

$$z_i \leq u_i y_i^2 \quad (i \in V \setminus \{0\}) \tag{43}$$

$$\sum_{(i,j)\in E} t_{ij} x_{ij} + \sum_{i \in V \setminus \{0\}} (r_i y_i^1 + z_i + u_i y_i^3) \leq L \tag{44}$$

$$w_i \leq z_i \frac{f_i(u_i)}{u_i} \quad (i \in V \setminus \{0\}) \tag{45}$$

$$z_i \geq 0 \quad (i \in V \setminus \{0\}) \tag{46}$$

$$y_i^k = 0 \text{ or } 1 \quad (i \in V \setminus \{0\}, k \in \{1, 2, 3\}) \tag{47}$$

$$x_{ij} = 0 \text{ or } 1 \quad ((i, j) \in E). \tag{48}$$

The objective function (36) maximizes the collected profit. Constraints (37) are degree constraints, and constraints (38) are connectivity constraints, stated using the three new variable definitions. Constraints (39) force the vehicle to visit the compulsory vertices. Constraints (40) dictate that a non-compulsory vertex can be visited with no more than one of the given options. Constraints (41) ensure that at most one vertex is visited with an intermediate visit time. Constraints (42) and (43) set the lower and upper bounds for the time spent at a vertex visited with the option of intermediate collection time. Constraints (44) enforce the time limit, and constraints (45) define the initial (and imperfect) linearization. The nonnegativity constraints are defined by (46) and the integrality constraints are defined by (47) and (48).

As a final note, we stress the fact that variants of OPVP2 involving vertices with convex collection functions as well as vertices with concave collection functions have this property. To solve such problems, both linearization schemes should be used in conjunction.

6 Valid Inequalities and Branch-and-Cut Algorithm

The similarity in the structures of the OPVP and the CTP enables us to adapt valid inequalities for the CTP to OPVP1, OPVP2-CV-L, and OPVP2-CX-L2. The proofs of validity are identical for CTP and the OPVP2-CV-L, and extend to OPVP1 through the transformation $y_i = y_{i1}$, and to OPVP2-CX-L2 through the transformation $y_i = y_i^1 + y_i^2 + y_i^3$.

1) Arc-vertex constraints

Proposition 4. The inequalities

$$x_{ij} \leq y_i \quad (i, j \in V) \tag{49}$$

and

$$x_{ij} \leq y_j \quad (i, j \in V) \tag{50}$$

are valid for OPVP2-CV-L.

2) Strong connectivity constraints

Proposition 5. The following inequalities are valid for OPVP2-CV-L:

$$\sum_{\substack{i\in S, j\in V\setminus S \\ \text{or } i\in V\setminus S, j\in V}} x_{ij} \geq 2 \quad (S \subset V : 2 \leq |S| \leq |V|-2, T\setminus S \neq \emptyset, S\cap T \neq \emptyset). \tag{51}$$

3) Strong 2-matching constraints

Proposition 6. The following inequalities are valid for OPVP2-CV-L:

$$\sum_{i,j\in H} x_{ij} + \sum_{i,j\in E'} x_{ij} \leq \sum_{i\in H} y_i + \frac{1}{2}(|E'|-1), \tag{52}$$

for all $H \subset V$ and $E' \subset E$ satisfying
(i) $|\{i,j\}\cap H| = 1 \quad ((i,j)\in E')$,
(ii) $\{i,j\}\cap\{k,l\} = \emptyset \quad ((i,j)\neq(k,l)\in E')$,
(iii) $|E'| \geq 3$ and odd.

Note that Propositions 5 and 6 were already mentioned by Gendreau et al. (1998) as well as by Erdoğan et al. (2010), and are provided in this paper for the sake of completeness. We now describe the unified branch-and-cut algorithm capable of handling all three formulations OPVP1, OPVP2-CV-L, and OPVP2-CX-L2.

4) Branch-and-cut algorithm

Step 1 (Root node). Construct a subproblem consisting of the initial formulation. Note that for OPVP2-CV-L this corresponds to a subproblem that does not contain any linearization constraints. Insert this subproblem in a list.
Step 2 (Node selection). If the list is empty, stop. Else select and remove a subproblem from the list.
Step 3 (Subproblem solution). Solve the subproblem. If the objective function value is less than the best lower bound, go to Step 2.
Step 4 (Constraint generation). Identify violated members of the associated constraints and related valid inequalities (arc-vertex constraints, strong connectivity constraints, strong 2-matching constraints, and linearization constraints), and add them to the subproblem. If at least one constraint is generated, go to Step 3.
Step 5 (Integrality check). For OPVP1 and OPVP2-CV-L, if the solution is integer, update the best known lower bound, and go to Step 2. For OPVP2-CX-L2, if the solution is integer and the auxiliary variable for the vertex with

intermediate collection time does not violate the collection amount, go to Step 2.

