UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

SOFTWARE DEFINED WIRELESS BACKHAULING FOR 5G NETWORKS

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Marc Franch Isart

In partial fulfilment

of the requirements for the degree in

TELECOMMUNICATIONS TECHNOLOGIES AND SERVICES ENGINEERING

Advisors:

Josep Vidal Manzano

Adrián Agustín de Dios

Barcelona, June 2018

Abstract

Some of the important elements to guarantee a minimum level of performance in a network are: i) using an efficient routing of the data traffic and, ii) a good resource allocation strategy. This project proposes tools to jointly optimise these elements in an IEEE 802.11ac-based wireless backhaul network considering the constraints derived from an implementation in a software defined network. These tools have been designed using convex optimisation‘s theory in order to provide an optimal solution that ensures a circuit-mode routing where the impact in higher and lower layers of the network is considered. Additionally, the traffic dynamics of the network is controlled by a sensitivity analysis of the convex problem using the Lagrange multipliers to adapt the solution to the changes produced by the evolution of the traffic. Finally, results obtained using the proposed solutions show an improved performance in bit rate and end-to-end delay with respect to typical routing algorithms for both simple and complex network deployments.

Resum

Alguns elements importants per assegurar uns nivells mínims de rendiment en una xarxa són: i) utilitzar un encaminament eficient del trànsit de dades i, ii) una bona estratègia en l’assignació de recursos. Aquest projecte proposa eines per optimitzar aquests elements en una xarxa de backhaul sense fils basada en el protocol IEEE 802.11ac considerant les restriccions derivades d’una implementació en una software defined network (xarxa definida per software). Aquestes eines han estat dissenyades utilitzant la teoria d’optimització convexa per tal de proposar una solució òptima que asseguri un encaminament en mode circuit on es consideri l’impacte en capes altes i baixes de la xarxa. A més, la dinàmica del trànsit de la xarxa es controla mitjançant una anàlisi de sensibilitat del problema convex utilitzant els multiplicadors de Lagrange per adaptar la solució a canvis de la xarxa produïts per l’evolució del trànsit. Finalment, els resultats obtinguts a partir de les solucions proposades demostren un millor rendiment en bit rate i latència extrem a extrem respecte a algoritmes d’encaminament típics tant en desplegaments de xarxes senzilles com més complexes.

Resumen

Algunos elementos importantes para asegurar unos niveles mínimos de rendimiento en una red son: i) utilizar un enrutamiento eficiente del tráfico de datos y, ii) una buena estrategia en la asignación de recursos. Este proyecto propone herramientas para optimizar estos elementos en una red de backhaul inalámbrica basada en el protocolo IEEE 802.11ac considerando las restricciones derivadas de una implementación en una software defined network (red definida por software). Estas herramientas han sido diseñadas utilizando la teoría de optimización convexa para proponer una solución óptima que asegure un enrutamiento en modo circuito en el que se considere el impacto en capas altas y bajas de la red. Además, la dinámica del tráfico de la red se controla mediante un análisis se sensibilidad del problema convexo utilizando los multiplicadores de Lagrange para adaptar la solución a cambios de la red producidos por la evolución del tráfico. Finalmente, los resultados obtenidos a partir de las soluciones propuestas demuestran un mejor rendimiento en bit rate y latencia extremo a extremo respecto a algoritmos de enrutamiento típicos tanto en despliegues de redes sencillas como más complejas.

Acknowledgements

I would like to thank my advisors Josep Vidal and Adrián Agustín for their assistance during the whole project and for offering me the possibility to work with them. Being able to meet weekly allowed me to avoid getting stuck in the problems that appeared during the project and to use their valuable suggestions to solve these problems. It has been really useful for me to be able to spend hours every week discussing the details of the achievements that were made and to ask for help when I needed it.

Finally, I want to thank Daniel Camps from i2CAT for the time he spent in the meetings we held and for allowing us to understand how to achieve a better approach in the routing strategy.

Revision history and approval record

Revision	Date	Purpose
0	22/03/2018	Document creation
1	22/03/2018 - 13/06/2018	Document writing
2	15/06/2018	Document revision
3	16/06/2018- 24/06/2018	Document writing
4	25/06/2018	Document revision
5	25/06/2018- 27/06/2018	Document writing
6	28/06/2018- 29/06/2018	Document revision and writing

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Marc Franch Isart	mfranchisart@gmail.com
Josep Vidal Manzano	josep.vidal@upc.edu
Adrián Agustín de Dios	adrian.agustin@upc.edu

Written by:		Reviewed and approved by:	
Date	29/06/2018	Date	28/06/2018
Name	Marc Franch Isart	Name	Josep Vidal Manzano Adrián Agustín de Dios
Position	Project Author	Position	Project Supervisor

Table of contents

Abstract 1
Resum 1
Resumen 1
Acknowledgements 2
Revision history and approval record 3
Table of contents 4
List of Figures 6
List of Tables 7
1. Introduction 8
1.1. Motivation 8
1.2. Statement of purpose 9
1.3. Requirements and specifications 9
1.4. Methods and procedures 10
1.5. Work plan, milestones and Gantt diagram 10
1.6. Description of the deviations from the initial plan and incidences 10
1.7. Methodology 11
1.8. Competences 12
2. State of the art 13
2.1. Departing from a datagram-based routing algorithm 13
2.2. Benefits of using a convex-based optimisation problem 13
3. Routing and resource allocation in wireless interference 802.11-based networks 13
3.1. Initial steps 13
3.1.1. System model 14
3.1.1.1. Network topology 14
3.1.1.2. Traffic in the network 15
3.2. Common routing algorithms 15
3.2.1. Second-rate version of the Shortest Path routing algorithm (SRSP) 15
3.2.2. Modified Shortest Path routing algorithm (MSP) 15
3.2.3. Effect of the multiflow (MuF) 16
3.3. Circuit-mode routing proposed algorithm (MA) 16
3.3.1. Problem definition 16

3.3.2. Grouping unsplittable flow 16

3.3.3. Communication variables 17

3.3.4. Problem formulation 18

3.3.4.1. Cost function: Allocated power minimisation 19

3.3.4.2. Constraints from grouping unsplittable flows 19

3.3.4.3. Constraints from the communication variables 19

3.3.5. Structure of the algorithm 20

3.3.5.1. Maximum power constraint and selection of links for a given flow 21

3.4. Addressing latency 22

3.4.1. Ensuring a maximum end-to-end delay per flow (DMA algorithm) 22

3.4.2. Minimising the maximum link utilisation 23

3.5. Impact of the algorithm results in overall network performance 24

3.5.1. Impact in TCP layers 24

3.5.2. Impact in lower layers 26

3.6. Updating the solution with traffic dynamics 27

3.6.1. Fluctuation in the bit rate 27

3.6.2. Adding and removing flows 30

4. Results 30

4.1 Simulation and results for the typical circuit-mode routing algorithms 30

4.2 Simulation and results for the main algorithms 31

4.2.1. Suitability of the algorithms in different deployments 32

5 Budget 35

6 Conclusions and future development 36

Bibliography: 38

Appendices: 40

Appendix 1: OpenDaylight 40

Appendix 2: Work plan, milestones and Gantt diagram 40

Appendix 3: Comparison of the maximum link utilisation using MA and the algorithm in 3.4.2 46

Appendix 4: Maximisation of the bit rate while ensuring a maximum end-to-end delay per flow 47

Appendix 5: Results for the power allocation using each algorithm 48

Glossary 50

List of Figures

Figure 1 Scenario considered with the BSs and the controller 9

Figure 2 Block diagram for the aimed dynamic algorithm proposed in section 3 12

Figure 3 Example of a simple topology 14

Figure 4 Block diagram of the inputs and outputs of the main algorithm 20

Figure 5 Pseudocode main algorithm 22

Figure 6 Diagram of the algorithm and the checker 26

Figure 7 Real value of the cost function vs Lagrange multipliers' approximation 28

Figure 8 Pseudocode of the algorithm to maximise the bit rate ensuring an average end-to-end delay per flow 29

Figure 9 Comparison between circuit-mode and datagram routing algorithms 31

Figure 10 Simulated deployment with 30 routers placed in the Sants area 32

Figure 11 Normalised histogram of the flows not routed using SRSP 32

Figure 12 Normalised histogram of the flows not routed using MSP 33

Figure 13 Normalised histogram of the flows not routed using DMA 33

Figure 14 Cumulative distribution function for the bit rate 34

Figure 15 Cumulative distribution function for the average end-to-end delay per flow 35

Figure 16 Gantt diagram 45

Figure 17 Maximum link utilisation comparing MA and algorithm in 3.4.2 46

Figure 18 Topology used for the simulation 47

Figure 19 Bit rate evolution 47

Figure 20 Average end-to-end delay per flow evolution 48

Figure 21 Cumulative distribution function for the power allocation 48

List of Tables

Table 1 Incidence-solution table 11
Table 2 Summary of the differences in the bit rate for each algorithm 34
Table 3 Costs separated by main tasks 36
Table 4 Work package 0 41
Table 5 Work package 1 42
Table 6 Work package 2 42
Table 7 Work package 3 43
Table 8 Work package 4 43
Table 9 Milestones table 44
Table 10 Configuration for the flows routed.. 47

1. Introduction

1.1. Motivation

Recent studies [1] have mentioned that 5G will bring a huge increase of traffic demand to the mobile networks and this is why it is fundamental to have a high capacity and dynamic backhaul which meets the demands of the network and provides with the necessary resources instead of having a static allocation designed for peak traffic conditions. As 5G technology will be a reality in the following years, guaranteeing a target end-to-end delay is one of the key aspects of 5G with views to accomplish given key performance indicators (KPIs). These typical KPIs for 5G are latency, network energy efficiency, connection density, area traffic capacity, peak data rate, user experienced data rate, spectrum efficiency and mobility. Then, the motivation of this project is the need to jointly optimise routing and allocation of resources in backhaul networks to be able to satisfy the KPIs requested to 5G.

For joint optimisation of routing and resource allocation, it is fundamental to consider a software defined network (SDN) [2] that constantly monitors the network status, performs the computations required and updates the network parameters. SDN is a networking architecture which is called to be a key tool in 5G. It was invented to improve the traditional network operation, where the nodes (i.e. routers) could make control decisions without knowing the overall performance of the network. In SDN, it is this controller (see appendix 1) who takes the decisions and communicates them to the nodes, separating the control plane (the responsible for making the network routing decisions) and the data plane (the responsible for the forwarding of the data). This is a huge advantage because the optimisation of the network can be done having an overview of its whole performance and the network intelligence is centralised in a software-based controller [3]. Then, the optimisation can be done by having a controller monitoring the network and computing the optimal allocation of resources and the paths for the traffic. Once obtained, it communicates all this information to the routers and the network is configured.

The SDN architecture can be applied to the common wired networks as well as to the more affordable wireless backhaul network. The densification in base stations (BSs) that is being proposed for 5G makes it hard that all the BSs can be connected using optical fibre. A wireless approach could solve this situation. However, this wireless implementation adds additional challenges related to the interference management and practical constraints related to the protocol employed to interconnect the access points of the network. On the other hand, having wireless links allows placing the network in a scenario like a city centre, with an important amount of generated traffic. The routers of the network can be located in lampposts and 5G access points can be connected to them to be able to receive the traffic generated by user terminals. Therefore, a node can be seen as a traffic source (place where traffic is injected) or a traffic sink (where traffic goes to).

1.2. Statement of purpose

This project aims to provide a joint routing and radio resource allocation algorithm for a wireless backhaul network taking into account the practical constraints that an implementation in a real environment SDN would have. The research institution i2CAT has played an important role in defining this project because the work done is aimed to be testable in the future in their infrastructure using protocol IEEE 802.11ac and its specifications. To clearly understand the project's aim, this section will contain an introduction of the main tools used in the project. First, a backhaul network is defined as the network that connects the operators' BSs (access network) and their core networks, and carries traffic both in uplink and downlink directions. If the backhaul network is shared by several operators, the controller of the SDN network is managed by the so-called infrastructure provider (InP) which monitors and allocates resources to different mobile network operators (MNOs) in a transparent way, by dividing the resources into network slices, one for each MNO (see Figure 1[1]).

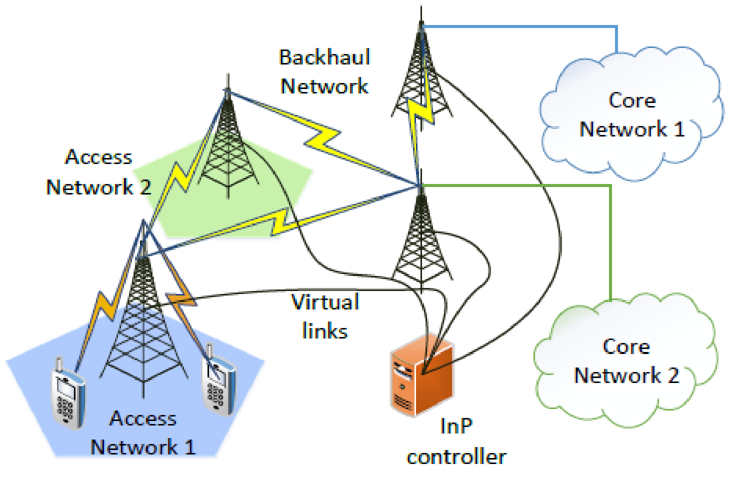


Figure 1 Scenario considered with the BSs and the controller

From this, the project's purpose is to provide an optimal algorithm that generates slices, as defined by routes and radio resources in a dynamic backhaul network, given the netwoek status, connectivity and traffic specifications. The aim is to propose an algorithm that controls the delay of the traffic in the network to ensure a minimum level of throughput performance, as explained in 1.1, while the radiated energy consumption is minimised. To design the optimal routing algorithm, convex optimisation is going to play a strong role as it determines the whole formulation of the optimisation problem. Additionally, the theoretical approach of the project is to be later tested in a typical real environment network to provide the practical approach by checking the algorithm's dynamic capabilities as a function of the traffic demand. On the other hand, the theory contained in the project will be able to control the impact of the solution found in higher and lower layers, changing the obtained solution to fit with their constraints.

1.3. Requirements and specifications

This project's requirements are:

- All the work has to be adapted to the practical constraints given by a real environment implementation (e.g. the need of having a circuit mode routing algorithm rather than a datagram one) and the optimization problem and its solution have to be understood under these constraints.
- The project must be developed considering that the network has a given number of flows. Each flow will be characterised by a pair origin-destination depending if it is considered

[1] Image taken from [4]

uplink or downlink. Following the circuit-mode routing of 802.11-based networks, each flow will follow just one path.
- The optimal routing algorithms proposed must aim at ensuring a maximum end-to-end delay per flow as it is required in some 5G use cases.
- KPIs (listed in 1.1) will be computed in the results section to compare between routing algorithms and to justify the suitability of the proposed algorithm.
- The offered solutions will take into account the traffic dynamics and the network state. Once the problem is solved and a parameter of the network changes, we aim at evaluating the incremental changes in network parameters.

The specifications are the following:

- The wireless network assumes links working with the IEEE 802.11ac [5] standard (WiFi) under a TDD mode and a carrier frequency of 5 GHz. All these parameters have been set under i2CAT proposal.
- The algorithms will read/write the SDN controller to implement the solutions obtained. This means that the algorithm could be implemented outside the controller's device and make the computations remotely.
- There will be a limited available total power for each link of the network and the results provided by the algorithm cannot exceed this power in any case.
- The algorithm has to consider the impact that higher layers (transmission control protocol or TCP) can have in the traffic of the network and provide a feasible bit rate.

1.4. Methods and procedures

This project is a continuation of a previous one [6] carried out at the SPCOM group of the Signal Theory and Communications (TSC) department of UPC. It belongs to one of the lines of work [4] that have been followed since 2016. In [6], a routing algorithm is proposed using optimisation techniques but the formulation and solution proposed cannot be used in a realistic environment because some practical constraints were not solved. The solution in [6] provides a routing algorithm with a datagram approach, something that this project will not. In real networks, datagram approaches involve having high overheads in the transmitted packets to be able to later put them in order if one packet uses a faster path than another one transmitted before.

