POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering

Master's Degree Thesis

Analysis of logistic plants equipped with wireless charging systems for electric vehicles

Supervisors

Prof. Franco Lombardi
Dott.ssa Giulia Bruno
Ing. Alberto Faveto
Ing. Emiliano Traini

Candidate

Luca Cortese

Academic Year 2020-2021

Sincerely thanks to all the people
who made me able to reach this objective

Luca

ABSTRACT

Vehicles electrification is nowadays a valid way to reduce environmental pollution. However, one of the main problems related to battery-powered vehicles is the charging duration. Dynamic charging represents an excellent solution in reducing or even eliminating vehicle downtimes during the recharge. This technology can be efficiently applied to electric forklift trucks and used in logistic plants. Enermove S.r.l. is a company which has developed modular wireless charging units to be installed in warehouse flooring. Their solution can perform both static and dynamic wireless charging. This work aims to formulate a standardized methodology to determine the optimal wireless charging units layout to be installed. In the first part of the thesis, a mathematical-statistical model of the warehouse is proposed, based on the discretization of the operative area and the description of the storage/picking operations executed by the forklifts. Then, in order to find the best possible layout of the charging infrastructures, an Integer Linear Programming (ILP) problem is applied to the discrete warehouse model. The ILP optimal solution must respect some energetic requirements given by the customer and guarantee to be minimal in overall system cost. In the second part of the work, some verification studies have been performed to verify the range of application of the methodology and the validity of the provided results. Finally, the proposed approach is applied to a real-size case study, in order to provide evidence of the ease of usage and reliability of obtained results.

TABLE OF CONTENTS

Page

TABLES LIST

Table **Page**

FIGURES LIST

Figure **Page**

VARIABLES LIST

$i, j, k =$ Indices used, respectively, for nodes/edges, operations, bays;

b = Index used in bays section;

$\mathcal{N}$ = Set of all the nodes in the modelled warehouse;

$\mathcal{E}$ = Set of all the edges in the modelled warehouse;

$\mathcal{O}$ = Set of all the operations that can be executed in the modelled warehouse;

$\mathcal{B}$ = Set of all the bays in the modelled warehouse;

$\mathcal{H}$ = Set of all the horizontal corridors in the modelled warehouse;

$\mathcal{V}$ = Set of all the vertical corridors in the modelled warehouse;

$\mathcal{OP}_j$ = Set of the nodes crossed by the forklift during Outward Trip of operation j;

$\mathcal{RP}_j$ = Set of the nodes crossed by the forklift during Return Trip of operation j;

$\mathcal{OT}_j$ = Set of the Outward Trip intermediate nodes of operation j;

$\mathcal{RT}_j$ = Set of the Return Trip intermediate nodes of operation j;

$\mathcal{HN}_i$ = Set of Radius-2 horizontal neighbors of node i;

$\mathcal{VN}_i$ = Set of Radius-2 vertical neighbors of node i;

$\mathcal{HNN}_i$ = Set of Distance-5 horizontal neighbors of node i;

$\mathcal{VNN}_i$ = Set of Distance-5 vertical neighbors of node i;

N = Total number of nodes in the modelled warehouse;

E = Total number of edges in the modelled warehouse;

O = Total number of operations to be executed in the modelled warehouse;

B = Total number of bays in the modelled warehouse;

H = Total number of horizontal corridors in the modelled warehouse;

V = Total number of vertical corridors in the modelled warehouse;

HN_i = Number of Radius-2 horizontal neighbors of node i;

VN_i = Number of Radius-2 vertical neighbors of node i;

HNN_i = Number of Distance-5 horizontal neighbors of node i;

VNN_i = Number of Distance-5 vertical neighbors of node i;

cat_i = Category of node i;

$horc_i$ = Horizontal corridor of node i;

$verc_i$ = Vertical corridor of node i;

onf_i = Binary parameter indicating if an operation has to be executed on node i;

$n1_i$ = First node that edge i is connecting;

$n2_i$ = Second node that edge i is connecting;

ot_j = Operation time of operation j;

on_j = Operation node of operation j;

bt_j = Bay time of operation j;

b_j = Operation bay of operation j;

brt_j = Bay recharge time of operation j;

p_j = Operation probability of operation j;

inf_j = Intermediate node flag of operation j;

oot_j = Overall operation time of operation j;

bn_k = Bay node of bay k;

bwf_k = Bay WPT flag of bay k;

v_f = Forklift average speed;

n_s = Nodes spacing;

ct = Crossing time;

$k_{i,j}$ = Multiplicity of node i, that is how many times node i is present in either $\mathcal{OP}_j$ and $\mathcal{RP}_j$;

$TNTV$ = Total Node Times Vector;

$TNMTV$ = Total Node Movement Times Vector;

$TNOTV$ = Total Node Operation Times Vector;

$TBTV$ = Total Bay Times Vector;

$TBOTV$ = Total Bay Operation Times Vector;

$TBRTV$ = Total Bay Recharge Times Vector;

tnt_i = Total node time of node i;

$tnmt_i$ = Total node movement time of node i;

$tnot_i$ = Total node operation time of node i;

tbt_k = Total bay time of bay k;

$tbot_k$ = Total bay operation time of bay k;

$tbrt_k$ = Total bay recharge time of bay k;

$PNTM$ = Partial Node Times Matrix;

$PNMTM$ = Partial Node Movement Times Matrix;

$PNOTM$ = Partial Node Operation Times Matrix;

$PBTM$ = Partial Bay Times Matrix;

$PBOTM$ = Partial Bay Operation Times Matrix;

$PBRTM$ = Partial Bay Recharge Times Matrix;

pnt_{ij} = Partial node time of node i during operation j;

$pnmt_{ij}$ = Partial node movement time of node i during operation j;

$pnot_{ij}$ = Partial node operation time of node i during operation j;

pbt_{kj} = Partial bay time of bay k during operation j;

$pbot_{kj}$ = Partial bay operation time of bay k during operation j;

$pbrt_{kj}$ = Partial bay recharge time of bay k during operation j;

ST_{total} = Total duration of a shift;

ST_{breaks} = Total duration of the breaks in a shift;

ST_{eff} = Effective duration of the shift, i.e., duration of a shift excluding the breaks;

P_n = Nominal WPT system charging power;

η_{stat} = Static efficiency;

η_{dyn} = Dynamic efficiency;

$E_{battery,max}$ = Fully-charged battery stored energy;

$P_{cons,max}$ = Maximum forklift power consumption;

k_{breaks} = Fraction of break time forklifts are charged on static WPTs;

$P_{cons,bay,op}$ = Forklift power consumption when operating in bays;

$P_{cons,bay,rech}$ = Forklift power consumption when non-operating in bays;

$P_{cons,node,mov}$ = Forklift power consumption when moving in the warehouse;

$P_{cons,node,op}$ = Forklift power consumption when operating in the warehouse;

$E_{in,breaks}$ = Total energy intaked, in a shift, during breaks;

$E_{in,bays}$ = Total energy intaked, in a shift, from recharge in bays;

$E_{in,dynamic}$ = Total energy intaked, in a shift, from the recharge on "dynamic WPTs";

$E_{out,bay,op}$ = Energy lost during operation in bays;

$E_{out,bay,re}$ = Energy lost when the forklift is not operating in bays;

$E_{out,node,mov}$ = Energy lost due to forklift movement in the warehouse;

$E_{out,node,op}$ = Energy lost during forklift operation in the warehouse;

$E_{in,shift}$ = Total energy intaked in a shift;

$E_{out,shift}$ = Total energy lost in a shift;

ΔE_{shift} = Energy net balance in a shift;

$E_{battery,actual}$ = Actual energy stored in the battery;

SoC = Battery State-of-Charge;

ΔSoC_{shift} = Variation of forklift battery SoC during a shift;

$\Delta SoC_{desired}$ = Desired variation of forklift battery SoC during a shift;

$\Delta E_{shift,desired}$ = Minimum acceptable value of forklift battery energy variation in a shift;

$cost_{tot}$ = Cost, in euros, of the WPT system to be installed;

$cost_{stat}$ = Cost, in euros, of the static WPTs to be installed;

$cost_{dyn}$ = Cost, in euros, of the dynamic WPTs to be installed;

θ = Set of the optimization variables;

x_i = Optimization variable – presence of a WPT on node i;

h_i = Optimization variable – presence of an Horizontal WPT Part on node i;

hc_i = Optimization variable – presence of an Horizontal WPT Center on node i;

v_i = Optimization variable – presence of a Vertical WPT Part on node i;

vc_i = Optimization variable – presence of a Vertical WPT Center on node i;

xb_k = Optimization variable – presence of a static WPT in bay k;

c = Cost vector, indicating the cost of each optimization variable;

c_d = Cost of each dynamic WPT Part/Center;

c_s = Cost of each static WPT;

c_o = Fictitious null cost used in the cost function.

Chapter 1

Introduction

The problems of climate change, environmental pollution and greenhouse gases emissions have become even more relevant through this century. Many governments and institutions are oriented towards the direction of the decarbonification, and have set some milestones to achieve this goal in the course of next years. In 2011, the European Commission has published a roadmap for moving to a competitive low-carbon economy in 2050. The objectives are to cut domestic greenhouse gas emissions by at least 80% by 2050 compared to 1990. The EU targets to reduce greenhouse gas emissions inside the EU by at least 40% by 2030 [1]. Figure 1 shows the expected carbon emissions reduction trends from 1990 to 2050 for achieving these objectives.

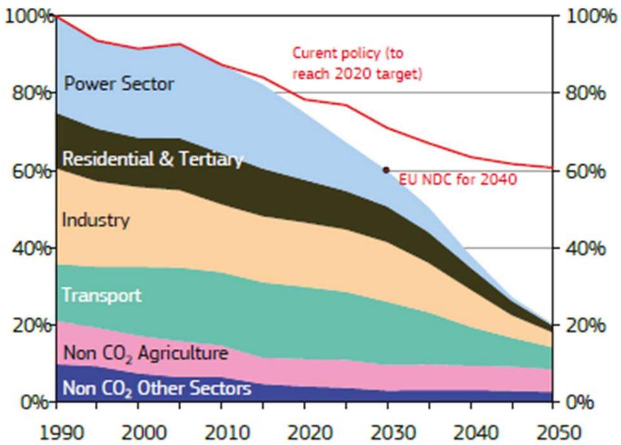

Figure 1: Reduction of carbon emissions goal for 2050 [1].

This concerns have led the rapid growth of Electric Vehicles (EV). Development plans with the aim of increasing EVs in developing countries have been proposed. Utilization of EVs has the potential to reduce pollution, reduce greenhouse gas emissions, and save fuel cost for EV owners [2]. Though, one of the main problems related to electric vehicles is the charging duration. Nowadays, there are various level of charging for EVs. A classification of them on the basis of the rate at which batteries are charged is shown in Table 1. EVs are more expensive compared to conventional vehicles, and charging time of EVs is higher compared to refueling of conventional vehicles [2].

Level	Battery Capacity (kWh)	Power Level	Charging Time (hours)
Level 1	5-15 16-50	1.4 kW (12 A) 1.9 kW (20 A)	4-11 11-36
Level 2	5-15 16-30 3-50	4 kW (17 A) 8 kW (32 A) 19.2 kW (80 A)	1-4 2-6 2-3
Level 3	20-50	50 100	0.4-1 0.2-0.5

Table 1: Level of charging for EVs and relative parameters [2].

Consequently, also the material handling equipment used in industries is shifting towards the electrification. Interest in and need for electric material handling vehicles (EMVs), such as electric forklifts and automatic guided vehicles (AGVs), are growing larger and larger [3]. One study forecasted the electric forklift market will grow more than four times, from 10000 units in 2016 to 47000 units in 2025 [4]. The main drivers of this change are the reduced operation cost and fueling associated with EMVs, limited energy consumption and reducing greenhouse gas emissions, even though movement into EMVs has higher upfront costs [3].

Forklifts are part of the industrial environment, and are used in daily tasks when moving heavy loads (e.g. pallets) from one place to another. They can be classified according to their power source: liquified petroleum gas (LPG), diesel or electric [5]. Diesel forklifts have been largely used because of their good durability and good torque. However, for indoor applications, zero-emissions is mandatory, which makes electric forklifts suitable for such uses. They also provide very high torque and zero noise pollution thanks to the electric motor. However, charging time is a big concern. Even if some improvements in the materials used to build the batteries, like the use of lithium-ion technology, or in the charging techniques, like the introduction of fast charging systems [6], have brung a significant reduction of the charging time, it still remain high compared to the other types of fueling listed before. Since forklifts are critical resources which directly influence the overall efficiency of any warehouse or manufacturing facility [7], and considering that battery charging introduces machine downtime periods leading to a decrease of such efficiency, it is of interest to find some strategies leading to the reduction, or even the elimination, of those unproductive time amounts.

One of the most promising technologies for strongly reducing EVs dead times during charging, and to foster a large EV diffusion is represented by the Dynamic Wireless Power Transfer (DWPT) technology, which is based on the magnetic coupling between coils installed under the ground level and a coil mounted under the vehicle floor, connected to vehicle battery by means of a power electronics converter [8]. Thanks to the absence of electric contacts, DWPT allows powering the EV while driving, eliminating the necessity of stops for the recharge [8]. Then, this technology could be applied to electric forklifts too, and can potentially solve their recharge problem, with the related machine downtimes and the consequent logistic plant efficiency loss.

Enermove S.r.l. is a company born in 2019 from the collaboration of the entrepreneur Flavio Cavallo with Paolo Guglielmi and Vincenzo Cirimele which are, respectively, a professor and a researcher at the Politecnico di Torino [6]. This company has developed a technology for the dynamic wireless charging of electric forklifts trucks, which is based on the installation of charging lanes under the warehouse flooring. Forklifts transiting over such lanes could be charged by them, without the need of stopping. Alongside, the company has also developed static wireless charging pads, which are optimized for the forklift recharge when they are parked over them. In essence, the combination of the static and dynamic charging systems developed by the company may really solve the forklifts charging problem which has been described up to now. However, in order to create a charging system that, when installed in a warehouse, is really able to satisfy the warehouse owners requirements about the total energy which can be actually transferred to forklifts during each working shift, some analyses of the warehouse logistic structure must be done.

This work is oriented to logistic plants which are served by battery-powered electric forklifts, which could be potentially charged by static or dynamic wireless power transfer systems. The objective of this work is the development of a mathematical-statistical model to discretize a warehouse and, on the basis of it, to realize an algorithm able to determine the number and the positions of the static or dynamic charging modules developed by Enermove S.r.l. to be installed in the warehouse. Such system must satisfy some energetic requirements, in terms of energy intake per working shift, given by the customer, and must have the lowest cost possible.

Chapter 4 describes the warehouse modelling step: aisles and lanes where forklifts can move are discretized using nodes and edges, so to create a mathematical graph. Then, all the picking/storage operations which must could be executed by the forklifts must be modelled, describing their duration, the path followed by the forklifts, and the probability to be executed with respect to the whole set.

Chapter 5 illustrates the procedure to estimate the zones of the warehouse which are more utilized by the forklifts, starting from the warehouse modelling elements. In principle, if we are interested to minimize the cost of the system, the dynamic charging modules must be preferentially placed in zones with high utilization, since forklifts spend more time over them.

Chapter 7 describes the optimization problem, which in this work is a Linear Programming one, developed to find the minimum-cost charging modules layout which satisfies the customer energetic requirements, starting from the utilization of each warehouse zone. To conclude, Chapter 8 presents four examples of verification of the whole procedure, applied to some ad-hoc designed warehouses and an existing one, to demonstrate the validity of the model, the optimality of the results in terms of cost, and the capability of the developed algorithm to be applied to almost any kind of existing warehouse, being able to calculate the optimal solution in a reasonable amount of time.

Chapter 2

State of the Art

2.1 WPT

2.1.1 Introduction on WPT Systems

It is possible to define as Wireless Power Transfer (WPT) the different ways to transfer energy at distance without wires [9]. Today, this definition covers several technologies in a wide range of applications, power and distances. Some of them are [10] [11]:

- Smartphones;
- PCs, tablets, audio players;
- Electric vehicles and trams;
- Medical applications, for devices implanted in human body like cardiac implants;
- UAVs, whose range can be significantly increased without changing their battery size;
- Consumer appliances like refrigerators, washing machines, air conditioners etc.

A first classification of WPT systems regards the distance at which they can transmit power. We can distinguish between far-field WPT and near-field WPT. The former is concerning about energy transfer at long distances. Microwave energy falls into this category. The latter concerns transmission of energy at short distances. We can classify near-field WPT systems into four categories, depending on the type of coupling used [10]:

- Magnetic resonant coupling;
- Inductive coupling;
- Capacitive coupling;
- Magneto dynamic coupling.

The first two are the most suitable ones, being good compromises in terms of system efficiency and physical size of the components. In this work, we will focus our attention on near-field WPT technologies for vehicular applications.

The technology developed by Enermove S.r.l. is called inductive power transfer (IPT). It basically consists of the inductive coupling between a coil above or below the ground, defined as the transmitter, and a movable coil placed under the vehicle, defined as the receiver. The transmitter is powered through a power electronics converter, which provides a high-frequency current and a high-frequency field. This field couples with the receiver and allows the wireless transfer of electrical power. Downstream the receiver, a rectification stage converts the signal to dc, which allows the battery of the vehicle to be charged. Thanks to the absence of electrical contact, the transmitter and receiver of an IPT are independent, so the recharge process can start automatically when the vehicle is over the transmitter. Moreover, the absence of electrical contacts eliminates the typical problems of electrical erosion and dust deposition, providing a more robust system with longer life cycle [9].

This technology can be used for EVs, for both static, when the vehicle is parked during charging, and dynamic, when energy transfer occurs during vehicle motion, applications. Dynamic charging, also known as Charging-While-Driving, can potentially solve the range anxiety problem related to the limited range a generic EV is able to cover. People are constantly scared about having enough charge in the battery to complete their trips, and about looking for charging stations nearby when battery is almost exhaust [12]. This problem results to be the most relevant for the negative social perception of EVs [13].

Efficiency of a WPT system can reach values around 90%, provided that there is a precise alignment of both transmitter and receiver coils [11]. Moreover, efficiency depends on the width of the gap between the two coils: the larger the gap, the lower the efficiency [14].

Behind this powerful technology, there is a big question on the safety of human body while exposed to large, and potentially dangerous, electric, magnetic, and EM fields generated by WPT systems. First of all, adverse effects on human health mainly depend on the frequency and intensity of EM field, so it is important to keep them well below the established human safety limits by the proper design of the systems. For resonant coupled WPTs, some human exposure guidelines can be found in [15], [16], [17]. It has been proved that human exposure to EM fields may increase the body temperature or may heat body tissues, and may stimulate muscle tissue and nerve, but there is no established evidence that this EM field causes cancer. However, no particular human health risks have been proved, if exposure limits required by the regulations are respected [18].

2.1.2 Starting Considerations

Feasibility of WPT systems for vehicular applications have been verified in several researchs [12] [19] [20]. However, most of them are related to outdoor applications, focusing on CWD of vehicles travelling on electrified road lanes. Although similar to them, no literature related to forklift CWD has been found, so this thesis represents a new case study of WPT application, for further improvements.

Figure 2: Electric forklift to which this study could be oriented [48].

The idea behind the foundation of Enermove S.r.l. started from the need to find a technical solution to the problem of the battery charging of industrial forklifts. Forklift batteries are typically of three types [5] [6]:

- Lead-acid batteries: are largely used due to their relatively low cost. Though, they present some downsides, such as a deep discharge, which is critical to the lifespan of the battery, and high weight, even if in forklift application they are used as a counterweight and help to maintain the center of gravity during operational lifts. Moreover, they create several problems to the plant since, during their charging, they generate harmful gases. So, charging stations must be positioned outside the plant, or even inside in dedicated areas. However, these zones are costly and space-consuming.
- Sealed lead-acid batteries: they eliminate the harmful gases emission problem. They can be charged in “standard” or “fast” mode, and both can be executed inside the plant. However, a machine downtime period is necessary during the charging operation.
- Lithium-ion batteries: Most modern solution. They have the same features described for the sealed lead-acid battery, but batteries are lighter and present better performances in terms of energy density, power discharge, cycle life, efficiency and charging operations. Though, they are more expensive.

The most diffused method to execute the recharge consists in the battery swap: the exhaust battery is substituted with a charged one. Though, this operation requires skilled personnel. For sealed lead-acid and lithium-ion batteries it is possible to use a single battery, that should be frequently partially charged using particular power supplies, able to provide a current up to three times higher than the nominal charging one. However, the drawback of this solution is represented by the high number of breaks needed for frequently charging the battery [6].

Figure 3: Examples of charging areas and battery swap operations [6].

2.1.3 System developed by Enermove S.r.l.

2.1.3.1 System description

The most relevant problem related with the "classic" battery charging technologies described before is due to the machine downtime periods, and consequently also the breaks the personnel have to do, for executing battery charging. These periods are "wasted time", and should be reduced as much as possible to maximize the plant income [6].

The solution is represented by the dynamic wireless charging. Enermove system is a WPT based on resonant magnetic coupling technology. The two coils, one placed on the floor (transmitter), and the other one mounted below the forklift (receiver), are magnetically coupled. Transmitter is powered by a DC/AC converter that produces a high-frequency ac current in that coil. The produced magnetic field couples with the receiver coil, and allows to transfer power to the receiver, contactless. Then, the two coils are connected to capacitors, so to create a resonant system. This choice allows to maximize the power transfer, to reduce cost and size of the power electronics, and to increase charging system efficiency.

This technology can be used to create a charging lane constituted by multiple transmitting coils, placed below plant pavement. These coils activate automatically when the forklift, and so the receiver, is over them, regardless if it is still or in movement. So, this system allows to continuously charge the vehicle, eliminating the charging breaks typical of the battery swap method. Moreover, the power needed for this application is similar to the one for the slow charging of batteries, and so it does not require any electrical system improvement, which typically must be realized during the installation of fast charging systems.

Figure 4: Scheme of Enermove technology principles [6].

Note that, even if the described WPT system is suitable for charging a forklift battery regardless if forklift is still or moving over the transmitter, it is not optimal for static charging. As a consequence, Enermove has developed another type of transmitter, adapt for executing static charging, based on the same principles as the already described system, but with different features. Details about the two types of transmitter can be found in section 2.1.3.3. For simplicity, in this work we will refer to *Dynamic WPT* for indicating the first system type, suitable for dynamic charging, and to *Static WPT* to indicate the second type, adapt for static recharge.

2.1.3.2 Advantages

Advantages of this technology are as follows [6]:

- It is not mounted in dedicated areas of the plant;
- No personnel dedicated to battery charging is needed;
- Electrical system does not need modifications;
- May be used by more than one forklift;
- Is particularly suitable to be used in combination with renewable sources electric energy generation systems, especially photovoltaic systems, since it allows the produced electricity to be immediately used, without storing it;
- Eliminates dead times resulting from battery charging/swapping operations;
- It may allow a significative personnel cost reduction.

Moreover, since WPTs are modular, the system can be customized on the basis on the plant layout and the customer energetic requirements.

2.1.3.3 Technical parameters and considerations

Working at high frequencies, the coils are made using litz wires technologies: a series of thin, twisted wires, each one electrically insulated. This technology allows to reduce losses caused by skin effect and proximity effect [6].

A dynamic transmitting coil has a length of 120 cm, and a width of 20 cm. Receiver has a square surface, with 35 cm side. Figure 5 shows them. This shape has been used to reduce magnetic coupling with undesired features, like the iron present in the electro-welded meshes embedded in the industrial flooring. The selected shape also allows mis-alignments up to 20% between the two coils. It will be driver's responsibility to drive the forklift truck as much as possible over the transmitter, so to maximize the efficiency. However, since the dynamic transmitter has a rectangular surface and the receiver a square one, some losses arise due to the fact that not all the magnetic field lines generated by the transmitting coil can be captured by the receiver. This problem is much more evident during the static charge than the dynamic. From here, the need to develop a static transmitter as a square pad with side length ranging from 20 to 60 cm, to be used for static recharge only. It is reported in Figure 6.

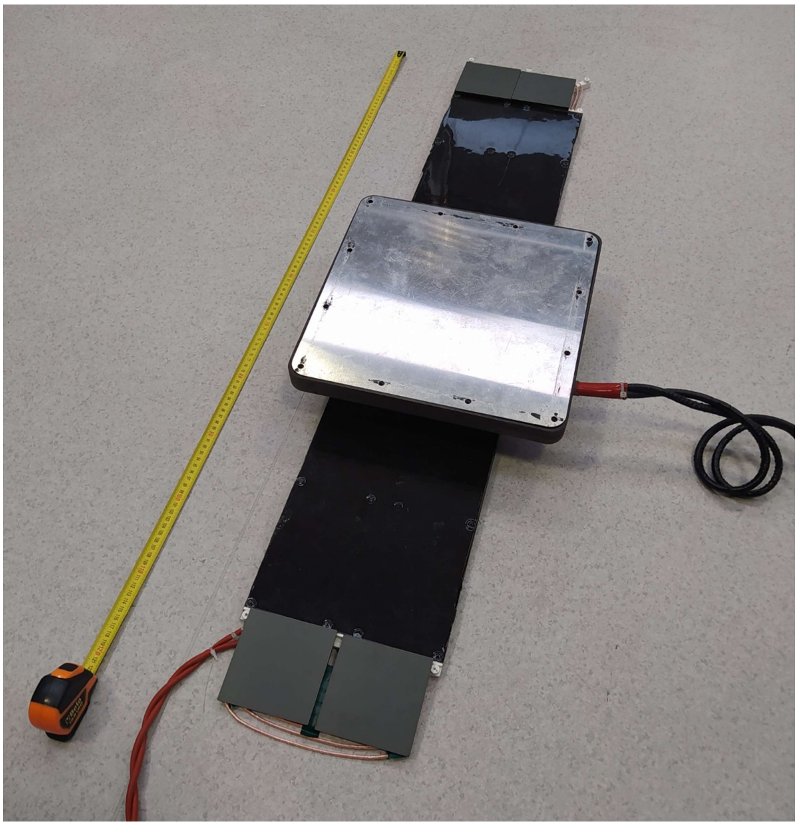

Figure 5: Dynamic transmitter coil (the black one), and receiver (the grey one), developed by Enermove S.r.l [48].

Figure 6: Static transmitter developed by Enermove S.r.l. [48].

The charging power can range from 1 kW to 14 kW, according to customer requirement. Recall that, for high charging powers, the plant electrical system might need some improvements. Efficiency of a dynamic WPT is around 87%, while for a static WPT is around 90%.

Installation of dynamic transmitting coils just requires a little scratching of the concrete superficial layer, so to create a slot where to place the coil. Then, the coil has to be resined, and subsequently covered by concrete. No further maintenance operations are required. On the other side, a static transmitter may also be mounted over the floor, without creating any slot in the concrete.

Note that batteries, especially the lead-acid ones, do not tolerate power supply discontinuities. Those discontinuities can occur not only in the lateral direction of the coil (due to excessive misalignment), but also in the longitudinal one, between two adjacent WPTs. To prevent this problem, relevant during dynamic charging, a technique called *Overlap* has been used: adjacent dynamic transmitting coils must be partially overlapped. In this way, magnetic decoupling between those coils is obtained. So, we can consider a dynamic WPT module as formed by two adjacent, overlapped, dynamic transmitters. Moreover, this technique allows to turn on just the WPT over which the forklift is located, and to supply the adjacent one when forklift enters into the overlapped zone.

Note that the dimension of the transmitters listed at the beginning of this section, and so the WPT module length, can slightly vary according to the case scenario. In this work, each dynamic module has been considered 2.5 m long. The cost of each dynamic WPT transmitting module is 4000 €. A static WPT transmitter costs 3000 €, while the cost of the receiver to be mounted on the forklift is 1000 €.

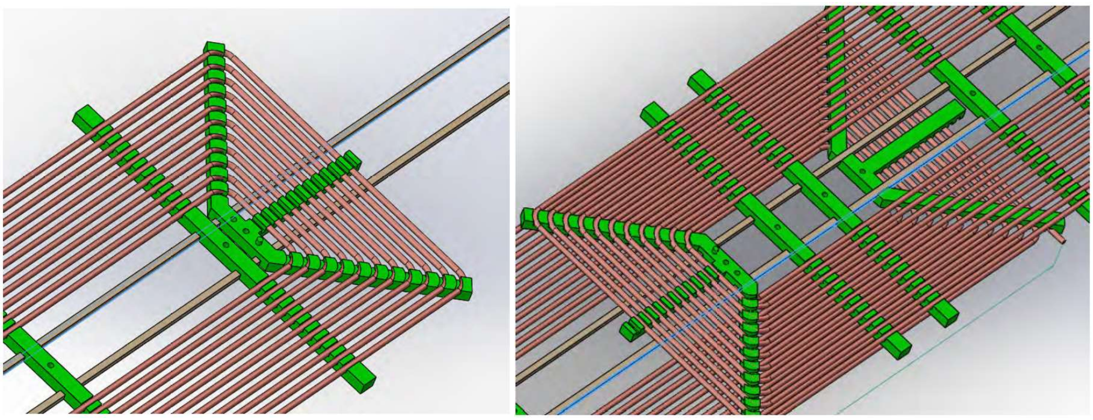

Figure 7: CAD of transmitting coil with supports (left) and CAD of transmitting coils with Overlap (right) [6].

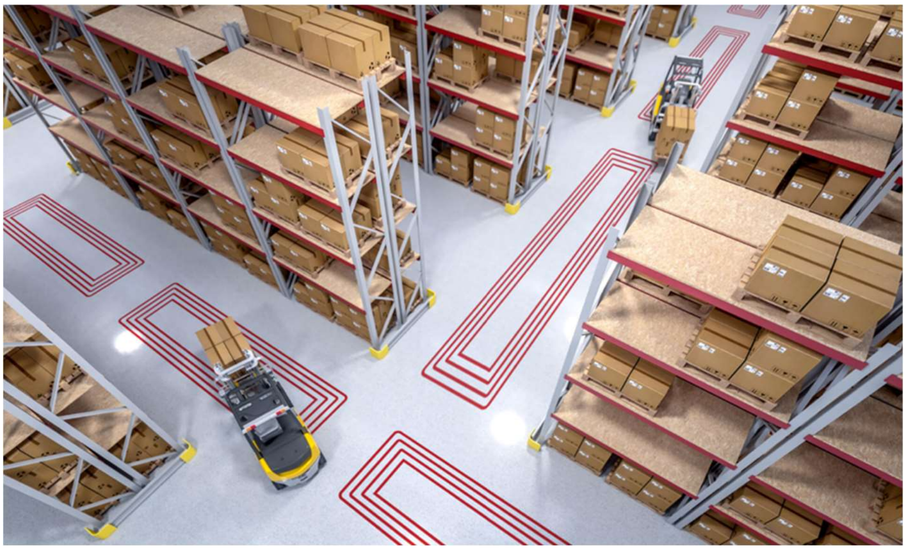

Figure 8: Possible application of the dynamic WPT system, for exemplification purposes only. Red rectangles drawn over the floor indicate the dynamic WPT lanes [48].

2.1.3.4 Safety issues

Forklifts and personnel transiting over a WPT system may be exposed to the produced EM field. However, shielding is guaranteed by the forklift frame, with the aid of an aluminum plate located among receiver coil and forklift truck bed. Note that this shielding is necessary also for avoiding the field to interact with on-board forklift electronics. Moreover, due to the automatic switching of the WPT system whenever a forklift is located over a transmitter, any worker transiting over the transmitting coil would find the system not in operation, and would therefore not be immersed in a magnetic field.

2.2 Optimization

Talking about generic road EVs applications, the potentialities of the WPT technologies are clear, in particular the ones of dynamic charging. First of all it can potentially solve the range anxiety problem which substantially decrease people desire of owning an EV.

Chen et al. [21] have compared the deployment of charging lanes, equipped with dynamic wireless power transfer systems, with respect to the installation of charging stations, to explore the competitiveness of charging lanes, which are more costly compared to charging stations. From a cost analysis it results that charging lanes can be more attractive than charging stations, for users wanting to reach their destination in the shortest time possible. Moreover, in the private provision scenario, to operate charging lanes results to be more profitable than operating charging stations [21].

However, the biggest problem related to mass deployment of dynamic WPTs under road lanes is their cost. In [22], an analysis of a CWD system installation on California freeways, estimates a system cost of $4 millions per lane mile. Considering the global extension of roads, the overall CWD system installation cost may be prohibitive. From here, the need to rely on optimization algorithms to be able, given a maximum cost the WPT system customer is willing to spend, to maximize the energy the system is able to provide to EVs transiting over it. A good starting point could be to be more likely to place WPT systems in zones where EVs spend more time. This is the idea followed by Khan et al. in [12]: in their work, the placement of dynamic wireless charging units at signalized intersection lanes has been investigated, being zones where vehicles stop frequently. The stopping time could allow the EV to be charged by them. The goal of that work is the formulation of an optimization problem which aims, among other things, to place wireless charging units at signalized intersection lanes, so to maximize the utility (the total amount of energy transferred to EVs by a wireless charging unit) within a certain budget [12].

2.2.1 Operation Research

The definition of Operation Research is not unique: Churchman et al. [23], defines OR/MS[1] as the application of scientific methods, techniques and tools to problems involving the operations of systems so as to provide those in control of the operations with optimum solutions to the problems. Winston [24] defines it as a scientific approach to decision making, which seeks to determine how best to design and operate a system, usually under conditions requiring the allocation of scarce resources. The definition used in [25] is: “a scientific approach to aid decisions making in complex systems”.

OR includes a wide set of methods and algorithms. Among them, the Linear Programming can be found. Being the very basic but powerful tool of OR, it involves the general problem of allocating limited resources among competing activities in the best possible way [26]. We will go into its details in section 2.2.4.

Churchman et al. [27] presents an OR process as a set of six stages [28]:

1. Formulating the problem;
2. Constructing a mathematical model;
3. Deriving a solution;
4. Testing the model and the solution;
5. Establishing controls over the solution;
6. Putting the solution to work – implementation.

Nowadays, these problems can be solved with the aid of dedicated softwares.

[1] Operation Research is also called Management Science

2.2.2 Algorithms efficiency and problems complexity

An important aspect to be taken into account when trying to solve Operation Research problems regards the efficiency of the algorithm that is being used, the problem complexity, and the computational time required [29].

Given a particular algorithm, its performance is strongly dependent on the size of the problem being solved. The size of a problem is related to the number of variables and of the relations between them which are mathematically stated: the higher their number, the larger the problem size. For this reason, the performance of an algorithm is often described as a function of a variable denoting the problem size.

Two classes of problems have been defined [29]:

- "P" Class: contains those problems that can be solved by an algorithm within an amount of computational time proportional to some polynomial function of problem size. These problems are solvable by *polynomial-time algorithms*. They are considered to be "easy" problems, in the sense that there exists efficient algorithms able to execute the problem in a reasonably small amount of time.
- "NP" Class: contains problems that may require the computation time to be proportional to some exponential (or larger) function of problem size. These problems are solvable by *exponential-time algorithms*. This means that unacceptably large amounts of computation time might be required for solving these problems, making them unsolvable in practice even if solutions for them exist. Note that the NP Class includes several subclasses, like NP-hard and NP-complete, each of them characterizing a different level of complexity.

Note that, depending on the nature of the data, the execution time for a given algorithm may vary. Due to this reason, computation time required to execute an algorithm should therefore consider the worst case performance, that is, the greatest number of steps that may be necessary for guaranteed completion of its execution [29]. For this purpose, we can introduce the *big-Oh* notation: an algorithm is said to be $O(f(n))$ if there exist constants c and n_0 such that, for all $n > n_0$, the execution time is $\leq c \cdot f(n)$, where:

- n indicates problem size;
- $f(n)$ a function of the problem size, representing the algorithm worst step count (the maximum number of steps necessary for its execution);
- c is a constant introduced to account for the extraneous factors influencing the execution time, such as hardware speed, computer system load etc.;
- n_0 is a threshold accounting for the fact that, for very small problem size n, the alghorithm may not reveal its characteristics worst case performance.

As a matter of example, Table 2 reports computational time versus problem size, for different algorithm complexities $f(n)$.

