A Branch-and-Price Algorithm for Combined Location and Routing Problems Under Capacity Restrictions

Z. Akca [*] R.T. Berger [†] T.K Ralphs [‡]

September 17, 2008

Abstract

We investigate the problem of simultaneously determining the location of facilities and the design of vehicle routes to serve customer demands under vehicle and facility capacity restrictions. We present a set-partitioning-based formulation of the problem and study the relationship between this formulation and the graph-based formulations that have been used in previous studies of this problem. We describe a branch-and-price algorithm based on the set-partitioning formulation and discuss computational experience with both exact and heuristic variants of this algorithm.

1 Introduction

The design of a distribution system begins with the questions of where to locate the facilities and how to allocate customers to the selected facilities. These questions can be answered using location-allocation models, which are based on the assumption that customers are served individually on out-and-back routes. However, when customers have demands that are less-than-truckload and thus can receive service from routes making multiple stops, the assumption of individual routes will not accurately capture the transportation cost. Therefore, the integration of location-allocation and routing decisions may yield more accurate and cost-effective solutions.

In this paper, we investigate the so-called *location and routing problem* (LRP). Given a set of candidate facility locations and a set of customer locations, the objective of the LRP is to determine the number and location of facilities and construct a set of vehicle routes from facilities to customers in such a way as to minimize total system cost. The system cost may include both the fixed and operating costs of both facilities and vehicles. In this study, we consider an LRP with capacity constraints on both the facilities and the vehicles.

Vehicle routing models in general have been the subject of a wide range of academic papers, but the number of these devoted to combined location and routing models is much smaller. Laporte (1988) surveyed the work on deterministic LRPs and described different formulations of the problem, solution methods used, and the computational results published up to 1988. Min et al (1998) classified both deterministic and stochastic models in the LRP literature with respect to problem characteristics and solution methods.

[*]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, zea2@lehigh.edu

[†]100 Keyes Road #104, Concord, MA 01742, rosemary.berger@verizon.net

[‡]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, ted@lehigh.edu, http://coral.ie.lehigh.edu/~ted

Solution approaches for the LRP can be divided broadly into heuristics and exact methods, with much of the literature devoted to heuristic approaches. Most heuristic algorithms divide the problem into subproblems and handle these subproblems sequentially or iteratively. Examples of heuristics developed for the LRP can be found in Perl and Daskin (1985), Hansen et al (1994) and Barreto et al (2007).

Many fewer papers have been devoted to the study of exact algorithms for the LRP and most of these have been based on the so-called two-index vehicle flow formulation. Laporte and Nobert (1981) solved a single depot model with a constraint relaxation method and a branch-and-bound algorithm. They reported solving problems with 50 customer locations. Laporte et al (1983) solved a multi-depot model using a constraint relaxation method and Gomory cutting planes to satisfy integrality. They were able to solve problems with at most 40 customer sites. Laporte et al (1986) applied a branch-and-cut algorithm to a multi-depot LRP model with vehicle capacity constraints. They used subtour elimination constraints and chain barring constraints that guarantee that each route starts and ends at the same facility. They reported computational results for problems with 20 customer locations and 8 depots. Finally, Belenguer et al (2006) provided a two-index formulation of the LRP with capacitated facilities and capacitated vehicles and presented a set of valid inequalities for the problem. They developed two branch-and-cut algorithms based on different formulations of the problem. They reported that they could solve instances with up to 32 customers to optimality in less than 70 CPU seconds and could provide good lower bounds for the rest of the instances, which had up to 134 customers.

Berger (1997) developed a set-partitioning-based model for a special case of the LRP with route length constraints and uncapacitated vehicles and facilities. Berger et al (2007) extended that work to develop an exact branch-and-price algorithm in which they solved the pricing problem as an elementary shortest path problem with one resource constraint. They reported computational results for problems with 100 customers and various distance constraints. Akca et al (2008) developed an exact solution algorithm using branch-and-price methodology for the integrated location routing and scheduling problem (LRSP), which is a generalization of the LRP. In the LRSP, the assumption that each vehicle covers exactly one route is removed and the decision of assigning routes to vehicles subject to the scheduling constraints is considered in conjunction with the location and routing decisions. They considered instances with capacitated facilities and time- and capacity-limited vehicles. They provided solutions for instances with up to 40 customers.

In this study, we utilize a variant of the model presented by Akca et al (2008) to solve the LRP under capacity restrictions and modify their exact algorithm for the LRSP to solve the LRP. We develop a number of variants and heuristic extensions of the basic algorithm and report on our computational experience solving both randomly generated instances and instances from the literature. The remainder of the paper is organized as follows. Section 2 presents a formal description of the problem, provides two formulations, and investigates the relationship between the formulations. Section 3 describes details of the heuristic and the exact algorithms for the set-partitioning formulation. Section 4 provides some computational results evaluating the performance of the algorithms. Section 5 concludes the paper.

2 Problem Definition and Formulations

The objective of the LRP is to select a subset of facilities and construct an associated set of vehicle routes serving customers at minimum total cost, where the cost includes the fixed and operating costs of both facilities and vehicles. The constraints of the problem are as follows: (i) each customer must be serviced by exactly one vehicle, (ii) each vehicle must be assigned to exactly one facility at which it must start and end its route, (iii) the total demand of the customers assigned to a route must not exceed the vehicle capacity, and (iv) the total demand of the customers assigned to a facility must not exceed the capacity of the facility. In the literature, most of the exact methods developed for the described LRP or its special cases are based on the two-index vehicle flow formulation of the problem. To the best of our knowledge, an exact solution algorithm based on a set-partitioning formulation has not been applied to the case of the LRP with capacity constraints. The theoretical relationship between the two-index formulation and the set-partitioning formulation can be understood by considering a closely related three-index formulation that we present below. We show that applying Dantzig-Wolfe decomposition to the three-index formulation yields the set-partitioning formulation. This is turn shows that the bounds yielded by the LP relaxation of the set-partitioning model must be at least as tight as those of the three-index formulation.

2.1 Vehicle Flow Formulation

We let I denote the set of customers, J denote the set of candidate facilities, and $V = I \cup J$. To bypass the decision of assigning vehicles to facilities, we assume that each facility has its own set of vehicles and that a vehicle located at facility j can only visit customer locations and the facility j during its trip. Let H_j be the set of vehicles located at facility j, $\forall j \in J$ and H be the set of all vehicles, $H = \bigcup_{j \in J} H_j$. We define the following parameters and decision variables:

Parameters:

$$\begin{aligned} D_i &= \text{demand of customer } i, \forall i \in I, \\ C_j^F &= \text{capacity of facility } j, \forall j \in J, \\ C^V &= \text{capacity of a vehicle}, \\ F_j &= \text{fixed cost of opening facility } j, \forall j \in J, \\ O_{ik} &= \text{operating cost of traveling arc } (i,k)\ \forall i,k \in V. \end{aligned}$$

