The EVRP-TW with Heterogeneous Recharging Stations

An Exact Branch-and-Price Method

By

JACOPO PIEROTTI

Dipartimento di Elettronica, Informazione e Bioingegneria - DEIB
POLITECNICO DI MILANO

A dissertation submitted to the Politecnico di Milano in accordance with the requirements of the degree of MASTER OF SCIENCE in the Faculty of Automation and Control Engineering.

OCTOBER 2017

Abstract

Effective route planning for battery electric commercial vehicle (ECV) fleets has to take into account their limited autonomy and the possibility of visiting recharging stations with different technologies during the course of a route. In this thesis, I consider three variants of the electric vehicle-routing problem with hard time windows: (i) at most a single recharge per route is allowed, and batteries are fully recharged on visit of a recharging station; (ii) multiple recharges per route, full recharges only and (iii) at most a single recharge per route, and partial battery recharges are possible. For each variant, I present exact branch-price-and-cut algorithms that rely on customized monodirectional forward labeling algorithms for generating feasible vehicle routes. The main point of the implemented algorithms are the tailored resource extension functions (REFs) that enable efficient labeling with constant time feasibility checking and strong dominance rules, even if these REFs are intricate and rather elaborate to derive.

ACKNOWLEDGEMENTS

Acknoledgments go to the ETS and the GERAD group for letting me work with them, for the facilities and for the friendly yet stimulating environment, especially thanks to Prof. Guy Desaulniers for his guidance.
Many thanks to Prof. Ceselli Alberto for the useful practical advises.
Last but not least, my whole gratitude from the bottom of my heart goes to Professor Fausto Errico and to Professor Federico Malucelli without whom none of this document would have been possible.

Author's declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's regulations for research and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: .... DATE: ....

Dedication

non al denaro
non all'amore
ne' al cielo
(F.d.A)

Table of Contents

Page

List of Tables

Table **Page**

List of Figures

CHAPTER 1

INTRODUCTION

The utilization of battery electric commercial vehicles (ECVs) is steadily increasing, e.g., in the field of small package shipping or the distribution of food, beverages and light goods. This increase of ECV usage is due mainly to the following advantages:

(i) ECVs produce minimal noise and (almost) zero battery-to-wheel greenhouse gas emissions. Therefore, they can be employed to meet even the most strict emission targets of delivery fleets or to serve restricted inner city areas with noise and emission limits. Under certain country they can also perform by-night delivery thanks to the low-noise pollution.

(ii) ECVs help logistics companies to promote a green image, an important competitive factor given the increasing number of socially and environmentally aware customers. Moreover, relative autonomy from fluctuating oil prices can be achieved.

(iii) ECVs become attractive from a cost perspective because of heavy subsidies offered by several governments around the globe. In addition, governments and private companies are strongly investing to provide the required recharging infrastructure around the world.

1.1 Overview

Effective route planning of an ECV fleet requires solving vehicle-routing problems (VRPs) that take into account the limited driving range of ECVs and the possibility of visiting recharging stations during the course of a route. Several heuristic solution methods for such VRPs have recently been proposed in the literature. However, to the best of our knowledge, only one exact solution method (considering heterogeneous recharging stations) has been presented by Ceselli et al. ((2015)).

In this master thesis, we develop three branch-price-and-cut algorithms for as many variants of the electric VRP with time windows (EVRPTW). The following EVRPTW variants are addressed:

(i) At most a single (S) recharge per route is allowed, and batteries are fully (F) recharged on visit of a recharging station (EVRPTW-SF).

(ii) multiple (M) recharges per route, full (F) recharges only (EVRPTW-MF).

(iii) At most a single (S) recharge per route, and partial (P) battery recharges are possible (EVRPTW-SP).

We solved the three EVRPTW variants with branch-price-and-cut algorithms. Branch-price-and-cut means that an extensive formulation (a set-partitioning model in our case) is linearly relaxed, the relaxation is solved using column generation and integer solutions are finally enforced by branching. The solution of the master program, i.e., the linear relaxation of the extensive formulation, starts with restricting the master to a small subset of variables. The optimization of this restricted master problem (RMP) provides the necessary dual information needed to generate missing variables (columns) for the RMP itself. In our problem, as in many extensive formulations for vehicle routing and crew scheduling problems, the generation of variables uses a path representation of routes so that the column-generation subproblem is a variant of the elementary shortest-path problem with resource constraints (ESPPRC). The column-generation process alternates between RMP reoptimization and solution of the ESPPRC until no more columns with a negative reduced cost exist.
While the master program for EVRPTW is standard, the three variants give rise to different ESPPRCs. I think that the contribution of this thesis lies in the concise formulation of the different ESPPRC variants so that highly effective solution techniques can be applied. An important aspect is the modeling with as few as possible attributes (resource variables) in such a way that dominance rules allow the elimination of the majority of the partial paths constructed in the course of the ESPPRC labeling algorithm. For the variant with partial recharge (SP), there is an immanent trade-off between the amount recharged and the time spent for recharging: longer recharging extends the driving range while it may prohibit the timely arrival at a later customer because of its service time window. Therefore, we require a label that models this trade-off curve. The proposed methodology provides insights for the design of exact algorithms for more realistic versions of electric vehicle-routing problems and, more generally, for VRPs that consider time windows, a limited resource that can be refreshed en route, and in which the refreshing consumes time that depends on the amount to be refreshed.

1.2 Literature Review

Many others authors tackled similar problem both from a more "mathematical" point of view and an "economic" one.

The latter is extremely well explained and detailed in Margaritis et al. ((2016)) and Pelletier et al. ((2014)), which shows us the reasons why a company should invest in green transportation even though the (apparently) economic inconvenience. The authors also comment on the different policy of nations addressing this problem, especially in the EU area and, on the other hand, they also display some market difficulties as companies not willing to take risks due to the high initial cost of electrical vehicles.
From the more "mathematical" point of view many authors tackled variants related to the electric VRP. Hiermann et al. ((2016)) proposed a ALNS approach to solve the electric fleet size and mix vehicle routing problem with time windows and recharging stations, considering only full recharges. Also Keskin and Çatay ((2016)) proposed an ALNS to solve the electric vehicle routing problem but they took into account the possibility to partially recharge the vehicles.
Another widely discussed trend is to take into account non-linear recharging functions as Sweda et al. ((0a)) and Schiffer and Walther ((2017)) did. This of course complicates the model yet it gives a more detailed description of the real problem. Unfortunately, due to the complicated model, none of them could find exact method to solve medium-sized instances.
Some authors (Montoya et al.) took in consideration both of the previous aspect (partial as well as non-linear recharges) trough the use of a modified multi-space sampling heuristic (mMSH); while other authors as Felipe et al. ((2014)) focused more on allowing partial recharges in instances with heterogeneous recharging stations.
Roberti and Wen ((2016)) proposed extremely effective heuristic for the TSP with Time Windows capable of solving 20 costumers instances in about $\frac{1}{10}$ of a second, achieving good performances also in 200 costumers instances, while Gualandi and Malucelli ((2016)) proposed a filtering algorithm to reduce the graph's dimension in a resource constrained shortest path problem. Additionally Gualandi and Malucelli ((2012)) also proposed an exact solution approach to the constrained shortest path problem with a super additive objective function, which could come in handy when we will consider non linear recharging functions or the additional cost due to overcharging.
Contrarily to what Sweda et al. ((0b)) said in their paper "We assume that all charging stations have identical cost function, which is a reasonable assumption since most public charging stations in existence today have similar hardware configurations (most recharge at 220 volts, with the exception of a few fast charging stations that recharge at 440 volts). Furthermore, regional variations in electricity rates are minimal" we decide to take into account the possibility of allowing different types of recharges due to the huge difference (in cost but especially in time) between the traditional electric recharging stations and the so-called "super-charging station".
Other authors took in consideration completely different way of recharging an electrical vehicle, we would like to bring attention to Goeke et al. ((2015)) that proposed a formulation for the so called battery swamp problem. The battery swamp is the act of removing a battery, at a specialized recharging station, to substitute it with a completely charged one. This procedure is

interesting because it allows for a short and constant time to recharge, yet it has some drawbacks as the need to standardize batteries as well as the need to have an easy access to the battery for removal/inserting operations.

Again, as stated before, only Ceselli et al. ((2015)) proposed an exacted method to solve the EVRP-TW with Heterogeneous Recharging Stations, yet the results, especially from a time point of view are not completely satisfying. Finally Desaulniers et al. ((2016)) proposed an effective and exact branch-and-price algorithm capable of solving big instances in the electric homogeneous case. To the best of my knowledge these are the only authors proposing exact methods.
Finally, in table (1.1) we present a summary of the main variants treated in the cited papers.

Summary					
Author	Exact	Partial	Multiple	Non Linear	Heterogeneous
Hiermann et al. ((2016))	X	X	X	X	X
Keskin and Çatay ((2016))	X	✓	✓	X	X
Sweda et al. ((0b))	X	X	✓	✓	X
Schiffer and Walther ((2017))	X	X	✓	✓	X
Montoya et al.	X	✓	X	✓	X
Felipe et al. ((2014))	X	✓	✓	X	✓
Ceselli et al. ((2015))	✓	✓	✓	X	✓
Desaulniers et al. ((2016))	✓	✓	✓	X	✓

Table 1.1: Literature: Summary

1.3 General Comment

BEV are expected to have much development in the following years, thanks to many advantages they have with respect to traditional ICE. Almost all authors underline that the companies' decision to switch from ICE to electric logistic is moved by an anticipation of expected regulations for less environmentally-friendly vehicles becoming more restrictive in the future rather than for an economic reason. This may be imprecise if we consider that typically electrical motor requires much less maintenance than ICE and, despite the high initial cost, in the long run the low (and stable) cost of electric energy make the electric vehicle cost convenient with respect to the higher (and less stable) cost of oil. As Savaresi ((2016)) says, buying an electrical vehicle could be though as buying a discounted traditional vehicle with an enormous virtual tank, it pays off in the long run but has an high initial cost.
Many authors refers independently to electric or green problem, but it's worth to say that electric does not imply "green". For instance Poland electric system is still based on large coal burning furnace, returning to the atmosphere roughly 80% of how much traditional ICE would pollute

during electricity production. This aspect is cited by Pelletier et al. ((2014)) that say "while electric delivery vehicles can significantly decrease greenhouse gas emissions in areas with clean electricity generation sources, this environmentally based business case can be harder to make with more coal intensive production, in which case local pollution is simply replaced by upstream emissions"
Finally we want to point out that no author takes into account in the objective function the real cost. Most take into account distance, few time, stating that this quantities are directly proportional to the cost. This assumption indeed does not hold true in the case of heterogeneous recharging stations.

Chapter 2

Problem Description and Mathematical Formulation

To solve the EVRPTW with Heterogeneous Recharging Stations we use a "path-based formulation" as similar as possible as the one presented in Desaulniers et al. ((2016)).
In particular we exploit the very same structure of the network and formulation of the master problem, which are:

2.1 Network Structure

Let N be the set of customers that all require deliveries (collection from all customers is identical). Denote by q_i the demand of customer $i \in N$ and by $[a_i, b_i]$ the time window in which service has to start at this customer. A vehicle can arrive at a customer before the opening of its time window and wait to start service. We assume an unlimited fleet of identical ECVs with a storage capacity of Q and a battery capacity of B. At the beginning of the planning horizon, the ECVs are located in a single depot from which they start fully charged and to which they must return by the end of the planning horizon.
The vehicles shall start fully charged because it is possible to demonstrate that, given an optimal solution at these types of problems: or all vehicles start fully charged, or there exists another solution, with the same cost (thus optimal as well), where vehicles start fully charged. Let R be a set of recharging stations at which the vehicles can stop en route to recharge their battery, each recharging station can be equipped with different technologies. We assume that the battery recharging time is proportional to the amount of energy recharged through a coefficient depending on the used technology. Traveling from one location i (the depot, a customer or a recharging station) to another location j incurs a cost c_{ij} , a travel time t_{ij} (that includes service time at i if $i \in N$), and an energy consumption b_{ij} . We assume that a energy consumption proportional with

the distance is suitable.
In general this assumption is suitable for all types of transportation where the speed/battery consumption is not depending on the load. as [Maximilian Schiffer and Grit Walther] said "we assume vehicle speed to be constant and neglect differences in altitude ". Also according to Berman and Gartner (2013) of Navigant, approximately 37,000 BEVs and PHEVs were expected to be sold for fleet purposes in 2013. Some of these vehicles are used in the delivery of lighter goods. For example, battery electric cars have been tested in pizza delivery operations in Hamburg (E-Mobility NSR 2013). Since the load is light there is no significant consumption difference due to it. Indeed ECVs are more and more common in last-mile delivery distribution, for example in small packages shipping or in the distribution of food and beverages, and several companies have started deploying ECVs for their daily operations (see FedEx, 2010; Motavalli, 2010).
Basically this is to say that whenever the load is small with respect to the vehicle weight, we can neglect the battery consumption's dependency on the load.

