Two-echelon vehicle routing problem
Technical report

Markó Horváth[1,*] and Tamás Kis[1]

[1]*Institute for Computer Science and Control, H-1111 Budapest, Kende u. 13-17.*
[*]*Corresponding author, marko.horvath@sztaki.hu*

Abstract

In this paper we investigate a two-echelon vehicle routing problem where pickups and deliveries are planned simultaneously. Pickup offerings of semi-finished goods and delivery requests for finished goods may arrive throughout the planning horizon, each of them having its own time-window. A fleet of distinct vehicles divided into multiple depots is used to transport semi-finished goods to processing facilities, and then for transporting finished goods to customers. Processing is not instantaneous, i.e., each semi-finished good has a lead time. The goal is to simultaneously plan pickup and delivery routes such that all the problem constraints are respected and distinct operational and penalty costs are minimized. We propose an exact branch-and-price approach to solve the problem.

1 Introduction

The original motivation of this research is a wood recycling, where some customers offer wood waste (e.g., construction or demolition waste, etc.) for removal, and other customers demand recycled wood. Of course, wood waste cannot be transported directly to customers, it has to be first recycled at some processing facilities. So, assume there is a set of semi-finished goods which have to be processed to obtain finished goods. In the previous example these are distinct wood types which have to be recycled (that is, revised, cleaned, classified, etc.), however, these could be distinct parts which have to be anodized or painted, or machines which have to be repaired, etc. Pickup requests (offering semi-finished goods) and delivery requests (demanding finished goods) arrive over a finite planning horizon. Each request has an individual time-window (that is, a set of subsequent periods) when it can be satisfied. It is not mandatory to satisfy all of the requests, however, each pickup request has a rejection penalty, and each delivery request offers some profit.

In this paper we investigate a two-echelon many-to-many pickup-and-delivery problem, where semi-finished goods have to be first transported to an intermediate facility where they are processed, and then the finished goods are delivered to customers. Our main interest is an exact method for solving the problem optimally.

Structure of the paper

The paper is organized as follows. In Section 2 we overview related literature. In Section 3 we define the problem formally. In Section 4 we present our modeling approach along with a mixed-integer linear programming formulation. Then, in Section 5 we propose our branch-and-price solution approach for the problem. Finally, we present our computational results in Section 6.

2 Literature review

Briefly stated, the familiar *vehicle routing problem* aims to determine an optimal set of routes to be performed by a fleet of vehicles to fulfill customer demands at different locations. The problem was introduced more than 60 years ago by Dantzig and Ramser (1959), then generalized by Clarke and Wright (1964), and many variations have appeared since then, see e.g., Toth and Vigo (2002); Eksioglu et al. (2009); Braekers et al. (2016); Zhang et al. (2022).

In case of a *pickup-and-delivery problem* each order have to be transported between an origin and a destination. Using the terminology of Berbeglia et al. (2010), in a *many-to-many pickup-and-delivery problem* orders are not associated with a fix origin and destination, but several locations can serve as a pickup or a delivery point.

In contrast to direct delivery, in case of a multi-echelon vehicle routing problem orders are moved through some intermediate facilities (e.g., cross-docks, distribution centers, etc.) before reaching their destinations. Obviously, in case of a *two-echelon vehicle routing problem*, orders have to be transported only at one intermediate facility (Cuda et al., 2015; Sluijk et al., 2022).

3 Problem formulation

In this section we formalize the problem. Notation are summarized in Tables 1 to 3.

The planning horizon is divided into a finite set of consecutive periods, i.e., $T = \{1, 2, \ldots, t^{\text{end}}\}$.

Let P_1 and P_2 be the set of semi-finished and the finished goods, respectively. Note that $|P_1| = |P_2|$, and there is a bijection $b : P_1 \to P_2$ between these sets, that is, the finished good corresponds to semi-finished good $k \in P_1$ is $b(k)$, and the semi-finished good corresponds to finished good $k \in P_2$ is $b^{-1}(k)$. Finally, let $P_{1,2} = P_1 \cup P_2$ be the set of all goods.

Semi-finished goods are processed and become finished goods at given facilities. These processes follow detailed schedules, however, we assume that each good has a fixed lead time, and the total quantity that can be transported to a facility in a period is limited. Let V_f be the set of facilities, and let qlim_i denote the upper bound for the total quantity of semi-finished good that can be transported to facility $i \in V_f$ in a period. The lead time of a semi-finished good $k \in P_1$ at facility $i \in V_f$ is denoted by ltime_{ik}, that is, if semi-finished good k of quantity q is transported to facility i at period t, then finished good $b(k)$ of quantity q appears at the beginning of period $t + \text{ltime}_{ik}$ at facility i. To respect the planning horizon, such a good cannot be transported to the facility if $t^{\text{end}} < t + \text{ltime}_{ik}$.

Supplies of semi-finished goods and demands for finished goods (that is, *pickup requests* and *delivery requests*, respectively) may arrive throughout the planning horizon. Each pickup request belongs to a *pickup point* and each delivery request has a *delivery point*. For convenience it is assumed that these points are unique, that is, exactly one request belongs to each point. Thus expressions 'request' and 'point' are used interchangeably. The set of pickup points and delivery points are denoted by V_p and V_d, respectively. Let quantity_{ik} be the non-negative supply (demand) quantity from good $k \in P_1$ ($k \in P_2$) at pickup (delivery) point $i \in V_p$ ($i \in V_d$). Let $\text{tw}_i \subseteq T$ be the time-window (that is, a set of consecutive periods) when request $i \in V_p \cup V_d$ can be satisfied, let penalty_i be the penalty for rejecting pickup request $i \in V_p$ and profit_i be the profit for satisfying delivery request $i \in V_d$.