Decision Variables:

$$x_{ikh} = \begin{cases} 1 & \text{if vehicle } h \text{ travels directly from location } i \text{ to location } k, \forall i \in V, k \in V, h \in H \\ 0 & \text{otherwise}, \end{cases}$$

$$y_{ih} = \begin{cases} 1 & \text{if vehicle } h \text{ visits customer } i, \forall i \in I, h \in H \\ 0 & \text{otherwise}, \end{cases}$$

$$t_j = \begin{cases} 1 & \text{if facility } j \text{ is selected to be open}, \forall j \in J \\ 0 & \text{otherwise}. \end{cases}$$

A vehicle flow formulation of the LRP is as follows:

$$
\begin{aligned}
\textbf{(VF-LRP)}\text{ Minimize} \quad & \sum_{j\in J} F_j t_j + \sum_{h\in H}\sum_{i\in V}\sum_{k\in V} O_{ik} x_{ikh} && && (1)\\
\text{s.t.} \quad \sum_{h\in H}\sum_{k\in V} x_{ikh} &= 1 && \forall i \in I, && (2)\\
\sum_{k\in V} x_{ikh} - \sum_{k\in V} x_{kih} &= 0 && \forall i \in V, h \in H, && (3)\\
\sum_{k\in V} x_{kih} - y_{ih} &= 0 && \forall i \in I, h \in H, && (4)\\
y_{ih} - \sum_{k\in S}\sum_{l\in V\setminus S} x_{klh} &\le 0 && \forall S \subseteq I, i \in S, h \in H, && (5)\\
\sum_{i\in I} D_i y_{ih} - C^V &\le 0 && \forall h \in H, && (6)\\
\sum_{h\in H_j}\sum_{i\in I} D_i y_{ih} - C^F_j t_j &\le 0 && \forall j \in J, && (7)\\
x_{ikh} &\in \{0,1\} && \forall i,k \in V, h \in H, && (8)\\
y_{ih} &\in \{0,1\} && \forall i \in I, h \in H, && (9)\\
t_j &\in \{0,1\} && \forall j \in J. && (10)
\end{aligned}
$$

In (VF-LRP), the objective function (1) seeks to minimize the total cost, which includes the fixed cost of the selected facilities and the operating cost of the vehicles. Vehicle fixed costs can easily be incorporated into the model by increasing the cost of traveling from the facility to each customer location by a fixed amount. Constraints (2) specify that exactly one vehicle must service customer i. Constraints (3) require that each vehicle should enter and leave a location the same number of times. Constraints (4) determine the assignment of customers to vehicles. Constraints (5) eliminate subtours, i.e. routes that do not include a facility. Constraints (6) are vehicle capacity constraints. Constraints (7) ensure that the capacity of a facility is not exceeded by demand flows to customer locations. For notational convenience, we assume that variables associated with travel between different facilities or travel between a customer and a facility using a truck not associated with that facility are fixed to zero. Constraints (8), (9), and (10) are the set of binary restrictions on the variables.

2.2 Set-Partitioning Formulation

Here we utilize a modified version of the set-partitioning formulation for the LRSP presented by Akca et al (2008). We define the set P to be a set indexing all vehicle routes feasible with respect to vehicle capacity and originating from and returning to the same facility. We let $P_j \subseteq P$ index routes associated with vehicles assigned to facility j for $j \in J$. In addition to the parameters and sets defined for (VF-LRP), we use the following parameters and decision variables:

Parameters:

$$
\begin{aligned}
O_p &= \text{operating cost of route } p \in P,\\
a_{ip} &= \begin{cases} 1 & \text{if customer } i \in I \text{ is assigned to route } p \in P, \text{ and} \\ 0 & \text{otherwise}. \end{cases}
\end{aligned}
$$

Decision Variables:

$$z_p = \begin{cases} 1 & \text{if route } p \in P \text{ is selected, and} \\ 0 & \text{otherwise,} \end{cases}$$

$$t_j = \begin{cases} 1 & \text{if facility } j \in J \text{ is selected to be open, and} \\ 0 & \text{otherwise,} \end{cases}$$

The formulation is as follows:

$$\begin{aligned} \textbf{(SPP)}\ \text{Minimize} \quad & \sum_{j\in J} F_j t_j + \sum_{p\in P} O_p z_p && && (11) \\ \text{subject to} \quad & \sum_{p\in P} a_{ip} z_p = 1 && \forall i \in I, && (12) \\ & \sum_{p\in P_j} \sum_{i\in I} D_i a_{ip} z_p - C_j^F t_j \le 0 && \forall j \in J, && (13) \\ & z_p \in \{0,1\} && \forall p \in P, && (14) \\ & t_j \in \{0,1\} && \forall j \in J. && (15) \end{aligned}$$

The objective function (11) seeks to minimize the total cost, which includes the fixed cost of the selected facilities and the operating cost of the vehicles. Constraints (12) guarantee that each customer location is served by exactly one route. Constraints (13) ensure that the total demand of the selected routes for a facility does not exceed the facility capacity. Constraints (14) and (15) are standard binary restrictions on the variables.

2.3 Comparing the Formulations

Observe that the three-index formulation (VF-LRP) exhibits a high degree of symmetry under the assumption that the vehicle fleet assigned to each facility is homogeneous. This is due to the fact that the assignment of routes to a specific vehicle is essentially arbitrary, i.e., the cost of a given solution to (VF-LRP) is invariant under permutation of the indices assigned to specific vehicles. This symmetry can be dealt with either (i) by using a two-index formulation, which requires the addition of an exponential number of valid inequalities to the formulation or (ii) by applying Dantzig-Wolfe decomposition (DWD). Laporte et al (1986) and Belenguer et al (2006) have each developed branch-and-cut algorithms using the former approach. Here, we explore the latter approach.

As is standard in DWD, we decompose the constraints of (VF-LRP) into two subsystems. The *master problem* is defined by constraints (2) and (7), while the *subproblem* is defined by constraints (3), (4), (5), (6), (8), (9), and (10). All constraints of the subproblem are indexed by the set J and it is therefore immediate that the subproblem decomposes by facility. The objective of each of the resulting single-facility subproblems is to generate least-cost routes for all vehicles assigned to the facility, though without the constraint that customers appear on exactly one route. For each candidate facility $j \in J$, the set of integer solutions of the decomposed subproblem can be represented by the set of vectors

$$\{(x,y,t) \in \mathbb{Z}^{(|V|\times|V|\times|H_j|)\times(|V|\times|H_j|)\times\{0,1\}} \mid \quad (x,y,t) \text{ satisfies } (3)_j,(4)_j,(5)_j,(6)_j,(8)_j,(9)_j,(10)_j\},$$

which is described only by those constraints associated with facility j and specifies values only for those variables associated with vehicles assigned to facility j. Hence, the index j for the constraints represents the set of constraints associated with facility j. Let E be a set indexing the members of all of the above sets, with E_j the indices for vectors associated with facility j only, so that $E = \cup_{j \in J} E_j$. For a facility j and an index $q \in E_j$, the corresponding member $(\mathrm{x}^q, \mathrm{y}^q, \mathrm{t}^q)$ of the above set is then a vector with the following interpretation: for each pair $(i,k) \in |V| \times |V|$ and $h \in H_j$, the parameter $\mathrm{x}^q_{ikh} = 1$ if vehicle h travels on arc (i,k) in solution q and is 0 otherwise; for each $i \in I$ and $h \in H_j$, $\mathrm{y}^q_{ih} = 1$ if customer i is visited by vehicle h in solution q and is 0 otherwise; and $\mathrm{t}^q = 1$ if facility j is open in solution q. Note that the variable t indicating whether the facility is open does not appear in any of the linear constraints of the subproblem and can hence be set to either 0 or 1 without affecting feasibility.