A vehicle route is a sequence of locations that starts and ends at the depot and visits a nonempty subset of customers and possibly some recharging stations. Its cost is given by the sum of the travel costs c_{ij} between the pairs of consecutive locations i and j that it visits and the sum of the costs of the recharges.
A route is feasible if:

(i) it is elementary with respect to the customers (in some policies recharging stations may be visited more than once)

(ii) the total demand of the visited customers does not exceed the vehicle capacity

(iii) the battery charge level is always non negative along the route

(iv) the customer time windows are respected.

2.2 Master Problem

The EVRPTW consists of finding a set of feasible vehicle routes such that each customer $i \in N$ is visited exactly once by a vehicle and the sum of the route costs is minimized. Let Ω be the set of feasible routes. For each route $p \in \Omega$, denote by c_p its cost and by a_{pi} , $i \in N$, a binary parameter equal to 1 if route p visits customer i and 0 otherwise. With each route $p \in \Omega$, we associate a binary variable θ_p that takes value 1 if the route is part of the solution and 0 otherwise. Using this notation, the EVRPTW can be formulated as the following integer program:

$$\min \sum_{p\in\Omega} c_p\theta_p \tag{2.1}$$

$$s.t. \sum_{p\in\Omega} a_{pi}\theta_p = 1, \forall i \in N \tag{2.2}$$

$$\theta_p \in \{0,1\}, \forall p \in \Omega \tag{2.3}$$

Objective function (2.1) seeks to minimize total routing costs. Set-partitioning constraints (2.2) ensure that each customer $i \in N$ is visited exactly once by a vehicle. Binary requirements (2.3) restrict the domain of the route variables.
In practice, model (2.1) contains a huge number of variables, namely, one per feasible route in Ω. This number prohibits using a standard MIP solver or branch-and-bound algorithm for solving it. That's why we use a Branch-Price-and-Cut Algorithm, generating only promising routes.

2.3 Branch-Price-and-Cut Algorithm

To solve the set-partitioning model (2.1), we develop a branch-price-and-cut algorithm for each problem variant. Because the procedure that generates the routes very much depends on the EVRPTW variant, we will mainly focus on this aspect in chapter (3), while branching is discussed in section (4.2).

2.3.1 Column Generation

In this section, we focus on the initial linear relaxation of the extensive formulation (2.1), i.e., without cuts or branching decisions. Recall that the subproblem aims at generating negative reduced cost columns (route variables) to be added to the current RMP. If no such columns exist, the algorithm stops and the computed solution to the current RMP is also optimal for the complete linear relaxation.
For model (2.1), the column generation subproblem can be defined as follows. Let π_i for i ∈ N be the dual variables associated with constraints (2.2). Let $\overline{c_p}$, p ∈ Ω be the reduced cost of variable θ_p with respect to these dual variables, i.e., $\overline{c_p} = c_p - \sum_{i\in N} a_{pi}\pi_i$. The subproblem can be stated as:

$$\min_{p\in\Omega} \overline{c_p} \tag{2.4}$$

The set of feasible routes in Ω can be implicitly represented in a directed graph G=(V,A) with vertex set V and arc set A. The vertex set V is given by $V = \{o,d\} \cup N \cup R$, where o is a source and d a sink vertex, both associated with the depot. Demand q_i and time windows $[e_i, l_i]$ are associated with each vertex i ∈ N. For all vertices i ∈ $R \cup \{o,d\}$, we define $q_i = 0$ and associate a nonrestrictive time window (note that our algorithms can easily be adapted to restrictive time

windows). The arc set A contains all arcs (o,j) and (j,d) with $j \in N \cup R$ and all arcs $(i,j) \in (N \cup R)^2$ with $i \neq j$. With each arc $(i,j) \in A$, we associate the cost c_{ij} , the travel time t_{ij} that includes the service time at i if $i \in N$. Given these parameters, some arcs can be removed from A because they cannot be part of a feasible route, namely, the arcs (i,j) with $q_i + q_j > Q$, or $e_i + t_{ij} > l_j$. In the case of single recharge (SR) we can also filter each arc going from a recharging station to another recharging station.
Even though we achieve to filter some arcs, the resulting graph is almost complete and there are no efficient filtering strategies due to the fact that is (almost) always possible to find a feasible path including the generic arc (i,j).
We assume that the cost, travel time, and required recharging time matrices satisfy the triangle inequality. These hypothesis are important when we will check the reachability/unreachability of some nodes from an existing label. More detail on that in paragraph (3.1.1).
A feasible route in Ω corresponds to a path in G starting and ending at the depot node, in which any vertex $i \in N$ is visited at most once, i.e., elementarity is respected for the customer vertices. However, not all elementary depot-depot paths in G correspond to feasible routes as they may violate the time windows, the vehicle capacity, or the battery capacity. Additional constraints on the paths are therefore required to ensure that they represent feasible routes. We will define such resource constraints in section (3).
Subproblem (2.4) aims at finding a feasible route with minimum reduced cost. To compute the reduced cost of each depot-depot path in G, we replace the arc cost c_{ij} for each arc (i,j) ∈ A by a modified cost $\overline{c_{ij}} = c_{ij} - \pi_i$, where we set $\pi_i = 0$ if $i \in R \cup \{o,d\}$. Then, the sum of the modified costs $\overline{c_{ij}}$ of the arcs (i,j) of a path p is equal to its reduced cost $\overline{c_p}$. In this setting, the subproblem corresponds to an ESPPRC, in which elementarity is imposed only on the customer vertices and the path length is measured with respect to the modified arc costs $\overline{c_{ij}}, (i,j) \in A$. Each EVRPTW variant induces a specific subproblem. The two single-recharge variants require a resource constraint to ensure that at most one vertex in R is visited in a path. Furthermore, because full battery recharges are more restrictive than partial battery recharges, these two recharge types must be handled differently. In consequence, we consider three variants of the subproblem called ESPPRC-SF, ESPPRC-MF and ESPPRC-SP hereafter.
The ESPPRC on graph G will be solved by dynamic programming using a labeling algorithm. In this algorithm, labels are used to represent partial paths that start at the origin vertex o. Starting from an initial label associated with vertex o, paths are constructed iteratively by extending this label and its descendants forwardly in G. The extension of a label along an arc is performed using REFs (resource extension function). Each generated label is checked for feasibility with respect to the resource constraints and infeasible labels are discarded.
Furthermore, to avoid enumerating all feasible complete and partial paths, a dominance criterion is applied to eliminate partial paths for which no completion to full path could possibly lead to any improvement. These dominance rules are likely to be the most interesting (from a conceptual

point of view) contributions of this thesis. In the following, we propose monodirectional forward labeling algorithms for all subproblem variants. We will focus on the label components, the REFs, and the dominance rules.

CHAPTER 3

LABELING ALGORITHM

3.1 Single Recharge - Full Recharge

A partial path p from $depot$ to generic node i is associated with the following label: $L_i = (T_i^{cost}, T_i^{load}, T_i^{rch}, T_i^{time}, (T_i^{cust_n})_{n \in N}, T_i^{battery})$ where the label components are defined as follows:
T_i^{cost} : reduced cost of path p.
T_i^{load} : total load delivered along path p.
T_i^{rch} : number of recharges performed along path p.
T_i^{time}: earliest service time at vertex i.
$(T_i^{cust_n})_{n \in N}$: number of times that customer $n \in N$ is visited along path p. Also set to 1 if customer n is not visited but it's unreachable from p.
$T_i^{battery}$: battery consumption
In the initial label at vertex o, the depot node, all components are set to 0 except for T_i^{time} which is set to e_0. All the component are set to 0 because we could prove that, given an optimal solution in which the label at the depot has not all the components set to zero[1], there exist at least another optimal solution with equal or smaller objective function if all the components of the initial label have been set to zero. This is straightforward if we think of the actual meaning of the resource. There is no case where starting a route with an already consumed battery could be more performing that starting a route with a fully load battery.

[1]note that, due to the meaning of the labelling component/resource no quantity could go below zero.

3.1.1 Single Recharge - Full Recharge: REFs

The extension of a label L_i along an arc (i,j) is done according to the following extension functions. Note that we considered a variable $rate_j^{cost}$ equal to the cost of the recharging rate if $j \in R$ or equal to zero otherwise and a variable $rate_j^{time}$ that represent the time dynamic associated with the recharging station.

$$T_j^{cost} = T_i^{cost} + \overline{c_{ij}} + T_j^{battery} rate_j^{cost}$$

$$T_j^{battery} = \begin{cases} T_i^{battery} + b_{ij} & i \notin R, \\ b_{ij} & i \in R \end{cases}$$

$$T_j^{load} = T_i^{load} + q_j$$

$$T_j^{rch} = T_i^{rch} + \begin{cases} 1 & j \in R \\ 0 & otherwise \end{cases}$$

$$T_j^{time} = max(e_j, T_i^{time} + t_{ij} + T_i^{battery} rate_i^{time})$$

$$T_j^{cust_n} = \begin{cases} T_i^{cust_n} + 1 & n = j \\ max(T_i^{cust_n}, U_n^{fw}(T_j^{load}, T_j^{time})) & otherwise \end{cases} \quad \forall \text{ n} \in \text{N}.$$

Please note that the function U_n^{fw} is defined as follow

$$U_n^{fw}(T_j^{load}, T_j^{time}) = \begin{cases} 1 & \text{if } T_j^{load} + q_n > Q \vee T_j^{time} + t_{j,n} > \ell_n, \\ 0 & otherwise \end{cases}$$

Actually it could be specified with more details adding the following conditions:

$$U_n^{fw} = \begin{cases} 1 & \text{if } T_j^{rch} = 1 \wedge j \notin R \wedge T_j^{battery} + b_{j,n} > B \\ 1 & \text{if } T_j^{rch} = 0 \wedge T_j^{battery} + b_{j,n} > B \wedge [(T_j^{battery} + b_{j,r} > B, \forall r \in R) \vee \\ & \vee (\min_{\forall r \in R}((T_j^{battery} + b_{j,r}) rate_r^{speed} + T_j^{time} + tj,r + tr,n) > l_n)] \end{cases}$$

Considering that we are in the case *single recharge - full recharge*, the first condition ensures us that if the vehicle has already recharged, thus it can't recharge anymore, and the travel from j to n exceeds its battery capacity, costumer n is considered unreachable.
The second condition instead ensures that if the vehicle hasn't recharged yet, but its battery isn't enough to reach the n_{th} costumer nor it's enough to reach the nearest recharging station, or it's just enough to reach it but recharging would mean to spend too much time, thus it would lead to a violation of the time window, then the n_{th} costumer is considered unreachable.
These considerations could be expanded considering that after visiting the n_{th} costumer the vehicle has to be able to *at least* reach the depot, however I'm not sure if this more complicated

$U_n^{fw}(T_j^{load}, T_j^{time}, T_j^{rch}, T_j^{battery})$ function would lead to any concrete improvement of the labelling algorithm.
In our simulations we considered only the following condition to check unreachability:

$$U_n^{fw}(T_j^{load}, T_j^{time}) =$$
$$\begin{cases} 1 & \text{if } T_j^{load} + q_n > Q \vee T_j^{time} + t_{j,n} > \ell_n \vee (T_j^{rch} = 1 \wedge j \notin R \wedge T_j^{battery} + b_{j,n} > B) \\ 0 & \text{otherwise} \end{cases}$$

We can see how the triangular assumption on the arcs is fundamental. If that didn't hold we could not check unreachability in this simple way, but we should check it resolving a shortest path problem for each couple of nodes and that would be way more time-consuming.

3.1.2 Single Recharge - Full Recharge: Label Feasibility

A Label $L_i = (T_i^{cost}, T_i^{load}, T_i^{rch}, T_i^{time}, T_i^{battery}, T_i^{cust_n})$ is considered feasible if **all** the following conditions hold:

$$\begin{aligned} T_i^{load} &\leq Q \\ T_i^{rch} &\leq 1 \\ T_i^{time} &\leq l_i \\ T_i^{battery} &\leq B \\ T_i^{cust_n} &\leq 1 \forall n \in N \end{aligned}$$

3.1.3 Single Recharge - Full Recharge: Dominance Rule

In order to demonstrate that the following dominance rule holds true:

Given two labels L_1^i, L_2^i such that both point to the very same node i [2] both structured as $L_k^i = (T_{i,k}^{cost}, T_{i,k}^{load}, T_{i,k}^{rch}, T_{i,k}^{battery}, T_{i,k}^{time}, T_{i,k}^{cust})$ for k=1,2 ;
if $T_{i,1}^{any} \leq T_{i,2}^{any}$ for $any \in$(cost,load,rch,battery,time,cust), and L_1^i different form L_2^i then we can say that L_1^i *dominates* L_2^i, we will write it $L_1^i \preccurlyeq L_2^i$, and thus discard L_2^i

we have firstly to show that the REFs are non-decreasing functions, and then to prove that any possible extension of $path_2$ is also a possible extension of $path_2$.