Step 6 (Branching). For OPVP1 and OPVP2-CV-L, construct two subproblems by branching on a binary fractional variable. For OPVP2-CX-L2, construct two subproblems by branching on a binary fractional variable, or by applying the branching scheme of Proposition 2 on an auxiliary variable exceeding the collection amount. Add the subproblems to the list and go to Step 2.

7 Computational Results

We have implemented the algorithm described in Section 7, utilizing C++ and CPLEX 12.1, and we have run a number of experiments on instances adapted from the TSPLIB (Reinelt 1991) on the IRIDIS computing cluster having Intel Nehalem nodes with two 4-core processors and 22 GB RAM. We have used the instances kroA100, kroB100, kroC100, kroA200, and kroB200, which are randomly generated points in the plane. We have used the following scheme to convert the data for OPVP1 and OPVP2. We take the first vertex in the data file to be the depot. We designate the next $|T|-1$ vertices together with the first vertex to constitute T. The next $|V|-|T|$ vertices are used as elements of $V \setminus T$. We compute t_{ij} using the Euclidean distance formula, and rounded to the closest integer. Let $X_{max}, X_{min}, Y_{max}, Y_{min}$ denote the maximum X coordinate, minimum X coordinate, maximum Y coordinate, and minimum Y coordinate of all vertices, respectively. For OPVP1, we set two parameters as $p_{min} = 10$ and $p_{max} = 100$. Using these parameters, we determine the profit of a vertex $i \in V$ to be $p_i = p_{min} + \lfloor X_i + Y_i \rfloor \bmod (p_{max} - p_{min})$. To determine the capture ratio α_i, we set two parameters as $\alpha_{min} = 10$ and $\alpha_{max} = 100$, and determine α_i as $\alpha_i = \alpha_{min} + ((\lfloor X_i + Y_i \rfloor \bmod 20)/20) \times (\alpha_{max} - \alpha_{min})$. The dwell times are also determined in a similar manner, using the formula $r_i = r_{min} + ((\lfloor X_i + Y_i \rfloor \bmod 15)/15) \times (r_{max} - r_{min})$, where $r_{min} = (X_{max} - X_{min} + Y_{max} - Y_{min})/100$ and $r_{max} = (X_{max} - X_{min} + Y_{max} - Y_{min})/50$. For OPVP2, we have used a similar conversion scheme and set $\beta_i = \alpha_i/200$. For both models, we have determined the travel time limit as $L = \lfloor 2.5 \times (X_{max} - X_{min} + Y_{max} - Y_{min}) \rfloor$. For OPVP2-CV, we have set $u_i = L - 2t_{0i}$ whereas for OPVP2-CX, we have used $u_i = 10r_i$.

The results for the instances kro100A, kro100B, and kro100C for both OPVP1, OPVP2-CV-L, OPVP2-CX-L2 are presented in Tables 1, 2, and 3 respectively. We have also run experiments with larger instances adapted from kro200A and kro200B, the results of which are presented in Table 4. The column headings are defined as follows:

Instance : Name of the TSPLIB instance that was adapted.
$|\boldsymbol{V}|$: Number of vertices in the graph.
$|\boldsymbol{T}|$: Number of compulsory vertices.
Objective value : The objective value of the best solution found.
Final gap : Percent deviation of the best solution found from the best upper bound.
B&C nodes : Number of nodes generated in the branch-and-cut tree.
Arc-vertex : Number of arc-vertex constraints added.
Strong conn. : Number of strong connectivity constraints added.
Strong 2-match. : Number of strong 2-matching constraints added.
Lin. cons. : Number of linearization constraints added.
CPU time (sec) : CPU time in seconds.