$f(n)$	$n = 10$	$n = 20$	$n = 50$	$n = 100$
n	10 s	20 s	50 s	100 s
n^2	100 s	400 s ≈ 7 min	2500 s ≈ 42 min	10000 s ≈ 2.8 h
n^3	1000 s ≈ 17 min	8000 s ≈ 2 h	125000 s ≈ 35 h	10^6 s ≈ 12 d
2^n	1024 s ≈ 17 min	1048576 s ≈ 12 d	$1.126 \cdot 10^{15}$ s ≈ 350000 centuries	$1.268 \cdot 10^{30}$ s ≈ 10^{21} centuries
$n!$	3628800 s ≈ 1 month	$2.433 \cdot 10^{18}$ s ≈ 10^9 centuries	$3.041 \cdot 10^{64}$ s ≈ 10^{55} centuries	
n^n	10^{10} s ≈ 300 yr	$1.049 \cdot 10^{26}$ s ≈ 10^{17} centuries	$8.882 \cdot 10^{84}$ ≈ 10^{75} centuries	

Table 2: Computational time requested by different algorithms, characterized by complexities f(n), for different problem size n. Assume function f(n) denotes step count of the algorithm, and that each step can be executed in 1 second on a computer [29].

2.2.3 Heuristic Techniques

Looking at Table 2, it is clear that, for some kinds of algorithms, a "large" problem size will result in a prohibitive computation time. This is evident in many practical problems in science, engineering and management, in which the only way to be sure of finding an optimal solution is to search completely through the whole set of possible solutions. The idea of analyzing all the solutions is tempting, since we can be sure to be able to select the optimal one, but the computational time required to complete the search might make it impossible.

Heuristics methods can be used to simplify the searching process, so that it is no longer a complete search over all the possible solutions, but rather it reduces to an affordable search that is likely to discover a good, or near-optimal, solution. These methods can be applied to computationally intractable NP problems, making possible to solve them in a reasonable amount of time. However, the drawback is represented by the uncertainty of actually having obtained the optimal solution or not. In fact, they are likely to fall into local optima, leading to the calculation of suboptimal solutions. Metaheuristics could be used to escape such not optimal solution computed through heuristic methods, introducing more intelligent searching techniques [29].

In the work developed here, the possibility of using heuristics is allowed. It is up to the user to decide whether the algorithm should examine all the possible solutions, or it should stop whenever one or more heuristic solutions have been found. It was experimentally found that, in many cases, the allowance of heuristic search reduces significantly the computation time requested by the algorithm, although being not sure about the optimality of the found solution.

2.2.4 Linear Programming Problems

Linear Programming (LP) deals with the problem of minimizing or maximizing a linear function in the presence of linear equality and/or inequality constraints. Informations on linear programming problems structure have been taken by [30]-[33].

Given a set of real numbers $a_1, a_2, \dots, a_n$ we define a linear function f on those variables as:

$$f(x_1, x_2, \dots, x_n) = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n = \sum_{j=1}^{n} a_j x_j$$

If b is a real number and f is a linear function, then the equation:

$$f(x_1, x_2, \dots, x_n) = b$$

is a linear equality, and the inequalities:

$$f(x_1, x_2, \dots, x_n) \leq b$$

and:

$$f(x_1, x_2, \dots, x_n) \geq b$$

are linear inequalities. We can use the general term *linear constraints* (or, in this work, simply *constraints*) to denote either linear equalities or linear inequalities. Note that in linear programming, strict inequalities are not allowed.

Formally, a linear-programming problem is the problem of either minimizing or maximizing a linear function subject to a finite set of linear constraints. If the function has to be minimized, then we call the linear program a minimization linear program, and if we are to maximize, we call the linear program a maximization linear program.

In this work, we are dealing with a minimization problem, that is, to minimize the cost of the WPT system to be installed, subject to a set of constraints. The generic minimization linear-programming problem can be mathematically written as:

$$\text{minimize: } \sum_{j=1}^{n} c_j x_j$$

subject to:

$$\sum_{j=1}^{n} a_{i,j} x_j \leq b_i \quad \forall i = 1, \dots, l$$

$$\sum_{j=1}^{n} d_{i,j} x_j = g_i \quad \forall i = 1, \dots, m$$

$$lb_j \leq x_j \leq ub_j$$

which, in a more compact form, can be rewritten as:

$$\text{minimize: } c^T x$$

subject to:

$$Ax \leq b$$
$$A_{eq}x = b_{eq}$$
$$lb \leq x \leq ub$$

with:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{1,n} \\ \vdots & \vdots & & \vdots \\ a_{l,1} & a_{l,2} & \cdots & a_{l,n} \end{bmatrix}, \qquad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_l \end{bmatrix}$$

$$A_{eq} = \begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,n} \\ d_{2,1} & d_{2,2} & \cdots & d_{1,n} \\ \vdots & \vdots & & \vdots \\ d_{m,1} & d_{m,2} & \cdots & d_{m,n} \end{bmatrix}, \qquad b_{eq} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_m \end{bmatrix}$$

The function $f = \sum_{j=1}^{n} c_j x_j$ is called objective function, and represents the functional to be minimized. Coefficients $c_1, c_2, \dots, c_n$ are the (known) cost coefficients, while $x_1, x_2, \dots, x_n$ are the decision variables. We can call any setting of the decision variables that satisfies all the constraints a *feasible point* or a *feasible solution* to the linear program. The set of all such settings constitutes the feasible region or the feasible space. Using this terminology, the minimization linar programming problem can be stated as follows: Among all the feasible solutions, find one that minimizes the objective function.

The decision variables are typically split into three main categories:

- Positive real variables ($x_j \geq 0$);
- Positive integer variables ($x_j \geq 0$, integer);
- Binary Variables ($x_j \in \{0,1\}$).

If all the decision variables are integer, the problem is said to be an integer linear programming (ILP). If there is a mix of both real and integer decision variables, the problem is a mixed-integer linear programming problem (MILP). All the decision variables used in this work are integer (more specifically, binary), making the problem an integer linear programming.

Popularity of linear programming can be attributed to many factors, including its ability to model large and complex problems, and the ability of users to solve such problems in a reasonable amount of time, by the use of effective algorithms and modern computers. Linear programming has a large number of applications. Floudas and Lin [34] have shown the advantages of mixed-integer linear programming based approaches for the scheduling of chemical processing systems. Richards and How [35] have developed a model of aircraft dynamics using linear constraints only, enabling to use a MILP approach to be applied to aircraft collision avoidance. Moreover, books are filled with examples of the utility of linear programming problems to solve common everyday life problems.

2.3 Overview on warehouses and storage systems

Since this work is oriented to logistic plants, a brief introduction about warehouse layouts and picking policies is necessary, in order to understand the modelling strategy that was developed and used in this work.

2.3.1 Warehouse Layouts

First of all, we can define *layout* of a warehouse the arrangement of storage locations and aisles [36]. It has a significant impact on order-picking and traveling distances in the warehouse, so an appropriate choice of it is a key element for optimization tasks [37] [38]. It has been found out that layout design has more than 60% effect on the total travel distance [37].

The *traditional warehouse layout* is the most common nowadays [39]. It has a rectangular form, with parallel straight aisles, called picking aisles. Objects are stored on racks/shelves placed along both sides of them. Two cross aisles, orthogonal to picking ones, are located at each end of them, allowing to move from one picking aisle to another. Modifications of this basic form are possible by adding one or more cross aisles, creating the so-called multiple-block layout [36] [39] [40]. An example of traditional warehouse layout, alongside some modifications of them adding a cross aisle, are reported in Figure 9.

Other warehouse layouts exists, with the presence of non-orthogonal aisles. Among them, we can list the *Flying-V* layout and the *Fishbone Layout*, which can offer a 10% to 20% reduction of travel distance with respect to traditional layouts. Another option is the *Inverted-V Layout*, which can bring another 3% saving of traveling distance [37]. A drawback of the Fishbone design is the limited access to the storage space due to the single, central, Pickup & Deposit (P&D) point. Therefore, another design called *Chevron Layout* has been proposed, with expected traveling distances similar to the Fishbone Layout [39] [41]. All these non-traditional layouts are depicted in Figure 9 and 10.

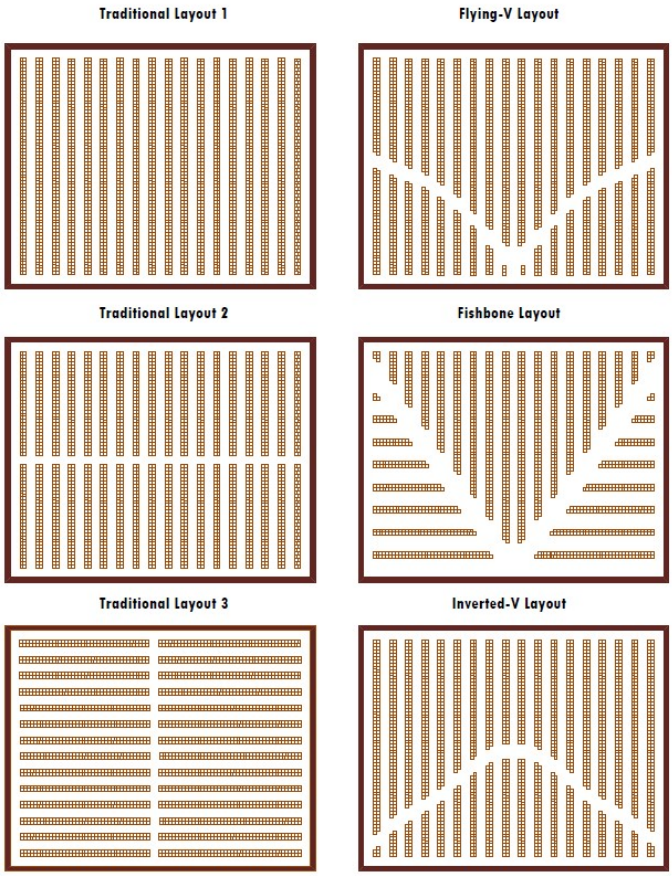

Figure 9: Possible warehouse layouts [37].

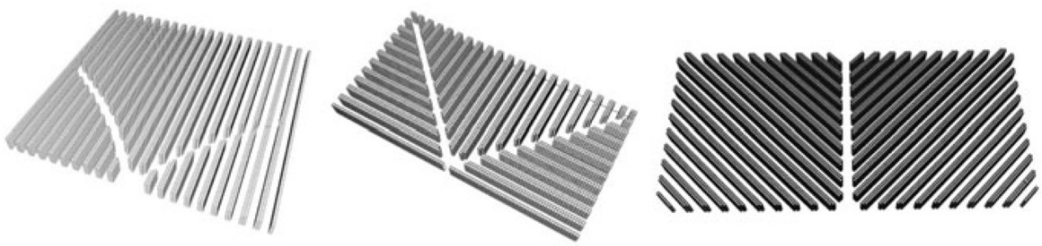

Figure 10: Non-traditional warehouse layouts. From left to right: Flying-V layout, Fishbone layout, Chevron layout [41].

Note that warehouse layout is strictly connected with aisles design, which tend to be narrow so to increase the space utilization with minimal costs, although this can lead to higher operational cost and more congestion among workers [37].

2.3.2 Storing assignment and Routing Strategies

There are two basic storing assignment policies: *random* strategy allows to store a pallet on an arbitrary empty location with the same probability, rather than on the closest empty location, while *dedicated* strategy allows to store pallets only on specified locations. The organization of those locations may exploit *class-based storage*, where items are clustered according to the frequency of orders, placing the most frequently requested items close to input/output gates. Another possibility is to use *family grouping*, where the goods are clustered according to similarities between either products or orders [37].

Single-order picking, the strategy where only one order at a time can be picked, is one of the most used policies. However, routing policies must be likely to select optimal travel paths for order picking, to maximize the efficiency [37]. Some common routing methods are reported in Figure 11.

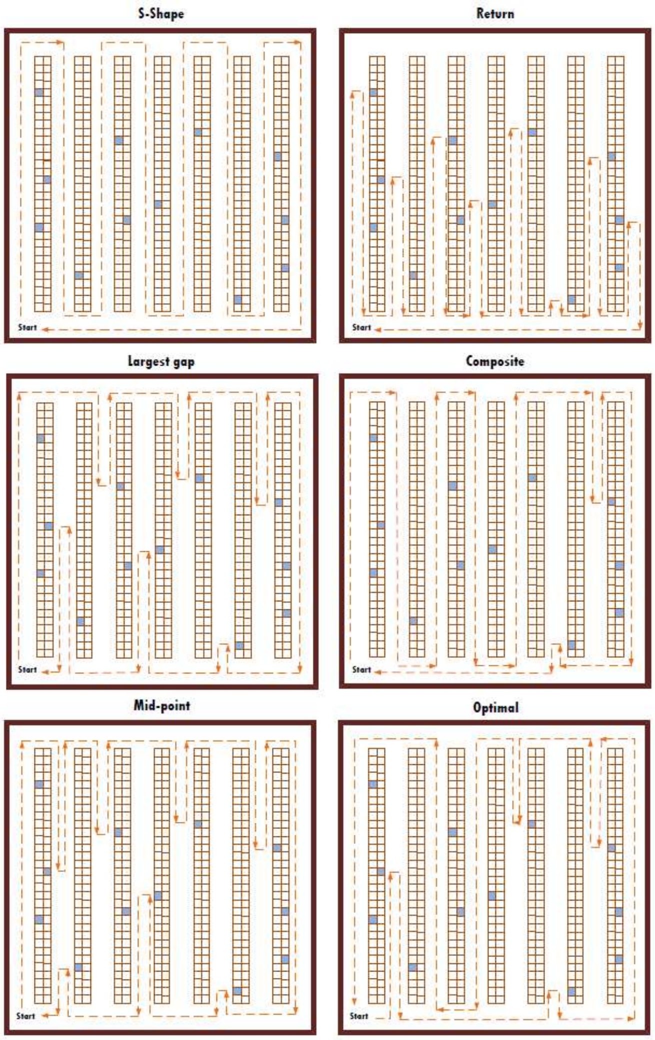

Figure 11: Common routing strategies [37].

We can now distinguish between two policies to store and retrieve the pallets in a generic warehouse, that are called *single-command* and *dual-command* operations [36]. In the former, workers travel from a P&D point to a single location in the warehouse, where they store or retrieve a single pallet before returning. One half of their travel is unloaded, so unproductive. In the latter, storage and retrieval operations are executed in the same cycle: workers perform a storage operation and then continue directly to the a retrieval location before returning to the P&D point. This policy reduces the empty fork-lift travel from half of the total travel distance to about one third. Pohl et al. [36] have demonstrated the higher efficiency of dual-command with respect to single-command, with savings in the range of 16-33% over a variety of shapes and sizes of the basic traditional layout. It has been found out that the maximum saving occurs for very tall warehouses with few aisles [36] [39]. Nonetheless, single-command operations are still common, due to the unavoidable and sometimes purposeful imbalances in receiving and shipping workloads [36].

2.3.3 Experimental analyses

Pohl et al. [36] have carried out an analysis on three warehouses with a traditional layout, with and without a central cross aisle. They are reported in Figure 12.

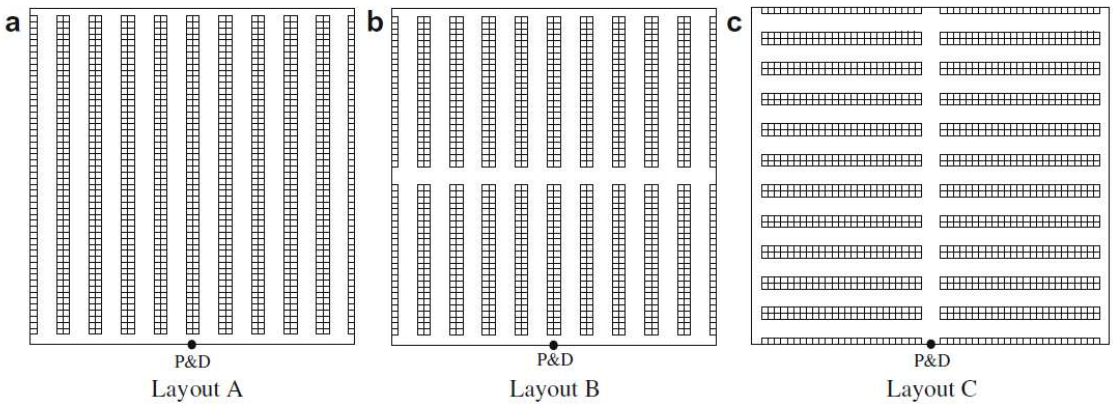

Figure 12: Traditional warehouse layouts analyzed by Pohl et al. in [36].

It has been demonstrated that Layout A is a better choice than Layout B for a unit-load warehouse performing strictly single-command cycles, while a properly configured Layout C could be equivalent to Layout A, under the same conditions. For dual-commmand operations, Layout C always outperforms Layout A. Also Layout B results to be more efficient than Layout A for all but very small warehouses. The most interesting aspect is that, even if Layout C results to be more efficient than Layout B for a wide range of parameters, the latter is much more common in practice than Layout C. This may be due to the fact that Layout C is more dependent on the assumption of one, central P&D

location. Moreover, it has been found out that the optimal location of the "middle cross aisle" of Layout B is not in the middle, but above it [36].

Dukic and Opetuk [39] have carried out an analysis on different warehouse layouts, to compare picking performance on non-traditional layouts with respect to the traditional one. Four layouts were analyzed: Traditional, Traditional with middle cross aisle, Fishbone, Chevron. Refer to [39] for the warehouse maps. S-shape and composite routing policies were used. Note that the latter is capable to establish whether picking in the aisle is done by entirely traversing it or by making a return route. The result of the analysis, carried out assuming to complete two order sizes of, respectively, 10 and 30 objects randomly located in each layout, are reported in Table 3.

S-shape routing policy		**Order Size**		**Composite routing policy**		**Order size**	
		10	**30**			**10**	**30**
Warehouse layout	Traditional (basic)	258.7	375.8	Warehouse layout	Traditional (basic)	228.2	363.9
	Traditional (middle cross-aisle)	193.9	329.0		Traditional (middle cross-aisle)	182.8	309.0
	Fishbone	227.5	351.9		Fishbone	213.1	317.3
	Chevron	268.5	397.2		Chevron	233.2	370.2

Table 3: Results of the analysis of [39] about average picking travel distance (in meters) versus order size, for different layouts.

Since the lower the picking travel distance, the more efficient is the layout in analysis, this analysis have demonstrated the higher efficiency of the traditional layout with middle-cross aisle with respect to non-traditional layouts. Moreover, the presence of a cross-aisle significantly increases the efficiency of the basic traditional layout. However, efficiency of fishbone layout is comparable to the one of traditional middle cross-aisle layout, especially for higher order sizes. Moreover, composite routing policy is confirmed to be superior in performance with respect to the S-shape one [39].

Chapter 3

Model Overview

As said in the introduction, the objective of this work is the development of a model for determining the optimal number and positions of the WPT modules developed by Enermove S.r.l. to be mounted in the warehouse. From now on, we will refer to the WPT modules as simply "WPTs". Such layout must satisfy some energetic requirements given by the customer, at the minimum cost possible. Both static WPTs and dynamic WPTs can be handled by the model, and the limitations to their placement will be listed later on, when talking about warehouse modelling.

The model was created under the assumption that the entities executing the picking/storage operations are human-driven electric forklifts. However, the procedure can be easily extended to other kinds of picking entities, like AGVs. Forklift are assumed to move with constant speed in the warehouse, neglecting the effects of accelerations and decelerations. Though, the latter should be considered while calculating forklift average speed v_f.

Forklifts are powered by batteries, which capacity must be specified. They discharge while forklift is moving, and they can recharge for all the time the forklift is located over a WPT, provided that the forklift receiver is aligned with the transmitter under the floor. If dealing with human-driven forklifts, the capacity to guarantee such alignment is up to the driver. It is assumed here that the alignment is enough to guarantee that the system efficiency coincides with the nominal one. Recall that, as said before, the static WPT efficiency is 90%, while the dynamic WPT efficiency is 87%.

The energetic requirement to be satisfied is assumed to be the variation of forklift battery SoC at the end of a generic working shift with respect to the beginning of it. Duration of the working shift, alongside the number and duration of the breaks have to be specified. The values of SoC at the beginning or at the end of the shift are not relevant, even if it is assumed that the initial battery SoC is enough not to let it to completely discharge during a generic shift. The variation of SoC is then measured through an energetic analysis of all the forklift movements during the shift, considering the overall amount of energy lost due to forklift movement, forks lifting/lowering etc., and the energy intake due to charging by means of the WPTs.

The key idea behind the formulation of the problem is that, to minimize the cost, WPTs must be preferentially placed in zones of the warehouse where the forklifts spend more time. A probabilistic approach has been used to estimate the time a forklift spends in each warehouse zone. The strategy used to realize this goal is to keep track of all the forklift operations that can be executed in the warehouse. In this work, we will call "operation" each picking/storage operation. Just an item per time can be handled during each operation (single-order picking), which is reasonable if we assume that unit loads used in the warehouse are constituted by pallets. Each item in the warehouse is assumed as located in a fixed position, and has a certain picking frequency, referred to how many times the item has been picked in a certain period. Knowing the picking frequency of all the items in the warehouse, we can assign to each of them a certain picking probability. Consequently, since each item per time can be picked/stored, the associated operation has the same probability to be executed. Due to the need to estimate a probability for each different operation, single-command picking policy has been assumed. Note that the same probability has been assumed for both picking and storage of an item.

Each operation begins and ends in a loading bay, where trucks are located while withstanding loading or unloading. There is no limit in the number and position of the bays which can be considered in this work. An item is assumed to be sent always to the same bay, and its associated operation is assumed to start and end in that bay.

Let us consider a generic picking operation: at its beginning, the forklift is located in the loading bay. Then, it moves through the warehouse following a determined route until it reaches the position of the item to be picked, and it spends some time there while loading the item on the forks. After that, it moves to the same loading bay to deposit the item on the truck and, after having reached the bay, it spends some time in loading the truck with the item. On the other side, a storage operation consists of the following steps: loading of the item on the forklift from the truck, motion to the point where the item has to be stored (considered the same as the picking point), unloading of the item into its storage location, and finally forklift motion to the bay.

If we assume that the loading/unloading time of an item from the truck are equal, and the same for the picking/storage of the item from the forklift to its storage point, we can conclude that, under the overall duration of the operation point of view, there are no differences between a picking or a storage operation. Moreover, if we consider the same probability of picking or storing an item, there is no more need to distinguish between them, and they can be treated in the same way. For this reason, when referring to a generic operation, there is no need to specify if it is referred to the item picking or storage.

By analyzing all the operations, and the path followed by the forklift in each of them, it is possible to determine, statistically, how much time the forklifts spend in each zone of the warehouse. Note that warehouse zones usage strongly depends on the different operations probabilities.

Once those times have been calculated, an integer linear programming problem is used to find the optimal WPT positions to satisfy the desired forklift battery SoC difference constraint. Such optimization algorithm is based on an energy balance between the energy lost during a shift, because of the forklift motion, and the energy intake from the wireless charging system for all the time a forklift is located over zones covered by WPTs. As said before, in principle, to minimize the cost of the system to be installed, the WPTs must be located mainly in the zones of the warehouse where forklifts spend more time, estimated by the analysis of all the possible operations. However, there might be some restrictions on the WPT placement. Some of them are listed below:

- Dynamic WPT modules have a predetermined length, so they cannot be installed in zones where not enough space is available;
- WPTs must be connected to the industrial grid, so they cannot be placed in zones where such connection is not possible, or it would require too large costs;
- Dynamic WPTs cannot be positioned in zones where the presence of in-floor structures (like pipes, wires etc.) obstruct their placement;
- The company policy does not allow the placement of single dynamic WPT modules in the warehouse. Where deemed necessary, at least two adjacent modules must be placed.

All these limitations must be considered when calculating the optimal WPT layout, and so they must be mathematically introduced in the LP problem.

The output of the optimization process is a list of the positions where a certain number of static or dynamic WPTs must be installed, alongside the total cost of the system. If an optimal solution is found, that system of WPT is able, in principle, to leave the forklift batteries at the end of each working shift with a residual SoC which respects the customer requirements. Such optimal solution guarantees also minimality of the system cost. Note that there may be cases in which no WPT layout is able to satisfy the desired battery SoC. An example could be a situation in which a large portion of the warehouse lanes cannot be equipped with WPTs. However, the most important point regarding the methodology is that the WPT layout proposed by the optimization process must be feasible, which means that all the WPTs can be physically installed in the indicated locations.

3.1 Summary of the Model key points

This section aims at providing a brief recap of the key points of this work, for introductory purposes. All the details about them can be found in the next sections.

The first step of the methodology is about the mathematical modelling the warehouse in analysis, which main objective is to specify in which zones of the warehouse WPTs can be installed. Such procedure is called *Warehouse Modelling*, and it is about the discretization of the warehouse, the definition of all the operation details and the paths followed by the forklifts during each of them. All the details about it can be found in chapter 4.

After having modelled the warehouse, the step of calculating, statistically, the time spent by the forklift in each discretized warehouse zone begins. In this work, we will call *Total Times* the amount of time spent by the forklift in each discretized warehouse zone, with reference to the overall working shift duration. Since we are working in a probabilistic framework, we can also define *Total Time* of a certain discretized warehouse zone the probability, at every time instant, to find the forklift located in that zone. Total Times calculation is executed by analyzing one by one all the operations specified in the Warehouse Modelling step. All the details are reported in chapter 5.

The final step is to formulate and to solve the linear programming problem. To summarize it briefly, the possibility to place a WPT in each discretized warehouse zone must be treated using the optimization variables. The whole charging system must be physically installable, so those optimization variables are subjected to a set of constraints, defining the mathematical rules for the correct WPT placement. Total Times previously calculated are used in the battery SoC constraint, which defines a lower bound to the value of SoC variation allowed. Note that, being the problem linear in the optimization variables, the constraints must be expressed in form of either linear equalities or inequalities. After the constraints definition, the LP problem is solved. If a solution is found, the values of the optimization variables must report the zones in which WPTs have to be installed, and in which orientation.

Chapter 4

Warehouse Modelling

The initial step of the methodology is concerned about creating a mathematical model of the warehouse. The features which must be included in this modelling phase are:

- The definition of all the zones of the warehouse in which forklifts can move, including the loading bays.
- The possibility, for a forklift, to move between couple of those zones, and the way to define the path followed.
- The definition of all the operations to be executed in the warehouse, alongside all their parameters (probability, picking point location etc.).

The warehouse is modelled as a graph, using a set of nodes and edges. Corridors and lanes where forklift can move are modelled as a set of uniformly-spaced nodes. Each node represents a point in the warehouse over which forklift can transit, while moving, or can stop, for executing an operation. Let us define *couple of adjacent nodes* as two nodes with mutual distance equal to the node spacing. Then, couples of adjacent nodes can be connected by an edge. An edge which is connecting a couple of nodes indicates the possibility, for the forklifts, to move from a node to another of that couple. So, forklifts can move from a point to another of the warehouse following a path defined by a set of nodes which are connected, in pairs, by a set of edges. The lower the spacing between the nodes, the more accurate can be warehouse definition, but the higher will be the computational time requested by the modelling and optimization procedure.

Each loading bay is not modelled by using nodes and edges like all the other warehouse zones. Its "entry", that is a point sufficiently near to it where the forklift is assumed to transit while approaching to the bay, is modelled with a node belonging to the graph. All the forklifts directed to that bay are assumed to leave the nodes coverage when transiting over that node. More details can be found in section 4.5. No dynamic WPT can be placed in loading bays, but at most one static WPT can be placed in each of them. The reason behind this way of modelling the bays resides in the fact that we are interested in modelling all the warehouse zones where the forklift can move, and where dynamic WPTs can be placed. Since is not possible to install any dynamic WPT in the bay, it is convenient to consider it outside the nodes coverage, modelling just their entry so to allow forklifts to virtually enter and exit from the bay while moving on the nodes belonging to the graph. Note also that, in the procedure defined here, static WPTs could be placed in loading bays only.

Scaffolds/racks where pallets are stored, or any other detail about the warehouse, should not be modelled. Zones in which forklift cannot move should not be modelled with nodes and edges. An example of warehouse modellization is shown in Figure 13.

Dynamic WPTs can be mounted across the nodes. However, since the WPT module length considered here has a length of 2.5 m, to mount a WPT on a single node will result in a node spacing of 2.5 m, that is too high to provide a flexible enough choice of the optimal WPT layout, as well as it would not be able, in many cases, to correctly model all the warehouse aisles/lanes. So, to reduce the node spacing, we have to consider the possibility that a single WPT should be mounted on more than one node. Note that the WPT module length must be a multiple of the node spacing. A node spacing of 0.5 m has been chosen, so that a single WPT must be installed on 5 consecutive nodes. This choice is seemed a good compromise between flexibility of WPT placement and graph complexity. An example of modelled warehouse with some dynamic WPTs placed is shown in Figure 14.

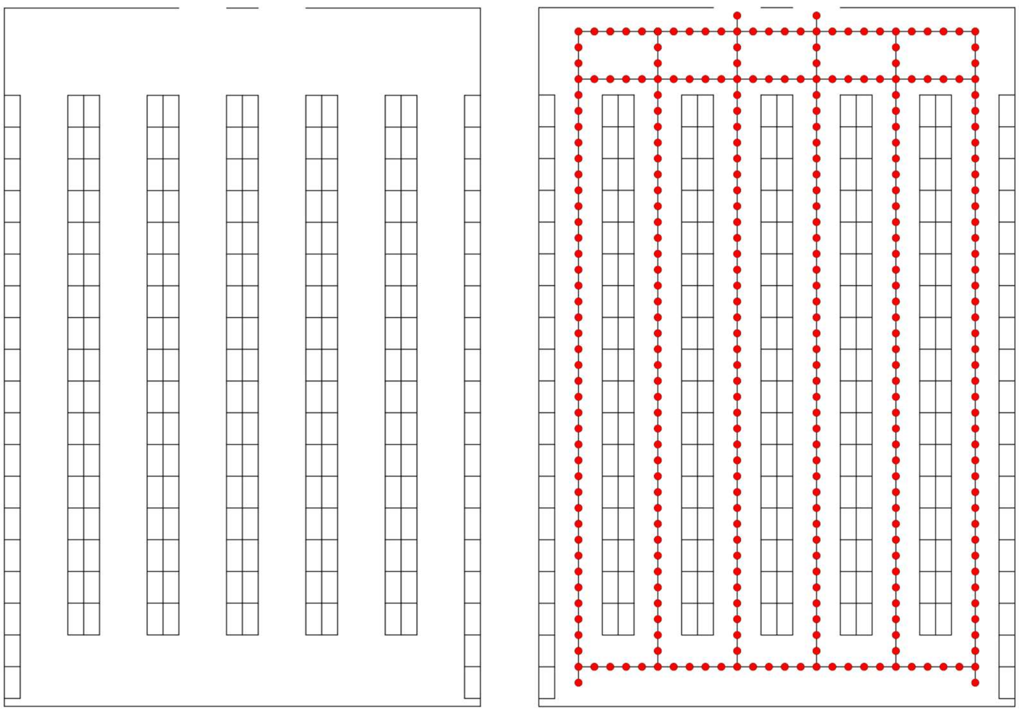

Figure 13: First example of warehouse modellization, with warehouse map (left), and its modellization with nodes and edges (right).

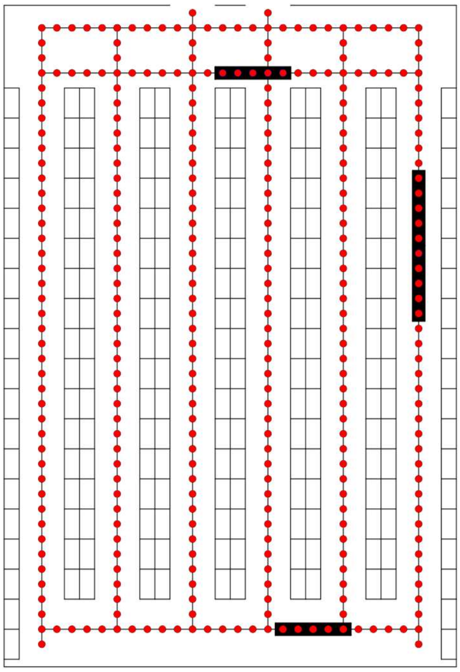

Figure 14: Example of allowed WPT placement over the nodes of the modelled warehouse. Black rectangles indicates the position of the WPTs, mounted on strips of 5 nodes.

In this initial part of the work, we can generically refer to each of the five nodes of the strip over which a WPT has to be mounted as *WPT part*. A further distinction between those nodes will be provided when dealing with the LP problem, in particular in section 7.2.

After having modelled the warehouse map using nodes and edges, all the operations that could be executed in the warehouse must be defined. Section 4.6 reports the details about their definition. Finally, the paths followed by the forklifts during each operation have to be stated. That phase is called *Forklift Routing*.

4.1 Assumptions on warehouse layout.

The warehouse modelling using nodes and edges could be applied, in principle, to every kind of warehouse layout. However, in this work, the WPT modellization introduced in the optimization step made it necessary to introduce some limitations in the warehouse layouts which can be modelled:

- All the roads/lanes where forklift moves must be located on the same height level, so as not to create overlapping road layers;
- Corridors/lanes where forklifts can move can be just oriented along two Carthesian axes x and y. From now on, we will generically refer to *Corridors* to indicate

whatever lane in which a forklift can move, regardless if one-way or not. Corridors oriented along x-axis are called *Horizontal Corridors*, while the ones oriented along y-axis are called *Vertical Corridors*. This denomination is applied also to the dynamic WPTs: those of them oriented along x-axis (namely, installed in a Horizontal Corridor) are called *Horizontal WPTs*, while the ones oriented along y-axis (namely, installed in a Vertical Corridor) are called *Vertical WPTs*. See next section for the description of corridors features.

Considering what reported in section 2.3 about the most common warehouse layouts used nowadays, these limitations are reasonable, since they do not excessively restrict the range of application of this methodology.

4.2 Corridors

Corridors are defined as lanes on which forklifts can either move or stop to execute a picking operation. They are modelled as a series of nodes, connected by edges. They can be one-way or two-way. Each node must belong to at least one corridor. So, the first operation to be executed in order to define all the nodes is to classify all the corridors of the warehouse. Corridors must be labeled using positive integer numbers, so that each corridor can be univocally defined by that number. Let us call it as *Corridor ID*. The numeration of the corridors must be sequential, and two different sequences must exist, one for the Horizontal Corridors, and the other for the Vertical ones. The Corridor ID has the following properties:

- It is a positive integer number different from 0.
- It univocally represents either an Horizontal corridor or a Vertical corridor. In a modelled warehouse, there may be present both an horizontal corridor and a vertical corridor with the same ID. The distinction between them is intrinsic, since the Horizontal and Vertical corridors sets will be treated differently in all the steps of this work.
- Corridor IDs assignment must be sequential, starting from 1. As said before, two sequences will be defined, one for Horizontal, the other for Vertical corridors.
- We can refer to "Corridor i" as the corridor having ID i, specifying its orientation. For example, we can call "Horizontal Corridor i" the Horizontal corridor with ID i.

Let us call $\mathcal{H}$ the set of all the horizontal corridors in the warehouse, and H their number. Similarly, $\mathcal{V}$ is the set of all the vertical corridors, and V their number:

$$\mathcal{H} = \{1, 2, \dots, H\}$$

$$\mathcal{V} = \{1, 2, \dots, V\}$$

Figure 15 shows an example of corridors numbering in a warehouse to be modelled. Each corridor over which forklift can move has been numbered, and the ideal paths forklifts should follow have been colored. A yellow strip indicates an Horizontal Corridor, which ID is preceded by the letter "H", represented for the purpose of distinction. Conversely, a blue strip indicates a Vertical Corridor, which ID is preceded by the letter "V". Red squares indicate the intersections between a Horizontal and a Vertical corridors. Sets $\mathcal{H}$ and $\mathcal{V}$ of the warehouse in that picture are:

$$\mathcal{H} = \{1, 2, \dots, 10\}$$

$$\mathcal{V} = \{1, 2, \dots, 12\}$$

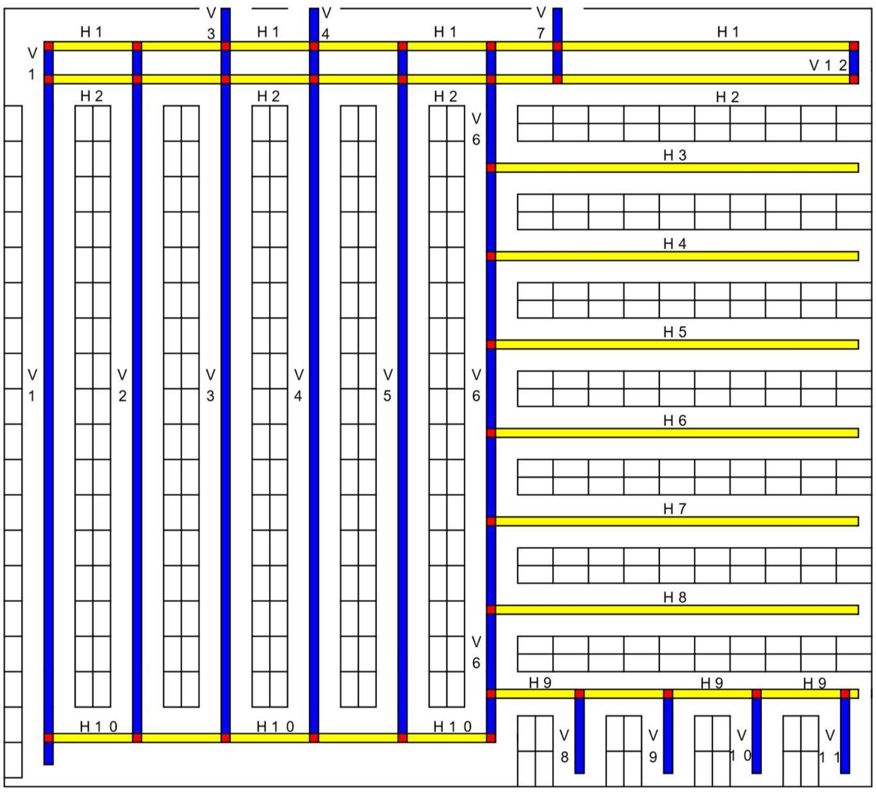

Figure 15: Example of corridors numeration.