Following a circuit mode approach has a big impact in reformulating the problem in [6] because a convex and optimal algorithm has not a direct implementation under this assumption. However, it is important to propose an algorithm with this approach to achieve the motivation of testing the algorithm in the infrastructure available in i2CAT.

1.5. Work plan, milestones and Gantt diagram

The detail of the Work Plan, milestones and the Gantt diagram of the project can be reviewed in the second appendix included.

1.6. Description of the deviations from the initial plan and incidences

The following table shows the deviations and incidences faced:

Initial plan	Incidence	Solution
Provide an algorithm that minimised the delay of the network.	When all was designed, the Python solver used to compute the solution could not provide a stable solution (details in the second appendix). **Critical incidence**	The algorithm's aim changed and a more practical objective was set: ensuring a maximum end-to-end delay per flow.
Optimising the allocation of bandwidth as done in [6].	To have a convex and circuit-mode orientated routing algorithm, bandwidth could not be optimised. **Moderate incidence**	To optimise the allocation of power and consider a given amount of bandwidth per link.
Testing the results in i2CAT's platforms.	i2CAT do not have the platforms installed yet and plan to have them by September. **Moderate incidence**	MATLAB's simulations have been performed instead and a more detailed study of traffic dynamics has been done.

Table 1 Incidence-solution table

1.7. Methodology

Once the objectives, the requirements and the specifications have been explained, the methodology followed in the TFG is explained and mapped to the sections of this document. The state of the art is going to be reviewed in section 2. Section 2 will contain the necessary background for the main topics of the project, going from a review of the work in [4] to a review of the mathematical tools used in the project that have been also used in previous studies to analyse optimisation problems. Then, in section 3, the routing algorithm that takes into account the wireless backhaul network following an SDN architecture is proposed. Section 3 will start with the previous steps to consider, where available MATLAB tools have been used to design less complex circuit-mode algorithms. These less complex algorithms will be useful to have a reference to compare results with the ones obtained by the algorithms proposed later in the project.

Also in section 3, the main algorithm of the project is formulated using a convex optimisation problem (justified in section 3.3) and the practical constraints are included. This part of the section will explain the theory behind the optimisation and justify the steps taken in order to let the reader understand why it has been done in that way. The main algorithm has been implemented using a set of variables defined in this document and the interaction between them follows the standards of disciplined convex programming [7]. After that, the different approaches derived from the main algorithm are also explained. These two approaches use the mathematical tools derived from convex optimisation and the impact that a quasi-convex cost function would introduce to the problem.

Additionally, section 3 presents a subsection for the analysis of the traffic dynamics which is performed using the Lagrange multipliers associated with the constraints of the convex problem in what is called a sensitivity analysis. The impact in the different communication layers is also considered in section 3 using background in the topic. First, a convex problem to simulate the impact of TCP is going to be used for higher layers and then the impact of the reuse of frequency bands and colouring algorithms is going to be analysed for lower layers. Finally, the project will end with the results obtained (section 4), where the less complex circuit-mode routing algorithms from section 3.2 that use the Shortest Path routing algorithm will be compared with the proposed solutions. These less complex algorithms will use a static allocation of resources designed for peak conditions, what means that all the available transmission power will always be used and a sequential routing of the flows is adopted, while the more complex algorithm proposed will be using an optimised allocation of power together with an optimal solution for the paths computed (taking all the flows into account at the same time). An analysis of the economical part is included in section 5 and the conclusions of the project in section 6.

The subsections of section 3 are linked and summarised in the following block diagram:

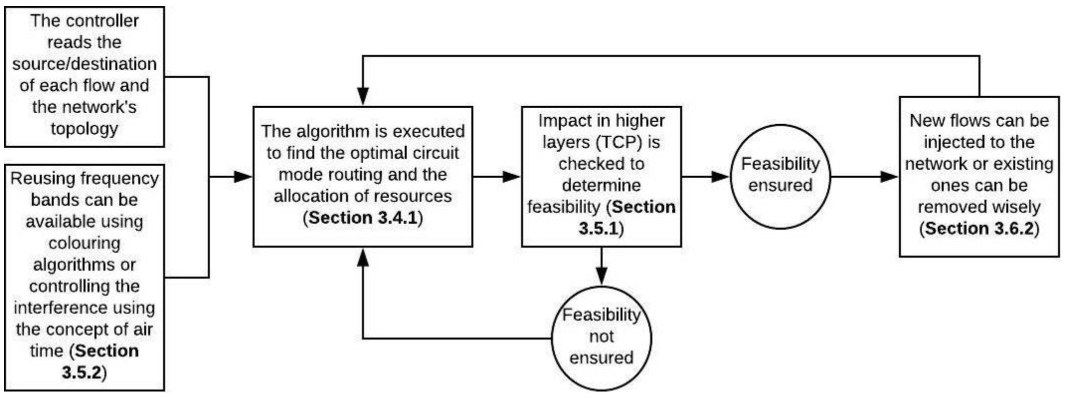

The controller reads the source/destination of each flow and the network's topology
Reusing frequency bands can be available using colouring algorithms or controlling the interference using the concept of air time (Section 3.5.2)
The algorithm is executed to find the optimal circuit mode routing and the allocation of resources (Section 3.4.1)
Impact in higher layers (TCP) is checked to determine feasibility (Section 3.5.1)
Feasibility ensured
Feasibility not ensured
New flows can be injected to the network or existing ones can be removed wisely (Section 3.6.2)

Figure 2 Block diagram for the aimed dynamic algorithm proposed in section 3

After reviewing all the sections in the document, the reader should be able to answer questions such as whether is suitable to allow multiflow in a network or not (section 3.2.3), how higher layers affect to routing algorithms (section 3.5.1) or how the appearance or disappearance of traffic in the network can be efficiently handled (section 3.6.2).

1.8. <u>Competences</u>

The work in this project has been done after a research on the needed topics. Although the project contains all the information to be understandable, the reader should know that the competences needed include theory on convex optimisation [7], knowledge of the Lagrange multipliers and the Lagrangian [7], MATLAB's coding language, MATLAB's library *CVX*, understanding SDN networks [2], basics of 802.11ac protocol [5], general knowledge of graph theory and knowledge of congestion control mechanisms [9].

2. State of the art

2.1. Departing from a datagram-based routing algorithm

The theory contained in this project departs from the work done in [4], where an optimisation of the routing and the resource allocation was done for a wireless backhaul network without taking into account the practical constraints that a real implementation entails. As explained in [4], previous studies considered optimising the routing and the allocation of resources separately but in [8] it was proposed a joint optimisation with improved results and this project is going to follow this approach. On the other hand, the solution proposed in [4] is decomposed in the network-plane and the communications-plane, a useful tool that this project will also consider as it is advantageous when formulating the optimisation. The main idea of the problem is the same in [4] and in this project: there is a given network characterised by a given amount of nodes (switches or routers) and wireless links that interconnect them. The nodes of the network are defined either as BSs or the access gateway to the core networks of a given number of MNOs.

In [4], the solution proposed is optimal in terms of delay minimisation but in real scenarios datagram routing algorithms are not used (as explained in 1.4) and this project aims to give an alternative to this datagram approach. Although this project shares the same scenario as in [4], all the analysis done is completely different. The work done here can be seen as the direct implementation of an optimal algorithm in any SDN network similar to the scenario in [4]. Considering all the practical constraints was not done in [4], which has completely changed the problem. Now, a circuit-mode approach is ensured after considering the impact of all the aspects of the network.

2.2. Benefits of using a convex-based optimisation problem

Another important aspect of this project is the use of the Lagrange multipliers defined in the convex optimisation problem and associated to its constraints. Previous work, [9] and [10], has exploited the Lagrange multipliers (or dual variables) to obtain information and use it to modify parameters of the problem. This has been done because the multipliers give information about the sensibility of the cost function and about possible infeasible solutions. Particularly in [10], the Lagrange multipliers are used to analyse a dynamic scenario and the effect of introducing changes to the constraints of the convex optimisation problem. There, a problem with a single BS is considered and the Lagrange multipliers identify between the users in the BS the candidates to be rejected from the network when power limitations arise. This is called performing a sensitivity analysis and allows to identify the impact of the changes in the cost function of the problem but also to determine how to apply the changes in order to obtain the desired performance of the cost function.

3. Routing and resource allocation in wireless interference 802.11-based networks

3.1. Initial steps

As part of the project a meeting was held with a member of i2CAT who suggested the use of circuit-mode routing algorithms rather than datagram approaches to ensure its feasibility in conventional platforms. As explained in 1.4, one of the problems of datagram approaches, under the TCP protocol, is packet reordering at the destination node, what makes the receiving process harder and may increase end-to-end delays. In addition, as the project considers an SDN architecture, the circuit-mode routing is also suitable because in this type of architectures the flows of the network can only be routed using one path. Then, the project starts providing an analysis of this type of algorithms compared to datagram approaches as the one considered in [4]. This analysis should show a deterioration in the results, as separating the traffic in flows and not in packets gives less granularity and limits the flexibility of routing.

Before proposing the optimal algorithm, the comparison will be made using one of the typical circuit-mode routing algorithms as the main differences are even more evident and provide consistent results to understand the next steps of the project.

3.1.1. System model

This section explains the model of the network topology and traffic, as these are two aspects defined in the same way for the typical (section 3.2) and more complex (sections 3.3 and **¡Error! No se encuentra el origen de la referencia.**) algorithms. Then, to design and implement the algorithms, the model is as follows.

3.1.1.1. Network topology

The backhaul network (see Figure 3) is modelled with a directed graph [11] (meaning that each link has a single direction) with nodes (n=1,…, N) and links (l=1,…, L). From here two sets are defined, $O(i)$ contains all the outgoing links from node *i* and $I\ (i)$ the incoming links from node *i*. To define the topology of the network, a matrix $A \in \mathbb{R}^{N\times L}$ is defined as the incidence matrix. Each component of the matrix can be:

$$a_{i,k} = \begin{cases} 1, & if\ k \in O\ (i) \\ -1, & if\ k \in I\ \ (i) \\ 0, & otherwise \end{cases} \tag{1}$$

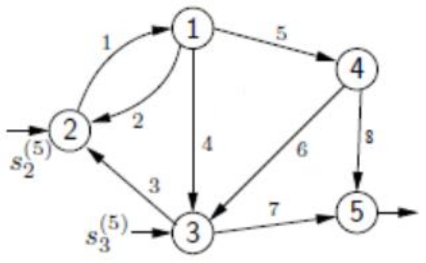


Figure 3 Example of a simple topology

$$\mathbf{A} = \begin{bmatrix} -1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \end{bmatrix} \tag{2}$$

Incidence matrix for topology in Figure 3

To have an easier control of variables, matrix A will be split in:

$A_{out} = \max(0, A)$, changing the negative components to 0 in order to contain only the outgoing links.

$A_{in} = -\min(0, A)$, changing the ones to 0 and the -1 to 1 in order to contain only the incoming links.

3.1.1.2. Traffic in the network

The traffic injected to the backhaul network is characterised by multiple source-destination pairs, each one identifying a single flow and containing the number of the source and destination nodes. Let the destination node be d, s_i^d will contain the flow injected in node i whose destination is node d, and a vector $\boldsymbol{s}_{\boldsymbol{i}}^{(d)} \in \mathbb{R}^{N\times 1}$ is defined containing the quantity of flow injected at the origin node (positive amount) and the destination node (negative amount). Additionally, this vector can be defined just for a destination node as $\boldsymbol{c}_{(d)} \in \mathbb{R}^{N\times 1}$, containing in a 1 the origin nodes sending a flow to destination d and a -1 for the destination node. Following the formulation given in [4], vector $\boldsymbol{x}^{(d)} \in \mathbb{R}^{L\times 1}$ is also defined for every destination $d \in D \subseteq N$ containing the total flow on each link going to node d. Finally, to limit the total traffic on each link k, a vector containing the capacity of the link is defined as $\boldsymbol{y} \in \mathbb{R}^{L\times 1}$ so that expression **(3)** relates both variables.

$$y_k \geq \sum_{d\in D} x_k^{(d)} \tag{3}$$

3.2. Common routing algorithms

First of all, after reviewing the theory explained in [12] a study of the delay is proposed considering typical circuit-mode routing algorithms, following the motivation of considering the delay as explained in section 1.1. As the work done in [4] is an optimisation of the delay and has the advantage of being a datagram approach, this algorithm should provide a lower bound for the rest of the results. In this way, different circuit-mode routing algorithms are proposed.

We introduce first the Shortest Path routing algorithm. This algorithm is one of the most typical algorithms in routing and assumes a topology modelled as in 3.1.1.1. Then, in Shortest Path each link is assigned a weight and an origin and destination nodes for a flow are set to be solved. What this algorithm does is to provide the path from the origin to the destination such that adding the weights of the used links the result obtained is the minimum. After all the typical circuit-mode routing algorithms are explained in this section, results for the study of the delay are included in section 4.1.

3.2.1. Second-rate version of the Shortest Path routing algorithm (SRSP)

While the lower bound routing algorithm in terms of delay has already been proposed [4], an upper bound strategy is proposed as an example of poor routing. To do that, a second-rate (simple) version of the Shortest Path routing algorithm is proposed which aims to solve the routing choosing the path with the minimum number of hops between source and destination, as the weight given to each link is equal to one. Then, in this second-rate version of the algorithm, it is assumed that all the wireless links provide equal performance and none is preferable amongst others. Additionally, to compute the delay that this strategy offers for a given number of flows, the allocated power and bandwidth for each link will be static and equal to P_{MAX} and $\frac{BW_{TOTAL}}{NUM\_LINKS}$ respectively.

3.2.2. Modified Shortest Path routing algorithm (MSP)

Once the lower and upper bounds are set, a more elaborated version of the SRSP algorithm is proposed. Now, the allocated power and bandwidth remain static and with the same values as for the SRSP, but different weights are going to be given to each link to prioritise the best links and to avoid the worst ones. Before computing a path the algorithm checks if the capacity of the used links is enough to route the flow, otherwise the links will be avoided. The initial weight for each link is the inverse of its capacity but when the link is already used by another flow, this weight is going to be updated using the remaining capacity in the link. In this way, links with more capacity will be selected over other links until another one has more remaining capacity. It should be noted that this algorithm is far from being optimal because the weights of the different links are based on the capacity that has been assigned to them, which has been computed with a static allocation while in the solution proposed in [4] the capacity is optimised.

3.2.3. Effect of the multiflow (MuF)

To obtain results closer to the datagram approach, an additional algorithm is proposed following the same static allocation. In this case the circuit-mode routing approach will be slightly modified to split each of the flows into half and obtain the subflows, following the idea of what is known in literature as multipath TCP [13]. Then, as each of these subflows can be routed in a different path, this adds flexibility to the paths selected and better results are expected as compared to the MSP.

3.3. Circuit-mode routing proposed algorithm (MA)

The network controller is continuously monitoring the network and runs the algorithm whenever it is necessary to solve the routing and the allocation of resources. As explained, this algorithm will work under the given practical constraints arising from a practical implementation in a real platform. As explained in section 3.1, the first aspect to solve is providing a circuit-mode routing algorithm that can be formulated using convex theory [7]. Convex optimisation guarantees a stable, single and optimal soluiton. This optimal solution has to be understood under the assumptions explained in previous sections and it is going to be justified in more detail in section 3.3.5.

3.3.1. Problem definition

The main algorithm proposed in this section of the document aims to provide a circuit-mode routing algorithm resulting from a convex optimisation problem that optimises the allocation of resources. To do that, the objective of the algorithm is to find a single path for each flow injected to the network and that is defined by a pair of nodes origin-destination, with the idea that the traffic demand of the network can be decomposed into multiple flows.

The algorithm proposed optimises the total power allocated to each link so as to provide the minimum level of performance requested. This amount of power can be translated into a minimum amount of capacity that each link needs through the Shannon capacity formula. Therefore, the capacity is also optimised according to the bit rate that each link is expected to carry when a configuration of flows is injected into the network. However, the bandwidth optimisation cannot be performed (as explained in section 3.3.3 below). The other aspect optimised in the algorithm is the routing of the different flows in contrast to what has been done in the algorithms in 3.2. Here all the flows in the network are jointly considered (not sequentially) to assign the path that best fits the bit rate of each flow but without penalising the rest of the flows. This is done by designing a variable to contain the paths of each flow as it is going to be explained in section 3.3.2.