Table 1: Notation (sets)

T	planning horizon
P_1	set of semi-finished goods
P_2	set of finished goods
$P_{1,2}$	set of goods: $P_1 \cup P_2$
V_0	set of depots
V_f	set of facilities
V_p	set of pickup points
V_d	set of delivery points
V	set of locations: $V_0 \cup V_p \cup V_d \cup V_f$
W	set of vehicles

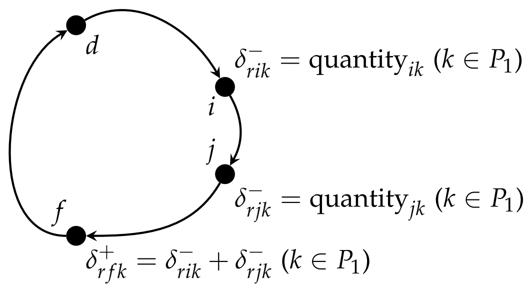
Table 2: Notation (parameters)

$\text{dist}_{ij} \in \mathbb{R}_{0\leq}$	distance between locations $i \in V$ and $j \in V$
$\text{dlim}_v \in \mathbb{R}_{0\leq}$	distance limit of vehicle $v \in W$ in a period
$\text{cap}_v \in \mathbb{Z}_{0\leq}$	capacity of vehicle $v \in W$
$\text{cost}_v \in \mathbb{R}_{0\leq}$	distance cost of vehicle $v \in W$
$\text{tw}_i \subseteq T$	time-window for pickup of delivery request $i \in V_p \cup V_d$
$\text{penalty}_i \in \mathbb{R}_{0\leq}$	penalty for rejecting pickup request $i \in V_p$
$\text{profit}_i \in \mathbb{R}_{0\leq}$	profit for satisfying delivery request $i \in V_d$
$\text{quantity}_{ik} \in \mathbb{Z}_{0\leq}$	quantity of good $k \in P_{1,2}$ at point $i \in V_p \cup V_d$
$\text{ltime}_{ik} \in \mathbb{Z}_{0<}$	lead time of semi-finished good $k \in P_1$ at facility $i \in V_f$
$\text{qlim}_i \in \mathbb{Z}_{0\leq}$	upper limit of transported goods to facility $i \in V_f$ in a period

A fleet of inhomogeneous vehicles divided into multiple depots are available to transport goods. Each vehicle starts and ends its routes at the same depot. Each vehicle can return to its depot several times during a period, however, the total distance traveled in one period is limited. Several aspects can be considered to restrict the goods transported by a vehicle at the same time (e.g., weight, volume, etc.). In this paper we assume that only the quantity of these goods are limited, however, our model can be easily extend to handle multiple criteria. Let W be the set of vehicles, and V_0 be the set of depots. For each vehicle $v \in W$, let dlim_v its traveling distance limit in a period, cap_v its capacity (that is the maximum quantity of goods), and cost_v be its distance cost (per kilometer).

A route is a sequence of locations along with pickup up and delivered quantities, however, it is more convenient to extend this concept such that each route belongs to a (period, vehicle) pair. As we mentioned earlier, in the rest of the paper we consider two types of routes (see Figure 1). A *pickup route* r belonging to vehicle $\nu_r \in W$, period $\tau_r \in T$, and facility $\lambda_r \in V_f$, has a sequence Λ_r of pairwise distinct pickup points, and has non-negative drop-off δ^+_{rik} and pickup quantities δ^-_{rik} for each location $i \in V$ and for each good $k \in P_{1,2}$ such that $\delta^-_{rik} = \text{quantity}_{ik}$ for all $i \in \Lambda_r$ and $k \in P_1$, $\delta^+_{r,\lambda_r,k} = \sum_{i \in \Lambda_r} \text{quantity}_{ik}$, and zero otherwise. The total distance and the total cost of a route r is denoted by σ_r and ν_r, respectively.

A pickup route r is *feasible* if its total distance does not exceed the daily limit of the corresponding vehicles (that is, $\sigma_r \leq \text{dlim}_{\nu_r,\tau_t}$), the transported quantities never exceed the capacity



d
$\delta^-_{rik} = \text{quantity}_{ik}\ (k \in P_1)$
i
j
$\delta^-_{rjk} = \text{quantity}_{jk}\ (k \in P_1)$
f
$\delta^+_{rfk} = \delta^-_{rik} + \delta^-_{rjk}\ (k \in P_1)$

(a) Pick-up route r with $\Lambda_r = (i, j)$ and $\lambda_r = f$.

Figure 1: Example for routes. Figure 1a: vehicle departs from its depot empty, visits some pickup points where picks up all of the available semi-finished goods, then visits a facility where drops off all the collected goods, and finally returns to its depot.

Table 3: Notation (routes)

R_p	set of feasible pickup routes
R_d	set of feasible delivery routes
R	set of feasible routes: $R_p \cup R_d$
$\tau_r \in T$	period of route $r \in R$
$\nu_r \in W$	vehicle assigned to route $r \in R$
$\lambda_r \in V_f$	facility belongs to route $r \in R$
Λ_r	sequence of pickup (delivery) points visited by route $r \in R_p$ $(r \in R_d)$
$\delta^+_{rik} \in \mathbb{Z}_{0\leq}$	amount of good $k \in P_{1,2}$ dropped off at location $i \in V$ by route $r \in R$
$\delta^-_{rik} \in \mathbb{Z}_{0\leq}$	amount of good $k \in P_{1,2}$ picked-up at location $i \in V$ by route $r \in R$
$\phi_r \in \mathbb{R}_{0\leq}$	cost of route $r \in R$
$\sigma_r \in \mathbb{R}_{0\leq}$	total distance of route $r \in R$
$\xi_{ri} \in \{0, 1\}$	indicates whether route $r \in R$ visits location $i \in V$

of the vehicle (that is, $\sum_{i\in\Lambda_r, k\in P_1} \delta^-_{rik} \leq \text{cap}_{\nu_r}$), and all its pickup points can be visited in the corresponding period (that is, $\tau_r \in \text{tw}_i$ for all $i \in \Lambda_r$). The set of all the feasible pickup routes is denoted by R_p.

