UNIVERSIDADE DO ALGARVE

Monitoring of Urban Transportation Networks Using Wireless Sensor Networks

Nuno Miguel G. M. Carvalho

Master Thesis in Computer Science Engineering

Work done under the supervision of: Prof. Noélia Correia

2015

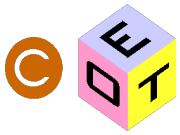
Statement of Originality

Monitoring of Urban Transportation Networks Using Wireless Sensor Networks

Statement of authorship: The work presented in this thesis is, to the best of my knowledge and belief, original, except as acknowledged in the text. The material has not been submitted, either in whole or in part, for a degree at this or any other university.

Candidate:

(Nuno Miguel G. M. Carvalho)



NETWORKING

Work done at Research Center of Electronics Optoelectronics and Telecommunications (CEOT)

Abstract

With today's technologies, constant growth of people in major cities and the permanent concern with our natural resources, such as energy, it is inevitable that we look for ways to improve our urban transportation networks. One way of improving transportation networks is to make an efficient vehicle routing. In this thesis vehicle routing with backup provisioning, using wireless sensor technologies, is proposed. We start by collecting the entry and exit of people inside urban transportations, which will give us a view of fleet load over time, using wireless sensor technologies. For this purpose a monitoring software was developed. Such data gathering and monitoring tool will allow data analysis in real time and, according to information extracted, to propose solutions for service improvements. In this thesis the possibility of using such data to plan vehicle routes with backup provisioning is discussed. That is, a variant of the open vehicle routing problem is proposed, called vehicle routing with backup provisioning, where the possibility of reacting to overloading/overcrowding of vehicles in certain stops is considered. After mathematically formalizing the problem a heuristic algorithm to plan routes is proposed. Results show that vehicle routing with backup provisioning can be a way of providing sustainable urban mobility with efficient use of resources, while increasing quality of service perceived by users. We expect this tool to be useful for the improvement of Urban Transportation Networks.

Keywords: wireless sensor networks, smart cities, urban traffic system, open vehicle routing problem, monitoring, tracking

Resumo

Com os atuais avanços tecnológicos, o aumento das populações nas grandes cidades e com a constante preocupação com os nossos recursos naturais, como a energia, é essencial sermos mais eficientes e melhorarmos as nossas cidades. Com sensores cada vez mais pequenos, baratos e eficientes, procuramos formas de transformar as nossas cidades em cidades inteligentes.

Este trabalho tem como objetivo melhorar a nossa rede de transportes urbanos. O transporte eficiente de pessoas e bens tem sido estudado nos últimos 50 anos. Milhares de trabalhos foram escritos com várias variantes deste problema de optimização. Mais recentemente foi introduzida uma variante do *vehicle routing problem* chamada de *open vehicle routing problem*, que não obriga os veículos a regressar à origem após concluida a sua última distribuição ou tarefa. O nosso problema, que pelo que sabemos não foi discutido na literatura disponível, é uma variante do *open vehicle routing problem* aplicado ao transporte de pessoas, mas com uma caraterística extra que é a possibilidade de fornecer *backup* a estações críticas. Ou seja, construímos as rotas de forma a fornecerem *backup* às estações críticas caso seja necessário. Este *backup* é efetuado dentro de um intervalo de tempo definido.

Começamos por recolher as entradas e saídas de pessoas nos transportes urbanos, o que nos dá uma visão geral da lotação dos veículos ao longo do tempo, tendo estes dados sido recolhidos com recurso a sensores sem fios. As redes de sensores sem fios têm sido usadas para monitorização e rastreamento em diversas áreas como: saúde, fornecimento de energia, monitorização de tráfego, rastreamento de animais e pessoas. Nesta tese vamos usar a monitorização e o rastreamento. Para rastreamento vamos recorrer a nós sensores com *passive infrared*, e para monitorização vamos desenvolver uma plataforma de visualização dos dados.

O software de monitorização foi desenvolvido com recurso a uma *framework Model-View-Controller* de PHP e um software de informação geográfica.

Para a recolha de dados utilizamos sensores da Coalsenses, mais concretamente, o *Security Module* e o *Gateway Module*. O primeiro foi usado para recolher os dados e o segundo para encaminhar os mesmos para a nossa plataforma. Ambos os sensores foram configurados com IPv6, mais concretamente com a stack protocolar 6LoWPAN, para dispositivos com energia limitada. O *Security Module* é composto, entre outros, por um *passive infrared*. Foi necessário reduzir a área de ação do *passive infrared* para $20\,^{\circ}$-

30° graus, em vez dos 110° que é o valor padrão para este tipo de sensores. Esta redução é necessária porque precisamos de uma faixa bem definida para a captura de eventos. Os sensores equipados com *passive infrared*, *Security Module*, foram configurados de forma a enviarem dados com o seu identificador e com o evento, que neste caso é a passagem de uma pessoa, e após validação estes dados são enviados para o *Gateway Module* que reencaminha para a nossa plataforma de monitorização. Optámos por colocar dois *passive infrared* por porta para podermos detetar a direção na entrada ou saída.

Com o software de monitorização, e com os dados por ele recolhidos, temos a possibilidade de analisar os dados em tempo real, permitindo-nos assim propor soluções para melhorar o serviço prestado. Nesta tese é discutida a possibilidade de analisar estes dados para planear rotas com o fornecimento de *backup*. É proposta uma variante do *open vehicle routing problem* chamada *vehicle routing with backup provisioning*, onde é considerada a possibilidade de reagir a eventos de sobrelotação dos veículos em algumas paragens consideradas críticas. Definimos o nosso problema e posteriormente fizemos a formalização matemática do mesmo. Propusemos ainda uma heurística para planear rotas, na qual definimos dois passos, criação das rotas e inclusão de backup. A inclusão de backup extende, se necessário, as rotas construídas para permitir o fornecimento de backup a uma estação crítica dentro de um determinado intervalo de tempo. Vários conjuntos de estações críticas e intervalos de tempo foram testados e analizados.

Depois de alguns testes para avaliar o desempenho da heuriistica, decidimos, então implementar procedimentos pós-optimais para melhorar a performance da nossa heurística e analisar os resultados utilizando dados conhecidos de *vehicle routing problems*. Utilizámos dados com 52 nós, onde definimos que as estações obrigatórias seriam o nó inicial e o nó mais distante (chamado *mandatory*, que depois de executada a heurística poderá ser o nó destino ou não; i.e., poderá haver extensão para além deste *mandatory* para que seja fornecido backup). Como nós críticos definimos os nós com mais procura ou *demand*. Os resultados mostram que o *vehicle routing with backup provisioning* possibilita uma mobilidade urbana sustentável, com uso eficiente de recursos e que os procedimentos pós-optimais melhoram significativamente a qualidade das soluções iniciais obtidas pela heurística. Os resultados mostram que o *vehicle routing with backup provisioning* pode ser uma solução muito flexível e de baixo custo para o melhoramento da rede de transportes e ao mesmo tempo melhorar a qualidade da experiência para os utilizadores. Esperamos que esta ferramenta seja útil para melhorar as nossas atuais redes de transportes urbanos.

Termos chave: redes de sensores sen fios, *smart cities*, sistema de tráfego urbano, *open vehicle routing problem*, monitorização, *tracking*

Acknowledgements

This work is dedicated to all those around me, family and friends for all support given, especially:
To my wife and son, who not only supported me but also believed and encouraged not only to finish this work but especially to be a better person. Thank you for all the love, understanding and support given.
My parents and brother who always believed in me.
To Prof. Noélia Correia and Prof. Gabriela Schütz for their help and support in the design and analysis of this work.
To the Research Center of Electronics Optoelectronics and Telecommunications (CEOT) a special thanks in providing not only the sensors but for all the infrastructure and material provided during this thesis.
A special thanks to all that directly or indirectly contributed for this work, professors, friends, ...
Without them I would not have managed to accomplish.

Contents

List of Figures

Nomenclature

6LoWPAN	IPv6 over Low power Wireless Personal Area Networks
CDT	C/C++ Development Tooling
CI	CodeIgniter
CRUD	Create, Read, Update and Delete
CSS3	Cascading Styles Sheets version 3
DARP	Dial-a-ride Problem
FSM	Finite State Machine
GIS	Geographical Information System
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile
HPP	Hamiltonian Path Problem
HTML5	HyperText Markup Language version 5
IETF	Internet Engineering Task Force
IDE	Integrated Development Environment
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ITS	Intelligent Transportation Systems
LED	Light Emitting Diode
MAC	Media Access Control
MDVRP	Multi-Depot Vehicle Routing Problem
MVC	Model-View-Controller
OVRP	Open Vehicle Routing Problem
PDT	PHP Development Tools
PIR	Passive Infrared
PHP	PHP Hypertext Preprocessor
RFC	Request for Comments
RFID	Radio Frequency Identification
RSUs	Road side units
UN	United Nations
UTCS	Urban Traffic Control System
UTGS	Urban Traffic Guidance System
UTN	Urban Transportation Network
UTS	Urban Transportation System
V2R	Vehicle to Road Side
V2V	Vehicle to Vehicle
VRwBP	Vehicle Routing with Backup Provisioning

VRP Vehicle Routing Problem
WSN Wireless Sensor Networks

List of Publications

1. Carvalho, N. and Schütz, G. and Correia, N. “Vehicle Routing with Backup Provisioning Using Wireless Sensor Infrastructure”, IEEE International Conference on Connected Vehicles and Expo 2014.

2. Carvalho, N. and Schütz, G. and Correia, N. “On the Planning of Vehicle Routing with Backup Provisioning Using Wireless Sensor Technologies”, IEEE Intelligent Transportation Systems Transactions and Magazine (submitted article).

CHAPTER 1

Introduction

With today's technologies, demographic changes leading people to larger cities and the permanent concern with our natural resources, such as energy, it is inevitable that we seek ways to improve our cities. Several steps are being given forward. In Boston, USA, converting street lighting to LEDs (Light Emitting Diode) led to a $2.8 million saving per year with only 40% light converted [1]. In Coimbra, Portugal, the trend is similar as LEDs were adopted for street lighting as an energy saving measure. Besides this, car companies are equipping their vehicles with parking sensors, cameras, luminosity sensors to turn on headlights.

1.1 Motivation

Sensors are a key issue in intelligent environments, and currently we are faced with smaller, cheaper and low-power sensors, which can be deployed in many places. These sensors have limited processing power, storage and battery, low bandwidth and short range. The possibility of organizing these sensors into networks, along with the Internet, brought several opportunities such as environment monitoring, battle field surveillance, among others [2, 3]. Many governments are looking to transform their cities into smart cities. The goal for most cities is to manage and probe their infrastructures and resources efficiently for environmental sustainability and better urbanization development, among others. As major cities continue to grow, these are forced to become "smarter".

According to United Nations (UN) forecast in 2050 70% of world population will live in cities (currently there are about 50%) [4] and there will be about 27 cities with more than 10 million people. Beside this, the human behaviour is leading to a breakpoint on world's natural resources. Global warming, air pollution, among others are the top challenges and the priority of cities (nowadays about 95% of cities still dump raw sewage into their waters). This forces the improvement of cities and urban infrastructures, so that society can deal with such growth of people in cities. This improvement requires tools, which can be related with:

i) Planning;

ii) Management;

iii) Operation.

Planning tools based on merged information about human behaviour are required to aid allocation of resources, as land, water, roads, transportation, sewage, among others. Concerning management concerns these are related with the coordination of infrastructures to plan interventions (such as build and maintain roads, cables and power lines). Operation can be executed depending on the demand and/or based on historical usage, like power capability (amount of power/energy on the lines) during holidays like Christmas. These tools could be applied, for example, to traffic, signalling and traffic lights. Information gathered could reduce the impact of future demands as well as the amount of time spend in traffic.

Transportation networks is another area where such improvement tools could be useful. Each day, we concern ourselves with being efficient with our resources and increasingly look for ways to improve productivity. One main concern is the time spent, each day, in transportation networks. It's reduction is essential in order to become more efficient. As the population of larger cities increases, good urban transportation system must, therefore, be provided. Wireless sensor networks are expected to help on the planning of Urban Transportation Systems (UTS) [5, 6] and even aid blind in UTS [7]. The core problem is its complexity as there are numerous variables such as weather conditions, time, accidents, which makes it virtually impossible to predict the behaviour of traffic flow and hampers urban transportation scheduling planning.

As new technologies emerge it is inevitable that we look for ways to improve our urban transportation networks. The decisions that might lead to such improvements mainly depend on the provisioning of timely information and monitoring tools. Thus, *intelligent transportation systems* (ITS) are now under strong research. These systems are endowed with communication capabilities for data transmission between vehicles (V2V) and between vehicles and roadside access points (V2R), also called *road side units* (RSUs). ITS are seen as key technologies for road safety improvement, congestion avoidance using optimal routing and scheduling, and for the improvement of quality of service experienced by users (both drivers and customers) [8]. This thesis falls within ITS as explained in the following section.