As can be observed from Table 1, OPVP1 solved all 36 instances with $|V| \leq 100$ within five minutes of CPU time. OPVP2-CV-L, as shown in 2, successfully solved 34 out of 36 instances. The maximum deviation from the best upper bound was observed to be 8.8%. Finally, OPVP2-CX-L2, as 3 demonstrates, successfully solved 31 out of 36 instances. The maximum deviation from the best upper bound was observed to be 8.0%. For all three formulations, the CPU time requirement is observed to drop as $|T|$ increases. A first explanation for this phenomenon lies in the increasing number of stronger connectivity constraints, which greatly improve the performance of the branch-and-cut algorithm despite the increased CPU time requirement for separation. The second reason is due to the fact that the determination of most of the routing by the compulsory vertices. The average CPU time requirement for OPVP2-CX-L2 is approximately three times that of OPVP2-CV-L, which shows the difference of strength of the linearization schemes. For the second instance set, OPVP1 was able to solve 21 out of 24 instances. The branch-and-cut algorithm failed to find a feasible solution for kroB200, with $|V| = 200$ and $|T| = 100$. Excluding this instance, the maximum optimality gap is observed to be 3.6%. Overall, the computational reach of OPVP1 is around 200 vertices, whereas that of OPVP2-CV-L and OPVP2-CX-L2 are about 75.

We now provide the results of OPVP1 when applied to OP instances from the literature. We have solved the "diamond shaped" instances with 64 vertices provided by Chao et al. (1996), using OPVP1 and the branch-and-cut algorithm. Out of the 14 instances, 11 were solved in less than 5 seconds, whereas the other three required 9, 11, and 45 minutes of CPU time. The instances with $|V| = 75$ and $|T| = 1$ are the most similar to this set of OP instances, which required 26.5 seconds of CPU time on the average. We conclude that OPVP1 requires more time than the majority of OP instances, but the results presented above are by no means exhaustive, and pathological instances may require considerably more time for both OP and OPVP1.

We have also performed an analysis on the results of all three models, the details of which are provided in Table 5. Additional column headings for this table are given below:

Vertices visited : The total number of vertices visited by the vehicle, including the depot.
Multiple passes : The number of vertices in which more than one pass have been performed.
Avg. passes : The average number of passes, among the vertices with multiple passes.
Max. passes : The maximum number of passes performed at any vertex.
Time limit : The total time L that can be spend for traveling and collection.
Min. time spent : The minimum collection time spent, among the visited vertices.
Avg. time spent : The average collection time spent at the visited vertices.
Max. time spent : The maximum collection time spent, among the visited vertices.
Min. visits : The number of visits performed with minimal collection time.
Inter. visits : The number of visits performed with the collection time between the maximum and minimum.
Max. visits : The number of visits performed with maximal collection time.

It can be easily observed that the optimal solution for the OPVP1 chooses to visit most vertices with multiple passes. On the average, 67% of vertices are visited with multiple passes. Among these vertices, the average number of passes is 4.16, with a maximum of 8 passes. This clearly shows that the OPVP1 returns different results than the OP, which requires at most one pass. The number of vertices visited increases as $|T|$ increases, though not in a linear fashion. On the contrary, the maximum number of passes decrease as $|T|$ increases. The reason beyond these phenomena is the tendency of the model to collect as much as possible once a vertex is visited, and move on to the next vertex as the marginal return of each additional pass drops.

OPVP2-CV-L shows a similar tendency, with both minimum time spent and maximum time spent at the vertices decreasing as T increases. In 75% of the instances, the minimum time spent actually drops to its absolute minimum as $|T|$ exceeds 50% of $|V|$. The maximum time spent also decreases so as to reflect the time spent in visiting mandatory vertices. The average time spent at the vertices is closer to the maximum time spent, although not in an extreme manner, reflecting the nature of the concave collection functions. OPVP2-CX-L2 behaves in a similar fashion to OPVP2-CV-L, with a higher and higher percentage of minimal visits as T increases. This model always opts for an intermediate time visit, to better utilize the travel time limit. OPVP2-CX-L2, in constrast with OPVP2-CV-L, favors the minimal visits over the maximal visits when $|T|$ is high. This is the result of the convex collection function, which requires more time to collect a considerable amount of the profit.

8 Conclusion

We have introduced a generalization of the OP, in which the collection of profits at a vertex require either a number of discrete passes or a continuous amount of time to be spent at the vertex. We have provided a linear integer programming model for the former case and a nonlinear integer programming model for the latter. We have devised linearization schemes for the nonlinear models for the cases of concave and convex collection functions. The linearization was achieved through valid inequalities in the concave case, whereas a branching scheme in conjunction with valid inequalities was required for the convex case. The linearization scheme for the concave collection functions also helped strengthen the original formulation through a linear number of linear constraints. A theoretical result from the supply chain literature was used for constructing an improved formulation for the convex case. We have adapted valid inequalities from the CTP, and presented a unified branch-and-cut algorithm using these valid inequalities. We have performed computational experiments on instances adapted from TSPLIB. Results show that the discrete pass model can be solved for about 200 vertices within two hours of computing time, whereas the continuous time model is beyond the computational reach for more than 75 vertices.