4.3 Nodes

After having labelled corridors, we have to model them using equispaced nodes. With reference to Figure 15, since colored strips represent the ideal paths forklifts should follow, those strips will be the locations of all the nodes and edges that will be defined.

Nodes are the fundamental entities of warehouse modelling: they represent points in the space over which forklifts may transit or stop. Each node is assumed to cover a section of corridor with length equal to the node spacing, and it must be located in the middle of that section. Consequently, at each time instant, a forklift will be located in the area covered by a single node. The only exception to this rule occurs if the forklift is located in a bay since, as said before, they are not modelled in the graph.

Dynamic WPTs can be mounted across nodes (see Figure 14). If a WPT part has to be mounted over a node, we say that the node is covered by a WPT. Such WPT must be installed exactly in the area covered by the node, following the orientation of the corridor. That dynamic WPT is assumed able to recharge forklift batteries for all the time they are located in the area covered by the node.

Nodes are first defined by assigning to each of them a positive integer number different from 0, called *Node ID*. Like corridors, each node ID univocally defines a node, and we can refer to "Node i" as the node having ID i.

The parameters of each generic node are defined as follows:

- *Node ID*.
- *Position in x-direction*, defined with respect to a carthesian reference system.
- *Position in y-direction*, defined with respect to the same carthesian reference system.
- *Node Category*: it refers to the possibility to mount a WPT part across the node, and its allowed orientation. Four node categories have been defined:
    - Category 1: if an eventual WPT can be installed along x direction only. Nodes belonging to this category are called *Horizontal Nodes*. Generally, they are nodes located on horizontal corridors.
    - Category 2: if an eventual WPT can be oriented just along y direction. Nodes belonging to this category are called *Vertical Nodes*. Generally, they are nodes located on vertical corridors.
    - Category 3: if an eventual WPT can be oriented along both x or y directions. Nodes belonging to this category are called *Cross Nodes*. Generally, they are nodes located on an intersection between a horizontal and a vertical corridor.
    - Category 4: if no WPT can be installed on the node. Nodes belonging to this category are called *Impossible Nodes*. So, whatever node that, for any reason, cannot host a WPT must belong to this category.

  Note that node category is not necessarly linked to the type of corridor the node belongs to. For example it may happen that, in the node located on the intersection between two corridors, a WPT placement is allowed just along y direction. So, even if the node belongs to two different corridors, it will have category 2. An example of node categories assignment is shown in Figure 16.

- *Horizontal Corridor*: the horizontal corridor the node belongs to. If the node does not belong to any horizontal corridor, it is set to 0.
- *Vertical Corridor*: the vertical corridor the node belongs to. If the node does not belong to any vertical corridor, it is set to 0.
- *Operation Node Flag*: Binary variable indicating if an operation can be executed in the zone covered by that node (and so in the area covered by an eventual WPT part mounted on that node):
    - If 1: an operation may occur;
    - If 0: an operation cannot occur there.

Note that Node IDs must be sequential, starting from 1. By calling N the maximum number of nodes in the warehouse, we can define a set $\mathcal{N}$ of all the nodes:

$$\mathcal{N} = \{1, 2, \dots, N\}$$

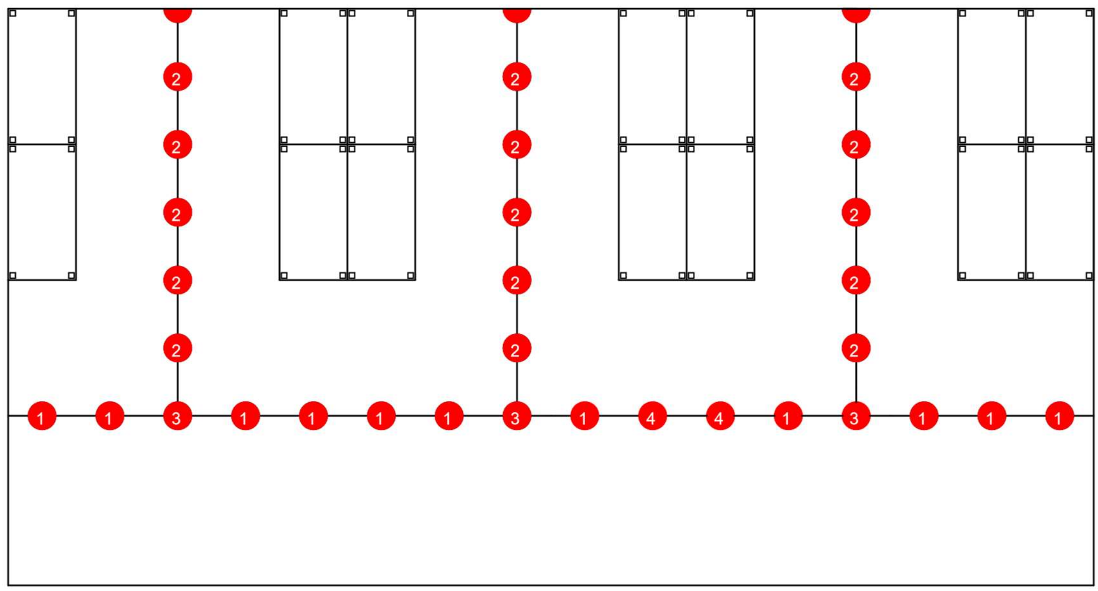

Figure 16: Example of node categories assignment in a detail of a modelled warehouse. Under the two nodes with category 4, the presence of pipes under the floor block WPT placement.

So, each node $i \in \mathcal{N}$ is defined by the following parameters:

- Node ID;
- Position in x direction;
- Position in y direction;
- cat_i – Node Category. $cat_i \in \{1, 2, 3, 4\}$;
- $horc_i$ – Horizontal corridor. $horc_i \in \mathcal{H} \cup \{0\}$;
- $verc_i$ – Vertical corridor. $verc_i \in \mathcal{V} \cup \{0\}$;
- onf_i – Operation node flag. $onf_i \in \{0,1\}$.

4.4 Edges

After having defined all the nodes, we have to properly connect them with edges in order to create a graph. An edge is used to connect a couple of adjacent nodes. Recall that, in the modelled warehouse, forklift may move from a node to another just if there is an edge which is connecting the couple. So, the last step to complete the warehouse map modelling is to define an edge for each couple of adjacent nodes belonging to the same corridor.

Like a node, an edge is defined by its *Edge ID*, that is a positive integer number different from 0 which univocally defines it. We can refer to "Edge i" as the edge having ID i.

By calling E the number of edges in the modelled warehouse, we can define a set $\mathcal{E}$ of all the edges:

$$\mathcal{E} = \{1, 2, \dots, E\}$$

Each edge $i \in \mathcal{E}$ is defined by the following parameters:

- *Edge ID*.
- $n_{1,i}, n_{2,i}$ – The two node IDs which are connected by edge i. Note that $n_{1,i} \in \mathcal{N}, n_{2,i} \in \mathcal{N}$.

In this model, edges are not oriented: if the couple $n_{1,i}$ and $n_{2,i}$ is connected by edge i, forklift movement is allowed either from $n_{1,i}$ to $n_{2,i}$ or vice versa. Conventionally, $n_{1,i}$ indicates the node with lowest ID among the two. Note that the choice not to allow edges orientation leads to the need of a way to handle the possible presence of one-way corridors. This subject will be deepened in section 4.7, which is about Forklift Routing.

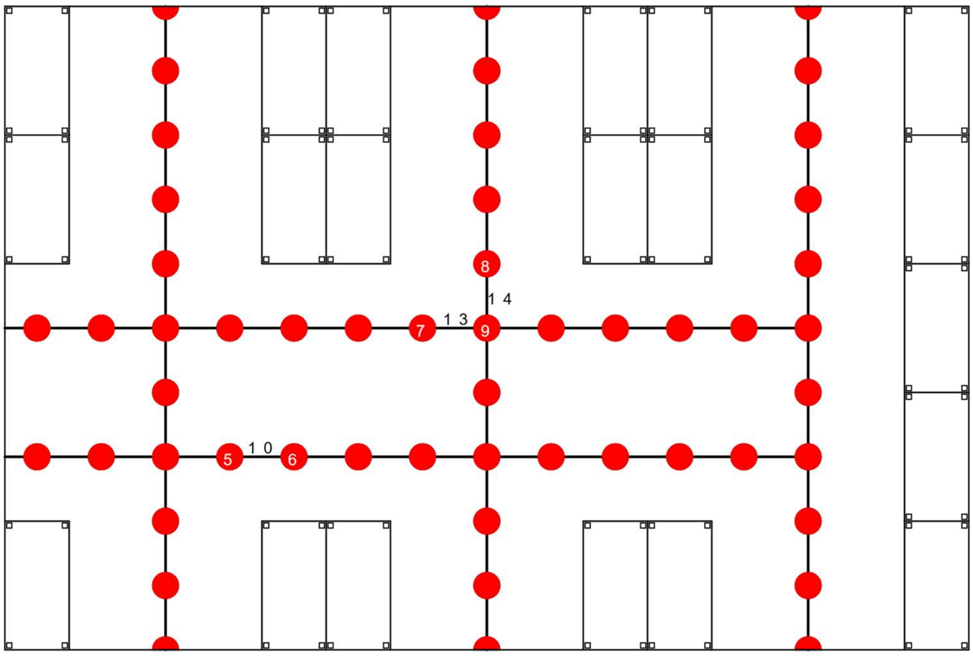

Figure 17: Example of edge modelling. Black and white numbers indicate, respectively, some Edge IDs and Node IDs.

Edge ID i	$n_{1,i}$	$n_{2,i}$
10	5	6
13	7	9
14	8	9

Table 4: Parameters of the edges defined in Figure 17.

It is worth to highlight that, like corridors, edges can be oriented along the two Carthesian directions only. Moreover, since edges are used to connect couple of adjacent nodes, their length must be equal to the node spacing. These two facts makes impossible to model any kind of curve. Actually, there could be the possibility to connect two nodes in different corridors using an edge, so to model a very primitive curve with approximate curvature radius equal to a multiple of the node spacing. However, in order to respect the two statements reported above, that situation must be avoided, and then curve modelling is not provided. In Figure 18, we can find an example of an incorrect edge modelling: blue curves, which represent the ideal trajectory of the forklift when moving from the horizontal corridor to the middle vertical one or vice versa, are modelled by defining two edges connecting node 1 with, respectively, nodes 2 and 3. Those edges have length greater than node spacing, and then this solution must be avoided. Moreover, to model the curves as simple “corners”, which elongate the path the forklift actually takes, seems reasonable since, as said at the beginning, forklift move with constant speed v_f, and neglects the typical decelerations and subsequent accelerations which occur while a curve is traveled.

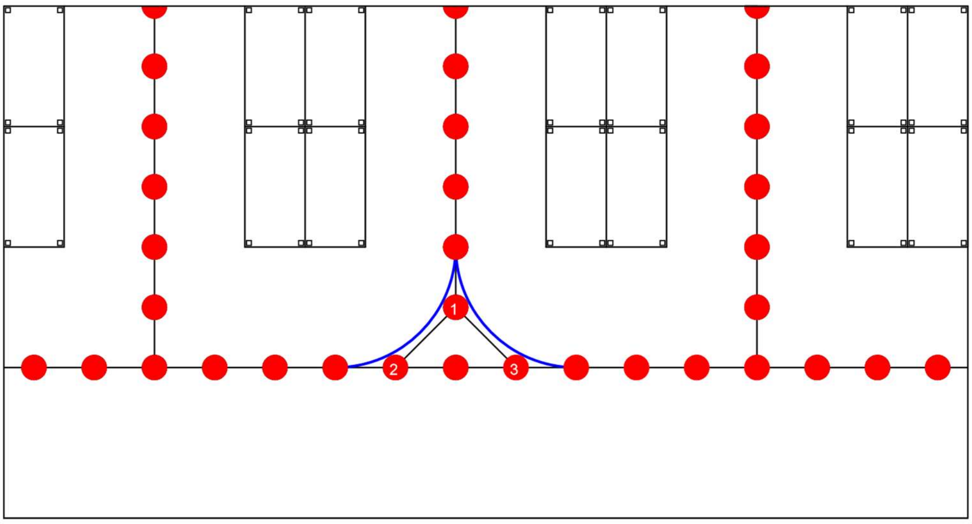

Figure 18: Example of incorrect edges modelling. Diagonal edges connecting node 1 to, respectively, nodes 2 and 3 are, wrongly, defined so to model blue curves. Left and right vertical aisles are modelled correctly.

4.5 Bays

Bays are the areas where forklifts load items on trucks, or withdraw objects from them. As said before, they are not modelled using nodes and edges. Just bay entry is modelled as part of the graph. It is represented by a node called *Bay Node*, which has the same features than any other node. Any forklift that wants to reach a bay must end its path on that bay node. Similarly, a forklift which leaves that bay to reach a desired picking point, begins its path from that bay node.

Even if a bay cannot host any dynamic WPT, there exists the possibility to install at most one static WPT on each of them. We can call *Bay WPTs* the static WPTs mounted in bays. If a Bay WPT is present, a forklift may be charged during part of the time it spends there. For safety purposes, so to avoid possible interactions of the magnetic field with the personnel working in the bay, it has been decided that such static WPT cannot be installed in the proper loading/unloading zone (namely, near the truck, where the forklift performs the loading/unloading of it), but they can be installed near it, in a dedicated area. Forklifts may spend dead times on that static WPT, while waiting between operations, during breaks, or when they are not operating. Note that static WPTs are not a-priori placed in the bays. Optimization algorithm will evaluate if it could be convenient to install a WPT on some of them, following the same principle valid for WPTs to be installed over nodes. Furthermore, in the event that static WPTs cannot be mounted in some bays, whatever the reasons, there is the possibility of preventing their positioning. Moreover, recall that bays are the only place in which it has been assumed the static WPTs to be positioned.

Each bay is univocally defined by a positive integer number called *Bay ID*. We can refer to “Bay k” as the bay having ID k. By calling B the number of different bays in the warehouse, we can define a set $\mathcal{B}$ of all the bays:

$$\mathcal{B} = \{1, 2, \dots, B\}$$

Each bay $k \in \mathcal{B}$ is defined by the following parameters:

- *Bay ID*.
- bn_k *– Bay node*. $bn_k \in \mathcal{N}$.
- bwf_k *– Bay WPT flag*. $bwf_k \in \{0,1\}$. It indicates whether a static WPT can be mounted in bay k or not:
    - If 1, a static WPT can be mounted in that bay. Optimization algorithm must consider the possibility of placing a WPT there.
    - If 0, that bay cannot host a static WPT. So, optimization algorithm cannot consider to install a bay WPT in it.

Figure 19 shows an example of bay modelling, showing the position of bay nodes and of the eventual static WPTs which can be installed there.

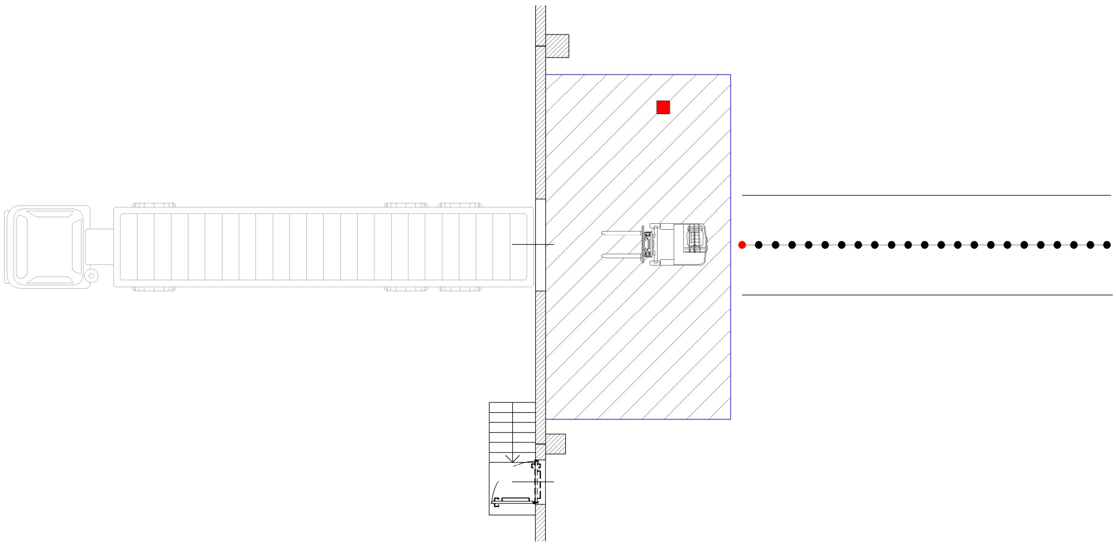

Figure 19: Example of bay modelling. Blue dashed area represents the zone in which the forklift is assumed to move while executing bay operations, and it is outside nodes coverage. Red node is the bay node. Red square is a static WPT mounted in the bay.

4.6 Operations

Operations are the key point of the methodology. An operation is defined as a set of steps to be executed in sequence by the forklift, when an object in the warehouse is required to be picked and shipped, as well as when an object in the truck has to be unloaded into the warehouse. Equivalence between picking and storage operations in this methodology has been treated in chapter 3. Each storage location must be covered by a node, over which forklift is assumed to stop for putting/retrieving the item from the racks. Each of those nodes is called *Operation Node*. Note that, if node i is an operation node, then $onf_i = 1$. More than one operation can occur on the same operation node, for example in case of items located on both the sides of a corridor. Recall that each operation has a certain probability to be executed. The sum of all the warehouse operations probabilities must be unitary.

Consider a picking operation: the steps necessary to execute it are as follows:

- At the beginning, forklift is assumed to be located in a bay.
- Forklift moves from the bay to the position of the item to be picked, following a determined path.
- Once reached item position, it stays under the operation node coverage for all the time required to execute the picking operation, including the time necessary for the maneuvers. This overall amount of time is called *Operation Time*.
- After having picked the object, forklift goes back to the loading bay, assumed the same where it has started the operation, following a certain path (which can be different than the one to go from bay to operation node).
- Once arrived in the bay, forklift stays there for the time necessary to load the item on the truck. This time interval is called *Bay Time*.
- Operation ends after the item has been loaded on the truck. It may happen that forklift may stay still in the bay after having ended the operation, because another one may not be immediately available. In that time interval, which must be included in the Bay Time, forklift could be charged by an eventual static WPT mounted in the bay.

The path of an operation is the set of nodes crossed in sequence by the forklift to go from bay to operation node and vice versa. We can distinguish between the two: path from bay to operation node is called *Outward Path*, while path from operation node to bay is called *Return Path*. The path a forklift follow to execute a certain operation is assumed to be fixed. Since we are dealing with a graph modelling of the warehouse, paths for each operations could be found automatically using some existing algorithm. In this work, Dijkstra Algorithm (refer to section 4.8 for the details) has been used for this purpose. By default, exploiting such algorithm properties, forklifts should follow the shortest path (optimal in terms of distance and, due to the forklift constant speed assumption, of time). It may happen that, due to one-way corridors presence (not modelled in the graph) or because of the warehouse picking policy, forklift should follow a path different from the shortest one. In that case, to make the path selection as automatic as possible, a set of intermediate nodes may be defined to make the algorithm to consider the desidired path instead of the shortest one. In this case, the selected path is the minimal one between bay node and operation node which intersects, one by one, all the specified intermediate nodes.

Each operation is characterized by the following parameters:

- *Operation ID*: a positive integer number univocally defining the operation.
- *Operation Node*: the ID of the node in which the item to be picked is assumed to be located.
- *Operation Probability*: the probability to execute the operation.
- *Operation Bay*: the ID of the bay where forklift is located when operation begins and ends.
- *Operation Time*: time necessary to execute the operation.
- *Bay Time*: the time spent by the forklift in the bay, either for loading item on truck or to wait for the next operation to start.
- *Bay Recharging Time*: the fraction of Bay Time in which the forklift remains idle in the bay, waiting for another operation.
- *Intermediate Node Flag*: binary variable used to select the algorithm used for the forklift path calculation:
    - If 1, forklift is guided through a series of intermediate nodes while going from bay to operation node and vice versa.
    - If 0, the shortest path between them is selected as forklift route.
- *Outward Trip Intermediate Nodes*: a list of eventual intermediate nodes to be used for forklift routing, used for the trip from bay to operation node (Outward Trip).
- *Return Trip Intermediate Nodes*: same list of intermediate nodes, but for the trip from operation node to bay (Return Trip).

We can refer to "Operation j" as the operation having ID j. By calling O the number of different operations which can be executed, it is possible to define a set $\mathcal{O}$ of all the operations:

$$\mathcal{O} = \{1, 2, \dots, O\}$$

Then, each operation $j \in \mathcal{O}$ is defined by:

- Operation ID;
- on_j − Operation Node. $on_j \in \mathcal{N}$;
- p_j − Operation Probability. $p_j \in [0,1]$;
- b_j − Operation Bay. $b_j \in \mathcal{B}$;

- ot_j − Operation Time [s];
- bt_j − Bay Time [s];
- brt_j − Bay Recharging Time. $brt_j \in [0,1]$;
- inf_j − Intermediate Node Flag. $inf_j \in \{0,1\}$;
- $\mathcal{OT}_j$ − Set of the Outward Trip intermediate nodes. $\mathcal{OT}_j \subseteq \mathcal{N}$;
- $\mathcal{RT}_j$ − Set of the Return Trip intermediate nodes. $\mathcal{RT}_j \subseteq \mathcal{N}$.

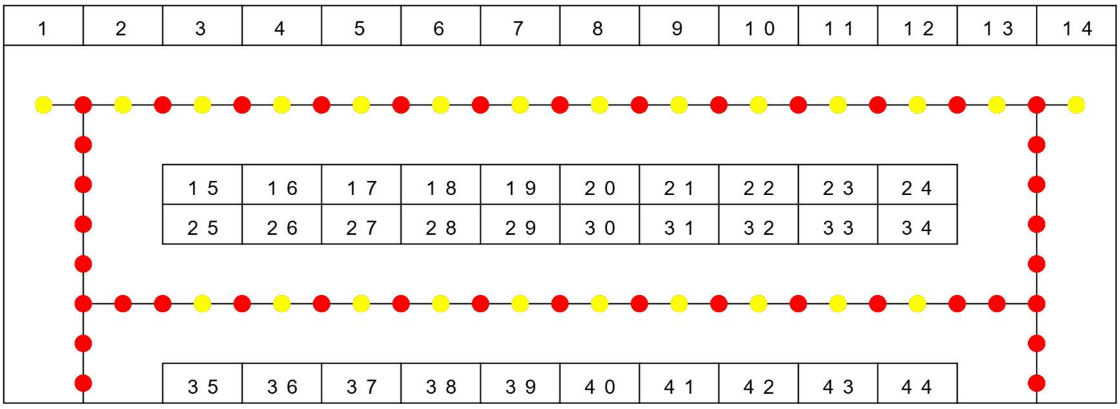

Figure 20: Example of warehouse modelling. Numbers on scaffolds indicate Operations ID. Red nodes indicate generic nodes, while the yellow ones are the operation nodes.

4.7 Forklift Routing

Forklift Routing is the procedure aiming at finding the paths, namely the set of nodes crossed by the forklift in each operation. For each operation j, we can define two sets $\mathcal{OP}_j$ and $\mathcal{RP}_j$:

- $\mathcal{OP}_j = \{\text{nodes crossed during Outward path of operation } j\}$, $\mathcal{OP}_j \subseteq \mathcal{N}$.

- $\mathcal{RP}_j = \{\text{nodes crossed during Return Path of operation } j\}$, $\mathcal{RP}_j \subseteq \mathcal{N}$.

Note that both the paths include either bay node and operation node, since they are proper crossed nodes too.

In "small" warehouses, with a low number of nodes and operations to be executed, those paths might be defined by hand. However, in order to let this procedure to be feasible also for bigger warehouses, an algorithm can be used to make it as automatic as possible. Dijkstra Algorithm is used to select the minimum path between any couple of nodes. In this work, the minimum (or shortest) path between a couple of nodes is defined as the smallest set of nodes to be crossed for moving from a node to another of the couple. However, there is the need to define a procedure which allows to select a path different from the minimum one, because of the picking policies to be used or the presence of one-way corridors, not modelled in the graph.

To define the paths of an operation, two options are available:

- Shortest path from bay to operation node and vice versa is searched, through Dijkstra Algorithm. In case more than one path with minimum length is present, the first one which has been found is used. However, even if optimal in terms of distance and of time, this path may not be feasible, that is it cannot be actually traveled by the forklift because of the presence of one-way corridors crossed in the prohibited sense. Moreover, the warehouse routing policy may want forklifts to follow determined routes, different from the optimal ones. In all those cases, second option shall be used.
- A set of intermediate nodes, one for the Outward trip, another for the Return trip, are defined. Those nodes must be crossed in sequence during the trips. Dijkstra Algorithm is then used between couple of nodes to be crossed, so to minimize the effort in defining all the nodes belonging to the paths. Those nodes must be manually listed. An accurate definition of them let to select the desired path which follows all the limitations previously listed.

The option selection can be done for each operation $j \in \mathcal{O}$, by selecting the appropriate value of inf_j.

Figure 21 shows an example of minimum path. Black nodes represent the path from node 24 to node 46, or vice versa. In this case, the path between that couple is the set $\{24, 25, 26, 46, 47, 48, 49, 57, 58, 59, 60, 61\}$. Note how it must include also starting and destination nodes. On the other side, Figure 22 shows an example of intermediate nodes definition. Forklift must move from node 1 to node 138 (Outward trip), and then back to node 1 (Return trip). Corridors one-ways are indicated by the arrows. Warehouse picking policy is assumed to force the forklifts to entirely cross each aisle once, even if without objects to be picked in them. In this case, some intermediate nodes, which are colored in yellow, shall be defined. Supposing this operation has ID equal to 9, then the set $\mathcal{OT}_9$ will be $\{29, 34, 8, 13, 39, 44, 18, 23, 49, 54\}$, while $\mathcal{RT}_9$ is $\{28\}$. Note that the return path coincides with the minimum one and is feasible, so there would not be the need to define a set of intermediate nodes for that path. However, the procedure of intermediate nodes definition requires to define both the sets.

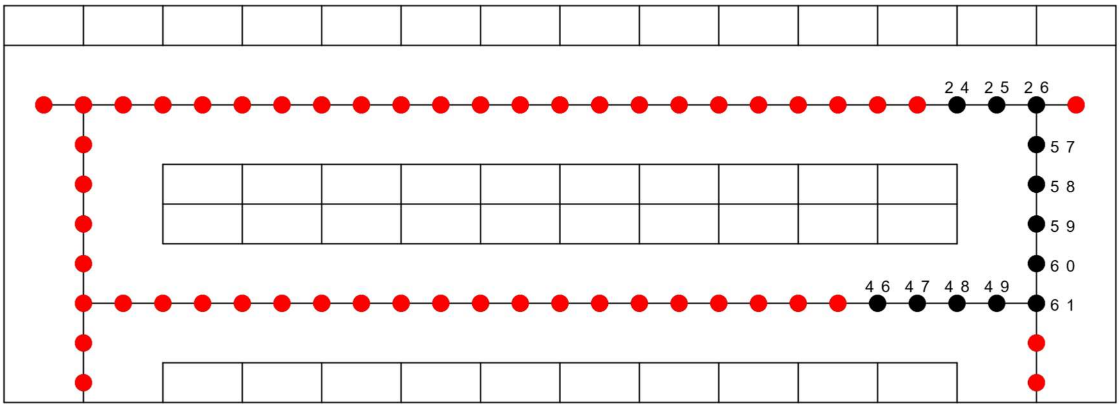

Figure 21: Example of minimum path calculation: black nodes represent the path from node 24 to node 46, or vice versa.

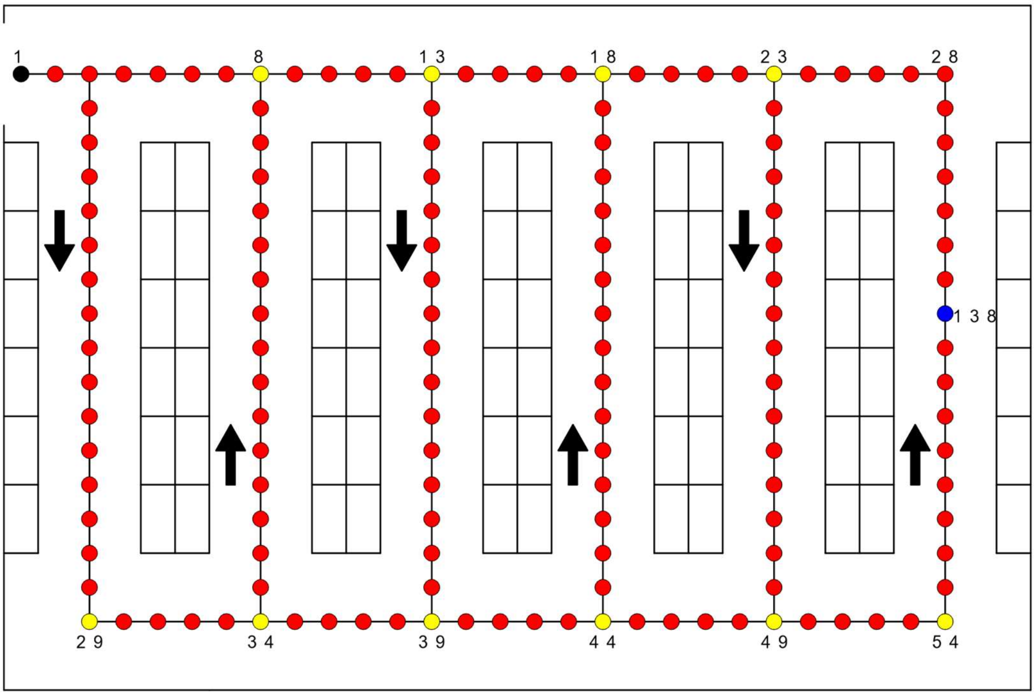

Figure 22: Example of intermediate nodes definition. Arrows indicate the one-way sense of the corridors. Yellow nodes are the intermediate nodes to move from node 1 to node 138, by entirely crossing all the previous aisles.

At that point, all the warehouse modelling elements has been introduced. Total Times calculation, which is based on the warehouse model that has just been defined, is the next big step of this methodology, and is reported in the next chapter.

4.8 Dijkstra Algorithm

Dijkstra Algorithm was developed by the Dutch scientist Edsger Dijkstra in 1956, which published it in 1959 [42] [43]. It solves the single-source shortest-paths problem on a weighted, directed graph, in the case of non-negative edge weights [31]. Such problem can be defined as follows: given a graph $G = (V, E)$, where V is the set of all the vertices (namely, the nodes), and E the set of all the edges, we want to find the minimum-weight path from a given source vertex $s \in V$ to each vertex $v \in V$ [31]. The complexity of this algorithm has been proved to be $O(|E| + |V| \log|V|)$ [44].

By defining as edge weight the node spacing, which represents the proper edge length, we can use Dijkstra Algorithm to find the minimum path between bay nodes and operation nodes by restricting its search to the minimum-lenght path from a given source vertex (the bay node) to a certain destination vertex (the operation node), and vice versa. However, since as said in section 4.4, the graph defined in this work do not allow edge orientation, the slightly modified version of Dijkstra algorithm which has been used in this work must not consider it, leading to the need of defining the sets of intermediate nodes to "guide" the minimum-length path search considering the eventual one-way of corridors. The possibility to implement edge orientation, which would simplify the modelling step as it would eliminate the need to define intermediate nodes, is left to future improvements.

The detailed structure of Dijkstra algorithm can be found in [31], while some intuitive examples about the way it works are reported in [45].

Chapter 5

Total Times Calculation

Total Times represent the main input to the optimization process. Considering just a single forklift per time that is operating in the warehouse, let us define *Total Time* of a certain node/bay the probability, at any time instant, to find that forklift in the node/bay. Pay attention that, even if we are improperly calling “Time” what actually is a probability, so a dimensionless quantity, those probabilities will be later used in relation to certain time intervals, so to become proper time quantities. More details will be provided in the next chapters.

We can proceed now with the Total Times Vectors definition, which are arrays of Total Times of either each node or bay, and are reported in section 5.1. Their calculation procedure is described in sections 5.2 and 5.3.

5.1 Total Times Vectors definition

Before defining the Total Times Vectors, let us make a consideration on the operation steps, which have been explained in details in section 4.6. From now on, we can divide an operation in four phases, according to the position of the forklift and to what is executing. In any instant of the operation, the forklift must be in one of the phases reported in Table 5:

Operation Phase	Description
Node Movement	Forklift is moving over a node belonging to the operation path. It is not executing one of the steps typical of the operation time parameter.
Node Operation	Forklift is on the operation node, and it is executing one of the movements typical of the operation time parameter (maneuvers, lifting/lowering forks, withdrawing item from the scaffolds etc.).

Bay Operation	Forklift is in the bay, and it is loading/unloading the items from the truck.
Bay Recharge	Forklift is idle in the bay, waiting for the assignment of further operations. During this phase only it could be charged by an eventual static WPT placed in the bay. If the WPT is not present, the forklift is simply still.

Table 5: Operation phases, according to the forklift actions.

In relation to the operation phases just described, it is possible to define six different Total Times Vectors, called:

- $TNTV$ – Total Node Times Vector;
- $TBTV$ – Total Bay Times Vector;
- $TNMTV$ – Total Node Movement Times Vector;
- $TNOTV$ – Total Node Operation Times Vector;
- $TBOTV$ – Total Bay Operation Times Vector;
- $TBRTV$ – Total Bay Recharge Times Vector.

The first two contain the probability, at any time instant, to find the forklift, respectively, over each node and over each bay, regardless of the operation phase they are executing. Conversely the other four indicate the probability to find it over each node/bay while in the corresponding operation phase. The reason behind the definition of all these Total Times is that forklift power consumption changes according to the operation phase. All the details about it can be found in chapter 6.

Let us now proceed to analyze each of those vectors:

- $\boldsymbol{TNTV}$:

$$TNTV = \begin{bmatrix} tnt_1 \\ tnt_2 \\ \vdots \\ tnt_i \\ \vdots \\ tnt_N \end{bmatrix} \quad i \in \mathcal{N}$$

tnt_i is called *Total Node Time* of node i. It is the probability, at any time instant, to find the forklift located on node i, regardless of the operation phase.

- $\boldsymbol{TNMTV}$:

$$TNMTV = \begin{bmatrix} tnmt_1 \\ tnmt_2 \\ \vdots \\ tnmt_i \\ \vdots \\ tnmt_N \end{bmatrix} \quad i \in \mathcal{N}$$

$tnmt_i$ is called *Total Node Movement Time* of node i. It is the probability, at any time instant, to find the forklift on node i, while in the Node Movement phase.

- $\boldsymbol{TNOTV}$:

$$TNOTV = \begin{bmatrix} tnot_1 \\ tnot_2 \\ \vdots \\ tnot_i \\ \vdots \\ tnot_N \end{bmatrix} \quad i \in \mathcal{N}$$

$tnot_i$ is called *Total Node Operation Time* of node i. It is the probability, at any time instant, to find the forklift that on node i, while in the Node Operation phase.

The following relation holds:

$$TNTV = TNMTV + TNOTV \tag{1}$$

Note also that, if a node is not an operation node, we have that:

$$tnot_i = 0 \;\Rightarrow tnt_i = tnmt_i \,, \forall i \in \mathcal{N} \,:\; onf_i = 0$$

- $\boldsymbol{TBTV}$:

$$TBTV = \begin{bmatrix} tbt_1 \\ tbt_2 \\ \vdots \\ tbt_k \\ \vdots \\ tbt_B \end{bmatrix} \quad k \in \mathcal{B}$$

tbt_i is called *Total Bay Time* of bay k. It is the probability, at any time instant, to find the forklift located in bay k, regardless of the operation phase.