1.2 Contributions

The present thesis explores ways of improving transportation networks. We try to achieve this by planning vehicle routes that include the possibility of reacting to the overloading/overcrowding of vehicles in certain stops. This vehicle routing with backup provisioning paradigm is based on the availability of vehicle load information in real time, which can be captured using wireless sensor technologies, as illustrated in Figure 1.1. It is assumed that vehicles incorporate two *passive infrared* (PIR) sensors at each entrance/exit, to sense movement and determine direction. Having such information available, a vehicle finishing its route in the neighbourhood may pickup more customers in case of overload. This is illustrated in Figure 1.2, which includes two routes $r_1 = \{s_1, s_2, s_3, s_4\}$ and $r_2 = \{s_5, s_6, s_7, s_8, s_9, s_{10}\}$. Stop s_8 is considered a critical one that occasionally gets congested and, therefore, routes have been planned so that there is a vehicle finishing its route near, being able to get fast to s_8 and pick up more customers. In this example, the vehicle finishing route r_1 at stop s_4 is the one moving to s_8. Time for backup to arrive should not exceed a certain time threshold. Note that, once planned, routes remain unchanged and route extensions (or backup) are only activated in case of overload.

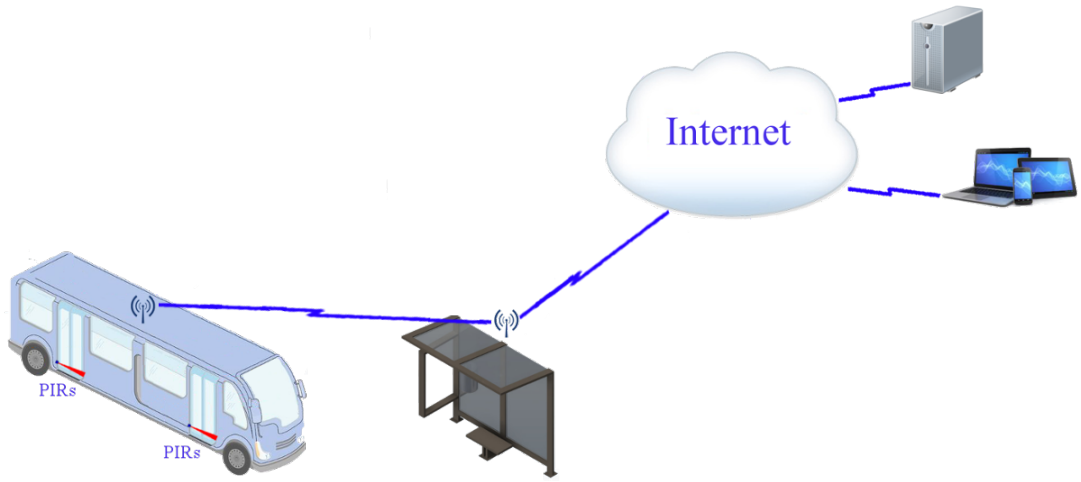


Figure 1.1: Vehicle to roadside access point example.

This vehicle routing with backup provisioning paradigm allows sustainable urban mobility with efficient use of resources, since an agile and adaptable fleet with not so big fleet elements and less schedules per day can be adopted, while improving the quality of service experienced by users.

In summary the benefits of this approach will be:

- *Sustainable urban mobility*: The availability of vehicle load information and reactive response in case of overload will allow sustainable urban mobility;
- *Efficient use of resources*: More efficient use of existing resources become possible

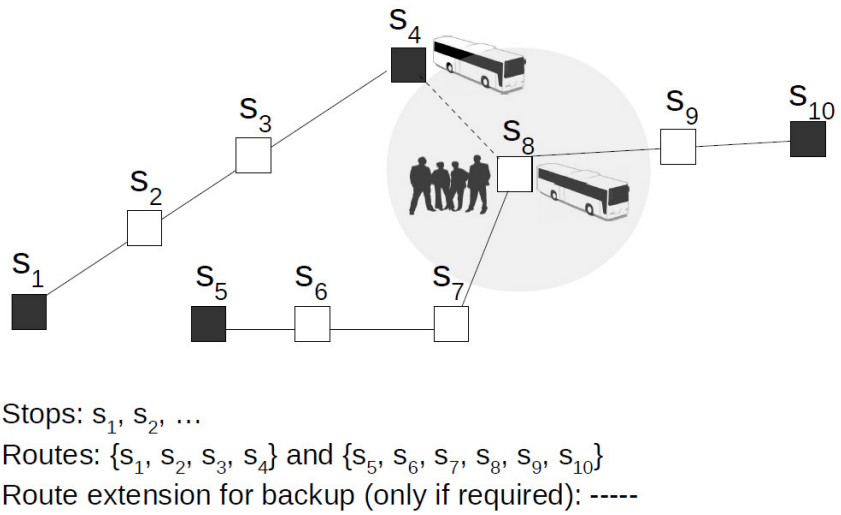


Figure 1.2: Illustration of vehicle routing problem with reactive response to overload.

(energy, infrastructure, and vehicles). An agile and adaptable fleet with not so big fleet elements and less schedules per day can be adopted;

- *Improve quality of service experienced by users*: Fast response to overload allows user waiting times to be reduced.

Besides route planning, including backup provisioning, a monitor tool is also developed so that the amount of people at urban transportation is tracked. This will allow companies to optimize their route plans, and schedules, considering traffic issues like delays and traffic flow at different hours, in order to achieve better tradeoff between customers satisfaction and resources used (e.g. energy and fuels wastes). For customers, the objective is also to use the collected data to inform them the time of arrival, free seats etc, available in train stations, bus stops or even through phones or any Internet connected device.

In summary, the contributions of this thesis include:

- Proposal of a vehicle routing with backup provisioning approach for urban transportation planning;
- Mathematical formalization of the problem;
- Proposal of heuristic algorithm using greedy strategy;
- Pos-optimal procedures to be incorporated into heuristic algorithm;
- Analysis of the benefits of incorporating backup provisioning into vehicle routes;
- Monitoring tool.

1.3 Organization

In Chapter 2 we analysed the state of the art in vehicle routing and work done using WSNs in monitoring and tracking. At Chapter 3 the monitoring tool developed is discussed. Among other things, this tool will allow the occupation of vehicles to be followed in real time, and critical stops to be detected. With this information the routes of vehicles can be planned with backup provisioning in mind, which is done in Chapter 4. At Chapter 5 we propose a heuristic algorithm to solve the routing problem in question. Since our algorithm uses a greedy methods and they can produce not so good results, because a solution can be a local optimum, in Section 5.2 we propose pos-optimal procedures to improve our algorithm performance. In Chapter 6 we analyse the results obtained. We discuss our results, draw conclusions and plan future work at Chapter 7.

CHAPTER 2

State of the art

2.1 Transportation networks

Efficient transportation of goods and persons are problems that have been studied over the last 50 years. Thousands of papers have been written on several variants of these optimization problems. The most well known are the *vehicle routing problem* (VRP) and the *dial-a-ride problem* (DARP). In the classical VRP the concern is to find a set of optimal routes for a fleet of vehicles, based at the depot, to satisfy the demand of geographically dispersed customers at the minimum total travelling cost, while satisfying vehicle capacities. A lot of variants of the stated problem exists: considering time-windows, pick up and delivery customers, different vehicles, multiple depots, and so on [9]. In DARP the goal is to design vehicle routes and schedules for users specifying pick-up and drop-off requests between origins and destinations. A set of minimum cost vehicle routes, capable of accommodating as many users as possible under a set of constraints, will be the output [10].

More recently, a variant of the classical VRP, called OVRP, attracted the attention of practitioners and researchers. In this case vehicles are not required to return to the depot after serving the last customer on a route [11]. This usually arises in cases where industries do not own a vehicle fleet, or their private fleet is inadequate to fully satisfy customer demand, and distribution services (or part of them) are either entrusted to external contractors or assigned to a hired vehicle fleet. In these cases, vehicles are not required to return to the central depot after their deliveries have been satisfied. The main difference between VRP and OVRP is that in VRP the routes are Hamiltonian cycles and in the OVRP the routes are Hamiltonian paths originated at the depot and ending at one of the customers, so the shortest *Hamiltonian Path Problem* (HPP) with a fixed source node has to be solved for each vehicle in the OVRP. The shortest HPP is known to be NP-hard since an instance of the HPP can be converted into an instance of the traveling salesperson problem that is known to be NP-hard. Consequently, the OVRP is also an NP-hard problem. Our problem, as far as known not yet discussed in the literature, is a variant of the OVRP applied to the transportation of persons, considering multiple depots and having the possibility of

backup provision to certain critical stops, having high number of customers.

2.2 Wireless sensor networks

Wireless sensors have been used for monitoring and tracking in several areas: health and wellness, power supply, automated machinery monitoring, traffic monitoring, tracking of human and animals. Concerning its use in the context of traffic and urban transportations, wireless sensors have been used to improve traffic control in large cities [12], collect information of passenger volume [5], to plan traffic signals giving priority to buses [6], provide optimal routes in real-time [13], and even aid blind [7]. The system in [13], is useful as an information system and coordination between different transportations. In [5] the system used for data gathering is discussed but no further planning using such information is done. Here we go further and use the gathered data for vehicle route planning with backup provisioning.

There are several detection/tracking sensor types of thermal signature and/or light changes. The authors in [14] present the weaknesses of several approaches. For our problem we have decided to use PIR, leaving aside other type of sensors like photoelectric sensor that, depending on the type, may requires two points to install, a sensor and a receptor.

We can divide Wireless Sensor Networks (WSNs) into two categories: Monitoring and Tracking. In Monitoring we can include, among others, health and wellness, power supply, automated machinery monitoring, traffic monitoring. In tracking applications we can include object tracking, animals (like ZebraNet [15]), human (PinPtr [16]), vehicles, objects (MAX [17]).

There are several proposals for tracking and monitoring, including cluster based architectures [18, 19], tree-based approaches [20, 21], and Prediction-based Mobility Adaptive Tracking (P-MAT) [22]. All previous proposals are based on the high accuracy of tracking while minimizing the power consumption of the WSN. Some work focus on energy conservation [20] or the tradeoff between energy efficient vs. accuracy of target [23].

Concerning Urban Traffic Systems, which are related with the work developed in this thesis, these have been discussed in [12, 24, 25, 26]. In [12] a city traffic system was proposed for traffic flow control using a WSN, which was expected to deal with the stochastic nature of traffic flows. The Urban Traffic Control System (UTCS) and Urban Traffic Guidance System (UTGS) at China were analysed in [25] and it is stated that these are expected to be the most efficient possible, with effective techniques to reduce traffic congestion, making driving more secure, easier and more comfortable.

A *fleet management system* (FMS), using *Global System for Mobile* (GSM) and *General Packet Radio Service* (GPRS) was discussed in [27], with the advantage of GPS to track in real time the company fleet. A *Radio Frequency Identification* (RFID) was one of

the techniques used in [28, 29] to identify the fleet along with *Geographical Information System* (GIS).

For human and movement detection in [30] they used the *received signal strength indicator* (RSSI) between stationary 2.4 GHz smart outlets. In [31] PIR sensors were used for multiple human tracking, while [24, 26] used PIR to detect human intruders. Our job is not to detect intruders, but we must be able to count how many persons enter and exit the urban transportation. The work to be developed here can be seen as fitting into both tracking and monitoring systems:

- *Tracking*: PIR will be used to detect direction in transportation entries.
- *Monitoring*: Data collected and stored within a monitoring tool, at edge devices.

The PIRs we will be using have similar characteristics to the ones used in [24, 26]. Have a detection angle of $110\,^{\circ}$, to be more exact $93\,^{\circ}$ horizontally and $110\,^{\circ}$ vertically, have a range of 10 meters and can detect moving objects that feature a temperature different from the environment (such as humans).

CHAPTER 3

Monitoring tool

3.1 Development tools and geographical information software (GIS)

In this thesis the Eclipse Integrated Development Environment (IDE) was used with PHP Development Tools (PDT) and C/C++ Development Tooling (CDT). The PDT was used to build the monitoring tool and the CDT to program the sensors.

The monitoring tool was developed mainly using PHP Hypertext Preprocessor (PHP), HyperText Markup Language version 5 (HTML5), Cascading Style Sheets version 3 (CSS3) and Javascript. In any Web tool there are two sides: server-side and client-side. The server is responsible for serving the pages and several programming languages can be used at the server-side, like PHP, C#, among others. The client requests pages from the server and displays them to the user. Sometimes user interaction and dynamically generated content is needed and, therefore, a client-side language like Javascript is needed.

In this thesis the CodeIgniter (CI) is also used to develop the monitoring tool [32]. CI uses model-view-controller pattern (MVC).