3.3.2. Grouping unsplittable flow

After having defined the network topology and traffic model in 3.1.1, this part of the formulation defines the variables that guarantee that each flow can only be routed through a single path. First of all, in a situation with F flows, it is necessary to define the origin and destination nodes for each flow. To do that, variable $inflow \in \mathbb{R}^{F\times 1}$ is defined to contain the origin nodes of the F flows and the same is done with the destination nodes in $outflow \in \mathbb{R}^{F\times 1}$. Additionally, each flow carries a given bit rate that is defined in variable $\pi_f \in \mathbb{R}$. From this information the incidence flow vector can be defined for every flow too, following the same idea as vector $\boldsymbol{s}$ but now considering each flow separately:

$$b_{n,f} = \begin{cases} \pi_f \; if \; n = inflow\,(f) \\ -\pi_f \; if \; n = outflow\,(f) \\ 0 \; otherwise \end{cases} \tag{4}$$

Furthermore, as the vector $\boldsymbol{x}^{(\boldsymbol{d})}$ defined in section 3.1.1.2 only takes into account the destinations and a flow is defined by the pair origin-destination, it is useful to group all the flows going to the same destination. Thus, when a number of destination nodes num_D is given, the problem will work with num_D sets of flows where each set is created following the rule:

$$F_d = \{f \mid outflow(f) = d\} \tag{5}$$

To solve the problem for the circuit-mode approach, a variable similar to $\boldsymbol{x}^{(\boldsymbol{d})}$ needs to be designed but for each flow rather than only considering the destination, as this will add flexibility to the proposed constraints. To do so, $\boldsymbol{r}_f \in \mathbb{R}^{L\times 1}$ is defined containing in each component the bitrate of flow f on each link k, that in the used links will be equal to the bit rate of the flow. On the other hand, it is also necessary to define a binary variable indicating the usage of every link when a particular flow is taken into account. That is done with variable $\boldsymbol{z}_f \in \mathbb{R}^{L\times 1}$, indicating with a zero if a link is not used or a one if it is. It will be useful to relax the values of $\boldsymbol{z}_f$ so that each component of the vector follows:

$z_{k,f} \in \{0, z_{k,f}^{MAX}\}$, where $\boldsymbol{z}_f^{MAX} \in \mathbb{R}^{L\times 1}$ is a mask where $z_{k,f}^{MAX} = 1$ for flow f. While the main aim of the algorithm was to depart from a convex optimisation problem, it is a fact that working with binary variablea does not allow convexity. However, to address this issue, the algorithm will solve multiple convex problems as it will be detailed in section 3.3.5.

3.3.3. <u>Communication variables</u>

The communication variables of the formulation have an important role in the optimization problem as the allocation of these parameters from the physical layer are directly related to the performance of the algorithm. An important change with respect to the work in [4] is that using a circuit-mode approach together with a convex formulation prevents the optimisation of bandwidth (additional details on the bandwidth used in the algorithm are available at section 3.5.2 of the document). Although it may seem a big problem, it is not critical as in real testbeds it is difficult to be able to control its allocation. Then, in this problem formulation a vector $\boldsymbol{W}_{link} \in \mathbb{R}^{L\times 1}$ will contain the given values of the bandwidth allocated in each link assuming that a colouring algorithm has already been used applied and there is not interference.

To take into account the state of each wireless link, vector $\boldsymbol{\rho}_{lin} \in \mathbb{R}^{L\times 1}$ will also be specified by the user as the maximum signal to noise ratio for each link. This variable is assumed to be computed using:

$$\rho_l = \frac{1}{L\cdot N_0} * \frac{P_{maximum,l}}{BW_l} \tag{6}$$

As it contains the maximum radiated power per node (specified by the user) it makes sense to see it as the maximum signal to noise ratio. From here, the power that the algorithm is going to optimise can be understood as a normalised power that will modify the real signal to noise ratio in each link by multiplying the maximum value of SNR. Then, $\boldsymbol{P}^{link} \in \mathbb{R}^{L\times 1}$ will be the normalised power that the algorithm will allocate in each link as a result of the optimisation. However, to formulate the problem it is advantageous to define the second power-related variable $\boldsymbol{p} \in \mathbb{R}^{L\times F}$ that is going to contain the virtual power dedicated to each flow in each link if it was alone. It is also a normalised power but in this case it does not represent a power that can be measured in the real problem, it is just an instrument variable that allows to take into account each flow separately when optimising.

3.3.4. Problem formulation

Considering all the variables defined above, a convex optimisation problem is defined. Before trying to focus on the delay, this first algorithm optimises the allocation of the available resource, the power, while offering a circuit-mode approach. As a consequence, the convex optimisation problem that ensures a circuit-mode routing with an optimal and minimum allocation of the power is the following:

$$\underset{\boldsymbol{P}^{link},\boldsymbol{p},\boldsymbol{x}^{(d)},\boldsymbol{y},\boldsymbol{s}^{(d)},\boldsymbol{r}_f,\boldsymbol{z}_f}{\text{minimise}} \quad t$$

$$s.t.$$

$$\sum_k P_k + \sum_k \sum_f p_{k,f} - t \leq 0 \qquad \text{Communications}$$

$$\boldsymbol{z}_f \leq \boldsymbol{z}_f^{MAX}, \qquad \forall f$$

$$(\boldsymbol{A}_{out} - \boldsymbol{A}_{in})\boldsymbol{z}_f \leq 1$$

$$\boldsymbol{A}_{out}\boldsymbol{z}_f \leq 1$$

$$\boldsymbol{A}_{out}(\boldsymbol{inflow}(f),:)\boldsymbol{z}_f = \boldsymbol{1}$$

$$\boldsymbol{A}_{in}(\boldsymbol{outflow}(f),:)\boldsymbol{z}_f = \boldsymbol{1}$$

$$(\boldsymbol{A}_{out} - \boldsymbol{A}_{in})\boldsymbol{r}_f = \boldsymbol{b}_f$$

Grouping unsplittable flow

$$r_{k,f} - z_{k,f} W_k log_2(1 + \rho_k p_{k,f}/W_k z_{k,f}) \leq 0$$

$$\nu_{k,f} : -r_{k,f} + \pi_f z_{k,f} \leq 0 \qquad \textbf{(7)}$$

$$\lambda_{k,f} : -p_{k,f} + \frac{W_k}{\rho_k}\left(2^{\frac{\pi_f}{W_k}} - 1\right) z_{k,f} \leq 0$$

$$\sum_f \boldsymbol{p}_f - \boldsymbol{P}^{link} \leq \boldsymbol{0}$$

$$\sum_{f\in F_d} \boldsymbol{r}_f - \boldsymbol{x}^{(d)} \leq \boldsymbol{0}$$

$$\sum_{d\in D} x_k^{(d)} - y_k \leq 0$$

$$y_k - C_k(P_k^{link}) \leq 0$$

$$(\boldsymbol{A}_{out} - \boldsymbol{A}_{in})\boldsymbol{x}^{(d)} = \boldsymbol{s}^{(d)}$$

$$\boldsymbol{c}_{(d)}^T \boldsymbol{s}^{(d)} = 0$$

$$\boldsymbol{P}^{link} - t \leq 0$$

In this convex problem the variables that are computed from the optimisation are $\boldsymbol{P}^{link}$, $\boldsymbol{p}$, $\boldsymbol{x}^{(d)}$, $\boldsymbol{y}$, $\boldsymbol{s}^{(d)}$, $\boldsymbol{r}_f$, $\boldsymbol{z}_f$ and t. The explanation of the restrictions is found below and all the variables used are the ones defined.

3.3.4.1. Cost function: Allocated power minimisation

First of all the problem cost function is the addition of the sum of the power allocated in all the links and the virtual power per link and flow. By setting this cost function, the problem is performing a minimisation of the total power and can be placed with the rest of constraints by using variable t.

3.3.4.2. Constraints from grouping unsplittable flows

First, the constraint $\boldsymbol{z}_f \leq \boldsymbol{z}_f^{MAX}$ is used to ensure that $\boldsymbol{z}$ takes values allowed by the mask and, when the mask is set to zero for a given flow, then the link cannot be active. The following constraint is $(\boldsymbol{A}_{out} - \boldsymbol{A}_{in})\boldsymbol{z}_f \leq 1$ and, while it does not guarantee values of $\boldsymbol{z}_f$ equal to 0 or 1, it is used to stimulate a circuit-mode routing, making that the traffic injected to a node is extracted and transmitted. Then, constraint $\boldsymbol{A}_{out}\boldsymbol{z}_f \leq 1$ is used to guarantee that the output links of each node (taking into account the addition of $\boldsymbol{z}_f$ values) is less than or equal to one.

The following two constraints are $\boldsymbol{A}_{out}(\boldsymbol{inflow}(f), :)\boldsymbol{z_f} = \boldsymbol{1}$ and $\boldsymbol{A}_{in}(\boldsymbol{outflow}(f), :)\boldsymbol{z_f} = \boldsymbol{1}$, used to specify that the source and destination nodes should have an incoming and outcoming respectively equal to one. Then, two constraints have been designed to ensure the flow conservation law and they are $(\boldsymbol{A_{out}} - \boldsymbol{A_{in}})\boldsymbol{r_f} = \boldsymbol{b_f}$ and $(\boldsymbol{A}_{out} - \boldsymbol{A}_{in})\boldsymbol{x}^{(d)} = \boldsymbol{s}^{(d)}$. With them it is ensured that, considering both a single flow and all flows together, what it is transmitted to a node has to be retransmitted to other nodes except it is the origin or destination nodes.

Additionally, another constraint is proposed for the bit rate of each flow carried in the links. It is $-r_{k,f} + \pi_f z_{k,f} \leq 0$, a way of relating variable $r_{k,f}$ with the bit rate of a flow when the link is used and equal to zero when the link is not used. For this constraint a Lagrange multiplier is defined as $\nu_{k,f}$ and will be used later in section 3.6.1. Then, to relate variable $\boldsymbol{r}_f$ with $\boldsymbol{x}^{(d)}$, every destination has to be checked to add the flows going to the same node, and this is done using the constraint $\sum_{f \in F_d} \boldsymbol{r}_f - \boldsymbol{x}^{(d)} \leq \boldsymbol{0}$. Finally, the last constraint $\boldsymbol{c}_{(d)}^T \boldsymbol{s}^{(d)} = 0$ is ensuring that variable $\boldsymbol{s}^{(d)}$ is computed in the optimisation problem.

3.3.4.3. Constraints from the communication variables

Now, it must be specified that the bit rate of a flow carried in a link ($r_{k,f}$) is lower than the capacity of the link. This is done with the constraint $r_{k,f} - z_{k,f} W_k log_2(1 + \rho_k p_{k,f} / W_k z_{k,f}) \leq 0$ that uses the Shannon formula but only for links used by flow f, that is why factor $z_{k,f}$ is included. The aim of including $z_{k,f}$ is to obtain some values of $z_{k,f}$ equal to 0 so that the constraint is not active and other values of $z_{k,f}$ equal to 1 to obtain the bit rate if the link only carries flow f. However, when all the flows that use a link are taken into account, the capacity of the link (y_k) has to be obtained using Shannon too but now considering the power allocated per link rather than the virtual power: $y_k - C_k(P_k^{link}) \leq 0$, where $C_k(P_k^{link}) = W_k log_2(1 + \rho_k P_k^{link} / W_k z_{k,f})$.

To relate the virtual power of each link and flow to the bit rate of the flow, the constraint proposed is $-p_{k,f} + \frac{W_k}{\rho_k}(2^{\frac{\pi_f}{W_k}} - 1)z_{k,f} \leq 0$. This constraint has been obtained using the Shannon capacity expression and the Lagrange multiplier $\lambda_{k,f}$ is associated with this constraint. Then, the virtual power per link and flow has to be related to the real allocated power per link. This is done using the constraint $\sum_f \boldsymbol{p}_f - \boldsymbol{P}^{link} \leq \boldsymbol{0}$, assuming that in each link the addition of the virtual power of each flow cannot be larger than the power that will be finally allocated, as the real power would be greater than the addition of virtual powers. Then, the capacity of each link has to be greater or equal than the total bit rate that will be carried in the link, and this is expressed with $\sum_{d\in D} x_k^{(d)} - y_k \leq 0$.

Finally, as t takes into account the real allocated power and also the virtual power, it is specified that t has to be bigger than the allocated power with the following constraint $\boldsymbol{P}^{link} - t \leq 0$.

3.3.5. Structure of the algorithm

As input parameters of the problem, the information needed is the topology for each flow (the input and the output nodes), the already explained given communication variables and the minimum bit rate for each flow. The variables computed in the optimisation problem are divided in three blocks:

- From the *network variables* they are the bit rate that will be carried in each link ($\boldsymbol{x}^{(d)}$), the amount of traffic allowed to be injected in each node ($\boldsymbol{s}^{(d)}$) and the value of the cost function of the total power taking into account both the real and the virtual power.
- From the *communication variables*, the total power per link ($\boldsymbol{P}^{link}$), the virtual power per flow and link ($\boldsymbol{p}$) and the capacity assigned to each link ($\boldsymbol{y}$).
- From the *flow variables* the bit rate per flow carried in each link ($\boldsymbol{r}_f$) and the links used for each flow ($\boldsymbol{z}_f$).

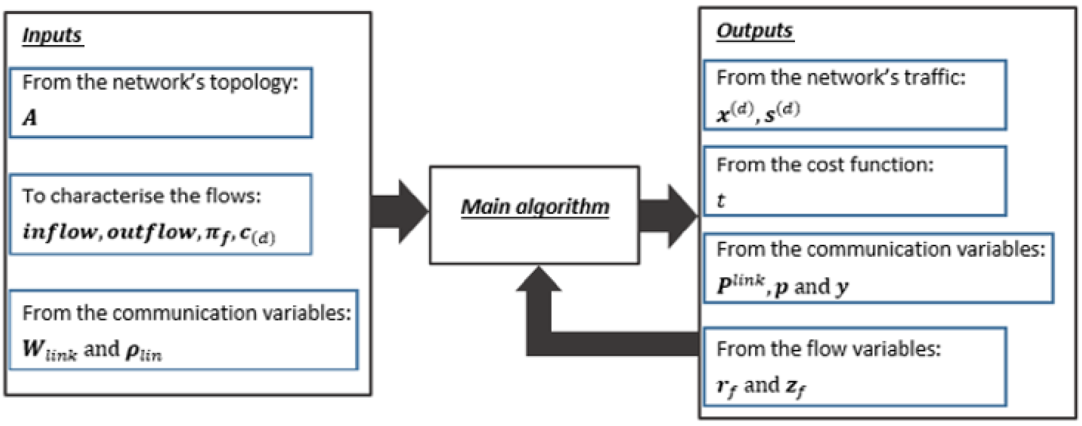

Inputs
From the network's topology: $\boldsymbol{A}$
To characterise the flows: $\boldsymbol{inflow}, \boldsymbol{outflow}, \boldsymbol{\pi_f}, \boldsymbol{c_{(d)}}$
From the communication variables: $\boldsymbol{W}_{link}$ and $\boldsymbol{\rho}_{lin}$
Main algorithm
Outputs
From the network's traffic: $\boldsymbol{x}^{(d)}, \boldsymbol{s}^{(d)}$
From the cost function: $\boldsymbol{t}$
From the communication variables: $\boldsymbol{P}^{link}, \boldsymbol{p}$ and $\boldsymbol{y}$
From the flow variables: $\boldsymbol{r_f}$ and $\boldsymbol{z_f}$

Figure 4 Block diagram of the inputs and outputs of the main algorithm

Figure 4 presents a feedback between the outputs of the convex problem and the main algorithm. This is going to be fundamental in the algorithm and is explained in the following subsection in order to understand that the algorithm uses a series of convex problems that determines the different parts of the whole algorithm.