- $\boldsymbol{TBOTV}$:

$$TBOTV = \begin{bmatrix} tbot_1 \\ tbot_2 \\ \vdots \\ tbot_k \\ \vdots \\ tbot_B \end{bmatrix} \quad k \in \mathcal{B}$$

$tbot_i$ is called *Total Bay Operation Time* of bay k. It is the probability, at any time instant, to find the forklift in bay k while in the Bay Operation phase.

- $\boldsymbol{TBRTV}$:

$$TBRTV = \begin{bmatrix} tbrt_1 \\ tbrt_2 \\ \vdots \\ tbrt_k \\ \vdots \\ tbrt_B \end{bmatrix} \quad k \in \mathcal{B}$$

$tbrt_i$ is called *Total Bay Recharge Time* of bay k. It is the probability, at any time instant, to find the forklift in bay k while in the Bay Recharge phase.

The following relation holds:

$$TBTV = TBOTV + TBRTV \tag{2}$$

Furthermore, the sum of all the elements of TNTV and TBTV must be unitary, since it represents the sum of all the probabilities to find the forklift on each node or bay:

$$\sum_{i=1}^{N} tnt_i + \sum_{k=1}^{B} tbt_k = 1 \tag{3}$$

5.2 Partial Times Matrices

As said before, Total Times calculation is based on the analysis of all the operations to be executed in the warehouse, in a probabilistic framework. Before going into the proper details of their calculation, we have to define the Partial Times.

5.2.1 Partial Times Matrices definition

Partial Times are the basis for the Total Times computation. We can generically define *Partial Time* of a node/bay in a determined operation as the time interval spent by the forklift on that node/bay during that operation. So, for each operation $j \in \mathcal{O}$, there exist a different Partial Time for each node/bay. Moreover, unlike the Total Times, Partial Times are independent on the probability of executing operations, and they represent proper time quantities.

As it was done for the Total Times, it is possible to define six different Partial Times Matrices in relation to the operation phases listed in Table 5, which are called:

- $PNTM$ – Partial Node Times Matrix;
- $PBTM$ – Partial Bay Times Matrix;
- $PNMTM$ – Partial Node Movement Times Matrix;
- $PNOTM$ – Partial Node Operation Times Matrix;
- $PBOTM$ – Partial Bay Operation Times Matrix;
- $PBRTM$ – Partial Bay Recharge Times Matrix.

The choice of using matrices instead arrays resides in the fact that each Partial Time is defined not only for a certain node as in the Total Times, but also for a certain operation. So, each element of the matrices will be referred to both of them.

Let us analyze the matrices more in details:

- $\boldsymbol{PNTM}$:

$$PNTM = \begin{bmatrix} pnt_{11} & pnt_{12} & \cdots & pnt_{1j} & \cdots & pnt_{1O} \\ pnt_{21} & pnt_{22} & \cdots & pnt_{2j} & \cdots & pnt_{2O} \\ \vdots & \vdots & & \vdots & & \vdots \\ pnt_{i1} & pnt_{i2} & \cdots & pnt_{ij} & \cdots & pnt_{iO} \\ \vdots & \vdots & & \vdots & & \vdots \\ pnt_{N1} & pnt_{N2} & \cdots & pnt_{Nj} & \cdots & pnt_{NO} \end{bmatrix} \text{ with } \begin{cases} i \in \mathcal{N} \\ j \in \mathcal{O} \\ [pnt_{ij}] = s \end{cases}$$

pnt_{ij} is called *Partial Node Time* of node i during operation j. It represents the time spent by the forklift on node i during operation j, regardless of the operation phase. For example, if $pnt_{ij} = 4$ s, it means that forklift spends 4 seconds over node i, during operation j.

- $\boldsymbol{PNMTM}$:

$$PNMTM = \begin{bmatrix} pnmt_{11} & pnmt_{12} & \cdots & pnmt_{1j} & \cdots & pnmt_{1O} \\ pnmt_{21} & pnmt_{22} & \cdots & pnmt_{2j} & \cdots & pnmt_{2O} \\ \vdots & \vdots & & \vdots & & \vdots \\ pnmt_{i1} & pnmt_{i2} & \cdots & pnmt_{ij} & \cdots & pnmt_{iO} \\ \vdots & \vdots & & \vdots & & \vdots \\ pnmt_{N1} & pnmt_{N2} & \cdots & pnmt_{Nj} & \cdots & pnmt_{NO} \end{bmatrix} \text{ with } \begin{cases} i \in \mathcal{N} \\ j \in \mathcal{O} \\ [pnmt_{ij}] = s \end{cases}$$

$pnmt_{ij}$ is called *Partial Node Movement Time* of node i during operation j. It represents the time spent by the forklift on node i during operation j while in the Node Movement phase.

- $\boldsymbol{PNOTM}$:

$$PNOTM = \begin{bmatrix} pnot_{11} & pnot_{12} & \cdots & pnot_{1j} & \cdots & pnot_{1O} \\ pnot_{21} & pnot_{22} & \cdots & pnot_{2j} & \cdots & pnot_{2O} \\ \vdots & \vdots & & \vdots & & \vdots \\ pnot_{i1} & pnot_{i2} & \cdots & pnot_{ij} & \cdots & pnot_{iO} \\ \vdots & \vdots & & \vdots & & \vdots \\ pnot_{N1} & pnot_{N2} & \cdots & pnot_{Nj} & \cdots & pnot_{NO} \end{bmatrix} \text{ with } \begin{cases} i \in \mathcal{N} \\ j \in \mathcal{O} \\ [pnot_{ij}] = s \end{cases}$$

$pnot_{ij}$ is called *Partial Node Operation Time* of node i during operation j. It represents the time spent by the forklift on node i during operation j while in the Node Operation phase.

The following relation holds:

$$PNTM = PNMTM + PNOTM \tag{4}$$

Note that, in every operation $j \in \mathcal{O}$, the only node over which the forklift can be in Node Operation phase is specified by on_j. Therefore:

$$pnot_{ij} = 0 \ \Rightarrow pnt_{ij} = pnmt_{ij}\,, \forall i \in \mathcal{N}, \forall j \in \mathcal{O} :\ i \neq on_j$$

- $\boldsymbol{PBTM}$:

$$PBTM = \begin{bmatrix} pbt_{11} & pbt_{12} & \cdots & pbt_{1j} & \cdots & pbt_{1O} \\ pbt_{21} & pbt_{22} & \cdots & pbt_{2j} & \cdots & pbt_{2O} \\ \vdots & \vdots & & \vdots & & \vdots \\ pbt_{k1} & pbt_{k2} & \cdots & pbt_{kj} & \cdots & pbt_{kO} \\ \vdots & \vdots & & \vdots & & \vdots \\ pbt_{B1} & pbt_{B2} & \cdots & pbt_{Bj} & \cdots & pbt_{BO} \end{bmatrix} \text{ with } \begin{cases} k \in \mathcal{B} \\ j \in \mathcal{O} \end{cases}$$

pbt_{kj} is called *Partial Bay Time* of bay k during operation j. It represents the time spent by the forklift in bay k during operation j, regardless of the operation phase.

- $\boldsymbol{PBOTM}$:

$$PBOTM = \begin{bmatrix} pbot_{11} & pbot_{12} & \cdots & pbot_{1j} & \cdots & pbot_{1O} \\ pbot_{21} & pbot_{22} & \cdots & pbot_{2j} & \cdots & pbot_{2O} \\ \vdots & \vdots & & \vdots & & \vdots \\ pbot_{k1} & pbot_{k2} & \cdots & pbot_{kj} & \cdots & pbot_{kO} \\ \vdots & \vdots & & \vdots & & \vdots \\ pbot_{B1} & pbot_{B2} & \cdots & pbot_{Bj} & \cdots & pbot_{BO} \end{bmatrix} \text{ with } \begin{cases} k \in \mathcal{B} \\ j \in \mathcal{O} \end{cases}$$

$pbot_{kj}$ is called *Partial Bay Operation Time* of bay k during operation j. It represents the time spent by the forklift in bay k during operation j, while in the Bay Operation phase.

- $\boldsymbol{PBRTM}$:

$$PBRTM = \begin{bmatrix} pbrt_{11} & pbrt_{12} & \cdots & pbrt_{1j} & \cdots & pbrt_{1O} \\ pbrt_{21} & pbrt_{22} & \cdots & pbrt_{2j} & \cdots & pbrt_{2O} \\ \vdots & \vdots & & \vdots & & \vdots \\ pbrt_{k1} & pbrt_{k2} & \cdots & pbrt_{kj} & \cdots & pbrt_{kO} \\ \vdots & \vdots & & \vdots & & \vdots \\ pbrt_{B1} & pbrt_{B2} & \cdots & pbrt_{Bj} & \cdots & pbrt_{BO} \end{bmatrix} \text{ with } \begin{cases} k \in \mathcal{B} \\ j \in \mathcal{O} \end{cases}$$

$pbrt_{kj}$ is called *Partial Bay Recharge Time* of bay k during operation j. It represents the time spent by the forklift in bay k during operation j, while in the Bay Recharge phase.

The following relation holds:

$$PBTM = PBOTM + PBRTM \quad (5)$$

Note that, in every operation $j \in \mathcal{O}$, the only bay that is utilized is specified by b_j. Therefore:

$$pbt_{kj} = 0 \;\; \forall k \in \mathcal{B}, \forall j \in \mathcal{O} \; : \; k \neq b_j$$

5.2.2 Partial Times Matrices calculation

Partial Times Matrices calculation are the first step for finding Total Times. They are are calculated using the following parameters:

- Node spacing n_s;
- Forklift average speed v_f;
- Operation parameters;
- Outward Path and Return Path for each operation;

Initially, and just initially, all the entries of the six matrices must be set to 0. Then, the procedure for the calculation, which is listed below, must be executed for each different operation $j \in \mathcal{O}$:

1. **Calculation of Partial Node Movement Times:** All the nodes crossed by the forklift during operation j belong to $\mathcal{OP}_j$ and $\mathcal{RP}_j$. Over all of them, the forklift is assumed to move with constant speed v_f, neglecting the effects of curves, accelerations and decelerations. Knowing node spacing n_s, we can estimate an average crossing time ct as the time spent by the forklift over each node while crossing it with constant speed v_f:

$$ct = \frac{n_s}{v_f} \quad (6)$$

The forklift spends an amount of time equal to ct on each of the nodes belonging to either $\mathcal{OP}_j$ or $\mathcal{RP}_j$, while in Node Movement phase. If a node is crossed more than once, it means that forklift will stay there for multiple amounts of ct. Therefore, for each node i belonging to the paths of operation j, we can calculate the time spent there by the forklift while in Node Movement phase as:

$$pnmt_{ij} = k_{ij} \cdot ct \quad \forall i \in \mathcal{OP}_j \cup \mathcal{RP}_j \quad (7)$$

where k_{ij} represents the multiplicity of node i during operation j, that is how many times node i is crossed during that operation (how many times node i appears in either $\mathcal{OP}_j$ and $\mathcal{RP}_j$).

2. **Calculation of Partial Node Operation Time:** in operation j, forklift spends a time equal to ot_j on the Operation Node on_j. In this time interval, the forklift is in Node Operation phase. Consequently:

$$pnot_{on_j,j} = ot_j \tag{8}$$

3. **Calculation of Partial Bay Time:** in operation j, forklift spends some time in bay b_j. That time is represented by bt_j. During it, forklift can be in both Bay Operation or Bay Recharge phases, so bt_j should be treated as a Partial Bay Time:

$$pbt_{b_j,j} = bt_j \tag{9}$$

4. **Calculation of Partial Bay Recharging Time:** it is known that, during operation j, forklift may spend a fraction of bay time in Bay Recharge phase. Such fraction is given by parameter brt_j. The proper time interval forklift is in that phase is obtained as $brt_j \cdot bt_j$, and it represents the Partial Bay Recharging Time of operation j:

$$pbrt_{b_j,j} = brt_j \cdot bt_j \tag{10}$$

5. **Computation of Partial Node Time:** In points 1 and 2 we have calculated, respectively, $pnmt_{ij}$ and $pnot_{ij}$, $\forall i \in \mathcal{N}$. Partial Node Time for operation j can be obtained considering equation (4), which can be rewritten as:

$$pnt_{ij} = pnmt_{ij} + pnot_{ij} \quad i \in \mathcal{N} \tag{11}$$

6. **Computation of Partial Bay Operation Time:** In points 3 and 4 we have calculated, respectively, $pbt_{k,j}$ and $pbrt_{k,j}$, $\forall k \in \mathcal{B}$. Partial Bay Operation Time for operation j can be obtained by manipulating equation (5), obtaining:

$$PBOTM = PBTM - PBRTM \tag{12}$$

that, in details, can be written as:

$$pbot_{kj} = pbt_{kj} - pbrt_{kj} \quad k \in \mathcal{B} \tag{13}$$

Repeating the steps from 1 to 6 for each operation $j \in \mathcal{O}$, the complete Partial Times Matrices can be built.

5.3 Total Times Vectors calculation

Total Times Vectors are calculated starting from the informations on the overall operations set contained in the Partial Times Matrices. First, for each operation $j \in \mathcal{O}$, compute its *Overall Operation Time* oot_j, defined as the duration of the whole operation j. Mathematically, it is the sum of all the Partial Node Times and the Partial Bay Times of that operation:

$$oot_j = \sum_{i=1}^{N} pnt_{ij} + \sum_{k=1}^{B} pbt_{kj} \tag{14}$$

To explain the procedure of Total Times Calculation, we can consider Total Node Times and Total Bay Times as case study, describing their computation in a detailed manner. Then, we can extend the procedure to all the other vectors, being the procedure similar.

- **Total Node Times**

  Considering a generic node $i \in \mathcal{N}$, its Total Node Time tnt_i can be calculated as:

$$tnt_i = \sum\nolimits_{j=1}^{O} \left(\frac{pnt_{ij}}{oot_j} \cdot p_j \right) \tag{15}$$

  The meaning of each term is as follows:

  - $\frac{pnt_{ij}}{oot_j} \in [0,1]$ represents, during operation j, the “weight” of the time spent over node i, that is pnt_{ij}, on the overall operation time oot_j. This ratio represents, during operation j only, the probability, at a randomly chosen time instant, to find the forklift located on node i. The higher the time spent on this node with respect to the whole operation duration, the higher this probability.
    Example: if during operation j we have that, in a certain node i, it results that $\frac{pnt_{ij}}{oot_j} = 0.1$, it means that, during such operation only, we have a 10%

probability, at a randomly chosen time instant, to find the forklift located over that node i.

- $\frac{pnt_{ij}}{oot_j} \cdot p_j \in \left[0, \frac{pnt_{ij}}{oot_j}\right]$ considers not only the weight of the time spent over node i with respect to the whole duration of operation j, but also the weight of the probability to execute operation j with respect to the whole set of operations O. The higher the probability p_j to execute such operation, the bigger the relevance of the time spent over node i during such operation with respect to the time spent on it during all the other operations.
  Example: considering the example presented before, with $\frac{pnt_{ij}}{oot_j} = 0.1$, consider now that probability to execute operation j is $p_j = 0.1$. This leads to:

  $$\frac{pnt_{ij}}{oot_j} \cdot p_j = 0.1 \cdot 0.1 = 0.01$$

  The meaning of this result is that, at any time instant, there is a probability of 1% to find a forklift located over node i while executing operation j.

- $\sum_{j=1}^{O}\left(\frac{pnt_{ij}}{oot_j} \cdot p_j\right)$: In order to end up with the proper definition of total time, that should not consider which operation $j \in O$ forklift is executing, we need to sum the contributions of all of them.
  Ex: if we obtain that, for node i:

  $$tnt_i = \sum_{j=1}^{O}\left(\frac{pnt_{ij}}{oot_j} \cdot p_j\right) = 0.05$$

  This means that, at any time, there is a 5% probability to find the forklift over node i. This is exactly what we were searching for: this probability isindependent on the operation forklift is executing. It just indicates that a forklift is present over that node. The higher is tnt_i, the higher is the time,statistically, spent by the forklift over it with reference to a time interval like a working shift. So, recall that for optimization purposes it is more convenient to place WPTs in zones with higher Total Times.

  Note that $tnt_i = 0$ occurs if and only if node i is never crossed during any operation $j \in O$.

- **Total Bay Times**

  Considering each bay $k \in \mathcal{B}$, its Total Bay Time tbt_k is:

  $$tbt_k = \sum_{j=1}^{O} \left(\frac{pbt_{kj}}{oot_j} \cdot p_j \right) \tag{16}$$

  Similarly to the Total Node Times, the meaning of each term is as follows:

  - $\frac{pbt_{kj}}{oot_j} \in [0,1]$: it represents, during operation j, the "weight" of the time spent in bay k on the overall operation time oot_j. This ratio represents, during operation j only, the probability, at a randomly chosen time instant, to find the forklift located in bay k. The higher the time spent here with respect to the whole operation duration, the higher this probability.

  - $\frac{pbt_{kj}}{oot_j} \cdot p_j \in \left[0, \frac{pbt_{kj}}{oot_j}\right]$: it considers not only the weight of the time spent in bay k with respect to the whole duration of operation j, but also the weight of the probability to execute operation j with respect to the whole set of operations $\mathcal{O}$. The higher the probability p_j to execute such operation, the higher the relevance of the time spent in bay k during such operation with respect to the time spent in it during all the other operations.

  - $\sum_{j=1}^{O} \left(\frac{pbt_{kj}}{oot_j} \cdot p_j \right)$: In order to end up with the proper definition of total time, that must not consider which operation $j \in \mathcal{O}$ forklift is executing, we have to sum the contributions of all of them.
    Ex: if we obtain that, for bay k, $tbt_k = 0.10$, this means that, at any time, there is a 10% probability to find a forklift in that bay.

    Note that $tbt_k = 0$ if and only if bay k is never used by any forklift, during any operation $j \in \mathcal{O}$.

In principle, Total Node Times and Total Bay Times could be enough to obtain an optimal WPT distribution, representing the probabilities, at any time instant, to find the forklift either on each node or in each bay. Nevertheless, the objective is to realize a WPT system able to guarantee a certain battery SoC at the end of the working shift. Battery SoC depends not only on the energy intaked (that, as we will see in section 6.3, can be estimated using TNTV and TBTV), but also on the energy lost, which in turns depends on the type of operation phase forklift is performing. From here, the need to find all the other four Total Times Vectors.

- **Total Node Movement Times / Total Node Operation Times**

  In an analogous way, for each node $i \in \mathcal{N}$, let us calculate $tnmt_i$ and $tnot_i$ as:

$$tnmt_i = \sum_{j=1}^{O} \left(\frac{pnmt_{ij}}{oot_j} \cdot p_j \right) \tag{17}$$

$$tnot_i = \sum_{j=1}^{O} \left(\frac{pnot_{ij}}{oot_j} \cdot p_j \right) \tag{18}$$

  The meaning of each term is the same as the correspondent one in tnt_i.

- **Total Bay Operation Times / Total Bay Recharge Times**

  In an analogous way, for each bay $k \in \mathcal{B}$, let us calculate $tbot_k$ and $tbrt_k$ as:

$$tbot_k = \sum_{j=1}^{O} \left(\frac{pbot_{kj}}{oot_j} \cdot p_j \right) \tag{19}$$

$$tbrt_k = \sum_{j=1}^{O} \left(\frac{pbrt_{kj}}{oot_j} \cdot p_j \right) \tag{20}$$

  The meaning of each term is the same as the correspondent one in tbt_k.

To conclude, let us recall some properties on Total Times, which can be derived by equations (1) and (2):

$$tnt_i = tnmt_i + tnot_i \quad i \in \mathcal{N} \tag{21}$$

$$tbt_k = tbot_k + tbrt_k \quad k \in \mathcal{B} \tag{22}$$

that leads to:

$$\sum_{i=1}^{N} tnt_i = \sum_{i=1}^{N} tnmt_i + \sum_{i=1}^{N} tnot_i \tag{23}$$

$$\sum_{k=1}^{B} tbt_k = \sum_{k=1}^{B} tbot_k + \sum_{k=1}^{B} tbrt_k \tag{24}$$

5.4 Considerations and possible improvements

Equation (3) should be checked after Total Times calculation. It represents the most effective “alarm bell” to identify possible errors in the warehouse modelling phase.

The idea of using Total Times is effective: it is fully reasonable, from the optimization point of view, to be inclined to place WPTs where forklifts spend more time. This improves the efficiency of the system, since WPTs are used for a higher amount of time, while reducing the cost. However, this procedure has some drawbacks that, especially for “large” warehouses, makes it difficult to be applied. Its fundamental issue is strictly related with the probabilistic approach that has been used. In general, the higher the number of operations, the more difficult is to accurately state a probability for each of them. In “large” warehouses, with several hundreds of different operations that might be executed, the definition of accurate enough probabilities for each operation is critical. Such uncertainties may produce a set of Total Time Vectors that are not fully reflecting the situation in analysis. In that case, there is a real risk to calculate a WPT placement that is not able to fulfil battery SoC requirements, being the optimization process based on wrong, or however inaccurate, inputs.

A good solution to overcome this potential problem may be to mount some position sensors on the forklifts, and to collect data on their positions for a long enough time interval (that may be, for example, two months). If the data collected during the period well reflect the real situation, through some manipulations which include the warehouse modelling without defining the whole operations set, it would be possible to assign a Total Time to each node and bay. Then, optimization procedure will carry out an optimal WPT system layout basing on those Total Times, as like as they were computed using the procedure presented here. That WPT system should be able to respect customer SoC demand, provided that the collected data are well refecting the real warehouse processes. This procedure also eliminates the need of modelling all the set of operations and paths. Nonetheless, it requires to wait until data acquisition about forklift positions is accurate enough (not easy to be stated). Other details about this solution can be found in section 9.2.

Chapter 6

Energy Balance

Total Times previously calculated will be used in the optimization step, to build the constraint on the forklift battery State of Charge. Such constraint accounts for the objective of the problem, namely to find a WPT layout which allows the variation of forklift battery SoC at the end of a generic working shift, with respect to the beginning of it, to be greater or equal to a threshold value. Let us first analyze how to calculate such variation.

6.1 Forklift State-of-Charge calculation

Battery SoC, at any instant, is given by:

$$SoC = \frac{E_{battery,actual}}{E_{battery,max}} \tag{25}$$

where:

- $E_{battery,actual}$ is the energy stored in the battery when SoC is measured;
- $E_{battery,max}$ is the maximum energy which can be stored in the battery, calculated from fully-charged battery capacity.

Note that SoC $\in [0,1]$. SoC = 1 means the battery is fully charged. SoC = 0 means the battery is completely exhaust.

We can now define ΔSoC_{shift} as the variation of forklift battery SoC during a shift:

$$\Delta SoC_{shift} = \frac{\Delta E_{shift}}{E_{battery,max}} \tag{26}$$

where ΔE_{shift} is the variation of forklift battery energy during the shift. It can be find as:

$$\Delta E_{shift} = E_{in,shift} - E_{out,shift} \tag{27}$$

with:

- $E_{in,shift}$: Energy Intaked during the working shift, namely the total energy the charging system injects in forklift battery;
- $E_{out,shift}$: Energy Lost during the working shift, due to all the movements executed by the forklift.

Three cases can be distinguished:

1. $\Delta E_{shift} > 0$: Forklift has absorbed more energy than it has lost;
2. $\Delta E_{shift} = 0$: Energy absorbed is exactly equal to the lost one;
3. $\Delta E_{shift} < 0$: Forklift has absorbed less energy than it has lost.

Note that $\Delta SoC_{shift} \in [-1, 1]$. Its sign depends on ΔE_{shift}. Again, let us distinguish three cases:

1. $\Delta SoC_{shift} > 0$ (if $\Delta E_{shift} > 0$): at the end of the shift, forklift battery has more residual charge than at the beginning to it;
2. $\Delta SoC_{shift} = 0$ (if $\Delta E_{shift} = 0$): at the end of the shift, forklift battery has the same charge than at its beginning;
3. $\Delta SoC_{shift} < 0$ (if $\Delta E_{shift} < 0$): at the end of the shift, forklift battery has less residual energy than at the beginning of it.

So, the constraint on the forklift battery SoC can be written as:

$$\Delta SoC_{shift} \geq \Delta SoC_{desired} \tag{28}$$

where $\Delta SoC_{desired}$ represents the minimum desired value, chosen by the customer. This formulation takes the name of *SoC Constraint*, and will be deeply explained in section 7.5.9. Note that the value of ΔSoC_{shift} depends on $\Delta E_{shift} > 0$, which in turns depends on both $E_{in,shift}$ and $E_{out,shift}$. By exploiting the Total Times, we could estimate either the total Energy Intaked and the total Energy Lost during each working shift, in order to finally calculate ΔSoC_{shift}. Before defining how to compute them, let us provide an analysis of the working shift structure considered in this work.

6.2 Working Shift analysis

The working shift is the reference time interval in this work for the evaluation of charging system performance. In this work, it is assumed to have an overall duration of 8 hours. Due to safety rules and worker protections laws, five 15-minutes breaks in each working shift are provided. During that amount of time, forklifts are non-operating, and they should be placed over WPTs for all the break duration, if possible, to exploit as much as possible those dead times to charge forklifts batteries. The effective duration of the shift, that is the amount of it in which forklifts are operating (namely, they are executing an Operation Phase), must take into account overall breaks duration. We can then define the following quantities:

- Overall Shift Duration: $ST_{total} = 8\ hr = 28800\ s$;
- Overall Breaks Duration: $ST_{breaks} = 1\ hr\ 15\ min = 4500\ s$;
- Effective Shift Duration: $ST_{eff} = ST_{total} - ST_{breaks} = 6\ hr\ 45\ min = 24300\ s$;

These quantities will be used in the Energy Intaked or Lost estimation. Let us proceed now in their definition.

6.3 Energy Intaked

Forklifts intake energy when charged by a WPT. Nominal WPT system power is P_n. However, the charging power depends on the WPT efficiency, that changes from static to dynamic case. Let us call η_{stat} the static efficiency and η_{dyn} the dynamic one.

Forklift can be recharged by a WPT in each of the following situations:

- During breaks, in which forklifts should be placed on WPTs, if possible;
- In bays during Bay Recharge phase, if the bay is equipped with a static WPT;
- While moving over dynamic WPTs

Then, the total Energy Intaked in a shift is divided in three contributions, each one related to one of those specific situations:

$$E_{in,shift} = E_{in,breaks} + E_{in,bays} + E_{in,dynamic} \quad (29)$$

with:

- $E_{in,breaks}$: energy intaked, in a shift, during breaks;
- $E_{in,bays}$: energy intaked, in a shift, while the forklift is in bays;
- $E_{in,dynamic}$: energy intaked, in a shift, while the forklift is over dynamic WPTs.

Let us analyze each of them in details:

- **Energy Intaked – Breaks**

During breaks, forklifts are not used. They should be placed over WPTs as long as possible, so to efficiently exploit this period. Parameter k_{breaks} is introduced so to account for the fraction of the break time forklifts may be charged, so that the amount of time in which this occurs, in a whole shift, is given by the product $k_{breaks} \cdot ST_{breaks}$, with $k_{breaks} \in [0,1]$. Note that it may also account for the number of forklifts which can be simultaneously charged during the same break: for example, if there are two forklifts in the warehouse but just one of them can be placed on a static WPT during each break, k_{breaks} will be 0.5.

The total energy intake, in shift, due to the charging during breaks is then:

$$E_{in,breaks} = P_n \cdot \eta_{stat} \cdot ST_{breaks} \cdot k_{breaks} \tag{30}$$

- **Energy Intaked – Bay Recharge**

A forklift can be charged by a static WPT placed in a bay for all the time spent there in Bay Recharge phase. In the whole working shift, the total amount of time spent in a generic bay $k \in \mathcal{B}$ while in Bay Recharge phase is given by the product $tbrt_k \cdot ST_{eff}$. However, the forklift may be actually charged just if bay k is equipped with a static WPT. We can then define a set of binary variables xb_k, with $k \in \mathcal{B}$, such that:

$$xb_k = \begin{cases} 1 & if\ there\ is\ a\ WPT\ in\ bay\ k \\ 0 & otherwise \end{cases}$$

The total energy intaked, in a shift, due to charging in bays is then:

$$E_{in,bays} = P_n \cdot \eta_{stat} \cdot ST_{eff} \cdot \sum_{k=1}^{B} (tbrt_k \cdot xb_k) \tag{31}$$

Note that the product $ST_{eff} \cdot tbrt_k \cdot xb_k$ indicates the time, in a shift, spent by the forklift in bay k being charged. If there is not a WPT in such bay, $xb_k = 0$, and then the whole product goes to zero too. Moreover, remark that the values of the variables xb_k are not a-priori known. They will be inserted as optimization variables in the linear programming problem, with the same meaning as the one reported above. Details on the optimization variables can be found in section 7.3.

- **Energy Intaked – Dynamic**

Whenever the forklift is located over a dynamic WPT, it is charged by it. As we have previously seen, dynamic WPTs are mounted on strips of five consecutive nodes. The total time, in a shift, spent over node i is given by tnt_i. Battery charging occurs during all this time, regardless the operation phase, if that node is equipped with a WPT part.

Again, let us define a set of binary variables x_i, with $i \in \mathcal{N}$, such that:

$$x_k = \begin{cases} 1 & if\ there\ is\ a\ WPT\ on\ node\ i \\ 0 & otherwise \end{cases}$$

The total energy intaked, in a shift, due to dynamic charging is then:

$$E_{in,dynamic} = P_n \cdot \eta_{dyn} \cdot ST_{eff} \cdot \sum_{i=1}^{N} (tnt_i \cdot x_i) \quad (32)$$

Note that, as like as xb_k, the values of all the x_i are not known a-priori, since will be inserted as optimization variables of the linear programming problem, with the same meaning as the one reported above.

6.4 Energy Lost

Forklift Energy Lost calculation is simpler than the Energy Intaked, since it does not depend on the presence of a WPT on nodes/bays. It just depend on Total Times, working shift parameters, and forklift power consumption. The latter depends on which operation phase forklift is executing.

We can call $P_{cons,max}$ the nominal forklift power. During each opearation phase, forklift has a typical power consumption, which is generally lower than $P_{cons,max}$. Those values are reported in Table 6:

Operation Phase	Power Consumption
Bay Operation	$P_{cons,bay,op}$
Bay Recharge	$P_{cons,bay,rech}$
Node Movement	$P_{cons,node,mov}$
Node Operation	$P_{cons,node,op}$

Table 6: Forklift power consumption for each Operation Phase.

So, the total Energy Lost in a shift is can be splitted into four contributions, each one related to one of the operation phases:

$$E_{out,shift} = E_{out,bay,op} + E_{out,bay,rech} + E_{out,node,mov} + E_{out,node,op} \quad (33)$$

where:

- $E_{out,bay,op}$: total energy lost, in a shift, during Bay Operation phase;
- $E_{out,bay,rec}$: total energy lost, in a shift, during Bay Recharge phase;
- $E_{out,node,mov}$: total energy lost, in a shift, during Node Movement phase;
- $E_{out,node,op}$: total energy lost, in a shift, during Node Operation phase.

Let us analyze each of them in details:

- **Energy Lost – Bay Operation**

In the working shift, a forklift elapses an amount of time time equal to $ST_{eff} \cdot tbot_k$ in bay k while in Bay Operation phase. Forklift power consumption in this phase is defined by $P_{cons,bay,op}$. So, $E_{out,bay,op}$ is given by:

$$E_{out,bay,op} = P_{cons,bay,op} \cdot ST_{eff} \cdot \sum_{k=1}^{B} tbot_k \quad (34)$$

- **Energy Lost – Bay Recharge**

Similarly to the previous case, during the working shift, a time equal to $ST_{eff} \cdot tbrt_k$ is spent by the forklift in bay k while in Bay Recharge phase. Therefore, $E_{out,bay,rech}$ is given by:

$$E_{out,bay,rech} = P_{cons,bay,rech} \cdot ST_{eff} \cdot \sum_{k=1}^{B} tbrt_k \quad (35)$$

- **Energy Lost – Node Movement**

In the working shift, a forklift is in Node Movement phase over a certain node $i \in \mathcal{N}$, for an amount of time defined by $ST_{eff} \cdot tnmt_i$. Recalling that power consumption in this phase is represented by $P_{cons,node,mov}$, the total energy lost, in the shift, due to movement over nodes is given by equation (36).

$$E_{out,node,mov} = P_{cons,node,mov} \cdot ST_{eff} \cdot \sum_{i=1}^{N} tnmt_i \qquad (36)$$

- **Energy Lost – Node Operation**

In the same way, the forklift spends, in the shift, an amount of time given by $ST_{eff} \cdot tnot_i$ in Node Operation phase in the area covered by node i. Then, $E_{out,node,op}$ is given by:

$$E_{out,node,op} = P_{cons,node,op} \cdot ST_{eff} \cdot \sum_{i=1}^{N} tnot_i \qquad (37)$$

At this point, it is known how to calculate the forklift energy balance in the working shift. All the quantities defined in this section will be used to build the SoC constraint in the LP problem. Details about it are reported in section 7.5.9.

Chapter 7

Optimization Problem

All the steps described up to now are necessary in order to build the Linear Programming problem aiming at estimating the optimal WPT layout. More specifically, the problem is an Integer Linear Programming one. The whole procedure, from the starting considerations to the results interpretation is reported in this chapter, while the details about the features and the formulation of a generic LP problem have been previously listed in section 2.2.4.

7.1 Objective

The objective of the optimization problem is to find the number and the positions of the WPTs to be installed, alongside with the overall cost of the system and the forecasted value of ΔSoC_{shift}. The cost must be minimal, and ΔSoC_{shift} must be greater or equal than $\Delta SoC_{desired}$. Alongside the already introduced SoC Constraint, a set of constraints on the optimization variables has to be defined in order to allow the computed WPT system to be actually installed. That set must be defined mainly as a consequence of the positioning of each dynamic WPT on more than one node. All the constraints are listed in section 7.5.

7.1.1 Computational time

Another implicit objective is that the computation of the optimal solution have to be done in a “reasonable” amount of time. Computational time required by the optimization algorithm strongly depends on the size of the problem, in particular, on the number of nodes used. The higher the number of nodes in the modelled warehouse, the higher the required computational time. It has been experimentally found that the time requested for finding the optimal solution ranges from few seconds (in case of “small” warehouses, with some hundreds of nodes) to several hours (in case of “big” warehouses, with several thousands of nodes).

7.2 WPT modelling concepts

This section is dedicated to the presentation of some basic concepts used for the WPT modelling, and how they will be handled by the optimization problem.

7.2.1 Node Spacing

One of the most important goals of the model was to decouple as much as possible the allowed dynamic WPT positions with the chosen nodes locations. This means that, in principle, WPT placement should be as more independent as possible from the locations where node have been modelled. The way to achieve this goal would be to reduce nodes spacing as much as possible, so to allow the optimization problem to explore the possibility of placing dynamic WPTs in as many combinations as possible. The lower the node spacing, the higher the possible locations of each single WPT and, theoretically, the lower could be the optimal system cost found by the solver. The effect of node spacing on WPT placement is reported in Figure 23, where the same warehouse corridor has been modeled using three different node spacings. Each time node spacing is reduced, new possible positions for dynamic WPTs arise, that would not have been possible using higher values of it.

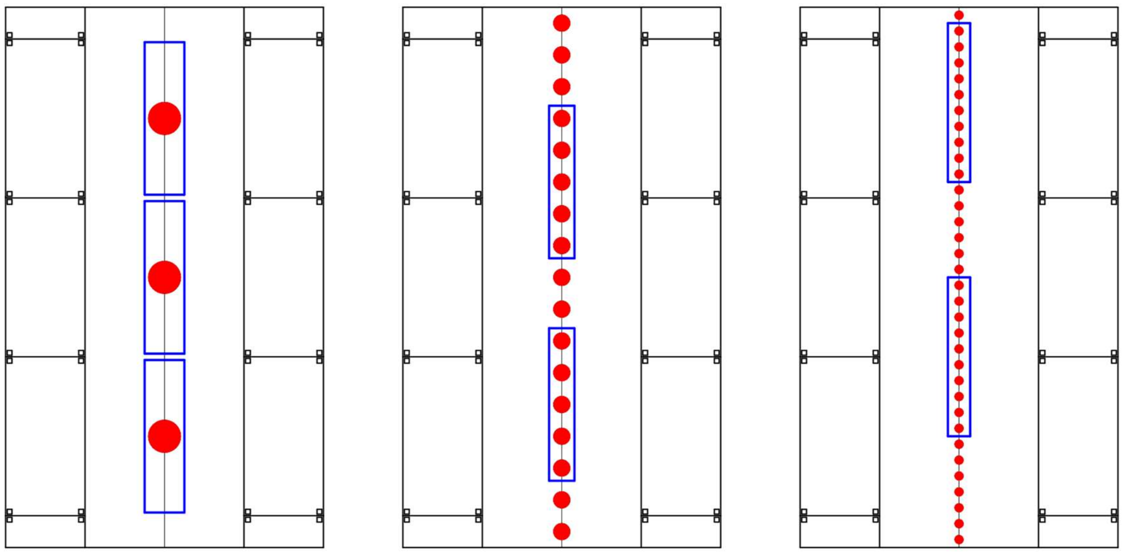

Figure 23: Warehouse modelled with node spacing of, respectively, 2.5 m (left), 0.5 m (middle), 0.25 m (right). Blue rectangles represent some allowed WPT positions in each case.