Because the subproblem decomposes as described above, solutions to the original problem can be seen as vectors obtained by "recomposing" convex combinations of the members of E_j for each $j \in J$. In other words, any solution (x, y, t) to the LP relaxation of the original problem can be written as:

$$x_{ikh} = \sum_{q \in E} \mathrm{x}^q_{ikh} \theta_q \quad \forall i, k \in V, h \in H, \tag{16}$$

$$y_{ih} = \sum_{q \in E} \mathrm{y}^q_{ih} \theta_q \quad \forall i \in I, h \in H, \tag{17}$$

$$t_j = \sum_{q \in E_j} \mathrm{t}^q \theta_q \quad \forall j \in J, \tag{18}$$

$$\sum_{q \in E_j} \theta_q = 1 \quad \forall j \in J, \tag{19}$$

$$\theta_q \geq 0 \quad \forall q \in E. \tag{20}$$

Using (16) - (20), we can then formulate the LP relaxation of the master problem as:

$$\textbf{(MDW)} \text{ Minimize } \sum_{q \in E} \tilde{C}_q \theta_q$$

$$\text{s.t.} \quad \sum_{q \in E} b_{iq} \theta_q = 1 \quad \forall i \in I, \tag{21}$$

$$\sum_{q \in E_j} \sum_{i \in I} b_{iq} D_i \theta_q - C^F_j \sum_{q \in E_j} \theta_q \mathrm{t}^q \leq 0 \quad \forall j \in J, \tag{22}$$

$$\sum_{q \in E_j} \theta_q = 1 \quad \forall j \in J, \tag{23}$$

$$\theta_q \geq 0 \quad \forall q \in E, \tag{24}$$

where

$$b_{iq} = \sum_{h \in H} \sum_{k \in I} \mathrm{x}^q_{ikh} = \sum_{h \in H} \mathrm{y}^q_{ih}, \quad \forall i \in I, q \in E,$$

$$\tilde{C}_q = F_q \mathrm{t}^q + \sum_{i \in V} \sum_{k \in V} \sum_{h \in H} O_{ik} \mathrm{x}^q_{ikh}, \quad \forall q \in E,$$

where $F_q = F_j$ when $q \in E_j$ for $j \in J$. Here, b_{iq} can be interpreted as the number of times customer i is visited in solution q, and $\tilde{C}_q$ is the cost of solution q (including facility fixed cost) for all $q \in E$.

The similar forms of (SPP) and (MDW) should now be evident, but to rigorously show their equivalence, we need to dissect the relationship between set E_j and P_j for a given facility $j \in J$. A member of set E_j consists of a collection of routes assigned to vehicles located at facility j. A member of set P_j, on the other hand, is a single route that can be assigned to any vehicle at facility j. Therefore, a member of E_j can be constructed by associating at most $|H_j|$ members of set P_j and some number of empty routes (zero vectors representing vehicles that are not used) to the vehicles assigned to facility j.

Now, by utilizing the integrality requirements from the original problem and carefully eliminating the indices of symmetric solutions from E, we get a much smaller set that we will show is in one-to-one correspondence with collections of members of P_j of cardinality at most $|H_j|$. We proceed as follows:

1. First, as the vehicles associated with a given facility j are identical, a set of routes from P_j can be assigned to vehicles in any arbitrary order and each route can also visit the customers in either clockwise or counterclockwise order. Hence, we obtain different members of E_j that are all equivalent from the standpoint of both feasibility and cost. To eliminate superfluous equivalent members of E_j, we divide the members of set E_j into equivalence classes, where two members of E_j are considered equivalent if the set of customers assigned to the facility and the partition of that set of customers defined by the routes are identical. It is clear that any two members of E_j that are equivalent by this definition will have exactly the same impact on both cost and feasibility. We then form equivalence classes from which all but one member may safely be eliminated from E_j.

2. Let $\hat{\theta}$ represent a solution to (MDW) for which the corresponding solution in the original space obtained by applying equations (16)-(18) is feasible for the original problem (VF-LRP). From (16) and (8), along with the fact that x^q_{ikh} is binary for $q \in E$, we get that $\mathrm{x}^q_{ikh} = 1$ whenever $\hat{\theta}_q > 0$ in the solution to (MDW). In other words, all members q of E_j with $\hat{\theta}_q > 0$ must correspond to routes visiting exactly the same customers in exactly the same order. Hence, we must in fact have $\hat{\theta}_q = 1$ for exactly one $q \in E_j$ for each $j \in J$ and so $\theta_q \in \{0,1\}$ for all $q \in E$.

3. It then follows easily that index q can be removed from E_j if there exist vehicles $h_1, h_2 \in H_j$ such that $\mathrm{x}^q_{ikh_1} = \mathrm{x}^q_{ikh_2}\ \forall i,k \in V$, where $\mathrm{x}^q_{ikh_1} > 0$ for some $i,k \in I$ (i.e., this is not an empty route). In this case, vehicles h_1 and h_2 define exactly the same set of routes, which means that $b_{iq} > 1$ for some $i \in I$. Because of constraint (21), such a solution must have $\hat{\theta}_q = 0$.

4. Finally, from (18) and (22), we can conclude that if $\hat{\theta}_q = 1$ and we have $\mathrm{x}^q_{ikh} > 0$ for some $i,k \in V$ and $h \in H$, then t^q must be 1. Hence, we can eliminate any solutions for which $\mathrm{t}^q = 0$ that does not correspond to a zero solution (i.e., closed facility). All of this allows us to rewrite (22) in the form

$$\sum_{q \in E_j} \sum_{i \in I} b_{iq} D_i \theta_q - C^F_j \sum_{q \in E_j} \theta_q \leq 0 \quad \forall j \in J. \tag{25}$$

If we restrict set E_j according to the rules described above and call the restricted set $\bar{E}_j$ for each $j \in J$, then we can finally conclude the following.