3.3.5.1. Maximum power constraint and selection of links for a given flow

While the previous formulation defines a convex problem, two aspects are not totally controlled and are going to be addressed in this section. Then, the complete proposed algorithm needs an additional part which will guarantee that the optimal solution obtained is feasible, modifying it if needed. The first aspect to be controlled is that the allocated power in each link is not greater than the maximum allowed power per link. This could be fixed with an additional constraint containing the maximum power per link but infeasibility problems can appear when solving the optimisation problem. To avoid this situation, the problem is solved without the total power constraint and, once the solution is obtained, the algorithm checks if all the allocated powers respect this constraint. If a link does not respect it, the algorithm decides which flow needs to avoid that link in order to accomplish the constraint. This decision is made according to the Lagrange multiplier $\lambda_{k,f}$ defined before.

As the problem is convex, the solution provides a single minimum: the optimal solution. For this reason, the Lagrange multipliers' theory [7] can be applied and the maximum multiplier for each link and flow gives an idea of which is the flow that affects the most to the cost function. Let us analyse how the obtained minimum of the cost function (defined as t^*) depends on the constraint:

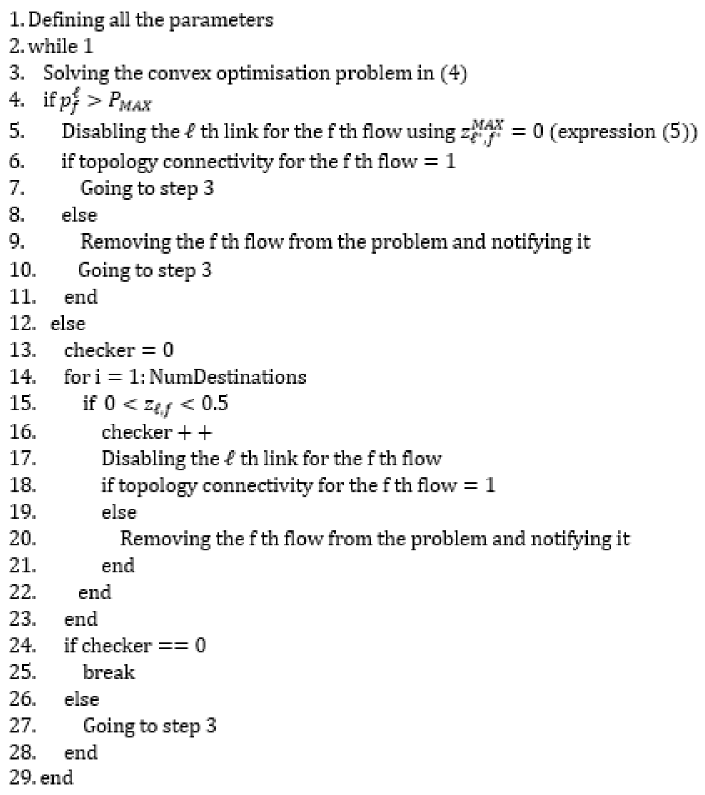
$$\lambda_{k,f} : -p_{k,f} + \psi_{k,f} z_{k,f} \leq 0 \rightarrow \frac{\partial t^*(\psi_{k,f})}{\partial \psi_{k,f}} = \lambda_{k,f} z_{k,f} \quad \textbf{(8)}$$

It may be observed that it depends on the multiplier and the value of $z_{k,f}$. This result is important because it may happen that two flows equally affect the cost function but in one case the obtained $z_{k,f}$ is much lower than in the other but larger than 0 (e.g. 0.25 and 0.9). In this case it would be unfair to discard the flow with the higher value of $z_{k,f}$, for this reason a penalisation is added when choosing the flow to be discarded by using:

$$f^* = \arg\max_f \left(\frac{\lambda_{k^*,f}}{z_{k^*,f}}\right) \quad \textbf{(9)}$$

To discard the selected flow in the link where the maximum power is violated, its mask is changed for $z_{k^*,f^*}^{MAX} = 0$ and the convex optimisation problem is solved again after checking that an alternative path between the origin and destination nodes exists for that flow. Connectivity is checked using Shortest Path algorithm and otherwise the flow is not routed.

Even if there are not power issues there is a second aspect that has to be controlled: the values of $z_{k,f}$ have to be either 0 or 1. This check is done after the convex problem has been solved and no power issues are found. For this second aspect the algorithm checks, for all the destination nodes, if in the set of flows going to the same destination there is a flow with $0 < z_{k,f} < 0.5$. If this is the case, its mask $z_{k,f}^{MAX}$ is set to 0 to avoid using the link when the optimisation problem is solved again. It is important to notice that before solving the problem again the algorithm needs to make sure that, as some links are going to be avoided, the connectivity between the origin and the destination nodes is ensured following an alternative path, otherwise it will notify that the flow could not be routed and the problem will be solved for the rest of the flows. Then, the whole algorithm can be understood as a series of convex problems where in each iteration the values of power and the values of $z_{k,f}$ are checked. The pseudocode of the whole algorithm is:



```
1. Defining all the parameters
2. while 1
3.   Solving the convex optimisation problem in (4)
4.   if $p_f^\ell > P_{MAX}$
5.     Disabling the $\ell$ th link for the f th flow using $z_{\ell',f'}^{MAX} = 0$ (expression (5))
6.     if topology connectivity for the f th flow = 1
7.       Going to step 3
8.     else
9.       Removing the f th flow from the problem and notifying it
10.      Going to step 3
11.    end
12.  else
13.    checker = 0
14.    for i = 1: NumDestinations
15.      if $0 < z_{\ell,f} < 0.5$
16.        checker + +
17.        Disabling the $\ell$ th link for the f th flow
18.        if topology connectivity for the f th flow = 1
19.        else
20.          Removing the f th flow from the problem and notifying it
21.        end
22.      end
23.    end
24.    if checker == 0
25.      break
26.    else
27.      Going to step 3
28.    end
29. end
```

Figure 5 Pseudocode main algorithm

3.4. Addressing latency

The goal of this section is to provide approaches to control the impact of the delay of the network in the results. First, an approach following the typical realistic implementations is proposed and then an approach similar to the one made in [4] is discussed.

3.4.1. Ensuring a maximum end-to-end delay per flow (DMA algorithm)

For this scenario, the aim of the optimisation problem is to ensure a given maximum end-to-end delay per flow, while the power allocation is still optimal to be the minimum necessary in the circuit-mode routing. This approach follows what is expected in resource slicing in realistic environments, as MNOs are interested in being allocated network slices where end-to-end communication delays are guaranteed. To implement it, some changes are needed in the algorithm proposed in 3.3. This modified algorithm is going to be called DMA (Delay Main Algorithm). First, the problem defines a new vector $\boldsymbol{D}_{MAX} \in \mathbb{R}^{F\times 1}$ that will be given to the problem containing in each component the maximum end-to-end delay allowed per flow. Then, a vector $\boldsymbol{D}_{MAX,link} \in \mathbb{R}^{L\times 1}$ will also be given containing the maximum delay allowed in each link and will be used to define the worst case delay for a given flow.

However, to model the actual end-to-end delay per flow, an M/G/1 queueing system model [14] is considered for every link. In M/G/1 queueing arrivals are modulated by a Poisson process with a general (unknown) distribution for the time required to service each user and a single service provider. Then, to model the average delay in link l the expression used in [12] is:

$$E(D_l) = \frac{(1-\beta)/\mu}{y_l} + \frac{\beta/\mu}{y_l - \sum_{s:l\in L(s)} x_s} \tag{10}$$

In the expression, $\beta = (1 + \mu^2\sigma^2)/2$ and it is assumed a general packet length distribution of mean $1/\mu$ and variance σ^2 specified by the user. The traffic in the link corresponds to the term $\sum_{s:l \in L(s)} x_s$. To implement a maximum delay for each flow in the convex algorithm, two additional convex constraints are needed:

$$\boldsymbol{D}_{MAX,link}^{T} \boldsymbol{z}_f \leq \boldsymbol{D}_{MAX,f}$$

$$\frac{(1-\beta)/\mu}{y_l} + \frac{\beta/\mu}{y_l - \sum_{s:l \in L(s)} x_s} \leq D_{MAX,link} \qquad \textbf{(11)}$$

Other aspects need to be fixed in the problem proposed in 3.3.4 to control the impact of the delay using the model of this section. First, when checking the connectivity after the convex problem is solved (lines 6 and 18 in the pseudocode in Figure 5), it has to be checked that the connectivity proposes an alternative path where the end-to-end delay per flow is lower than the maximum set, otherwise the flow has to be removed. Second, it must be ensured that there is enough power to bear the minimum delay without traffic, what is going to discard some links for all the flows. This is done by considering a given minimum delay without traffic for every link and considering the following:

$$\frac{(1-\beta)/\mu}{y_l} + \frac{\beta/\mu}{y_l} \leq d_{l,no\ traffic} \rightarrow y_l \geq \frac{1}{d_{l,no\ traffic}} \cdot ((1-\beta)/\mu + \beta/\mu) \qquad \textbf{(12)}$$

Using the previous equation a minimum value of capacity for each link is obtained, what can be used to determine in which links it cannot be achieved due to low values of SNR. As explained in 3.3.2, links where this capacity cannot be achieved will be disabled for any flow using the corresponding mask in $\boldsymbol{z}_f^{MAX}$.

As a result, this first modification of the algorithm is not only minimising the power allocated but also ensuring a maximum end-to-end delay per flow in the computed paths. This approach seems the most efficient in an algorithm aimed to be tested in a real environment platform and is still based on a convex problem. Therefore, this will be the main algorithm used in this project.

3.4.2. Minimising the maximum link utilisation

In this scenario, a methodology explained for the optimisation of a function closely related to the delay is presented. It is based on the approach followed in [4] through the definition of a quasi-convex cost function. While in 3.4.1 the end-to-end delay per flow was considered, now the total delay of the network is optimised. To summarise, the function of the total delay of a network is widely used in literature for non-convex optimisation problems [12] and the expression is close to the occupation/utilisation of the links in a network.

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_l \frac{x_l}{y_l - x_l} \qquad \textbf{(13)}$$

However, to be able to use it in a convex problem, an alternative function with similar properties needs to be used. A typical alternative is the maximum link utilisation of a network, a quasi-convex function that can be implemented in the algorithm using a series of convex feasible problems [9, Section 4.2]. The maximum link utilisation of a network can be computed using:

$$f(\boldsymbol{x}, \boldsymbol{y}) = \max_{l} \frac{x_l}{y_l} \qquad \textbf{(14)}$$

The aim is to use the algorithm proposed in 3.3 but adding the objective to minimise the maximum link utilisation of the network, what would result in minimising the delay of the network. To do that it is not necessary to change the already set cost function because a quasi-convex function is being used. To implement it, following the theory exposed in [7], the maximum link utilisation of the network is defined as $f_0(x, y)$. Then, there has to exist a family of convex functions g_t such that:

$$f(\boldsymbol{x}, \boldsymbol{y}) \leq t \leftrightarrow g_t(\boldsymbol{x}, \boldsymbol{y}) \leq \mathbf{0}$$

$$t_1 \geq t_2 \rightarrow g_{t_1}(\boldsymbol{x}, \boldsymbol{y}) \leq g_{t_2}(\boldsymbol{x}, \boldsymbol{y}) \tag{15}$$

for a $t \in \mathbb{R}$, so the problem is modified using for every destination d in the network:

$$f^{(d)}(\boldsymbol{x}, \boldsymbol{y}) = \max_l \frac{x_l^{(d)}}{y_l^{(d)}} \rightarrow g_t^{(d)}(\boldsymbol{x}, \boldsymbol{y}) = \max_l (x_l^{(d)} - t y_l^{(d)}) \tag{16}$$

This result in adding D constraints to the problem where t is the value of the objective function whose feasibility is being checked. The series of convex feasible problems consist in iterating with an initial t_0 and if feasibility is ensured then for the following iteration $t_i < t_0$ and otherwise $t_i > t_0$. For details on how the value of t is updated in each iteration, the bisection algorithm used can be checked in Algorithm 1 at [9, pg.146].

The approach proposed in 3.4.2 has the advantage of following the theory in [4] but it cannot be proposed as an optimal solution in an implementation in a real platform. This is because the end-to-end delay is not ensured, as it is the case in the solution proposed in 3.4.1. Results comparing the maximum link utilisation with the MA algorithm and the approach described in this section are included in Appendix 3.

3.5. Impact of the algorithm results in overall network performance

The algorithm proposed is affecting the allocation of radio resources and routing, but it has not taken into consideration other aspects of the network performance. For an implementation in a real platform, we need to take into account how the algorithm impact in both the higher and the lower layers of the network.

3.5.1. Impact in TCP layers

The aim of this section is to check if the results obtained with the algorithm are feasible considering the TCP level of the network. TCP is a standard protocol defining the tools to guarantee the establishment and the maintenance of data exchange in a network. Based on the TCP theory, a network knows how to divide the data contained in the traffic into packets that will be carried by a flow going from the origin node to the destination node. Additionally, as these packets will be sent and received at a fixed rate, TCP also has to specify the flow control mechanism that guarantees the performance of the network without having problems in receiving the packets.

Then, to check that the solution provided by the proposed routing algorithm (DMA) fits with the flow control, it has to be analysed by the so-called TCP checker. First of all, it has to be considered that to implement the flow control mechanism TCP uses a window to control the pace in the transmission of packets. In this window there are all the packets that the source node has transmitted but that have not been acknowledged yet by the destination node. The idea is that when the network is congested this acknowledgements are delayed and TCP notifies the source that the window size should be decreased to avoid increasing the congestion. An important measurement in this mechanism is the *round-trip time*, measured as the time difference between the instant a packet is sent and the instant the acknowledgement arrives if there is no congestion (the packet and the acknowledgement can use different paths). Therefore, this section will adapt the DMA algorithm proposed to the congestion control mechanism implemented by TCP.

The *congestion control* consists in an algorithm that notifies the source node about the suitable rate according to the congestion in the network, and also another algorithm that measures congestion in the links and notifies the sources about it. As per the congestion control mechanisms, the two main ones are TCP Reno and TCP Vegas as the latter being an improved version of the first. The TCP Reno mechanism starts with what is called a slow start, setting a small-sized window and incrementing it by one each time that an acknowledgement is received at the source node. Then, when the window size reaches a threshold, the slow start finishes and the congestion avoidance phase begins, making the increase of the window size even slower. If the source node detects that a packet has been lost because a received acknowledge sent by the destination node is duplicate, the window size is halved and the congestion avoidance phase starts again in what is called the fast retransmit or fast recovery. If a packet lost is detected through a timeout, slow-start begins again.

In the proposed modification of the routing algorithm, TCP Vegas is assumed as the congestion control mechanism. TCP Vegas changes the slow-start in TCP Reno by increasing the window size in a more prudent way, leading to fewer losses. Furthermore, the retransmission mechanism implemented in TCP Vegas is an enhancement of the TCP Reno one. This is because when the first duplicated acknowledgement is received the source immediately checks the timeout while in TCP Reno this was done when the third duplicate acknowledgement arrived. This is why the detection of loss is much faster in TCP Vegas. In addition, a new congestion avoidance mechanism is implemented in TCP Vegas where the sources anticipate the congestion by monitoring the packet transmission rate seen in the network and comparing it to the rate that it is expected. Using this mechanism, TCP Vegas adjusts the source's window size to ensure a low number of packets buffered in the routers along the path.

Literature in the topic [9] proposes that the behaviour of the network in stable regime can be obtained by solving a convex problem and then this checker is going to be added to the main algorithm. The idea is that once the optimal routing algorithm has provided the solution for the routing and the allocation of resources, this checker would determine whether the specified bit rate can be routed using the allocated resources or not considering the TCP protocol used in the communication. Thus, work in [9] proposes that the convex problem needs to simulate the behaviour of the network in stable regimen when TCP Vegas is used.

