Optimising Communication Satellites Payload Configuration with Exact Approaches

Apostolos Stathakis\textsuperscript{a,*}, Grégoire Danoy\textsuperscript{b}, Pascal Bouvry\textsuperscript{b}, El-Ghazali Talbi\textsuperscript{c} and Gianluigi Morelli\textsuperscript{d}

\textsuperscript{a}Interdisciplinary Centre for Security, Reliability, and Trust, University of Luxembourg; \textsuperscript{b}CSC Research Unit, University of Luxembourg; \textsuperscript{c}INRIA-Lille Nord Europe, Université Lille 1, France; \textsuperscript{d}SES Engineering, Betzdorf, Luxembourg

The satellite communications market is competitive and rapidly evolving. The payload, which is in charge of applying frequency conversion and amplification to the signals received from Earth before their retransmission, is made of various components. These include reconfigurable switches which permit to reroute the signals based on the market demand or because of some hardware failure. In order to meet the modern requirements, the size and the complexity of current communication payloads are increasing significantly. Consequently, the optimal payload configuration, which was previously done manually by the engineers with the use of computerised schematics, is now becoming a difficult and time consuming task. Efficient optimisation techniques are therefore required to find the optimal set(s) of switch positions to optimise some operational objective(s). In order to tackle this challenging problem for the satellite industry, this work proposes two Integer Linear Programming (ILP) models. The first one is single-objective and focuses on the minimisation of the length of the longest channel path, while the second one is bi-objective and additionally aims at minimising the number of switch changes in the payload switch matrix. Experiments are conducted on a large set of instances of realistic payload sizes using the CPLEX solver and two well-known exact multi-objective algorithms. Numerical results demonstrate the efficiency and limitations of the ILP approach on this real-world problem.

Keywords: exact optimisation; satellite payload reconfiguration; integer linear programming;

1. Introduction

The payload of a communication satellite is in charge of receiving and filtering the uplink signals, which are sent from the Earth, applying frequency conversion, amplification and finally transmitting the signals on the downlink, i.e. back to the Earth. To fulfill these requirements, a satellite payload consists of a number of hardware components, such as amplifiers, channel filters and interconnected switches. The channels, that are used for the transmission of the dedicated services, such as satellite television and Internet, are characterised by a certain frequency and bandwidth. Each channel may be associated to one or more customers of the satellite operator. The switches, that compose the payload switch matrix, are used to route each channel to an appropriate amplifier and finally

\textsuperscript{*}Corresponding author. Email: apostolos.stathakis@uni.lu
to a corresponding output. They can be configured, i.e. the position of each switch can be changed remotely using specific commands sent from Earth stations. Each possible position allows different paths for the routing of the channels.

The design of the payload switch matrix is done by the manufacturer of the satellite in order to ensure redundancy and flexibility to the routing of the channels. Redundancy is required when a failure occurs to a payload component (e.g. an amplifier), and flexibility is necessary due to business or operational needs that may arise. These required properties come to the expense of large and more complex switch matrices. As a consequence, the process of finding optimal payload switch matrix configurations in the operational context, is now becoming a hard and error prone problem for the payload engineers. Some commercial softwares already permit to solve the payload configuration problem, but these are black-box approaches, only focus on single-objective optimisation and list all feasible solutions.

Therefore, this work aims to help the payload engineers on how to optimally configure a payload switch matrix in the operational context, when a given set of channels has to be activated. The target is to obtain the best or the set of best (non-dominated) solutions using exact methods. More precisely, given a payload switch matrix, we focus on the initial configuration (IC) problem which consists in finding an optimal switch matrix configuration for connecting an initial set of channels, when no other channels are connected in the payload. This article extends the previous work from Stathakis et al. (2012a) by defining the two cases that the IC problem may arise, namely the planning and the in-orbit case. During the planning case, where the satellite is not operated in-orbit yet, the optimal solutions can help the engineers to define the initial positions of the switches according to the channels that will be activated, based on a long-term business planning. The problem is considered as a single-objective one, with a new objective to be minimised, i.e., the longest path length (LPL). An extension of the ILP mathematical model presented in Stathakis et al. (2012a) is thus proposed in order to optimise this objective. When the satellite is operated in-orbit, the problem becomes multi-objective as both the longest path length and the total number of switch changes have to be minimised. In this case, the process of finding an optimal configuration is a time critical process for the engineers. We thus set a time constraint of ten minutes, defined by the engineers, and the performance of exact multi-objective algorithms, on large realistic payloads, on a single CPU core is compared. The engineers can use the optimal solutions in this case to change the positions of the switches via telecommands.

The remainder of this paper is organised as follows. The next section provides the related work on the subject. Section 3 details the considered problem and the new planning and in-orbit initial configuration cases. The proposed mathematical models are presented in Section 4 and the experimental results are analysed in Section 5. Lastly, the conclusions and some perspectives for future work are presented.

2. Related Work

The market of satellite communications is rapidly increasing. In order to adapt to its evolutions and meet the current demands, the satellite operators aim apart from providing the maximum possible capacity, to reach certain levels of flexibility in operations. Seven different flexibility levels in the domain have been presented in Balty, Gayrard, and Agnieray (2009). They concern among others, flexibility at orbit location, allowing the same mission to be provided from several possible orbit locations, flexibility at the frequency plan, that concerns the number and bandwidth of channels, and flexibility at the connectivity and routing level. At this latter level, which is studied in this article, Radio Frequency (RF) switches are used, to achieve the highest possible flexibility at the
routing and to ensure the redundancy in case of failures that may take place during the satellite lifetime.