[2] this should be more specified. With very same node we mean the actual node i, for i $\notin$ R.
If i $\in$ R then the dominance rule is applied to all the dummy nodes simultaneously considered as one

3.1.4 Proof that REFs are non-decreasing functions

In order to demonstrate that the REFs are non-decreasing functions we examine every REF one by one.
Remember that our initial hypothesis is that $T_{i,1}^{any} \leq T_{i,2}^{any}$ for $any \in$(cost,load,rch,battery,time,cust) and our thesis is that $T_{j,1}^{any} \leq T_{j,2}^{any}$ for $any \in$(cost,load,rch,battery,time,cust).
Note that T_j is the expansion of T_i through arc (*i*,*j*).
Let's now examine our first REF:

1. We know that:

$$T_j^{cost} = T_i^{cost} + \overline{c_{ij}} + T_j^{battery} rate_j^{cost}$$

   since $\overline{c_{ij}}$ and $rate_j^{cost}$ are equal for both labels, it's straightforward to notice that a *sufficient* condition to verify $T_{j,1}^{cost} \leq T_{j,2}^{cost}$ is that $T_{j,1}^{battery} \leq T_{j,2}^{battery}$
   Now we check if the latter holds.

2. The REFs referred to the battery is:

$$T_j^{battery} = \begin{cases} T_i^{battery} + b_{ij} & i \notin R, \\ b_{ij} & i \in R \end{cases}$$

   It's easy to note that condition
   $T_{i,1}^{battery} \leq T_{i,2}^{battery}$ implies $T_{j,1}^{battery} \leq T_{j,2}^{battery}$, leading to
   $T_{i,1}^{battery} \leq T_{i,2}^{battery} \bigwedge T_{i,1}^{cost} \leq T_{i,2}^{cost} \rightarrow T_{j,1}^{cost} \leq T_{j,2}^{cost}$, which fulfills our former REF.

3. the load REF is

$$T_j^{load} = T_i^{load} + q_j$$

   it's obvious that $T_{i,1}^{load} \leq T_{i,2}^{load} \rightarrow T_{j,1}^{load} \leq T_{j,2}^{load}$

4. Another obvious non-decreasing extension is the following

$$T_j^{rch} = T_i^{rch} + \begin{cases} 1 & j \in R \\ 0 & otherwise \end{cases}$$

5. while the time REF is:

$$T_j^{time} = max(e_j, T_i^{time} + t_{ij} + T_i^{battery} rate_i^{time})$$

   since $T_{i,1}^{battery} \leq T_{i,2}^{battery} \bigwedge T_{i,1}^{time} \leq T_{i,2}^{time} \rightarrow T_{j,1}^{time} \leq T_{j,2}^{time}$

6. the last REF is , for any n∈N

$$T_j^{cust_n} = \begin{cases} T_i^{cust_n} + 1 & n = j \\ max(T_i^{cust_n}, U_n^{fw}(T_j^{load}, T_j^{time})) & otherwise \end{cases}$$

Please note that the function U_n^{fw} is defined as follow

$$U_n^{fw}(T_j^{load}, T_j^{time}) =$$
$$\begin{cases} 1 & \text{if } T_j^{load} + q_n > Q \vee T_j^{time} + t_{j,n} > \ell_n \vee (T_j^{rch} = 1 \bigwedge j \notin R \bigwedge T_j^{battery} + b_{j,n} > B) \\ 0 & \text{otherwise} \end{cases}$$

This ensure the non-decreasing property of this last label.

Thus we proved that the set $T_{i,1}^{any} \leq T_{i,2}^{any}$ for *any* ∈(cost,load,rch,battery,time,cust) is a *sufficient* set of hypothesis to guarantee the REFs function to be non-decreasing; the next step is to demonstrate that every possible expansion of $path_1$ is also a possible extension of $path_2$ as well.

3.1.5 Proof on possible extensions

In order to prove this we begin considering that an extensions is unfeasible if and only if it violates at least one of the following constraints:

$$T_i^{load} \leqslant Q,$$
$$T_i^{rch} \leqslant 1,$$
$$T_i^{time} \leqslant \ell_i \ ,$$
$$T_i^{cust_n} \leqslant 1 \ ,$$
$$T_i^{battery} \leqslant B$$

but given our hypothesis that $T_{i,1}^{any} \leq T_{i,2}^{any}$ for *any* ∈(cost,load,rch,battery,time,cust) and the non-decreasing property of the REFs, it follows that $T_{j,1}^{any} \leq T_{j,2}^{any}$. Thus if label $L_{j,1}$ violates any of the previously constraints $L_{j,2}$ must violate them as well.
Thus we can infer that any non unfeasible expansion $L_{j,2}$ entails the non unfeasibility of $L_{j,1}$, hence every feasible extension of the second path is feasible as well for the first path.
In a more formal logical language (with intuitive meaning of the variables) we may write:

$$\neg L_{i,1} \rightarrow \neg L_{i,2}$$
$$\neg\neg L_{i,2} \rightarrow \neg\neg L_{i,1}$$
$$L_{i,2} \rightarrow L_{i,1}$$

3.1.6 conclusion

Since we were able to demonstrate that the REFs are non-decreasing functions and that given a proper set of hypothesis the possible extensions of $path_2$ are included in the possible extensions of $path_1$, we can safely state that the following dominance rule apply:

Given two labels L_1^i, L_2^i such that both point to the very same node i and both structured as $L_k^i = (T_{i,k}^{cost}, T_{i,k}^{load}, T_{i,k}^{rch}, T_{i,k}^{battery}, T_{i,k}^{time}, T_{i,k}^{cust})$ for k=1,2 ;
if $T_{i,1}^{any} \leq T_{i,2}^{any}$ for $any \in$(cost,load,rch,battery,time,cust), and L_1^i different form L_2^i then we can say that L_1^i *dominates* L_2^i, we will write it $L_1^i \preccurlyeq L_2^i$, and thus discard L_2^i

What said (but for the possible modification on U_n^{fw}) holds true even for the multiple recharge case, once we properly remove the conditions on T_i^{rch}.

Finally we would like to specify that in our simulation we took in consideration a depot without cost of recharging ($rate_{cost} = 0$), this is reasonable if we consider that a green logistic company is likely to have a "flat" cost rate for electricity.

3.2 Single Recharge - Partial Recharge

3.2.1 Formal Model for the SR-PR

For the partial recharge case the main difficulty to overcome is that the amount of energy to be recharged has to be determined a posteriori. Therefore the earliest service start time at a given vertex j following a recharge station becomes a linear function of the amount of energy recharged. To overcome this issue we may use the following label algorithm, whose main idea is to keep track of the earliest and latest time for the service to start and of the "slack time", then progressively modify them in order to ensure feasibility without violating the battery capacity constraint.
Another important difference with respect to the homogeneous case is that in the latter, every time we had some "slack time" (formal definition in the following) we supposed to use that time to recharge our vehicle while in the heterogeneous case we can't. A way to avoid this issue is to formulate the problem in terms of cumulative quantities, as we shall see in the following.

To overcome these problems we adopt the following label to describe a path p, starting from vertex o and ending in vertex i:
$L_i = (T_i^{cost}, T_i^{load}, T_i^{rch}, T_i^{time_{Min}}, T_i^{time_{MAX}}, T_i^{slack}, T_i^{ToBeR}, T_i^{battery}, T_i^{cust_n})$

3.2.2 Label Definitions

The definitions and the REFs of components: $T_i^{load}, T_i^{rch}, T_i^{cust_n}$ are identical to the previous ones, while $T_i^{cost}, T_i^{battery}$ keep the same definition but change their REFs.
Now we provide definitions for the remaining components:
$T_i^{time_{Min}}$: Earliest service Time at Vertex i assuming that, if a recharging station has been visited prior i, a minimum recharge, just to ensure battery feasibility up to i has been performed.
$T_i^{time_{MAX}}$: Earliest service Time at Vertex i assuming that, if a recharging station has been visited prior i, a maximum recharge that still ensure time window feasibility up to i has been performed.
T_i^{slack} : this variable represents the (cumulative) amount of time the vehicle has to wait, if any, from when it arrives at a costumer until its time window starts, considering only the costumers visited after a recharge.
T_i^{ToBeR} : this variable instead represents the (cumulative) amount of battery capacity the vehicle has to have recharged in order to ensure battery feasibility up to i

Note that if no recharging station has been visited along path p
$T_i^{rch} = 0 \rightarrow T_i^{time_{MAX}} = T_i^{time_{Min}} \wedge T_i^{slack} = 0$.
And it's also true that $T_i^{rch} = 0 \wedge p\ is\ feasible \rightarrow T_i^{ToBeR} = 0$

The initial label at root node will be composed by all components set to 0 but for $T_i^{time_{MAX}} = T_i^{time_{Min}} = e_o$.

3.2.3 REFs

The following REFs are then applied:
For $T_i^{load}, T_i^{rch}, T_i^{cust_n}$ the REFs are identical to the previous ones, while:

$$T_j^{battery} = \begin{cases} T_i^{battery} + b_{i,j} & if T_i^{rch} = 0 \\ min(T_i^{battery} + b_{i,j}, B)) & otherwise \end{cases}$$

Please note that the cases are defined if $T_i^{rch} = 0$, not T_j^{rch}. Also this definition takes advantage that being in the single partial recharge case visiting a recharging station doesn't ensure that we will actually recharge anything. In fact it's most likely that after we reach the maximum capacity of the battery at a generic node i all labels forward will have $T_j^{battery} = B$, since we will recharge just enough to arrive at node j.
It is possible to create examples in which we visit a recharge station with no actual need to recharge, and discover that we had to had performed a recharge only when we visit the last node, the depot. That's why we need to consider also the cases where we visit a recharging station even

if there is no urgent need to recharge.

The REF of T_i^{ToBeR} is defined as follows:

$$T_j^{ToBeR} = T_i^{ToBeR} + max(0, T_i^{battery} + b_{i,j} - B)$$

Note that this variable will assume value 0 until there is no need for a recharge, then it will assume the (cumulative) value of how much energy we shall have recharged in order to arrive up to j. Also note that, with an improper but intuitive language we can say that, thanks to the previous consideration we know that once our battery will be empty at a given generic node i, $T_i^{battery}$ is likely to assume value B for all nodes j following i , so T_j^{ToBeR} is likely to increase by $b_{i,j}$ each time.
All in all, we can say that $T_i^{battery}$ detects the battery consumption up to B value, and T^{ToBeR} detects it afterwards. Indeed if $T_i^{battery} < B \rightarrow T^{ToBeR} = 0$ and the quantity $T_i^{battery} + T^{ToBeR}$ indicates the total amount of battery consumed in the route up to costumer i.

$$T_j^{time_{Min}} = \begin{cases} max(e_j, T_i^{time_{Min}} + t_{i,j}) & if T_i^{rch} = 0 \\ max(e_j, T_i^{time_{Min}} + t_{i,j} + max(0, T_j^{ToBeR} * rate_r^{time} - T_j^{slack})) & otherwise \end{cases}$$

note that since we are in the single recharge case there is only one possible recharging station where we could have been recharging our battery at, we called it r.

$$T_j^{time_{MAX}} = \begin{cases} min(l_j, max(e_j, T_i^{time_{MAX}} + t_{i,j})) & if i \notin R \\ min(l_j, (max(e_j, T_i^{time_{MAX}} + t_{i,j}) + max(0, T_i^{battery} rate_i^{time} - T_j^{Slack}))) & if i \in R \end{cases}$$

Note that if $T_i^{rch} = 0$ then $T_i^{time_{MAX}} = T_i^{time_{Min}}$, and $T_i^{time_{MAX}}$ carries within itself the information about the most restrictive lower bound of each visited time window.

The slack time REF instead is defined as follows:

$$T_j^{slack} = \begin{cases} 0 & if T_i^{rch} = 0 \\ min(T_i^{slack} + max(0, e_j - (T_i^{time_{Min}} + t_{i,j})), T_r^{battery} rate_r^{time}) & otherwise \end{cases}$$

Again note that since we are in the single recharge case, we have only one possible station where we can recharge at, I denote it by r, and the maximum admissible slack time is defined by the maximum quantity of energy I can recharge when I'm at that particular station, in fact the upper bound on the slack time is $T_r^{battery} rate_r^{time}$.
The so defined cumulative slack time is also equal to:

$$T_j^{slack} = T_j^{time_{min}} - (T_r^{time_{min}} + \sum_{r+1}^{j-1} t_{i,i+1})$$

So the slack time equals the minimum arrival time at node j, minus the minimum arrival time at node r, minus the time taken to transit through all the arcs of the path from r to j. This formula will be helpful later on.
Note that we expressed every variable in terms of "cumulative" quantities, so no incompatibility problem shall rise.
Last label: T_j^{cost}.

$$T_j^{cost} = \begin{cases} T_i^{cost} + c_{i,j} & j \neq depot \\ T_i^{cost} + c_{i,j} + ToBeR_j rate_r^{cost} & otherwise \end{cases}$$

Unluckily this model has some problem in the cost label because it doesn't exploit at best our knowledge about the variable T_j^{ToBeR}.
The formulation below is more performing, it's not based on the "structure" of the label but on the following considerations.

$$T_j^{cost} = \begin{cases} T_i^{cost} + c_{i,j} & T_j^{battery} < B \\ T_i^{cost} + c_{i,j} + [b_{i,j} - (B - T_i^{battery})]rate_r^{cost} & T_i^{battery} < B \wedge T_j^{battery} = B \\ T_i^{cost} + c_{i,j} + b_{i,j} rate_r^{cost} & T_i^{battery} = B \end{cases}$$

Independently of the fact that the vehicle has visited a recharging station or not, if the path's battery consumption is lower than the total battery capacity, it's reasonable to suppose that the vehicle didn't actually recharge at the station (because it would imply to pay an extra cost, with no actual improvement), that's why in the first case we only add up the arc costs.
Instead if the vehicle has already reached its maximum battery capacity at node i and it moves to j, the cost to be added on what it already costed up to i is the cost of the arc plus the cost of the extra recharging to ensure battery feasibility up to j, so $b_{i,j} rate_r^{cost}$.
Last case, if the battery consumption up to node i is less than the total admissible battery consumption, while the battery consumption at node j is equal or greater than the total admissible battery consumption this means that during this arc we reached our maximum consumption thus our battery is now empty. The cost we had to pay to move on this arc is then the sum of the arc cost itself $c_{i,j}$ plus the amount of battery we had to recharge in order to arrive with a battery consumption up to B at node j (arriving with battery consumption lower than B would imply to pay an extra cost with no actual improvement on the path). Note that since this is a minimum problem we will always recharge just enough to arrive at the following node, not more, that's why in this latter case we additionally have to pay only $[b_{i,j} - (B - T_i^{battery})]rate_r^{cost}$.