This proposed convex problem needs as input parameters the computed path per every flow $\boldsymbol{z} \in \mathbb{R}^{L\times F}$, the round trip time of the source nodes in vector $\boldsymbol{d}_S \in \mathbb{R}^{1\times F}$, the difference between the expected and actual rate for each source defined in vector $\boldsymbol{\alpha}_S \in \mathbb{R}^{1\times F}$ and the capacity of each link $\boldsymbol{y} \in \mathbb{R}^{L\times 1}$. What TCP does is to model the communication considering the source node where the transmitted bit rate is given and the destination node that specifies the queue needed and notifies the source node whether the packets in the flow are being received or not. This is going to be used in the routing algorithm by checking the maximum bit rate that TCP can handle (defined as $\boldsymbol{x}_S \in \mathbb{R}^{F\times 1}$) and if it is lower than the one provided by the routing algorithm it will be kept as the allowed one. The whole process will remain as follows:

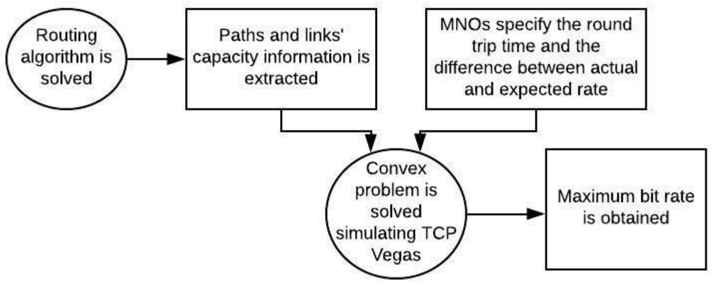

Routing algorithm is solved
Paths and links' capacity information is extracted
MNOs specify the round trip time and the difference between actual and expected rate
Convex problem is solved simulating TCP Vegas
Maximum bit rate is obtained

Figure 6 Diagram of the algorithm and the checker

The convex problem to simulate TCP Vegas is the following:

$$\begin{aligned} &\underset{\boldsymbol{x}_s}{\text{minimise}} -\boldsymbol{d}_s \cdot \boldsymbol{\alpha}_s \cdot \log(\boldsymbol{x}_s) \\ &subject\ to \quad \boldsymbol{z} \cdot \boldsymbol{x}_s - \boldsymbol{y} \leq \boldsymbol{0} \end{aligned} \qquad \textbf{(17)}$$

Then, in this optimisation problem all the variables are input parameters except $\boldsymbol{x}_s$, which is computed in the optimisation and returned as the TCP Vegas allowed bit rate per flow.

3.5.2. Impact in lower layers

This section focuses in analysing the impact that bandwidth has in the algorithm. As explained in section 3.3.3, the algorithm does not optimise the bandwidth to ensure a circuit routing mode using a convex problem. However, in 802.11-based platforms the bandwidth allocated in each link cannot be assigned in blocks of arbitrary size, there are some restrictions. To allocate the bandwidth in the links there are two approaches to follow but as the algorithm considers a given amount of bandwidth per link, it is assumed that this two approaches have been already implemented before using the algorithm.

The first approach is using frequency bands, which are blocks of a fixed amount of bandwidth that come from the division of the total amount of bandwidth available. By dividing the total amount of bandwidth in multiple blocks, these blocks can be interpreted as orthogonal frequency bands and they can be used simultaneously in the same or close links without interference. In 802.11-based platforms using more than one band in the same link is called carrier aggregation and it is a built-in feature. Carrier aggregation is used to be able to work with a higher amount of bandwidth without interference but it must be ensured that in close links the same band cannot be used. Then, it is interesting to divide the total amount of bandwidth in a low number of frequency bands to be able to have a small number of bands but with a higher amount of bandwidth in them.

However, when there is a network with a given number of links there is a minimum number of bands that are needed to avoid interference. An analysis to determine this minimum number of bands (or colours) can be using the theory on graph colouring found in [4, Section D] and it would involve using different bands to links entering or leaving the same node and using the same band in a link at transmitter and receiver nodes. When the minimum number of bands needed is found, colouring algorithms, as the proposed in [15], can be used to solve how the bands are to be distributed. In this project the colouring algorithms will not be implemented because the algorithm explained in 3.3 considers a given amount of bandwidth per link, what means that the amount specified is the result of a proper colouring and there is not interference between links.

On the other side, there is the option to use just a single band with all the available bandwidth in it and try to limit the interference that would appear when using the same band in close links. This would allow to use a bigger amount of bandwidth in one band but also would imply some interference problems with close links. However, there is a way to reduce the effects of interference and it is done using the concept of air time applicable to CSMA-CA access. *Air time* is used when the frequency reuse is equal to 1 (using just 1 band) or when two different nodes are transmitting to a common node with the same frequency band. This technique consists in allocating a fraction of time to each of the nodes that are transmitting in the same band to avoid interference. Although we have not implemented the air time mechanism to be able to work with coloured bands, the implementation is feasible by adding some additional convex constraints that follow the DCP syntax.

3.6. Updating the solution with traffic dynamics

This section aims to provide the tools to control the network traffic dynamics using the routing algorithm. To start, one of the aspects of traffic dynamics is the fluctuation in the bit rate of the flows. This slow fluctuations in the bit rate are found in real networks, as flows are not static and their bit rates change from one instant to another, and fast fluctuations are controlled by TCP and are not considered in this section. However, the DMA algorithm proposed solving the problem for a specific instant with a specific bit rate and this has to be controlled. Additionally, the flows found in a network are not always the same because they appear and disappear after the communication has been done. This is important because when a new flow appears or a flow disappears the solution might change. The analysis for the appearance and disappearance is going to be made trying to reduce the modifications in the current network configuration. This is going to be advantageous for the network because as it has an SDN architecture, changing the paths for the flows would involve increasing the control traffic in the network to carry the information to notify the changes to the routers and extra configuration time to set the routing tables in the routers.

3.6.1. Fluctuation in the bit rate

As the routing and allocation of resources is done using the algorithm for fixed bit rate conditions of flows, fluctuations can modify the optimum network design. This section proposes an algorithm that studies the possible increase in the traffic demanded by flows and provides the maximum allowed bit rate on each flow while ensuring a fixed end-to-end delay per flow. Then, if fluctuations provide bit rates below the maximum values found, flows can be routed and problems would be avoided.

The idea is to use the information of Lagrange multipliers on perturbations, performing a sensitivity analysis, in order to start from an initial bit rate for each flow and iterate until the maximum bit rate is obtained. To know how the bit rate of each flow is going to be increased, the sensitivity analysis will provide coherent values. A sensitivity analysis evaluates how the cost function (the one proposed in 3.3) changes with a perturbation in the constraints. Remembering that the cost function was the total radiated power, this analysis will tell us a first order change in power if a change in a specific constraint is produced. In this case the change produced by the fluctuation is in the bit rate, so the constraints related to the bit rate are going to be analysed.

In the problem formulated in **(7)**, there are two constraints on the minimum bit rate, so the sensitivity analysis has to be taking both of them into account by using the Lagrangian.

Performing the sensitivity analysis, the two constraints on the bit rate are:

$$\nu_{k,f} : -r_{k,f} + \pi_f z_{k,f} \leq 0$$

$$\lambda_{k,f} : -p_{k,f} + \frac{W_k}{\rho_k}(2^{\frac{\pi_f}{W_k}} - 1) z_{k,f} \leq 0 \qquad \textbf{(18)}$$

Then, from the theory on sensitivity [7], in a problem with a convex structure **(19)** the perturbations are defined as *u*u_i and v_i.

$$\begin{array}{ll} minimise & f_0(x) \\ subject\ to & f_i(x) \leq u_i, \quad i = 1, \dots, m \\ & h_i(x) = v_i, \quad i = 1, \dots, p \end{array} \qquad \textbf{(19)}$$

With absence of perturbations, the optimal value of the cost function is defined as $p^*(0,0)$ and when a perturbation is introduced the new value after solving the problem is $f_0(x)$. Then, the new cost function value after the perturbation can be computed using the expression:

$$f_0(x) \geq p^*(0,0) - {\lambda^*}^T u - {v^*}^T v \qquad \textbf{(20)}$$

The Lagrange multipliers give a linear approximation of how the cost function varies with a modification in the constraints. As the cost function is convex and the constraints are linear, big perturbations cannot be made to avoid having the approximation too far from the real value of the cost function. Figure 7 shows the approximation in orange and the actual value of the cost function in blue.

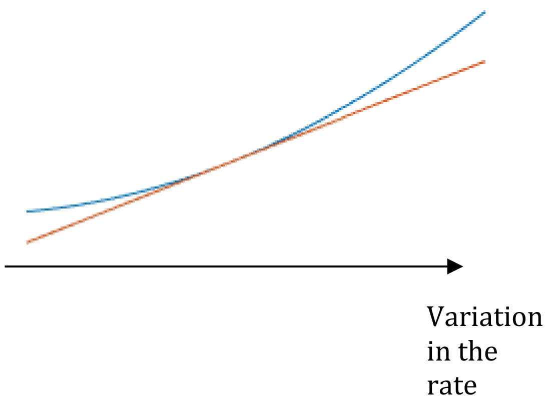


Figure 7 Real value of the cost function vs Lagrange multipliers' approximation

In Figure 7, the slope of the orange function is the multiplier. This means that when the rate (π_f) is modified, using the Lagrange multiplier it can be obtained a lower bound of the new value that the cost function is going to take. Using the expression in **(20)** it can be computed which is the produced change in the cost function. Then, if a big variation in the rate is observed, the approximation of the value of the cost function will be far from the actual one. As explained, since we are using a convex cost function the variation computed using the Lagrange multipliers will be smaller than the real one, so to measure the actual variation the only option is to solve the problem again.

To see the effect of combining the two constraints about the bit rate and the impact that a perturbation has in the cost function, the Lagrangian has to be derived. Using the two constraints (defined as f_1 and f_2*)* involving the bit rate and the Lagrange multipliers defined:

$$\frac{\partial L}{\partial \pi_f} = \boldsymbol{\nu} \frac{\partial f_1}{\partial \pi_f} + \boldsymbol{\lambda} \frac{\partial f_2}{\partial \pi_f} \text{ where } f_1 = -r_{k,f} + \pi_f z_{k,f} \text{ and } f_2 = -p_{k,f} + \frac{W_k}{\rho_k}(2^{\frac{\pi_f}{W_k}} - 1) z_{k,f} \qquad \textbf{(21)}$$

Hence:

$$\frac{\partial L}{\partial \pi_f} = \sum_{k=1}^{L} (\nu_{k,f} \cdot z_{k,f} + \lambda_{k,f} \cdot \ln(2) \cdot z_{k,f} \cdot 2^{\frac{\pi_f}{W_k}} \cdot \frac{1}{\rho_k}) \qquad \textbf{(22)}$$

The combination of both constraints gives the weighted addition of the Lagrange multipliers. Using this result and assuming *F* flows, the expression in **(20)** remains as:

$$f_0(x) \geq f(0,0) + \sum_{f=1}^{F} \left(\boldsymbol{\nu}_f \cdot \boldsymbol{z}_f + \ln(2) \sum_{k=1}^{L} \lambda_{k,f} \cdot z_{k,f} \cdot 2^{\frac{\pi_f}{W_k}} \cdot \frac{1}{\rho_k} \right) \cdot \left(\pi_f^{next} - \pi_f \right) \tag{23}$$

As the parameters to find using the sensitivity analysis were the new bit rates that the following iteration would have for each flow *(π_f^{next})* and there are *F* values, to provide expressions with just one unknown variable a similar expression for each flow is going to be formulated assuming that the cost function can change at most a 1% of its value. Then, for each flow the increase in the bit rate can be easily computed by obtaining the value of $\Delta\pi$ in the following expression:

$$1.01 \cdot f(0,0) = f(0,0) + \left(\boldsymbol{\nu}_f \cdot \boldsymbol{z}_f + \ln(2) \sum_{k=1}^{L} \lambda_{k,f} \cdot z_{k,f} \cdot 2^{\frac{\pi_f}{W_k}} \cdot \frac{1}{\rho_k} \right) \cdot \Delta\pi_f \tag{24}$$

After the sensitivity analysis is performed, the pseudocode of the proposed algorithm is:

```
1. Defining all the parameters and the initial values of $\pi_f$
2. while end_iterations == 0
3.   Solving the algorithm in 3.2 with the values of $\pi_f$
4.   Checking TCP Vegas feasibility
5.   if is not feasible with TCP
6.     Taking TCP values of $\pi_f$ and exiting the while loop
7.   else
8.   end
9.   if max(Power_links) $\leq$ 0.5
10.    for 1:F
11.      Calculating the increase of each $\pi_f$ allowing a 1% variation of the cost function
12.    end
13. elseif 0.5 < max(Power_links) $\leq$ 0.9
14.    for 1:F
15.      Calculating the increase of each $\pi_f$ allowing a 0.5% variation of the cost function
16.    end
17. else
18.   Storing in variable $full$ the flows using links with allocated power > 0.9
19.   Setting the increase of $\pi_f$ equal to 0 for the flows in $full$
20.   if length($full$) == F
21.     end_iterations = 1
22.   else
23.     For the flows not in $full$, calculating the increase of each $\pi_f$ as in step 15
24.    end
25. end
26. Updating the values of $\pi_f$
26. end
```

Figure 8 Pseudocode of the algorithm to maximise the bit rate ensuring an average end-to-end delay per flow

This algorithm starts with an initial value of bit rate for every flow and then solves the DMA algorithm. As explained, the algorithm will find the maximum bit rate that each flow can have in the given network. Then, the allocation of the power for each link in the network would be computed and from all the links the one with more power is searched. When the link is identified, the algorithm checks if it is in low, medium or high load conditions setting the thresholds in 0.5 and 0.9. If there is a link with more than 0.9 of normalised power, the link is considered full and it is checked which flows are using that link. For these flows the algorithm stops increasing the bit rate and continues for the rest of flows. In every iteration of the algorithm, the values of bit rate are checked using the TCP checker explained in 3.5.1. The criterion to select the amount of bit rate that is increased in each iteration is the sensitivity analysis already explained, obtaining small increases for flows with high Lagrange multipliers and big increases otherwise. Results for this algorithm can be checked in Appendix 4.

3.6.2. Adding and removing flows

As explained, this section's aim is to be able to efficiently add or remove a flow in a network where the routing has already been performed.

To add a new flow, first it has to be identified by its bit rate, maximum delay end-to-end to ensure and its source and destination nodes. To improve efficiency, the key idea is to solve the problem again forcing to obtain the same results for the old flows and compute the available ones for the new flow. First, the proposed algorithm to include the new flow has to make sure that there is connectivity between the source and destination nodes. Then, using the variables explained in 3.3, the variable $\boldsymbol{z}_f^{MAX}$ is going to be used with each flow to be able to modify the problem to keep the paths for the old flows.

First of all, the new values of $\boldsymbol{z}_f^{MAX}$ for the old flows will be the values of $\boldsymbol{z}_f$ that the network is using when the new flow is going to be added. By using these values the paths of the old flows will be exactly the same when the DMA algorithm is computed again adding the new flow. Additionally, a mask has to be designed for the new flow too, trying to avoid saturation problems in the network. To compute the new mask for the new flow, the algorithm is going to read the bit rate that each link in the network is carrying and analyse what could be the new bit rate of each link if the new flow's bit rate was added to every link. Then, using the Shannon capacity expression it can be computed the normalised power that would be allocated in each link to satisfy these obtained theoretical bit rates. To avoid using saturated links, the algorithm is going to disable the links where this computed normalised power is greater than 0.9 by setting the corresponding components of $\boldsymbol{z}_{new\ flow}^{MAX}$ to zero. The rest of the components of $\boldsymbol{z}_{new\ flow}^{MAX}$ will be set to one and the DMA algorithm can be then recomputed including the variables for the new flow and the masks. To avoid undesired problems, if the algorithm cannot route any of the old flows then it has to reject the new flow and notify it.

Removing a flow from the network can be easily done without modifying the existing paths. In real platforms, the routing is set in the routers' tables during a period of time before the whole problem is recomputed after some time. If a flow ends before this time is over, the best option is to keep the existing paths rather than recalculating the problem because it would carry computational work and all the process to notify the new paths to the routers. However, once the problem has to be recomputed again after a period of time because the controller needs to be updated, the algorithm would then be able to search the optimal paths and allocation of resources for the flows that remain in the network.