Acknowledgments: Thanks are due to the referees and to the Associate Editor for their valuable comments which have greatly improved the paper. We would also like to thank Maria Battarra for a fruitful discussion of the Concave Cost Supply Problem. This work was partly funded by the Canadian Natural Science and Engineering Research Council under grant 39682-10. This support is gratefully acknowledged.

References

C. Archetti, A. Hertz, and M. Speranza. Metaheuristics for the team orienteering problem. *Journal of Heuristics*, 13:49–76, 2007.

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

J.-F. Bérubé, M. Gendreau, and J.-Y. Potvin. An exact ϵ-constraint method for bi-objective combinatorial optimization problems: Application to the traveling salesman problem with profits. *European Journal of Operational Research*, 194:39 – 50, 2009a.

J.-F. Bérubé, M. Gendreau, and J.-Y. Potvin. A branch-and-cut algorithm for the undirected prize collecting traveling salesman problem. *Networks*, 54:56–67, 2009b.

I-M. Chao, B.L. Golden, and E.A. Wasil. A fast and effective heuristic for the orienteering problem. *European Journal of Operational Research*, 88: 475–489, 1996.

S.S. Chauhan and J.M. Proth. The concave cost supply problem. *European Journal of Operational Research*, 148:374–383, 2003.

M.S. Daskin. A maximum expected covering location model: Formulation, properties, and heuristic solution. *Transportation Science*, 17:48–70, 1983.

G. Erdoğan, J.-F. Cordeau, and G. Laporte. The attractive traveling salesman problem. *European Journal of Operational Research*, 203:515–530, 2010.

D. Feillet, P. Dejax, and M. Gendreau. Traveling salesman problems with profits. *Transportation Science*, 39:188–205, 2005.

M. Gendreau, G. Laporte, and F. Semet. The covering tour problem. *Operations Research*, 45:568–576, 1997.

M. Gendreau, G. Laporte, and F. Semet. A branch-and-cut algorithm for the undirected selective travelling salesman problem. *Networks*, 32:263–273, 1998.

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

S. Kataoka and S. Morito. An algorithm for the single constraint maximum collection problem. *Journal of Operations Research Society of Japan*, 31: 515–530, 1988.

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

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

I. Quesada and I. E. Grossman. An LP/NLP based branch and bound algorithm for convex minlp optimization problems. *Computers & Chemical Engineering*, 16:937–947, 1992.

G. Reinelt. TSPLIB – a traveling salesman problem library. *ORSA Journal on Computing*, 3:376–384, 1991.

W. Souffriau, P. Vansteenwegen, J. Vertommen, G. Vanden Berghe, and D. Van Oudheusden. A personalized tourist trip design algorithm for mobile tourist guides. *Applied Artificial Intelligence*, 22:964 – 985, 2008.

W. Souffriau, P. Vansteenwegen, G. Vanden Berghe, and D. Van Oudheusden. A path relinking approach for the team orienteering problem. *Computers & Operations Research*, 37:1853–1859, 2010.

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

P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden. A guided local search metaheuristic for the team orienteering problem. *European Journal of Operational Research*, 196:118 – 127, 2009.

P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209:1 – 10, 2011.