To ease our notation we introduce value ξ_{ri} indicating whether route $r \in R$ visits location $i \in V$, that is, $\xi_{ri} = 1$ if $i \in \Lambda_r$ or $i = \lambda_r$, and zero otherwise.

4 Modeling approach

We handle pickup and delivery routes separately, that is, we assume that semi-finished goods and finished goods cannot be transported by the same truck at the same time. The reason is that, after these routes are determined, schedules at processing facilities can be determined to yield accurate completion times for the goods. By this, fixing the pickup routes, and using these completion times, one can resolve the problem to determine delivery routes.

In this paper we only consider the global problem, that is, where pickup and delivery routes should be determined simultaneously.

We introduce three types of variables. Binary variable $\mathbf{x}_r$ indicates whether route $r \in R$ is performed. Binary variable $\mathbf{y}_i$ indicates whether pickup or delivery point $i \in V_p \cup V_d$ is visited by a route, or in other words, whether pickup or delivery request is satisfied. Non-negative continuous variable $\mathbf{s}_{ikt}$ indicates the closing stock level at facility $i \in V_f$ from good $k \in P_2$ at the end of period $t \in T \cup \{0\}$.

We formalize the problem as follows.

$$\text{minimize} \sum_{r \in R} \phi_r \mathbf{x}_r - \sum_{i \in V_p} \text{penalty}_i \, \mathbf{y}_i - \sum_{i \in V_d} \text{profit}_i \, \mathbf{y}_i \tag{1}$$

$$\sum_{r \in R:\ i \in \Lambda_r} \mathbf{x}_r = \mathbf{y}_i \quad \text{for all } i \in V_p \cup V_d \tag{2a}$$

$$\sum_{r \in R(t,v)} \sigma_r \mathbf{x}_r \leq \text{dlim}_v \quad \text{for all } t \in T,\ v \in W \tag{2b}$$

$$\mathbf{s}_{ikt} = \mathbf{s}_{i,k,t-1} + \sum_{r \in R_p(t - \text{ltime}_{i,b^{-1}(k)})} \delta^+_{r,i,b^{-1}(k)} \mathbf{x}_r - \sum_{r \in R_d(t)} \delta^-_{rik} \mathbf{x}_r \quad \text{for all } i \in V_f,\ k \in P_2,\ t \in T \tag{2c}$$

$$\sum_{r \in R_p} \sum_{k \in P_1} \delta^+_{ikt} \mathbf{x}_r \leq \text{qlim}_i \quad \text{for all } i \in V_f,\ t \in T \tag{2d}$$

$$\mathbf{x}_r \in \{0,1\} \quad \text{for all } r \in R \tag{2e}$$

$$\mathbf{y}_i \in \{0,1\} \quad \text{for all } i \in V_p \cup V_d \tag{2f}$$

$$0 \leq \mathbf{s}_{ikt} \quad \text{for all } i \in V_f,\ k \in P_2,\ t \in T \tag{2g}$$

$$\mathbf{s}_{ik0} = 0 \quad \text{for all } i \in V_f,\ k \in P_2 \tag{2h}$$

The objective (1) minimizes the total cost of performed routes plus the total penalties of rejected pickup requests minus the total profit of satisfied delivery request. Note that using a constant offset $\sum_{i \in V_p} \text{penalty}_i$ the objective function is equivalent with the following one

$$\text{minimize} \sum_{r \in R} \phi_r \mathbf{x}_r + \sum_{i \in V_p} \text{penalty}_i (1 - \mathbf{y}_i) - \sum_{i \in V_d} \text{profit}_i \, \mathbf{y}_i.$$

Constraint (2a) ensures that $\mathbf{y}_i = 1$ if and only if a route visits point $i \in V_p \cup V_d$, and by this, no pickup or delivery point is visited by multiple routes. Distance limits of vehicles per period are respected due to constraints (2b). Constraint (2c) expresses the closing stock level of processing facilities at the end of periods. Quantity limits for facilities are forced by constraint (2d). Initial stock levels are set by constraint (2h) (which is zero in this case).

Note that $\mathbf{y}$-variables could be eliminated from the problem and equation (2a) could be replaced by inequality

$$\sum_{r \in R:\ i \in \Lambda_r} \mathbf{x}_r \leq 1 \quad \text{for all } i \in V_p \cup V_d,$$

however, we will take advantage of the former modeling approach in our branching strategies (see Section 5.3).

5 Solution approach

In this section we propose an exact branch-and-price approach to solve master problem (1), (2a)–(2h). An initial restricted master problem consists of only a subset of variables is building (see Section 5.1), and additional variables may be added to the problem during the global search procedure, that is, in each node of the search-tree variable pricing is performed (see Section 5.2). Problem-specific rules are introduced to perform branching when the optimal solution to the restricted master problem is fractional (see Section 5.3).

5.1 Initial restricted master problem

The initial restricted master problem (initial RMP) consists of all the **s**- and **y**-variables, however, **x**-variables are priced out during the global search procedure. Remark that the initial RMP is feasible, since rejecting all the requests yields a feasible solution.