4. Results

4.1 Simulation and results for the typical circuit-mode routing algorithms

The simple topology in Figure 3 is proposed to simulate a network and implement the algorithms explained in 3.2 (including the solution proposed in [4]): five nodes and eight links are used while two flows going to the same destination node are going to be routed. The maximum available bandwidth is set to 100 MHz and the maximum power per link 0.01 W, being allocated in the different links as explained in 3.2. In the simulation, links are assigned with a different SNR according to random path losses and noise.

As seen in Figure 3, one flow is routed from node 2 to node 5 and the other one from node 3 to node 5. To simulate the network in high conditions of traffic, the maximum amount of bit rate adding both flows has to be the one that saturates the upper bound algorithm.

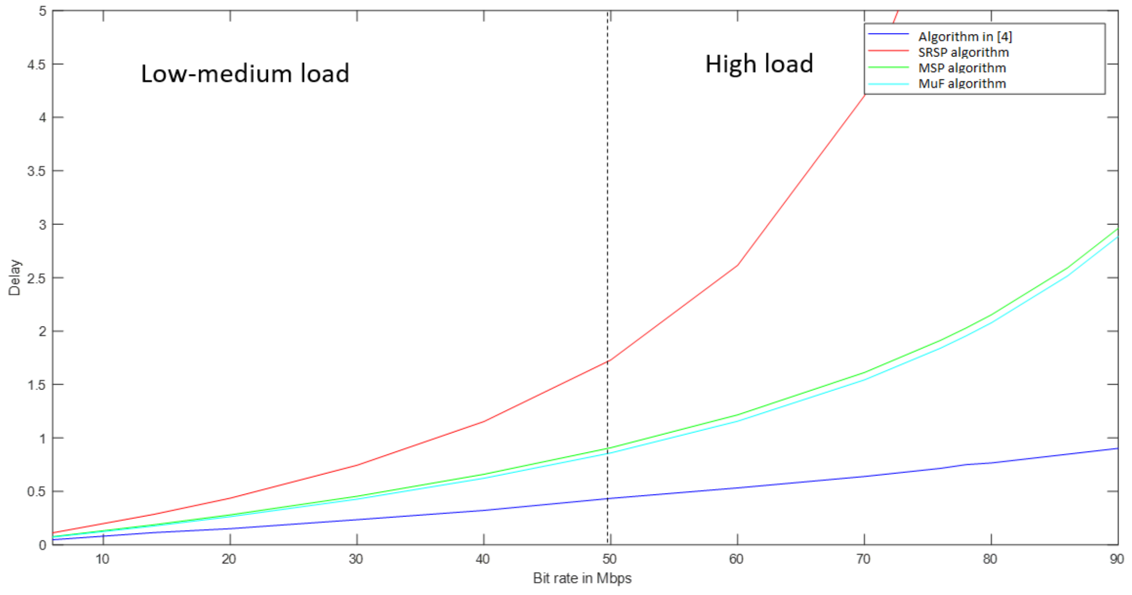


Figure 9 Comparison between circuit-mode and datagram routing algorithms

Figure 9 shows that the impact of selecting a circuit-mode routing approach to the algorithm is considerable, then this should be taken into account for the rest of results. Additionally, with a really simple network the impact of introducing multiflow (MuF) is positive as seen in Figure 9. This means that allowing multiflow in a network provides a better routing.

4.2 <u>Simulation and results for the main algorithms</u>

As explained in section 1.6 of this document, testing in the i2CAT platforms has not been possible during the lifetime of the project but a realistic scenario has been implemented in MATLAB. The results in this section have been obtained using this code and deployments of 30 access points (nodes) in an area of 1 km^2 have been simulated. These deployments assume a maximum transmitted power of 23 dBm in each access point and the channel gains in each link have been implemented using the pathloss defined in 3GPP TR 38.901 v14.3.0. [16] (UMa channel). Hence, all the access points in the network can be connected using wireless links with better or worse quality. To simulate realistic conditions, the carrier frequency is 5 GHz as the used in i2CAT platforms and the minimum intersite distance between two access points is 100 meters.

An example of topology used is the one depicted in Figure 10, where after applying a colouring algorithm the connection of the nodes has been limited to the one represented in solid blue lines.

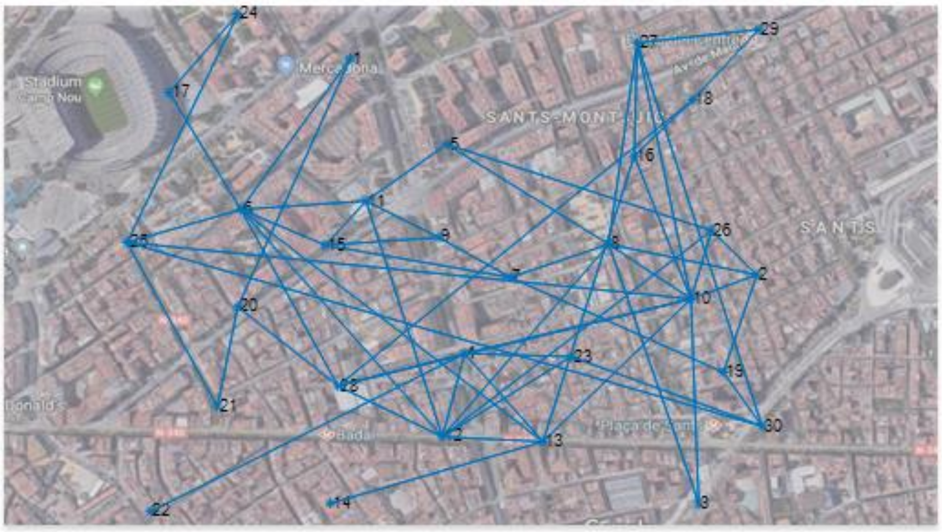


Figure 10 Simulated deployment with 30 routers placed in the Sants area

4.2.1. Suitability of the algorithms in different deployments

This first section tests the algorithms proposed in 3.2.1 (SRSP), 3.2.2 (MSP) and the DMA algorithm in 60 different deployments simulated with the assumptions explained in the previous paragraph. After each of these deployments has been created, the algorithm explained in 3.6.1 has been executed to find the bit rates per flow that correspond to high load traffic conditions in each of the deployments (see example in Appendix 4), considering the effect of TCP Vegas and random source and destination nodes for the four flows that will be considered. Then, the maximum bit rate per flow is obtained and it corresponds to having the network with a big amount of traffic.

Each of the three algorithms is used to perform the routing with the obtained bit rates Figure 11, Figure 12 and Figure 13 present a normalised histogram of the number of flows that could not be routed considering all the deployments using each algorithm when the same conditions and flows were considered.

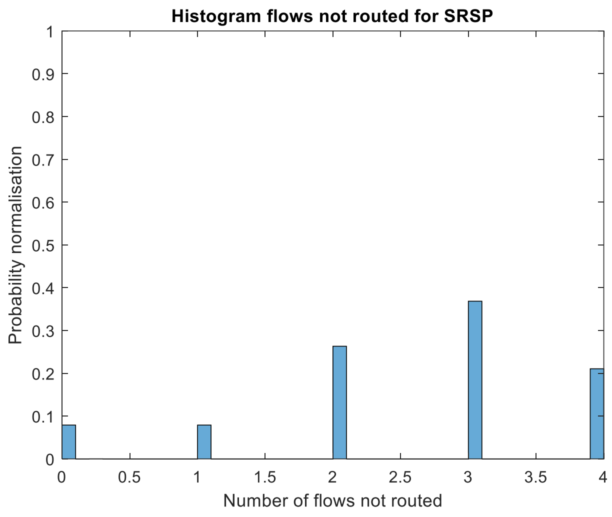


Figure 11 Normalised histogram of the flows not routed using SRSP

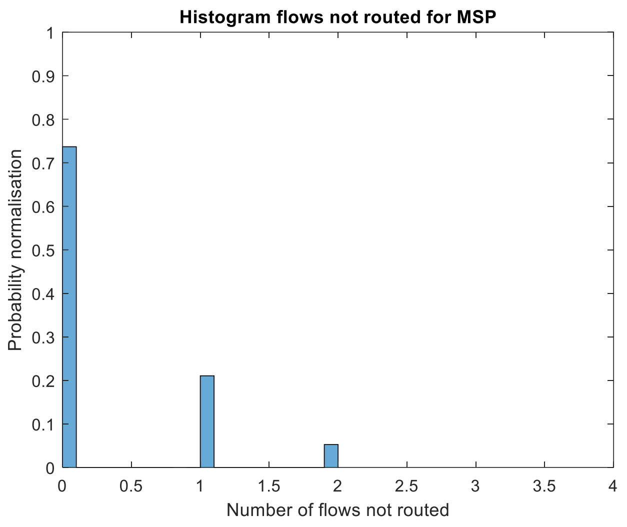


Figure 12 Normalised histogram of the flows not routed using MSP

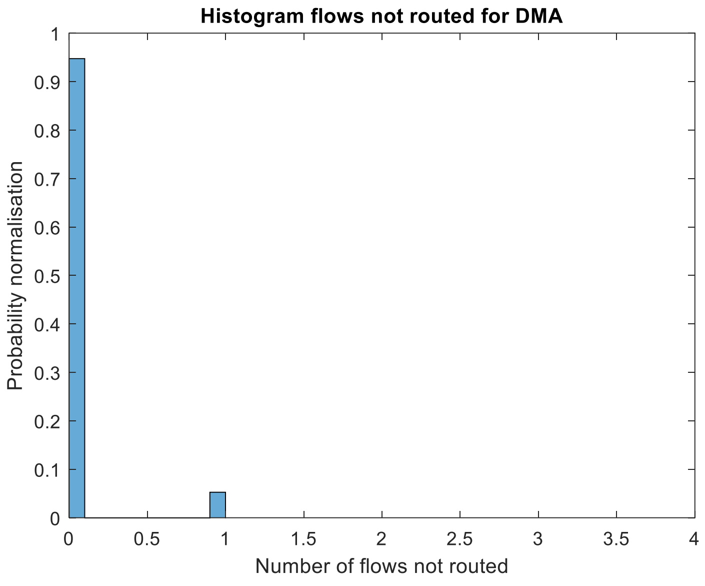


Figure 13 Normalised histogram of the flows not routed using DMA

For these high load traffic conditions the SRSP can only route all the 4 flows in almost the 10% of situations because it only proposes the shortest path between the source and the destination nodes without ensuring feasibility in terms of capacity. Additionally, MSP can route the 4 flows almost 75% of the times in high traffic conditions. This improvement with reference to SRSP is due to considering the capacity restriction and an alternative path is searched when a flow finds this problem. However, having the sequential routing of the flows does not optimise the routes in their global and routing a specific flow using one path can obstruct the rest of the flows. This means that if a link is "full" in terms of capacity, the connectivity for one of the following flows to be routed could not be ensured. It is true that the sequential routing would provide different results if flows were routed in a different order, but optimising the order of the flows increases the complexity exponentially with the number of flows and for the aim of these results a simple algorithm is used.

On the other hand, DMA uses the optimisation problem and provides an ensured average end-to-end delay per flow. This optimisation provides the best paths for the global of the flows and this is seen with almost a 95% of the situations where routing the 4 flows is possible. However, there is a

5% of cases where one flow cannot be routed and this is due to not being able to ensure the maximum end-to-end delay because a flow has to use a really long path. It is worth remarking that this 5% would increase if the MNO determined a lower maximum delay end-to-end per flow because DMA would only route flows with very short paths. In SRSP and MSP the maximum end-to-end delay is not ensured and this is not a problem when routing the flows.

Additionally, the bit rate and end-to-end delay per flow have been analysed to compare the algorithms by taking into account all the simulated deployments. For each algorithm it has been checked the feasibility of the solution considering the algorithm explained in 3.5.1.

Figure 14 shows that the DMA algorithm allows routing a higher bit rate compared to MSP and SRSP. A part from the flows that cannot be routed using SRSP and MSP that correspond to bit rate equal to 0, the plot reflects that when these two simpler algorithms can route flows they cannot satisfy the same bit rate as in DMA, what means that TCP Vegas only allows lower values of bit rate.

The differences between the algorithms are summarised in Table 2:

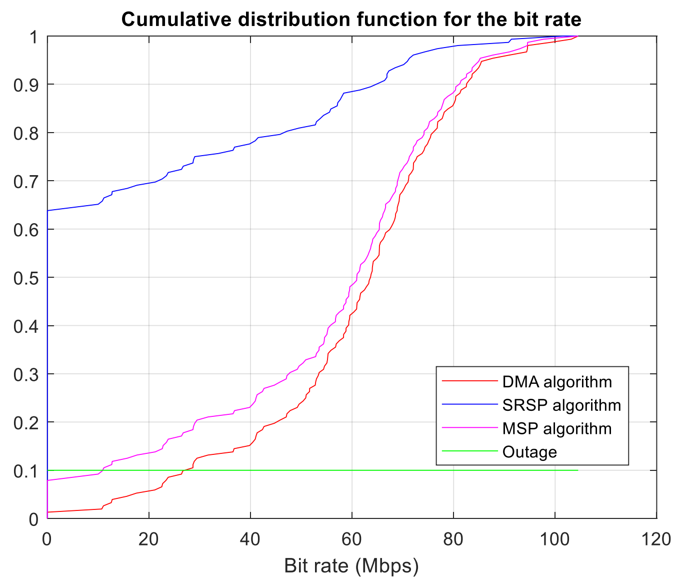


Figure 14 Cumulative distribution function for the bit rate

Algorithm	Maximum routable bit rate	Average routable bit rate	Outage bit rate
DMA	104.54 Mbps	60.02 Mbps	27 Mbps
SRSP	101.70 Mbps	17.39 Mbps	0 Mbps
MSP	104.54 Mbps	54.52 Mbps	11 Mbps

Table 2 Summary of the differences in the bit rate for each algorithm

For the average end-to-end delay the maximum has been set to 80 μs for each flow. Then, using the DMA this value should be ensured.

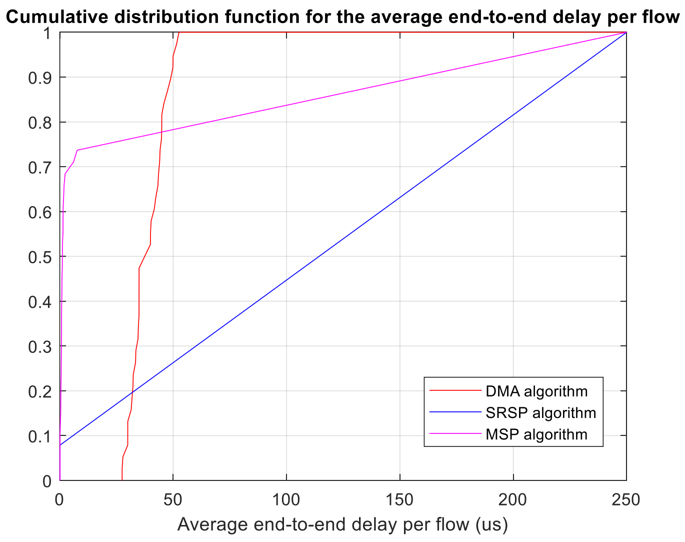


Figure 15 Cumulative distribution function for the average end-to-end delay per flow

In Figure 15 it is checked that the maximum end-to-end delay is ensured only with the DMA algorithm. This means that DMA allocates the necessary power in each link to ensure this delay in all the routed flows, obtaining delay values close to the bound fixed as power is being minimised. However, SRSP and MSP have a static allocation of power and always equal to the maximum available. Due to allocating the maximum power, there are some flows routed with SRSP and MSP that can ensure a delay lower than the achieved with the DMA but it is important to notice that using DMA all the flows ensure the maximum delay while this does not happen with the other two algorithms. A detailed study of the power allocated is found in Appendix 5.