Proposition 2.1 *There is a one-to-one correspondence between subsets of P_j with cardinality less than $|H_j|$ and members of $\bar{E}_j$.*

The proof follows easily from the definition of sets P_j and $\bar{E}_j$ and the restriction rules. By replacing set E_j with $\bar{E}_j$ for all $j \in J$ in (MDW), as well as replacing (22) with (25) and adding the constraint $\theta_q \in \{0,1\}$ for all $q \in E$, we obtain a new (equivalent) formulation (MDW'). Then, we finally have the equivalence of (SPP) and (MDW) as follows

Proposition 2.2 *There is a one-to-one correspondence between solutions to (SPP) and solutions to (MDW') such that corresponding solutions also have the same objective function value. Thus, (SPP), (MDW'), and (MDW) are all equivalent.*

3 Solution Algorithm

3.1 Branch-and-Price

Having shown that (SPP) is equivalent to a DWD of (VF-LRP), we now discuss an exact solution algorithm based on a branch-and-price implementation utilizing the formulation (SPP). First, we strengthen the original formulation by adding the following additional valid constraints:

$$\sum_{p \in P_j} a_{ip} z_p - t_j \leq 0 \qquad \forall i \in I, \forall j \in J, \tag{26}$$

$$\sum_{j \in J} t_j \geq N^F, \tag{27}$$

$$\sum_{p \in P_j} z_p = v_j \qquad \forall j \in J, \tag{28}$$

$$\sum_{j \in J} v_j \geq \left\lceil \frac{\sum_{i \in I} D_i}{C^V} \right\rceil \tag{29}$$

$$v_j \geq t_j \qquad \forall j \in J, \tag{30}$$

$$v_j \in \mathbb{Z}^+ \qquad \forall j \in J, \tag{31}$$

where v_j represents the number of vehicles used at facility j, $\forall j \in J$ and N^F is a lower bound on the number of facilities that must be opened in any integer solution, calculated as follows:

$$N^F = \text{argmin}_{\{l=1,\ldots,|J|\}} \left(\sum_{t=1}^{l} C^F_{j_t} \geq \sum_{i \in I} D_i \right) \text{ s.t. } C^F_{j_1} \geq C^F_{j_2} \geq \ldots \geq C^F_{j_n}.$$

Constraints (26) force a facility to be open if any customer is assigned to it. Constraint (27) sets a lower bound on the total number of facilities required in any integer feasible solution. Constraints (26) (from Berger et al (2007) and Akca et al (2008)) and constraint (27) (from Akca et al (2008)) are shown computationally to improve the LP relaxation of the model. Constraints (28) are only added to facilitate branching on the integrality of the number of vehicles at each facility in the branch-and-price algorithm. Constraint (29) sets a lower bound for the total number of vehicles in the solution. Finally, constraints (30) force the number of vehicles used at a facility to be at least 1 if the facility is open. We refer to formulation (11)–(15) and (26)–(31) as (SP-LRP) in the rest of the paper.

The formulation (SP-LRP) contains a variable for each possible vehicle route originating from each facility. Hence, the number of routes will be too large to enumerate even for small instances. To solve the LP relaxation of models that contain exponentially many columns, column generation algorithms can be used. By solving a sequence of linear programs and dynamically generating columns eligible to enter the basis, such an algorithm implicitly considers all columns but generates only those that may improve the objective function. In order to generate integer solutions, a branch-and-bound approach is used in combination with the column generation and the overall approach is referred to as branch-and-price. Desrochers et al (1992), Vance et al (1994), Berger et al (2007), and Akca et al (2008) provide examples of branch-and-price algorithms from the literature.

Here, we modify the branch-and-price algorithm used in Akca et al (2008) to solve the LRSP. Therefore, some parts of the algorithm are described only briefly. For details, we refer the reader to Akca et al (2008). To initialize the algorithm, we construct a *restricted master problem* (RMP), that is, an LP relaxation of (SP-LRP) that contains all facility variables (t_j for $j \in J$) and vehicle variables (v_j for $j \in J$), but only a subset of the vehicle route variables (z_p for $p \in P$). The branch-and-price algorithm consists of two main components: an algorithm for solving the *pricing problem* or *column generation subproblem*, which is used to construct new columns in each iteration, and *branching rules*, which specify how to partition the feasible region into subsets to which the algorithm is then applied recursively until exhaustion.

At each iteration of the solution process for the LP relaxation of (SP-LRP), the objective of the column generation subproblem is to find a feasible vehicle route originating from each facility $j \in J$ with minimum reduced cost with respect to the current dual solution of the RMP. The reduced cost of a given route $p \in P$ is

$$\hat{C}_p = O_p - \sum_{i \in N} a_{ip}\pi_i + \sum_{i \in N} a_{ip} D_i \mu_j + \sum_{i \in N} a_{ip}\sigma_{ji} - \nu_j, \tag{32}$$

where π, μ, σ and ν are the dual variables associated with constraints (12), (13), (26), and (28), respectively, from the RMP. Hence, the column generation subproblem for facility $j \in J$ can be formulated as follows:

$$\begin{aligned}
\text{Minimize} \quad & \sum_{i \in M}\Big(\sum_{k \in I}(O_{ik} - \pi_k + D_k\mu_j + \sigma_{jk})x_{ik} + O_{ij}x_{ij}\Big) - \nu_j && (33)\\
\text{s.t.} \quad & y_j = 1, && (34)\\
& \sum_{k \in M} x_{ik} = \sum_{k \in M} x_{ki} = y_i \qquad \forall i \in I, && (35)\\
& y_i - \sum_{k \in S}\sum_{l \in V \setminus S} x_{kl} \le 0 \qquad \forall S \subseteq I, i \in S, && (36)\\
& \sum_{i \in I} D_i y_i - C^V \le 0, && (37)\\
& x_{ik} \in \{0,1\} \qquad \forall i,k \in M, && (38)\\
& y_i \in \{0,1\} \qquad \forall i \in M, && (39)
\end{aligned}$$

where $M = I \cup \{j\}$, $x_{ik} = 1$ if link (i,k) is used in solutions, and $y_i = 1$ if customer i is assigned to the route.

The column generation subproblem above can be seen as an instance of the well-known *elementary shortest path problem with resource constraints* (ESPPRC), which is well-studied and arises as a subproblem in many different routing applications. To cast the above subproblem as an ESPPRC, we consider a network with $|I|+2$ nodes, one node for each customer, one for the facility j as a source node and a copy of the facility j as a sink node. We assign each customer node a demand equal to its demand in the original problem and let the cost of each arc (i,l) in the network equal the coefficient of x_{il} in (33). A shortest path from source to sink visiting a customer at most once (called *elementary*) and satisfying the constraint that the total demand of customers included in the path does not exceed the vehicle capacity then corresponds to a vehicle route. The total cost of the path plus the fixed cost $-\nu_j$ is the reduced cost of the associated column.

To solve the ESPPRC, we use Feillet et al (2004)'s label setting algorithm with a single resource (the vehicle capacity). In the algorithm, each path from source to sink that is not dominated by another with respect to vehicle capacity, cost, and the set of nodes that can still be visited is explored. More details on the variants of this approach that were used in the computational experiments are given in Section 3.2 below.

When the pricing problem cannot identify any more columns with negative reduced cost, then the current solution to the LP relaxation of the master problem is optimal. If this optimal LP solution is not integral, then we are forced to branch. Devising good branching rules is an important step in developing a branch-and-price algorithm. Since the LP relaxations of the new nodes generated after branching are also solved using column generation, the branching constraints must be incorporated into the pricing problem and the columns to be generated must satisfy branching constraints for the node. Therefore, the specific branching rules employed may affect the structure of the pricing problem, causing it to become more difficult to solve. Here, we implement the same four branching rules we used for our work on the LRSP: branching on fractional variables t and v, forcing/forbidding the assignment of a specific customer to a specific facility, and forcing/forbidding flow on a single arc (originally used in Desrochers and Soumis (1989)). All branching rules can easily be incorporated into the pricing problem without changing the structure. Details on the effect of these rules on the pricing problem and the implementation of them can be found in Akca et al (2008).