However, the lower the spacing, the higher the number of nodes, and the higher the computational time. The value of $n_s = 0.5\ m$ used in this work has been selected, being a good tradeoff between these aspects.

7.2.2 WPT parts and centers

In order to correctly model the dynamic WPTs in the optimization problem, and to define the set of constraints for making the system properly installable, we need to distinguish between the five nodes composing the dynamic WPT. Moreover, we need a way to univocally define the WPT position in the modelled warehouse without necessarily specifying all the five nodes ID over which it has to be mounted. From now on, we will refer to *WPT Center* as the central node of the five consecutive nodes strip on which the dynamic WPT has to be mounted, and to *WPT Part* each of the four other nodes composing the strip.

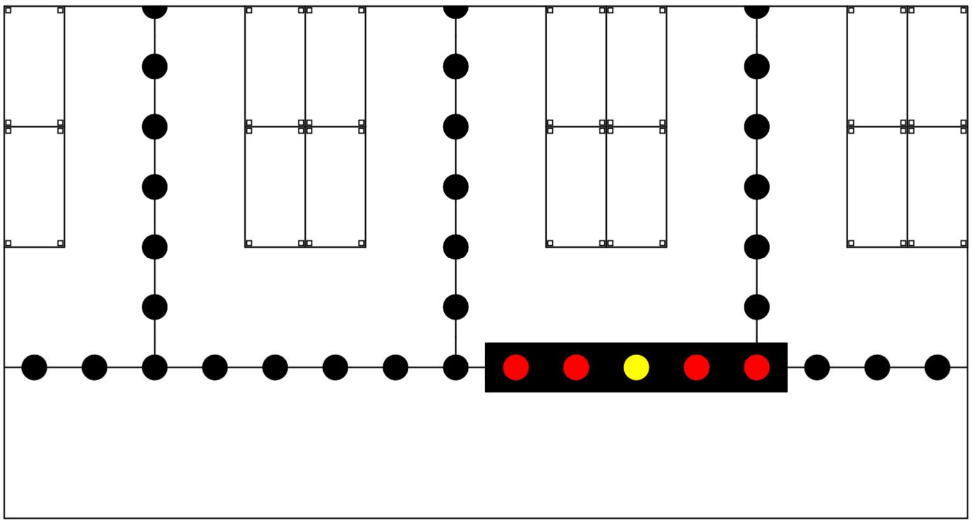

Figure 24: WPT Parts and Center representation. Black rectangles represent the WPT positions. Red nodes represent the WPT parts. Yellow node is the WPT center.

Note that, even if they are called in different ways, there are no physical differences between a WPT Part and a WPT Center, except the position of them in relation to the whole dynamic WPT. In principle, each WPT location could be defined by the position of its WPT center only. However, in order to univocally define them, we have to specify its orientation too.

7.2.3 WPT orientation

A WPT can be oriented just along x-axis or y-axis. Recall that the former is called *Horizontal WPT*, while the latter is called *Vertical WPT*. As reported in previous sections, nodes category cat_i is related to the allowed WPT orientation over node $i \in \mathcal{N}$. Table 7 recaps them for each different node category.

cat_i	**Allowed WPT orientations**
1	Horizontal
2	Vertical
3	Horizontal or Vertical
4	-

Table 7: WPT orientation allowed for each node category.

It is necessary that every node of the strip over which a WPT has to be mounted must be really able to allow its placement in its orientation. So, let us distinguish between Horizontal and Vertical WPT Parts and Centers:

- Each Horizontal WPT is composed by a Horizontal WPT Center (simply called *Horizontal Center*) and by four Horizontal WPT Parts (called *Horizontal Parts*), in a way that the Horizontal Center, that is the central node, is surrounded by the four Horizontal Parts, two in each horizontal side of it.
- Each Vertical WPT is composed by a Vertical WPT Center (simply called *Vertical Center*) and by four Vertical WPT Parts (called *Vertical Parts*), in a way that the Vertical Center, that is the central node, is surrounded by the four Vertical Parts, two in each vertical side of it.

We can refer to *WPT type* to indicate, generically, one of the four entities just defined. Hence, each WPT location can be univocally defined by the position and the orientation of its WPT Center. An example of WPTs definition can be seen in Figure 25. There, the horizontal WPT has to be located on nodes $\{9, 10, 11, 12, 13\}$, while the vertical one on nodes $\{17, 18, 19, 20, 21\}$. By using the univocal WPT definition, we can say that there are two WPTs, one with Horizontal Center on node 11, the other with Vertical Center on node 19. Then, WPT orientation can be read by the Center node one, and all the other nodes composing them can be retrieved directly from the graph.

Moreover, Table 8 shows the allowed WPT categories for each of the node types belonging to a WPT.

Node Type	**Allowed node categories**
Horizontal Center	1, 3
Horizontal Part	1, 3
Vertical Center	2, 3
Vertical Part	2, 3

Table 8: Allowed WPT categories for nodes with a WPT Part/Center.

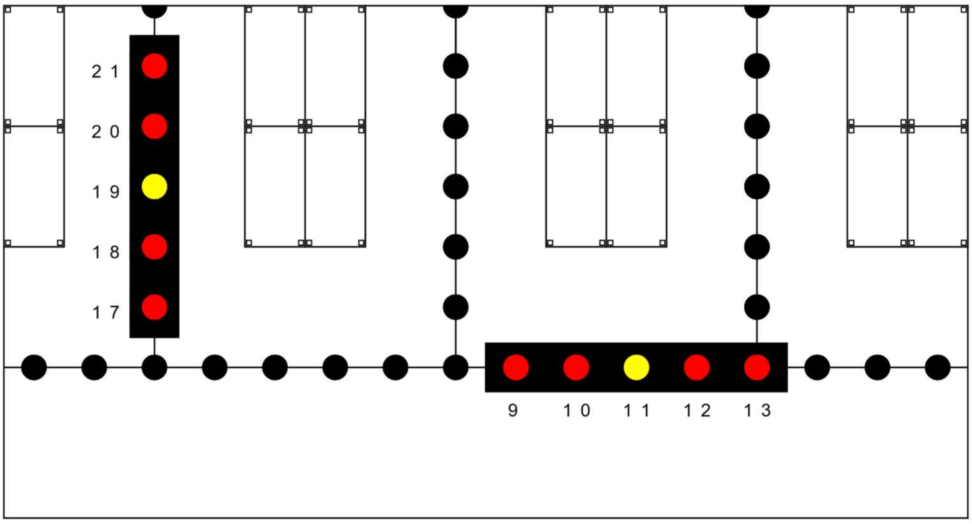

Figure 25: Example of two WPTs, one Horizontal and one Vertical, placed in a warehouse. Black rectangles represent their locations. Red nodes are the WPT Parts, while yellow ones the WPT Centers.

7.2.4 Golden rule on WPT modelling

Following the rules defined up to now, we can define the golden rule for the WPT modelling. This rule represents the heart of the WPT placement in the optimization process:

"A node could accomodate at most one WPT, placed either horizontally or vertically, according to the orientation allowed by its node category. If a WPT is present on the node, it must be one and only one of the following:

- *Horizontal WPT Part;*
- *Horizontal WPT Center;*
- *Vertical WPT Part;*
- *Vertical WPT Center."*

This rule will be mathematically inserted in the LP problem, as a constraint on the optimization variables. More details can be found in section 7.5.1.

7.3 Optimization Variables

The number of optimization variables used in this problem linearly increases with the number of nodes and bays used in the warehouse modelling. Using N nodes and B bays, there will be $5N + B$ different optimization variables. We can define a set θ of all the optimization variables, so that:

$$\theta \in \mathbb{R}^{5N+B} : \theta = \{\theta_1, \theta_2, \dots, \theta_{5N+B}\}$$

Each optimization variable θ_j is binary:

$$\theta_j \in \{0,1\}, \forall j$$

From here, the definition of the Linear Programming Problem as an Integer Linear Programming one, since all the optimization variables are binary and, consequently, integer.

The set θ should now be divided into six subsets, which elements are called, respectively, $x_i, h_i, hc_i, v_i, vc_i, xb_k$, $\forall i \in \mathcal{N}, \forall k \in \mathcal{B}$, so that:

$$\theta = \{x_1, \dots, x_N, h_1, \dots, h_N, hc_1, \dots, hc_N, v_1, \dots, v_N, vc_1, \dots, vc_N, xb_1, \dots, xb_B\}$$

Each one has its own function in the optimization problem. Their meaning, alongside their values, is shown in Table 9:

Variables	Meaning	Value
x_i	Presence of a generic dynamic WPT on node i, regardless its orientation or if it is a Part or a Center	$\begin{cases} 1 & \text{if the WPT is present} \\ 0 & \text{otherwise} \end{cases}$
h_i	Presence of a Horizontal WPT Part on node i	$\begin{cases} 1 & \text{if the WPT is present} \\ 0 & \text{otherwise} \end{cases}$
hc_i	Presence of a Horizontal WPT Center on node i	$\begin{cases} 1 & \text{if the WPT is present} \\ 0 & \text{otherwise} \end{cases}$
v_i	Presence of a Vertical WPT Part on node i	$\begin{cases} 1 & \text{if the WPT is present} \\ 0 & \text{otherwise} \end{cases}$
vc_i	Presence of a Vertical WPT Center on node i	$\begin{cases} 1 & \text{if the WPT is present} \\ 0 & \text{otherwise} \end{cases}$
xb_k	Presence of a static WPT in bay k	$\begin{cases} 1 & \text{if the WPT is present} \\ 0 & \text{otherwise} \end{cases}$

Table 9: Optimization variables meaning and values.

Optimization variable x_i must be set to 1 if any WPT Part/Center is placed over node i. The variable related to the corresponding type of Part/Center must be set to 1 too. For example, if node i has a Vertical WPT Center, both x_i and vc_i must be set to 1.

7.4 Cost Function

Cost Function, which represents the overall cost of the WPT system, represents the functional to be minimized. It must be a function of the optimization variables. The cost of each WPT Part/Center is given by c_d, while each Static WPT has a cost given by c_s.

In this section, θ represents the vector of optimization variables, arranged as:

$$\theta = [x_1, \dots, x_N, h_1, \dots, h_N, hc_1, \dots, hc_N, v_1, \dots, v_N, vc_1, \dots, vc_N, xb_1, \dots, xb_B]^T$$

The Cost Function is a functional:

$$f(\theta) = c^T\theta \tag{38}$$

Where c is the cost vector, indicating the cost of each optimization variable. It must be representative of the real cost charged to the customer for the WPT system to be installed. Therefore, it has to be arranged as follows:

$$c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{5N+B} \end{bmatrix} \qquad \text{where } c_i = \begin{cases} c_d & i \in [1, N] \\ c_0 & i \in [N+1,\ 5N] \\ c_s & i \in [5N+1,\ 5N+B] \end{cases}$$

The term $c_0 = 0$ represents a fictitious cost, used to assign null cost to optimization variables $h_i, hc_i, v_i, vc_i, \forall i$. In fact, each dynamic WPT Part or Center has the same cost c_d, so it is convenient to assign such cost to optimization variable x_i which, as said before, is always set to 1 if a whatever type of WPT has to be installed on node i.

A more intuitive formulation of the cost function is as follows:

$$f(\theta) = \sum_i c_d x_i + \sum_i c_0 h_i + \sum_i c_0 hc_i + \sum_i c_0 v_i + \sum_i c_0 vc_i + \sum_k c_s xb_i \tag{39}$$

or alternatively, by considering only the optimization variables with a non-null cost:

$$f(\theta) = \sum_i c_d x_i + \sum_k c_s xb_k \tag{40}$$

7.5 Constraints

Constraints in Linear Programming Problems are a set of linear equalities or inequalities used to mathematically express some "limitations" in the values given to optimization variables that, in this work, are representative of the WPT placement. Their correct definition is fundamental, and each of them must be deeply analyzed to understand which are its limitations, and then to decide if one or more further constraints have to be defined in order to avoid the calculation of a not "real" WPT layout. As a matter of example, some of the situations that must be constrained are:

- A WPT is constituted by five consecutive nodes. So, each strip of less than five nodes with a WPT Part/Center is meaningless, and must not be present in the optimal solution.
- In a strip of five consecutive nodes constituting a WPT, four of them must be WPT Parts, and one of them a WPT Center. More specifically, the WPT Center must be the central node, with two WPT Parts on each side of it.
- The company policy states to avoid the placement of single dynamic WPT modules in the warehouse. No less than two contiguous modules, oriented in the same direction, can be installed. The reason behind this choice is that is not convenient to place a set of single modules in many zones of the warehouse, since too many "jumps" inside and outside the WPT coverage may damage forklift batteries due to of the power supply discontinuities.

Each of those limitations must be expressed in a mathematical way, either as an equality or an inequality, that must be linear in the optimization variables.

In the following sections, constraints will be presented one by one, explaining their meaning, the reason why they were added, and their mathematical formulation.

7.5.1 Unicity Constraint

The first constraint is concerning about the "intrinsic" meaning of the optimization variables about WPT on nodes. It is basically the mathematical formulation of the golden rule on WPT modelling, reported in section 7.2.4, in relation to the node categories.

Recall that for that rule, if a node $i \in \mathcal{N}$ is part of a WPT, then $x_i = 1$. The WPT type mounted on that node must be exactly one among: Horizontal WPT Part, Horizontal WPT Center, Vertical WPT Part, Vertical WPT Center. Then, one and only one among h_i, hc_i, v_i, vc_i must be equal to 1. The limitations on the WPT type and orientation which can be placed on that node are given by its node category cat_i. Each node $i \in \mathcal{N}$ such that $cat_i \in \{1, 2, 3\}$ has to be considered here. Note that category 4 nodes will be treated in the next section.

The formal definition of the Unicity Constraint is as follows:

If a node has a WPT on it:

- *If that node is an Horizontal Node, then that WPT must be either a Horizontal Part or a Horizontal Center;*
- *If that node is a Vertical Node, then that WPT must be either a Vertical Part or a Vertical Center;*
- *If that node is a Cross Node, then that WPT must be one among: Horizontal Part, Horizontal Center, Vertical Part, Vertical Center.*

which, in mathematical terms, can be formulated as:

$\forall i \in \mathcal{N}$, if $x_i = 1$, then:

- If $cat_i = 1 \Rightarrow$ It must be $h_i = 1$ or $hc_i = 1$;
- If $cat_i = 2 \Rightarrow$ It must be $v_i = 1$ or $vc_i = 1$;
- If $cat_i = 3 \Rightarrow$ It must be $h_i = 1$ or $hc_i = 1$ or $v_i = 1$ or $vc_i = 1$;

To mathematically insert this constraint into the LP problem, a set of linear equalities can be used. For each node $i \in \mathcal{N}$ such that $cat_i \in \{1, 2, 3\}$ an equality constraint has to be formulated. The structure of that equality depends on the value of cat_i. The mathematical structure of these constraints is as follows:

$$\begin{cases} 3x_i - 3h_i - 3hc_i - v_i - vc_i = 0 & \forall i \in \mathcal{N} : \ cat_i = 1 \quad (41) \\ 3x_i - h_i - hc_i - 3v_i - 3vc_i = 0 & \forall i \in \mathcal{N} : \ cat_i = 2 \quad (42) \\ x_i - h_i - hc_i - v_i - vc_i = 0 & \forall i \in \mathcal{N} : \ cat_i = 3 \quad (43) \end{cases}$$

The results achieved using this formulation, which perfectly matches the formal definition of Unicity Constraint, are as follows:

- If a generic node i has a WPT on it (if $x_i = 1$), then just one WPT type is assigned to that node, among the ones allowed by its category, and vice versa.
- If no WPTs are mounted on node i (if $x_i = 0$), then no WPT types can be assigned to that node. So, if $x_i = 0$, then it must be $h_i = 0$, $hc_i = 0$, $v_i = 0$, $vc_i = 0$, and vice versa.

7.5.2 WPT Placement Impossibility Constraint

It is now time to consider category 4 nodes. No WPTs, of any type, can be placed on them. So, if node $i \in \mathcal{N}$ has $cat_i = 4$, then $x_i = 0$. Moreover, all the other optimization variables h_i, hc_i, v_i, vc_i must be set to 0 too. The reason is that, without constraining them, they can be set to 1, without a real meaning, but to solve some issues related to the constraints defined in next sections. That fact will produce an unreal WPT layout, not recognized by the optimization algorithm since is would not be violating any mathematical constraints.

We can formally define this constraint as follows:

If a node is an Impossible Node, then no WPT, of any type, can be placed on it.

which, in mathematical terms, can be formulated as:

$\forall i \in \mathcal{N}$, if $cat_i = 4$, then it must be $x_i = 0$, $h_i = 0$, $hc_i = 0$, $v_i = 0$, $vc_i = 0$

To mathematically insert this constraint into the LP problem, a set of linear equalities can be used. For each node $i \in \mathcal{N}$ such that $cat_i = 4$, an equality constraint has to be formulated. The mathematical structure of this constraints is as follows:

$$x_i + h_i + hc_i + v_i + vc_i = 0 \quad \forall i \in \mathcal{N} : cat_i = 4 \tag{44}$$

7.5.3 Corridors Constraints

Corridors can potentially host an arbitrary number of WPTs, obviously limited by their maximum length. In principle, each corridor can just host WPTs oriented like it. However, WPTs oriented oppositely could be placed on Cross Nodes belonging to that corridor. This is the only exception to that rule, and it is clear from the Unicity Constraint, that is responsible of preventing placement of WPT Parts/Centers oriented in not allowed ways on corridor nodes.

The idea behind this new constraint is to force the optimization algorithm to position a "correct" number of WPT Parts and Centers in each corridor. Recalling that a dynamic WPT must be composed by four WPT Parts and one WPT Center, with the same orientation, then in each corridor there must be a WPT Center for each four WPT Parts. In other words, the number or WPT Parts in a corridor must be four times the number of WPT Centers in it. To avoid the problem of WPT Parts/Centers placed in a different orientation with respect to the corridor one (allowed on Cross Nodes), which may alter the expected result, this proportion must be applied just for WPT Parts/Centers oriented like the corridor. Figure 26 shows this situation: in the horizontal corridor, 5 WPT modules have been correctly placed over 5 Centers and 20 Parts. However, in the whole corridor there are 6 Centers and 21 Parts, since an incorrect corridor constraint would also consider those composing the Vertical WPTs and belonging to both the Horizontal and the Vertical Corridors. With the correct constraint formulation, this situation would be rightly accepted.

For this reasons, we should divide this constraint into two distinct definitions: one for Horizontal Corridors, the other for the Vertical ones.

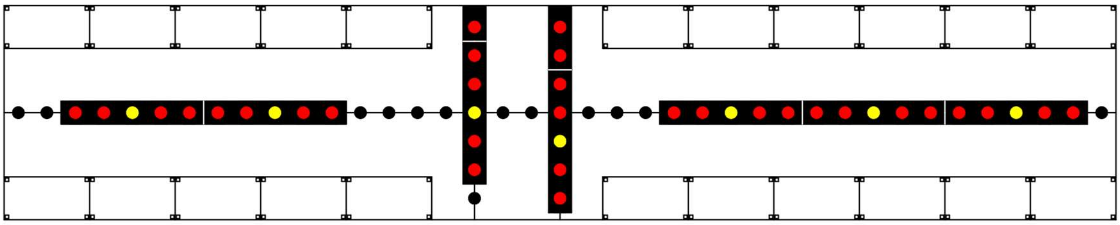

Figure 26: Example of application of the corridor constraint to the Horizontal one, over which there are 5 Horizontal Centers, 20 Horizontal Parts, but also a Vertical Center and a Vertical Part, placed over the intersections with the Vertical Corridors.

7.5.3.1 Horizontal Corridor Constraints

With reference to what explained above, recalling that $\mathcal{H}$ is the set of all the Horizontal corridors, the formal definition of this constraint is:

In any Horizontal corridor, the sum of all the Horizontal WPT Parts must be exactly four times the sum of all the Horizontal WPT Centers.

To mathematically insert it into the optimization problem, we need to define an equality constraint for each Horizontal corridor $j \in \mathcal{H}$, which is constraining all the nodes $i \in \mathcal{N}$ belonging to it:

$$\sum_{i=1}^{N} h_i - 4\sum_{i=1}^{N} hc_i = 0 \quad \forall i \in \mathcal{N} : \; horc_i = j, \;\; j \in \mathcal{H} \tag{45}$$

7.5.3.2 Vertical Corridor Constraints

In an analogous way, recalling that $\mathcal{V}$ is the set of all the Vertical corridors, the formal definition of this constraint is:

In any Vertical Corridor, the sum of all the Vertical WPT Parts must be exactly four times the sum of all the Vertical WPT Centers.

Then, we need to define an equality constraint for each Vertical corridor $j \in \mathcal{V}$, which is constraining all the nodes belonging to it:

$$\sum_{i=1}^{N} v_i - 4\sum_{i=1}^{N} vc_i = 0 \quad \forall i \in \mathcal{N} : \; verc_i = j, \;\; j \in \mathcal{V} \tag{46}$$

7.5.3.3 Summing Up

Considering the constraints imposed up to now (Unicity, WPT Placement Impossibility, Corridors), we can develop an optimization problem able to implement the following features:

- Each node on which is possible to place a WPT must host at most one WPT Type among those allowed by the node category;
- No WPTs can be placed on category 4 nodes;
- The sum of WPT Parts and Centers in each corridor is "real", which means that, using the WPT modelling with Parts and Centers, optimization algorithm is able to determine a number of them which truly allows the placement of physical WPTs.

However, the first arising problem is that, even if a correct number of WPT Parts and Centers to be positioned is guaranteed by the Corridor Constraints, no instructions on their positions in the corridors has still been imposed. A real WPT must composed by a strip of five consecutive nodes, with the WPT Center in the middle, surrounded by two WPT Parts per side. Though, up to now, no constraints on the need to form those strips has been imposed. Consequently, situations like the ones reported in Figure 27 and 28 would not violate any constraint, and so could be proposed in the optimal solution even if conceptually wrong.

To solve this issue, we need to define other constraints. Let us start by defining the concept of Node Neighbors which has been used in this work.

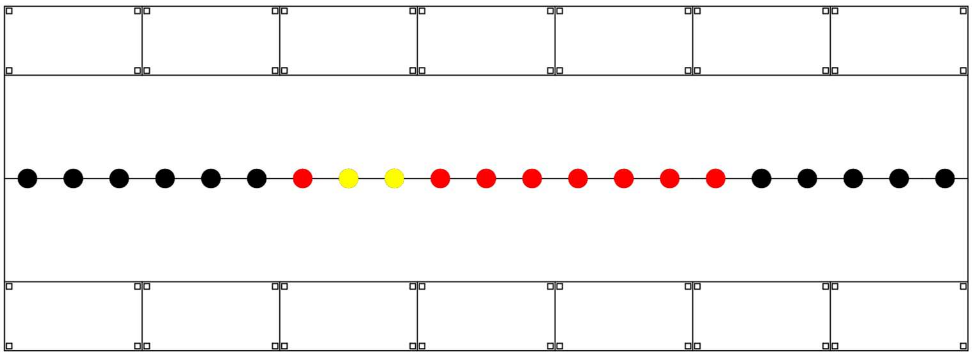

Figure 27: Example of incorrect WPT Parts and Centers placement, allowed by the Corridor Constraint.

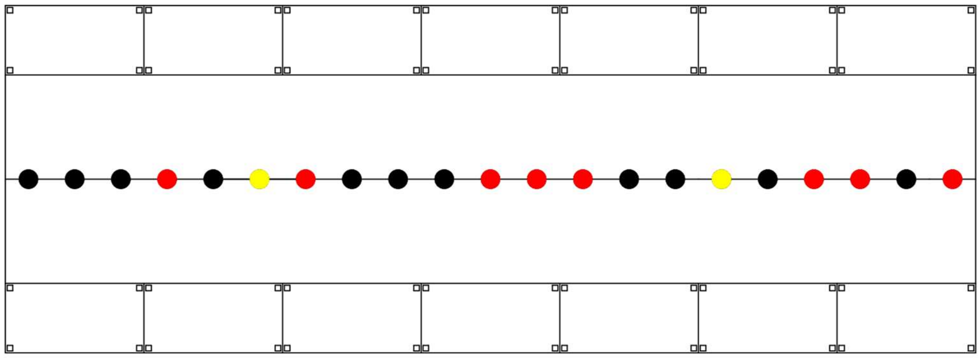

Figure 28: Other example of incorrect – but allowed – WPT Parts and Centers placement in the corridor.

7.5.4 Node Neighbors

Let us define *Neighbor* of node $i \in \mathcal{N}$ whatever node $j \in \mathcal{N} : i \neq j$ such that the couple (i, j) is connected by a single edge [46] [47]. Let us also define *Neighborhood* of node i the set of all the nodes which are neighbors of node i [47]. Note that, if node j belongs to the neighborhood of node i, also the contrary holds, that is node i belongs to the neighborhood of node j.

The concept of node neighbor must be introduced since, as said before, we need to define some constraints to force the creation of strips of five WPT Parts and Centers such that a Center is in the middle of it, surrounded by two Parts per side. Moreover, we still have to force that no less than two adjacent WPT modules must be placed. To develop a mathematical formulation for these constraints such that they can be introduced in the LP problem, it has been found that the simple concept of node neighbor is not sufficient. Instead, we need to define more complex entities called “Radius-2-Neighbors” and “Distance-5-Neighbors”, that are strictly related to the concept of Neighborhood of a node.

7.5.4.1 Radius-2-Neighbors

We can define *Radius-2-Neighbor* of a node $i \in \mathcal{N}$ whatever node $j \in \mathcal{N} : i \neq j$ that is connected to node i through at most 2 edges. Roughly speaking, we can say that whatever node $j \in \mathcal{N}$ that is a Radius-2-Neighbor of node i, belongs to the neighborhood or a node which in turn is a neighbor of node i. Let us also call *Radius-2-Neigborhood* of node i the set of nodes $j \in \mathcal{N} : i \neq j$ which are Radius-2-Neighbors of node i.

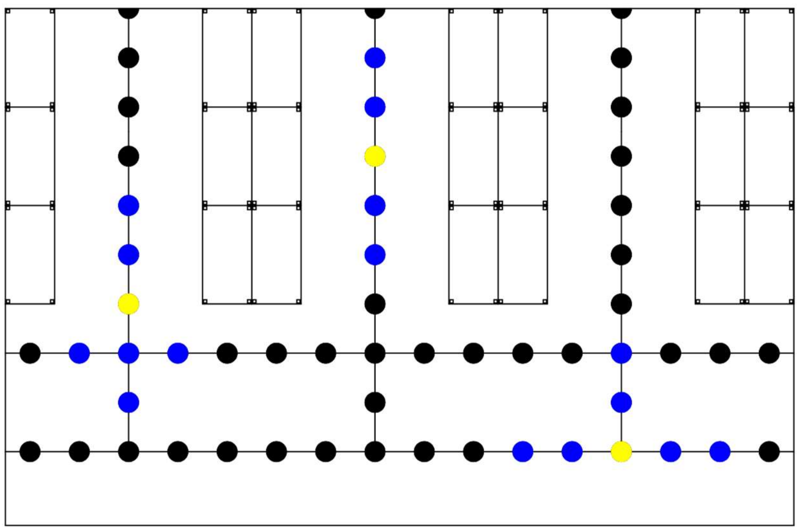

Figure 29: Example of node neighbors. Each set of blue nodes represent the Radius-2-Neighborhood of the corresponding yellow node.

We can define now two subclasses of Radius-2-Neighbors of a node:

7.5.4.2 Radius-2 Horizontal Neighbors

Given a node $i \in \mathcal{N}$, we can define *Radius-2 Horizontal Neighbor* of node i whatever node $j \in \mathcal{N} : i \neq j$ such that:

- j belongs to the same horizontal corridor as $i \Leftrightarrow horc_j = horc_i$;
- j is connected to node i through at most 2 different edges;
- If j is connected to node i through exactly two different edges, the node that is connected to both i and j through those 2 edges must belong to the same horizontal corridor of both nodes i and j.

Then, we can define *Radius-2 Horizontal Neigborhood* of node i the set $\mathcal{HN}_i$ of nodes $j \in \mathcal{N} : i \neq j$ that are Radius-2 Horizontal Neighbors of node i. The maximum number of nodes belonging to set $\mathcal{HN}_i$ is 4. Nodes $i \in \mathcal{N}$ which do not belong to any Horizontal Corridor, have $\mathcal{HN}_i = \emptyset$. Therefore, the set $\mathcal{HN}_i$ is relevant for nodes $i \in \mathcal{N}$ such that $horc_i \neq 0$ only.

Figure 30 shows an example of Radius-2 Horizontal Neighbors of some nodes. The Radius-2 Horizontal Neighborhood of yellow nodes in that figure is reported in Table 10.

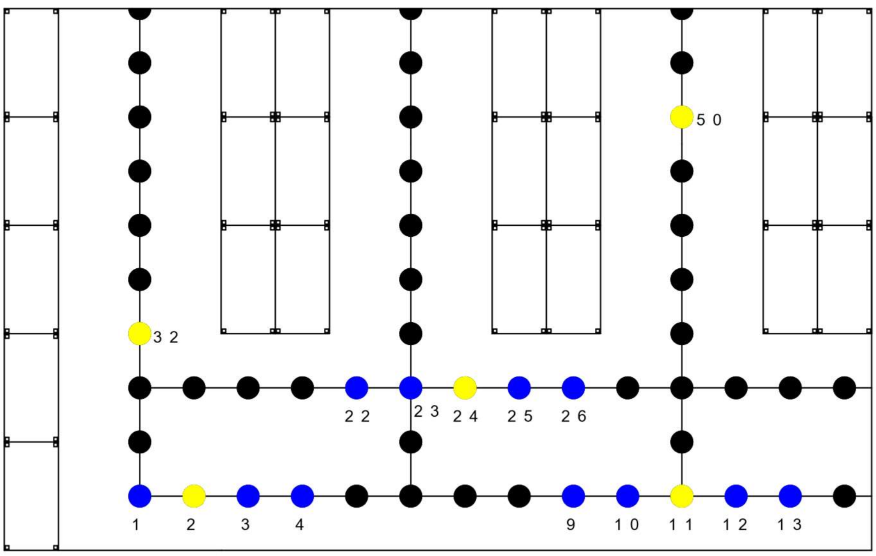

Figure 30: Example of Radius-2 Horizontal Neighbors. Each set of blue nodes represent the Radius-2-Neighborhood of the corresponding yellow node.

Node $\boldsymbol{i}$	$\boldsymbol{\mathcal{HN}_i}$
2	$\{1, 3, 4\}$
11	$\{9, 10, 12, 13\}$
24	$\{22, 23, 25, 26\}$
32	$\emptyset$
50	$\emptyset$

Table 10: Radius-2 Horizontal Neighborhood of yellow nodes in Figure 30.

7.5.4.3 Radius-2 Vertical Neighbors

In a similar way, given a node $i \in \mathcal{N}$, we can define *Radius-2 Vertical Neighbor* of node i whatever node $j \in \mathcal{N} : i \neq j$ such that:

- j belongs to the same vertical corridor as $i \Leftrightarrow verc_j = verc_i$;
- j is connected to node i through at most 2 different edges;
- If j is connected to node i through exactly 2 different edges, the node that is connected to both i and j through those 2 edges must belong to the same vertical corridor of both nodes i and j.

Let $\mathcal{VN}_i$ be the *Radius-2 Vertical Neigborhood* of node $i \in \mathcal{N}$, namely the set of nodes $j \in \mathcal{N} : i \neq j$ which are Radius-2 Vertical Neighbors of node i. The maximum number of nodes belonging to set $\mathcal{VN}_i$ is 4. Nodes $i \in \mathcal{N}$ which do not belong to any Vertical Corridor, have $\mathcal{VN}_i = \emptyset$. Therefore, the definition of set $\mathcal{VN}_i$ is relevant only for nodes $i \in \mathcal{N}$ such that $verc_i \neq 0$.

Note that the relevant Radius-2 Horizontal and Vertical Neighborhoods will be used for constraining the correct WPT Parts and Center positions in each WPT strip. More details can be found in sections 7.5.5 and 7.5.6.

7.5.4.4 Distance-5-Neighbors

Let us define now the Distance-5 Neighbors of a node, by directly focus on their sub-classes.

7.5.4.5 Distance-5 Horizontal Neighbors

Given a node $i \in \mathcal{N}$, we can define *Distance-5 Horizontal Neighbor* of node i any node $j \in \mathcal{N} : i \neq j$ such that:

- j belongs to the same horizontal corridor as $i \Leftrightarrow horc_j = horc_i$;
- j is connected to node i through exactly five different edges;
- All the nodes, different from node i and j, which are connected by those five edges, must belong to the same Horizontal corridor as node i and node j.

Let $\mathcal{HNN}_i$ be the *Distance-5 Horizontal Neigborhood* of node $i \in \mathcal{N}$, namely the set of nodes $j \in \mathcal{N} : i \neq j$ which are Distance-5 Horizontal Neighbors of node i. The maximum number of nodes belonging to set $\mathcal{HNN}_i$ is 2. Nodes $i \in \mathcal{N}$ which do not belong to any Horizontal Corridor, have $\mathcal{HNN}_i = \emptyset$. As a consequence, the definition of set $\mathcal{HNN}_i$ is relevant only for nodes $i \in \mathcal{N}$ such that $horc_i \neq 0$.

An example of Distance-5 Horizontal Neighbors of some nodes can be seen in Figure 31. The Distance-5 Horizontal Neighborhood of yellow nodes in that figure is reported in Table 11.

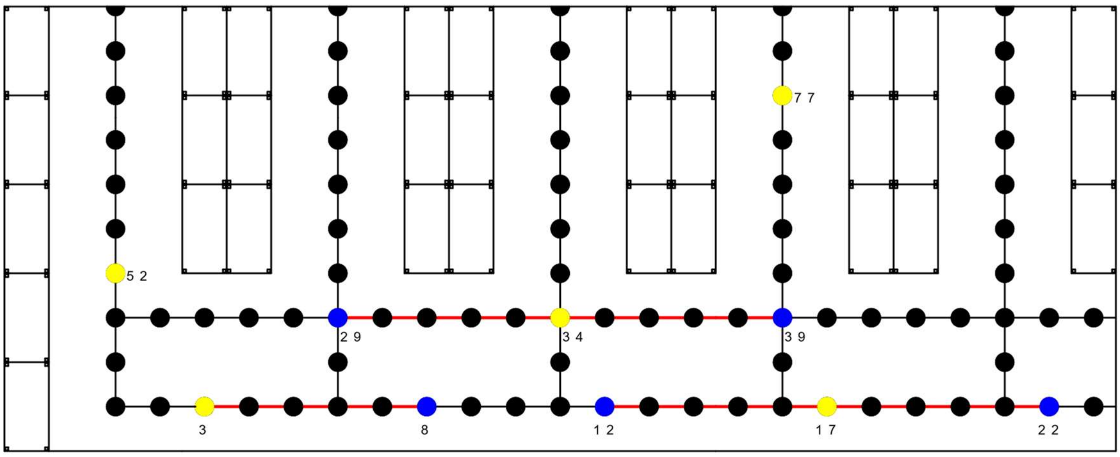

Figure 31: Example of Distance-5 Horizontal Neighbors. Each set of blue nodes represent the Distance-5 Neighborhood of the corresponding yellow node, to which are connected by means of the red edges.

Node $\boldsymbol{i}$	$\boldsymbol{\mathcal{HNN}_i}$
3	$\{8\}$
17	$\{12, 22\}$
34	$\{29, 39\}$
52	$\emptyset$
77	$\emptyset$

Table 11: Distance-5 Horizontal Neighborhood of yellow nodes in Figure 31.