5 Budget

As explained in the introduction section of this document, the aim of the project is to provide the necessary tools that would be later tested in a real environment platform or network. For this reason the prototyping part is unnecessary, as a real network implemented using an SDN architecture is assumed to be used by the users of the product (the MNOs). Thus, the software proposed can be used in any SDN network controller to manage the routing. However, a list of components that the company interested in using the product should have to generate the backhaul network is included:

- To set up the backhaul network, the hardware needed includes routers, access points, antennas, a server and the controller of the network but also a spot to place the antennas is needed.
- To run the backhaul network, the needed tools will be the energy to transmit the signal, the license to transmit the carrier frequency in the wireless links, the software proposed in this project and a server with MATLAB installed that will be remotely connected with the SDN controller to read/write the routing solutions obtained with the algorithm.

The costs are the corresponding to the design part tasks considering a cost of a junior engineer and the software used, which consists of libraries that can be used for free and a MATLAB license (800 euros an annual professional license [17]).

Work package	Task	Hours	Hourly cost (gross salary[18]) €	Total cost €
WP 0	Reviewing the given scenario and the SDN architecture	15	18	270
	Understanding typical KPIs and practical constraints found in real networks	25	18	450
	Setting the problem definition and objectives of the project	20	18	360
WP 1	Reviewing previous work on routing algorithms	35	18	630
	Learning about convex optimisation	35	18	630
	Using software for convex optimisation	30	18	540
	Research on routing algorithms	10	18	180
	Learning about TCP and queueing	20	18	360
	Reviewing state of the art	70	18	1,260
WP 2	Analysing impact in higher and lower layers	40	18	720
	Study of the traffic dynamics	60	18	1,080
WP 3	Designing and implementing less complex routing algorithms	20	18	360
	Designing and implementing the proposed algorithms	130	18	2,340
	Obtaining results	80	18	1,440
		590		10,620

Table 3 Costs separated by main tasks

6 Conclusions and future development

This project aimed to provide the tools to perform a joint optimisation of the routing and the resource allocation for a wireless backhaul network with an SDN architecture working with protocol IEEE 802.11ac. Additionally, all the work in the project has been done focusing in the practical constraints that would carry implementing the algorithms proposed in a real environment platform (such as using a circuit-mode approach) and ensuring a minimum level of performance in terms of average end-to-end delay per flow to boost the implementation in a future 5G network.

Firstly, the scenario and the type of network considered have been explained departing from the work done in [4], where a datagram approach for the routing was proposed. After that, some less complex circuit-mode algorithms have been designed (SRSP and MSP) in order to have the tools to compare between the algorithms that are proposed in the project. More complex algorithms have been proposed adding features to the main convex problem designed in the project. This main algorithm is composed of a convex problem and an additional part that allows introducing convexity to the problem and avoiding infeasibility problems. The main algorithm consists of a series of variables containing the information of the routing and allocation of resources and the constraints that relate these variables.

After designing the main algorithm two approaches have been considered to quantify the impact of the delay in the problem:

1. Using additional constraints to ensure a maximum average end-to-end delay per flow, following the way real implementation's routing is done. Once a maximum average end-to-end delay per flow is fixed, the algorithm provides a routing and a resource allocation that ensure these fixed values.
2. Continuing with the line of work in [4] and working with a quasi-convex cost function to minimise the maximum utilisation of the links.

After that, the project has also analysed the impact of the solution proposed in both higher and lower layers and has proposed modifications to ensure a correct operation. These modifications consider having the TCP Vegas congestion control mechanism, whose stable regimen has been modelled in the algorithm as a convex problem following the work in [9]. Additionally, although the problem does not consider a bandwidth optimisation, a review of the impact of bandwidth allocated to each link has also been included, by analysing the effects of whether applying colouring algorithms or not (section 3.5.2).

Finally, two main tools to control the evolution of the solution with traffic dynamics have been proposed:

1. A sensitivity analysis with the Lagrange multipliers associated to specific constraints of the problem has been used to find high load traffic conditions for any deployment, what can be used to find the maximum fluctuation of the bit rate with the available resources.
2. Adding and removing flows has been solved to minimise the impact of resetting the routing tables at the nodes of the network and the extra bit rate associated with this process.

The project has fulfilled its initial objectives and results have proved that the algorithms proposed provide better performance than the typical routing algorithms used. Although results using datagram approaches could provide better results, real implementations of routing algorithms all use circuit-mode routing approaches and this is what has been done. On the other hand future development could include testing this project's work in a real platform as the ones that are planned to be installed in the research centre i2CAT. This project has considered all type of network topologies, from simple ones to more complex ones. In the future, and if a fixed topology is given, the algorithm could be adapted to a specific topology in order to reduce computational cost. Finally, depending on the network where the algorithm is going to be tested in the future, the discussion in 3.5.2 can be used to implement either the air time concept (as a way to separate transmissions in time) or the colouring algorithms (separation of a transmission in frequency).

Once the work proposed in the project has been tested in i2CAT's platforms, the obtained results are expected to be submitted to a technical workshop.

Bibliography:

[1] Collinear, “Advanced Wireless Backhaul for the Data-Driven Future”, Mobile World Live, Feb. 2018

[2] I. Tanzeena, N. Abu-Ghazaleh.”Wireless Software Defined Networking: A Survey and Taxonomy”. *IEEE Communications surveys & tutorials*, May 2016. DOI: 10.1109/COMST.2016.2571118

[3] S. Azodolmolky. *Software Defined Networking with OpenFlow,* 1st ed., UK: Packt Publishing, October 2013.

[4] D. Surís, J. Vidal and A. Agustín, “Delay Minimization in Dynamic and Scalable Multi-operator Wireless Backhauling”. ICC, May 2017

[5] M. S. Gast. *802.11ac: A survival Guide*, 2nd ed., USA: O’Reilly Media, 2015.

[6] D. Surís Coll-Vinent, “Joint routing and resource allocation for wireless backhauling of small cell networks,” Treball Fi de Grau, Escola Tècnica Superior d’Enginyeria de Telecomunicació de Barcelona, June 2016.

[7] S. Boyd and L. Vandenberghe, Convex Optimitzation. New York: Cambridge University Press, 2004.

[8] L. Xiao, M. Johansson, and S. P. Boyd, “Simultaneous routing and resource allocation via dual decomposition,” IEEE Transactions On Communications, vol. 52, no. 7, pp. 1136 – 1144, July 2004.

[9] S. H. Low. “A Duality Model of TCP and Queue Management Algorithms”. In *IEEE/ACM Transactions on networking, Vol. 11, No.4,* August 2003. DOI: 10.1109/TNET.2003.815297

[10] O. Muñoz, A. Pascual, J. Vidal and P. Baquero. “Preemption and QoS management algorithms for coordinated and uncoordinated base stations”. IEEE, 2011. DOI: 10.1109/SPAWC.2011.5990461

[11] K. Srivatsa. “An Introduction to Graph Theory and Network Analysis”. [Online] Available: https://www.analyticsvidhya.com/blog/2018/04/introduction-to-graph-theory-network-analysis-python-codes/. [Accessed: 29 June 2018]

[12] D. P. Bertsekas and R. G. Gallager, Data Networks, Englewood Cliffs, NJ: Prentice Hall, 1992.

[13] O. Bonaventure, M. Handley, C. Raiciu. "An Overview of Multipath TCP". October 2012. [Online] Available: http://www0.cs.ucl.ac.uk/staff/M.Handley/papers/9346-login1210_bonaventure.pdf [Accessed: 15 March 2018].

[14] J. Virtamo. "38.3143 Queueing Theory". [Online] Available: https://www.netlab.tkk.fi/opetus/s383143/kalvot/E_mg1jono.pdf [Accessed: 20 May 2018].

[15] T. Husfeldt. “Graph colouring algorithms”. [Online] Available: https://compscicenter.ru/media/course_class_attachments/Thore_Husfeldt_Graph_colouring_algorithms.pdf [Accessed: 22 May 2018]

[16] ETSI. “ETSI TR 138 901 V14.3.0”. [Online] Available: https://www.etsi.org/deliver/etsi_tr/138900_138999/138901/14.03.00_60/tr_138901v140300p.pdf. [Accessed: 29 June 2018]

[17] MathWorks. “Pricing and Licensing”. [Online] Available: https://es.mathworks.com/pricing-licensing.html . [Accessed: 27 June 2018].

[18] R. Salvador. "La demanda de ingenieros y sus sueldos se disparan en Barcelona". January 2017. [Online] Available: http://www.lavanguardia.com/economia/20170108/413173102223/demanda-ingenieros-barcelona-sueldo-ofertas.html [Accessed: 27 June 2018].

[19] X. Shen, S. Diamond, Y. Gu and S. Boyd. “Disciplined Convex-Concave Programming”.

Appendices:

Appendix 1: OpenDaylight

As explained in section 1.1, in SDN the network's control and data plane are decoupled, assigning the control plane to a controller and the data plane to the routers of the network. A widely used tool to implement the controller of the network is the OpenFlow controller, used as an interface to program the routers of the network. This interface (API) contains the code of the network applications (Net Apps), the tools needed in the control plane to control and manage the network as a single system. Then, the routers of the network are the responsible for the data plane and before making a decision they consult the OpenFlow controller. This is understood as a centralised control of the network, one of the main advantages of using SDN.

Additionally, the OpenFlow technology has a control granularity in the scale of flows. This means that as this project aims to provide solutions that can be tested in a real SDN platform, a circuit mode approach has to be followed to design the routing algorithms (as specified in 1.3). Using this approach the routers have to consult the controller for the first packet of the flow they are routing and maintain the decision made for the rest of the packets of the same flow. Inside the OpenFlow technology there are some different implementations that are currently used. This project will assume using OpenDaylight, the largest open source SDN solution, which is found in Linux and uses Java. The advantage of using OpenDaylight is that all the code is open source and all the technical issues in the platform are handled by all the community of developers. In addition, OpenDaylight is compatible with any SDN architecture, what makes it suitable to avoid incompatibility problems, and is used in networks of any size. The motivation to create this open-sourced tool was mainly to reduce operational complexity found in other alternative implementations.

The OpenDaylight controllers are also suitable for modern heterogeneous multi-vendor networks, the scenario considered in this project and depicted in Figure 1. Furthermore, one of its use cases is the network resource optimisation to handle an increase of traffic avoiding the inefficient solution of just adding hardware.

To conclude, OpenDaylight is a solution to implement OpenFlow that gives flexibility to the users to choose the features that are most important in their networks. In addition, as it is open-sourced, the community is growing really fast and evolutions are constantly being presented to improve the existing tools.

Appendix 2: Work plan, milestones and Gantt diagram

This appendix provides the information to understand how the work done has been evolving. In the Work plan there are some parts included that are not presented in this report because the evolution of the project determined better alternatives. First of all the Project Proposal document of this project set the objective of providing a routing algorithm that implemented a connection-orientated solution rather than the datagram approach given in [4]. The aim of this project was to follow as much as possible the optimisation made in [4] to be able to finally test the work in a real environment platform. However, after working with the power and bandwidth optimisation proposed in [4] and trying to include the circuit-mode approach, it became evident that using DCP was not possible and therefore the problem convexity was lost.

The first problem designed to solve the situation could not be solved because the constraints proposed were not suitable. However, a modification of the constraints was proposed and the problem became a convex-concave problem that could follow the theory of disciplined convex-concave optimisation and could be implemented using disciplined convex-concave programming (DCCP) [19]. As explained in [19], DCCP problems cannot be solved using MATLAB (the code the project was planning to use) but using Python. After translating all the work done to Python, a library called *cvxpy* was used to solve the problem. However, the solvers available in *cvxpy* were unable to find a stable solution for complex topologies and the optimal solution was not even ensured for simple ones. For this reason the project decided to omit the bandwidth optimisation in order to be able to propose a convex problem following DCP that was able to provide an optimal routing and allocation of resources providing a circuit-mode approach.

Taking the previous considerations into account, the Work Plan of the project was divided in the following work packages:

Project: System definition	**WP ref: WP0**	
Major constituent: Documentation and Planning	Sheet 1 of 5	
Short description: **Defines the roles of the final user of the product (mobile operator) and details technical requirements. Provides methods to check the proper operation of the system as well as summarizing the constraints caused by the practical implementation. Finally, a definition of the theoretical part is made and objectives are set.**	Start date: 28/02/2018 End date: 09/03/2018 Start event: T1 End event: T4	
Internal task T1: Use Cases and Technical Scenarios **Internal task T2: Key-Performance Indicators review** **Internal task T3: Practical constraints summary** **Internal task T4: Problem definition and list of objectives**	Deliverables: Problem definition and objectives	Dates: 09/03/2018

Table 4 Work package 0

Project: Review	**WP ref: WP1**
Major constituent: Research and Documentation	Sheet 2 of 5
Short description: **Research on the different topics needed during the project. This includes previous work on the topic, learning present algorithm and mathematical tools used in optimization and the ways to implement them using software.**	Start date: 01/12/2017 End date: 30/05/2018 Start event: T1 End event: T5

Internal task T1: Paper ICC-17 [4] **Internal task T2: Problem solvers using Convex Optimization** **Internal task T3: Disciplined Convex Concave Optimization** **Internal task T4: Python Optimization tools** **Internal task T5: Routing algorithms (datagram, virtual circuit…)** **Internal task T6: End to end delay and M/G/1** **Internal task T7: TCP Congestion Control**	Deliverables: DCCP presentation	Dates: 22/02/2018

Table 5 Work package 1

Project: Routing and resource allocation in wireless interference 802.11-based networks	**WP ref: WP2**	
Major constituent: Theoretical approach	Sheet 3 of 5	
Short description: **Propose different routing algorithms for 802.11-based networks with random access taking into account the impact in higher and in lower layers. Analysis of the evolution of the solutions with the traffic dynamics taking into account non-static flows and the evolution of the network.**	Start date: 24/01/2018 End date: 25/05/2018 Start event: T1 End event: T5	
Internal task T1: Connection-orientated routing algorithms **Internal task T2: Impact in higher layers** **Internal task T3: Impact in lower layers** **Internal task T4: Evolution of the solutions with traffic dynamics**	Deliverables: Routing algorithms Impact in layers Dynamic solution and problem separation	Dates: 23/03/2018 15/05/2018 25/05/2018

Table 6 Work package 2

Project: System Level simulation	**WP ref: WP3**
Major constituent: Formulation, coding and simulation	Sheet 4 of 5
Short description:	Planned start date: 30/01/2018

Reviewing previous algorithms proposed, formulating a new algorithm considering the approaches in WP2 and implementing the code. Extracting results comparing to typical routing algorithms using simulations and checking their consistency.	Planned end date: 16/06/2018	
	Start event: T1 End event: T2	
Internal task T1: Implementing Matlab and Python codes **Internal task T2: Formulation of the optimization problem** **Internal task T3: Results study**	Deliverables: Different versions of the code	Dates: 10/04/2018

Table 7 Work package 3

Project: Economic Analysis	**WP ref: WP4**	
Major constituent: Analysis	Sheet 5 of 5	
Short description: **Analysis of the economic part of the project. It includes checking the viability of implementing the work done during the project.**	Planned start date: 25/05/2018 Planned end date: 06/06/2018	
	Start event: T1 End event: T1	
Internal task T1: Budget and viability analysis	Deliverables: Economic analysis	Dates: 06/06/2018