Most PHP frameworks use the model-view-controller pattern (MVC), but there are differences in what they are capable of [33], development speed, learning curve, code generation, among others. The Model is the application object, the View is its screen presentation, and the Controller defines the way the user interface reacts to user input. The MVC pattern diagram is shown in Figure 3.1. The main advantages of using such pattern is the increased flexibility and code reuse. Our previous experience allow us to work with some PHP Frameworks CakePHP, Yii and CodeIgniter (CI) [32]. From these three CakePHP [34] and Yii [35] have the ability to generate code, for example, creation of *Create, Read, Update and Delete* (CRUD).

All these frameworks need a well designed and implemented database. In the case of CakePHP and Yii the code generated is based on the database design so any database change during development (e.g. new feature that requires a database change on a already build table) may force us to rebuild the code, and any changes made on those models, views or controllers may be lost or recoded. Since the tool being developed in this thesis

will be constantly evolving, depending on evolving needs, the CI framework has been chosen.

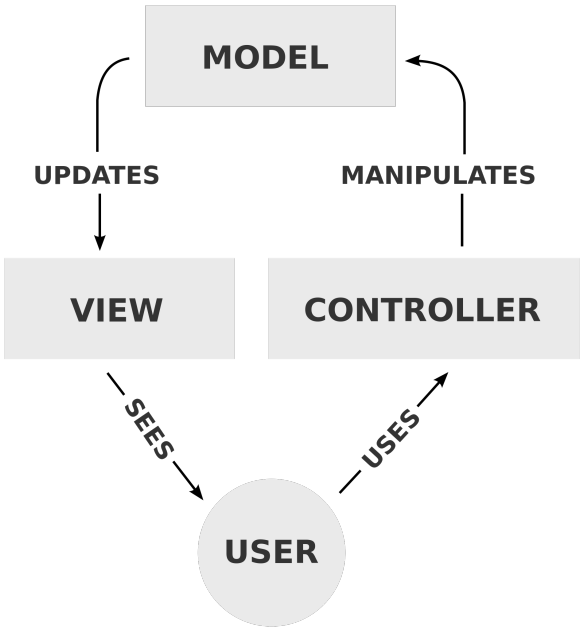


Figure 3.1: MVC pattern diagram.

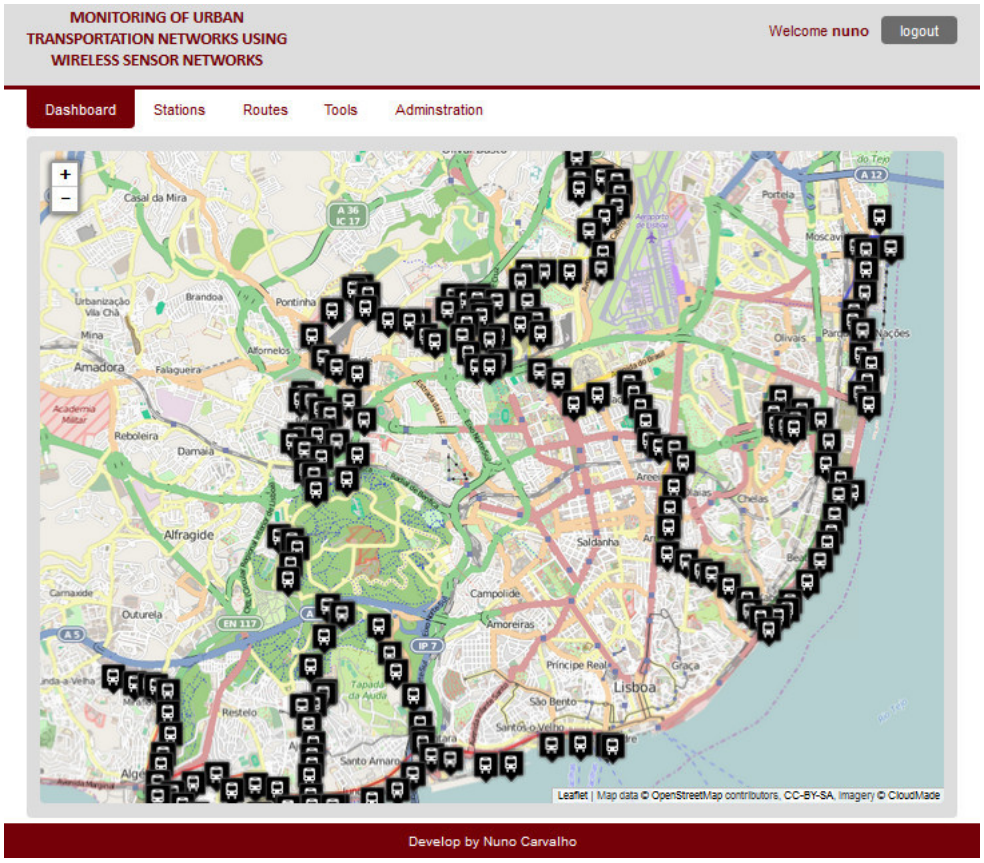


Figure 3.2: Monitoring tool - Home page.

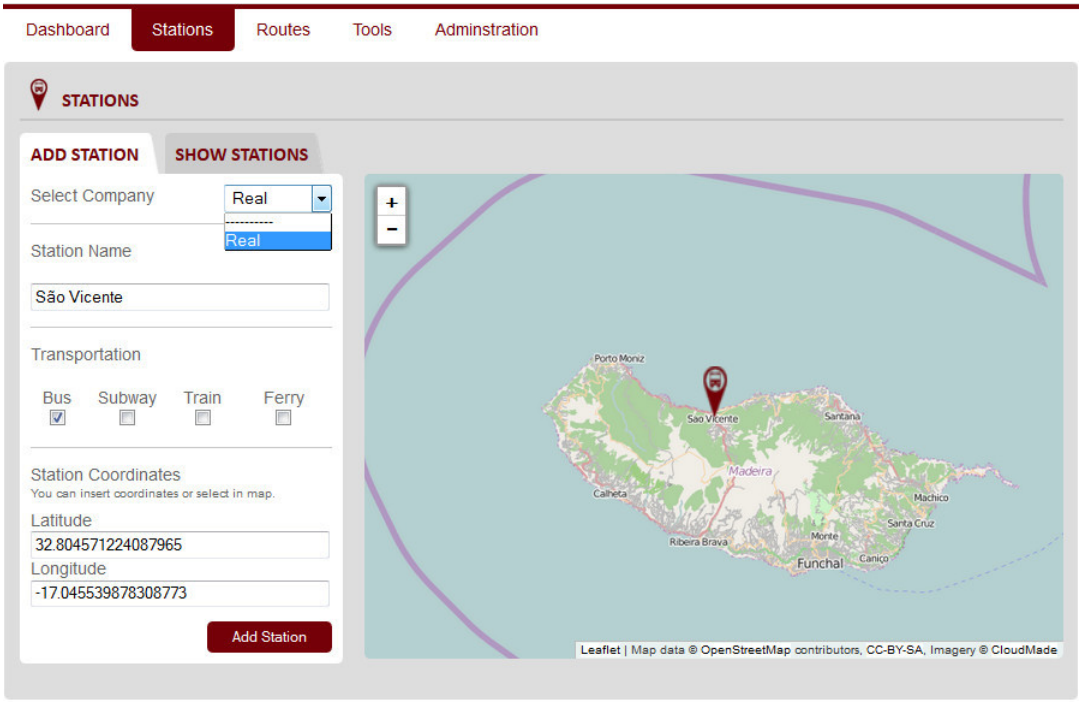


Figure 3.3: Monitoring tool - Adding stations.

Along with CI, a free Geographical Information Software (GIS) [36], the Leaflet, has also been used. GIS "integrates hardware, software, and data for capturing, managing, analysing, and displaying all forms of geographically referenced information" [37]. GIS is usually seen as a map, but it can make spatial information analysis and data editing, and it can perform basic geoprocessing tasks, such as routing and map image display.

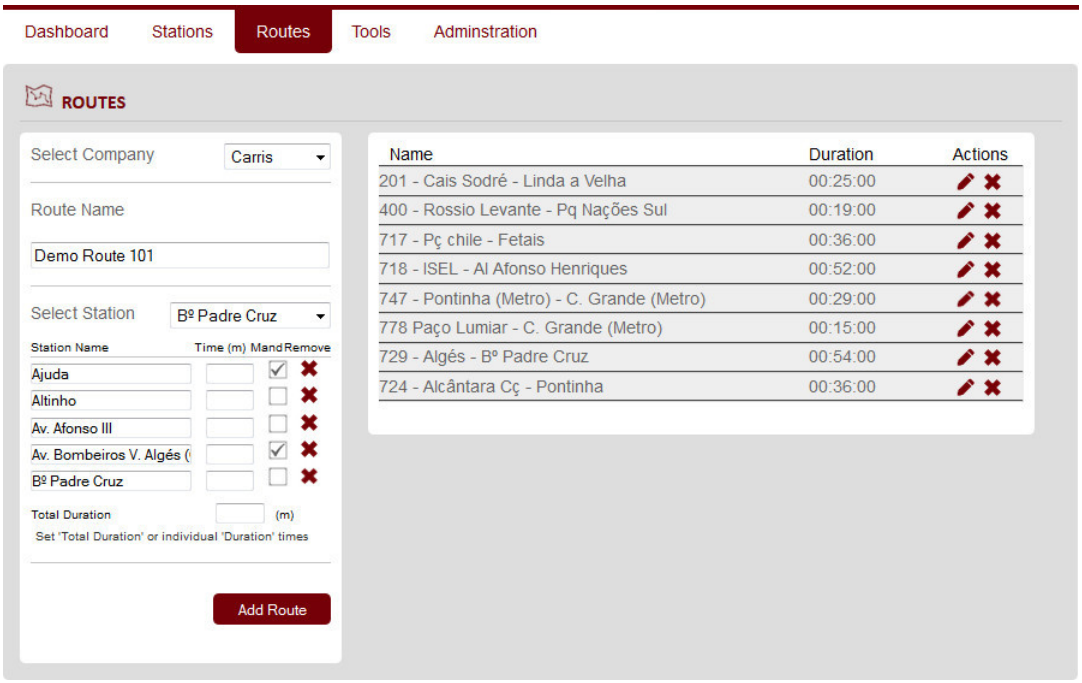


Figure 3.4: Monitoring tool - Adding routes.

In our case this will allow us to add stations/stops, collect data like distance between stations/stops, display all necessary information, among others.

Figure 3.2 shows the home page of the developed tool, including several stations. The main goal for this tool is constant evaluation. We need stations to be inserted in the platform and there are two ways of doing it:

i) upload a file with stations name and coordinates;

ii) inserting them directly in the platform shown in Figure 3.3.

In Figure 3.4 routes are built by selecting a name for the route, then stations are incorporated, mandatory stations are set, and time between stations or maximum time to reach mandatory stations can be defined.

3.2 Infrastructure and data collection

In this thesis the iSense Security Modules [38], referred from now on as nodes, have been used to capture data while the Coalesenses Gateway Module, from now on referred as sink or gateway, have been used to forward data to the monitoring tool. The main reason to work with these sensors is the possibility of setting up them according to our needs, and code/parameter change possibility at deployment time, like identification numbers for nodes.

We choose to implement our communications using IPv6 rather than IPv4. IPv4 allows us to allocate 2^{32} addresses, but with the constant grow of devices with communication needs it is mandatory to implement IPv6 that allows us to have 2^{128} IP numbers. Internet Engineering Task Force (IETF) published, in December 1998, their specification of IPv6 [39] and several updates to this Request for Comments (RFC) were published in the last years, and most devices work now with both IPv6 and IPv4 stacks. The use of both stacks is due to the high cost of changing all IPv4 devices, those that do not support IPv6. There is no easy way for this transaction, and it is being done progressively with equipments incorporating both stacks.

Our first step was to implement IPv6 communications between the nodes and the sink or gateway. The protocol stack implemented at our devices was the IPv6 over Low power Wireless Personal Area Networks (6LoWPAN), which allows an easier integration of nodes in the Internet and allows similar management to standard IP-based networks. Our nodes and sinks have limited power, short range and low bandwidth, so it is important to choose a low power stack. We have reduced the sensor's area of action to a narrow beam, more specifically from the default $110\,^\circ$ to $20\,^\circ$-$30\,^\circ$. We have developed some experiences to see the behaviour of the sensor and gateway, regarding communications, IPv6 addresses assigned, node identification and people counting.

The gateway was configured to assign IPv6 addresses to all discovery nodes, and once the nodes receive an IPv6 address they automatically store the IPv6 address of the gateway allowing them to send their data over to the gateway. Note, however, that a gateway could assign IP addresses to nearby vehicles, which might not be acceptable. Therefore, the IP assignment should take MAC addresses (of nodes inside the same vehicle as the gateway) into consideration. That is, the assignment of IP addresses, done by the gateway, should be restricted to the nodes inside the same vehicle. This can be easily achieved since our sensors allow on site configuration. The data send by nodes was:

- node Identification;
- node counting.