Once we have done all these considerations we can re-write this last label in the following compact but less readable form (note that if $T_i^{rch} = 0$ then $rate_r^{cost}$ is undefined but or $max(0, [b_{i,j} - (B - T_i^{battery})])$ is zero or the path is unfeasible):

$$T_j^{cost} = T_i^{cost} + c_{i,j} + max(0, [b_{i,j} - (B - T_i^{battery})])rate_r^{cost}$$

Another way to express this last REF may be the following:

$$T_j^{cost} = T_i^{cost} + c_{i,j} + (T_j^{ToBeR} - T_i^{ToBeR})rate_r^{cost}$$

Where we add and subtract the cumulative amount of battery to be recharged at the head and the tail of each arc because their difference is the incremental (non-cumulative) amount of battery needed to cross that particular arc.
With this formulation we should have properly defined every aspect of our problem and avoid every "loop" between variable, like defining the $T^{time_{Min}}$ label depending on the T^{slack} and the latter depending on the former.
The value of any of these labels becomes meaningless (and maybe even undefined) in case of battery infeasibility or violation of any other constraints.

3.2.4 Label Feasibility

Given the previously defined labels and REFs we now define the conditions under which a label can be considered feasible or unfeasible.
A Label $L_i = (T_i^{cost}, T_i^{load}, T_i^{rch}, T_i^{time_{Min}}, T_i^{time_{MAX}}, T_i^{slack}, T_i^{ToBeR}, T_i^{battery}, T_i^{cust_n})$ is considered feasible if **all** the following conditions hold:

$$T_i^{load} \le Q \tag{3.1}$$

$$T_i^{rch} \le 1 \tag{3.2}$$

$$T_i^{time_{Max}} \le l_i \tag{3.3}$$

$$T_i^{time_{Min}} \le T_i^{time_{MAX}} \tag{3.4}$$

$$T_i^{ToBeR} \le T_i^{rch} T_r^{battery} \tag{3.5}$$

$$T_i^{battery} \le B \tag{3.6}$$

$$T_i^{cust_n} \le 1 \forall n \in N \tag{3.7}$$

Some of these formulas deserve a deeper explanation.
For instance note that the constraint on the battery (3.6) can be violated if and only if $T_i^{rch} = 0$, otherwise this value will always be upper bounded by B (see the corresponding REF formula). So it seems to be no control on the battery constraint once we perform a visit to a recharge station. Indeed there is and it's modeled by (3.5). Firstly note that until no visit at a recharging station has been performed the value of T_i^{ToBeR} and T_i^{rch} are fixed to 0, actually if $T_i^{rch} = 0, T_r^{battery}$ is

not even defined. Only once $T_i^{rch} = 1, T_r^{battery}$ is finally defined, and this constraint models the fact that in the single recharge case, I can recharge up to the amount of battery I had consumed when I visited the recharging station.
Constraints (3.1),(3.2),(3.3),(3.7) simply model the limits on the maximum possible load carried by any vehicle and the maximum times a vehicle can visit a recharging station or a costumer without violating any time window.
In the end we analyze constraint (3.4). This constraint apparently only models the mandatory chronic order between $T_i^{time_{Min}}$ and $T_i^{time_{MAX}}$; while actually it entails even an "hidden" condition on the slack time. In fact this constrain can be violated only if an additional recharging time yields to a time windows violation, while non-violating constraint (3.4) imply that:

$$T_j^{slack} = T_j^{time_{min}} - (T_r^{time_{min}} + \sum_{r+1}^{j-1} t_{i,i+1}) \leq T_j^{time_{MAX}} - (T_r^{time_{min}} + \sum_{r+1}^{j-1} t_{i,i+1})$$

So ensuring that, if at the recharging station I recharged up to T_j^{slack} units of time this would not lead to *any* time window violation.
Another condition that must be met in order to have a feasible label is the following:

$$T^{slack} + T^{MAX} - T^{min} \leq T^{Battery} rate_r^{time}$$

this condition ensures us that there is no way in which we could ask the vehicle to recharge more time than how much it actually could. Luckily this condition is entailed with the others so we don't have to check it.

3.3 Dominance Rule

A label $L_{i,1}$ dominates $L_{i,2}$, both structured as:
$L_{i,k} = (T_{i,k}^{cost}, T_{i,k}^{load}, T_{i,k}^{rch}, T_{i,k}^{time_{Min}}, T_{i,k}^{time_{MAX}}, T_{i,k}^{slack}, T_{i,k}^{ToBeR}, T_{i,k}^{battery}, T_{i,k}^{cust_n})$, with $k \in (1,2)$
if both the associated paths end at the very same vertex i, $T_{i,1}^{any} \leq T_{i,2}^{any}$ with $any \in (load, rch, time_{Min}, (cust_n)_{n \in N})$ and for every start service time $T_2 \in [T_{i,2}^{time_{Min}}, T_{i,2}^{time_{MAX}}]$, there exists a service start time $T_1 \in [T_{i,1}^{time_{Min}}, T_2]$ such that the associated path p_1 has a battery more charged and still it costs less with respect to the associated path p_2.
Note that the above definition implies that every possible extensions of path p_2 is feasible as well for path p_1. The condition p_1 has a battery more charged than p_2 can be translate into "p_1 has a battery capacity less consumed than p_2".

Firstly we treat two "special" cases where $T_1^{rch} = 0$, then the most general one, $T_1^{rch} = 1$.

3.3.1 Dominance Rule when $T_1^{rch} = 0$

If $T_{i,1}^{rch} = 0 \wedge T_{i,2}^{rch} = 0$ then the dominance conditions simply translate in:

$$T_{i,1}^{battery} \leq T_{i,2}^{battery} \text{ and } T_{i,1}^{cost} \leq T_{i,2}^{cost}$$

If $T_{i,1}^{rch} = 0 \wedge T_{i,2}^{rch} = 1$ instead I need to ensure that even in the most critical case, when $vehicle_2$ recharges all of what it can recharge, $vehicle_1$ still has more battery available, and when $vehcle_2$ doesn't recharge at all, $vehicle_1$ is still cost convenient. Satisfying this two condition ensure that $path_1$ is always a better choice with respect to $path_2$. In formulas we need to satisfy the two following conditions:

$$T_{i,1}^{battery} \leq T_{i,2}^{battery} + T_{i,2}^{ToBeR} - (T_{i,2}^{slack} + T_2^{time_{MAX}} - T_{i,2}^{time_{Min}})rate_{i,2}^{time}$$
$$T_{i,1}^{cost} \leq T_{i,2}^{cost}$$

The previous formula can be further specified considering that: if $T_{i,2}^{ToBeR} > 0$ then there is an other "hidden" cost to be paid in the second route. In fact the second route should at least pay $b_{j,depot} rate_r^{cost}$ more than $path_1$. Thus:
If $T_{i,1}^{rch} = 0 \wedge T_{i,2}^{rch} = 1 \wedge T_{i,2}^{ToBeR} > 0$ then the second condition become:

$$T_{i,1}^{cost} \leq T_{i,2}^{cost} + b_{j,depot} rate_r^{cost}$$

We didn't consider this particular case in our computation because if we consider this extra cost on $vehicle_2$, then we should also consider that $vehicle_1$ may incur in the same extra cost if its battery capacity left it's not enough to go back to the depot. This only complicate the model without any real advantage.
Note that a case where $T_{i,1}^{rch} = 1 \wedge T_{i,2}^{rch} = 0$ is meaningless because it violates one of the initial hypothesis, $T_{i,1}^{rch} \leq T_{i,2}^{rch}$.

Last case, the most general one, is when $T_{i,1}^{rch} = 1 \wedge T_{i,2}^{rch} = 1$.

3.3.2 Battery&Cost Dominance Rule when $T_1^{rch} = 1$

Let's start with some consideration on the Battery-time graph. Given a generic label L_i the battery consumption in i, from now on called $Battery$, always follows the same trend as we can see in figure (3.1).

Initially a straight line from up to down when $time = T_i^{time_{min}}$ and then with continuity an oblique line with slope $-rate_r^{time}$ until $time = T_i^{time_{MAX}}$.
The vertical line is due to the fact that initially, if our accumulated slack time is more than what needed to recharge T^{ToBeR} ,we can use it to recharge the battery, and the oblique line shows that we can start our service later in order to recharge more our battery.
Remember that label $T^{battery}$ tells us information on the maximum level of battery consumption, but in a given point a vehicle may have a set of possible Battery values.
This trend is described by the following formula:
In order to be as clear as possible and because it has a vertical component that can't be expressed

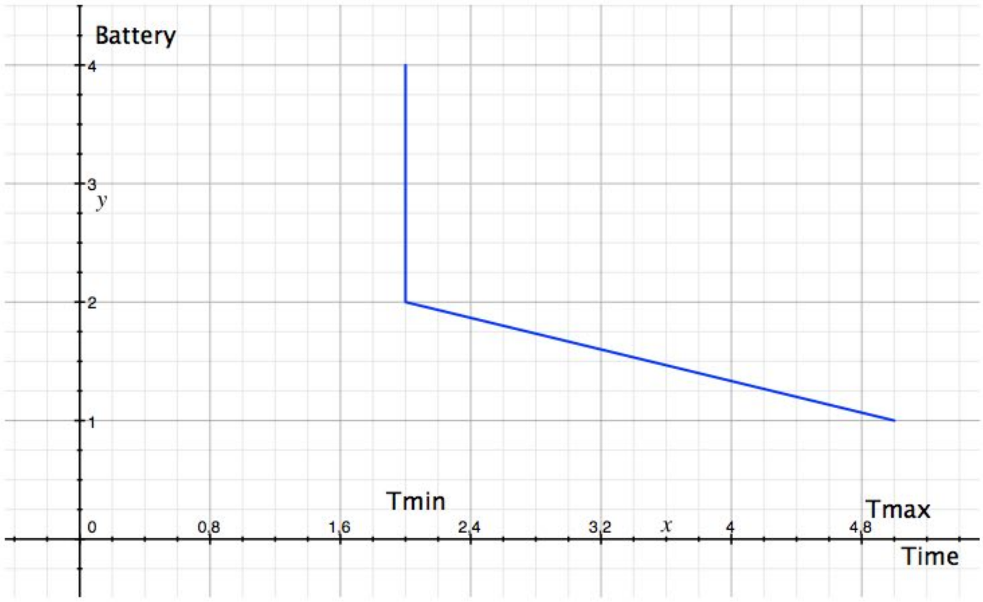


Figure 3.1: Curve of a generic Battery level

as y=f(x) we use a notation like x=f(y) for the vertical part, and we use a more clear notation y=f(x) for the oblique part.

$$\text{time}= T^{time_{Min}} \text{ if } T^{battery} - max[0, \frac{T^{slack}}{rate_r^{time}} - T^{ToBeR}] < Battery \leq T^{battery}$$

$$\text{Battery}=T^{battery} - max[0, \frac{T^{slack}}{rate_r^{time}} - T^{ToBeR}] - \frac{(time - T^{time_{Min}})}{rate_r^{time}} \text{ if } T^{time_{Min}} < time \leq T^{time_{MAX}}$$

Note that $max[0, \frac{T^{slack}}{rate_r^{time}} - T^{ToBeR}]$ indicates the amount of battery I may recharge during the accumulated slack time taking in consideration that a part of it it's used to recharge the quantity T^{ToBeR}.
We start the following analysis considering that if $T^{ToBeR} rate_r^{time} - T^{slack} \geq 0$ the trend becomes just an oblique line, easier to treat, as we can see in figures (3.2) and (3.3).
In this case we can say that L_1 dominates L_2 if and only if the followings apply:

$$Battery_1(min[T_1^{time_{MAX}}, T_2^{time_{Min}}]) \leq Battery_2(T_2^{time_{Min}})$$
$$Battery_1(min[T_1^{time_{MAX}}, T_2^{time_{MAX}}]) \leq Battery_2(T_2^{time_{MAX}})$$
$$Price_1(min[T_1^{time_{MAX}}, T_2^{time_{Min}}]) \leq Price_2(T_2^{time_{Min}})$$
$$Price_1(min[T_1^{time_{MAX}}, T_2^{time_{MAX}}]) \leq Price_2(max[T_2^{time_{Min}}, min[T_1^{time_{MAX}}, T_2^{time_{MAX}}]])$$