7.5.4.6 Distance-5 Vertical Neighbors

In a similar way, given a node $i \in \mathcal{N}$, we can define *Distance-5 Vertical Neighbor* of node i whatever node $j \in \mathcal{N} : i \neq j$ such that:

- j belongs to the same vertical corridor as $i \Leftrightarrow verc_j = verc_i$;
- j is connected to node i through exactly 5 different edges;
- All the nodes, different from node i and j, which are connected by those five edges, must belong to the same Vertical corridor as node i and node j.

Moreover, let $\mathcal{VNN}_i$ be the *Distance-5 Vertical Neigborhood* of node $i \in \mathcal{N}$, namely the set of nodes $j \in \mathcal{N} : i \neq j$ which are Distance-5 Vertical Neighbors of node i. Note that the maximum number of nodes belonging to set $\mathcal{VNN}_i$ is 2. Nodes $i \in \mathcal{N}$ which do not belong to any Vertical Corridor, have $\mathcal{VNN}_i = \emptyset$. As a consequence, the definition of $\mathcal{VNN}_i$ is relevant only for nodes $i \in \mathcal{N}$ such that $verc_i \neq 0$.

Note that the relevant Distance-5 Horizontal and Vertical Neighborhoods will be used for constraining the placement of no less than two adjacent WPT modules oriented in the same direction. More details can be found in section 7.5.7.

7.5.5 WPT Center Position Constraints

We can exploit the Radius-2-Neighbors of a node to define the first constraint aiming at choosing the optimization variable values so to obtain WPTs which can be rightly installed. This constraint is applied to all nodes which may potentially be WPT Centers.

A WPT center can be positioned on a certain node, whether horizontally or vertically, just if four WPT Parts could be positioned, respectively, on its Radius-2 Horizontal or Vertical Neighbors. Then, nodes which does not have four Radius-2 Horizontal or Vertical Neighbors, cannot be, respectively, Horizontal or Vertical Centers. On the other side, it could happen that a node having four Radius-2 Neighbors is not a Center, also if some or even all the nodes belonging to its Radius-2 Neighborhood are WPT Parts (note that, in that case, those parts belong to a WPT centered in another node).

As was done in the Corridor Constraints, let us divide this constraint formulation in two parts, one relative to the placement of Horizontal WPTs, the other relative to the placement of Vertical WPTs.

7.5.5.1 Horizontal WPT Center Position Constraint

This constraint is imposed to force the correct positioning of each Horizontal WPT Center, which must be located among four Horizontal Parts, two on each horizontal side of it. So, all the nodes belonging to its Radius-2 Horizontal Neighborhood must be Horizontal Parts. This constraint is applied to each node $i \in \mathcal{N}$ which can be a Horizontal WPT Center, namely which has four Radius-2 Horizontal Neighbors.

The formal definition of this constraint is:

If a node is a Horizontal WPT Center, it must have exactly four Radius-2 Horizontal Neighbors, and all of them must be Horizontal WPT Parts.

In mathematical terms, we can formulate it as follows:

$\forall i \in \mathcal{N}$, if $hc_i = 1$, then it must be $|\mathcal{HN}_i| = 4, h_j = 1, \forall j \in \mathcal{N} : j \in \mathcal{HN}_i$

To mathematically insert this constraint into the optimization problem, an inequality constraint must be defined for each node $i \in \mathcal{N}$ having $|\mathcal{HN}_i| = 4$. The constraint is as follows:

$$4hc_i - \sum_j h_j \leq 0 \quad \forall i \in \mathcal{N},\ \forall j \in \mathcal{HN}_i : |\mathcal{HN}_i| = 4 \tag{47}$$

Note that the choice to express it as an inequality allows node i not to be a WPT Center, even if it has 4 Radius-2 Horizontal Neighbors, and if some, or even all, of them are WPT Parts.

7.5.5.2 Vertical WPT Center Position Constraint

Similarly, this constraint is imposed to force the correct positioning of each Vertical WPT Center, whose nodes belonging to its Radius-2 Vertical Neighborhood must be Vertical Parts.

The formal definition of this constraint is:

If a node $i \in \mathcal{N}$ is a Vertical WPT Center, it must have exactly four Radius-2 Vertical Neighbors, and all of them must be Vertical WPT Parts.

In mathematical terms, we can formulate it as follows:

$\forall i \in \mathcal{N}$, if $vc_i = 1$, then it must be $|\mathcal{VN}_i| = 4, v_j = 1, \forall j \in \mathcal{N} : j \in \mathcal{VN}_i$

To mathematically insert it into the optimization problem, an inequality constraint must be defined for each node $i \in \mathcal{N}$ having $|\mathcal{VN}_i| = 4$. The constraint is as follows:

$$4vc_i - \sum_j v_j \leq 0 \quad \forall i \in \mathcal{N},\ \forall j \in \mathcal{VN}_i : |\mathcal{VN}_i| = 4 \qquad (48)$$

7.5.5.3 Summing Up

The WPT Center Position Constraint has introduced other rules to force the optimization process to produce an output that can be physically implemented. At a first sight it would seem that, considering both the inequalities of this constraint and the equalities coming from the Corridor Constraint, real WPTs can be positioned in the modelled warehouse. However, there are still two crucial lacks:

- With the WPT Center Position Constraint, we have just forced the Centers to have all the four Radius-2-Neighbors to be WPT Parts. But the nodes on which those WPT Parts have to be located may belong to the Radius-2-Neighborhood of more than a WPT Center. Therefore, a situation like the one in Figure 32 would be totally acceptable, since:
  - In the corridor there are 12 Parts and 3 Centers (Corridor Constraint respected);
  - Each WPT Center has four Radius-2-Neighbors, and all of them are WPT Parts (WPT Center Position Constraint respected).

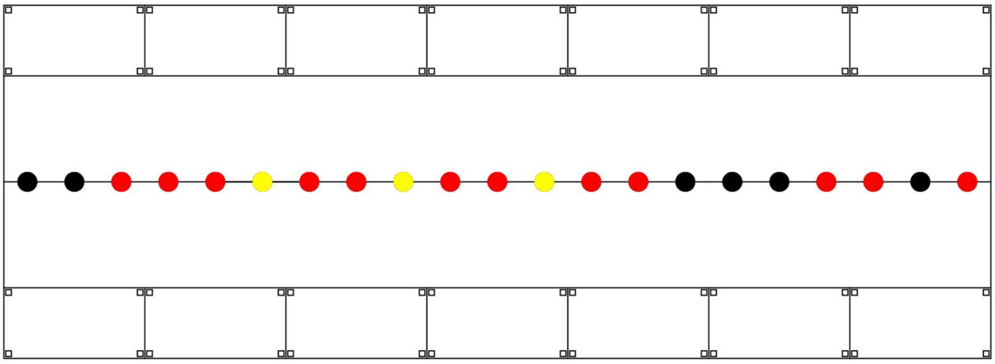

Figure 32: Example of incorrect - but still allowed by the constraints system - WPT Parts and Centers layout, which are indicated by, respectively, red and yellow nodes.

- WPT Center Position Constraint applies just to nodes $i \in \mathcal{N}$ having four Radius-2-Neighbors. WPT Centers placement on nodes which do not respect this latter condition is therefore not ruled by that constraint. Then, a situation like that reported in Figure 33, which is clearly not admissible using our modelling approach, would be acceptable again.

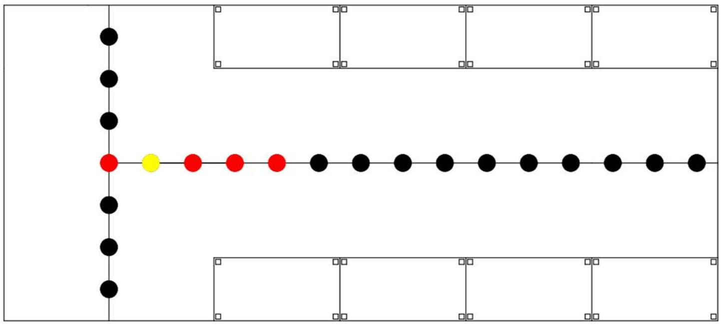

Figure 33: Other example of incorrect - but still allowed by the constraint system - WPT Parts and Centers layout.

Another constraint has be introduced in order to solve those issues and provide a correct WPT deployment. In both the cases reported above, it would be enough to impose a constraint on the WPT Parts, forcing them to have one and only one WPT Center among their Radius-2-Neighbors. We can then define a new set of constraints, which are called *WPT Part Position Constraints*.

7.5.6 WPT Part Position Constraints

We can exploit again the Radius-2-Neighbors of a node to force each WPT Part to have exactly one WPT Center among the nodes belonging to its Radius-2 Neighborhood. This constraint must be applied to every node $i \in \mathcal{N}$. Let us divide this constraint formulation in two parts, one for Horizontal WPT Parts, the other for Vertical WPT Parts.

7.5.6.1 Horizontal WPT Part Position Constraint

The formal definition of this constraint is as follows:

A Horizontal or Cross node can be a Horizontal WPT Part if and only if exactly one node belonging to the Radius-2 Horizontal Neighbors of the former is a Horizontal WPT Center.

In mathematical terms, it can be formulated as:

$\forall i \in \mathcal{N} \,:\, cat_i \in \{1,3\}$, it can be $h_i = 1 \Leftrightarrow \exists!\ j \in \mathcal{HN}_i \,:\, hc_j = 1$

To mathematically insert this statement into the LP problem, an equality constraint must be defined for each node $i \in \mathcal{N} \,:\, cat_i \in \{1,3\}$. The constraint is as follows:

$$h_i - \sum_j hc_j = 0 \quad \forall i \in \mathcal{N},\ \forall j \in \mathcal{HN}_i : cat_i \in \{1,3\} \tag{49}$$

7.5.6.2 Vertical WPT Part Position Constraint

The formal definition of this constraint is as follows:

A Vertical or Cross node can be a Vertical WPT Part if and only if exactly one node belonging to the Radius-2 Vertical Neighbors of the former is a Vertical WPT Center.

In mathematical terms, it becomes:

$\forall i \in \mathcal{N} \,:\, cat_i \in \{2,3\}$, it can be $v_i = 1 \Leftrightarrow \exists!\ j \in \mathcal{VN}_i \,:\, vc_j = 1$

To mathematically insert this statement into the LP problem, an equality constraint must be defined for each node $i \in \mathcal{N} \,:\, cat_i \in \{2,3\}$. The constraint is as follows:

$$v_i - \sum_j vc_j = 0 \quad \forall i \in \mathcal{N},\ \forall j \in \mathcal{VN}_i : cat_i \in \{2,3\} \tag{50}$$

It is relevant to note that, by expressing the constraint using this equality, all the nodes in the Radius-2-Neighborhood of a WPT Center are forced, correctly, to be WPT Parts, since the WPT Center belongs to the Radius-2-Neighborhoods of all those four Parts.

7.5.6.3 Summing Up

With the addition of the WPT Part Position Constraint, the whole set of linear equalities and inequalities defined up to now can guarantee the optimal solution found by the LP problem to be physically implementable, provided that all the input data are correct. However it is necessary to define another constraint forcing the deployment of WPT strips composed by at least two adjacent modules. Indeed, a situation like the one shown in Figure 34, which do not violate any constraint defined up to now, cannot be realized in practice, due to the presence of single WPT modules in the corridor which violate the company policy.

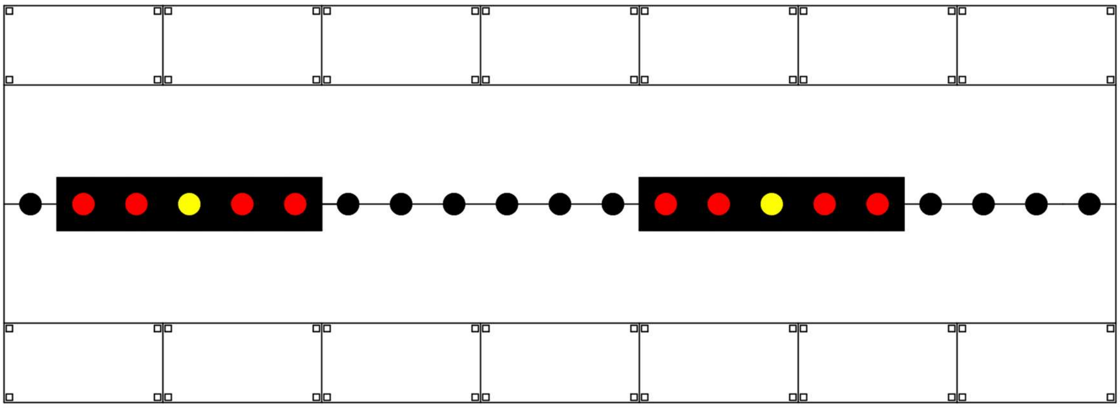

Figure 34: Example of WPT layout which do not respect the minimum strip length. Black rectangles represent the position of each WPT module.

7.5.7 Minimum WPT Length Constraint

The idea behind this constraint definition is that all the WPT Centers belonging to a strip formed by at least two modules, are spaced each other by exactly five nodes. In other words, within that strip, each WPT Center has at least another WPT Center of that strip which belongs to its Distance-5 Neighborhood. Then to guarantee the formation of strips with at least two modules, it is sufficient to constraint each Horizontal/Vertical Center to have at least another Horizontal/Vertical Center among its Distance-5 Horizontal/Vertical Neighbors. Moreover, if a node does not have any Distance-5 Neighbor, it could not be the Center of a WPT in a strip. So, the placement of WPT Centers on nodes without any Distance-5 Neighbors must be prevented.

This new constraint takes the name of "Minimum WPT Length Constraint", and is divided in two formulations: one for Horizontal WPTs, the other for the Vertical ones.

7.5.7.1 Minimum Horizontal WPT Length Constraint

This constraint is imposed to force each node $i \in \mathcal{N}$ which is a Horizontal WPT Center to have at least another Horizontal WPT Center among its Distance-5 Horizontal Neighbors. Moreover, any node $i \in \mathcal{N}$ which has not any Distance-5 Horizontal Neighbors, cannot be a Horizontal WPT Center.

The formal definition of this constraint is:

Any Horizontal or Cross node can be a Horizontal WPT Center if and only if has at least one Horizontal Center belonging to its Distance-5 Horizontal Neighborhood.

In mathematical terms, we can express it as follows:

$\forall i \in \mathcal{N} : cat_i \in \{1,3\}$, it can be $hc_i = 1 \Leftrightarrow |\mathcal{HNN}_i| > 0 \wedge \exists j \in \mathcal{HNN}_i : hc_j = 1$

To mathematically insert this statement in the LP problem, two types of constraints can be formulated for each node $i \in \mathcal{N} : cat_i \in \{1, 3\}$, according to whether it has Distance-5 Horizontal Neighbors of not. First, an inequality constraint must be defined for each of them with at least one Distance-5 Horizontal Neighbor. Its formulation is as follows:

$$hc_i - \sum\nolimits_j hc_j \leq 0 \quad \forall i \in \mathcal{N}, \forall j \in \mathcal{HNN}_i : cat_i \in \{1,3\}, \mathcal{HNN}_i \neq \emptyset \quad (51)$$

On the other side, an equality constraint has to be defined for each of those without any Distance-5 Horizontal Neighbor. The equality formulation is as follows:

$$hc_i = 0 \quad \forall i \in \mathcal{N} : cat_i \in \{1,3\}, \mathcal{HNN}_i = \emptyset \quad (52)$$

7.5.7.2 Minimum Vertical WPT Lenght Constraint

In the same way, this constraint is imposed to force each node $i \in \mathcal{N}$ which is a Vertical WPT Center to have at least another Vertical WPT Center among its Distance-5 Vertical Neighbors. Moreover, any node $i \in \mathcal{N}$ which has not any Distance-5 Vertical Neighbors, cannot be a Vertical WPT Center.

The formal definition of this constraint is:

Any Vertical or Cross node can be a Vertical WPT Center if and only if has at least one Vertical Center belonging to its Distance-5 Vertical Neighborhood.

In mathematical terms, we can define it as follows:

$\forall i \in \mathcal{N} : cat_i \in \{2,3\}$, it can be $vc_i = 1 \iff |\mathcal{VNN}_i| > 0 \land \exists j \in \mathcal{VNN}_i : vc_j = 1$

Again, to mathematically insert this statement in the LP problem, two types of constraints can be formulated for each node $i \in \mathcal{N} : cat_i \in \{2, 3\}$, according to whether it has Distance-5 Vertical Neighbors of not. First, an inequality constraint must be defined for each of them with at least one Distance-5 Vertical Neighbor. Its formulation is as follows:

$$vc_i - \sum_j vc_j \leq 0 \quad \forall i \in \mathcal{N},\ \forall j \in \mathcal{VNN}_i\colon cat_i \in \{2,3\},\ \mathcal{VNN}_i \neq \emptyset \qquad (53)$$

On the other side, an equality constraint has to be defined for each of those without any Distance-5 Vertical Neighbor. Its formulation is as follows:

$$vc_i = 0 \quad \forall i \in \mathcal{N} : cat_i \in \{2,3\}, \mathcal{VNN}_i = \emptyset \qquad (54)$$

7.5.7.3 Summing Up

All the constraints on the correct dynamic WPTs placement have been presented. Using this set of constraints, the optimization algorithm is able to calculate an optimal solution with a WPT Parts/Centers layout which can can be physically installed, and which is compatible with the modelling approach used in this work. Figure 35 shows an example of a possible dynamic WPT layout calculated by the optimization process using the set of constraints defined up to here.

We have now to define a constraint aiming at preventing the placement of static WPTs in bays where its positioning is prohibited by the bay parameters.

7.5.8 Bay WPT Impossibility Constraint

As said in section 4.5, at most one Static WPT can be generically placed in each loading bay $k \in \mathcal{B}$. However, there may be situations in which its placement must be a-priori prohibited. Therefore, this constraint must be imposed to prevent the static WPT placement in bays $k \in \mathcal{B} : bwf_k = 0$.

The formal definition of this constraint is:

If a bay has parameter bwf = 0, then no Static WPT can be placed in it.

This statement can be mathematically inserted in the LP problem by defining an equality constrant for each bay $k \in \mathcal{B} : bwf_k = 0$. The equality should be formulated as:

$$xb_k = 0 \quad \forall k \in \mathcal{B} : bwf_k = 0 \qquad (55)$$

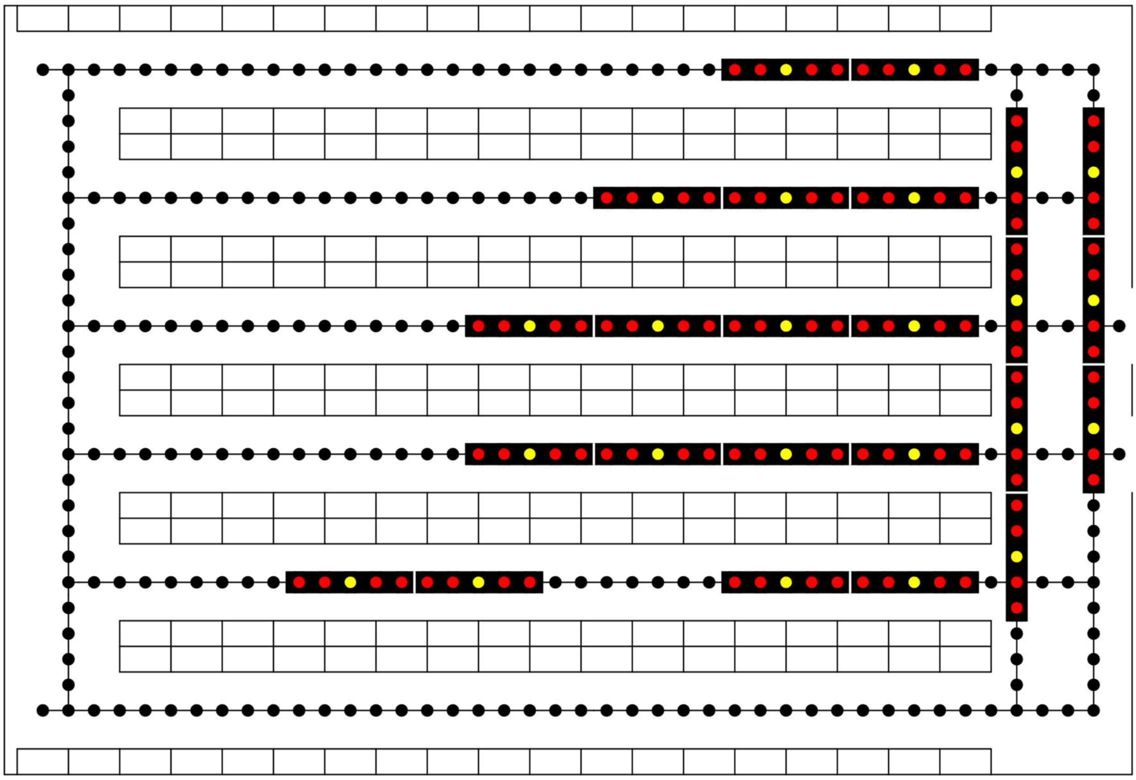

Figure 35: WPT layout, for exemplification purposes only, which can be a solution of the optimization process for the warehouse modelled in Figure 13. Red and yellow nodes indicate, respectively, WPT Parts and Centers. Black rectangles are the dynamic WPTs. Note that, even if they are represented as separate, all the WPT modules belonging to a strip are overlapped in couples.

7.5.9 SoC Constraint

This is the last constraint to be imposed in this optimization problem. It is related to the variation of forklift SoC during the whole working shift (ΔSoC_{shift}), that must be greater or equal than the threshold value $\Delta SoC_{desired}$.

The constraint has been formulated on the basis of the considerations about the Energy Balance made in chapter 6. Its mathematical formulation starts from:

$$\Delta SoC_{shift} \geq \Delta SoC_{desired} \tag{28}$$

which means:

$$\Delta SoC_{shift} - \Delta SoC_{desired} \geq 0 \tag{56}$$

We can write $\Delta SoC_{desired}$ as:

$$\Delta SoC_{desired} = \frac{\Delta E_{shift,desired}}{E_{battery,max}} \tag{57}$$

and, since from equation (26) we have that:

$$\Delta SoC_{shift} = \frac{\Delta E_{shift}}{E_{battery,max}} = \frac{E_{in,shift} - E_{out,shift}}{E_{battery,max}} \tag{58}$$

we can rewrite equation (56) as:

$$E_{in,shift} - E_{out,shift} - \Delta E_{shift,desired} \geq 0 \tag{59}$$

which, put in a form that can be recognized by the solver, becomes:

$$-E_{in,shift} + E_{out,shift} \leq -\Delta E_{shift,desired} \tag{60}$$

and, by considering equation (29), we can write equation (60) as:

$$-E_{in,bays} - E_{in,dynamic} \leq -\Delta E_{shift,desired} - E_{out,shift} + E_{in,breaks} \tag{61}$$

Note that all the terms of this inequality, except $\Delta E_{shift,desired}$ depend on the Total Times, which has been previously computed. The terms $E_{in,dynamic}$ and $E_{in,bays}$ depend on the optimization variables of the problem.

By rewriting equation (61) according to (31) and (32), we can obtain the final form of the constraint:

$$-P_n \cdot \eta_{stat} \cdot ST_{eff} \cdot \sum_{k=1}^{B} (tbrt_k \cdot xb_k) - P_n \cdot \eta_{dyn} \cdot ST_{eff} \cdot \sum_{i=1}^{N} (tnt_i \cdot x_i) \leq - \Delta E_{shift,desired} - E_{out,shift} + E_{in,breaks} \tag{62}$$

Note that $E_{in,breaks}$ and $E_{out,shift}$ have been calculated in, respectively, equation (30) and (33). Note also that the value of $\Delta E_{shift,desired}$ is found, similarly to equation (26), as:

$$\Delta SoC_{desired} = \frac{\Delta E_{shift,desired}}{E_{battery,max}} \Rightarrow \Delta E_{shift,desired} = \Delta SoC_{desired} \cdot E_{battery,max} \tag{63}$$

Remind that $\Delta SoC_{desired}$ is the main responsible for the increase or decrease of the cost of WPT system. Generally, the higher its value, the higher the system cost.

7.6 Optimization Problem

The formulation of the optimization problem to be solved is**:**

Minimize: $\sum_i c_d x_i + \sum_i c_s xb_i$

Subject to:

- $3x_i - 3h_i - 3hc_i - v_i - vc_i = 0 \quad \forall i \in \mathcal{N} : cat_i = 1$ (41)
- $3x_i - h_i - hc_i - 3v_i - 3vc_i = 0 \quad \forall i \in \mathcal{N} : cat_i = 2$ (42)
- $x_i - h_i - hc_i - v_i - vc_i = 0 \quad \forall i \in \mathcal{N} : cat_i = 3$ (43)
- $x_i + h_i + hc_i + v_i + vc_i = 0 \quad \forall i \in \mathcal{N} : cat_i = 4$ (44)
- $\sum_{i=1}^{N} h_i - 4\sum_{i=1}^{N} hc_i = 0 \quad \forall i \in \mathcal{N} : \ horc_i = j\,, \ \ j \in \mathcal{H}$ (45)
- $\sum_{i=1}^{N} v_i - 4\sum_{i=1}^{N} vc_i = 0 \quad \forall i \in \mathcal{N} : \ verc_i = j, \ \ j \in \mathcal{V}$ (46)
- $4hc_i - \sum_j h_j \leq 0 \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{HN}_i : |\mathcal{HN}_i| = 4$ (47)
- $4vc_i - \sum_j v_j \leq 0 \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{VN}_i : |\mathcal{VN}_i| = 4$ (48)
- $h_i - \sum_j hc_j = 0 \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{HN}_i : cat_i \in \{1,3\}$ (49)
- $v_i - \sum_j vc_j = 0 \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{VN}_i : cat_i \in \{2,3\}$ (50)
- $hc_i - \sum_j hc_j \leq 0 \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{HNN}_i : cat_i \in \{1,3\}, \mathcal{HNN}_i \neq \emptyset$ (51)
- $hc_i = 0 \quad \forall i \in \mathcal{N} : \ cat_i \in \{1,3\}, \mathcal{HNN}_i = \emptyset$ (52)
- $vc_i - \sum_j vc_j \leq 0 \quad \forall i \in \mathcal{N}, \ \forall j \in \mathcal{VNN}_i : cat_i \in \{2,3\}, \mathcal{VNN}_i \neq \emptyset$ (53)
- $vc_i = 0 \quad \forall i \in \mathcal{N} : \ cat_i \in \{2,3\}, \mathcal{VNN}_i = \emptyset$ (54)
- $xb_k = 0 \quad \forall k \in \mathcal{B} : bwf_k = 0$ (55)
- $-P_n \cdot \eta_{stat} \cdot ST_{eff} \cdot \sum_{k=1}^{B}(tbrt_k \cdot xb_k) - P_n \cdot \eta_{dyn} \cdot ST_{eff} \cdot \sum_{i=1}^{N}(tnt_i \cdot x_i) \leq$
  $-\Delta E_{shift,desired} - E_{out,shift} + E_{in,breaks}$ (62)
- $x_i, h_i, hc_i, v_i, vc_i, xb_k \in \{0,1\} \quad \forall i, \forall k$

The optimal solution is:

$$f^* = \min\left(\sum_i c_d x_i + \sum_i c_s x b_i\right) \tag{64}$$

subject to all the constraints listed above.

The WPT system cost is equal to f^*:

$$cost_{tot} = f^* \tag{65}$$

Moreover:

- $\sum_i c_d x_i = cost_{dyn}$, namely the cost of all the static WPTs
- $\sum_i c_s x b_i = cost_{stat}$, namely the cost of the all the dynamic WPTs

Chapter 8

Model Verification

The whole model has been described. Some tests are needed in order to understand whether it works properly or not. In particular, we are interested in verifying whether:

1. Total Times calculation procedure is correct;
2. WPTs are placed in a way allowed by the defined constraints and by the input parameters;
3. The optimal solution of the LP problem is a WPT system which is really minimal in terms of cost.

The first point can be easily verified for almost any kind of warehouse, through some hand calculations. The second point just requires to check parameters of the nodes over which WPTs were placed (to detect, for example, if a WPT has been placed on a category 4 node) and to verify if every WPT Part/Center position respects all the constraints. On the contrary, verification of the third point is almost never possible: even if designer experience may allow, relying on the computed Total Times, to estimate the regions where WPTs will be placed, their optimal positions cannot be found by hand. However, there are some peculiar cases, typically simple warehouses with particular nodes and operations parameters, in which the optimal WPT positions can be found without the linear programming minimization problem.

We can take as case scenarios some particular warehouses, with relatively small dimensions and with "unusual" layouts, and we can make some analyses on them, for the purpose of verificating the points reported on the top of the section.

8.1 1st Verification – Constraints check

We can first analyze a warehouse with a very unusual layout and distorted dimensions, built for exemplification purposes only. Its modelled map is reported in Figure 36, 37 and 38.

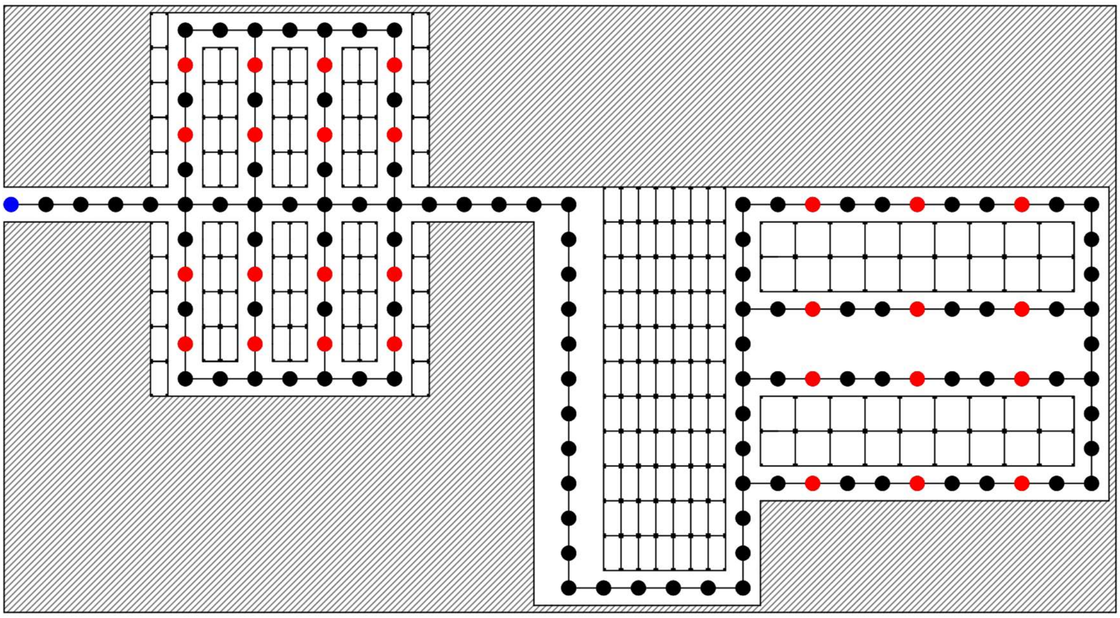

Figure 36: Modelled warehouse map for 1st verification case scenario. Red nodes indicate the Operation Nodes, while blue node is the Bay Node.

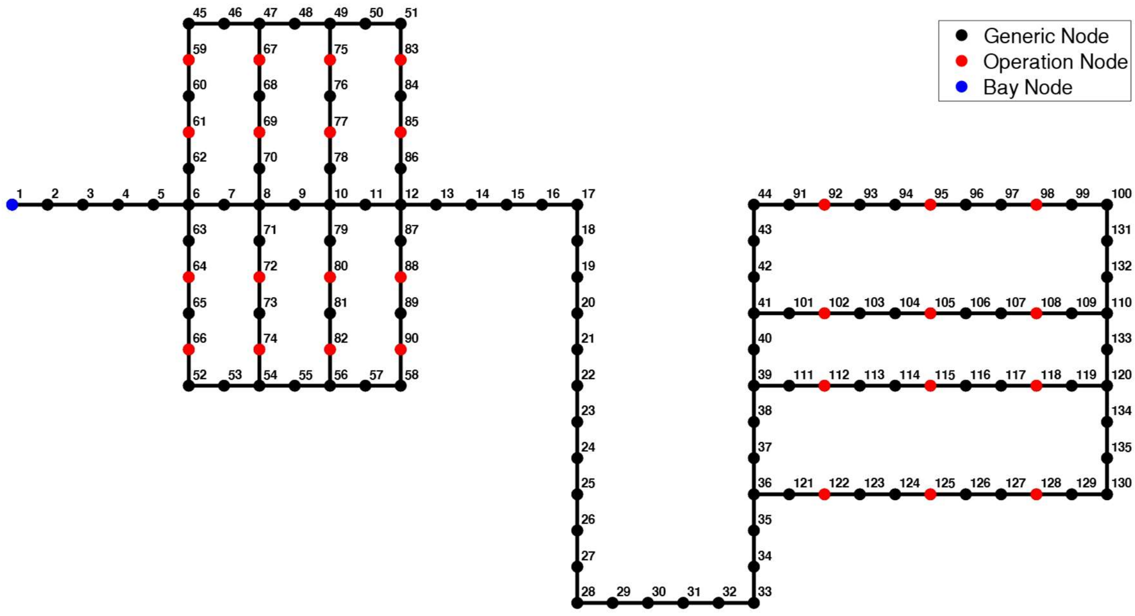

Figure 37: Modelled warehouse map for 1st verification case scenario, built using Matlab. Number over each node indicate its corresponding Node ID.

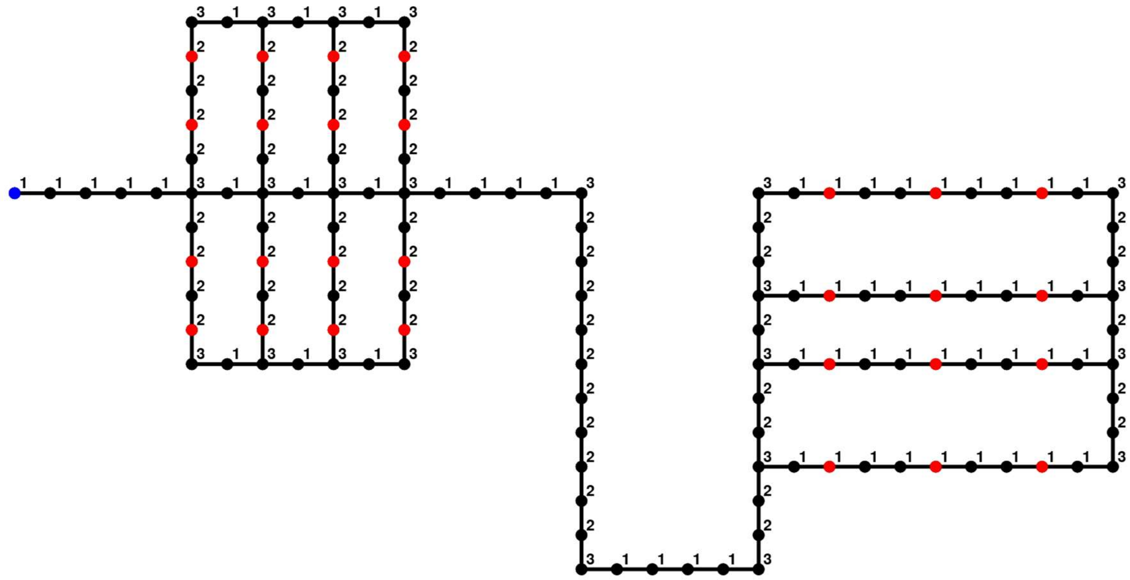

Figure 38: Warehouse map of 1st verification case scenario. Numbers near nodes indicate their Node Category.

Operations to be executed are reported in Table 12.

Operation ID j	on_j	ot_j [s]	Absolute Prob.	p_j	b_j	bt_j [s]	brt_j	inf_j
1	59	20	2	1/22	1	25	0.30	0
2	61	20	2	1/22	1	25	0.30	0
3	64	20	2	1/22	1	25	0.30	0
4	66	20	2	1/22	1	25	0.30	0
5	67	20	2	1/22	1	25	0.30	0
6	69	20	2	1/22	1	25	0.30	0
7	72	20	2	1/22	1	25	0.30	0
8	74	20	2	1/22	1	25	0.30	0
9	75	20	2	1/22	1	25	0.30	0
10	77	20	2	1/22	1	25	0.30	0
11	80	20	2	1/22	1	25	0.30	0
12	82	20	2	1/22	1	25	0.30	0
13	83	20	2	1/22	1	25	0.30	0
14	85	20	2	1/22	1	25	0.30	0
15	88	20	2	1/22	1	25	0.30	0
16	90	20	2	1/22	1	25	0.30	0
17	92	30	1	1/44	1	30	0.30	0
18	95	30	1	1/44	1	30	0.30	0
19	98	30	1	1/44	1	30	0.30	0

20	102	30	1	1/44	1	30	0.30	0
21	105	30	1	1/44	1	30	0.30	0
22	108	30	1	1/44	1	30	0.30	0
23	112	30	1	1/44	1	30	0.30	0
24	115	30	1	1/44	1	30	0.30	0
25	118	30	1	1/44	1	30	0.30	0
26	122	30	1	1/44	1	30	0.30	0
27	125	30	1	1/44	1	30	0.30	0
28	128	30	1	1/44	1	30	0.30	0

Table 12: Operation parameters used in 1st verification case scenario.