We study the possibility of identifying each node with its MAC address but a more

simple and easier to manage solution was found. Nodes identification was composed by two parts:

- *Transportation Identification*: An unique number identifying each transportation;
- *PIR Identification*: A number between 1 and 4 identifying each PIR sensor, PIR 1 and 2 at the entrance and PIR 3 and 4 at the exit. Even numbers placed on the inside and odd numbers placed more at the outside, as showed in Figure 3.5.

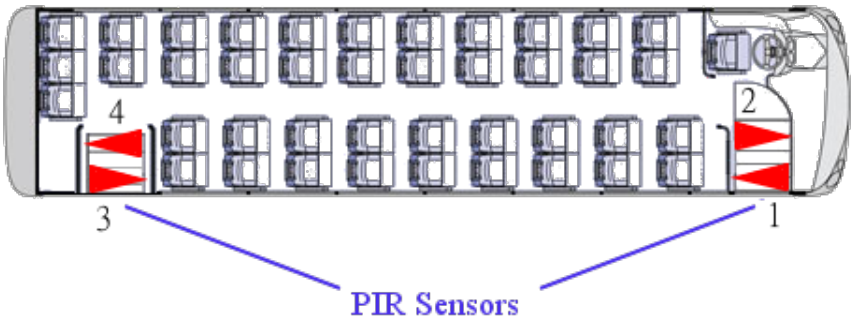


Figure 3.5: Sensor application in a bus.

The Figure 3.5 shows an example of where the sensors will be placed, inside a bus, where the red area is the coverage of the PIR sensor. Figure 3.6 shows how the sensors will connect to the Internet. Data is sent to the sink and this one is responsible for transmitting data through the Internet. Sink nodes are connected to the Internet with an Ethernet cable.

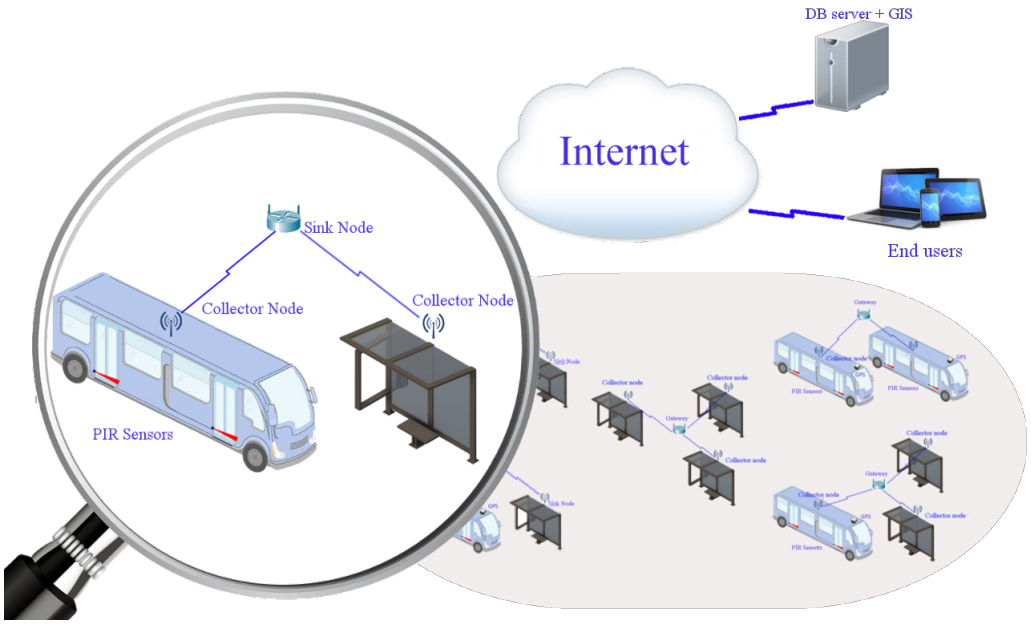


Figure 3.6: Architecture and data communication schema.

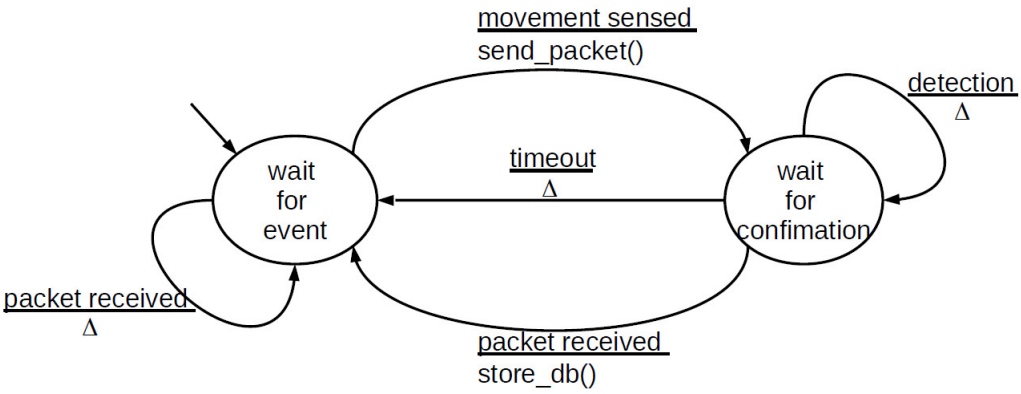


Figure 3.7: FSM diagram for each sensor node of dual implementation.

The infrastructure assumed in vehicles includes a dual PIR implementation at each entrance/exit. This dual PIR is used to control the direction of people passing. The *finish state machine* (FSM) diagram of each PIR node of such dual implementation is shown in Figure 3.7. The diagram is the same for both nodes.

While in the "wait for event" state, node i might sense movement of people and, in this case, a packet is sent to its neighbour j. After sending this packet the node waits for a confirmation from node j. When such confirmation arrives a $i - j$ movement is stored at a remote database, using WiFi or WiMAX, and the node enters in "wait for event" state again. When waiting for confirmation another detection may occur, and in this case no action is taken to avoid counting the same person many times. If timeout occurs, meaning that the person probably walked back, also no action is taken and the node enters in "wait for event" state. While in the "wait for event" state, a node i might receive a packet from its neighbour j (meaning that it has detected movement) asking for a confirmation. Node i remains waiting because a movement must be sensed by i for a packet to be sent to j, which will serve as confirmation to j. Upon receiving such packet, its neighbour (which is in "wait for confirmation" state) will send a $j - i$ movement for storage in the remote database.

3.3 Improvement of transportation networks

The development of a tool to capture urban transportation information allows the optimization of such networks. In case different transportation networks are integrated, or information about other networks is available, integrated policies may also be outlined.
A final goal is to develop mathematical models and/or algorithms based on the collected data, which might include load, time of arrival to collecting nodes, or other. The problems that can be addressed include the improvements of:

- routes
- schedulings
- coordination between different transportations

These may include proactive or reactive strategies. In this thesis scheduling and coordination between different transportations is not handled.

CHAPTER 4

Problem formalization

4.1 Problem definition

In the following it is assumed that routes are planned without mention to their schedules. The resulting routes can then be adopted many times a day according to the needs, requiring further assignment of schedules and drivers.

4.1.1 Notation and definitions

In the definitions discussed next $\mathcal{R}$ is a given set of routes that must be planned, each route initially having just the source and one or more mandatory stops, and $\mathcal{S}$ is a given set of stops/locations that must be incorporated into the routes for them to be visited. A set of stops requiring backup, in case of overload, is given by $\mathcal{B} \subset \mathcal{S}$.

Definition 1 (Route) *A route $r \in \mathcal{R}$ can be seen as a path that initially has a starting point, $src(r)$, and a set of mandatory stops, $\mathcal{M}^r \subset \mathcal{S}$. There are at least two mandatory stops, $src(r)$ and another location that might end up being the destination or not. Other stops will be inserted during route construction. The final destination $dst(r)$ can be any stop, mandatory or not.*

Definition 2 (Mandatory Stop) *A mandatory stop $s \in \mathcal{M}^r$ is a location point that must be visited by the vehicle of route r. A mandatory stop s has a time window info $i(s) = \{s', \Delta(s', s)\}$ associated with it, which indicates the maximum allowed time from the previous location $s' \in \mathcal{M}^r$, also mandatory and to be included in the route, until $s \in \mathcal{M}^r$. The extra stops to be inserted in a route, during route construction, can not violate the time window associated with mandatory stops.*

The $i(s) = \{s', \Delta(s', s)\}$ pair for a mandatory stop s basically expresses the following: "if the departure from s' is done at time $\Delta_{s'}$, or after, the arrival to s must be no later than

$\Delta_{s'} + \Delta(s', s)$". The insertion of mandatory stops, with a reference to a previous stop and time window, allows round trips or trips with specific connection points to be planned.

Definition 3 (Stop) *Any stop $s \in \mathcal{S}$, mandatory or not, is a location point that must be visited by at least one route. A pair of stops (s_i, s_j) has a cost $t(s_i, s_j)$ associated with it that is basically the estimated time when going from s_i directly to s_j. If $s \in \mathcal{B}$ then backup must also be provided for a reactive response in case of overload.*

Definition 4 (Backup) *A backup for a critical stop $s \in \mathcal{B}$ can be supplied by any vehicle reaching its destination stop at some place near, and able to reach s before a specified time threshold T. That is, the backup for s, denoted by $b(s)$, will be a route, $b(s) \in \mathcal{R}$, whose vehicle will have to make an extra traversal from $dst(b(s))$ to s in less than T, a threshold for the maximum allowed time for backup to arrive.*

Note that every stop must be visited at least once. A stop may be visited more than once because routes will be built having backup for critical stops into consideration. That is, routes might be extended to provide backup, leading to more visits at some stops. However, this is expected to be better than increasing scheduling and number of vehicles/drivers. A minimum number of stops per route is not imposed since it is assumed that the set of routes to be planned is adequate to the fleet and region covered. Routes with few stops can always be eliminated from the final solution.
Having the previous notation and definitions in mind, our route planning problem can be defined as follows. This problem is called here *vehicle routing with backup provisioning* (VRwBP).

Definition 5 (VRwBP) *Given a set of routes $\mathcal{R}$, each route $r \in \mathcal{R}$ with a reference to a set of mandatory stops $\mathcal{M}^r$ with time constraints, a set of stops $\mathcal{S}$ that need to be visited and a set of critical stops $\mathcal{B} \subset \mathcal{S}$ requiring backup, insert stops into routes while not violating time constraints, and ensuring backup to critical stops, in less than a time threshold T.*

Please note that routes are to be planned without taking into consideration schedule assignment. After determining the routes, these can be adopted many times a day according to the needs and, therefore, further schedule/fleet/drivers planning is required. These issues are determinant for the availability of a backup vehicle when overcrowding is detected.

4.2 Mathematical formalization

To find the optimal solution for the VRwBP problem we proceed with its mathematical formalization and discussion. This will also allow for a deep understanding of the problem, allowing adequate procedures to be used by the heuristic algorithm proposed in the next section. In the following formalization it is assumed that $t(s_i, s_j) = \infty$ if $s_i = s_j$. Before going into the formalization let us summarize the known information:

R	Given set of routes to be planned, initially with source and one or more mandatory stops.
S	Given set of stops to be visited at least one time.
M^r	One or more mandatory stops in route r.
B	Set of stops requiring backup.
$src(r)$	Source for route r.
$dst(r)$	Final destination for route r.
$s', \delta(s', s)$	Mandatory stop time window information.
$t(s_i, s_j)$	Cost between stops s_i and s_j.

The variables that need to be found are:

$\xi^r_{s_i}$	One if route $r \in \mathcal{R}$ selects stop $s_i \in \mathcal{S}$ to become its final destination, $dst(r)$, zero otherwise.
$\upsilon^r_{s_i,s_j}$	One if route $r \in \mathcal{R}$ includes going directly from stop s_i to $s_j \in \mathcal{S}$, zero otherwise.
$\kappa^{r,s}_{s_i,s_j}$	When ensuring that the time window associated with mandatory stop s of route r is non violated, this accounts the time items flowing from s_i to $s_j \in \mathcal{S}$.

– Objective Function:

$$\text{Minimize} \sum_{r\in\mathcal{R}} \sum_{s_i\in\mathcal{S}} \sum_{s_j\in\mathcal{S}} t(s_i, s_j) \times \upsilon^r_{s_i,s_j} \tag{4.1}$$

This avoids unnecessary long routes and minimizes the number of visits to stops.