Where $Price_k(t)$ is a function such that given a time t, it returns the value of the cost function as if I had been recharging up to time t. In this case it will be the following:

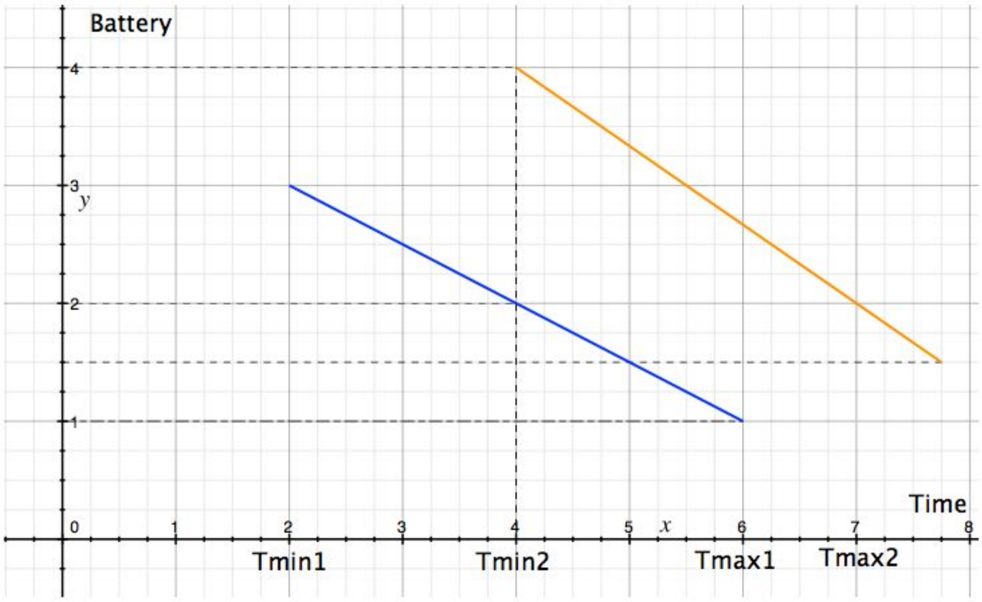


Figure 3.2: Battery curve with no slack time

$$Price(t) = T^{cost} + (t - T^{time_{Min}})rate^{cost}$$

Graphically we can see it in figure (3.3),

The first two inequalities ensure us that for every time t_2, $Battery_2(t_2)$ can't be more charged with respect to $Battery_1(t_1)$where t_1 is the maximum time such that $t_1 \leq t_2 \bigwedge Battery_1(t_1)$ is a defined function .
The last two inequalities ensure us that for every time t_2, $Price_2(t_2)$ is higher with respect to $Price_1(t_1)$ where t_1 is the maximum time such that $t_1 \leq t_2 \bigwedge Price_1(t_1)$ is a defined function .
When $T^{ToBeR} rate^{time} - T^{Slack} > 0$ we need to pay more attention.
Now we have to consider even the vertical line. Note that this carries some problems because we are no more dealing with functions. Now for each value of x, it may correspond more values of y. Also note that there is no case where it would be more convenient to postpone the service start time without using previously all the available slack time.
First thing to check is that, for any given time, there can't be a case where $path_2$ has more battery than $path_1$; otherwise we could always created an example where $path_2$ has an extension not feasible for $path_1$. Secondly we have to compare values of prices for all possible battery levels. To check if there is a case where $path_2$ has more battery than $path_1$ we divide the problem in three subcases: $T_1^{time_{MAX}} \leq T_2^{time_{Min}}$,$T_2^{time_{Min}} \leq T_1^{time_{MAX}} \leq T_2^{time_{MAX}}$ and $T_1^{time_{MAX}} \geq T_2^{time_{MAX}}$.

For the first subcase (figure (3.4)) a sufficient condition is that the minimum battery consump-

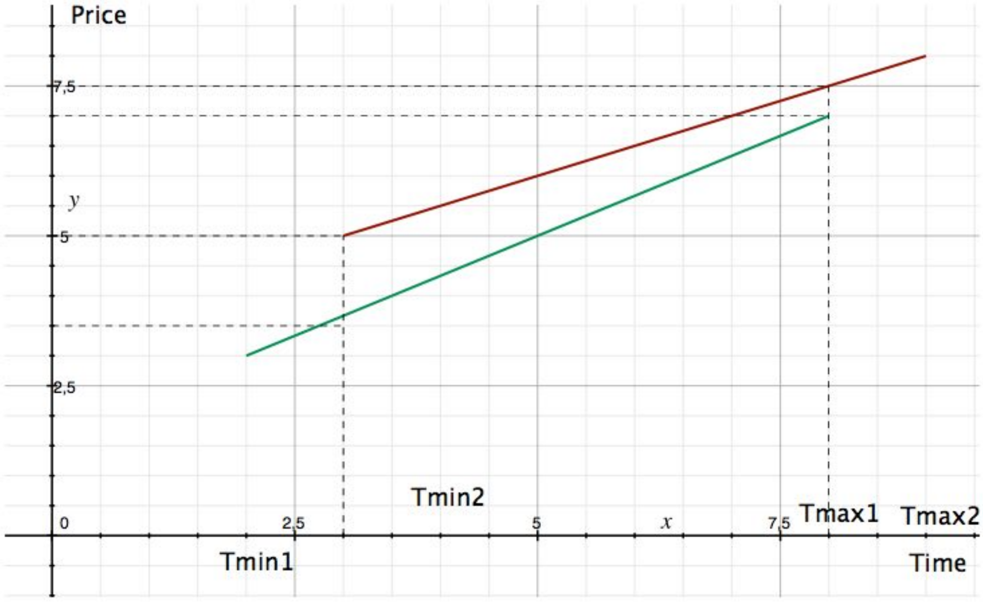


Figure 3.3: Cost curve with no slack time

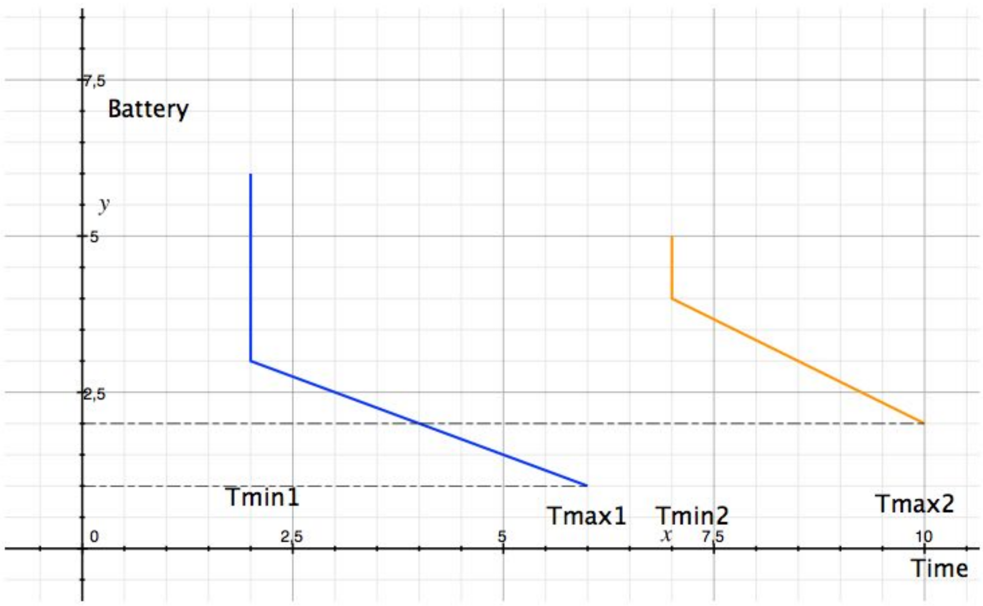


Figure 3.4: Battery curve first subcase

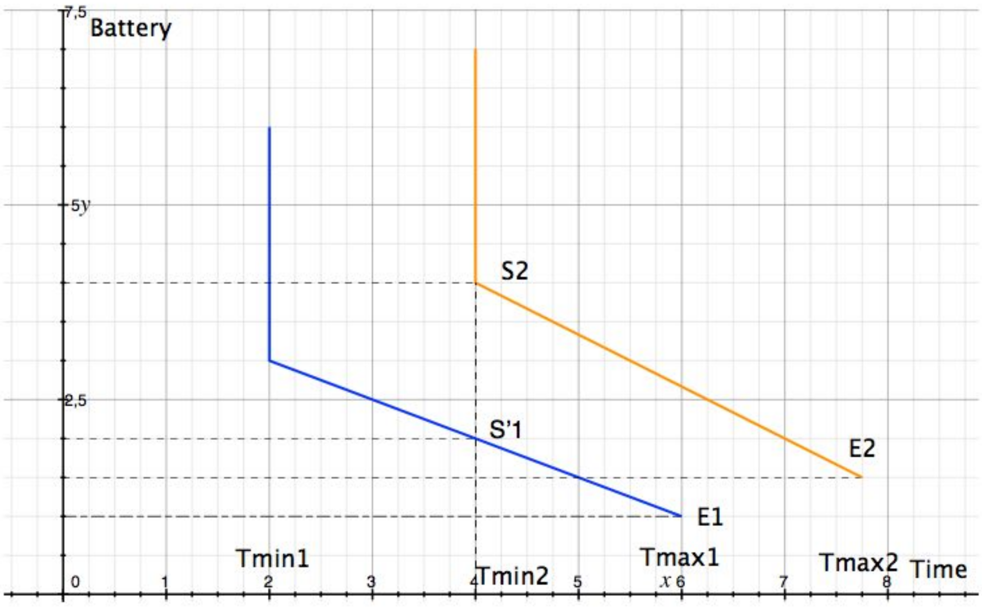


Figure 3.5: Battery curve second subcase

tion of $path_1$ is lower than the minimum battery consumption in $path_2$, in formulas:

$$T_1^{battery} + T_1^{ToBeR} - \frac{(T_1^{slack} + T_1^{time_{MAX}} - T_1^{time_{Min}})}{rate_{r1}^{time}} \leq$$
$$T_2^{battery} + T_2^{ToBeR} - \frac{(T_2^{slack} + T_2^{time_{MAX}} - T_2^{time_{Min}})}{rate_{r2}^{time}}$$

Note that since we are in the case where $T^{ToBeR} rate^{time} - T^{Slack} > 0$ holds the previous formula is equal to the following:

$$T_1^{battery} - max[0, \frac{T_1^{slack}}{rate_{r,1}^{time}} - T_1^{ToBeR}] - \frac{(T_1^{time_{MAX}} - T_1^{time_{Min}})}{rate_{r1}^{time}} \leq$$
$$T_2^{battery} - max[0, \frac{T_2^{slack}}{rate_{r,2}^{time}} - T_2^{ToBeR}] - \frac{(T_2^{time_{MAX}} - T_2^{time_{Min}})}{rate_{r2}^{time}}$$

For the second case (figure (3.5)) instead sufficient conditions are the following: the battery consumption of the first vehicle evaluated in S_1^1 is lower than the battery consumption of the second vehicle at S_2 and the battery consumption of the first vehicle evaluated in E_1 is lower than the battery consumption of the second vehicle at E_2, where with S_2 we denoted the point at the bottom end of the vertical line, E_i the lowest point of the line and with S_1^1 the point determined by the first battery function when $time = T_2^{time_{Min}}$, in formulas:

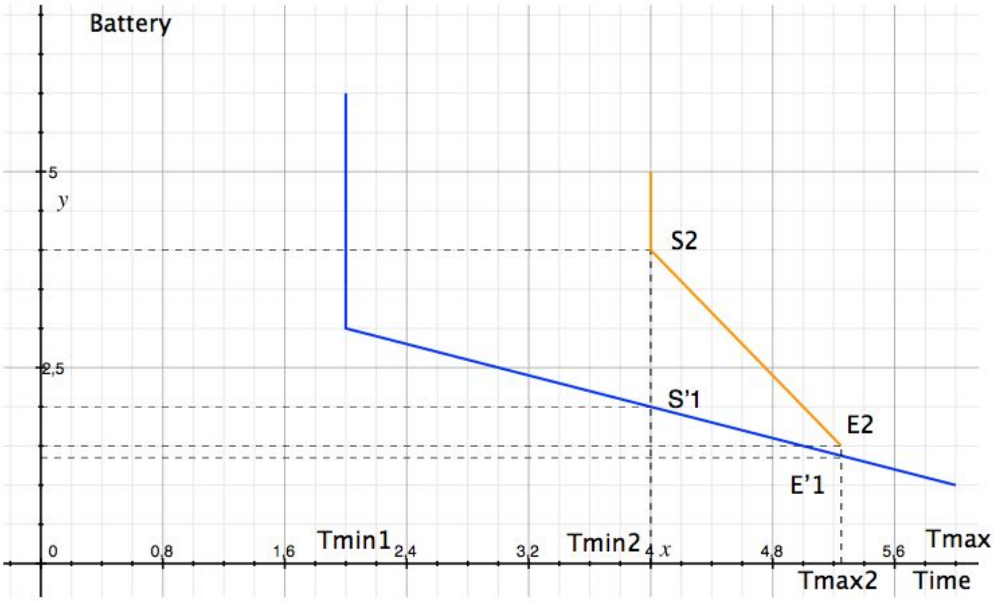


Figure 3.6: Battery curve third subcase

$$T_1^{battery} + T_1^{ToBeR} - \frac{(T_1^{slack} + T_2^{time_{Min}} - T_1^{time_{Min}})}{rate_{r1}^{time}} \le T_2^{battery} + T_2^{ToBeR} - \frac{T_2^{slack}}{rate_{r2}^{time}}$$

and..

$$T_1^{battery} + T_1^{ToBeR} - \frac{(T_1^{slack} + T_1^{time_{MAX}} - T_1^{time_{Min}})}{rate_{r1}^{time}} \le$$
$$T_2^{battery} + T_2^{ToBeR} - \frac{(T_2^{slack} + T_2^{time_{MAX}} - T_2^{time_{Min}})}{rate_{r2}^{time}}$$

Finally for the last subcase (figure (3.6)) necessary and sufficient conditions are the following: the battery consumption of the first vehicle evaluated in S_1^1 is lower than the battery consumption of the second vehicle at S_2 and the battery consumption of the first vehicle evaluated in E_1^1 is lower than the battery consumption of the second vehicle at E_2,where with E_1^1 I mean the point determined by the first battery function when $time = T_2^{time_{MAX}}$, in formulas:

$$T_1^{battery} + T_1^{ToBeR} - \frac{(T_1^{slack} + T_2^{time_{Min}} - T_1^{time_{Min}})}{rate_{r1}^{time}} \le T_2^{battery} + T_2^{ToBeR} - \frac{T_2^{slack}}{rate_{r2}^{time}}$$

and..