The first optimisation problem, which is faced by satellite manufacturers when constructing the spacecrafts, is the design of an optimal switch matrix topology, given the operational requirements. Switches are expensive components and thus designing such a topology that will satisfy all the routing requirements, for a given number of channels and amplifiers, while minimising the cost, is of importance. Some research works have tackled this design problem like in Amini et al. (2010) where a graph-based analysis is used to minimise the number of needed switches, while ensuring a fault-tolerant switch matrix design. In Bermond, Havet, and Toth (2006) the authors considered priorities on the channel inputs for designing the switch matrix topology, where it should be possible in the network to route $p$ priority channels inputs to the best $p$ quality amplifiers for any set of $k$ faulty and $p$ best quality amplifiers.

However, from the satellite operator point of view which is of our interest, efficient techniques are also required by the engineers to find optimal configurations and reconfigurations in an operational context, for a given payload switch matrix. Commercial software packages exist on the market, like TRECS (Kratos Integral Systems International (2013)) and Smartrings (Chaumon et al. (2006)). Details though concerning the algorithms and the models used by both packages are not accessible due to commercial restrictions. Smartrings applies a recursive search to compute all possible payload configurations. The algorithm is controlled by constraints like the number of switches used or the number of interrupted channels. After the solutions have been generated, the engineers can rank them based on the selected criteria and choose the most efficient configuration. Similarly, TRECS uses an algorithm to find all feasible solutions and sorts them by output signal quality, while rejecting many millions of non-feasible solutions. The algorithm allows constraints like minimising the number of changes, limiting the number of changes to $n$ or considering the first $n$ solutions. However, the main drawback of those packages is the lack of flexibility. Their closed APIs do not allow efficient interaction and integrations in company workflows. Besides, the solver and the model used can not be changed or customised based on the problem case. Finally, enumerating all feasible solutions is not of interest for the engineers. Many generated solutions will be of bad quality and this process might be time consuming for large payloads. Additional time will be also needed for post-processing, in order to select the solution or set of solutions that better fit the criteria.

From the academic point of view, only few works have tackled the considered problem. In Gulgonul et al. (2012), a recursive algorithm was proposed to perform a breadth-first-search (BFS) in order to find all feasible paths that connect channels to amplifiers. Experiments showed the efficiency of the proposed method on a small switch network (i.e 7 switches and 9 amplifiers). However, it can be expected that for larger problems the BFS algorithm will be limited due to its time complexity as every vertex and every edge will be explored in the worst case.

Instead of enumerating all feasible solutions, a single optimal solution for the single-objective problem or a limited set of non-dominated solutions for the multi-objective problem, based on well defined objective functions, is of interest. The considered problem has similarities with the classical network optimization problems, such as the Integer Multicommodity Flow Problem (IMCF). Such network flow models are widely applied in the filed of telecommunications for optimal routing and network design (Minoux (2006)). They are also used in different applications for instance on project scheduling problems (Chen et al. (2014)). However, the direct application of these models is not possible due to the specificities of the considered problem. In the payload switch matrix context, several channels are crossing simultaneously the network, each one using a single path. One key difference though, is the notion of the switch position that modifies the topology of
Figure 1. Payload simplified example and possible solution connecting channels 1 and 3 to amplifiers 2 and 4 respectively.

the network. Thus, new specific flow propagation constraints have to be defined. To this scope, an ILP optimisation model was proposed and validated in Stathakis et al. (2012a). ILP formulations have been used for solving optimization problems related to communication satellites such as planning an optimal frequency plan (number, bandwidth and spacing of channels) and maximising the number of users that the system can serve in a service area Alouf et al. (2005). Integer programming formulations are also used for other engineering applications since long time such as the design of water distribution networks Awumah, Bhatt, and Goulter (1989). For the switch matrix configuration optimization problem, the model presented in Stathakis et al. (2012a) allows the minimisation of the number of switch changes. In Stathakis et al. (2012b) the optimisation model was enhanced to minimise the channel interruptions when the payload has to be reconfigured, i.e., when new channel are activated. Both works dealt with single-objective problems.

This work proposes an extension of the mathematical model presented in Stathakis et al. (2012a), in order to solve exactly two new variants of the initial configuration problem: planning and in-orbit. In the planning variant, a novel operational objective has to be minimised, i.e the LPL. The in-orbit case considers a multi-objective problem to minimise simultaneously the LPL and the total number of switch changes, within an acceptable time for the engineers. We apply for the first time to the considered problem two well-known multi-objective exact algorithms, namely the $\epsilon$-constraint method (Chankong V. (1983)) and the adaptive $\epsilon$-constraint method (Laumanns, Thiele, and Zitzler (2006)) in order to provide the Pareto front from which the engineers can extract the optimal solution that better fits their requirements.

3. Problem Description

The considered problem is detailed in this section. At first, a brief description of the satellite payload is provided in Section 3.1. The general problem of payload reconfiguration is classified in three related subproblems namely the initial configuration problem, the reconfiguration and the restoration problem. In Section 3.2, the initial configuration problem, which is studied in this work, is presented.

3.1 Satellite payload

As aforementioned, a payload is composed of different hardware components, including channels, amplifiers and switches organised in a matrix. The input signals, referred to as channels, cross the input switch matrix and are guided for amplification to an appropriate
amplifier. A simplified payload example is given on the left hand side of Figure 1, with 8 input channels, 4 amplifiers and 16 switches. Different types of switches may be used, each of them having different sets of possible positions allowing different connectivities. In the example of Figure 1, switches of type R and C (first column) are used. The 4 possible positions of an R-type switch are shown in Figure 2. Other switches, like C-type switches, have only 2 possible positions as shown in Figure 3. A link is a connector between any two payload components. Each switch has 4 neighbouring links. There is only one switch connected to each channel, and only one switch connected to each amplifier. When a switch changes position different links are connected allowing different communication paths. It is considered in this work that any channel can be connected to any amplifier. The proposed mathematical models though, can be easily adapted for the cases where only some amplifiers are appropriate for a specific subset of channels. As has been previously mentioned, the topology of the switch matrix is designed with the minimum cost while ensuring the routing requirements for a given number of channels and amplifiers. Therefore, the switch matrix is not necessarily symmetric, as illustrated in the simplified example of Figure 1. A payload switch matrix can have any topology and channels can be placed at different locations of the matrix, i.e. some channels may require longer or shorter paths to reach some amplifiers. The channels, after being amplified, are guided through a similar switch matrix to the corresponding outputs.