3.2 Solving the Column Generation Subproblem

The ESPPRC is an NP-hard optimization problem, but small instances may generally be solved effectively in practice using dynamic programming-based labeling algorithms. In general, the number of labels that must be generated and evaluated in the label setting algorithm increases as either the number of customers or the vehicle capacity increases. To enhance efficiency, we therefore augment the basic scheme with some heuristic versions of the algorithm, since it is not necessary to find the column with the most negative reduced cost in every iteration. We refer to the exact pricing algorithm (that is guaranteed to produce a column with the smallest reduced cost) as ESPPRC. The following are heuristic versions of the exact algorithm and are not guaranteed to produce a column with negative reduced cost when one exists.

ESPPRC-LL(n). The ESPPRC algorithm keeps all non-dominated labels at each node. Depending on the size of the instance, the number of labels kept can become very large. In this heuristic version proposed by Dumitrescu (2002), we set a limit n on the number of unprocessed labels stored at each node. At each iteration, labels are sorted based on reduced cost and among the unprocessed labels, the n with smallest reduced cost are kept and the rest are permanently deleted. Therefore, the ESPPRC-LL(n) algorithm tends to terminate much more quickly than the ESPPRC for small values of n.

ESPPRC-CS(n). The ESPPRC algorithm is efficient for instances with small numbers of customers. In addition, the total number of customers in a route is restricted by the vehicle capacity. To be able to take advantage of this, we choose a subset of customers C_S with size n based on the cost of arcs in the network (coefficients of link variables in (33)). Since the objective of the pricing problem is to find a route with smallest reduced cost, we determine the n customers in the subset by constructing a minimum spanning tree over the customer locations. We stop the spanning tree algorithm when we have n customers in the tree. The first customer in the subset (and the tree) is chosen based on the cost of the links from source to customers. Then to find valid vehicle routes, we run ESPPRC that include only customers in C_S.

2-Cyc-SPPRC-PE(n). The shortest path problem with resource constraints (SPPRC) is a relaxation of the ESPPRC in which the path may visit some customers more than once. The SPPRC is solvable in pseudo-polynomial time, but use of this further relaxation of the column generation subproblem results in reduced bounds. Eliminating solutions containing cycles of length two strengthens this relaxation of the pricing problem and improves the bound (for details of the algorithm, see Irnich and Desaulniers (2005)). We refer to this pricing algorithm as 2-Cyc-SPPRC. In addition, we can also generate paths that are elementary with respect to a given subset of customers. We call the resulting algorithm 2-Cyc-SPPRC-PE(n), where n is the size of the customer set to be considered. At each iteration of the pricing problem, for each facility, the algorithm consists of the following steps:

1. Solve the 2-Cyc-SPPRC.
2. Consider the column with minimum cost and choose at most m customers that are visited more than once. Let C_{E1} be the set of these customers. If the set is empty, stop (the path is already elementary).
3. Solve 2-Cyc-SPPRC-PE(m) with set C_{E1} from step 2.
4. Pick at most m customers that are visited more than once. Let C_{E2} be the set of these customers. Let $C_{E3} = C_{E1} \cup C_{E2}$ and $m_3 = |C_{E3}|$. If the set is empty, let $C_{E3} = C_{E1}$ and stop.

5. Solve 2-Cyc-SPPRC-PE(m_3) with set C_{E3}.

In our experiments, we generally had the same set of customers C_{E3} in every step of the column generation. Thus, we decided to determine set C_{E3} for each facility at the first iteration of column generation at each node, and we use the same set for the rest of the iterations at the node.

For computational testing, we implemented four variants of the branch-and-price algorithm based on the above pricing schemes.

- *Heuristic Branch-and-Price* (HBP): The purpose of this algorithm is to provide a good upper bound. At each node of the tree, we use ESPPRC-CS(n) and ESPPRC-LL(m) with small values of *n* and *m* (*n* is chosen to be 12 to 15 depending on the average demand and the vehicle capacity, while *m* is chosen to be 5 or 10). In addition, we also used combinations of ESPPRC-CS(n) and ESPPRC-LL(m) for larger values of *n*. In the algorithm, we use an iteration limit for the number of pricing problems solved at any node of the tree. If the number of iterations exceeds the limit, we branch.

- *Elementary Exact Branch-and-Price* (EEBP): The purpose of this algorithm is to prove the optimality of the solution or provide an integrality gap. At each node of the tree, we use ESPPRC-CS(n), ESPPRC-LL(m) and ESPPRC.

- *2 Step Elementary Exact Branch-and-Price* (EEBP-2S): In this variant, HBP is run first to generate initial columns and an upper bound. Then, EEBP is initiated with the columns and the upper bound obtained from HBP.

- *Non-elementary Exact Branch-and-Price (NEBP)*: This is similar to elementary exact branch-and-price algorithm except that the pricing problem solved is 2-Cyc-SPPRC-PE(n).

4 Computational Results

In this section, we discuss the performance of our branch-and-price algorithm for the LRP on two sets of instances. The first set contains the LRP instances used in Barreto et al (2007) that are available from Barreto (2003). We used these instances to test the performance of our HBP and NEBP algorithms. The second set of instances were random instances we generated to test the performance of our EEBP and EEBP-2S algorithms. We evaluate the effect of facility capacity constraints and other parameters using this set of instances. For all of our experiments, we used a Linux-based workstation with a 1.8 GHz processor and 2GB RAM.

4.1 Instances From the Literature

To the best of our knowledge, there are no benchmark instances available specifically for the LRP. Barreto et al (2007) used the instances in the literature available for other types of problems to construct a set of LRP instances. They report lower bounds found by applying a branch-and-cut algorithm to the two-index formulation of the problem (Barreto, 2004) and upper bounds found by applying a sequential heuristic based on clustering techniques (Barreto et al, 2007). They listed 19 instances, three of which have more than 150 customers, too large for our approach to work efficiently. We removed these three instances plus one more with 117 customers and fractional demand, since we assume integer demands. The labels of the instances denote the source of the instance and the number of customers and facilities in the instances (for more details about the references, see Barreto (2003)).

We first ran HBP with a time limit of 3 CPU hours, focusing on producing quality upper bounds. Table 1 presents the instances we tested and compares the results with the upper bounds reported in Barreto et al (2007). Since neither our HBP nor Barreto et al (2007) can provide a valid lower bound for the problem, we used the best lower bounds as found in Barreto (2004) (second column in Table 1) to measure the quality of our upper bound. The "Gap" in Table 1 is the percent gap between the upper bound and the LB listed in the second column. HBP is capable of finding better upper bounds (usually optimal) for the instances of small and medium size. In these cases, the computation time is also very short. However, for larger instances, the upper bounds reported by Barreto et al (2007) are generally better. In addition, their heuristic algorithm is very efficient—they report that in most of the instances, it provides the result in less than one second.