Figure 3.7: Price curve

$$T_1^{battery} + T_1^{ToBeR} - \frac{(T_1^{slack} + T_2^{time_{MAX}} - T_1^{time_{Min}})}{rate_{r1}^{time}} \leq$$
$$T_2^{battery} + T_2^{ToBeR} - \frac{(T_2^{slack} + T_2^{time_{MAX}} - T_2^{time_{Min}})}{rate_{r2}^{time}}$$

In the unlucky case where $T_1^{time_{min}} = T_2^{time_{min}}$ we should also check the following condition: $T_1^{battery} \leq T_2^{battery}$.

This formulation may seems to not take into account that the maximum amount of energy I may recharge at a given station are upperbounded by $T_{r,i}^{battery}$, indeed this constraint is propagated thanks to a proper formulation of $T_i^{time_{MAX}}$.
Another interesting point is that we considered variable T^{ToBeR} only when $T^{ToBeR} rate^{time} - T^{Slack} > 0$. That's because when $T^{ToBeR} rate^{time} - T^{Slack} < 0$ the effect of variable T^{ToBeR} is entailed within variable $T^{time_{Min}}$.
Once we checked that there exists no configuration where, at the same time, $path_2$ can have less battery consumed than $path_1$, we have to confront the costs for each given battery level.

But first we take a look at how it's structured the Price function, figure (3.7).

As we can see in the (time,Price) graph, the trend is again one vertical segment when $time = T_i^{time_{Min}}$, this time oriented from the bottom to the top, and then with continuity an

oblique segment with slope $+rate_{r_i}^{cost}$ until $time = T_i^{time_{MAX}}$. As before the vertical line shows as at the same time we may have different costs due to how much slack time we spend recharging our vehicle, while the oblique line represents the cost increasing if we postpone our minimum service start time to additionally spend more moments recharging.
In formulas that function is described by (for the same previous reasons we use again a mixed notation):

$$\text{time}= T^{time_{Min}} \text{ if } T^{cost} < Price < T^{cost} + max[0, \frac{T^{slack}}{rate_r^{time}} - T^{ToBeR}]rate_r^{cost}$$
$$\text{Price}=T^{cost} + max[0, \frac{T^{slack}}{rate_r^{time}} - T^{ToBeR}]rate_r^{cost} + \frac{(time - T^{time_{Min}})}{rate_r^{time}} rate_r^{cost} \text{ if } T^{time_{Min}} < time < T^{time_{MAX}}$$

To check dominance we build another graph, a Battery-Price graph.
Building this graph is pretty simple, for each curve we need three pair (Battery, Price) of values to determine each point and to find them we can use the information in the graphs we have already seen. The three pairs of data we need are the following:

- Price and Battery at time $T^{time_{Min}}$ when no available slack time is used, point Q.
- Price and Battery at time $T^{time_{Min}}$ when all the available slack time is used, point W.
- Price and Battery at time $T^{time_{MAX}}$, point E.

Note that Price and Battery linearly (piecewise) depend on time, and it's straightforward that they linearly (piecewise) depend on each other. So we just need to connect points Q-W and W-E to obtain the Battery-Price function, shown in figure (3.8).

We can notice how these curves are always straight segments with slope $-rate^{cost}$!
This simplify a lot our analysis, in fact we now need only point Q and E to fully determine our line, and we can use procedures similar to the ones seen in *Battery&Cost Dominance when* $T^{ToBeR}rate^{time} - T^{Slack} > 0$ to determine the dominance.
In fact with this function we can directly confront the price for every possible battery level. In particular, since it's a linear function, we can avoid checking all possible levels but just a few "meaningful" ones.
In particular we have to check (see figure (3.9)):

$$Price_1(Battery(Q_2)) \leq Price_2(Battery(Q_2))$$
$$Price_1(Battery(E_2)) \leq Price_2(Battery(E_2))$$

where $Battery(P)$ is just a function that returns the $Battery$ at point P, while function Price works like this:

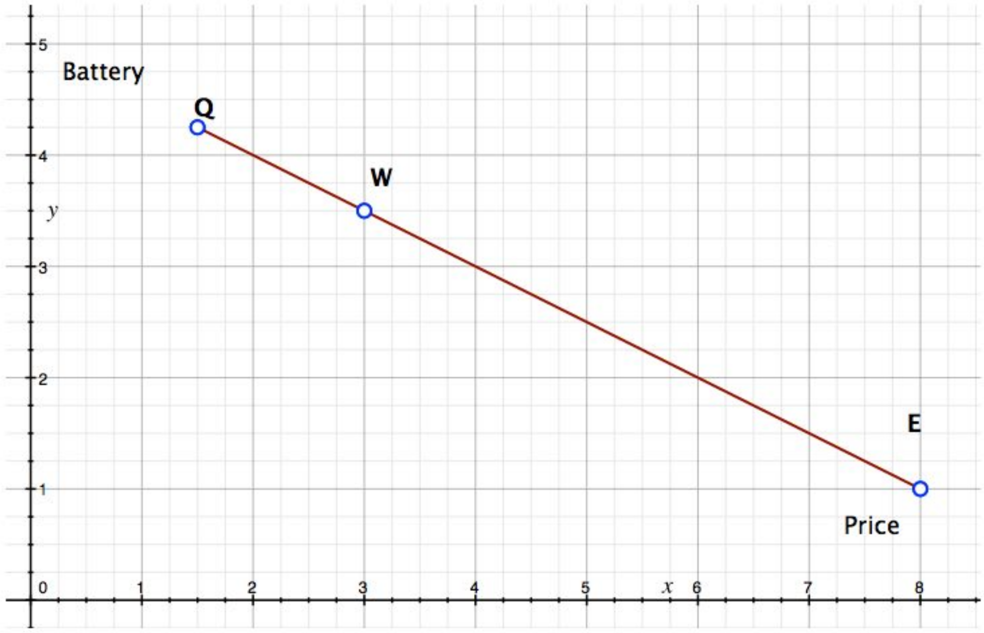


Figure 3.8: Price Battery curve

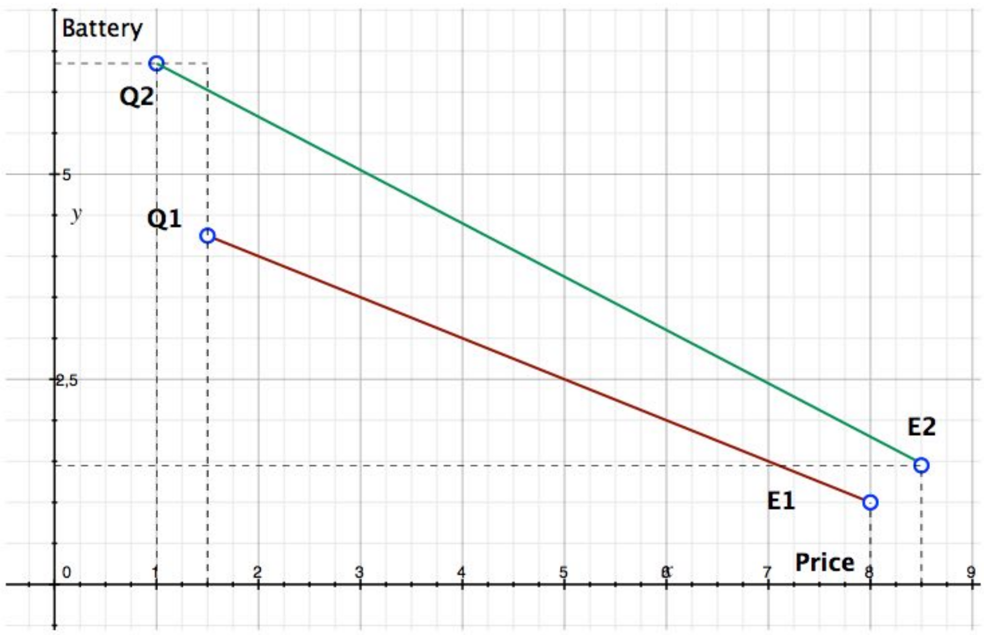


Figure 3.9: Battery Price Comparison

$$Price(B) = \begin{cases} Price(Q) & B > Battery(Q) \\ Price(Q) + \\ \dfrac{(Price(E) - Price(Q))(Battery(Q) - B)}{Battery(Q) - Battery(E)} & Battery(E) \le B \le Battery(Q) \\ \infty & B < Battery(E) \end{cases}$$

Last case has ∞ because if we are trying to look at a value not comprehended by our function it means there is a point where $path_2$ has more battery then $path_1$.
Ensuring the "Battery" conditions means that for every possible battery level reached by $vehicle_2$, $vehicle_1$ can reach the same level (or lower), in the same time (or lower).
Ensuring the previous "Price" conditions means that for every possible battery level reached by $vehicle_2$, the corresponding cost for that battery level for $vehicle_1$ is cost-convenient.

This end what we will call in the next paragraph the Battery&Cost Dominance rule.

3.3.3 Dominance Rule: Conclusion

With the so-defined Battery&Cost Dominance we may say:

A label $L_{i,1}$ dominates $L_{i,2}$, both structured as:
$L_{i,k} = (T^{cost}_{i,k}, T^{load}_{i,k}, T^{rch}_{i,k}, T^{time_{Min}}_{i,k}, T^{time_{MAX}}_{i,k}, T^{slack}_{i,k}, T^{ToBeR}_{i,k}, T^{battery}_{i,k}, T^{cust_n}_{i,k})$, with $k \in (1,2)$ if $L_{i,1} \neq L_{i,2}$, both the associated paths end at the very same vertex i, $T^{any}_{i,1} \le T^{any}_{i,2}$ with $any \in (load, rch, time_{Min}, (cust_n)_{n \in N})$ and L_1 Battery&Cost dominates L_2.
The above definition implies that every possible extensions of path p_2 is feasible as well for path p_1, but p_1 cost equal or less.

3.3.4 Dominance Rule: Addendum

The previous considerations about the case $T^{rch}_1 = T^{rch}_2 = 1$ can be extended considering the following two subcases:
If $T^{ToBeR}_2 > 0 \bigwedge T^{ToBeR}_1 = 0$ whenever we wrote T^{cost}_2 we can substitute it with: $T^{cost}_2 + b_{i,depot} rate^{cost}_{r2}$ considering that the vehicle has at least to come back to the depot, and it will have to pay an extra $b_{i,depot} rate^{cost}_{r2}$. Whenever we see T^{cost}_1 instead we should substitute with $T^{cost}_1 + max[0, T^{battery} + b_{i,depot} - B] rate^{cost}_{r1}$ which represents the additional cost we should pay if our battery runs empty when we're trying to reach the depot.
Similarly If $T^{ToBeR}_1 > 0$ (note that this implies that $T^{ToBeR}_2 > 0$ *with probability one*)

Whenever we wrote T^{cost}_k we should substitute it with: $T^{cost}_k + b_{i,depot} rate^{cost}_{rk}$.
This could be done with a modification of the cost REF, from:

$$T_j^{cost} = \begin{cases} T_i^{cost} + c_{i,j} & T_j^{battery} < B \\ T_i^{cost} + c_{i,j} + [b_{i,j} - (B - T_i^{battery})]rate_r^{cost} & T_i^{battery} < B \wedge T_j^{battery} = B \\ T_i^{cost} + c_{i,j} + b_{i,j}rate_r^{cost} & T_i^{battery} = B \end{cases}$$

To:

$$T_j^{cost} =$$
$$\begin{cases} T_i^{cost} + c_{i,j} + max[0, T_j^{battery} + b_{j,depot} - B]rate_r^{cost} + & \\ -max[0, T_i^{battery} + b_{i,depot} - B]rate_r^{cost} & T_j^{battery} < B \\ T_i^{cost} + c_{i,j} + [b_{i,j} - (B - T_i^{battery})]rate_r^{cost} + b_{j,depot}rate_r^{cost} + & \\ -max[0, T_i^{battery} + b_{i,depot} - B]rate_r^{cost} & T_i^{battery} < B \wedge T_j^{battery} = B \\ T_i^{cost} + c_{i,j} + b_{i,j}rate_r^{cost} - b_{i,depot}rate_r^{cost} + b_{j,depot}rate_r^{cost} & T_i^{battery} = B \end{cases}$$

Actually the previous formulation can be reduced to:

$$T_j^{cost} = T_i^{cost} + c_{i,j} + max[0, T_j^{battery} + b_{j,depot} - B]rate_r^{cost} - max[0, T_i^{battery} + b_{i,depot} - B]rate_r^{cost}$$

where for each cases we add the extra quantity to be paid, if any, to go from node j to the depot and subtract the extra quantity to be paid, if any, to go from node i to the depot.
Anyway we didn't use the rules described in this section in our computational studies.