The general problem of finding optimal payload configurations, consists in defining the positions of the switches that allow the connection of a given set of channels to the amplifiers, while optimising one or more objectives.

In some previous work Stathakis et al. (2013), a first classification of the problem in three related cases was proposed, namely the initial configuration, the reconfiguration and the restoration problem. The initial configuration problem consists in finding an optimal payload configuration for connecting an initial set of channels in an empty payload (i.e., without any pre-connected channel paths that carry services to customers). We here
propose to further refine the initial configuration problem in planning or in-orbit phase, which will be detailed in the next sections. A simplified example of an initial configuration problem is shown in Figure 1. It illustrates one possible solution for connecting channel 1 to amplifier 2 and channel 3 to amplifier 4. Channel 1 follows a path composed of 8 links (7 switches used in the path) and channel 3 follows a path composed of 7 links (6 switches used in the path). 11 switches have changed from their initial position.

The reconfiguration problem occurs when there exists a set of pre-connected channel paths carrying services and some additional channels must be activated due to some new demands. In this case, interruptions of the pre-connected channels have to be avoided as they impact the provided services. An example of this case is provided in Figure 4, where channel 3 is initially connected and in the final payload configuration, channel 6 has been activated. Channel 3 has been interrupted, i.e. re-connected through a different path.

The third problem, named as the restoration problem, arises when one or more amplifiers and/or switches fail. The connected channels that are affected by those failures need to be re-routed through alternative paths. A restoration problem case example is shown in Figure 5 where a failure in amplifier 4 occurs, and channel 3, which was initially connected to this amplifier, is rerouted in the solution example to amplifier 3.

All the aforementioned subproblems have some common operational objectives. For example, the number of required switch changes has to be minimised. Changing the position of a switch may cause its mechanical failure (e.g. the switch may get stuck in a usable or a non-usable position). Since the satellite is in-orbit, the switch cannot be replaced and this failure is permanent. The reconfiguration and the restoration problem have an additional objective which is minimising the interruptions of the already connected channels that carry services. Another objective of high interest for the engineers
is the minimisation of the length of the longest channel path (LPL). Long paths should be avoided for two reasons. Firstly, longer paths imply high signal attenuation, and secondly when many switches are used, the flexibility for future reconfiguration processes is restricted.

As this work focuses on the initial configuration problem, a more detailed description of this case is provided in the next section.

3.2 Initial configuration problem

The initial configuration problem is considered when the payload is "empty" (i.e. no pre-existing channel paths). This case may occur either in a planning phase, when the satellite is still on Earth, or in an in-orbit phase, when the satellite is operated in-orbit. We distinguish those cases as single objective and multi-objective problem cases.

3.2.1 Planning case - single objective problem

In the planning case, the satellite is still on Earth and engineers have to define an initial configuration of the payload switch matrix based on the channels they want to activate according to their long term business planning. The objective is to set appropriately the initial positions of the switches in order to allow the activation of the planned channels during the satellite lifetime period. Since the satellite is not in-orbit, the number of switch changes is not of importance yet, indeed failed switches can still be replaced. The planning case is thus a single objective optimisation problem where the longest channel path (LPL) has to be minimised. No restriction applies on the computational time required for this optimisation process.

More formally, given \( n \) channels to connect, the solution to the problem consists of the set of positions of all the switches that allows the construction of the \( n \) paths from each required channel to an amplifier. The objective is to find the solution such that the length of the longest channel path is minimum. Let \( k \) be the number of switches in the switch matrix and \( C \) the set of channels to connect. Let \( \text{path}_c, c \in C \) be the function returning the length of the channel path \( c \) in number of switch crossings or links used. The solution vector \( p = (\text{pos}_1, \ldots, \text{pos}_k) \) of size \( k \) denotes the position of each switch. The objective is to find a solution \( p \) which connects the \( n \) channels and minimises:

\[
f_1(p) = \max_{c \in C} \{\text{path}_c(p)\} \tag{1}
\]

3.2.2 In-orbit case - multi-objective problem

The initial configuration problem may also occur while the satellite is in-orbit. This can happen for example when all the transmitted channels must be replaced by a totally new set of channels, due to some new business needs or technical constraints. In this case, for the new set of channels to connect, the problem becomes bi-objective, since not only the LPL has to be minimised but also the number of required switch changes. The latter is of importance as changing the position of a switch may cause a mechanical failure which, while the satellite is in-orbit, is permanent. Contrary to the planning case, when the satellite is in-orbit, the payload configuration becomes a time critical operation. Optimisation approaches must therefore provide a solution in acceptable time for the engineers, which is set to 10 minutes in this work.

More formally, given \( n \) channels to connect, the solution to the problem consists in finding the set of positions of all the switches used in the switch matrix, that allows the construction of the \( n \) paths, from each required channel to an amplifier. Let \( S \) of size \( k \) be the set of switches in the switch network and \( C \), of size \( n \), the set of channels to connect. Let \( \text{pinit}_s \) be the initial position of switch \( s \in S \). Let \( \text{change}_s, s \in S \) be a binary...
variable denoting whether switch \( s \) has changed from its initial position or not. It holds:

\[
change_s = \begin{cases} 
1 & \text{if } pinit_s \neq pos_s, \\
0 & \text{otherwise.}
\end{cases}
\] (2)

The solution vector \( p = (pos_1, \ldots, pos_k) \) of size \( k \) denotes the position of each switch, and the objective is to find a solution \( p \) which connects the \( n \) channels and minimises both:

\[
f_1(p) = \max_{c \in C} \{ \text{path}_c(p) \}
\] (3)

\[
f_2(p) = \sum_{s \in S} change_s
\] (4)

4. Methodology

The proposed ILP mathematical models for solving the two initial configuration problem cases are presented in this section. Section 4.1 provides the single objective model for solving the planning case. Section 4.2 details the multi-objective approach, including the description of the two exact methods.