Table 1: Computational results for OPVP1

Instance	$\lvert V\rvert$	$\lvert T\rvert$	**Objective value**	**Final gap**	**B&C nodes**	**Arc-vertex**	**Strong conn.**	**Strong 2-match.**	**CPU time (sec)**
kroA100	25	1	754.37	0.0%	83	28	0	0	0.45
	25	6	782.46	0.0%	20	17	29	0	0.12
	25	12	814.27	0.0%	203	9	29	0	0.21
	25	18	869.43	0.0%	91	4	32	0	0.09
	50	1	1075.17	0.0%	220	99	0	0	9.64
	50	12	1112.29	0.0%	74	42	130	0	0.84
	50	25	1231.32	0.0%	65	25	184	0	0.56
	50	37	1225.40	0.0%	134	9	102	10	0.68
	75	1	1238.88	0.0%	212	151	0	53	35.15
	75	18	1309.66	0.0%	10	77	525	2	6.58
	75	37	1333.48	0.0%	585	36	325	0	5.51
	75	56	1492.97	0.0%	455	20	734	16	7.63
	100	1	1342.30	0.0%	189	236	0	15	122.60
	100	25	1466.14	0.0%	1266	103	1321	119	85.43
	100	50	1475.50	0.0%	2068	61	2439	85	77.11
	100	75	1792.08	0.0%	103	21	396	78	4.70
kroB100	25	1	888.39	0.0%	50	21	0	0	0.41
	25	6	855.43	0.0%	10	14	28	0	0.08
	25	12	885.19	0.0%	392	10	40	0	0.30
	25	18	913.62	0.0%	224	7	43	0	0.28
	50	1	1150.01	0.0%	288	63	0	0	4.74
	50	12	1127.02	0.0%	17	49	408	0	5.40
	50	25	1254.80	0.0%	0	21	178	0	0.32
	50	37	1303.86	0.0%	4	10	160	0	0.22
	75	1	1366.71	0.0%	891	113	0	5	24.16
	75	18	1374.22	0.0%	228	80	682	10	12.49
	75	37	1447.59	0.0%	365	36	202	0	3.56
	75	56	1385.03	0.0%	1256	19	153	0	10.04
	100	1	1629.90	0.0%	1264	179	0	18	162.66
	100	25	1596.72	0.0%	1359	97	814	31	42.60
	100	50	1604.51	0.0%	851	54	625	27	11.80
	100	75	1621.05	0.0%	3559	50	3796	2197	287.19
kroC100	25	1	838.74	0.0%	44	11	0	8	0.15
	25	6	847.79	0.0%	56	4	4	0	0.04
	25	12	737.32	0.0%	101	11	27	12	0.11
	25	18	778.97	0.0%	35	6	30	0	0.06
	50	1	1058.67	0.0%	130	95	0	4	6.56
	50	12	897.86	0.0%	908	49	338	246	8.77
	50	25	993.93	0.0%	3	22	85	7	0.18
	50	37	1113.33	0.0%	392	14	110	23	1.75
	75	1	1301.35	0.0%	183	153	0	0	20.47
	75	18	1054.04	0.0%	327	67	703	27	23.48
	75	37	1205.22	0.0%	378	38	1001	1379	16.24
	75	56	1568.47	0.0%	101	14	337	27	2.26
	100	1	1488.69	0.0%	511	213	0	194	91.51
	100	25	1196.24	0.0%	600	117	988	44	108.49
	100	50	1535.96	0.0%	1099	54	1210	190	27.93
	100	75	1945.74	0.0%	751	22	416	4	16.28

Table 2: Computational results for OPVP2-CV-L

Instance	$\lvert V \rvert$	$\lvert T \rvert$	**Objective value**	**Final gap**	**B&C nodes**	**Arc-vertex**	**Strong conn.**	**Strong 2-match.**	**Lin. cons.**	**CPU time (sec)**
kroA100	25	1	512.38	0.0%	41	40	0	0	116	0.40
	25	6	486.84	0.0%	12	31	34	0	69	0.18
	25	12	476.30	0.0%	16	12	51	0	71	0.12
	25	18	514.46	0.0%	2	9	47	0	64	0.05
	50	1	659.54	0.0%	1925	261	0	4	202	110.75
	50	12	613.02	0.0%	195	85	202	1	143	3.28
	50	25	651.87	0.0%	57	47	229	21	122	1.31
	50	37	627.16	0.0%	16	13	193	2	116	0.52
	75	1	736.68	0.0%	2456	410	0	10	279	530.68
	75	18	672.47	0.0%	629	175	997	109	183	42.22
	75	37	661.40	0.0%	117	65	533	120	150	7.32
	75	56	695.19	0.0%	7	23	462	48	131	3.30
	100	1	800.52	0.0%	5423	634	0	78	380	2842.79
	100	25	709.74	0.0%	2896	240	1692	663	213	333.19
	100	50	674.53	0.0%	295	113	1361	96	175	40.18
	100	75	799.25	0.0%	35	27	393	45	168	5.72
kroB100	25	1	533.22	0.0%	186	60	0	0	118	0.91
	25	6	473.56	0.0%	162	39	61	0	101	0.55
	25	12	485.27	0.0%	49	23	71	1	104	0.26
	25	18	487.99	0.0%	8	8	64	0	70	0.10
	50	1	641.43	0.0%	4063	201	0	72	213	102.32
	50	12	603.11	0.0%	547	107	292	26	167	8.93
	50	25	594.71	0.0%	75	34	221	20	130	1.65
	50	37	592.11	0.0%	13	17	170	8	111	0.55
	75	1	764.11	0.0%	10102	407	0	38	305	2554.16
	75	18	674.92	0.0%	608	175	585	56	177	33.36
	75	37	670.18	0.0%	59	69	425	16	169	4.48
	75	56	576.29	0.0%	1	23	231	0	108	1.00
	100	1	886.32	8.8%	4456	797	0	0	377	7200.00
	100	25	751.41	0.0%	2710	267	2027	229	230	304.66
	100	50	703.60	0.0%	204	100	1038	137	182	28.21
	100	75	649.29	0.0%	516	52	1478	499	130	69.67
kroC100	25	1	514.79	0.0%	272	65	0	12	108	1.32
	25	6	503.02	0.0%	28	31	21	0	79	0.18
	25	12	391.55	0.0%	10	21	52	2	68	0.12
	25	18	406.00	0.0%	2	7	27	0	61	0.05
	50	1	650.54	0.0%	594	191	0	23	207	24.20
	50	12	452.25	0.0%	326	92	188	141	125	4.63
	50	25	477.95	0.0%	42	39	189	8	119	0.97
	50	37	512.94	0.0%	23	15	206	50	100	0.81
	75	1	746.38	0.0%	4615	459	0	42	288	1280.76
	75	18	517.94	0.0%	962	149	476	66	190	33.97
	75	37	544.06	0.0%	647	74	744	171	155	28.93
	75	56	707.90	0.0%	44	19	301	70	143	3.73
	100	1	875.47	3.7%	4766	845	0	26	334	7200.00
	100	25	569.47	0.0%	4389	264	1676	173	233	390.10
	100	50	701.09	0.0%	356	85	927	271	193	39.71
	100	75	878.11	0.0%	41	36	490	153	189	8.73