– Building routes:

$$\sum_{s_j\in\mathcal{S}:t(s_i,s_j)\neq\infty} \upsilon^r_{s_i,s_j} = 1, \forall r \in \mathcal{R}, s_i = src(r) \tag{4.2}$$

$$\sum_{s_j\in\mathcal{S}:t(s_i,s_j)\neq\infty} \upsilon^r_{s_i,s_j} - \sum_{s_j\in\mathcal{S}:t(s_j,s_i)\neq\infty} \upsilon^r_{s_j,s_i} = (-1) \times \xi^r_{s_i}, \quad \forall r \in \mathcal{R}, \forall s_i \in \mathcal{S}\backslash\{src(r)\} \tag{4.3}$$

$$\sum_{s \in \mathcal{S}} \xi_s^r = 1, \forall r \in \mathcal{R} \tag{4.4}$$

Expression 4.2 forces routes to emanate from the source $src(r)$, expression 4.3 imposes the flow conservation law to routes and expression 4.4 forces the existence of a final destination stop.

– Mandatory stops:

$$\sum_{s_j \in \mathcal{S}: t(s_i,s_j) \neq \infty} \upsilon_{s_i,s_j}^r + \sum_{s_j \in \mathcal{S}: t(s_j,s_i) \neq \infty} \upsilon_{s_j,s_i}^r \geq 1, \\ \forall r \in \mathcal{R},\ \forall s_i \in \mathcal{M}^r \tag{4.5}$$

The vehicle of a route r is forced to arrive/depart from every mandatory stop of that route.

– Ensuring backup:

$$min_{r \in \mathcal{R}} \{ \sum_{s_j \in \mathcal{S}} \xi_{s_j}^r \times t(s_j, s_i) \} \leq T, \forall s_i \in \mathcal{B} \tag{4.6}$$

where T is a threshold for the maximum allowed delay for backup to arrive. Expression 4.6 may be expanded to a set of linear expressions.

– Visited times:

$$\sum_{r \in \mathcal{R}} \sum_{s_j \in \mathcal{S}: t(s_i,s_j) \neq \infty} \upsilon_{s_i,s_j}^r + \sum_{r \in \mathcal{R}} \sum_{s_j \in \mathcal{S}: t(s_j,s_i) \neq \infty} \upsilon_{s_j,s_i}^r \geq 1, \\ , \forall s_i \in \mathcal{S} \tag{4.7}$$

Constraints 4.7 state that every stop must be visited at least once.

– Time windows:

$$\kappa_{s'}^{r,s} = \Delta(s', s),\ \forall r \in \mathcal{R}, \forall s \in \mathcal{M}^r \wedge s \neq src(r), \\ \{s', \Delta(s', s)\} \wedge s' \in \mathcal{M}^r \tag{4.8}$$

$$\kappa_{s_i}^{r,s} - t(s_i, s_j) + K \times (1 - \upsilon_{s_i,s_j}^r) \geq \kappa_{s_j}^{r,s}, \forall r \in \mathcal{R}, \\ \forall s \in \mathcal{M}^r, \forall s_i, s_j \in \mathcal{S} \tag{4.9}$$

$$\kappa_s^{r,s} \geq 0, \forall r \in \mathcal{R}, \forall s \in \mathcal{M}^r \tag{4.10}$$

where K is a big time constant. These constraints ensure that the time windows associated with mandatory stops, i.e. the maximum allowed time from the previous mandatory stop, are not violated. For this purpose, considering a route r and mandatory stop s with time window $\{s', \Delta(s', s)\}$, the stop s' is assumed to have a $\Delta(s', s)$ time in it that will vanish as the route is traversed toward s (can be seen as time items that will be deposited in intermediate stops from s' to s). This is what constraints 4.8-4.9 do. Expression 4.10 ensures that the stops used to control the non violation of time windows are the ones at the routes being built.

– Non-negative assignments

$$\kappa_{s_i}^{r,s} \in \mathbb{R};\ \upsilon_{s_i,s_j}^r, \xi_{s_i}^r \in \{0, 1\}. \tag{4.11}$$

A minimum number of points per circuit is not imposed since it is assumed that the set of circuits to be planned is adequate to the fleet and region covered. Circuits with few stops can always be eliminated at solution is provided.

As previously said, this problem can be seen as a variant of the OVRP applied to the transportation of persons, considering multiple depots and having the possibility of backup provision to certain critical stops. In OVRP vehicles are not required to return to the central depot after their deliveries have been satisfied [11]. The main difference between VRP and OVRP is that in VRP the routes are Hamiltonian cycles and in OVRP the routes are Hamiltonian paths originated at the depot and ending at one of the customers, so the shortest *Hamiltonian path problem* (HPP) with a fixed source node has to be solved for each vehicle in the OVRP. The shortest HPP is known to be NP-hard since an instance of the HPP can be converted into an instance of the traveling salesperson problem that is known to be NP-hard. Therefore, the OVRP, and consequently our problem, are also NP-hard problems. For this reason the development of an efficient heuristic approach is needful. This will be done in the following Chapter.

CHAPTER 5

Heuristic algorithm

5.1 The algorithm basis

The heuristic approach proposed in this section has two main steps:

- *Framework route building*: Here the concern is to build the basis framework of routes, where all stops are visited while not violating time window constraints.
- *Extension for backup inclusion*: After building the basis framework, routes are extended for backup provision to be ensured.

To evaluate the possibility of inserting stops in specific routes, neighbourhood information around stops is built. Neighbourhood information is associated with consecutive pairs of mandatory stops, to evaluate the possibility of inserting an extra stop between them. Please note that given $i(s) = \{s', \Delta(s', s)\}$ info for a mandatory stop s will change when stops are permanently inserted into routes. Initially, when s is inserted after $s', t(s', s)$ is subtracted from $\Delta(s', s)$ so that it stores the time margin left for further insertions of stops. Whenever a stop is permanently inserted between s' and s this will be updated.

5.1.1 Route framework

When building the basis framework of routes the following variables are required:

$\eta^r_{(s_i,s_j)}$ Neighbourhood for the pair of mandatory stops s_i and $s_j \in \mathcal{M}^r$ of route $r \in \mathcal{R}$. Used to evaluate the possibility of inserting an extra stop between s_i and s_j.

$\omega^r_{(s_i,s_j)}$ Cost of inserting a specific stop between the pair of mandatory stops s_i and s_j of route $r \in \mathcal{R}$. Used when building neighbourhoods.

θ Specific subset of all the neighbourhoods.

Neighbourhood elements are (s_k, φ) pairs, where s_k is a potential extra stop and φ the time margin left if s_k is to be inserted. The output will be:

L^r List of stops for each route $r \in \mathcal{R}$.

$dst(r)$ Destination stop for each route $r \in \mathcal{R}$.

5.1 The algorithm basis

Building the route framework includes inserting mandatory stops first and then non-mandatory. While retrieving from the pools of stops $\mathcal{M}^r$ and S, and inserting in the ordered L^r lists, the time windows are updated to stay with the time margin left. During the following discussion an example is used, for illustration, that considers the stops in Figure 1.2.

- **Inserting mandatory stops**: Routes start by having mandatory stops.

```
1  for each r ∈ R do
2      /*list of stops for route r starts with source*/;
3      L^r = {src(r)};
4      s = {s ∈ M^r : i(s) = {src(r), Δ(src(r), s)}};
5      repeat
6          /*mandatory stop is s*/;
7          s' = s;
8          /*insert mandatory stop*/;
9          L^r ← s';
10         /*determine next mandatory stop*/;
11         s = {s ∈ M^r : i(s) = {s', Δ(s', s)}};
12         /*update margin left*/;
13         Δ(s', s) = Δ(s', s) − t(s', s);
14     until s == NULL;
15     /*destination is last stop inserted*/;
16     dst(r) = s';
17 end
```

- **Inserting non-mandatory stops**: Here the neighbourhoods are built first and then non-mandatory stops are inserted using this info. Stops not inserted in any route will be attached at the end of routes.

  – **Neighbourhood formation**: Neighbourhood info is built for each pair of consecutive mandatory stops.

```
/*neighbourhoods start empty*/;
for each r ∈ R do
    for each pair of consecutive mandatory stops (s'_j, s_j) ∈ L^r do
        η^r_(s'_j,s_j) = {};
    end
end
/*fill neighborhood info*/;
for each r ∈ R do
    for each pair of consecutive mandatory stops (s'_j, s_j) ∈ L^r do
        for each s_k ∈ {S − ⋃_{r∈R} M^r} do
            /*cost of inserting s_k between s'_j and s_j*/;
            ω^r_(s'_j,s_j) = t(s'_j, s_k) + t(s_k, s_j) − t(s'_j, s_j);
            if ω^r_(s'_j,s_j) < Δ(s'_j, s_j) then
                /*time margin*/;
                φ = Δ(s'_j, s_j) − ω^r_(s'_j,s_j);
                /*neighbourhood is increased*/;
                η^r_(s'_j,s_j) ← {(s_k, φ)};
            end
        end
    end
end
```

– **Insertion using neighbourhood info**: Insert first the stops included in a single neighbourhood and then stops included in many neighbourhoods according to time margins left.

for *each* $s_k \in \{\mathcal{S} - \bigcup_{r\in\mathcal{R}} \mathcal{M}^r\}$ **do**
  /*extract neighbourhood elements referring to s_k*/;
  $\theta = \{(s,\varphi) \in \bigcup_{r\in\mathcal{R}, s_j\in\mathcal{M}^r} \eta^r_{(s'_j,s_j)} : s = s_k\}$;
  /*number of times s_k is included in neighbourhoods*/;
  if $|\theta| == 1$ **then**
    /*extract place where s_k must be inserted*/;
    $\{r,(s_i,s_j)\} = \{r,(s_i,s_j) : \theta \subset \eta^r_{(s'_j,s_j)}, \forall r \in \mathcal{R}, (s_i,s_j) \in L^r\}$;
    /*insert s_k into route r between stops s_i and s_j*/;
    InsertInBetween(L^r,(s_i,s_j),s_k);
    /*update time window margin*/;
    $\Delta(s'_j,s_j) = \Delta(s'_j,s_j) - [t(s_i,s_k) + t(s_k,s_j) - t(s_i,s_j)]$;
  end
end
for *each* $s_k \in \{\mathcal{S} - \bigcup_{r\in\mathcal{R}} \mathcal{M}^r\}$ **do**
  /*extract neighbourhood elements referring to s_k*/;
  $\theta = \{(s,\varphi) \in \bigcup_{r\in\mathcal{R}, s_j\in\mathcal{M}^r} \eta^r_{(s'_j,s_j)} : s = s_k\}$;
  /*number of times s_k is included in neighbourhoods*/;
  if $|\theta| > 1$ **then**
    /*extract place where s_k must be inserted; the one leaving the largest margin*/;
    $\{r,(s_i,s_j)\} = \{r,(s_i,s_j) : (s_k,\varphi) \in \eta^r_{(s'_j,s_j)} \wedge (s_k,\varphi) =$
    $argmax_{(s_k,\varphi')\in\theta}\{\varphi'\}, \forall r \in \mathcal{R}, (s_i,s_j) \in L^r\}$;
    /*insert s_k into route r between stops s_i and s_j*/;
    InsertInBetween(L^r,(s_i,s_j),s_k);
    /*update time window margin*/;
    $\Delta(s'_j,s_j) = \Delta(s'_j,s_j) - [t(s_i,s_k) + t(s_k,s_j) - t(s_i,s_j)]$;
  end
end

– **Remaining stops**: Stops not inserted at previous steps will be inserted at the end of current routes.

for *each* $s_k \notin L^r$ **do**
  /*Select route whose destination is more near s_k*/;
  $r = argmin_{r\in\mathcal{R}}\{t(dst(r),s_k)\}$;
  /*insert stop*/;
  $L^r \leftarrow s_k$;
  /*destination is last stop inserted*/;
  $dst(r) = s_k$;
end

5.1.2 Backup inclusion

When determining how backup is to be provided the following variables are required:

- ζ^s Set of routes that can provide backup to $s \in \mathcal{B}$.
- δ Time it takes for backup to arrive.
- ϑ List of stops resulting from a shortest path calculation.

The output will be:

- $b(s)$ Backup for every stop $s \in \mathcal{B}$.
- L^r Final list of stops for every route $r \in \mathcal{R}$, after extending if necessary.
- $dst(r)$ Final destinations for every route $r \in \mathcal{R}$, after extending if necessary.

The following code is used to extend the route framework in order to incorporate backup provision, where T is a threshold for the maximum allowed time for backup to arrive, as mentioned in Section 4.1.1.

```
1  for each s ∈ B do
2      /*initially list of potential backup providers is empty and time for backup arrival is a
       big value*/;
3      ζ^s = {};
4      δ = ∞;
5      for each r ∈ R do
6          if t(dst(r), s) ≤ T then
7              /*after finishing route r, vehicle is able to arrive to s before T*/;
8              ζ^s = ζ^s ⋃ {r};
9          end
10         if t(dst(r), s) ≤ δ then
11             /*time for backup arrival is updated*/;
12             δ = t(dst(r), s);
13             b(s) = r;
14         end
15     end
16     if ζ^s == ∅ then
17         /*in next step do not consider direct links with cost/time higher than T*/;
18         ϑ=ShortestPath(source = dst(b(s)), destination = s);
19         while t(dst(b(s)), s) > T do
20             dst(b(s))=RemoveHead(ϑ);
21             L^{b(s)} ← dst(b(s));
22         end
23     end
24 end
```

The above procedures plan routes without taking into consideration schedule assignment. After determining the routes, these can be adopted many times a day according to the needs and, therefore, further schedule/plan/drivers planning is required.