4.1 Planning case - Mathematical Model

The basic variables of the proposed model, are the flow variables that are associated to each link and the integer position variables that are associated to each switch. Figure 6 illustrates the associated variables and constraints for each of the basic components of the considered problem. Given a channel to connect, a unique integer positive flow value is assigned to its neighboring link. The value is propagated through the switch matrix, according to the flow propagation constraints that are defined for each component of the payload. An amplifier is considered as used, as long as its input link has a positive flow value. The capacity on each link is equal to one, i.e. a physical link can be used by only one channel. In more details, the optimisation model consists of the following constants, sets, variables, constraints and objectives.

4.1.1 Constants and Sets:

Let define:
- \( q \) as the number of channels to connect.
- \( n \) as the maximum number of states (positions) of the switches of the switch matrix (e.g. if \( R \) switches with 4 states are used, \( n=4 \)).
- \( P \) as a set of size \( n \), with integer values from 0 to \( n-1 \), representing the set of positions a switch can have. The position of a switch refers to the number of steps the switch should take from its current state to reach a new state.
- \( S \) as the set of all switches.
- \( T \) as the set of all amplifiers.
- \( C \) as the set of all channels.
- \( L = CL \cup TL \cup SL \) as the set of all links. \( L \) consists of the following subsets:
CL as the set of all links connected to the channels.
- TL as the set of all links connected to the amplifiers. Let \( tl_{in} \) be the input link of amplifier \( t \).
- SL as the set containing the links between any two switches.

- \( l_0 \) as a special link, or link 0, when signal can not be propagated. For instance in Figure 2 with a switch in state 2, link \( a \) is connected with \( l_0 \).
- Let \( C_{conn} \subseteq C \), of size \( q \), to be the set of channels to connect. We represent each channel to connect with an integer value. Thus, let the elements of this set to be \( 1, 2, \ldots, q \), representing the 1st, 2nd, \ldots, \( q \)th channel to connect respectively.
- \( CL_{conn} \subseteq CL \) of size \( q \), as the set of links neighbouring with the input channels that will be connected. Let the elements of this set to be \( l_1, l_2, \ldots, l_q \), indicating that the neighbouring link of the channel to connect \( c_1 \) is \( l_1 \). Respectively, the neighbouring link of the channel to connect \( c_2 \) is \( l_2 \) etc. When starting the flow propagation the link \( l_1 \) will be assigned flow value 1, the link \( l_2 \) will be assigned flow value 2 etc.
- The matrix \( M \) of size \(|S| \times |P| \times |L| \times (|L| \cup \{l_0\})|\), describes which links are connected at each position of each switch:

\[
m_{s,p,l_j} = \begin{cases} 1 & \text{if } l_i \text{ is connected with } l_j \text{ via switch } s \text{ at position } p, \\ 0 & \text{otherwise.} \end{cases}
\]

Table 1 summarises the constants and sets used.

### 4.1.2 Variables

- Let define \( Pos \) of size \(|S|\), as the solution vector, providing the position for each switch. The initial state of each switch is considered as \( pos_s = 0 \). For instance, if an R-type switch (with 4 states) is initially in state 3, and in the solution \( pos_s = 1 \), then the switch will have to move to state 4. If \( pos_s = 0 \) the switch does not change position.
- Integer vector \( Flow \) of size \(|L|\), denoting the flow value (from 0 to \( q \)) that is carried by
Table 1. Constants and Sets.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q$</td>
<td>Number of channels to connect</td>
</tr>
<tr>
<td>$n$</td>
<td>Max number of switch positions</td>
</tr>
<tr>
<td>$P$</td>
<td>Set of positions</td>
</tr>
<tr>
<td>$S$</td>
<td>Set of switches</td>
</tr>
<tr>
<td>$T$</td>
<td>Set of amplifiers</td>
</tr>
<tr>
<td>$C$</td>
<td>Set of channels</td>
</tr>
<tr>
<td>$L = CL \cup TL \cup SL$</td>
<td>Set of links</td>
</tr>
<tr>
<td>$l_0$</td>
<td>Link where flow is not propagated.</td>
</tr>
<tr>
<td>$C_{conn}$</td>
<td>Set of channels to connect.</td>
</tr>
<tr>
<td>$CL_{conn}$</td>
<td>Set of the links of the channels to connect.</td>
</tr>
</tbody>
</table>

$L = CL \cup TL \cup SL$ is a set of links.

Each link has a flow associated with it. The flow of each link $l$ is denoted by $flow_l$. It follows:

$$flow_l = \begin{cases} 
  x & \text{with } 0 < x \leq q, x \in \mathbb{Z} \quad \text{if link } l \text{ is used by the } x^{th} \text{ channel to connect}, \\
  0 & \text{otherwise}.
\end{cases}$$

- Binary vector $B$ of size $|S| \times |P|$, is used to activate or deactivate the flow propagation constraints, such that:

$$b_{s,p} = \begin{cases} 
  1 & \text{if } \text{pos}_s = p, \\
  0 & \text{otherwise}.
\end{cases}$$

- Boolean vector $\text{Ampused}$ of size $|T|$, indicating whether an amplifier is active (used) or not.

- Binary variable $flow_{l,c}, \forall l \in L, \forall c \in C_{conn}$, indicating whether link $l$ is used by channel $c$ or not:

$$flow_{l,c} = \begin{cases} 
  1 & \text{if } l \text{ is used by channel } c, \\
  0 & \text{otherwise}.
\end{cases}$$

- Integer variable $\text{path}_c, \forall c \in C_{conn}$, indicating the length of the path of channel $c$.