Table 3: Computational results for OPVP2-CX-L2

Instance	$\lvert V\rvert$	$\lvert T\rvert$	**Objective value**	**Final gap**	**B&C nodes**	**Arc-vertex**	**Strong conn.**	**Strong 2-match.**	**CPU time (sec)**
kroA100	25	1	648.68	0.0%	153	66	0	11	9.56
	25	6	554.88	0.0%	329	54	120	0	6.24
	25	12	537.31	0.0%	123	19	92	0	0.81
	25	18	550.67	0.0%	148	6	92	0	0.50
	50	1	796.06	0.0%	776	156	0	371	593.19
	50	12	639.34	0.0%	1291	88	704	15	181.97
	50	25	589.28	0.0%	3535	44	907	47	422.52
	50	37	501.79	0.0%	1406	14	445	17	25.66
	75	1	866.62	0.4%	1499	292	0	1140	7200.00
	75	18	651.18	0.0%	2929	147	3361	132	3549.89
	75	37	532.94	0.0%	4049	56	2120	48	571.26
	75	56	491.83	0.0%	612	28	1734	150	142.82
	100	1	880.96	8.0%	1051	323	0	2753	7200.00
	100	25	661.48	0.0%	2152	161	5053	141	3320.91
	100	50	496.75	0.0%	1333	95	6904	253	1871.49
	100	75	549.48	0.0%	681	28	1244	163	149.50
kroB100	25	1	680.00	0.0%	409	75	0	0	15.41
	25	6	576.43	0.0%	298	34	82	0	3.82
	25	12	558.21	0.0%	451	30	138	0	2.32
	25	18	521.96	0.0%	232	6	80	0	0.76
	50	1	836.00	0.0%	601	170	0	226	449.84
	50	12	638.60	0.0%	1308	82	966	18	346.95
	50	25	567.89	0.0%	971	42	865	24	84.52
	50	37	522.49	0.0%	500	14	488	12	9.74
	75	1	949.00	0.0%	1413	279	0	1412	3632.80
	75	18	660.69	0.0%	1845	149	2767	44	2337.43
	75	37	576.05	0.0%	1827	76	2205	147	394.09
	75	56	351.53	0.0%	1004	24	1414	157	128.14
	100	1	1043.54	1.7%	1146	238	0	4285	7200.00
	100	25	671.00	0.0%	1806	190	6094	95	4101.94
	100	50	480.10	0.0%	1954	87	5670	379	2049.92
	100	75	341.75	0.0%	1620	49	5895	1783	1312.31
kroC100	25	1	724.00	0.0%	266	65	0	19	19.59
	25	6	699.27	0.0%	274	45	54	5	2.94
	25	12	486.53	0.0%	307	18	121	10	2.65
	25	18	501.47	0.0%	143	7	86	0	0.50
	50	1	806.27	0.0%	461	140	0	133	128.28
	50	12	512.00	0.0%	683	68	628	40	81.76
	50	25	478.32	0.0%	480	29	513	2	26.72
	50	37	437.05	0.0%	742	18	375	83	16.36
	75	1	862.14	7.2%	1273	366	0	3229	7200.00
	75	18	539.58	0.0%	1301	114	2308	48	791.37
	75	37	450.51	0.0%	2469	80	1319	189	401.83
	75	56	550.11	0.0%	752	26	1144	68	102.04
	100	1	924.20	5.4%	1135	393	0	3356	7200.00
	100	25	511.50	0.0%	1288	127	3702	267	2043.80
	100	50	523.38	0.0%	1572	56	2387	102	423.53
	100	75	612.52	0.0%	774	19	1132	28	78.88