5.2 Variable pricing

In an exact branch-and-price procedure in each search-tree node variable pricing is necessary to get valid lower bound on the objective value or to prove that the current subproblem is infeasible. If the current LP-relaxation is feasible, additional variables may contribute to improve its solution value, thus variable pricing should search for variables with negative reduced cost. If no variable with negative reduced cost exists, then the solution value is a valid lower bound on the objective function. If the current LP-relaxation is infeasible, additional variables may make the problem feasible, thus variable pricing is also needed, however, the so-called Farkas coefficients have to be used instead of the reduced costs. If no variable with negative Farkas coefficient exists, then the subproblem of the node is indeed infeasible.

If the current LP-relaxation is feasible, let $a_i \in \mathbb{R}$ $(i \in V_p \cup V_d)$, $b_{tv} \in \mathbb{R}_{0\leq}$ $(t \in T,\ v \in W)$, $c_{ikt} \in \mathbb{R}$ $(i \in V_f,\ k \in P_2,\ t \in T)$, $d_{it} \in \mathbb{R}_{0\leq}$ $(i \in V_f,\ t \in T)$ be the dual values corresponding to inequalities (2a), (2b), (2c) and (2d) respectively. The pricing problem for the pickup routes and the delivery routes are:

$$\min\nolimits_{r \in R_p} \left\{ \phi_r + \sum_{i \in \Lambda_r} a_i + b_{\tau_r,\nu_r}\sigma_r + \sum_{i \in V_f}\sum_{k \in P_2} c_{i,k,\tau_r + \mathrm{ltime}_{ib^{-1}(k)}} \delta^{+}_{rib^{-1}(k)} + \sum_{i \in V_f}\sum_{k \in P_1} d_{i,\tau_r}\delta^{+}_{rik} \right\} \tag{3}$$

and

$$\min\nolimits_{r \in R_d} \left\{ \phi_r + \sum_{i \in \Lambda_r} a_i + b_{\tau_r,\nu_r}\sigma_r - \sum_{i \in V_f}\sum_{k \in P_2} c_{i,k,\tau_r}\delta^{-}_{rib^{-1}(k)} \right\}, \tag{4}$$

respectively.

If the current LP-relaxation is infeasible, then instead of dual values, the so-called Farkas multipliers (these are the values that prove the infeasibility of the LP) are available for the constraints. The Farkas pricing is similar to the reduced cost pricing, however, zero objective function and Farkas multipliers are used. That is, for example in case of pickup-up routes, the pricing problem is similar to (3), however, term ϕ_r is omitted, and values a_i, b_{tv}, c_{ikt}, d_{it} refer to the Farkas multipliers.

In Section 5.2.1 we propose a solution approach for pricing problem (3). Since we use similar solution approach for pricing problem (4) and for the corresponding Farkas pricing problems, we omit the description of these procedures.

Note that branching decisions also should be taken into consideration during the pricing, that is, only those routes have to be considered which respect the corresponding branching constraints. We detail these modifications in Section 5.3.

5.2.1 Pickup routes

We solve problem (3) separately for each tuple $(v, t, f) \in W \times T \times V_f$, that is, instead of all routes of R_p we only consider those pickup routes which is performed by vehicle v, belongs to period t, and drops off semi-finished goods at facility f.

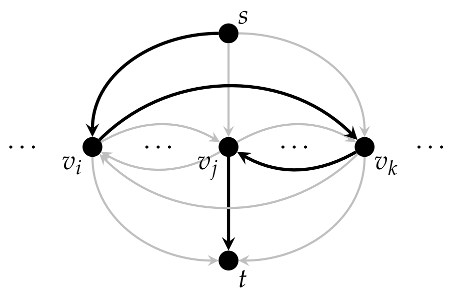


Figure 2: Directed graph for pricing. The path of solid black arcs represents pickup route which visits pickup points i, k and j.

Fix a tuple $(v, t, f) \in W \times T \times V_f$, and assume that vehicle v belongs to depot d. We define a directed graph $D = (N, A)$ with node set $N = \{s, t\} \cup \{v_i : i \in V_p\}$ and arc set $A = \{(s, v_i) : i \in V_p\} \cup \{(v_i, t) : i \in V_p\} \cup \{(v_i, v_j) : (i, j) \in V_p \times V_p,\ i \neq j\}$, where node s represents depot d, and node t represents the path from facility t to depot d, see Figure 2.

Observation 1. *Every elementary path (that is, nodes are not repeated) from s to t represents a (not necessarily feasible) pickup route, and each (not necessary feasible) pickup route is represented by an elementary s–t path.*

We define functions $\ell : A \to \mathbb{R}$, $w_1 : A \to \mathbb{R}_{0\leq}$, and $w_2 : A \to \mathbb{Z}_{0\leq}$ on the arcs as follows. Let $w_1(s, v_i) := \mathrm{dist}_{di}$ for all $i \in V_p$, $w_1(v_i, v_j) := \mathrm{dist}_{ij}$ for all $(i, j) \in V_p \times V_p$ $(i \neq j)$, and $w_1(v_i, t) := \mathrm{dist}_{if} + \mathrm{dist}_{fd}$ for all $i \in V_p$. Let $w_2(s, v_i) := 0$ for all $i \in V_p$, $w_2(v_i, v_j) := \sum_{k \in P_1} \mathrm{quantity}_{ik}$ for all $(i, j) \in V_p \times V_p$ $(i \neq j)$, and $w_2(v_i, t) := \sum_{k \in P_1} \mathrm{quantity}_{ik}$ for all $(i, j) \in V_p$. Finally, let $\ell(s, v_i) := \mathrm{dist}_{di}(\mathrm{cost}_v + b_{tv})$ for all $i \in V_p$, $\ell(v_i, v_j) := \mathrm{dist}_{ij}\,\mathrm{cost}_v + \mathrm{quantity}_{ik}\, d_{it} + \sum_{k \in P_1} \mathrm{quantity}_{ik}\, c_{fkt}$.