- Integer variable $z$ denoting the length of the longest channel path.

The presented variables are summarised in Table 2.

Table 2. Variables.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z \in \mathbb{Z}$</td>
<td>Longest Path Length</td>
</tr>
<tr>
<td>$pos_s \in \mathbb{Z}, \forall s \in S$</td>
<td>Position of each switch</td>
</tr>
<tr>
<td>$flow_l \in \mathbb{Z}, \forall l \in L \cup {l_0}$</td>
<td>Flow of each link</td>
</tr>
<tr>
<td>$b_{s,p} \in {0; 1}, \forall s \in S, \forall p \in P$</td>
<td>Denotes whether switch $s$ is at position $p$</td>
</tr>
<tr>
<td>$\text{Ampused}_t \in {0; 1}, \forall t \in T$</td>
<td>Denotes whether amplifier $t$ is used</td>
</tr>
<tr>
<td>$flow_{l,c} \in {0; 1}, \forall l \in L, \forall c \in C_{conn}$</td>
<td>Denotes whether link $l$ is used by channel $c$</td>
</tr>
<tr>
<td>$\text{path}<em>c \in \mathbb{Z}, \forall c \in C</em>{conn}$</td>
<td>Length of the path of each channel.</td>
</tr>
</tbody>
</table>
4.1.3 Constraints

- To start the flow distribution, flow values are assigned to each link \( l_i \) from the set \( CL_{conn} \) that will carry the flow value of the \( i^{th} \) channel:

\[
flow_{l_i} = 1, \quad \forall l_i \in CL_{conn}.
\]

- The integer variable \( flow_l \) must be equal to:

\[
flow_l = \sum_{c \in C_{conn}} c \cdot flow_{l,c}, \quad \forall l \in L.
\]

- To avoid flow paths starting from an input channel and returning in another input channel, flow values must be set to 0 for all the unused input channels:

\[
flow_{l_i} = 0, \quad \forall l_i \in \{\{CL\} - \{CL_{conn}\}\}.
\]

- It is ensured that a link is either not used or used by only a single channel:

\[
\sum_{c \in C_{conn}} flow_{l,c} \leq 1, \quad \forall l \in L.
\]

- The flow propagation constraints are expressed for each position of each switch and describe the established connections at each case. It is ensured, that based on each switch position, the connected links have the same flow value. Figure 2 shows the non-linear expressions for flow propagation constraints for all states of an R-type switch. The linear equivalent constraints can be expressed as follows:

\[
flow_{l_1} + b_{s,p} \cdot q - q \leq flow_{l_2} \leq flow_{l_1} - b_{s,p} \cdot q + q:
\]

\[
\forall s \in S, \forall p \in P, \forall l_1, l_2 \in (L \cup \{l_0\})^2, \text{ such as } m_{s,p,l_1,l_2} = 1.
\]

- The following constraints ensure that only one position is selected for one switch:

\[
\sum_{p \in P} b_{s,p} = 1, \quad \forall s \in S.
\]

\[
pos_s = \sum_{p \in P} p \cdot b_{s,p}, \quad \forall s \in S.
\]

- An amplifier is used if its input link has a positive flow value. The number of active amplifiers has to be equal to the number of connected channels:

\[
ampused_t \cdot q \geq flow_{l_{int}}, \quad \forall t \in T.
\]

\[
\sum_{t \in T} ampused_t = q.
\]
Table 3. Constraints.

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{flow}<em>{i,i} = 1 ), ( \forall i \in C</em>{\text{L}\text{conn}} )</td>
<td>Starting flow distribution for the channels to connect.</td>
</tr>
<tr>
<td>( \text{flow}<em>{l,i} = 0 ), ( \forall l,i \in { { C } \setminus { C</em>{\text{L}\text{conn}} } } )</td>
<td>Set flow ( 0 ) for the non-used channels.</td>
</tr>
<tr>
<td>( \text{flow}<em>{l,i} + b</em>{p,l} \cdot s - q \leq \text{flow}<em>{l,i} \leq \text{flow}</em>{l,i} - b_{p,l} \cdot s + q ), ( \forall s \in S ), ( \exists i ), ( m_{s,l,p,i} = 1 )</td>
<td>Flow propagation constraints for each switch.</td>
</tr>
<tr>
<td>( \sum_{p \in P} b_{p,l} = 1 ), ( \forall s \in S )</td>
<td>Unique position for each switch.</td>
</tr>
<tr>
<td>( \text{pos}<em>s = \sum</em>{p \in P} p \cdot b_{s,p} ), ( \forall s \in S )</td>
<td>Position of each switch.</td>
</tr>
<tr>
<td>( \sum_{t \in T} \text{ampused}_t = q )</td>
<td>Number of used amplifiers equals to number of channels to connect.</td>
</tr>
<tr>
<td>( \sum_{c \in C_{\text{conn}}} \text{flow}_{l,c} \leq 1 ), ( \forall l \in L )</td>
<td>Each link is used by at most one channel.</td>
</tr>
<tr>
<td>( \text{flow}<em>l = \sum</em>{c \in C_{\text{conn}}} c \cdot \text{flow}_{l,c} ), ( \forall l \in L )</td>
<td>Flow value of each link.</td>
</tr>
<tr>
<td>( \text{path}<em>c = \sum</em>{l \in L} \text{flow}<em>{l,c} ), ( \forall c \in C</em>{\text{conn}} )</td>
<td>Length of the path of each channel.</td>
</tr>
<tr>
<td>( \text{flow}_{l_0} = 0 )</td>
<td>Link ( l_0 ) never carries any signal:</td>
</tr>
<tr>
<td>( \text{path}<em>c \leq z ), ( \forall c \in C</em>{\text{conn}} )</td>
<td>Longest path length</td>
</tr>
</tbody>
</table>

- Link \( \{ l_0 \} \) never carries any signal:

\[
\text{flow}_{l_0} = 0.
\]

- The length of the path of each channel is equal to the sum of all links that carry the channel:

\[
\text{path}_c = \sum_{l \in L} \text{flow}_{l,c}, \quad \forall c \in C_{\text{conn}}.
\]

- The length of each path must be smaller or equal to the length of the longest path:

\[
\text{path}_c \leq z, \quad \forall c \in C_{\text{conn}}
\]

The above mentioned constraints are summarised in Table 3.