5.2 Improvements to algorithm

Since our algorithm uses greedy methods and they can produce not so good results, because a solution can be a local optimum, we choose to apply pos-optimal procedures. After the routes have been constructed the following pos-optimal local search procedures are executed. These include intra and inter route relocations and exchanges as proposed in [40], but these were adapted to our VRwBP specific problem. The improvements done were the following:

Intra-route relocation

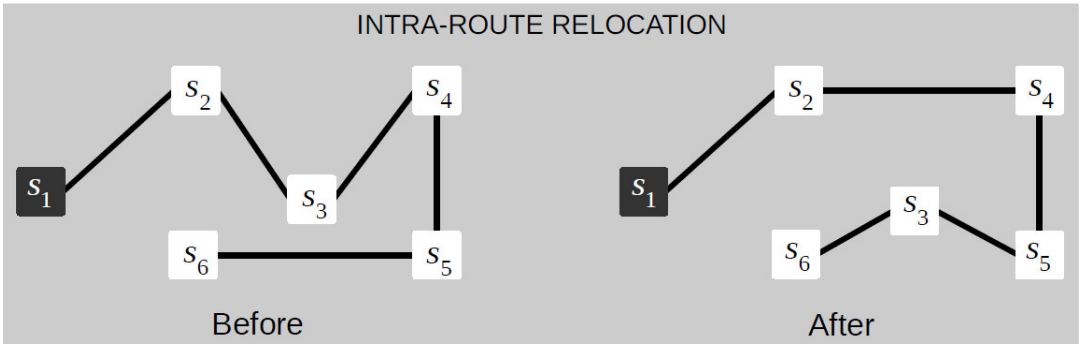


Figure 5.1: Intra-route relocation.

A stop may change its position inside its route. This is illustrated in Figure 5.1. That is, at each route $r \in \mathcal{R}$ the non-mandatory stop s with the longest distance from its predecessor and successor is picked. If a better feasible solution is obtained by moving this stop to another place then removal and relocation occurs. The new place should be the one with the shortest distance from s to the new predecessor and successor stops. This can be done many times, while the feasible solution is improved.

Intra-route exchange

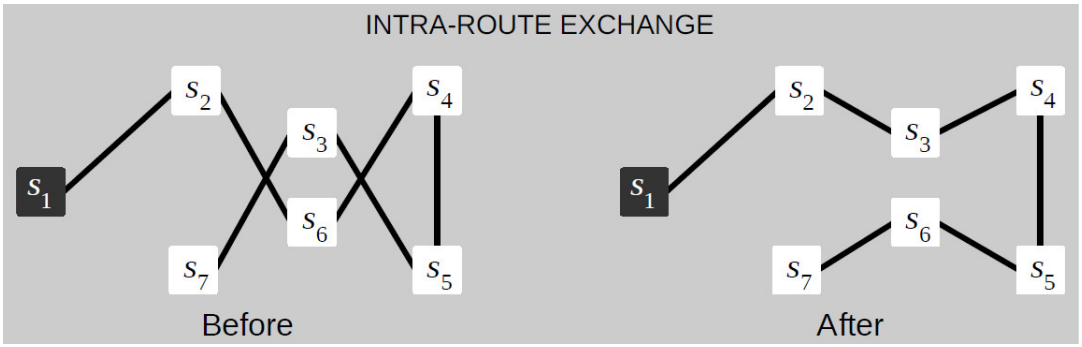


Figure 5.2: Intra-route exchange.

5.2 Improvements to algorithm

A stop may switch its position with another stop, both in the same route. This is illustrated in Figure 5.2. That is, at each route $r \in \mathcal{R}$ two non-mandatory stop si and sj , not between the same pair of mandatory stops, with the shortest distance between them are picked. If a better feasible solution is obtained by switching these stops then removal and relocation occurs. This can be done many times, while the feasible solution is improve.

Inter-route relocation

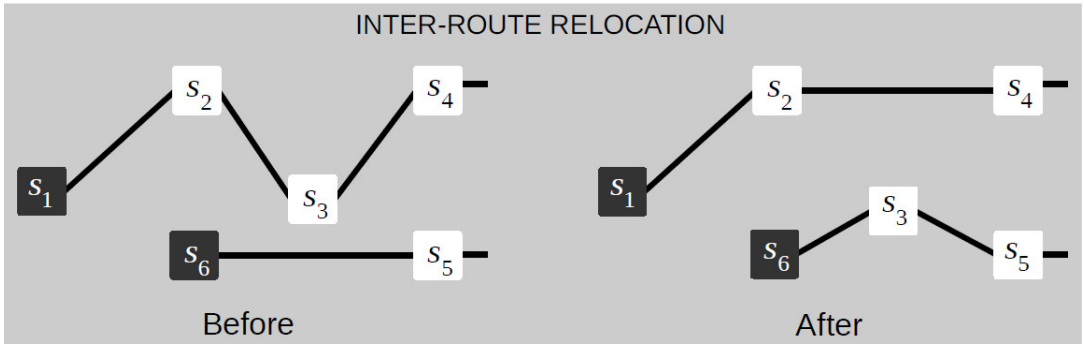


Figure 5.3: Inter-route relocation.

Inter-Route Relocation: A stop may be moved to another route. This is illustrated in Figure 5.3. That is, for each route $r \in \mathcal{R}$ the non-mandatory stop s with the longest distance from its predecessor and successor is picked. If a better feasible solution is obtained by moving this stop to another route then removal and relocation occurs. The new route and place should be the one with the shortest distance from s to the new predecessor and successor stops. This can be done while the feasible solution is improved.

Inter-route cross-exchange

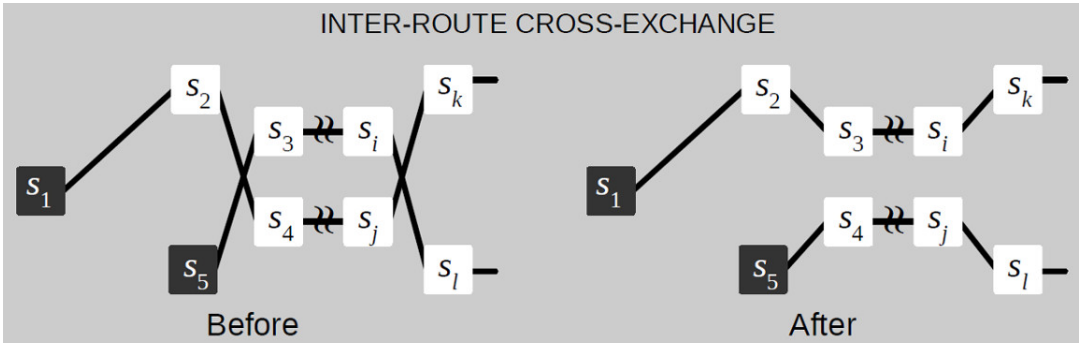


Figure 5.4: Inter-route cross-exchange.

Two sets of consecutive stops of two different routes may be switched. This is illustrated in Figure 5.4. That is, the two stops si and sj with the shortest distance between

them, and belonging to different routes, are initially picked. Stop si will be part of set A while stop sj will be part of set $\mathcal{B}$. Then, sets A and $\mathcal{B}$ are extended, by including a consecutive nearby stop, while a better feasible solution is obtained if sets were switched. The consecutive nearby stop should be the one that minimizes the total distance from all stops of one set to all stops of the other set. This can be done while the feasible solution is improved.

CHAPTER 6

Result analysis

6.1 Result analysis

In this section the results obtained by the proposed heuristic algorithm are analysed. Since the instance sets available in the literature to evaluate vehicle routing problems can not be directly applied to our VRwBP problem, the instance sets commonly used to evaluate multi-depot vehicle routing problem (MDVRP) heuristics, described in [41], have been adopted while making some assumptions. Both instance sets and best known results for the MDVRP are available in [42].

6.1.1 Assumptions

The MDVRP instances include information about the number of depots available, maximum number of vehicles available per depot, and nodes/stops to be visited. For each stop, its coordinates, service time and demand are given. The P instances have 0 of service time and vehicles do not return to the depot while PR instances have non-zero service times and vehicles return to the depot. For the VRwBP problem under analysis the following assumptions must be applied for each instance:

- Stops requiring backup, $\mathcal{B}$, will be the ones having more demand. Different $\mathcal{B}$ set sizes will be tested.
- To define the total number of routes in $\mathcal{R}$, the solutions provided by the best known heuristic results were used. In such solutions, if m vehicles leave a specific depot then m empty routes starting at that depot will be included in $\mathcal{R}$. This choice is based on the fact that best known solutions provide good upper bounds on the number of routes required to cover all stops. Therefore, it is not necessary to consider all vehicles available per depot (info provided by MDVRP instances).
- Considering a specific depot s_i and the m routes starting at s_i, determined as stated previously, the mandatory stops of these routes will include s_i and the m stops

farther away from s_i, one for each route. That is, each route has two mandatory stops, the depot and one of the m stops.

- Due to the way mandatory stops have been defined, a reasonable time window for a mandatory stop s at route r will be:

$$\Delta(s', s) = duration(r) - \frac{\sum_{s_k \in \mathcal{S}} service(s_k)}{|\mathcal{R}|}$$

where $duration(r)$ is the total duration of route r, provided by the best known results, and $service(s_k)$ is the service time at stop s_k. That is, the average total service time per route is removed from the total duration of the route, since service time is not relevant for our VRwBP problem. This result must be divided by 2 for PR instances because the total duration of routes include returning to the depot, and this must not be considered when defining the time windows of mandatory stops.

- The threshold of a critical node s_j, denoted by T_{s_j}, will be a percentage of s_j's route duration, i.e, route of the best known results where s_j is inserted. Thus, $T_{s_j} = p \times duration(r), s_j \in r$, $0 < p \leq 1$. Different percentage values will be tested.

The following analysis includes only the results for the 52 node instances available in [42]. Please note that the planning under discussion in this article does not take route schedules into consideration. Therefore, besides the just mentioned instance related assumptions, it is also assumed that route schedule assignment is done, after route planning, in a way that vehicles finish their route in time to provide backup to their associated critical nodes.

6.1.2 Backup provisioning and pos-optimal impact analysis

In this section we analyse the impact of planning vehicle routes with backup provisioning. Since the heuristic will run for many instances, having different service times, demands, etc, there might be different performances for different instances. For this reason, to analyse the impact of backup provisioning (BP) the following gap is analysed:

$$Gap_i^{BP} = \frac{Distance_i^{BP} - Distance_i^{NoBP}}{Distance_i^{NoBP}} \tag{6.1}$$

where $Distance_i^{BP}$ is the distance value obtained when backup is provided and $Distance_i^{NoBP}$ is the distance value obtained if no backup is provided, considering a specific instance i. Results shown in the following plots include the average of these gaps, over all analysed instances.

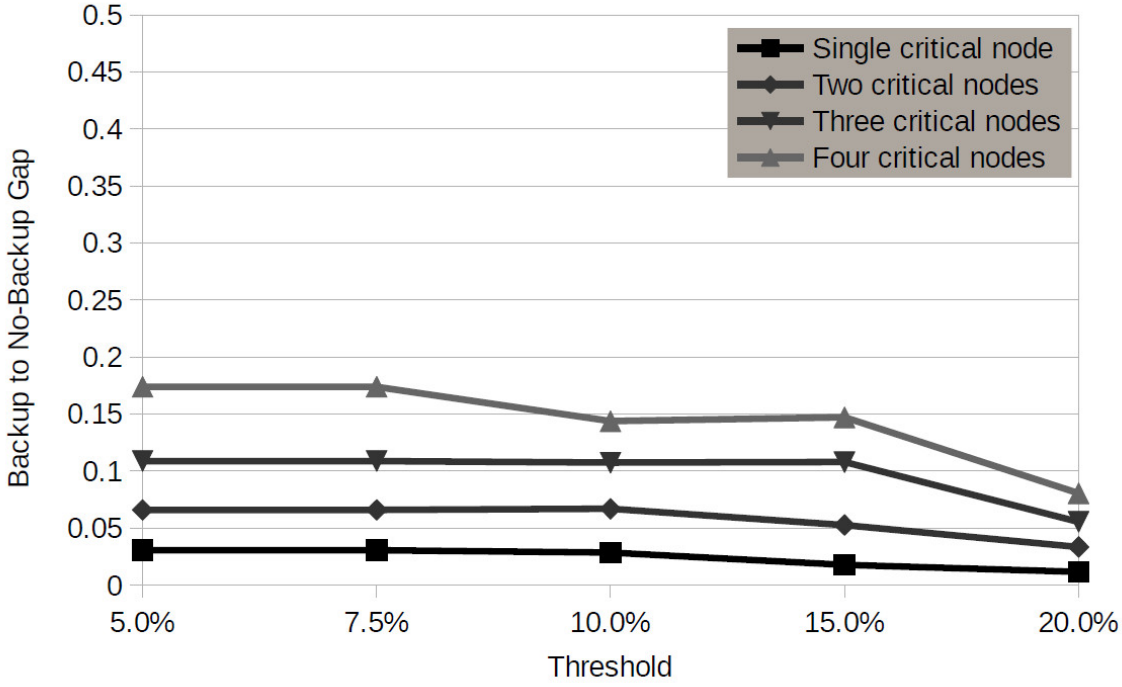


Figure 6.1: Increase on distance values, obtained by the heuristic, when backup is provided.