Observation 2. *For each elementary s–t path P, $w_1(P) := \sum_{e \in P} w_1(e)$ equals to the total distance of the corresponding route, $w_2(P) := \sum_{e \in P} w_2(e)$ equals to the total picked up quantity during the corresponding route, and $\ell(P) := \sum_{e \in P} \ell(e)$ equals to the reduced cost of the corresponding route.*

Based on the previous observations we have the following proposition.

Proposition 1. *There is a bijection between the feasible pickup routes and those elementary s–t paths P in D where $w_1(P) \leq \mathrm{dlim}_{vt}$ and $w_2(P) \leq \mathrm{cap}_v$. Moreover, the reduced cost of such a route is equals to $\ell(P)$.*

Thus, finding a route with negative reduced cost is equivalent to finding an elementary s–t path P with negative cost ℓ such that $w_1(P) \leq \mathrm{dlim}_{vt}$ and $w_2(P) \leq \mathrm{cap}_v$.

5.3 Branching rules

In this section we present our strategies to branch on binary **x**- and **y** variables when the optimal solution $(\bar{\mathbf{s}}, \bar{\mathbf{x}}, \bar{\mathbf{y}})$ to the restricted master problem of the corresponding search-tree node is fractional (i.e., some of the **x**-variables or **y**-variables take on fractional values).

Since all of the **y**-variables are added to the initial restricted master problem, the custom 0-1 branching can be performed on these variables. However, the **x**-variables are priced out during the branch-and-price procedure, thus when branching is performed on these variables, for each branch a constraint is created and sticked to the corresponding child node. In a search-tree node a constraint is *active*, if it is sticked to one of the ancestor nodes. When the branch-and-price solver selects a node to be optimized, active constraints are endorsed, that is, existing variables violating these constraints are fixed locally to zero. Active constraints are also taken into consideration in variable pricing.

5.3.1 Branching on y-variables

First of all, branching on $\mathbf{y}$-variables is performed. That is, if there is at least one $\mathbf{y}$-variable in the branching candidate list, one of them is choosing to perform the custom 0-1 branching. That is, let $i \in V_p \cup V_d$ be a point such that the value $|0.5 - \bar{\mathbf{y}}_i|$ is minimal. Then, two branches are created, locally fixing variable $\mathbf{y}_i$ to 0 on the first branch, and to 1 on the other branch.

5.3.2 Assignment branching

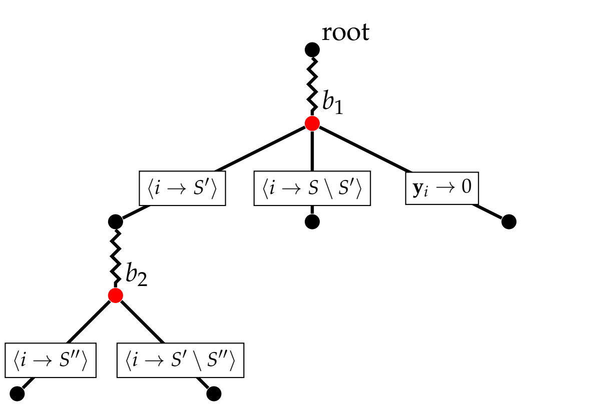
The next three branching rules are based on a similar idea. Briefly stated, if a pickup point is visited, then it is visited in a single period by a single vehicle, and its semi-finished goods are transported to a single facility. We can say, in an integer solution each visited pickup point is committed to a single period, to a single vehicle and to a single facility. However, in a fractional solution a pickup point i might be committed for example to multiple periods, that is, there are routes r_1 and r_2 with $0 < \bar{\mathbf{x}}_{r_1}, \bar{\mathbf{x}}_{r_2} < 1$ and $\tau_{r_1} \neq \tau_{r_2}$. In this case two branches can be made forbidding to visit pickup point i in period τ_{r_1} on the first branch and forbidding to visit pickup point i in period τ_{r_2} on the second branch.

In general, let S be a set of periods, or vehicles, or facilities. If there is a pickup point $i \in V_p$ committed to multiple elements of S, say s_1 and s_2, then S is partitioned into subsets S' and $S \setminus S'$ such that $s_1 \in S'$ and $s_2 \in S \setminus S'$, and three branches are created with constraints $\langle i \rightarrow S' \rangle$, $\langle i \rightarrow S \setminus S' \rangle$ and $\langle i \rightarrow \emptyset \rangle$, respectively. That is, pickup point i is required to be committed to one of the elements of S' on the first branch, to one of the elements of $S \setminus S'$ on the second branch, and pickup request at point i is rejected in the third one. Note that the third, *reject branch* is not necessary if a branching related to pickup point i is already performed at some of the ancestor nodes, since it would yield an infeasible node. Also remark that such a branch refers to the case when $\mathbf{y}_i$ is fixed locally to zero. See Figure 3 for an example.

When a node is selected by the branch-and-price solver to be optimized, a constraint $\langle i \rightarrow S \rangle$ can be easily endorsed. That is, each existing variable $\mathbf{x}_r$ of the corresponding restricted master problem with $\xi_{ri} = 1$ can be locally fixed to zero, if S is a set of periods and $\tau_r \notin S$, if S is a set of vehicles and $\nu_r \notin S$, or if S is a set of facilities with $j \notin S$ and $\xi_{rij} = 1$. The corresponding cover variable $\mathbf{y}_i$ is also fixed to 1 locally.