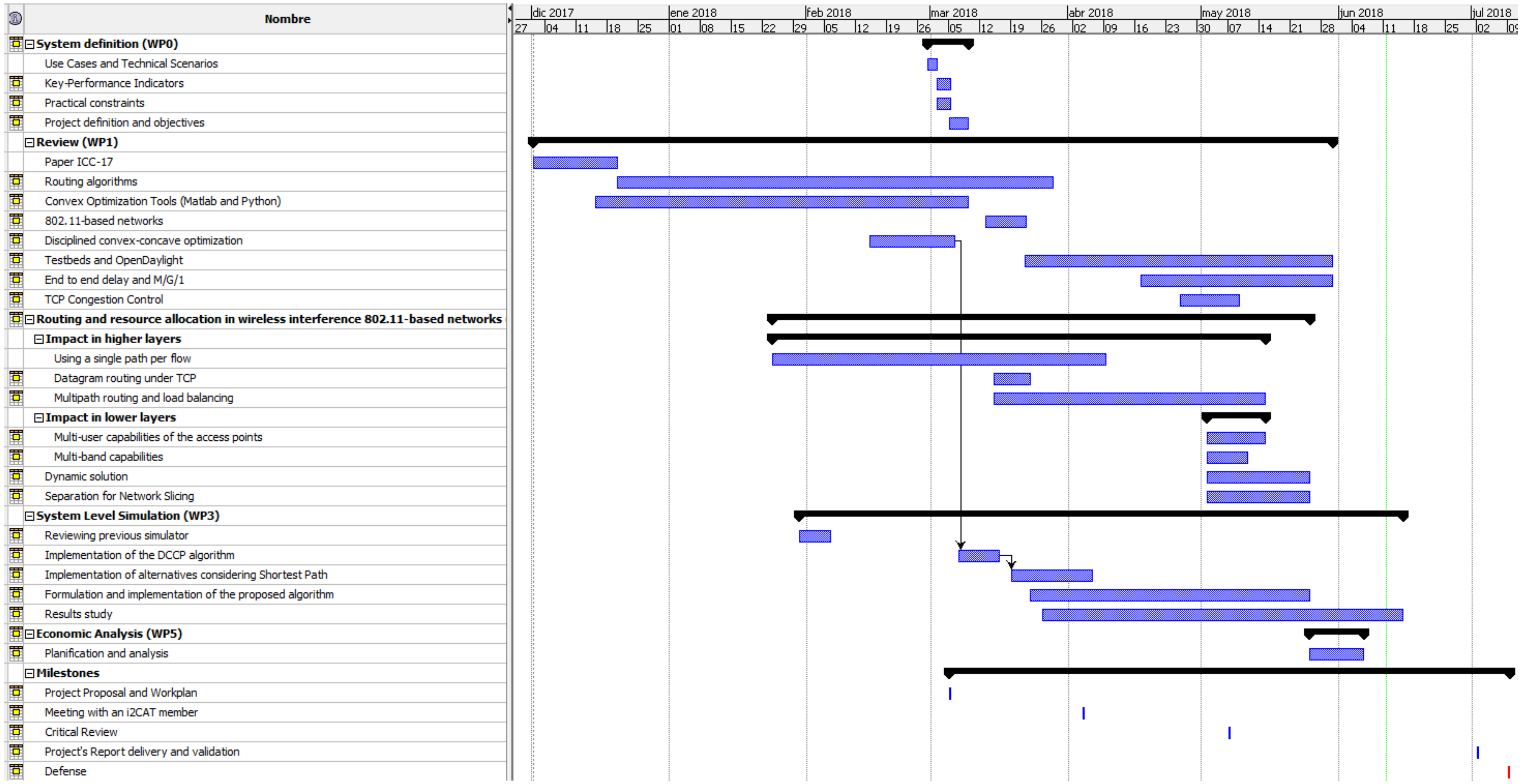
Table 8 Work package 4

The milestones set during the project were the following:

WP#	TASK#	SHORT TITLE	MILESTONE / DELIVERABLE	DATE
1	1	Disciplined convex-concave programming	Research and understanding of the topic	22/02
-	2	Project Proposal and Workplan	Delivery of the document	05/03
0	3	Problem definition and objectives	Document containing the definition of the problem	09/03
2	4	Algorithms based on Shortest Path algorithm	Implementation in code of alternative algorithms	15/03
4	5	Meeting with i2CAT	Meeting to discuss the availability of i2CAT tools	04/04
3	6	First version of the algorithm	Having the first version implemented	09/04
3	7	Modification in the algorithm	Including conclusions in [1] in the algorithm	16/04
3	8	Results comparing the algorithm	Providing the first results comparing the algorithms	25/04
-	9	Critical Review	Delivery of the document	07/05
2	10	Impact in layers	Document containing the impact of the algorithm in higher and lower layers	15/05
2	11	Dynamic solution	Implementing the tools for a dynamic solution	25/05
4	12	Checking the algorithms in simulated deployments	Using the implementation in Matlab to run the proposed algorithms and to compare them	16/06
5	13	Economic analysis	Document with the economic analysis	06/06
-	14	Final report	Delivery of the document	02/07
-	15	Defense	Oral presentation	12/07

Table 9 Milestones table

The Gantt diagram is also included:



Nombre
⊟ **System definition (WP0)**
Use Cases and Technical Scenarios
Key-Performance Indicators
Practical constraints
Project definition and objectives
⊟ **Review (WP1)**
Paper ICC-17
Routing algorithms
Convex Optimization Tools (Matlab and Python)
802.11-based networks
Disciplined convex-concave optimization
Testbeds and OpenDaylight
End to end delay and M/G/1
TCP Congestion Control
⊟ **Routing and resource allocation in wireless interference 802.11-based networks**
⊟ **Impact in higher layers**
Using a single path per flow
Datagram routing under TCP
Multipath routing and load balancing
⊟ **Impact in lower layers**
Multi-user capabilities of the access points
Multi-band capabilities
Dynamic solution
Separation for Network Slicing
⊟ **System Level Simulation (WP3)**
Reviewing previous simulator
Implementation of the DCCP algorithm
Implementation of alternatives considering Shortest Path
Formulation and implementation of the proposed algorithm
Results study
⊟ **Economic Analysis (WP5)**
Planification and analysis
⊟ **Milestones**
Project Proposal and Workplan
Meeting with an i2CAT member
Critical Review
Project's Report delivery and validation
Defense

Figure 16 Gantt diagram

Appendix 3: Comparison of the maximum link utilisation using MA and the algorithm in 3.4.2

This appendix provides results regarding the maximum link utilisation after implementing the algorithms MA and the one in 3.4.2.

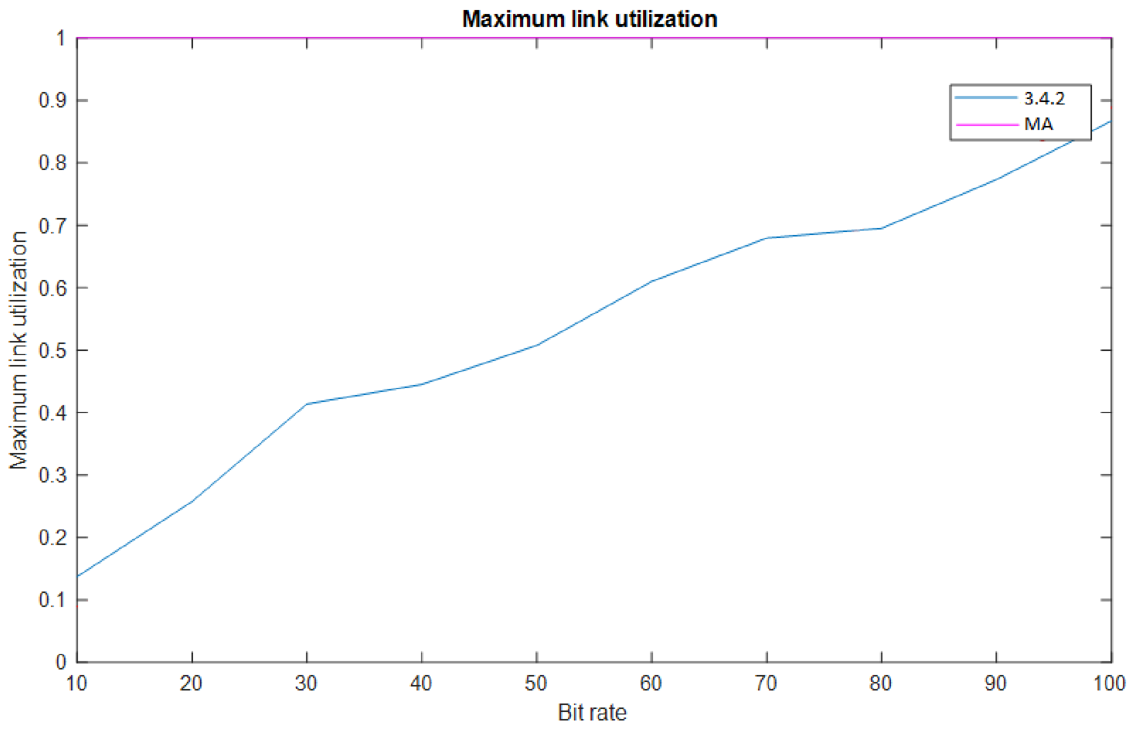


Figure 17 Maximum link utilisation comparing MA and algorithm in 3.4.2

The network topology used is the one in Figure 3 and the horizontal axis measures the total amount of bit rate in the network (considering 4 flows with equal bit rate). Following the explanation in 3.4.2, the second approach makes the solution change in order to allocate more power in each link to minimise the maximum link utilisation. On the other side, MA algorithm only allocates the necessary power and this leads to having a high link utilisation, as the capacity is equal to the bit rate that each link is carrying. However, as explained in 3.4.2, this maximum utilisation does not ensure a maximum end-to-end delay per flow, something that is done using DMA.

Appendix 4: Maximisation of the bit rate while ensuring a maximum end-to-end delay per flow

In this section a single deployment is simulated and the algorithm explained in 3.6.1 is executed, providing the maximum feasible bit rates for the flows routed. This approach has been used in 4.2 to obtain the high load traffic conditions for each deployment and now this process is detailed. To check that the average end-to-end delay is ensured, a plot is also included when the maximum end-to-end delay per flow has been set to 80 μs. For the simulation, the routing will be performed with 4 flows with the following source-destination pairs:

Flow number	Source node	Destination node
1	30	15
2	8	7
3	3	11
4	18	9

Table 10 Configuration for the flows routed

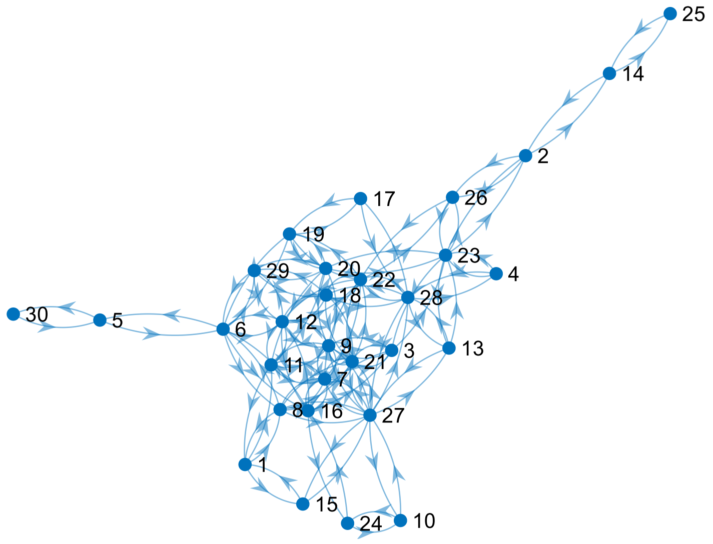


Figure 18 Topology used for the simulation

As explained in 3.6.1, this algorithm consists in a sensitivity analysis that determines the variation of the bit rate of each flow in every iteration to finally obtain the maximum values of bit rate routable. In the following plots, both the bit rate per flow and the delay end-to-end per flow are detailed for each iteration made in the algorithm:

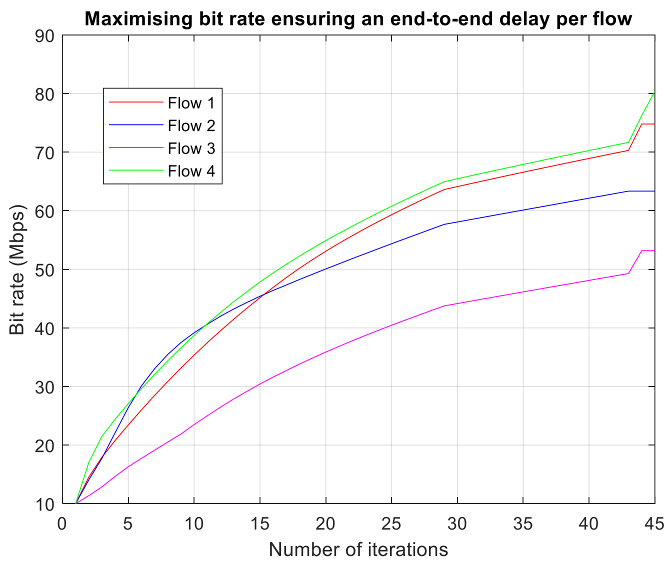


Figure 19 Bit rate evolution

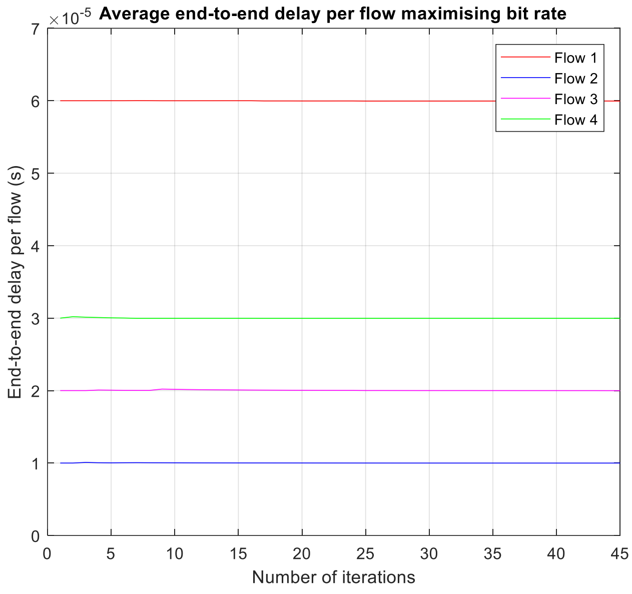


Figure 20 Average end-to-end delay per flow evolution

From the bit rate plot it can be seen that the increase in the first iterations is greater than in the last iterations. This is due to the change from 1% to 0.5% implemented in Figure 8 when the network starts to support higher load traffic conditions. The differences in the maximum bit rate obtained for each flow are due to the different SNR values that each path has, this is why in some flows a greater bit rate is allowed.

Appendix 5: Results for the power allocation using each algorithm

The total power allocated is an important parameter that the solutions provided in this project optimise. Following the same methodology explained in 4.2, this appendix has analysed the total power allocated in the network using each of the three algorithms (SRSP, MSP and DMA). This analysis has been made plotting the cumulative distribution function of the power allocated when all the deployments have been simulated and the maximum power per link has been assumed to be 0.2 W.

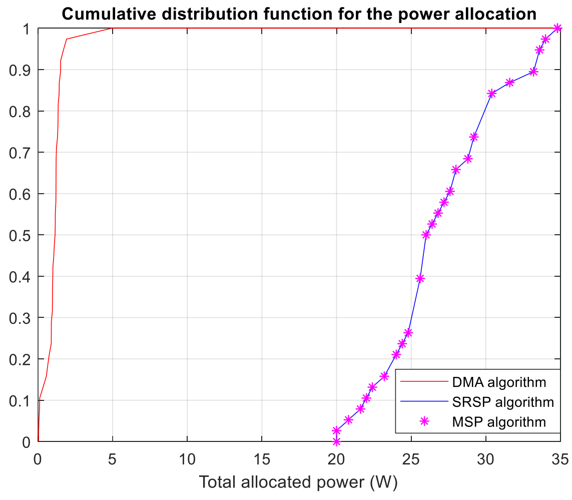


Figure 21 Cumulative distribution function for the power allocation

As seen in Figure 21, optimising the allocated power makes a big difference when computing the total power allocated in the network. While the DMA algorithm can handle the high load traffic conditions using an average allocated power in the network of 1.124 W, the SRSP and MSP use a static allocation that requires using an average of 27.179 W.

Glossary

A list of all acronyms and the meaning they stand for.

BSs: Base stations

DCCP: Disciplined convex-concave programming

DCP: Disciplined convex programming

InP: Infrastructure provider

KPIs: Key performance indicators

MNOs: Mobile network operators

MSP: Modified version of the Shortest Path

SDN: Software defined network

SNR: Signal to noise ratio

SRSP: Second-rate version of the Shortest Path

TCP: Transmission control protocol

TSC: Signal Theory and Communications department