CHAPTER 4

Acceleration and Branching

In this chapter we will see the acceleration strategy and the branching techniques.

4.1 Acceleration strategy

The use of the following strategies aim at accelerating the time spent solving the subproblem, which is NP-hard for all EVRPTW variants due to the elementarity requirements on the customers and as we will see in the computational studies (chapter 5) it's the real bottleneck of the whole problem.
The first acceleration strategy is the most widely used in column generation. Instead of looking for the minimal reduced column and add only that particular one to the master problem, we look for any the first κ negative reduced cost columns we find and add all of them to the master problem.
The other acceleration strategy consist in rapidly generating negative reduced cost columns using a graph of reduced size. More precisely, at each iteration of the column generation, the labeling algorithm is executed first on a simplified graph G that contains only a subset A^1 of the arcs in A. If it fails to find negative reduced cost columns, then the algorithm is executed again on a larger subset of arcs A^2 with $A^2 \supseteq A^1$, and if again no negative reduced cost column is found the algorithm is executed again on the complete graph G.
As in Desaulniers et al. (2008), the subset $A^i, i \in (1,2)$ varies in each iteration: the arcs in $A^i, i \in (1,2)$ are selected based on the arc reduced costs $\overline{c_{ij}}$, which depend on the current value of the dual variables in the restricted master problem. First, for every vertex $i \in R \bigcup N$, we sort separately all incoming arcs and all outgoing arcs in increasing order of their reduced cost and

put them in separate ordered lists denoted I_i and O_i, respectively. An arc (i,j) is removed from A if:

- $i,j \in R \bigcup N$, so no outgoing arcs from/to the depot are removed
- the rank of (i,j) in list I_j is greater than a predefined parameter μ_i
- the rank of (i,j) in list O_j is also greater than μ_i

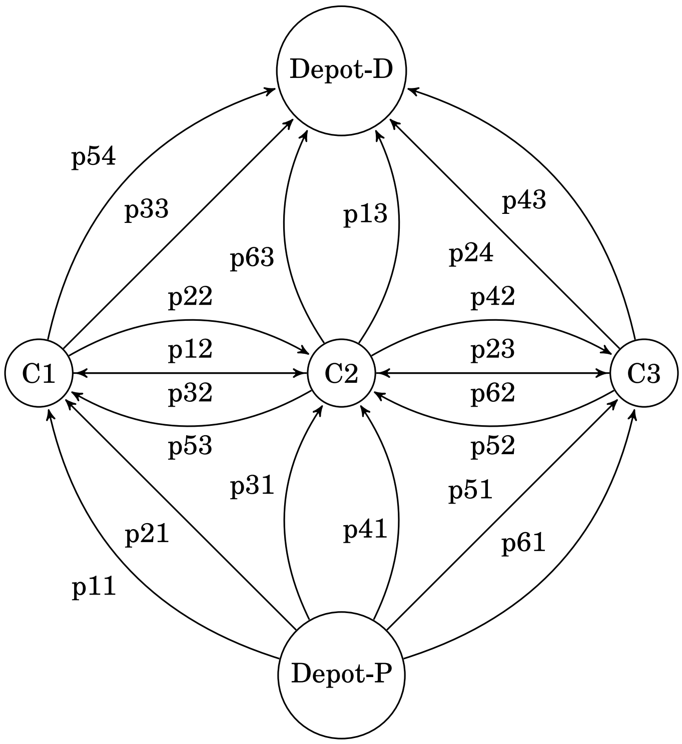
Thus $A^i, i \in (1,2)$ contains all arcs leaving or entering the depot and, for every vertex, at least μ_i incoming and μ_i outgoing arcs (unless initially there exist less than μ_i of these arcs).
In our numerical studies, we used $\kappa = 50$ for the reduced graphs, $\kappa = 10$ for the complete graph, $\mu_1 = 3$ and $\mu_2 = 4$.

4.2 Branching

To derive integer solutions, we impose the following types of branching decisions in the branch-and-bound search tree:

- on the total number of routes
- on the total number of recharges
- on the total number of recharges at a given recharging station
- on the total flow on an arc

Given a fractional-valued solution, these types of decisions are evaluated in the given order and the first type that can be imposed is selected. If the total number of recharges is fractional for several stations, we choose to branch on a station for which the fractional part of its total number of recharges is closest to 0.5. Similarly, if the arc flow is fractional for several arcs, we choose an arc for which the fractional part of its flow is closest to 0.5. For every decision, two branches are created. The previous decisions are imposed by adding inequalities to the restricted master problem. The dual variable of these inequalities alter the reduced cost of certain route variables. Moreover, all routes in the master problem incompatible with the fourth branching rule are removed. The Branch-and-Bound three is explored with a Depth-first strategy, exploring always the more promising node every time it generates a pair of them. With the more promising we simply mean the node with the lowest objective function.
A last clarification about branching it's needed.
There exists pathological case where the previous branching decisions are not enough. Take for example the following situation.



In this case there are six paths;

- $Path_1$ is composed by arcs: p_{11},p_{12},p_{13}
- $Path_2$ is composed by arcs: p_{21},p_{22},p_{23},p_{24}
- $Path_3$ is composed by arcs: p_{31},p_{32},p_{33}
- $Path_4$ is composed by arcs: p_{41},p_{42},p_{43}
- $Path_5$ is composed by arcs: p_{51},p_{52},p_{53},p_{54}
- $Path_6$ is composed by arcs: p_{61},p_{62},p_{63}

If solving the linear problem we end up having each variable set as 0.5, the total number of paths is 3 and each arc has a total flow on itself of 1. In such cases we should also branch on the variables themselves.
However this is a extremely unlikely situation and never happened in our instances.

Chapter 5

Computational Studies

In this section, we present computational experiments to analyze the effectiveness of the proposed branch-price-and-cut algorithms assessing the benefits of allowing multiple and partial recharges, i.e., we compare the three EVRPTW problem variants. Section (5.1) describes the the instance sets used in the experiments while section (5.2) show the computational results. All algorithms were implemented in Python 2.7, using Gurobi 7.0.2 for solving linear programs. The experiments were performed on a MacBook Pro, 2,9 Ghz Intel Core i7, RAM 8Gb 1600 Mhz DDR3, running macOS Sierra 10.12.

5.1 Benchmark Instances

For our experiment we used the very same instances used in Desaulniers et al. ((2016)), which in turn were introduced in Schneider et al. (2014). Those instances are based on the VRPTW benchmark set of Solomon (1987) and modified as follows: to each of the Solomon instances, Schneider et al. (2014) applied the following modifications to obtain an EVRPTW instance:

- 21 randomly generated recharging stations are added;
- the battery capacity is suitably set depending on the average route length of the corresponding VRPTW instance; and
- the time windows of some customers are enlarged to ensure feasibility.

The energy consumption $b_{i,j}$ along an arc $(i,j) \in A$ is set equal to the arc cost $c_{i,j}$ and the proportionality factor α is chosen such that a complete battery recharge requires three times the average customer service time of the considered instance. From those 100-costumers instances we generate a set of smaller instances, which are obtained by randomly extracting 25 customers from each 100-customer instance and keeping the 21 recharging stations, obtaining doing so 9 instances. This was necessary because, as we will see, the algorithm is quite slow due to the Python implementation (C/C++ would have been more performing) and the absence of cutting planes, thus it won't be able to solve in reasonable time big instances. Those instances were modified equipping each recharging station in R with two different technologies: one slower but cheaper and one faster but more expansive[1]. The recharging station at the depot was not equipped with such technologies, instead it was equipped with only one technology, slow and "for free", to mimic the behaviour of a flat cost rate. Additionally, since the algorithm variant for multiple recharges is considerably slow, from each of the 25-costumer, 21-recharges, 2-technology instances, a 20-costumers, 11-recharges, 2-technology instance has been extracted. That was done by randomly eliminating 5 costumers from the bigger instance and subsequently removing randomly 10 recharging stations considering only a subset of removable recharging station so composed: all the recharging stations but the depot and the closest recharging stations to costumers unreachable with a path with no recharge. Please note that it's more convenient to reduce the number of recharging stations that the number of costumer in order to obtain "easier" instances. From now on we will call the 25-costumers instances the complete ones and the 20-costumers instances the reduced ones. Finally, the single full (SRFR) and partial recharge (SRPR) algorithm was tested both on the reduced and complete instances while the multi recharge case (MRFR) was tested only on the reduced instances. In the next section we show the computational results.

5.2 Algorithmic Performances

In this section we will highlight the performances of our algorithm trough summary tables and some comparison between them. Each table is composed by seven columns indicating (in order): name of the instance, total cost of the solution, number of nodes explored, total time took to solve it, number of routes, number of visits to the fast recharging station and number of visits to the cheaper recharging station. We firstly analyze the reduced instances and afterwards the complete ones. A time limit of 6 hours was set for the reduced instances and 12 hours for the complete ones.

5.2.1 Reduced Instances

The following three tables (5.1)(5.2)(5.3) show the results of the three algorithms (SRFR,SRPR,MRFR) on the very same reduced instances, called c10i with $i \in \{1,2,3,4,5,6,7,8,9\}$.

As we can better see from the following computations introducing partial and multiple recharge can only improve our solution. Indeed quantifying what above, we have on average

[1]This was done to emulate the behaviour of traditional recharging and supercharging stations

Single Recharge Full Recharge						
Name	Cost	#nodes	Time	#paths	#Visit Fast	#Visit Cheap
c101	584.39	1	1267.96	6	2	2
c102	563.65	1	308.15	5	3	0
c103	369.46	6	18525	3	1	1
c104	520.33	3	1003	4	0	4
c105	515.2	1	617.17	6	1	3
c106	531.4	2	1356.82	5	0	4
c107	451.42	1	635.29	5	1	3
c108	428.32	2	1108.03	4	1	1
c109	494.38	2	2948.8	4	2	1

Table 5.1: Test: SRFR - Reduced Instances

Multiple Recharge Full Recharge						
Name	Cost	#nodes	Time	#paths	#Visit Fast	#Visit Cheap
c101	584.39	0	2337,204	5	2	2
c102	540.35	2	15830,17	5	4	1
c103	N/A	N/A	N/A	N/A	N/A	N/A
c104	509.89	4	3214.42	5	1	3
c105	503.37	0	647.83	5	1	3
c106	531.4	2	1388	5	1	5
c107	451.43	1	525.25	5	1	4
c108	401.34	1	1104.99	4	2	2
c109	474.41	1	2345.99	4	0	3

Table 5.2: Test: MRFR - Reduced Instances

Single Recharge Partial Recharge						
Name	Cost	#nodes	Time	#paths	#Visit Fast	#Visit Cheap
c101	583.687	8	1947.67	6	1	3
c102	513.55	1	2777.08	7	0	3
c103	344.24	10	9841.91	4	1	1
c104	483.2	2	6866.39	5	1	3
c105	512.45	1	164.41	6	0	4
c106	516.32	8	1100.34	5	0	4
c107	440.94	1	233.00	5	2	2
c108	407.44	2	891.45	5	0	2
c109	452.69	4	2260.52	5	2	2

Table 5.3: Test: SRPR - Reduced Instances

the performances illustrated in table (5.4). Please note that the average is computed on all the instances' results but c103's one, because it could not be solved whit-in the time limit in the multiple recharge case.

Average Results for the Reduced Instances						
Variant	Cost	#nodes	Time	#paths	#Visit Fast	#Visit Cheap
SRFR	511,14	1,625	1155,65	4,88	1,25	2,25
MRFR	499,57	1,875	3424,23	4,75	1,5	2,88
SRPR	488,78	3,38	2030,11	5,5	0,75	2,86

Table 5.4: Test: Comparison - Reduced Instances

It's clear how introducing multiple and partial recharge improve our solution; numerically, on average, introducing multiple recharge improves the solution of the 2,26%, increasing the resolution time by 196%, while introducing partial recharge improves the solution of the 4,37%, increasing the resolution time by 75,67%.

5.2.2 Complete Instances

Tables (5.5),(5.6) show the results of the SRFR and SRPR algorithm for each complete instances while table (5.7) show the aggregate average result of both for comparison purposes. Again instance c103 was not solved in twelve hours computation time, hence the average results are computed disregarding that particular instance.