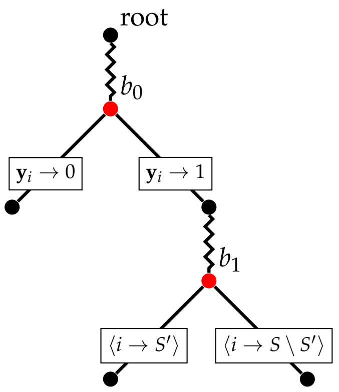
When pricing is performed in a search-tree node (see Section 5.2) active constraints can be taken into consideration by removing nodes from the pricing graph, that is, the resource feasible elementary s–t paths of the modified graph represent those feasible pickup routes which respect the branching constraints, and vice-versa. Assume that the current pricing belongs to the tuple $(v, t, f) \in W \times T \times V_f$, then each node v_i $(i \in V_p)$ is deleted from the pricing graph, if there is an active constraint $\langle i \rightarrow S \rangle$ such that S is a set of vehicles which does not include v, or S is a set of periods which does not include period t, or S is a set of facilities which does not include f.

Assign a pickup point to a time-window In this branching rule pickup requests are forced to be managed in given time-windows, that is, constraints in the form of $\langle i \rightarrow [t_s, t_e] \rangle$ are created, where $i \in V_p$ and $1 \leq t_s \leq t_e \leq t^{\text{end}}$, or $\langle i \rightarrow \emptyset \rangle$ to forbid pickup point i to be visited.



root
b_1
$\langle i \to S' \rangle$
$\langle i \to S \setminus S' \rangle$
$\mathbf{y}_i \to 0$
b_2
$\langle i \to S'' \rangle$
$\langle i \to S' \setminus S'' \rangle$

(a) Assignment branching with no 0-1 branching before



root
b_0
$\mathbf{y}_i \to 0$
$\mathbf{y}_i \to 1$
b_1
$\langle i \to S' \rangle$
$\langle i \to S \setminus S' \rangle$

(b) Assignment branching after 0-1 branching

Figure 3: Examples for the search-tree, where in the red nodes, branchings are performed related to pickup point i. In the left example, since no 0-1 branching is performed before the first assignment branching (node b_1), a discarding branch is also created, however, in the next time in this sub-tree (node b_2), discarding branch is omitted since it would yield az infeasible sub-problem. In the right example, since a 0-1 branching is performed before assignment branching, no discarding branch is necessary.

Let $C_{it} := \sum_{r \in R_p : \tau_r = t} \xi_{ri} \bar{\mathbf{x}}_r$ denote the commitment of pickup point $i \in V_p$ to period $t \in T$. Note that in case of integer solutions $C_{it} \in \{0, 1\}$ holds, moreover, for each pickup point i there is at most one period t with $C_{it} = 1$. We say that a pickup point is committed to multiple periods (with respect to the fractional solution), if there are distinct periods t_1 and t_2 such that $0 < C_{i,t_1} < 1$ and $0 < C_{i,t_2} < 1$.

If there is no pickup point committed to multiple periods, then the branching rule cannot be performed. Otherwise, let $i \in V_p$, $t_1 \in T$ and $t_2 \in T$ such that $0 < C_{i,t_1} < 1$, $0 < C_{i,t_2} < 1$, $C_{it} = 0$ for all $t_1 < t < t_2$, $\sum_{t=1}^{t_1} C_{it} \leq 0.5$, $0.5 < \sum_{t=1}^{t_2} C_{it}$ and the value $0.5 - \sum_{t=1}^{t_1} C_{it}$ is minimal. Finally, let $[t_s, t_e]$ be the widest time-window when pickup point i can be visited (in the current sub-search-tree), and $t_b := \lfloor (t_1 + t_2)/2 \rfloor$. Then, three branches are created with the following constraints $\langle i \to [t_s, t_b] \rangle$, $\langle i \to [t_b + 1, t_e] \rangle$, and $\langle i \to \emptyset \rangle$ respectively.

Assign a pickup point to a set of facilities In this branching rule we force pickup offerings to be transported to given facilities, that is, we create constraints in the form of $\langle i \to S \rangle$ where $i \in V_p$ and $S \subsetneq V_f$.

Let $C_{ij} := \sum_{r \in R_p} \xi_{rij} \bar{\mathbf{x}}_r$ denote the commitment of pickup point $i \in V_p$ to facility $j \in V_f$. Note that in case of integer solutions $C_{ij} \in \{0, 1\}$ holds, moreover, for each pickup point i there is at most one facility j with $C_{ij} = 1$. We say that a pickup point is committed to multiple facilities (with respect to the fractional solution), if there are distinct facilities j_1 and j_2 such that $0 < C_{i,j_1} < 1$ and $0 < C_{i,j_2} < 1$.

If there is no pickup point committed to multiple facilities, then the branching rule cannot be performed. Otherwise, such a pickup point, say i, is chosen. Let $\langle i \to S \rangle$ be the strictest active constraint concerns to this pick-ip point where S is a set of facilities. S is partitioned into subsets S' and $S \setminus S'$ such that $0 < \sum_{j \in S'} C_{ij} < 1$ and $0 < \sum_{j \in S \setminus S'} C_{ij} < 1$. Then, three branches are created with the following constraints $\langle i \to S' \rangle$, $\langle i \to S \setminus S' \rangle$, and $\langle i \to \emptyset \rangle$ respectively.

Assign a pickup point to a set of vehicles In this branching rule we force pickup offerings to be transported by given vehicles, that is, we create constraints in the form of $\langle i \to S \rangle$ where $i \in V_p$ and $S \subsetneq W$.