Parameters used in this simulation are summarized in Table 13.

Parameter	**Value**
c_d [€]	800
c_s [€]	3000
v_f $[km/h]$	6
$E_{battery,max}$ $[kWh]$	30
$P_{cons,max}$ $[W]$	4300
$P_{cons,bay,op}$ $[W]$	2700.4
$P_{cons,bay,rech}$ $[W]$	258
$P_{cons,node,mov}$ $[W]$	2399.4
$P_{cons,node,op}$ $[W]$	2700.4
P_n $[W]$	4000
η_{stat}	0.90
η_{dyn}	0.87
k_{breaks}	1
$\Delta SoC_{desired}$ [%]	0

Table 13: Parameters used in 1st verification case scenario.

The analysis on it can be divided in two parts:

1. A first optimal WPT layout can be calculated by the algorithm implemented on Matlab, using the data reported above.
2. The parameters of some nodes, in particular the categories, can be changed, in a way such that some of the WPTs positioned in the first part are now in zones which do not allow their placement. Optimization algorithm should be used again to determine a new WPT layout, to see if actually none of them are positioned in zones where the node parameters make that positioning infeasible.

The optimal WPT layout of the first part, so using the node categories depicted in Figure 38, is reported in Figure 39.

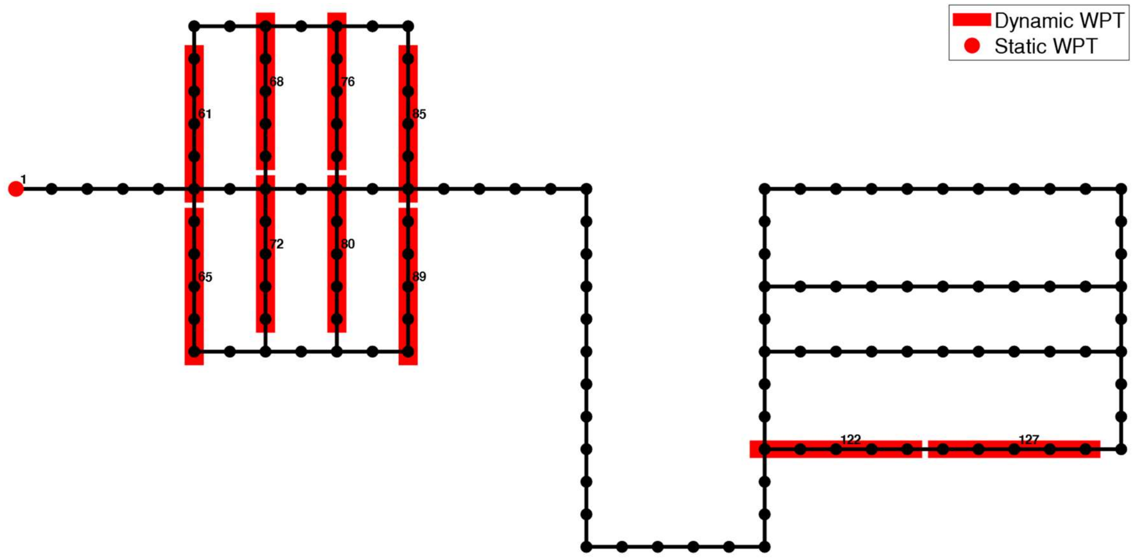

Figure 39: Optimal WPT layout found by Matlab solver in the 1st verification case scenario. Node numbers near each static or dynamic WPT indicate the ID of, respectively, bay nodes or WPT Centers.

The total cost of this first solution is 43000 €, and the obtained ΔSoC_{shift} is 1.776%. The computational time requested by the optimization step is 0.67 s.

Now, referring to Figure 37 and 38, let us assume to modify the parameters of some nodes as follows:

- No WPTs can be placed on nodes 59, 60, 61, 62, 63, 64, 65, 66. So, their category must be set to 4.
- No WPTs can be placed on node 23. Its category becomes 4.
- Nodes 36, 39, 41, 44 can just host a Vertical WPT. Their category must be set to 2.
- Nodes 45, 47, 49, 51 can just host a Horizontal WPT. Their category must be set to 1.

By combining these modifications with the features related to WPT placement and with the constraints of the optimization problem, we can derive a map of the "forbidden zones", namely the set of nodes over which no WPT, of any kind, can be present. Such map is reported in Figure 40:

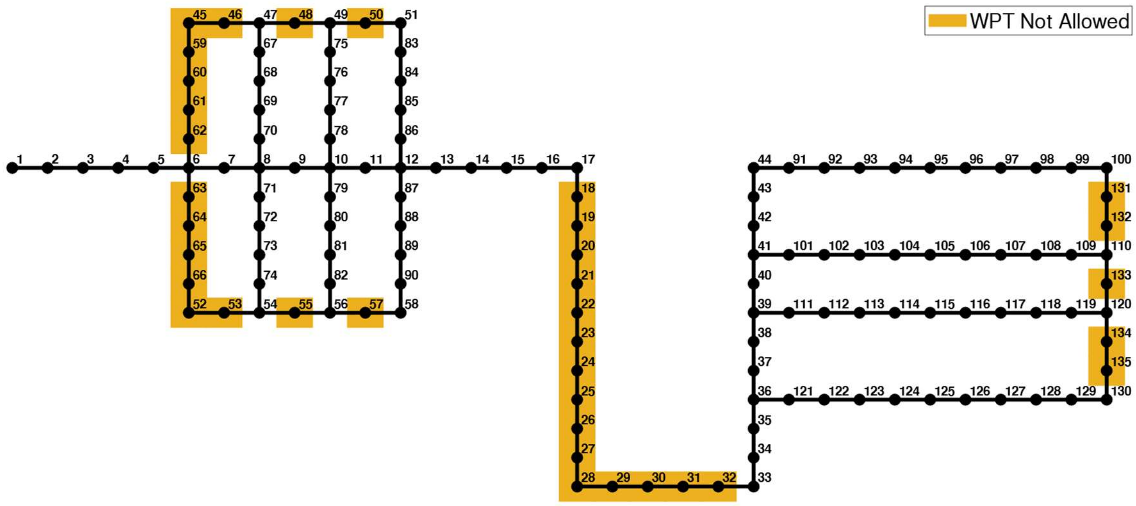

Figure 40: Forbidden zones locations in the 1st verification case scenario, after nodes parameters modification.

By running again the optimization process, we can verify whether there exists an optimal solution able to satisfy all the constraints or not. Note that it may occur that no feasible solution can be found. In that case, the meaning is that forklift SoC constraint cannot be satisfied by any WPT layout. A reduction of the value of $\Delta SoC_{desired}$ would be likely to solve such infeasibility.

In our case, an optimal solutions exists, and is reported in Figure 41.

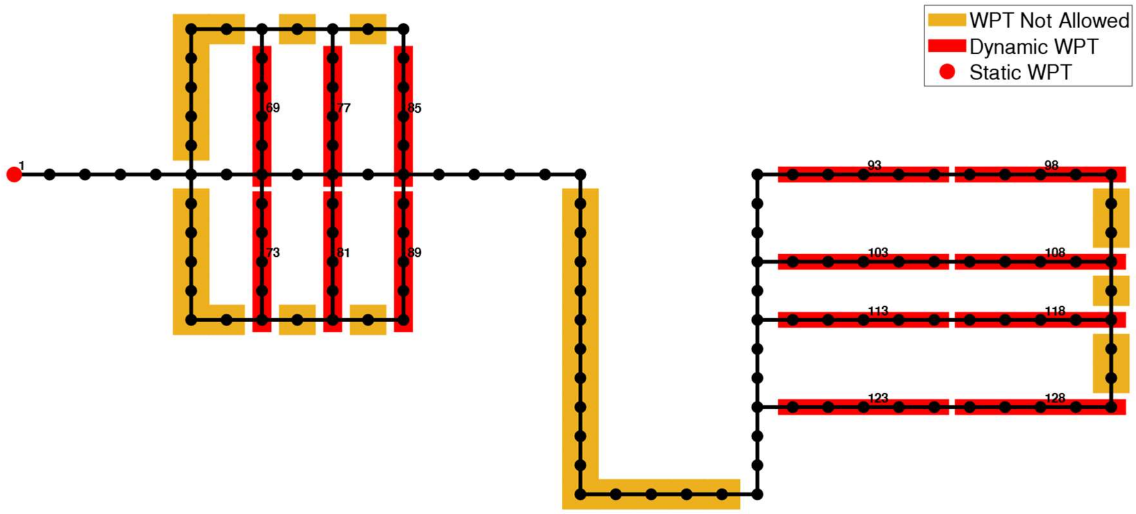

Figure 41: Optimal WPT layout, in relation to the forbidden zones, found by Matlab solver in the 1st verification case scenario, using the modified node categories.

The total cost of this new solution is 59000 €, and the obtained ΔSoC_{shift} is 0.687%. The computational time requested by the optimization step is 0.05 s. The WPT layout do not violate any forbidden zone, and it respects all the constraints. We can then conclude that the algorithm works correctly for what concerning the constraints and the WPT placement in allowed zones only.

8.2 2nd Verification – Optimal WPT positions check

Once having verified that the output of the optimization process respects all the constraints, we have still to verify the correctness of calculated Total Times, and the actual optimality of the WPT system suggested by the algorithm. To do so, let us consider another warehouse, which is modelled as shown in Figure 42, 43 and 44. Note that, again, this warehouse has been designed for verification purposes only.

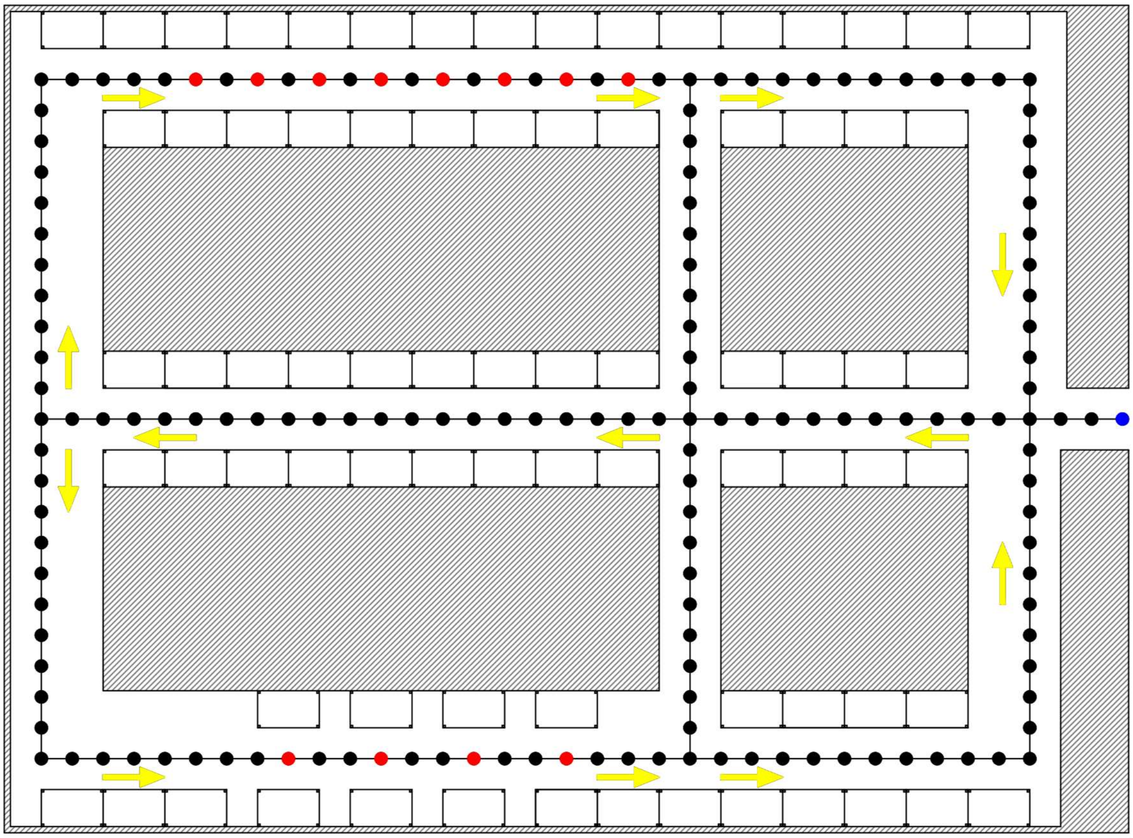

Figure 42: Modelled warehouse map for the 2nd verification case scenario. Red nodes indicate the Operation Nodes, while blue node is the Bay Node.

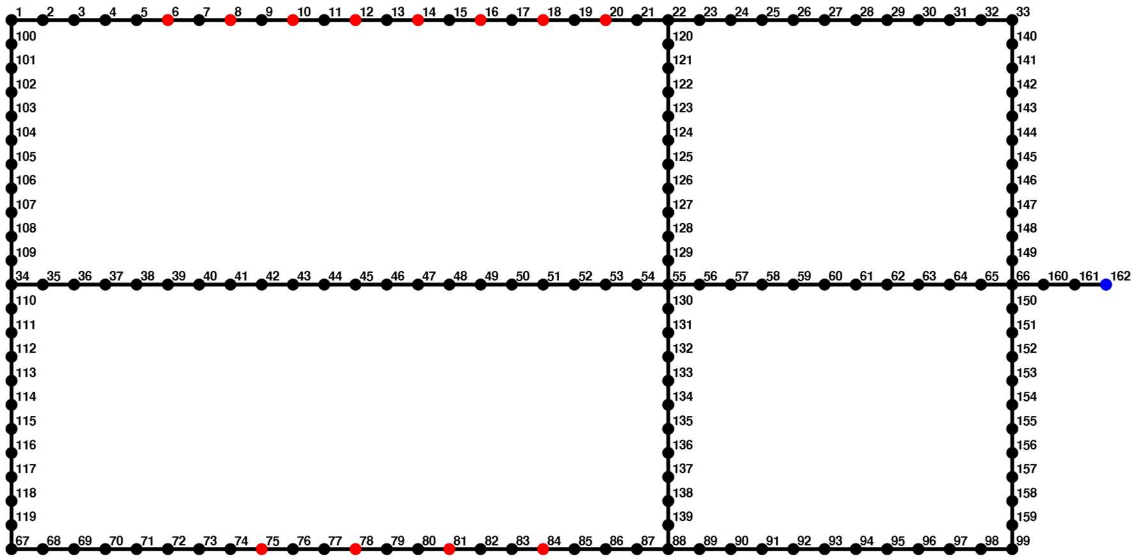

Figure 43: Modelled warehouse map for 2nd verification case scenario, built using Matlab. Number over each node indicate its corresponding Node ID.

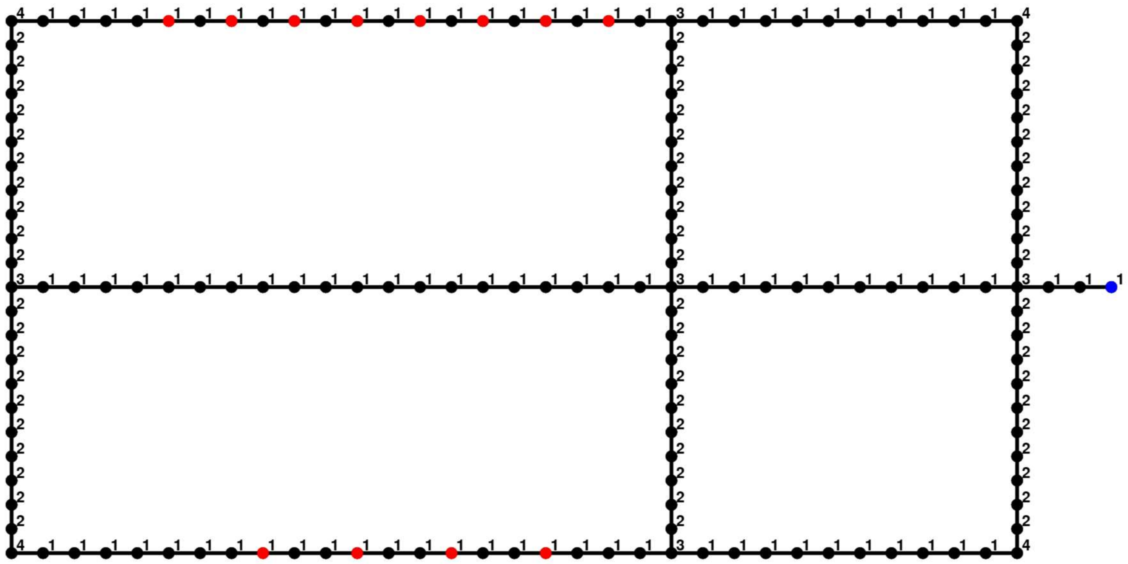

Figure 44: Warehouse map of 2nd verification case scenario. Numbers near nodes indicate their Node Category.

Operations to be executed are reported in Table 14.

Operation ID j	on_j	$ot_j\ [s]$	**Absolute Prob.**	p_j	b_j	$bt_j\ [s]$	brt_j	inf_j
1	6	30	1	0.0625	1	30	0.30	1
2	8	30	1	0.0625	1	30	0.30	1
3	10	30	1	0.0625	1	30	0.30	1
4	12	30	1	0.0625	1	30	0.30	1
5	14	30	1	0.0625	1	30	0.30	1
6	16	30	1	0.0625	1	30	0.30	1
7	18	30	1	0.0625	1	30	0.30	1
8	20	30	1	0.0625	1	30	0.30	1
9	75	30	2	0.1250	1	25	0.30	1
10	78	30	2	0.1250	1	25	0.30	1
11	81	30	2	0.1250	1	25	0.30	1
12	84	30	2	0.1250	1	25	0.30	1

Table 14: Operation Parameters of the 2nd verification case scenario.

Note that, in this case, parameters $inf_j, \forall j \in \mathcal{O}$ have been set to 1 since intermediate nodes, which are not reported here for simplicity, needed to be defined to account for the one-ways shown in Figure 42. Parameters used in this simulation are the same as the ones used in the previous case scenario, and they were reported in Table 13.

The Total Times are shown in Figure 45.

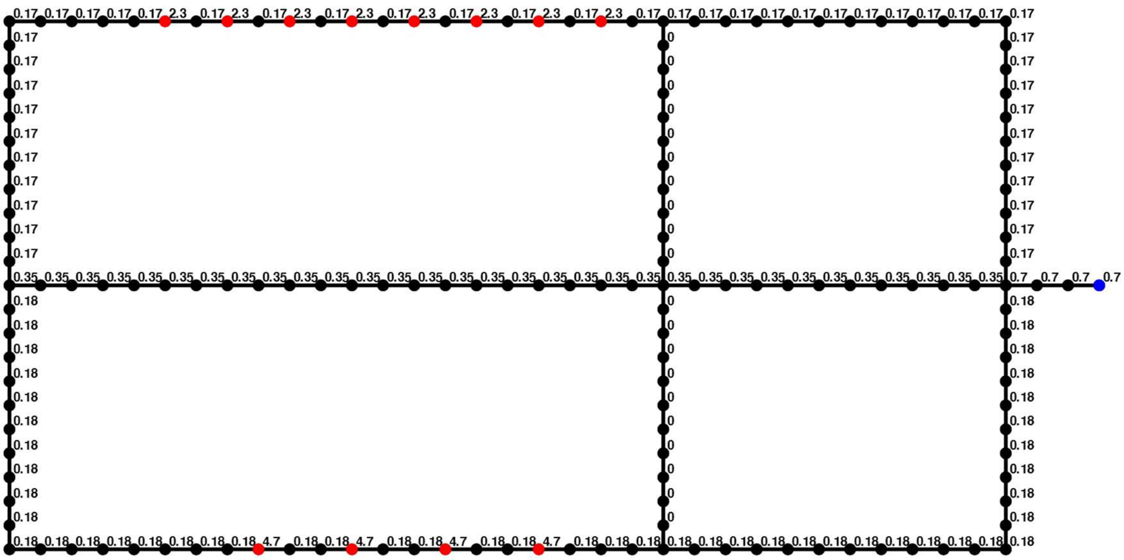

Figure 45: Total Times of the 2nd verification case scenario, reported near the correspondent node.

The sum of all the Total Node Times is 67.86%, Total Bay Time is 32.14% and Total Bay Recharging Time is 9.64%. After having verified their correspondence to the ones calculated by hand (the procedure has not been reported here for simplicity), we can try to analyze them to understand where WPTs should be placed. This warehouse has been designed in a way that there is just one WPT layout which satisfies both constraint requirements and cost optimality. The optimal WPT locations are reported in Figure 46 and Table 15.

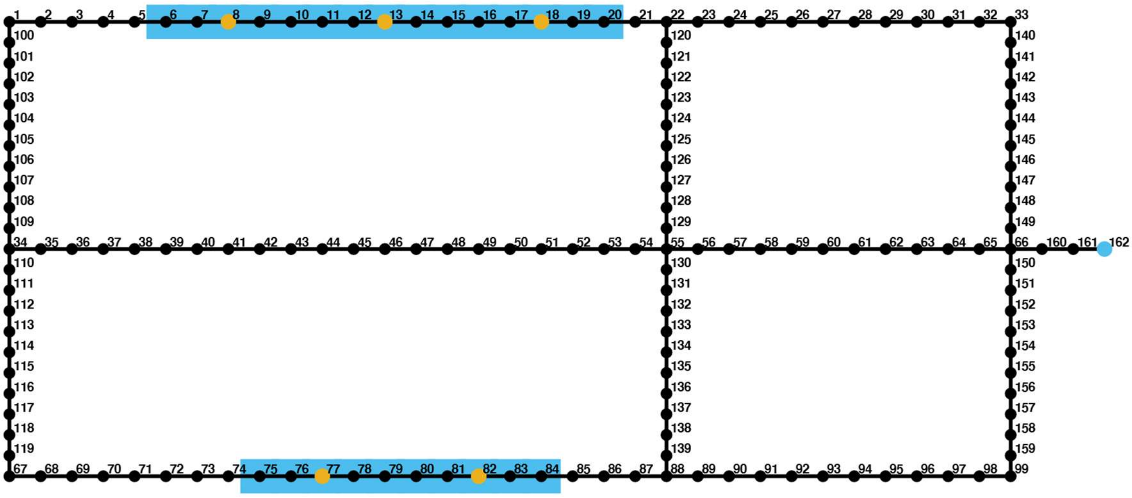

Figure 46: Optimal WPT layout of the 2nd verification case scenario, which should correspond to the one calculated by Matlab as optimal solution of the LP problem. Blue rectangles indicate the dynamic WPT modules, which have as centers the yellow nodes. Blue circles show the positions of the static WPTs.

WPT Number	**Type**	**Center/ Bay ID**	**Parts ID**			
1	HOR	8	6	7	9	10
2	HOR	13	11	12	14	15
3	HOR	18	16	17	19	20
4	HOR	77	75	76	78	79
5	HOR	82	80	81	83	84
6	BAY	162	-	-	-	-

Table 15: List of the WPT features, whose positions are shown in Figure 46.

The optimal system cost is therefore 23000 €. Obtained ΔSoC_{shift} is 0.516%.

Other reasonable WPT layouts, with different costs, have been tested. The results are summarized in Table 16. However, neither of them is able to guarantee the ΔSoC_{shift} requested by the customer.

Solution no.	Horizontal Centers ID	Bay WPT ID	ΔSoC_{shift}	Cost
1	8, 13, 18, 77, 82	1	0.187%	23000 €
2	7, 12, 17, 77, 82	1	-1.154%	23000 €
3	10, 15, 20, 77, 82	1	-1.154%	23000 €
4	13, 18, 62, 77, 82, 160	-	-9.120%	24000 €
5	47, 52, 57, 62, 160	1	-22.451%	23000 €
6	13, 18, 77, 82	1	-5.163%	19000 €
7	8, 13, 18, 77, 82	-	-7.295%	20000 €
8	13, 18, 77, 82	-	-12.974%	16000 €

Table 16: Tests on some WPT layouts in the 2nd verification case scenario. Solution 1 is the optimal one.

We are now interested in verifying if the output of the optimization process is actually the optimal one. Such output is reported in Figure 47.

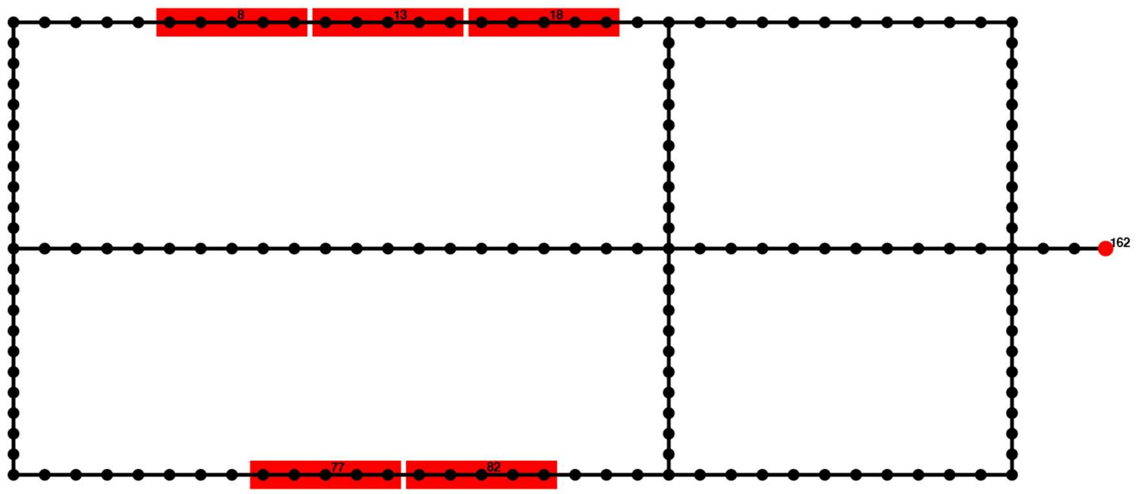

Figure 47: Optimal WPT layout found by Matlab solver in the 2nd verification case scenario. Red circles and rectangles indicate the positions of, respectively, static and dynamic WPT modules.

As we can see, the WPT layout found by the algorithm implemented on Matlab is exactly what forecasted during the analysis made before. Then, the optimization process is, in principle, actually able to find the optimal solution, if exists. Note that computational time requested by it is approximately 1.15 s. On the other hand, computational time requested by the whole process (loading matrices in Matlab, Total Times calculation, LP problem solution, figures production) is approximately 7.5 s.

8.3 3rd Verification – Realistic case scenario

We can test now the methodology in a "realistic" warehouse, to check its correctness and speed in finding an optimal WPT layout in a more real case scenario. The warehouse in analysis, designed following what reported in section 2.3, has a traditional layout, without middle cross aisle, and it is reported in Figure 48. It has 10 picking aisles, each one hosting 40 picking locations per side, resulting in 800 different picking operations in the overall warehouse. Picking locations are areas of 1000x1400 mm, which are suitable for Europallets storage. Operation probabilities associated to each picking location are showed in Figure 49, and were defined according to class-based storage. A single bay is present, which entry is assumed to be located in the middle of the lowest vertical corridor. Note that the proper loading bay area is not shown, for the sake of simplicity. All forklifts must reach each picking location following the shortest path, both from the bay to picking location and vice versa.

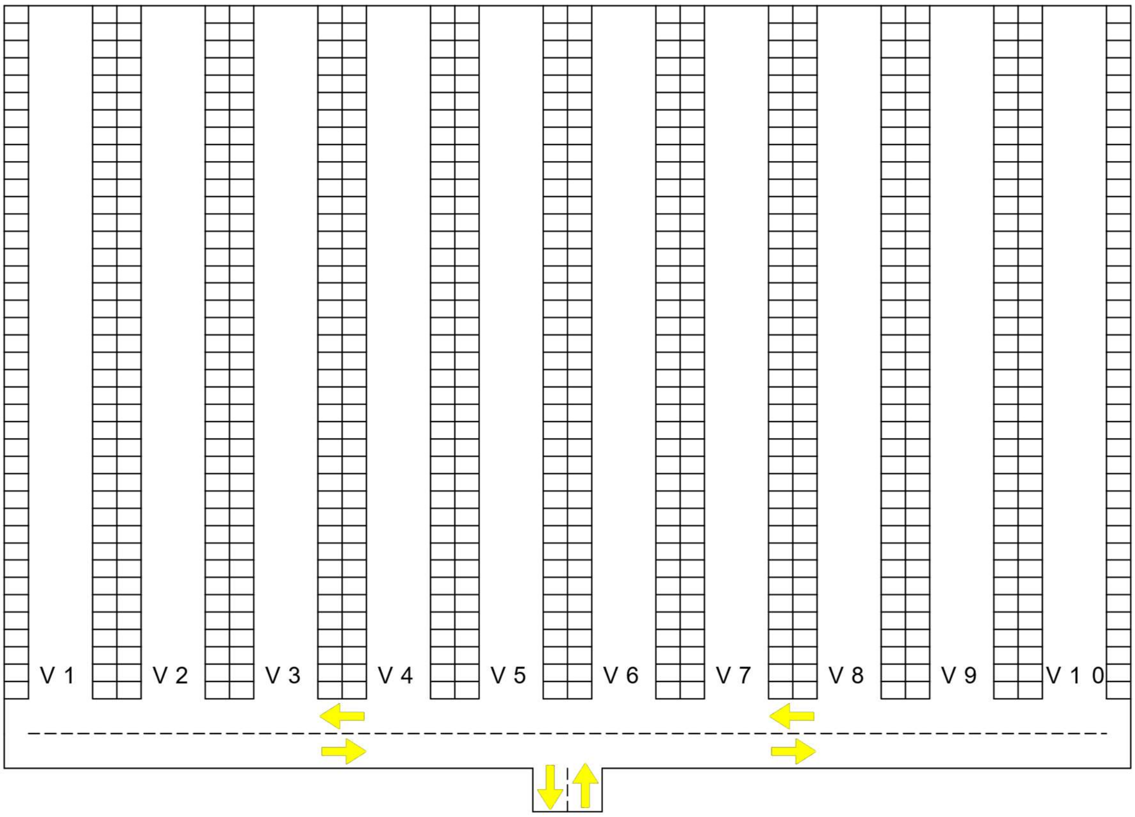

Figure 48: Warehouse map for 3rd verification case scenario.

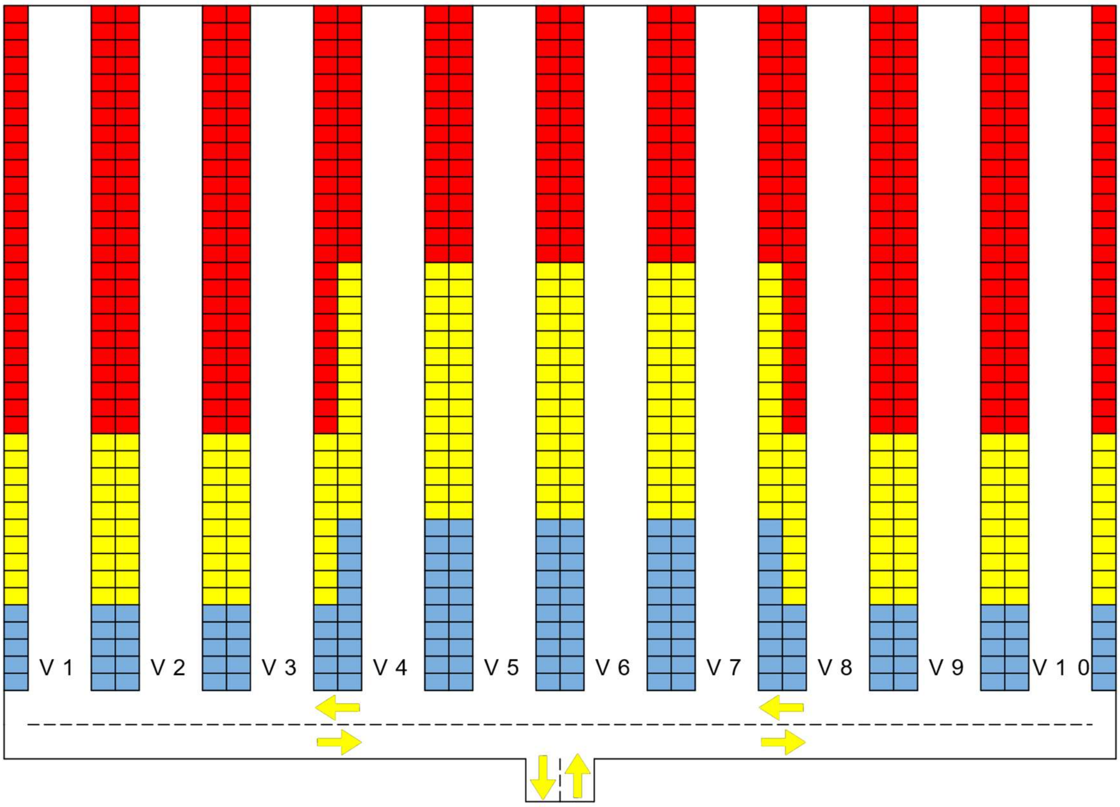

Figure 49: Operation probabilities for different picking locations in the 3rd verification case scenario. Blue locations have a picking probability three times higher than red zones. Yellow zones have picking probability two times higher than red zones.

The warehouse can then be modelled as in Figure 50, using approximately 1100 nodes.

Some operation parameters are reported in Table 17. They are valid for each operation $j \in \mathcal{O}$. Picking probabilities are shown in Figure 49, and all the operations are carried out in the only bay available. Intermediate nodes, not reported here, have been defined to account for the one-way corridors. The generic parameters used in the simulation are the same as the previous case scenarios, and they have been listed in Table 13.

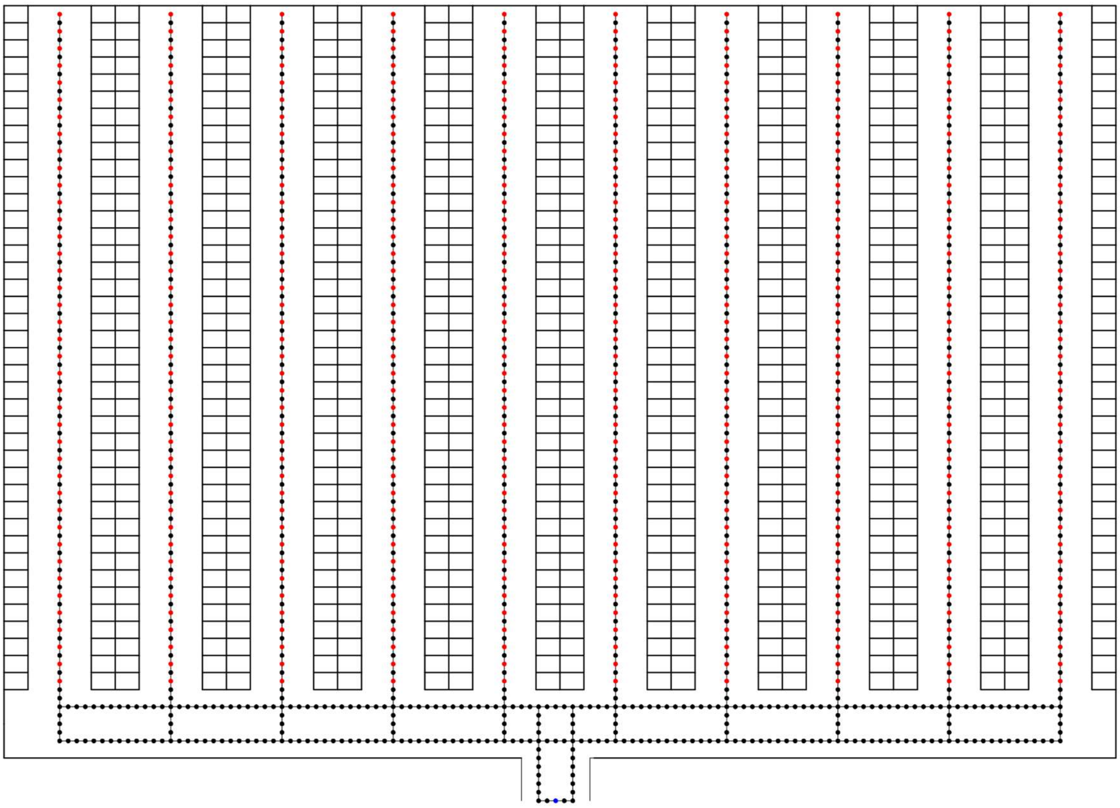

Figure 50: Warehouse modellization for 3rd verification case scenario. Red nodes are the Operation Nodes, while the blue one indicate the Bay Node.