### 4.1.4 Objectives

- The objective is to minimise the length of the longest channel path:

\[
\min z
\]

### 4.2 In-orbit case - multi-objective exact optimisation

In this case, the aim is to minimise simultaneously the LPL and the total number of switch changes. These are conflicting objectives as one solution may require few switch changes but the channels may follow long paths, whereas in another solution the channel paths can be shorter with more switch changes. In such multi-objective optimisation problems, we do not deal with a unique optimal solution but we are interested in generating the Pareto front of the optimal solutions (Ehrgott and Gandibleux (2002)).

In order to obtain the optimal Pareto front for the in-orbit case, two multi-objective exact algorithms have been considered in this study, namely the \( \epsilon \)-constraint method (Chankong V. (1983)) and the adaptive \( \epsilon \)-constraint method (Laumanns, Thiele, and Zitzler (2006)). Both approaches transform the multi-objective problem as a set of constrained single-objective problems. One objective function is used as the objective, while the second objective function is used as a constraint, which permits to obtain different
solutions in the front. The next two subsections detail the $\epsilon$- and adaptive $\epsilon$-constraint methods. The last section describes the adaptation to the ILP model presented in section 4.1 in order to solve the considered multi-objective problem.

4.2.1 The $\epsilon$-constraint method

The $\epsilon$-constraint method is based on the optimisation of one of the objectives, defined as the primary one, while considering the other objectives as constraints bound by some allowable levels $\epsilon_i$. Supposing we have a procedure, namely $\text{opt}(f, \epsilon, \epsilon')$ that returns the optimal solution $x$ of the following constrained problem:

$$\begin{align*}
\text{lex min } f(x) &= (f_1(x), f_2(x)) \\
\text{subject to } \epsilon_i < f_2(x) &\leq \epsilon_i' \\
x &\in X
\end{align*}$$

The “lex min” denotes the lexicographic minimisation of the objectives. The pseudo-code of the $\epsilon$-constraint method is provided in Algorithm 1. The lower and upper bounds $f, \bar{f}$ are known for each objective. At each iteration of the algorithm, the constrained single objective problem is solved (line 5) and if a new solution is found which is not dominated, the solution is added to the Pareto set (lines 6, 7). The constraint bounds of the second objective are modified with a predefined constant $\delta$ (line 9). The algorithm stops when the region between the lower and upper bounds has been searched.

4.2.2 The adaptive $\epsilon$-constraint method

The adaptive $\epsilon$-constraint method additionally uses information from the objective space (when available) during the search. In this case the bounds of the objectives do not need to be known. Supposing we have a procedure, called $\text{opt}(f, \epsilon')$ that solves the following constrained problem:

$$\begin{align*}
\text{lex min } f(x) &= (f_1(x), f_2(x)) \\
\text{subject to } f_2(x) &< \epsilon' \\
x &\in X
\end{align*}$$
and returns the optimal solution $x$ of the problem or null in case of unfeasibility. The pseudocode of the method is provided in Algorithm 2.

**Algorithm 2** adaptive $\epsilon$-constraint method

1. $P := \emptyset$
2. $\epsilon := \infty$
3. while $\text{opt}(f, \epsilon)$ not infeasible do
4. \hspace{1em} $x := \text{opt}(f, \epsilon)$
5. \hspace{1em} if $\nexists x' \in P$ such that $x' \succ x$ then
6. \hspace{2em} $P := P \cup \{x\}$
7. \hspace{1em} end if
8. \hspace{1em} $\epsilon := f_2(x)$
9. end while
10. **Output:** Set of Pareto-optimal decision vectors $P$

At each iteration of the algorithm, the constrained single objective problem is solved (line 4) and if a new non-dominated solution is found it is added to the Pareto set (lines 5,6). The constraint bound for the second objective function is modified based on the found value $f_2(x)$ (line 8). As soon as the problem is infeasible the algorithm stops.

### 4.2.3 Mathematical Model

In order to solve the multi-objective problem, the following variables, constraints and objectives are added to the ILP model. These permit to minimise the second objective, i.e. the number of switch changes, and apply the aforementioned exact approaches.

#### 4.2.3.1 Variables.
- Binary vector $\text{Change}$ of size $|S|$, indicating whether a switch has to change from its initial state or not.

$$change_s = \begin{cases} 1 & \text{if } pos_s > 0, \\ 0 & \text{otherwise}. \end{cases}$$

#### 4.2.3.2 Constraints.
- It is ensured that $change_s = 1$, if and only if $pos_s > 0$:

$$change_s \cdot n \geq pos_s, \hspace{1em} \forall s \in S.$$

$$change_s \leq pos_s, \hspace{1em} \forall s \in S.$$

#### 4.2.3.3 Objectives.

$$\text{Min } \sum_{s \in S} change_s$$
4.2.3.4 Complete ILP model

The corresponding complete ILP model is presented below.