Let $C_{iv} := \sum_{r \in R_p : v_r = v} \xi_{ri} \bar{\mathbf{x}}_r$ denote the commitment of pickup point $i \in V_p$ to vehicle $v \in W$. Note that in case of integer solutions $C_{ij} \in \{0, 1\}$ holds, moreover, for each pickup point i there is at most one vehicle j with $C_{ij} = 1$. We say that a pickup point is committed to multiple vehicles (with respect to the fractional solution), if there are distinct vehicles v_1 and v_2 such that $0 < C_{i,v_1} < 1$ and $0 < C_{i,v_2} < 1$.

If there is no pickup point committed to multiple vehicles, then the branching rule cannot be performed. Otherwise, such a pickup point, say i, is chosen. Let $\langle i \to S \rangle$ be the strictest active constraint concerns to this pick-ip point where S is a set of vehicles. S is partitioned into subsets S' and $S \setminus S'$ such that $0 < \sum_{v \in S'} C_{iv} < 1$ and $0 < \sum_{v \in S \setminus S'} C_{iv} < 1$. Then, three branches are created with the following constraints $\langle i \to S' \rangle$, $\langle i \to S \setminus S' \rangle$, and $\langle i \to \emptyset \rangle$ respectively.

5.3.3 Follower branching

Assume that none of the previously introduced branching rules can be applied at the current search-tree node. It follows that in the current LP-solution each visited pickup point is committed to a single vehicle, to a single period, and to a single facility, however – since the solution is fractional –, some pickup points are committed to multiple routes belongs to the same period and the same facility, and performed by the same vehicle. When each pickup point is assigned to a vehicle, a facility, and a period, the problem is similar to a set partitioning problem (however, the sequence of the visited locations also matters), where the items correspond to the pickup points and sets correspond to routes. Thus, the well-known branching rule of Ryan and Foster (1981) could be applied, that is, two appropriate pickup points are chosen, then two branches are created forcing to visit those pickup points by the same route on the first branch and forcing to visit them by different routes on the other branch. However, handling these types of branching constraints are inconvenient in the solution approach for the pricing problems presented in Section 5.2, since it requires to introduce additional resources (may be with negative values) to the problem. For similar reasons Desrochers and Soumis (1989) modifies this branching rule to use in a crew scheduling problem.

For two distinct pickup points i and j let $Q_{ij} \subseteq \{r \in R_p : \xi_{ri} = \xi_{rj} = 1\}$ is the set of those routes where point j is visited immediately after point i. Define $C : V_p \times V_p \to [0, 1]$ as follows. For pickup points $i \neq j$ let $C_{ij} := \sum_{r \in Q_{ij}} \bar{\mathbf{x}}_r$. Note that in case of integer solutions $C_{ij} \in \{0, 1\}$ holds. Choose $(i, j) \in V_p \times V_p$ such that the value $|0.5 - C_{ij}|$ is minimal. Then, the following two branches are created

$$\text{0-branch: } \sum_{r \in Q_{ij}} \mathbf{x}_r = 0 \text{ and}$$

$$\text{1-branch: } \sum_{r \in Q_{ij}} \mathbf{x}_r \geq 1,$$

that is, forbidding to visit pickup point j immediately after pickup point i is visited on the 0-branch, and requiring that if either pickup point i or j is visited, then both of them have to be visited in this order, and no other points can be visited between them.

When a node is selected by the branch-and-price solver, these constraints can be easily endorsed. That is, in case of a 0-branch, each existing variable $\mathbf{x}_r$ of the corresponding restricted master problem can be locally fixed to zero if route r visits pickup point i and immediately after visits pickup point j. In case of a 1-branch, each existing variable $\mathbf{x}_r$ of the corresponding restricted master problem can be locally fixed to zero if route r visits exactly one pickup point of i and j, or visits both of them such that the immediately successor of i is not j.

When pricing is performed in a given search-tree node (see Section 5.2) these constraints also have to be taken into consideration. In case of a 0-branch, arc (v_i, v_j) is removed from the pricing graph. In case of a 1-branch, arcs (v_i, v_k) $(k \neq j)$ and arcs (v_k, v_j) $(k \neq i)$ are removed from the pricing graph.

5.4 Primal heuristics

In order to improve the upper bound, i.e., to find an integer feasible solution to the master problem, we apply a local-search based heuristics.

Assume that an integer feasible solution $(\tilde{\mathbf{s}}, \tilde{\mathbf{x}}, \tilde{\mathbf{y}})$ is found to the master problem. Then, we create a *schedule* which consists of three types of routes. First, the schedule contains *real routes* which refer to the routes used by this solution (i.e., routes r with $\tilde{\mathbf{x}}_r = 1$). Second, for each vehicle, for each period, and for each facility we also create a *pointless* pickup route and a *pointless* delivery route, that is, such a route refers to a vehicle itinerary where the vehicle visits the given facility on the given period but does not visit any pickup or delivery points. Finally, we also create a *fictive route* which does not belong to any vehicle or period or facility, but contains all of the uncovered pickup and delivery points (i.e., points $i \in V_p \cup V_d$ with $\tilde{\mathbf{y}}_i = 0$).

Each schedule is associated with an *evaluated value* which is basically the corresponding objective value plus some penalty costs for violating certain constraints. Namely, for a given schedule S with real routes $\tilde{R}$

$$\text{eval}(S) := \sum_{r \in \tilde{R}} \phi_r - \sum_{i \in \tilde{V}_p} \text{penalty}_i - \sum_{i \in \tilde{V}_d} \text{profit}_i + \lambda \times \Pi(S)$$

with

$$\Pi(S) := \sum_{v \in W} \left(\text{dlim}_v - \sum_{r \in \tilde{R}(t,v)} \sigma_r \right) + \sum_{i \in V_f} \sum_{t \in T} \left(\sum_{r \in \tilde{R}_p} \sum_{k \in P_1} \text{qlim}_i - \delta^+_{ikt} \right),$$

where $\tilde{V}_p$ and $\tilde{V}_d$ are the set of pickup and delivery points of real routes, respectively, and λ is a large constant.