Parameter	Value
ot_j	30 s
bt_j	40 s
brt_j	0.30

Table 17: Operation Parameters for the 3rd verification case scenario.

From a first analysis, being all the operation parameters the same, we expect, in principle, the WPTs to be located mainly in the lower warehouse zone (in the "blue zone" of Figure 49), where not only picking probabilities are higher, but also because forklift must cross those zones while reaching upper parts of the picking aisles. Moreover, since a single bay is present, we expect a static WPT to be placed there.

The output of the optimization process is reported in Figure 51.

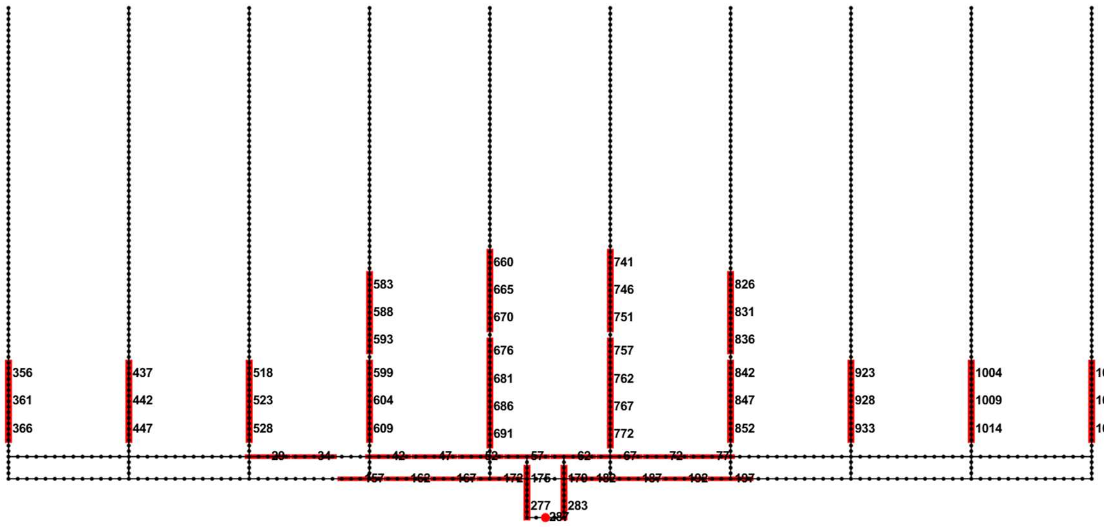

Figure 51: Optimal WPT layout found by Matlab solver in the 3rd verification case scenario.

As we can see, the optimal WPT layout follows our expectations. This layout includes 66 dynamic WPTs, corresponding to 165 m of electrified lanes, and one static WPT, mounted in the bay, for an overall cost of 267000 €. The ΔSoC_{shift} is +0.0583%. Computational time requested to find this solution, including all the modelling steps, was approximately 18 min. Note that, if it is desired to reduce the computational time, heuristic techniques may help in finding a solution, even this one, in a significantly lower amount of time. However, in that case, optimality of the found solution cannot be guaranteed. In this case scenario, heuristics allows to find the same solution in just 16 s. This fact highlights the importance of heuristics in performing some first, rough, estimations of the system cost.

8.4 4th Verification – Real Case Scenario

This last verification aims at verifying the capability of the methodology to be applied to "big" warehouses, modelled using several thousands of nodes. Since the correctness of the algorithm to compute a WPT layout which respects the defined parameters has been verified in the previous analyses, this verification step will focus on the computational time requested, to check if the algorithm is able to solve a big-size problem in a reasonable amount of time.

Basing on the previous verifications we expect that, due to the big warehouse dimensions, a large amount of time may be needed to calculate the Total Times and to build the whole constraints set. Moreover, the linear programming problem may require a lot of time too, being the number of optimization variables particularly large.

The warehouse considered here is reported in Figure 52, and it satisfies the needs of a company which produces tires for motor vehicles. It has rectangular shape, with dimensions equal to 245 m and 183 m. There are 50 one-way picking aisles, with picking locations on both their sides. Other aisles, with two lanes each, are located along the warehouse perimeter. Moreover, two cross aisles are present, traversing the middle of the picking aisles. 16 bays are present, eight of them in the left side of the building, the other eight on the right side. A total of 2102 operations are possible, with different picking probabilities. The whole warehouse is then modelled using 11094 nodes and 11264 edges, and all of them are shown in Figure 52 as, respectively, red dots and red segments. Moreover, that figure shows all the corridor IDs, colored in blue, with a prefix “H” or “V” according to whether they are horizontal or vertical. Note that all the corridors are one-way, and the blue arrows indicate the allowed travel direction.

Picking probabilities are defined according to class-based storage, so that objects with higher picking probability are located closer to loading bays. Four picking probabilities classes have been defined, which are named, respectively, purple, red, yellow and blue class. Objects belonging to red class, yellow class and blue class have, respectively, twice, five times and eight times the probability to be picked than an item belonging to purple class. An approximate distribution of the location of the items belonging to the different classes is reported in Figure 53.

For simplicity, it has been assumed that all the objects located in the same picking aisle, regardless their picking probability, has to be sent to the same loading bay. The loading bay for each operation node are not reported here. All the loading bays are used, and each of them allows the placement of a static WPT on it.

When executing an operation, forklifts travel from bay to operation node and vice versa, following the minimal path, even if with some deviations. Note that forklifts paths are constrained by the one ways of each corridor/lane (not showed here). Recall also that Dijkstra algorithm used here does not allow one-way edges definition. Hence, in order to correctly model the path of each operation, a set of intermediate nodes to be crossed has been defined for each of them, in order to guide the forklift along the one-ways corridors. Items belonging to the same picking aisle have the same set of intermediate nodes, since they are assigned to the same bay. Paths followed by the forklift during each picking operation are not shown here, for the sake of conciseness.

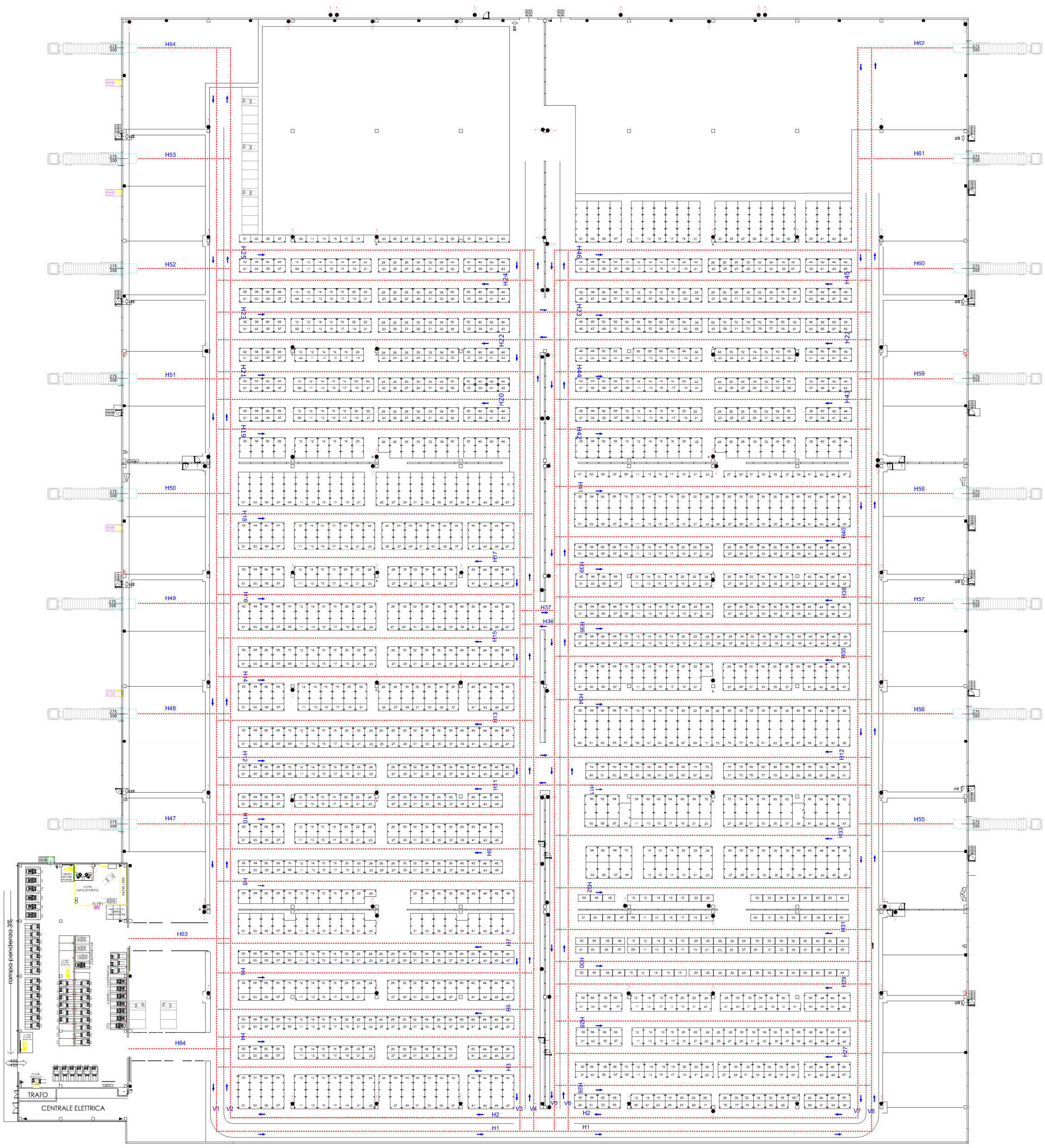

Figure 52: Map of the warehouse, with the modelling entities, used in the 4th verification.

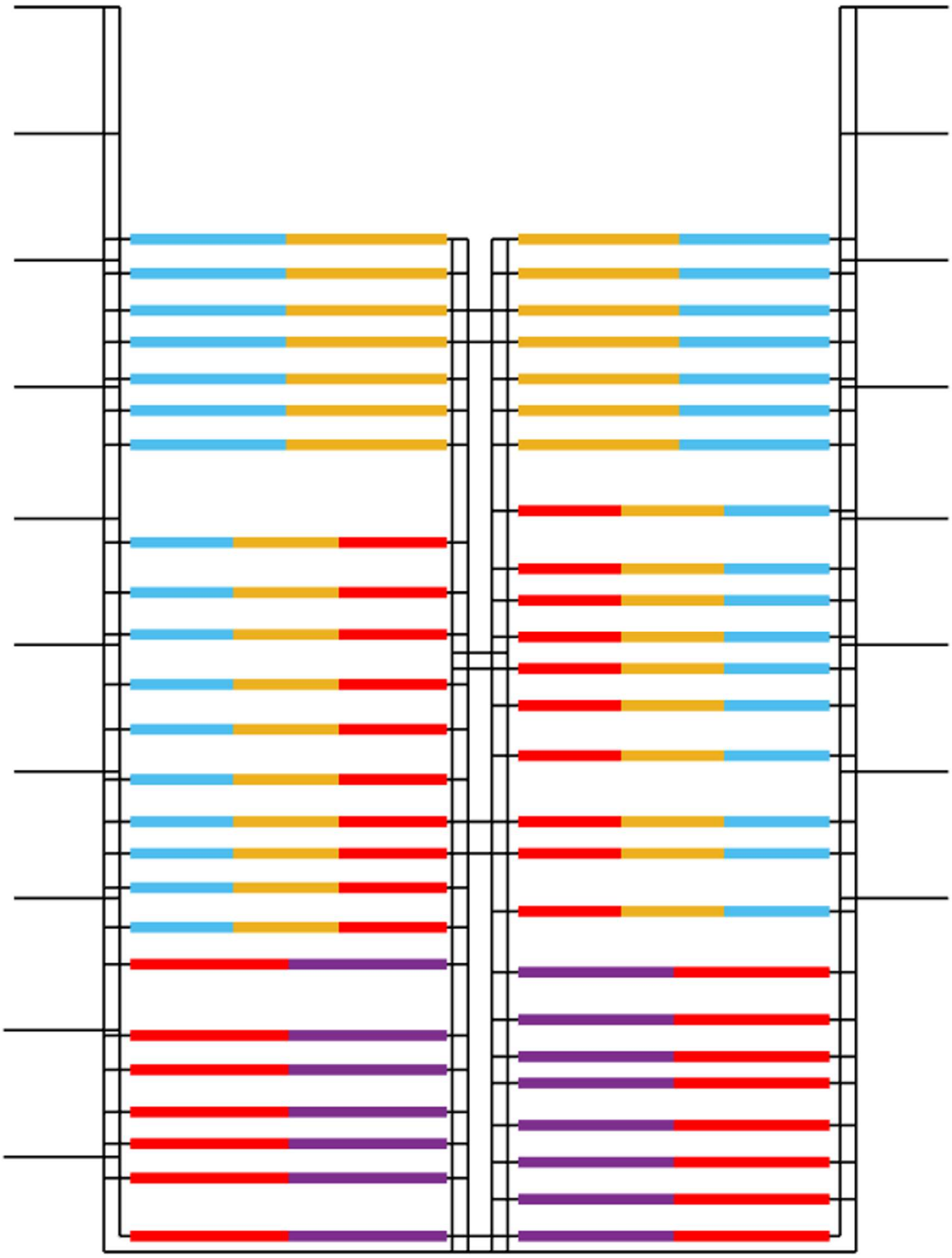

Figure 53: Picking probabilities distribution of the 4th verification. Each picking aisle has been colored according to the probability of the operations to be executed in different parts of them, using the four picking probability classes defined here.

Some limitations in the WPT placement on some nodes have been introduced:

- Cross nodes located along the two-lanes vertical corridors can just host vertical WPTs;
- Cross nodes located along the two-lanes horizontal corridors can host horizontal WPTs only;
- To correctly model the curves located at both ends of the lowest horizontal corridor, nodes located at the extremity of them cannot host any WPT, as like as some nodes around them;
- To correctly model the end of a vertical corridor, nodes located at the upper extremity of each two-lanes vertical corridor cannot host any WPT, as like as some nodes around them;

Operation parameters, valid for each operation $j \in O$, are reported in Table 18. Simulation parameters, identical to the ones used in the past verifications, have been shown in Table 13.

Parameter	Value
ot_j	40 s
bt_j	80 s
brt_j	30%

Table 18: Operation parameters used in the 4th verification.

Being this verification mainly concerned with the optimization time, it is useful to report the characteristics of the computer used for the simulation. They are reported in Table 19.

Model	HP Probook 450 G5
CPU	Intel Core i7-8550U @1.80 GHz
RAM	16 GB
GPU	NVIDIA GeForce 930MX

Table 19: Characteristics of the computer used for the 4th verification.

The WPT layout obtained as output of the algorithm built using Matlab is reported in Figure 54, and the results are summarized in Table 20 and 21. Note, due to the high number of modelling entities, the ID of the nodes with a WPT is not shown in that picture, but just the position of the WPT modules in the warehouse map. The complete list of the nodes with a Part or a Center can be obtained as output of the algorithm, but has not been reported here for the sake of brevity.

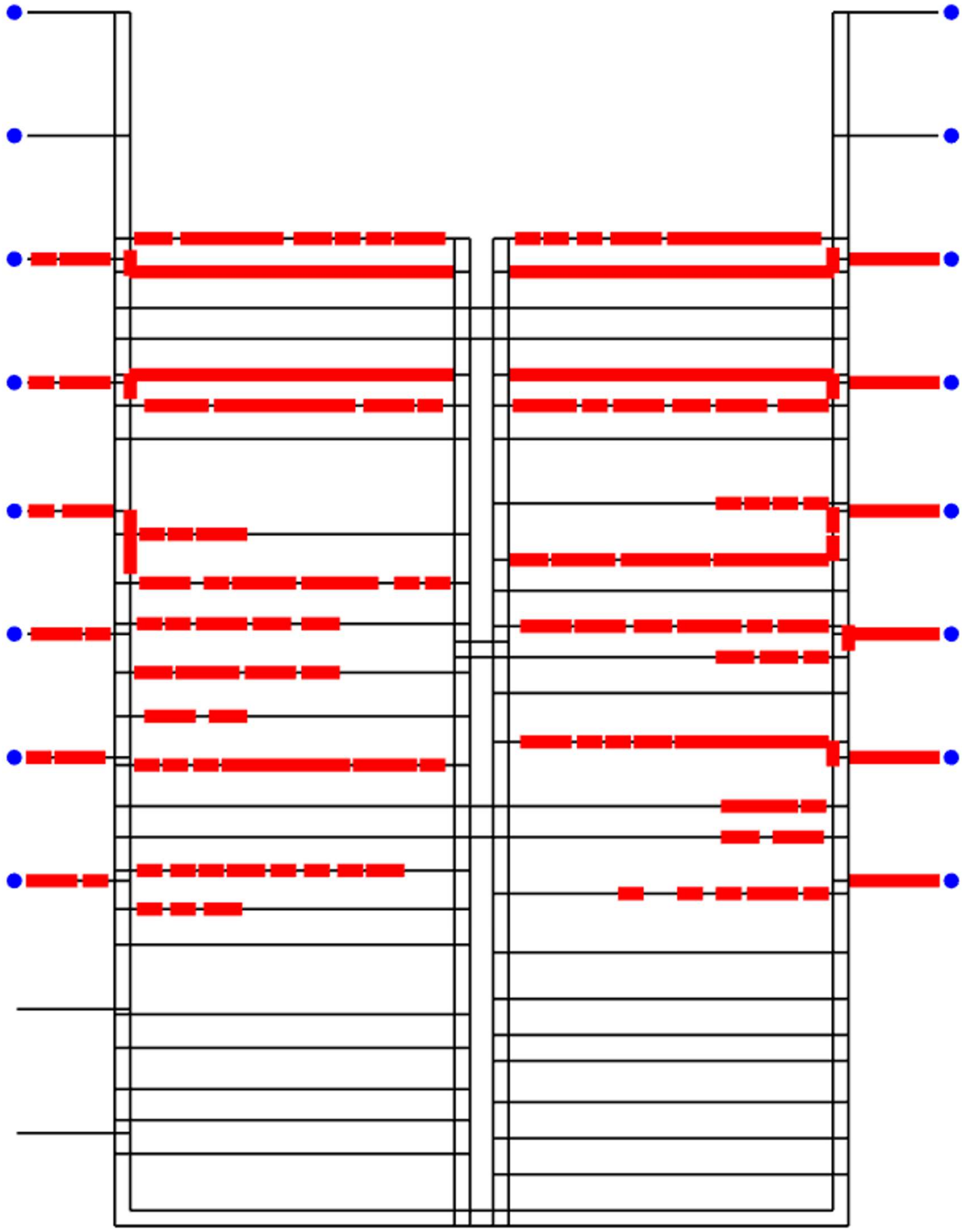

Figure 54: Optimal WPT layout of the 4th verification, calculated by the Matlab algorithm. Red rectangles indicate the WPT modules positions, while blue dots are placed near the entry of a bay which must be equipped with a static WPT.

Dynamic WPT modules	512
Static WPT modules	16
Cost	2096000 €
$\Delta \boldsymbol{SoC}_{shift}$	+0.0115 %

Table 20: Summary of the results of the 4th verification.

Algorithm Part	**Approx. Computation Time requested**
Total Times calculation	62 min
Constraints Building	5 h 45 min
Optimization	8 h
Total	**14 h 50 min**

Table 21: Computational time required by the 4th verification, using the computer with specifications listed in Table 19.

Note that the overall number of dynamic WPT modules placed correspond to 1280 m of electrified lanes.

Through this last verification we have seen that the methodology and the algorithm which has been developed work also in real cases. By analyzing Figure 54, it is clear that WPT modules have been placed in a physically realizable way, while the system capability to calculate the optimal solution has been verified in the second case scenario. Though, the computational time needed to find this solution is really high. Consider that a case like this can be seen as a "worst case scenario", due to the high number of nodes, edges and operations and, generally, with such a number of operations it is not adviceable to calculate the Total Times by using the probabilistic procedure described here. The best choice is to estimate them using the position sensors solution which has been described in section 5.4 and, considering that such a solution requires some months of forklifts position monitoring, the time spent for calculating the optimal solution would result to be practically neglectable with respect to the monitoring time, making the whole developed methodology to be efficiently applicable to every kind of warehouse.

Chapter 9

Conclusions

9.1 Analysis of the results

The validation section has proven the correct functioning of the methodology and its wide range of application. Though the actual optimality of the algorithm solution cannot be always demonstrated, the generated WPT positions, both static and dynamic, fully respect all the assumptions and all the modelling concepts expressed up to now. The fact that each WPT must be located on multiple consecutive nodes – in this case, five – gives a strong flexibility to the calculation of the optimal charging system to be installed on the warehouse in analysis. Moreover, the warehouse modelling procedure results not to be excessively time consuming, and it allows to provide a detailed description of all the operations. It is worth to notice that the whole warehouse modelling data can be generated on Microsoft Excel, as like as any other generic spreadsheet, which can be afterwards read and loaded by Matlab to be finally handled by the optimization process.

Some relevant considerations to be done after having analyzed the results, coming not only from the four validation studies, but also from all the other studies on warehouses made while developing the methodology, are as follows:

- WPTs are always preferentially placed on zones whose node total times sum is higher than others, though taking into account all the limitations on the WPT placement, as like as nodes with category 4. This can be clearly seen in the 2nd validation, where total times are explicitly reported, and also in the 3rd one where, even if total times are not reported, the warehouse layout may suggest the more utilized areas.
- In particular, under the assumption made about the fact that a forklift may be charged for all the time it is executing an operation over a node equipped with a dynamic module, the WPT modules are mostly placed in zones dense of operation nodes with high operation probabilities. The reason is that, in those areas, total times are higher, due to the operation time spent there by the forklifts when executing the operations. If not there, the other optimal WPT locations are, generally, the zones near the mostly utilized bays. This occurs especially if the ratio between the number of different operations and the number of bays is in the order of the

hundreds, as happens in the 4th validation case scenario, and it is even more evident in a case where there is a single bay node and a large number of different operations in the warehouse, like in the 3rd validation.
- Both the warehouse modelling procedure and the optimization algorithm are suitable for real case scenarios, regardless their dimensions, as demonstrated in 4th validation. However, the larger the warehouse, the more complex the size of the optimization problem typically will be and, consequently, the time it takes for obtaining the optimal solution, if any.

9.2 Future Improvements

In this section, a list of the possible improvements which could be applied in the future to the described methodology is provided. These improvements are expected to solve some critical points about what has been presented in this work, or to simplify the optimization algorithm itself in order to reduce the computation time that is currently needed in case of a large problem size.

Some of these improvements are:

- Add the possibility of modelling aisles/lanes oriented in a non-carthesian way, so to make the model suitable for modelling all the warehouse layouts reported in section 2.3.1, in particular in Figure 9 and 10.
- Consider to introduce a parameter, or a set of parameters, which accounts for the eventuality that, during a picking/storage operation, the forklift charging by an eventual dynamic WPT placed over the operation node would not be possible, or would just be partially possible.
- In this work, the effects of multiple forklifts traveling in the warehouse, and their effects on traffic, has not been considered. This improvement can be introduced so to let a more realistic analysis of the warehouse forklift movements.
- For the same reason, consider to introduce a parameter which allows the optimization process to install more than one static WPT in each bay, according to the bay utilization and the average number of forklifts in it at any time.
- In some cases, it could be useful to define some areas, in the modelled warehouse and outside the bays, where static WPTs can be placed directly by the optimization process, like a forklift parking area. This improvement may remove the assumption saying that, during the breaks, forklifts can be charged exclusively on static WPTs located in bays, and the effective break time they are charging is regulated by parameter k_{breaks}.
- Consider the effects of accelerations, decelerations and curves on the forklift instantaneous speed, instead of simply defining the average forklift speed v_f, so to improve the calculation of the time spent by the forklift on each node, during each operation.

- Add the possibility to choose between different optimization criteria, such as to maximize the ΔSoC_{shift} statistically provided by the wireless charging system, given a certain cost the customer is willing to spend.

Another interesting aspect which could be considered is the formulation of a “hybrid” optimization problem, in which some fictitious costs are assigned to situations which could not be properly optimal under some technical aspects point of view. One of them could be the placement of several, separate, WPT strips (though respecting the minimal allowed length of the strip defined in section 7.5.7) which can cause more damages to forklift batteries with respect to longer charging strips, due to the more frequent power supply discontinuities occurring whenever the forklift enters and exits from the dynamic WPT coverage. Moreover, it could be interesting to compare the dynamic wireless charging system described here with a “classic” charging approach using the technologies reported in section 2.1.2., by considering the costs of the system, which are generally higher if using the dynamic charging, and the unavoidable machine downtime periods of the “traditional” charging approach.

However, the strongest and most effective improvement which can be applied is the complete elimination of the probabilistic approach. The problems and the eventual solutions relative to it have been initially described in section 5.4. With reference to them, we can conclude saying that the proposed solution, in which total times are estimated using some position sensors mounted on the forklifts, would be really able to let the optimization process to determine a WPT layout which accurately respects the customer energetic requirements. To achieve this goal, the forklifts positions must be analyzed for long enough time intervals, and in different periods of the year, so to reflect potentially high variabilities of the different warehouse areas usage according to the different months. By combining the accuracy of computing total times in this way with the capability of the model to be applied also to biggest warehouses, as demonstrated in 4th validation, it would indeed be possible to obtain a powerful tool, capable to be applied to almost whatever real case scenario, and which produces a result able to fully satisfy customer requirements.

REFERENCES

[1] European Commission (2011) A Roadmap for moving to a competitive low carbon economy in 2050. European Commission, Brussels.

[2] Shukla, Akanksha, Kusum Verma, and Rajesh Kumar. «Consumer Perspective Based Placement of Electric Vehicle Charging Stations by Clustering Techniques». In *2016 National Power Systems Conference (NPSC)*, 1–6. Bhubaneswar, India: IEEE, 2016. https://doi.org/10.1109/NPSC.2016.7858946.

[3] Lee, Seokgi, Hyun Woo Jeon, Mona Issabakhsh, and Ahmad Ebrahimi. «An Electric Forklift Routing Problem with Battery Charging and Energy Penalty Constraints». *Journal of Intelligent Manufacturing*, March 31, 2021. https://doi.org/10.1007/s10845-021-01763-6.

[4] Navigant Consulting, Inc. (2016). Retrieved May 16, 2019, from https://www.navigantresearch.com/news-and-views/salesof-advanced-electric-forkl ifts-in-north-america-are-expected-to-exceed-47000-by-2025.

[5] Paul, Théophile, Tedjani Mesbahi, Sylvain Durand, Damien Flieller, and Wilfried Uhring. «Sizing of Lithium-Ion Battery/Supercapacitor Hybrid Energy Storage System for Forklift Vehicle». *Energies* 13, n. 17 (September 1, 2020): 4518. https://doi.org/10.3390/en13174518.

[6] Cavallo, Flavio, Paolo Guglielmi, and Vincenzo Cirimele. «Dynamic Wireless Charging», s.d., 7.

[7] Alshaebi, Abdulhameed, Husam Daoud, Joseph Weiss, and Sang Won Yoon. «Evaluation of Different Forklift Battery Systems Using Statistical Analysis and Discrete Event Simulation», s.d., 7.

[8] Lazzeroni, Paolo, Vincenzo Cirimele, and Aldo Canova. «Economic and Environmental Sustainability of Dynamic Wireless Power Transfer for Electric Vehicles Supporting Reduction of Local Air Pollutant Emissions». *Renewable and Sustainable Energy Reviews* 138 (March 2021): 110537. https://doi.org/10.1016/j.rser.2020.110537.

[9] Cirimele, Vincenzo, Fabio Freschi, and IEEE Sm. «Inductive Power Transfer for Automotive Applications: State-of-the-Art and Future Trends», 2016, 8.

[10] Jawad, Aqeel Mahmood, Rosdiadee Nordin, Sadik Kamel Gharghan, Haider Mahmood Jawad, and Mahamod Ismail. «Opportunities and Challenges for Near-Field Wireless Power Transfer: A Review». *Energies* 10, n. 7 (July 18, 2017): 1022. https://doi.org/10.3390/en10071022.

[11] Shoki, Hiroki. «Issues and Initiatives for Practical Deployment of Wireless Power Transfer Technologies in Japan». *Proceedings of the IEEE* 101, n. 6 (June 2013): 1312–20. https://doi.org/10.1109/JPROC.2013.2248051.

[12] Khan, Zadid, Sakib Mahmud Khan, Mashrur Chowdhury, Ilya Safro, and Hayato Ushijima-Mwesigwa. «Wireless Charging Utility Maximization and Intersection

Control Delay Minimization Framework for Electric Vehicles». *Computer-Aided Civil and Infrastructure Engineering* 34, n. 7 (July 2019): 547–68. https://doi.org/10.1111/mice.12439.
[13] Egbue, Ona, and Suzanna Long. «Barriers to Widespread Adoption of Electric Vehicles: An Analysis of Consumer Attitudes and Perceptions». *Energy Policy* 48 (September 2012): 717–29. https://doi.org/10.1016/j.enpol.2012.06.009.
[14] Fisher, Taylor M., Kathleen Blair Farley, Yabiao Gao, Hua Bai, and Zion Tsz Ho Tse. «Electric Vehicle Wireless Charging Technology: A State-of-the-Art Review of Magnetic Coupling Systems». *Wireless Power Transfer* 1, n. 2 (September 2014): 87–96. https://doi.org/10.1017/wpt.2014.8.
[15] ICNIRP GuidelinesforLimitingExposuretoTime-varyingElectric.Magnetic and electromagnetic fields (1Hzto100kHz).HealthPhys2010;99(6):818–36.
[16] ICNIRP GuidelinesforLimitingExposuretoTime-varyingElectric.Magnetic and electromagnetic fields (upto300GHz).HealthPhys1998;74(4):494–522.
[17] IEEEStandardforSafetyLevelswithRespecttoHumanExposuretoRadioFrequencyElectromagneticFields,3kHzto300GHz.IEEEStdC95.1™-2005, April; 2006.
[18] Barman, Surajit Das, Ahmed Wasif Reza, Narendra Kumar, Md. Ershadul Karim, and Abu Bakar Munir. «Wireless Powering by Magnetic Resonant Coupling: Recent Trends in Wireless Power Transfer System and Its Applications». *Renewable and Sustainable Energy Reviews* 51 (November 2015): 1525–52. https://doi.org/10.1016/j.rser.2015.07.031.
[19] Helber, Stefan, Justine Broihan, Young Jang, Peter Hecker, and Thomas Feuerle. «Location Planning for Dynamic Wireless Charging Systems for Electric Airport Passenger Buses». *Energies* 11, n. 2 (January 23, 2018): 258. https://doi.org/10.3390/en11020258.
[20] Cirimele, Vincenzo, Fabio Freschi, and Paolo Guglielmi. «Wireless Power Transfer Structure Design for Electric Vehicle in Charge While Driving». In *2014 International Conference on Electrical Machines (ICEM)*, 2461–67. Berlin, Germany: IEEE, 2014. https://doi.org/10.1109/ICELMACH.2014.6960532.
[21] Chen, Zhibin, Wei Liu, and Yafeng Yin. «Deployment of Stationary and Dynamic Charging Infrastructure for Electric Vehicles along Traffic Corridors». *Transportation Research Part C: Emerging Technologies* 77 (April 2017): 185–206. https://doi.org/10.1016/j.trc.2017.01.021.
[22] Fuller, Micah. «Wireless Charging in California: Range, Recharge, and Vehicle Electrification». *Transportation Research Part C: Emerging Technologies* 67 (June 2016): 343–56. https://doi.org/10.1016/j.trc.2016.02.013.
[23] Churchman, C.W., Ackoff, R.L., Arnoff, E.L. (1957). «Introduction to Operations Research». John Wiley & Sons, New York.
[24] Winston, W.L., 1994. «Operations Research: Applications and Algorithms», third edition. Duxbury Press, Belmont, California.

[25] Altay, Nezih, and Walter G. Green. «OR/MS Research in Disaster Operations Management». *European Journal of Operational Research* 175, n. 1 (November 2006): 475–93. https://doi.org/10.1016/j.ejor.2005.05.016.
[26] Lee, Jihyun Rachel (2014), «Operations Research», Mathematics Honors Theses, Paper 4.
[27] C. W. Churchman, R. L. Ackoff and E. L. Arnoff (1960), «Introduction to Operations Research». Wiley, New York.
[28] R. N. Woolley & M. Pidd (1981), «Problem Structuring — A Literature Review», *Journal of the Operational Research Society*, 32:3, 197-206, DOI: 10.1057/jors.1981.42.
[29] Carter, Michael, Camille C Price, and Ghaith Rabadi. «Operations Research», s.d., 471.
[30] Bazaraa, M. S., Jarvis, J. J., and Sherali, H. D., «Linear Programming and Network Flows», John Wiley & Sons, Inc., fourth edition, 2010.
[31] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C., «Introduction to Algorithms», MIT press, third edition, 2009.
[32] Della Croce F., «Mixed Integer Linear Programming Models for Combinatorial Optimization Problems». In *Concepts of Combinatorial Optimization*, edited by Vangelis Th. Paschos. Wiley, 2010. https://doi-org.ezproxy.biblio.polito.it/10.1002/9781118600245.ch5
[33] https://www.mathworks.com/help/optim/ug/linprog.html#d123e106246.
[34] Floudas, Christodoulos A., and Xiaoxia Lin. «Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications». *Annals of Operations Research* 139, n. 1 (October 2005): 131–62. https://doi.org/10.1007/s10479-005-3446-x.
[35] Richards, A., and J.P. How. «Aircraft Trajectory Planning with Collision Avoidance Using Mixed Integer Linear Programming». In *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, 1936–41 vol.3. Anchorage, AK, USA: IEEE, 2002. https://doi.org/10.1109/ACC.2002.1023918.
[36] Pohl, Letitia M., Russell D. Meller, and Kevin R. Gue. «An Analysis of Dual-Command Operations in Common Warehouse Designs». *Transportation Research Part E: Logistics and Transportation Review* 45, n. 3 (May 2009): 367–79. https://doi.org/10.1016/j.tre.2008.09.010.
[37] Karasek, Jan. «An Overview of Warehouse Optimization». *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems* 2, n. 3 (November 22, 2013). https://doi.org/10.11601/ijates.v2i3.61.
[38] Gu, Jinxiang, Marc Goetschalckx, and Leon F. McGinnis. «Research on Warehouse Design and Performance Evaluation: A Comprehensive Review». *European Journal of Operational Research* 203, n. 3 (June 2010): 539–49. https://doi.org/10.1016/j.ejor.2009.07.031.

[39] Dukic, Goran, and Tihomir Opetuk. «Warehouse Layouts». In *Warehousing in the Global Supply Chain*, edited by Riccardo Manzini, 55–69. London: Springer London, 2012. https://doi.org/10.1007/978-1-4471-2274-6_3.
[40] Roodbergen, Kees Jan, and René de Koster. «Routing Order Pickers in a Warehouse with a Middle Aisle». *European Journal of Operational Research* 133, n. 1 (August 2001): 32–43. https://doi.org/10.1016/S0377-2217(00)00177-6.
[41] Meller, Russell D, and Kevin R Gue. «The Application of New Aisle Designs for Unit-Load Warehouses», 2009, 8.
[42] Dijkstra, E. W. «A Note on Two Problems in Connexion with Graphs». *Numerische Mathematik* 1, n. 1 (December 1959): 269–71. https://doi.org/10.1007/BF01386390.
[43] Deng, Yong, Yuxin Chen, Yajuan Zhang, and Sankaran Mahadevan. «Fuzzy Dijkstra Algorithm for Shortest Path Problem under Uncertain Environment». *Applied Soft Computing* 12, n. 3 (March 2012): 1231–37. https://doi.org/10.1016/j.asoc.2011.11.011.
[44] Barbehenn, M. «A Note on the Complexity of Dijkstra's Algorithm for Graphs with Weighted Vertices». *IEEE Transactions on Computers* 47, n. 2 (February 1998): 263. https://doi.org/10.1109/12.663776.
[45] Javaid, Muhammad Adeel. «Understanding Dijkstra Algorithm». *SSRN Electronic Journal*, 2013. https://doi.org/10.2139/ssrn.2340905.
[46] Diestel, Reinhard. «Graph Theory». 2nd ed. *Graduate Texts in Mathematics* 173. New York: Springer, 2000.
[47] West, Douglas B. «Introduction to Graph Theory», second edition. Pearson, 2001.
[48] Courtesy of Enermove S.r.l.