Regarding the effectiveness of pos-optimal (PO) procedures, and also due to the fact that there will be many instances, the following gap is analysed:

$$Gap_i^{PO} = \frac{Distance_i^{NoPO} - Distance_i^{PO}}{Distance_i^{NoPO}} \tag{6.2}$$

where $Distance_i^{NoPO}$ is the distance value obtained when no pos-optimal procedures are used and $Distance_i^{PO}$ is the distance value obtained after applying pos-optimal procedures, considering a specific instance i. Results plotted in the following figures include the average of these gaps over all analysed instances.

Figure 6.1 shows the increase on distance values, obtained by the heuristic, when backup is provided to critical nodes. Results show that the increase on the distance values gets higher as the number of critical stations increases, as expected, reaching $\backsim 15\%$ for four critical nodes, while for a single node it reached $\backsim 2.5\%$. However, when the threshold gets more loose the gap between backup and no backup solutions reduces, whatever the number of critical nodes, as less route extensions are required for backup provisioning.

Please note that extra route duration in case of backup provisioning also means serving more stops meaning that this is not necessarily a disadvantage. In fact stop weights can be used at the backup inclusion step in order to extend routes using stops with more visiting needs. Therefore, the right balance between threshold, influencing route extension, and number of routes should be carefully planned in order to reduce the operational costs related with the number of route schedules, and vehicles, while providing high quality

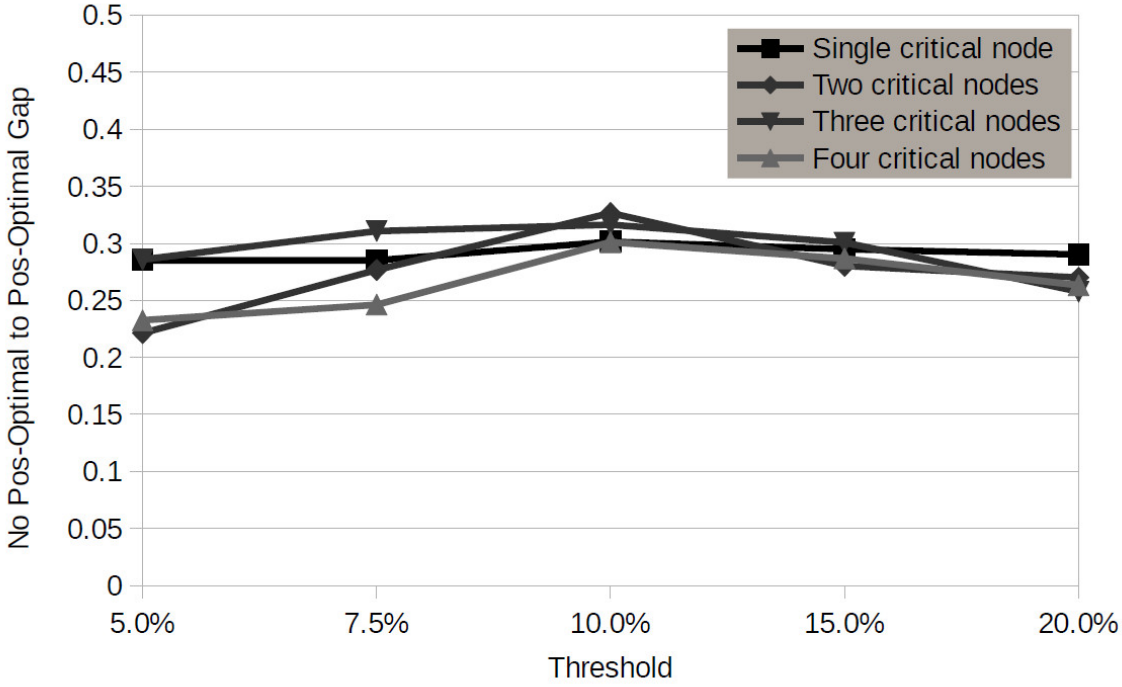


Figure 6.2: Decrease on distance values, obtained by the heuristic, when pos-optimal procedures are applied. Backup is being provided.

service and customer satisfaction.

Regarding the improvement on distance values, when using pos-optimal procedures, Figure 6.2 shows the decrease on the distance values obtained when pos-optimal procedures are applied, assuming that backup is being provided to critical stops. As it can be observed, pos-optimal procedures are very effective reducing the overall route lengths in approximately 30%. When no backup is provided the distance value reduction, which is independent of the number of critical stops and threshold, was also 30%.

It is also possible to conclude from Figure 6.2 that the optimal threshold value is around 10%. More tight thresholds result into lower pos-optimal procedure improvements. For threshold values higher than 10% the decrease of pos-optimal procedure improvements is low. Since tight thresholds lead to longer route extensions, while loose thresholds result into few or no route extensions, we can state that the insertion of just a few stations for route extension is enough to reach the best improvements. No station insertion means less flexibility when making exchange/relocation, while the insertion of too many stations result into longer routes and exchange/relocation is not able to compensate such increase.

Figure 6.3 shows the average of the ratios between the time it takes for a vehicle finishing its route somewhere in the neighbourhood, and providing backup, to come to the critical station and the time it takes for a new vehicle to come from the depot, so that

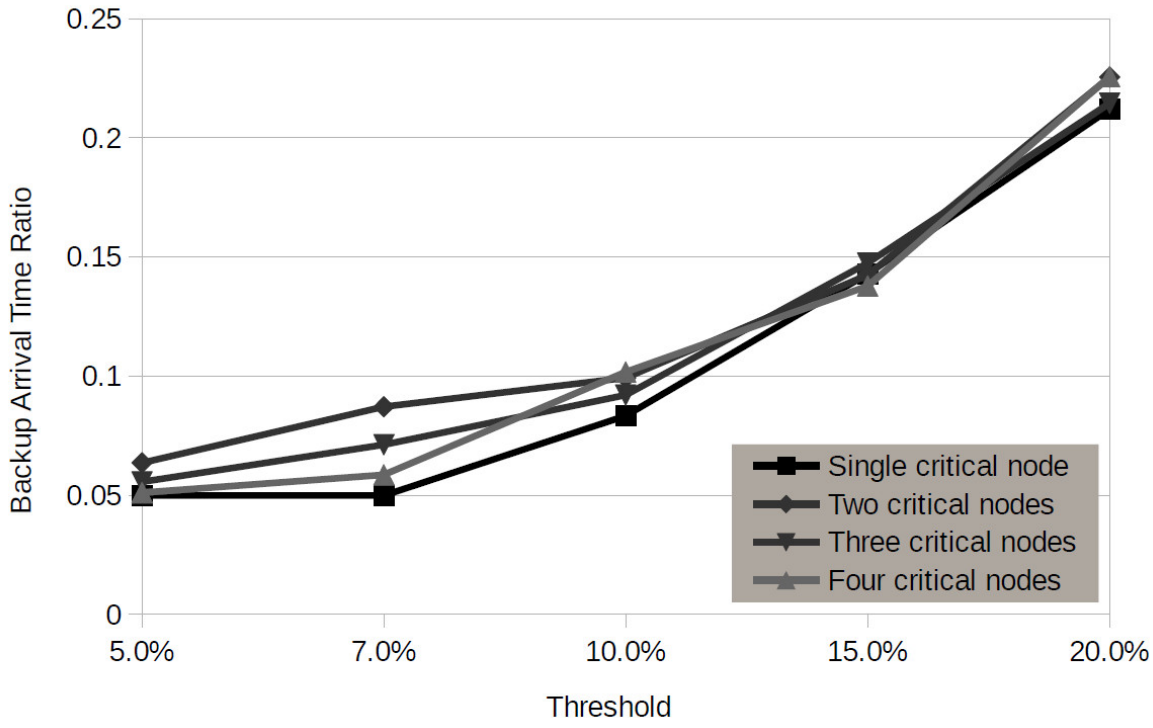


Figure 6.3: Backup provisioning vs vehicle from depot time radio.

extra clients are accommodated. When looking into this plot we can observe that backup provisioning times are always lower than the times associated with the routes that would be traversed if a new vehicle comes from the source. Note also that, in this last case, a new schedule, vehicle and driver needs to be introduced. In case of backup provisioning there is no need for a new schedule, extra vehicle and driver.

When observing this last plot we can state that higher threshold values result into a more pronounced increase of ratio values because not only backup takes more time to arrive but also routes become shorter. It is also possible to observe that since the threshold for a critical node is a percentage of the route duration, where the critical node is inserted, more critical nodes do not necessarily lead to a smoothing of the average values plotted, at least for lower threshold values. This would be the case if similar absolute thresholds were used for all critical nodes.

6.1.3 Overall perception

From the previous analysis done, the overall perception is that planning routes with backup provisioning can allow fast response to overload while reducing costs since:

- Backup provisioning allows fast response to overload and reduces waiting times for customers;
- Extension of routes to incorporate a backup can be done while serving stops with

more needs;

- Backup provisioning avoids more schedules/vehicles/drivers, reducing costs;
- The extra traversal by a vehicle providing backup is only done in case of need, namely when sensors detect overcrowding in a critical stop. This is flexible and cost effective solution for transportation network enhancement.

CHAPTER 7

Conclusions and future work

In this thesis a vehicle routing with backup provisioning approach is proposed for sustainable urban mobility with efficient use of resources. Besides mathematically formalizing the problem, a heuristic is also proposed that allows good solutions to be obtained faster. This heuristic incorporates pos-optimal procedures that significantly improve the quality of the initial solutions obtained by the heuristic. Results show that vehicle routing with backup provisioning can be a very flexible and cost effective solution for transportation network enhancement, while improving the quality of experience for customers.

The overall perception from these results is that planning routes with backup provisioning can allow fast response to overload while reducing costs since:

- Extension of routes to incorporate a backup can be done while serving stops with more needs;
- Backup provisioning from a vehicle in the neighbourhood allows fast response to overload and reduces waiting times for customers;
- Backup provisioning from a vehicle in the neighbourhood avoids having more schedules, and therefore vehicles and drivers, reducing costs. Also, fleet vehicles can be smaller since an empty vehicle in the neighbourhood will arrive in case of overload;
- The extra traversal by a vehicle providing backup is only done in case of need, namely when sensors detect overcrowding in a critical stop.

From such discussion we do believe that transportation network enhancement using sensors can bring real benefits as smart vehicle routing decisions, like reacting to overload, can be done.

For future work we expect to:

- *Add features to monitoring tool*: Show the new routes incorporating backup provisioning, time comparison with previous routes and allow companies to interchange data. Also, the possibility of assigning schedules and drivers can be added.

- *Heuristic*: Improve the insertion of the remaining stops (last step before backup inclusion).
- *Higher data sets*: Test higher data sets and analyse results. Different data sets may produce different optimal threshold values.
- *Real tests*: On the field implementation to see data collecting and real optimization of routes, in a big fleet scenario.

References

[1] B. City. http://www.cityofboston.gov/publicworks/lighting/led.asp, 2015. Accessed: 07/07/15.

[2] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[3] J. Zheng and A. Jamalipour, eds., *Wireless Sensor Networks, a Networking Perspective*. Wiley-IEEE Press, 2009.

[4] M. Naphade, G. Banavar, C. Harrison, J. Paraszczak, and R. Morris, "Smart cities," Tech. Rep. 6, IEEE Computer Society, Three Park Avenue, 17th Floor, New York, June 2011.

[5] C. qing Cai, Z. Zhang, and S.-D. Ji, "The intelligent bus scheduling based on zigbee," in *Computer Science Education (ICCSE), 2012 7th International Conference on*, pp. 1002–1005, 2012.

[6] Z. Wu, H. Chu, Y. Pan, and X. Yang, "Bus priority control system based on wireless sensor network (wsn) and zigbee," in *Vehicular Electronics and Safety, 2006. ICVES 2006. IEEE International Conference on*, pp. 148–151, 2006.