Variables:
- \( z \in \mathbb{Z} \)
- \( pos_s \in \mathbb{Z} \) \( \forall s \in S \)
- \( change_s \in \{0; 1\} \) \( \forall s \in S \)
- \( flow_l \in \mathbb{Z} \) \( \forall l \in L \cup \{l_0\} \)
- \( b_{s,p} \in \{0; 1\} \) \( \forall s \in S, \forall p \in P \)
- \( ampused_t \in \{0; 1\} \) \( \forall t \in T \)
- \( flow_{l,c} \in \{0; 1\} \) \( \forall l \in L, \forall c \in C_{conn} \)
- \( path_c \in \mathbb{Z} \) \( \forall c \in C_{conn} \)

Objective Functions:
- Min \( z \) (objective we chose to constrain)
- Min \( \sum_{s \in S} change_s \)

Constraints:
- \( flow_{l_1} = 1 \) \( \forall l_1 \in C_{L_{conn}} \)
- \( flow_{l_2} = 0 \) \( \forall l_2 \in \{\text{CL} - \{C_{L_{conn}}\}\} \)
- \( flow_{l_1} + b_{s,p} \times q - q \leq flow_{l_2} \leq flow_{l_1} - b_{s,p} \times q + q \)
- \( \sum_{p \in P} b_{s,p} = 1 \) \( \forall s \in S \)
- \( pos_s = \sum_{p \in P} p \times b_{s,p} \) \( \forall s \in S \)
- \( change_s \times n \geq pos_s \) \( \forall s \in S \)
- \( change_s \leq pos_s \) \( \forall s \in S \)
- \( ampused_t \times q \geq flow_{l_{1,m}} \) \( \forall t \in T \)
- \( \sum_{t \in T} \sum_{c \in C_{conn}} flow_{l,c} \leq 1 \) \( \forall l \in L \)
- \( flow_{l} = \sum_{c \in C_{conn}} c \times flow_{l,c} \) \( \forall l \in L \)
- \( path_c = \sum_{l \in L} flow_{l,c} \) \( \forall c \in C_{conn} \)
- \( path_c \leq z \) \( \forall c \in C_{conn} \)
- \( flow_{l_0} = 0 \)

5. Experimental Results

The analysis of the obtained experimental results is presented in this section. At first, information about the experimental setup is provided in Section 5.1. The results on the planning case and in-orbit operational case are detailed in Sections 5.2 and 5.3 respectively.

5.1 Experimental Setup

Information about the payload instances tackled is provided in Table 4. Two different switch matrices were used that are composed of 50 and 100 switches, with up to 23 and 35 amplifiers, respectively. These correspond to realistic sizes of current and upcoming
Table 4. Payload problem instances.

<table>
<thead>
<tr>
<th></th>
<th>Small Payload</th>
<th>Large payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of switches</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>Maximum nb. of channels to connect</td>
<td>23</td>
<td>35</td>
</tr>
<tr>
<td>Nb. of channels to connect</td>
<td>8, 13, 18, 23</td>
<td></td>
</tr>
<tr>
<td>Nb. of channels sets per size</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Nb. of initial payload status (switch changes objective only)</td>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>

communication payloads. In real business context, the specific channels to connect are provided to the engineers. An optimal switch matrix configuration has then to be defined in order to allow their connection. The number of channels to connect may be from 1 to the maximum number of available amplifiers. In this work, 30 different sets of channels of size 8, 13, 18 and 23 have been chosen uniformly at random. This permits to have statistical confidence in the results about the validity and efficiency of the method and to evaluate the impact of the choice of channels.

When considering the minimisation of the number of switch changes, the initial payload status, that refers to the initial set of positions of the switches, will influence the solution quality. For the 50 switches payload, the number of possible statuses is $4^{50}$ as each considered switch has 4 possible positions. Thus, 30 initial payload statuses chosen uniformly at random, are additionally considered when this objective is optimised. A total of 900 instances is therefore tackled per given size of channels to connect.

All the experiments were performed using the HPC platform facilities of the University of Luxembourg (Varrette et al. (2014)), on an homogenous type of nodes with Xeon 6C, 2.26GHz on a single core with 24GB of RAM. A single CPU core was used in order to create a benchmark that will be used for the comparisons of the experimental results. The exact solver IBM ILOG CPLEX 12.4.0.0 was used.

5.2 Planning Case - Single Objective Optimisation

In this section the experimental results for the initial configuration planning case are analysed. For these experiments two termination conditions were used. The first one is set to 120 hours, defined by technical reasons, which permits to first assess if the considered instances are solvable exactly, and if so to analyse the computational time upper bound. The second termination condition is set to 10 minutes in order to analyse the behavior of the solver within this strict constraint. This analysis will help us identifying which objective to optimise and which one to constraint when applying the multi-objective methods.

5.2.1 50 switches payload

The results when minimising LPL are provided in Table 5 for each of the above mentioned termination conditions. At first, the hit rate is displayed, which denotes the percentage of the instances solved exactly. Besides, the average fitness value, the average computational time as well as the minimum and the maximum required computational time in seconds are provided.

The first remark is that when minimising LPL, not only instances could be solved exactly, even after using the first termination condition of 120 hours. When connecting 8 channels, 93.33% of the instances were solved whereas for the sets of 18 and 23 channels the hit rate is 56.66%. With the termination condition of ten minutes, the hit rate is significantly lower. 90% of the instances were solved when 8 channels are connected, whereas for the sets of 18 channels to connect the hit rate drops to 6.66%. Finally, none instance could be solved when connecting 23 channels.

A high standard deviation of the required computational time is observed in all instances that indicates the influence of the set of channels to connect. As an example,
Table 5. Minimising LPL on 50 switches payload.