Table 4: Computational results for OPVP1 on larger instances

Instance	$\lvert V\rvert$	$\lvert T\rvert$	**Objective value**	**Final gap**	**B&C nodes**	**Arc-vertex**	**Strong conn.**	**Strong 2-match.**	**CPU time (sec)**
kroA200	125	1	1589.89	0.0%	1	281	0	0	207.12
	125	31	1342.30	0.0%	696	124	1096	84	52.34
	125	62	1846.14	0.0%	233	65	926	5	9.55
	125	93	2225.99	0.0%	249	37	949	803	32.45
	150	1	1645.89	0.0%	1064	432	0	401	2623.81
	150	37	1456.86	0.0%	359	157	2364	421	312.02
	150	75	2096.94	0.0%	211	75	1137	11	16.61
	150	112	2726.65	0.0%	867	39	1994	1277	71.71
	175	1	1757.80	0.0%	2929	502	0	172	5028.12
	175	43	1544.80	0.0%	753	224	5111	321	1021.76
	175	87	2157.93	0.0%	377	114	4532	1080	273.61
	175	131	2767.03	2.3%	883	98	36831	10517	7200.00
	200	1	1820.88	0.0%	1120	594	0	40	4026.56
	200	50	1708.65	0.0%	1089	196	3599	162	353.84
	200	100	2522.27	0.0%	889	138	7944	3340	848.29
	200	150	3262.74	0.0%	864	88	21122	7554	3191.51
kroB200	125	1	1703.64	0.0%	2070	275	0	115	646.40
	125	31	1595.58	0.0%	791	148	2156	23	354.47
	125	62	1540.96	0.0%	1174	108	4087	290	332.15
	125	93	2005.65	0.0%	732	60	7102	1775	390.54
	150	1	1811.37	0.0%	2309	328	0	40	1157.18
	150	37	1713.08	0.0%	42	123	1848	1	178.51
	150	75	1652.63	0.0%	1758	140	16618	5873	3995.93
	150	112	2424.15	0.0%	289	54	5296	755	317.23
	175	1	1835.60	0.0%	541	416	0	106	1271.23
	175	43	1698.13	0.0%	577	181	2399	9	603.47
	175	87	1768.38	3.6%	461	132	29810	6411	7200.00
	175	131	2760.51	0.0%	6717	60	15370	5836	3668.72
	200	1	1911.83	0.0%	82	453	0	0	1646.82
	200	50	1733.39	0.0%	594	188	1640	5	106.93
	200	100	N/A	N/A	0	115	9449	381	7200.00
	200	150	3146.84	0.0%	1566	79	16126	10419	2489.42