Table 1: Performance of Heuristic Branch-and-Price

Instance	LB[1]	Barreto et al (2007)		Heuristic Branch-and-Price		
		UB[2]	Gap	UB	Gap	CPU(s)
Gaskell67-21x5	424.9*	435.9	2.59	424.9	0	1.2
Gaskell67-22x5	585.1*	591.5	1.09	585.1	0	41.5
Gaskell67-29x5	512.1*	512.1	0	512.1	0	67.7
Gaskell67-32x5	556.5	571.7	2.73	562.3	1.04	10801.8
Gaskell67-32x5-2	504.3*	511.4	1.41	505.8	0.3	85.6
Gaskell67-36x5	460.4*	470.7	2.24	460.4	0	1077.6
C69-50x5	549.4	582.7	6.06	565.6	2.95	239.4
C69-75x10	744.7	886.3	19.01	852.1	14.42	10802.3
C69-100x10	788.6	889.4	12.78	929.5	17.86	10836.6
Perl83-12x2	204*	204	0	204.0	0	0.2
Perl83-55x15	1074.8	1136.2	5.71	1121.8	4.37	10800.0
Perl83-85x7	1568.1	1656.9	5.66	1668.2	6.38	10813.8
Min92-27x5	3062*	3062	0	3062.0	0	4.2
Min92-134x8	-	6238	-	6421.6	-	10850.9
Dsk95-88x8	356.4	384.9	8	390.6	9.58	10808.9

[1] Reported by Barreto (2004), found using branch-and-cut
[2] Reported by Barreto et al (2007), found using a heuristic
* Branch-and-cut used in Barreto (2004) proves the optimality

Next, we used the EEBP-2S algorithm to test the ability to produce lower bounds and prove optimality. With this algorithm, we could not solve all of the instances within the total time limit of 5 CPU hours (3 CPU hours for HBP and 2 CPU hours for EEBP). The lower bounds found by our algorithm, along with the best integer solution found, optimality gap, and computation time are reported in Table 2. Note that because Gaskell67-21x5 and Perl83-12x2 are very easy to solve, we used the EEBP algorithm instead of the EEBP-2S algorithm in these cases.

Finally, for instances that we could not provide lower bounds by using the EEBP or the EEBP-2S algorithm, we used the NEBP algorithm with a time limit of 5 CPU hours or evaluated node limit of 50 nodes. The results are reported in Table 3. For the instances C69-100x10, Min92-134x8 and Dsk95-88x8, we could not solve even the root node within the time limit.

In general, the lower bounds found by using branch-and-cut (Barreto, 2004) are better than our lower bounds. However, in some cases, their computation times are much larger than our time limits. The HBP algorithm can provide good upper bounds, but for the medium and large size problems, we need to improve our lower bounding, perhaps by adding dynamic cut generation to our algorithm in order to close the optimality gap.

Table 2: Performance of 2 Step Elementary Exact Branch-and-Price

Instance	LB	OPT/BestIP	Gap	CPU(s)	Total CPU(s)
Gaskell67-21x5[1]	424.9	424.9	0	3.0	3.0
Gaskell67-22x5	585.1	585.1	0	2999.9	3041.4
Gaskell67-32x5	544.1	562.3	3.24%	8453.1	19254.9
Perl83-12x2[1]	204.0	204.0	0	0.2	0.2
Min92-27x5	3062.0	3062.0	0	833.6	837.8

[1] EEBP algorithm is used

Table 3: Performance of Non-elementary Exact Branch-and-Price

Instance	LB	OPT/BestIP	Gap	CPU(s)
Gaskell67-29x5	441.2	512.1	13.85%	14411.5
Gaskell67-32x5-2	494.4	505.8	2.26%	5654.8
Gaskell67-36x5	455.5	460.4	1.05%	100.1
C69-50x5	526.2	565.6	6.96%	2634.1
C69-75x10	693.5	852.1	18.62%	8785.8
Perl83-55x15	852.3	1121.8	24.02%	319.5
Perl83-85x7	1272.4	1668.2	23.73%	1379.6

4.2 Random Instances

On random instances, Laporte et al (1986) provided computational results for an exact method (branch-and-cut algorithm) for the capacitated LRP. Belenguer et al (2006) also developed a branch-and-cut algorithm for the capacitated LRP, but neither the details of the instances nor the computational results are publicly available. Therefore, we evaluated our algorithm by generating random instances as in Laporte et al (1986), where they generated instances with 10, 15 and 20 customers and 4 to 8 facilities. In addition to these, we generated instances with 30 and 40 customers and 5 facilities to test the performance of the algorithm on larger instances. The coordinates of the customers and the facilities and the demand of each customer were generated using a Uniform distribution on [0,100]. We then calculated the Euclidean distance between each pair of customers and between customers and facilities and rounded the calculated distance to two decimal places. Demand for each customer was rounded to the nearest integer. Vehicle capacity C^V was calculated as

$$C^V = (1-\alpha) max_{i\in I}\{D_i\} + \alpha \sum_{i\in I} D_i, \tag{40}$$

where α was a parameter and the values were chosen in set $\{0, 0.25, 0.5, 0.75, 1\}$.

4.2.1 Small and Medium Random Instances

Laporte et al (1986) solved location and routing problems with capacitated vehicles, but they did not have a facility capacity. Instead, they had a lower and upper bound for the total number of facilities that could be open in a solution. In this experiment, in order to provide a better comparison of our algorithm with that of Laporte et al (1986), we removed constraint (13) from the SP-LRP. We set N^F, the minimum number of open facilities in (27), to 1, and we added constraint $\sum_{j\in J} t_j \leq M^F$, where M^F be the maximum number of facilities that can be open in any solution. Facility and vehicle fixed costs were set to be zero. As in Laporte et al (1986), three groups of instances with different numbers of customers and facilities were available. For each group, five different vehicle capacities were calculated by changing α. Some details about the instances are listed in Table 4.

Table 4: Details for the instances

# of Customers	# of Facilities	N^F	M^F	# of instances	α
10	4	1	3	3	$\{0, 0.25, 0.5, 0.75, 1\}$
15	6	1	4	3	$\{0, 0.25, 0.5, 0.75, 1\}$
20	8	1	5	3	$\{0, 0.25, 0.5, 0.75, 1\}$

Tables 5, 6 and 7 present the results achieved with our branch-and-price algorithm. Instances listed in these tables are labeled with the number of customers, facilities and with letters {a,b,c,d,e} based on the α value used. For example, the instance $r10x4$-a-1 has 10 customers, 4 facilities and $\alpha = 0$. The integer from 1 to 3 after the letters represents the id of the instances within the same group. The tables present the name of each instance, the LP solution value at the root node, the optimal solution value, the number of evaluated nodes, and the CPU time in seconds. In these instances, we first ran the EEBP algorithm for 5 minutes and if the algorithm did not terminate within 5 minutes, we switched to the EEBP-2S algorithm. Table 5 presents the results for instances with 10 customers, Table 6 presents instances with 15 customers, and Table 7 presents instances with 20 customers. We marked the instances with a "+" sign if the EEBP-2S algorithm was used. For problems with at least 20 customers, we needed to use the EEBP-2S algorithm. The branch-and-price algorithm was very successful in finding the optimal solution quickly. In general, our computation times were much smaller than those reported by Laporte et al (1986), but it is difficult to make a fair comparison, given advances in computing technology. In most of the instances, the LP solution found by our algorithm at the root node was the optimal.