[7] G. Lavanya, W. Preethy, A. Shameem, and R. Sushmitha, "Passenger bus alert system for easy navigation of blind," in *Circuits, Power and Computing Technologies (ICCPCT), 2013 International Conference on*, pp. 798–802, 2013.

[8] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: a survey and taxonomy," *Communications Surveys Tutorials, IEEE*, vol. 11, pp. 19–41, Fourth 2009.

[9] L. C. Yeun, W. R. Ismail, K. Omar, and M. Zirour, "Vehicle routing problem: Models and solutions," *Journal of Quality Measurement and Analysis*, vol. 4, no. 1, pp. 205–218, 2008.

[10] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: models and algorithms," *Annals of Operations Research*, vol. 153, no. 1, pp. 29–46, 2007.

[11] F. Li, B. Golden, and E. Wasil, "The open vehicle routing problem: Algorithms, large-scale test problems, and computational results," *Comput. Oper. Res.*, vol. 34, pp. 2918–2930, Oct. 2007.

[12] J. Liu and Y. Fang, "Urban traffic control system based on wireless sensor networks," in *Information Acquisition, 2006 IEEE International Conference on*, pp. 295–300, 2006.

[13] L. Pun-Cheng, "An interactive web-based public transport enquiry system with real-time optimal route computation," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, pp. 983–988, June 2012.

[14] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, "A line in the sand: A wireless sensor network for target detection, classification, and tracking," *Comput. Netw.*, vol. 46, pp. 605–634, Dec. 2004.

[15] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, "Hardware design experiences in zebranet," in *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, (New York, NY, USA), pp. 227–238, ACM, 2004.

[16] G. Simon, M. Maróti, A. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton, "Sensor network-based countersniper system," in *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, (New York, NY, USA), pp. 1–12, ACM, 2004.

[17] K.-K. Yap, V. Srinivasan, and M. Motani, "Max: Human-centric search of the physical world," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, SenSys '05, (New York, NY, USA), pp. 166–179, ACM, 2005.

[18] Q. Fang, F. Zhao, and L. Guibas, "Lightweight sensing and communication protocols for target enumeration and aggregation," in *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*, MobiHoc '03, (New York, NY, USA), pp. 165–176, ACM, 2003.

[19] H. Yang and B. Sikdar, "A protocol for tracking mobile targets using sensor networks," in *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, pp. 71–81, 2003.

[20] W. Zhang and G. Cao, "Dctc: dynamic convoy tree-based collaboration for target tracking in sensor networks," *Wireless Communications, IEEE Transactions on*, vol. 3, no. 5, pp. 1689–1701, 2004.

[21] W. Zhang and G. Cao, "Optimizing tree reconfiguration for mobile target tracking in sensor networks," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 4, pp. 2434–2445 vol.4, 2004.

[22] J. Yick, B. Mukherjee, and D. Ghosal, "Analysis of a prediction-based mobility adaptive tracking algorithm," in *Broadband Networks, 2005. BroadNets 2005. 2nd International Conference on*, pp. 753–760 Vol. 1, 2005.

[23] S. Pattem and B. Krishnamachari, "Energy-quality tradeoffs in sensor tracking: Selective activation with noisy measurements," in *Proc., SPIE 17th Annual Intl. Symposium on Aerospace/Defense Sensing, Simulation, and Controls, (Aerosense '03)*, 2003.

[24] A. A. Absar-ul Hasan, Ghalib A. Shah, "Intrusion detection system using wireless sensor networks," in *Australian Journal EJSE*, pp. 90–99, 2010.

[25] H. Dai, Z. sheng Yang, and X.-G. Li, "Hierarchical intelligent control and coordination of urban traffic management systems," in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, vol. 2, pp. 1170–1174 Vol. 2, 2005.

[26] R. Jisha, M. Ramesh, and G. Lekshmi, "Intruder tracking using wireless sensor network," in *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*, pp. 1–5, 2010.

[27] V. Gutierrez, M. Izaguirre, J. Perez, L. Munoz, D. Lopez, and M. Sanchez, "Ambient intelligence in intermodal transport services: A practical implementation in road logistics," in *Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on*, pp. 203–209, 2010.

[28] Y. Wang, O. Ho, G. Huang, D. Li, and H. Huang, "Study on rfid-enabled real-time vehicle management system in logistics," in *Automation and Logistics, 2008. ICAL 2008. IEEE International Conference on*, pp. 2234–2238, 2008.

[29] Y. Wang, O. K. Ho, G. Q. Huang, and D. Li, "Study on vehicle management in logistics based on rfid, gps and gis," *International Journal of Internet Manufacturing and Services*, vol. 1, no. 3, pp. 294–304, 2008-11-09T00:00:00.

[30] B. Mrazovac, M. Bjelica, D. Kukolj, B. Todorovic, and D. Samardzija, "A human detection method for residential smart energy systems based on zigbee rssi changes," *Consumer Electronics, IEEE Transactions on*, vol. 58, no. 3, pp. 819–824, 2012.

[31] J. Lu, J. Gong, Q. Hao, and F. Hu, "Space encoding based compressive multiple human tracking with distributed binary pyroelectric infrared sensor networks," in *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2012 IEEE Conference on*, pp. 180–185, 2012.

[32] CodeIgniter. `https://ellislab.com/codeigniter`, 2013. Accessed: 03/09/13.

[33] PHPFrameworks. `http://www.phpframeworks.com/`, 2014. Accessed: 26/06/14.

[34] CakePHP. `http://cakephp.org/`, 2013. Accessed: 03/09/13.

[35] Y. Framework. `http://www.yiiframework.com/`, 2014. Accessed: 03/09/13.

[36] Leaflet. `http://leafletjs.com/`, 2013. Accessed: 03/09/13.

[37] ESRI. `http://www.esri.com/`, 2014. Accessed: 07/01/14.

[38] CoalSenses. `http://www.coalesenses.com/`, 2013. Accessed: 30/10/13.

[39] I. E. T. Force. `http://www.rfc-base.org/rfc-2460.html`, 2015. Accessed: 07/07/15.

[40] T. Caric, A. Galic, J. Fosin, H. Gold, and A. Reinholz, "A modelling and optimization framework for real- world vehicle routing problems," 2008.

[41] G. J.-F. Cordeau, M. Gendreau and Laporte, "A tabu search heuristic for periodic and multi-depot vehicle routing problems," *Networks*, vol. 30, pp. 105–119, 1997.

[42] `http://neo.lcc.uma.es/vrp/vrp-instances/multiple-depot-vrpinstances/`, 2015. Accessed: 13/12/14.

APPENDIX A

Development environment and setup

In this work we will use a iSense Security Sensor Module (`http://coalesenses.com`), which consists of a PIR and an accelerometer (Figure A.1). One main reason to work with these sensors is the possibility of setting up them according to our need, as we can change any code or parameters at deployment time. If any node is damage or with any problem, it is possible to replace the node by a new one and program it on site.

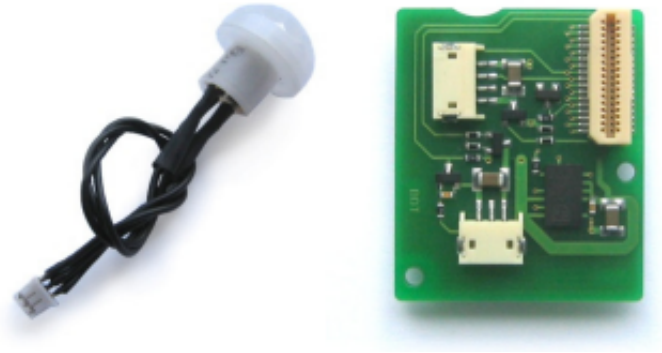


Figure A.1: Passive infrared.

To setup the environment we will need a Windows machine with Java 32 bits installed. The following steps have been performed:

1. Download and installation of Cygwin. During installation it is necessary to resolve Devel/make, Devel/cmake and Devel/gcc-g++ packages, for Cygwin. At the end, if not done automatically, it is mandatory to add the cygwin/bin path to environment variables.

2. Install BA2 Compiler (Windows compiler for isense devices with JN5148 controller), which can be downloaded from:

   - `http://coalesenses.com`

   In downloads → iSense Deveploment Environment. It is necessary to add the BA2 Compiler bin folder to environment variables.

3. Before we start working with sensors we need to download the iSense directory structure (can be found in the coalesenses website inside downloads) at:

   - `http://www.coalesenses.com/download/isense_sdk.zip`

   Extract all content. We should have, inside isense_sdk folder, iApps (where we put our code for the sensors) and iSense (where we put the firmware of the sensors). Then the iSense firmware must be downloaded from:

   - `http://www.coalesenses.com/download/`

   Or inside:

   - `http://www.coalesenses.com/index.php/downloads/i-sense-firmware`

   The IPv4 or IPv6 firmware are available. The firmware must be extracted inside isense_sdk/iSense folder.

4. At the same place where we have download BA2 Compiler we can find iShell (used to flash, monitoring the sensors). Just create a iShell folder and copy, to the inside, ishell.exe file.

5. Before we connect our iSense sensors to iShell we need to download the Gateway Module 2 driver (Choose "setup executable" under Windows drivers) at:

   - `http://ftdichip.com/Drivers/VCP.htm`

6. To program the sensors we need Eclipse IDE C/C++. After download and install Eclipse, disable "Build Automatically" in "Project" menu bar.

At the coalesenses website, inside downloads section, we find sensors demo code and projects with detailed documentation (`http://www.coalesenses.com/index.php/downloads/hardware-docs-and-demos/`). To change and compile code using Eclipse IDE, we may download the demo (in our case we start using Security Sensor Module) and unzip the demo inside isense_sdk/iApps. In Eclipse import a project, choose your demo inside iApps. When importing, **do not choose** "copy to workspace" because the project needs to be inside isense_sdk\iApps. After import it's necessary to include paths from isense_sdk\iSense\src to the project.
At Appendix B we include our initial setup code for PIR sensor (Figure B.1) that already includes a counter (incremental for entries and decremental for exits). The PIR has a non-detection period after a detection/reaction event, which is set to 2 seconds by default. In our experiments we tested with this default value later on we have set it to 1 second. This value cannot be too small neither too long since, in a simple implementation, a single

person could be counted as many (if too short) or many persons could be counted as a single one (if too large). In our case we intend to overcome this problem by using a dual PIR implementation, this way a person will not be counted more than once if the neighbour PIR did not detect any radiation. In this case a small value will be of more interest.
After this initial setup of the PIR sensor, code has been developed for communication between the sensor node and a sink or gateway. In relevant Figure B.2 we define some variables (like MAC addresses), and three relevant methods:

- Boot - present in every sensors, is executed every time the sensor is powered on. In this method we must initialize PIR.
- Execute - this method is used to send our data.
- Receive - called every time a message is received.

APPENDIX B

ISense security module

B.1 PIR setup

```
        // ----- configure PIR sensor -------------
        // set this application as the sensor event handler
        // --> handle_sensor will be called upon a PIR event
        pir_->set_sensor_handler(this);
        //set the PIR event duration to 2 secs
        pir_->set_pir_sensor_int_interval( 1000 );

        // switch on the PIR sensor
        pir_->enable();

        count_ = 0;
    } else
        os().fatal("Could not allocate sensors/iShell interpreter");
}

//--------------------------------------------------------------
void
SecurityModule::
handle_buffer_data( BufferData* buf_data )
{

//--------------------------------------------------------------
void
SecurityModule::
handle_sensor()
{
    //each time the PIR receives something a debug msg is send so we

    count_ ++;
    os().debug("PIR detected: count %d",count_);
    /*
```

Figure B.1: PIR setup

B.2 Sensor communication

```
#include <isense/application.h>
#include <isense/os.h>
#include <isense/uart.h>
#include <isense/modules/ethernet_module/net10_module.h>
#include <isense/modules/ethernet_module/enc28j60.h>
#include <isense/task.h>
#include <isense/Dispatcher.h>

#define SENSOR_ADDR 0x00158d00001c8888ULL
#define USER_PC 0x2c4138a83125ULL
#define GATEWAY_ETH_ADDR 0x000008002784f018ULL
#define GATEWAY_RADIO_ADDR 0x00158d0000149badULL

using namespace isense;

class Gateway:
    public Application,
    public Receiver,
    public Task
{
    public:
        Gateway (Os& os);
        virtual ~Gateway ();
        virtual void boot (void);
        virtual void execute (void* userdata);
        virtual void receive (link_layer_length_t len, \
                const uint8 * buf, \
                ISENSE_RADIO_ADDR_TYPE src_addr,\
                ISENSE_RADIO_ADDR_TYPE dest_addr,\
                uint16 signal_strength,\
                uint16 signal_quality, uint8 sequence_no,\
                uint8 interface, Time rx_time);
    private:
        Net10Module* netMod;
        Enc28J60* ethIf;
        int8 temp;
        uint32 lum;
};
```

Figure B.2: Gateway communication