Table 5: Evaluation of the computational results

Instance	$\|V\|$	$\|T\|$	OPVP1 Vertices visited	OPVP1 Multiple passes	OPVP1 Avg. passes	OPVP1 Max. passes	OPVP-CV-L Vertices visited	OPVP-CV-L Time limit	OPVP-CV-L Min. time spent	OPVP-CV-L Avg. time spent	OPVP-CV-L Max. time spent	OPVP2-CX-L2 Vertices visited	OPVP2-CX-L2 Min. visits	OPVP2-CX-L2 Inter. visits	OPVP2-CX-L2 Max. visits
kroA100	25	1	18	16	3.9	6	12	14212.0	462.1	859.0	1388.6	12	0	1	11
	25	6	18	15	3.7	5	12	15118.0	155.8	513.4	753.6	9	0	1	8
	25	12	21	17	3.5	5	16	16104.0	61.0	371.7	642.0	13	5	1	7
	25	18	23	19	3.5	5	19	17162.0	57.0	343.9	689.3	18	10	1	7
	50	1	21	19	3.3	5	16	14617.0	209.8	514.7	717.9	12	1	1	10
	50	12	27	20	2.9	4	21	16563.0	63.0	310.0	542.5	17	9	1	7
	50	25	36	23	2.7	4	32	18955.0	59.0	215.1	480.2	28	21	1	6
	50	37	43	21	2.5	4	40	21059.0	59.0	172.2	480.2	38	32	1	5
	75	1	26	25	3.0	5	18	14702.0	187.8	502.0	730.5	14	2	1	11
	75	18	34	23	2.6	4	27	17754.0	59.0	253.0	495.2	21	13	1	7
	75	37	49	20	2.2	4	43	21180.0	59.0	161.8	425.0	39	33	1	5
	75	56	60	21	2.2	3	58	24388.0	59.0	126.2	354.4	57	51	1	5
	100	1	29	28	2.8	5	20	14702.0	71.0	474.3	693.3	16	5	1	10
	100	25	45	25	2.4	4	37	19066.0	59.0	192.2	425.0	30	23	1	6
	100	50	58	19	2.1	3	52	23362.0	59.0	134.8	357.9	52	47	1	4
	100	75	80	22	2.1	3	76	27522.0	59.0	116.9	357.9	76	72	1	3
kroB100	25	1	17	15	4.1	6	13	13435.0	340.7	528.6	727.4	10	0	1	9
	25	6	18	16	3.6	6	13	14169.0	58.0	415.5	631.8	9	1	1	7
	25	12	20	17	3.6	6	15	15065.0	58.0	396.8	707.7	14	5	1	8
	25	18	22	19	3.2	5	20	16019.0	58.0	264.8	516.9	19	12	1	6
	50	1	23	22	3.1	4	17	14102.0	247.2	414.8	626.0	13	0	1	12
	50	12	26	22	3.0	4	21	15812.0	61.0	308.2	561.4	16	8	1	7
	50	25	35	23	2.7	4	32	17994.0	61.0	186.4	483.6	27	20	1	6
	50	37	41	24	2.4	4	38	19940.0	61.0	162.2	369.2	38	32	1	5
	75	1	27	26	2.9	4	18	14550.0	388.6	536.2	717.9	14	0	1	13
	75	18	36	25	2.5	3	28	17344.0	63.0	230.6	429.0	23	15	1	7
	75	37	46	26	2.3	3	41	20566.0	63.0	165.0	372.2	38	32	1	5
	75	56	58	13	2.0	2	57	23648.0	63.0	102.1	245.0	56	53	1	2
	100	1	33	31	2.8	4	20	14782.0	327.9	504.5	677.7	15	2	1	12
	100	25	45	26	2.1	3	36	18854.0	64.0	186.4	381.3	29	21	1	7
	100	50	59	17	2.0	2	56	23128.0	64.0	117.1	341.6	50	44	1	5
	100	75	78	6	2.0	2	77	27310.0	60.0	91.5	199.3	75	72	1	2
kroC100	25	1	17	16	4.9	8	11	14040.0	381.9	789.7	1250.3	11	0	1	10
	25	6	17	16	4.6	8	13	14800.0	131.3	521.7	806.7	12	2	1	9
	25	12	18	13	3.5	5	16	15748.0	64.0	303.9	610.1	14	7	1	6
	25	18	21	14	3.6	5	19	16578.0	57.0	256.3	647.7	18	11	1	6
	50	1	24	23	3.6	6	16	14372.0	161.9	584.8	1031.1	12	1	1	10
	50	12	26	17	2.6	4	20	16124.0	66.0	228.8	465.7	15	9	1	5
	50	25	34	15	2.4	3	30	18154.0	58.0	169.4	465.7	25	20	1	4
	50	37	41	17	2.4	3	39	20088.0	58.0	141.9	465.7	37	32	1	4
	75	1	29	26	3.2	5	18	14480.0	101.0	474.6	629.3	14	3	1	10
	75	18	34	19	2.3	3	27	17098.0	58.0	197.1	440.2	19	12	1	6
	75	37	46	17	2.2	3	42	20238.0	58.0	133.7	341.4	39	33	1	5
	75	56	62	22	2.2	3	59	23500.0	58.0	126.8	428.5	56	49	1	6
	100	1	29	27	2.9	4	19	14690.0	275.9	493.2	662.9	15	3	1	11
	100	25	39	18	2.2	3	37	18554.0	59.0	143.9	342.4	26	19	1	6
	100	50	62	16	2.1	3	57	22758.0	59.0	121.4	342.4	51	45	1	5
	100	75	81	21	2.1	3	80	27362.0	59.0	113.3	342.4	75	68	1	6