The instances become more difficult when the vehicle capacity increases (α increases) because the number of labels generated in the pricing problem depends directly on the vehicle capacity. Laporte et al (1986) observed the reverse effect with regard to their branch-and-cut algorithm. The number of cuts generated increases and the problem gets more difficult when the vehicle capacity is small. This is most likely due to the fact that the problem structure becomes more like that of the traveling salesman problem (TSP) as the capacity is increased and the TSP is much easier to solve by branch and cut than as a capacitated routing problem.

To strictly differentiate the instances from those with a single depot, we experimented with changing the value of parameter N^F, the minimum number of open facilities, to 2 and ran the r15x6 and r20x8 b and c instances. There were no significant changes in the computational times or the number of evaluated nodes. We then added facility capacities to the same set of problems and ran our algorithm again. Table 8 presents the computational results, as well as the facility capacity values used for the facilities. The capacity value was chosen in order to require at least two open facilities. The computational results do not show any significant difference from those of the uncapacitated instances. For larger instances, we expect that the LP solution times will tend to increase if the master problem has facility capacity constraints. Adding a facility capacity to a two-index vehicle flow formulation requires an additional set of constraints (Belenguer et al, 2006), the size of which can be large. In a branch-and-cut algorithm, it may require additional time to generate this set of constraints.

Laporte et al (1986) report that adding facility fixed costs to the problem makes the problem easier. We added facility fixed costs to r15x6 and r20x8 b and c instances. The performance of the branch-and-price algorithm was not affected in instances with 15 customers. However, for some of the 20 customer instances,

Table 5: Performance of Elementary Exact Branch-and-Price for 10 customer instances

Instance	LP	OPT.	# of Nodes	CPU(s)
r10x4-a-1	472.11	472.11	1	0.00
r10x4-a-2	421.44	421.44	1	0.00
r10x4-a-3	548.28	548.28	1	0.02
r10x4-b-1	313.01	313.18	3	0.04
r10x4-b-2	297.57	305.27	19	0.08
r10x4-b-3	352.66	354.92	3	0.03
r10x4-c-1	257.25	257.25	1	0.06
r10x4-c-2	259.76	259.76	1	0.04
r10x4-c-3	296.82	296.82	1	0.05
r10x4-d-1	243.42	257.25	21	0.52
r10x4-d-2	250.04	250.04	1	0.04
r10x4-d-3	296.82	296.82	1	0.04
r10x4-e-1	226.46	226.46	1	0.32
r10x4-e-2	225.82	225.82	1	0.17
r10x4-e-3	272.85	272.85	1	0.31

the computational times exceeded 2 CPU hours (for the results, see Akca (2008)).

4.2.2 Large Random Instances

In this section, we present the results of applying our algorithm to larger capacitated random instances. We generated 6 instances with 30 customers and 6 instances with 40 customers. Each instance had 5 facilities with capacity constraints. The facilities had a fixed cost of 100. The characteristics of each instance are listed in Table 9. The first column includes the name of each instance, labeled based on the number of customers and facilities and the vehicle capacity. For instances in group “a” the vehicle capacity value (listed in the fourth column) was chosen to be 7 times the average demand and for group “b”, the vehicle capacity value was 5.5 times the average demand. Facility capacity (listed in the second column) was chosen based on total demand such that at least two facilities (the minimum number of facilities is listed in the third column) should be open in an integer solution.

We used the EEBP-2S algorithm in which the HBP and EEBP algorithms were both used with a time limit of 3 CPU hours. Table 10 presents the results of both steps. The algorithm was very successful in finding optimal or near-optimal solutions for these larger instances. Some details, such as the number of open facilities, the number of vehicles used at each open facility, the average number of customers in each route (vehicle), and the number of customers in the longest route, are presented in Table 9.

5 Conclusion

We have presented a set-partitioning-based formulation for the capacitated location and routing problem, which to our knowledge is the first of its kind for this class of problem. We have demonstrated that it can be obtained by applying Dantzig-Wolfe decomposition to the graph-based formulation employed in most previously reported research. We have described a branch-and-price algorithm and reported on our experience using it to solve both problems from the literature and randomly generated instances. Our experiments

Table 6: Performance of Elementary Exact Branch-and-Price for 15 customer instances

Instance	LP	OPT.	# of Nodes	CPU(s)
r15x6-a-1	435.2	435.2	1	0.01
r15x6-a-2	663.32	663.32	1	0.01
r15x6-a-3	411.45	411.45	1	0.01
r15x6-b-1	313.46	313.46	1	0.31
r15x6-b-2	414.65	414.65	1	0.21
r15x6-b-3	285.01	285.01	1	0.12
r15x6-c-1	313.46	313.46	1	3.1
r15x6-c-2	392.75	392.75	1	1.98
r15x6-c-3	279.82	279.82	1	5.37
r15x6-d-1	313.36	313.46	3	4.92
r15x6-d-2	378.76	378.76	1	9.41
r15x6-d-3	279.82	279.82	1	14.13
r15x6-e-1	305.86	312.18	5	9.82
r15x6-e-2$^+$	374.86	374.86	1	16.62$^+$
r15x6-e-3	274.22	274.22	1	300.01

$^+$ The EEBP-2S algorithm was used, total time is reported.

indicate that the algorithm is very effective at producing quality solutions and can handle larger instances than previously suggested approaches, which have been primarily based on two-index formulations. The approach, however, does not seem as effective at producing quality lower bounds. This is likely due to the absence of dynamic cut generation. The next step in this research will be to incorporate the generation of known classes of valid inequalities into our algorithm. This should produce an algorithm exhibiting the advantages of both the branch-and-price and branch-and-cut approaches.