Single Recharge Full Recharge						
Name	Cost	#nodes	Time	#paths	#Visit Fast	#Visit Cheap
c101	648,24	1	1372,968	7	4	1
c102	574,85	2	13357,136	5	2	2
c103	352,19	2	556219	4	1	0
c104	525,29	2	13968,3	4	1	3
c105	558,03	3	6703	7	1	3
c106	602,41	4	11979	5	4	0
c107	545,22	10	2735,4	6	2	2
c108	565,65	18	24167,3	5	4	0
c109	506,06	14	14843,95	4	3	1

Table 5.5: Test: SRFR - Complete Instances

We can clearly notice how introducing partial recharges increase the problem difficulties, resolution time increased by 105%, yet it allows for more economic results, saving about 4,20%.

Single Recharge Partial Recharge						
Name	Cost	#nodes	Time	#paths	#Visit Fast	#Visit Cheap
c101	639,90	1	1716,21	7	0	5
c102	547,19	6	16696,42	7	0	4
c103	N/A	N/A	N/A	N/A	N/A	N/A
c104	479,76	12	42980,49	5	1	4
c105	556,39	14	14.660	7	2	2
c106	578,31	26	27061,06	6	3	3
c107	523,48	2	2227,34	6	1	3
c108	525,54	22	34829,08	7	1	2
c109	485,32	14	42697,67	5	2	2

Table 5.6: Test: SRPR - Complete Instances

Average Results for the Reduced Instances						
Variant	Cost	#nodes	Time	#paths	#Visit Fast	#Visit Cheap
SRFR	565,72	6,75	11140,88	5,375	2,625	1,5
SRPR	541,99	12,125	22858,57	6,25	1,25	3,125

Table 5.7: Test: Comparison - Complete Instances

5.3 General Comment

It's interesting to notice how the improving variants do not guarantee at all the reduction on the number of vehicles used. This is a substantial difference with the homogeneous case where, minimizing the routing cost somehow imply minimizing the travelled distance hence the number of vehicle used, on the other hand in the heterogeneous case we lose this proportionality between cost and travelled distance.
The measured average improvement from full to partial are comparable both in reduced and complete instances, yet we can see how the average number of nodes generated increase rapidly (F: from 1.625 to 6,75; P: from 3.38 to 12.125). This is the reason why even adding only 5 costumers and 10 recharging stations made the resolution time arise exponentially (more than 10 times), this much bigger three take more time to be explored.

Chapter 6

Conclusion

In this thesis, we present effective branch-price-and-cut algorithms for three variants of the EVRPTW with Heterogeneous Recharging Stations, which are defined according to the maximal number of recharges per route (single versus multiple) and the type of recharge (partial versus full). For each problem variant a labeling algorithm for generating feasible routes is presented. Their efficiency results from complex REFs, that allow for constant time feasibility checking, and strong dominance rules. In numerical studies, we demonstrate that the algorithms are capable of solving instances with up to 30 customers, 21 recharging stations and two technologies for each of the problem variants. Finally, we find that allowing multiple, but especially partial recharges help to reduce routing costs and the number of employed vehicles in comparison to the variant with single and full recharge.

Column generation is almost needed in situations like this one, where the total number of possible variables is too huge to be solved by explicitly considering every variable in the problem. In fact the number of variables is almost in the order of $25! * 21 * 2 \cong 6,51^{26}$ (almost 700 septilion alias million billion billion). Adopting CG allows us to work only on a subset of these variables. Our final condition to stop the column generation procedure with its dominance is basically to find a subset (hopefully the smallest one) of columns (variables) such that if they have reduced cost greater than zero then for sure all the other variables have reduced cost greater than zero as well! The effectiveness of the dominance rule rely on that, we only need to check the reduced cost of a subset of variables to make sure that all the other unconsidered variables have for sure reduced cost greater than zero, thus no improving effect in our problem.

APPENDIX A

APPENDIX: DIRECTION TO GROW

In this appendix we would like to highlight some bloopers in the model and the directions to grow of this project.

A.1 Bloopers

Since we decided to minimize only the cost, not the travelled distance or the number of vehicle used, having a recharge station situated at the depot with zero cost rate is basically useless. In fact since there is no penalty in routing another vehicle there is no reason why the algorithm should choose to use the very same vehicle to perform a path, recharge at the depot, and then perform another path, it's easier to route two different vehicles. The absence of cutting planes makes the algorithm explore quite big subthrees, especially on bigger instances (computational studies not shown in this thesis).

A.2 Future Challenges

This thesis deals with some variants introduced in the traditional vehicle routing problem by the use of electrical vehicle, with all the pros and cons they provide.
Future challenges certainly will consist in allowing multiple and partial recharges [1], different types of vehicles (in terms of battery and load capacity), considering non-linear recharge function, battery degradation but especially non-deterministic travel time.

[1] we didn't treat this particular case in our project because we would had to deal with confront of areas, not segments, in the dominance rule making the subproblem extremely more difficult to solve

APPENDIX

B

APPENDIX PYTHON

In this appendix I would like to sketch the implementation I coded in the Python scripts to comment on some feature.
The main script reads almost as the following:

```
for instance in SetOfInstances:
    (Net, Arcs, InitialLabel, InitialPaths) = FirstInizialization(instance)
    (ObjectiveFunction,Variables,DualVariables,problem) = LP(Paths, Net)
    (Paths, Cost, Variables)=ColumnGeneration(Arcs, DualVariables)
    (UpperBound)=MILP(Paths)
    (Solution, Cost, UpperBound)=RecursiveStep(Paths, Arcs, UpperBound,problem)
```

As we can see for each instances I repeat the following procedure.
From the file having the costumers' and stations' positions I extract the Net (set of nodes) and the Arcs. I initialize the column generation heuristically creating some initial feasible paths (warm start). Additionally I used a so called big-M approach, meaning that I create a dummy variable (big-M) with extremely high cost but such that it visits every costumer and it's considered feasible. Doing so we avoid the problem of entering in some branch where we didn't already generate the variables needed in order for that particular problem to be feasible.
For example think at a situation where you have three costumers and three different path each visiting only two of them (namely, path1 will visit costumer 1 and 2, path2 will visit costumer 2 and 3 and path3 will visit costumer 3 ans 1). The only feasible solution at that problem is that each variable assumes value 0.5. Our branching rule will then impose to branch on the total number of routes, imposing to have two or more routes in one case and less than one in the other. This latter linear problem would be unfeasible without a big-M approach because you haven't already generate a route passing through all the costumers.

After the initialization we solve the first linear problem extracting information about the dual variables, these ones are necessary in the following step of column generation in order to determine the reduced cost of the arcs. In the ColumnGeneration function we generate columns and reoptimize the linear problem with the newly added column as long as no negative reduced cost column is found. Once it's the case we solve a mixed integer linear problem to return a valid upper bound for the following depth-first branching three exploration. In the last function, RecursiveStep, the tree is finally explored, as we shall see just below:

```
def RecursiveStep(Paths, Arcs, UpperBound,problem):
    if Integer(Variables(Paths)):
        if Cost(Variables)<UpperBound:
            [Solution]=SolutionPaths(Paths)
            return (Solution, Cost(Variables), Cost(Variables))
        else:
            return (null, Cost(Variables), UpperBound)
    else:
         if Cost(Variables)>UpperBound:
             return (null,Cost(Variables), UpperBound)
         else:
            [UpperBoundMILP]=MILP(Paths)
            if UpperBoundMILP<UpperBound:
                UpperBound=UpperBoundMILP
    [probl1,probl2]=Branch(problema)
    (ObjectiveFunction,Variables,DualVariables) = LP(Paths, Net, probl1)
    (Paths1, Cost1, Variables1)=ColumnGeneration(Arcs, DualVariables)
    if Cost1>UpperBound:
        ..do nothing..
    else:
        (Solution1,Cost1,UpperBound)=RecursiveStep(Paths1,Arcs,UpperBound,probl1)
    (ObjectiveFunction,Variables,DualVariables) = LP(Paths, Net, probl2)
    (Paths2, Cost2, Variables2)=ColumnGeneration(Arcs, DualVariables)
    if Cost2>UpperBound:
        ..do nothing..
    else:
        (Solution2,Cost2,UpperBound)=RecursiveStep(Paths2,Arcs,UpperBound,probl2)
    [SolutionBest, CostBest]=Best(Solution1, Cost1, Solution2, Cost2)
    return (SolutionBest, CostBest, UpperBound)
```

We can see how each time after the column generation process we try to update the upper bound at the best of our knowledge solving a mixed integer problem. Updating as soon as possible the upper bound is crucial because the most time-consuming action is to generate new nodes and one of the main drawbacks of the algorithm is the absence of cutting planes, this causes the three to propagate deeply because of this relatively big integrality gap.

Bibliography

A. Ceselli, A. Felipe, M. T. Ortuño, G. Righini, and G. Tirado.
A branch-and-cut-and-price algorithm for the green vehicle routing problem with partial recharge and multiple technologies, 2015.

G. Desaulniers, F. Errico, S. Irnich, and M. Schneider.
Exact algorithms for electric vehicle-routing problems with time windows.
Operations Research, 64(6), 2016.

Á. Felipe, M. T. Ortuño, G. Righini, and G. Tirado.
A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges.
Transportation Research Part E: Logistics and Transportation Review, 71:111 – 128, 2014.
ISSN 1366-5545.
doi: https://doi.org/10.1016/j.tre.2014.09.003.
URL `http://www.sciencedirect.com/science/article/pii/S1366554514001574`.

D. Goeke, J. Hof, and M. Schneider.
Adaptive variable neighborhood search for the battery swap station location-routing problem with capacitated electric vehicles.
In *Darmstadt Technical University, Department of Business Administration, Economics and Law, Institute for Business Studies (BWL) Working paper*. 2015.

S. Gualandi and F. Malucelli.
Resource Constrained Shortest Paths with a Super Additive Objective Function, pages 299–315.
Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
ISBN 978-3-642-33558-7.
doi: 10.1007/978-3-642-33558-7_24.
URL `http://dx.doi.org/10.1007/978-3-642-33558-7_24`.

G. Hiermann, J. Puchinger, S. Ropke, and R. F. Hartl.
The electric fleet size and mix vehicle routing problem with time windows and recharging stations.
European Journal of Operational Research, 252(3):995 – 1018, 2016.

ISSN 0377-2217.
doi: https://doi.org/10.1016/j.ejor.2016.01.038.
URL `http://www.sciencedirect.com/science/article/pii/S0377221716000837`.

M. Keskin and B. Çatay.
Partial recharge strategies for the electric vehicle routing problem with time windows.
Transportation Research Part C: Emerging Technologies, 65:111 – 127, 2016.
ISSN 0968-090X.
doi: https://doi.org/10.1016/j.trc.2016.01.013.
URL `http://www.sciencedirect.com/science/article/pii/S0968090X16000322`.

F. Malucelli.
Lectures' notes.
2016.

D. Margaritis, A. Anagnostopoulou, A. Tromaras, and M. Boile.
Electric commercial vehicles: Practical perspectives and future research directions.
Research in Transportation Business & Management, 18:4–10, 3 2016.
doi: https://doi.org/10.1016/j.rtbm.2016.01.005.
URL `http://www.sciencedirect.com/science/article/pii/S2210539516000067`.

A. Montoya, C. Guéret, J. E. Mendoza, and J. G. Villegas.
The electric vehicle routing problem with nonlinear charging function.
Transportation Research Part B: Methodological, pages –.
doi: https://doi.org/10.1016/j.trb.2017.02.004.
URL `http://www.sciencedirect.com/science/article/pii/S0191261516304556`.

S. Pelletier, O. Jabali, and G. Laporte.
Goods distribution with electric vehicles: Review and research perspectives.
In *Technical Report CIRRELT-2014-44*. CIRRELT, Montréal, Canada, 2014.

R. Roberti and M. Wen.
The electric traveling salesman problem with time windows.
Transportation Research Part E: Logistics and Transportation Review, 89:32 – 52, 2016.
ISSN 1366-5545.
doi: https://doi.org/10.1016/j.tre.2016.01.010.
URL `http://www.sciencedirect.com/science/article/pii/S1366554516000181`.

S. Savaresi.
Lectures' notes.
2016.

M. Schiffer and G. Walther.
The electric location routing problem with time windows and partial recharging.
European Journal of Operational Research, 260(3):995 – 1013, 2017.
ISSN 0377-2217.
doi: https://doi.org/10.1016/j.ejor.2017.01.011.
URL `http://www.sciencedirect.com/science/article/pii/S0377221717300346`.

T. M. Sweda, I. S. Dolinskaya, and D. Klabjan.
Optimal recharging policies for electric vehicles.
Transportation Science, 0(0):null, 0a.
doi: 10.1287/trsc.2015.0638.
URL `http://dx.doi.org/10.1287/trsc.2015.0638`.

T. M. Sweda, I. S. Dolinskaya, and D. Klabjan.
Adaptive routing and recharging policies for electric vehicles.
Transportation Science, 0(0):null, 0b.
doi: 10.1287/trsc.2016.0724.
URL `http://dx.doi.org/10.1287/trsc.2016.0724`.