We apply two types of operations to modify a schedule. In case of a *node relocation* a pickup or a delivery point is removed from its current position and placed to an other position, either on the same route or an other one. This operation allows to assign a point to a different vehicle, a different factory, or a different period. Relocating a point from a real route to the fictive route means that the corresponding request is no longer satisfied. In the reverse case, if a point is relocated from the fictive route to a real or pointless route, the corresponding request becomes satisfied. In case of a *node exchange* two points are exchanged with each other. Clearly, exchanging two nodes in the fictive route is superfluous.

To improve the current incumbent schedule S, we first repeatedly apply node relocations, that is, we go consider those schedules which can be resulted from S by a node relocation, and chose a one, say S', with minimal evaluated value. If $\text{eval}(S') < \text{eval}(S)$, i.e., the schedule is improved, we continue with schedule S', otherwise we stop since schedule S cannot be improved by node relocation. Then we do the same with node exchanges. If the schedule is improved at least once, we start a new iteration, otherwise we terminate the local-search procedure. If the final schedule, say S'', is feasible, that is, $\Pi(S'') = 0$, then $\text{eval}(S'')$ is a valid upper bound for the master problem, and thus can be set as a cutoff bound.

Table 4: Results for single-depot, single-good instances

#Facilities	#P-D	#Periods	#Vehicles	Time	Gap
1	8-8	8	16	7.53	0.002
		16	16	6.50	0.002
1	16-16	8	16	1354.2	0.001
		16	16	135.4	0.002
1	32-32	8	1	1200.0	0.209
			8	1200.0	0.035
			16	1059.7	0.012
1	32-32	16	1	1200.0	0.276
			8	1177.7	0.208
			16	1023.6	0.096
4	8-8	8	16	87.5	0.002
		16	16	43.8	0.002
4	16-16	8	16	896.6	0.038
		16	16	860.5	0.021
4	32-32	8	1	1200.0	0.223
			8	1102.0	0.124
			16	1106.4	0.361

6 Computational results

In this section we present the results of our computational experiments. The aim of these experiments was to examine how each parameter (e.g., number of facilities, pickup and delivery point, vehicles, etc.) affects the efficiency of the solving procedure.

The solution approach is implemented in C++ programming language using the SCIP Optimization Suite (Gamrath et al. (2020), version 7.0.1) as branch-and-price framework. In case of each execution we applied a gap limit set to 0.5 %, and a time limit set to 1200 seconds. All experiments were performed on a workstation with an i9-7960X 2.80 GHz CPU with 16 cores, under Debian 9 operating system using a single thread.

6.1 Instances

All the instances were randomly generated where we considered 1 depot, 1 good, 1-4 facilities, 8-32 pickup and 8-32 delivery points, 8-16 periods, and 1-16 vehicles. For each combination of these parameters we generate 15 instances with distinct random seeds.

6.2 Results

In Table 4 we summarize the results where each row corresponds to the average of 15 instances, and we depict the number of facilities (*#Facilities*), the number of pickup and delivery points (*#P-D*), the number of periods (*#Periods*), the number of vehicles (*#Vehicles*), the average time in seconds (*Time*), and the average final gap (*Gap*) calculated as

$$|\text{Cutoff bound} - \text{Dual bound}| \,/\, \min(|\text{Cutoff bound}|, |\text{Dual bound}|).$$

The main observation is that, as expected, the running time significantly increases as the number of pickup and delivery points increase. The average final gap in the bigger cases (that is, 32-32 points) is also higher than in case of smaller instances (that is, 8-8 points). Similarly, increasing the number of facilities also increase computational time.

In contrast, increasing the number of periods or the number of vehicles often helps to decrease the computational time and the average gap.

Acknowledgments

This research has been supported by the National Research, Development and Innovation Office, grant numbers SNN 129178, and TKP2021-NKTA-01.

References

G. Berbeglia, J.-F. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8–15, 2010.

K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99:300–313, 2016.

G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.

R. Cuda, G. Guastaroba, and M. G. Speranza. A survey on two-echelon routing problems. *Computers & Operations Research*, 55:185–199, 2015.

G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

M. Desrochers and F. Soumis. A column generation approach to the urban transit crew scheduling problem. *Transportation science*, 23(1):1–13, 1989.

B. Eksioglu, A. V. Vural, and A. Reisman. The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57(4):1472–1483, 2009.

G. Gamrath, D. Anderson, K. Bestuzheva, W.-K. Chen, L. Eifler, M. Gasse, P. Gemander, A. Gleixner, L. Gottwald, K. Halbig, G. Hendel, C. Hojny, T. Koch, P. Le Bodic, S. J. Maher, F. Matter, M. Miltenberger, E. Mühmer, B. Müller, M. E. Pfetsch, F. Schlösser, F. Serrano, Y. Shinano, C. Tawfik, S. Vigerske, F. Wegscheider, D. Weninger, and J. Witzig. The SCIP Optimization Suite 7.0. Technical report, Optimization Online, March 2020.

D. Ryan and E. Foster. An integer programming approach to scheduling. 1981.

N. Sluijk, A. M. Florio, J. Kinable, N. Dellaert, and T. Van Woensel. Two-echelon vehicle routing problems: A literature review. *European Journal of Operational Research*, 2022.

P. Toth and D. Vigo. *The vehicle routing problem*. SIAM, 2002.

H. Zhang, H. Ge, J. Yang, and Y. Tong. Review of vehicle routing problems: Models, classification and solving algorithms. *Archives of Computational Methods in Engineering*, 29(1):195–221, 2022.