References

Akca Z (2008) Location Routing and Scheduling Problems: Models and algorithms. Working paper, COR@L Lab, Lehigh University

Akca Z, Berger RT, Ralphs TK (2008) Modeling and Solving Location Routing and Scheduling Problems. Working paper, COR@L Lab, Lehigh University

Barreto S (2003) LRP instances. URL http://sweet.ua.pt/iscfl43

Barreto S (2004) Analysis and Modelling of Location-Routing Problems (in portuguese). Phd thesis, University of Aveiro, Aveiro, Portugal

Barreto S, Ferreira C, Paixao J, Santos BS (2007) Using Clustering Analysis in a Capacitated Location-Routing Problem. European Journal of Operational Research 127(3):968–977

Belenguer JM, Benavent E, Prins C, Prodhon C, Wolfler-Calvo R (2006) A Branch and Cut method for the Capacitated Location-Routing Problem. Service Systems and Service Management, 2006 International Conference on 2:1541–1546

Berger RT (1997) Location-Routing Models for Distribution System Design. PhD thesis, Department of Industrial Engineering and Management Sciences, Northwestern University

Table 7: Performance of Elementary Exact Branch-and-Price for 20 customer instances

Instance	LP	OPT.	# of Nodes	CPU(s)
r20x8-a-1	639.77	653.11	57	0.41
r20x8-a-2	542.23	551.58	25	0.98
r20x8-a-3	760.42	760.42	1	0.05
r20x8-b-1	415.3	417.13	7	8.35
r20x8-b-2	383.19	383.19	1	6.32
r20x8-b-3	439.18	447.72	121	37.32
r20x8-c-1$^+$	398.34	398.34	1	26.25$^+$
r20x8-c-2$^+$	363.86	363.86	1	110.87$^+$
r20x8-c-3$^+$	402.85	402.85	1	28.74$^+$
r20x8-d-1$^+$	392.26	392.26	1	10.04$^+$
r20x8-d-2$^+$	359.49	359.49	1	200.29$^+$
r20x8-d-3$^+$	402.85	402.85	1	124.24$^+$
r20x8-e-1$^+$	392.28	392.28	1	12.39$^+$
r20x8-e-2$^+$	355.39	355.39	1	82.77$^+$
r20x8-e-3$^+$	402.46	402.46	1	103.27$^+$

$^+$ The EEBP-2S algorithm is used, total time is reported.

Berger RT, Coullard CR, Daskin M (2007) Location-Routing Problems with Distance Constraints. Transportation Science 41:29–43

Desrochers M, Soumis F (1989) A Column Generation Approach to the Urban Transit Crew Scheduling Problem. Transportation Science 23(1):1–13

Desrochers M, Desrosiers J, Solomon MM (1992) A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows. Operations Research 40(2):342–354

Dumitrescu I (2002) Constrained Path and Cycle Problems. PhD thesis, Department of Mathematics and Statistics, University of Melbourne

Feillet D, Dejax P, Gendreau M, Gueguen C (2004) An Exact Algorithm for the Elementary Shortest Path Problem with Resource Constraints: Application to Some Vehicle Routing Problems. Networks 44(3):216–229

Hansen PH, Hegedahl B, Hjortkjaer S, Obel B (1994) A Heuristic Solution to the Warehouse Location-Routing Problem. European Journal of Operational Research 76:111–127

Irnich S, Desaulniers G (2005) Column Generation, Springer, chap Shortest Path Problems with Resource Constraints, pp 33–65. GERAD 25th Anniversary Series

Laporte G (1988) Location Routing Problems. In: Golden B, Assad A (eds) Vehicle Routing: Methods and Studies, North-Holland, Amsterdam, pp 293–318

Laporte G, Nobert Y (1981) An Exact Algorithm for Minimizing Routing and Operating Cost in Depot Location. European Journal of Operational Research 6:224–226

Laporte G, Nobert Y, Pelletier Y (1983) Hamiltonian Location Problems. European Journal of Operational Research 12:82–89

Table 8: Instances with Capacitated Facilities

Instance	Fac. Cap	LP	OPT.	# of Nodes	Total Time
cr15x6-c-1	600	299.07	299.07	1	1.15
cr15x6-c-2	600	392.75	392.75	1	2.53
cr15x6-c-3	600	279.82	279.82	1	1.86
cr15x6-d-1	700	299.07	299.07	1	0.57
cr15x6-d-2	700	378.76	378.76	1	6.42
cr15x6-d-3	700	279.82	279.82	1	11.01
cr20x8-c-1	750	398.34	398.34	1	40.46
cr20x8-c-2	750	363.86	363.86	1	88.12
cr20x8-c-3	750	402.85	402.85	1	15.25
cr20x8-d-1	900	392.29	392.29	1	29.1
cr20x8-d-2	750	359.49	359.49	1	257.94
cr20x8-d-3	900	402.85	402.85	1	31.3

Laporte G, Nobert Y, Arpin D (1986) An Exact Algorithm for Solving a Capacitated Location-Routing Problem. Annals of Operations Research 6:293–310

Min H, Jayaraman V, Srivastava R (1998) Combined Location-Routing Problems: A Synthesis and Future Research Directions. European Journal of Operational Research 108(1):1–15

Perl J, Daskin MS (1985) A Warehouse Location-Routing Model. Transportation Research 19(B):381–396

Vance PH, Barnhart C, Johnson E, Nemhauser GL (1994) Solving Binary Cutting Stock Problems by Column Generation and Branch and Bound. Computational Optimization and Applications 3:111–130

Table 9: Characteristics of the Instances and the Optimal Solutions

Instance				OPT/BestIP Solution Info			
Name	Fac. Cap.	N^F	L^V	# of Fac.	# of Vec.	Avg. # of Cust/route	# of Cust. longest route
cr30x5a-1	1000	2	350	2	3,2	6	8
cr30x5a-2	1000	2	350	2	3,2	6	7
cr30x5a-3	1000	2	350	2	3,3	5	7
cr30x5b-1	1000	2	275	2	3,2	6	8
cr30x5b-2	1000	2	275	2	4,3	4.29	7
cr30x5b-3	1000	2	275	2	3,4	4.29	6
cr40x5a-1	1750	2	340	2	3,3	6.67	8
cr40x5a-2	1750	2	390	2	3,4	5.71	8
cr40x5a-3	1750	2	370	2	3,3	6.67	7
cr40x5b-1	1750	2	275	2	3,5	5	7
cr40x5b-2	1750	2	275	2	3,5	5	7
cr40x5b-3	1750	2	325	2	5,3	5	7

Table 10: Performance of the EEBP-2S for Instances up to 40 Customers with Capacitated Facilities and Facility Fixed Cost

Instance	HBP		EEBP					Total
	IP	CPU(s)	LP	IP	Gap	# of N.	CPU(s)	CPU (s)
cr30x5a-1	819.53	43.5	810.29	819.5	0	33	993.3	1036.8
cr30x5a-2	823.49	7202.3	790.49	823.49	2.55	500	10806.5	18008.8
cr30x5a-3	702.29	44.2	687.72	702.19	0	51	917.9	962.1
cr30x5b-1	880.02	164.3	865.47	880.01	0	251	6420.6	6585
cr30x5b-2	825.32	8.3	815.95	825.3	0	7	33.2	41.5
cr30x5b-3	884.62	13.4	881.33	884.55	0	19	41.7	55.1
cr40x5a-1	928.11	631.8	911.39	928.11	1.49	11	10882.8	11514.6
cr40x5a-2	888.37	378.4	871.66	888.37	0.93	13	11052.9	11431.3
cr40x5a-3	947.24	173	939.54	947.24	0.18	28	10862	11035
cr40x5b-1	1052.07	257.3	1043.62	1052	0	627	8084.6	8342
cr40x5b-2	981.52	60.1	976.88	981.27	0	47	862.5	922.7
cr40x5b-3	964.32	62.6	959.05	964.23	0	45	963	1025.6