<table>
<thead>
<tr>
<th>Channels</th>
<th>Hitrate</th>
<th>Avg. Fitness</th>
<th>Avg. Time</th>
<th>Min Time</th>
<th>Max Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>120 hours</td>
<td>8</td>
<td>93.66%</td>
<td>2.83 ± 0.22</td>
<td>6013.9 ± 3182.98</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>36.66%</td>
<td>3.63 ± 0.50</td>
<td>14264.47 ± 37931.85</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>56.66%</td>
<td>3.94 ± 0.42</td>
<td>46213.47 ± 48106.44</td>
<td>1.61</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>56.66%</td>
<td>4.94 ± 0.24</td>
<td>32371.45 ± 36075.48</td>
<td>1549.16</td>
</tr>
<tr>
<td>10 min</td>
<td>8</td>
<td>90%</td>
<td>2.81 ± 0.48</td>
<td>3141 ± 12.89</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>13.33%</td>
<td>3 ± 0</td>
<td>10.67 ± 13.64</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>6.66%</td>
<td>3 ± 0</td>
<td>4.52 ± 4.12</td>
<td>1.61</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>0%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7. 50 switches payload. Average time (in seconds) of the set of instances with the maximum and minimum length of longest path.

when 8 channels are connected, the average required time is 6013.36 seconds with maximum time among the solved instances 168288.7 seconds and minimum time only 0.19 seconds. When 13 channels are connected, the standard deviation of the time reaches 73931.85 seconds. As previously mentioned, the switch matrix, which is designed in order to fulfill the operational requirements with the minimum cost, is not symmetric and the channels can be placed at different locations. Therefore, based on the selected set of channels to connect, some instances may be solved easier compared to others. Figure 7 illustrates the average required time for the set of instances that had the worst fitness value (highest LPL) and the set of instances with the best fitness values (lowest LPL) considering the 50 switches payload and the termination condition of 120 hours. It can be observed that the instances with the worst fitness are the most difficult to solve while the instances solved in the shortest time have smaller fitness values. For instance, for the case of 13 channels to connect, the instances with the best fitness were solved in average after 8.12 seconds where the solved instances with the worst fitness required in average 66883 seconds.

5.2.2 100 switches payload

The experimental results on the 100 switches payload are presented in Table 6. Using the 120 hours termination condition, 70% of the instances of 8 channels to connect were solved whereas for the sets of 18 and 23 channels the hit rate is only 20% and 10% respectively. Considering the termination condition of 10 minutes, 66.66% of the instances were solved of size 8 channels to connect, whereas for the sets of 13 and 18 channels the hit rate is 30% and 10% respectively. None instance was solved for the cases of 23 channels to connect.

As denoted from the high standard deviation, the selected set of channels to connect
Table 6. Minimising LPL on 100 switches payload.

<table>
<thead>
<tr>
<th>Channels</th>
<th>Hitrate</th>
<th>Avg. Fitness</th>
<th>Avg. Time</th>
<th>Min Time</th>
<th>Max Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>120 hours</td>
<td>8</td>
<td>70%</td>
<td>2.95 ± 0.49</td>
<td>257.05 ± 940.13</td>
<td>0.16</td>
</tr>
<tr>
<td>8</td>
<td>70%</td>
<td>13</td>
<td>33.33%</td>
<td>3.3 ± 0.67</td>
<td>2016.76 ± 6061.92</td>
</tr>
<tr>
<td>18</td>
<td>20%</td>
<td>18</td>
<td>3.16 ± 0.408</td>
<td>20349.53 ± 6475.84</td>
<td>5.53</td>
</tr>
<tr>
<td>23</td>
<td>10%</td>
<td>23</td>
<td>5 ± 0</td>
<td>184541.9 ± 164183</td>
<td>50781</td>
</tr>
</tbody>
</table>

| 10 min | 8 | 66.66% | 2.9 ± 0.44 | 54.21 ± 144.24 | 0.16 | 489.68 |
| 13 | 30% | 3.11 ± 0.33 | 100.15 ± 120.42 | 0.94 | 344.33 |
| 18 | 10% | 3 ± 0 | 64.256 ± 86.16 | 5.53 | 163.17 |

Figure 8. 100 switches payload. Average time (in seconds) of the set of instances with the maximum and minimum length of longest path.

influences the required time. For the sets of 13 channels to connect the standard deviation of the time is 6061.92 seconds. For the cases of 18 channels the average computational time is 20349.53 with standard deviation 46475.84 seconds. As illustrated in Figure 8, similarly to the 50 switches payload, the instances with the worst fitness value (highest LPL) are the ones that required the highest computational time in average, whereas the instances with the best fitness (lowest LPL) were solved faster. When connecting 8 channels, the instances with the worst fitness required 2402 seconds whereas the ones with the lowest LPL only 0.32 seconds. For the sets of 18 channels to connect the easiest instances required 789 seconds whereas the ones with the longest paths required in average 115158 seconds. Lastly, for the sets of 23 channels to connect, the fitness of the solved instances was identical and thus no distinguish is done between the instances with the highest and the lowest fitness.

5.3 In-orbit Case - Multi-Objective Optimisation

The experimental results of the in-orbit problem case are presented in this section. While operating the satellite in-orbit, services are continuously provided to customers and as a result quick solutions are required to prevent long interruptions. Consequently, a realistic operational time constraint of 10 minutes is considered for all the experiments, which has been defined by payload engineers. The in-orbit case is a multi-objective optimisation problem where both the number of required switch changes and the LPL have to be minimised. Since the initial payload status (i.e. set of initial positions of all switches) will impact the switch changes objective, we start the analysis by minimising this objective individually. The aim is to experimentally quantify the influence of the payload status on the solution quality. 30 payload statuses were chosen uniformly at random. A total of
900 instances is thus tackled for each channel size.

5.3.1 Minimising the number of switch changes

In Table 7 the numerical results when the number of switch changes is minimised, for both payload sizes, are provided. As can be seen from the experimental results, minimising the number of switch changes is an easier problem compared to the problem of minimising LPL.

5.3.1.1 50 switches payload.

The hit rate is 100% in all cases for the 50 switches payload, i.e. all tackled instances were solved exactly. As an example, when connecting 8 channels to the 50 switches payload, the fitness value is in average 5.97 switch changes and for 23 channels to connect the fitness reaches an average of 19.82 switch changes.

The computational time is significantly smaller compared to the termination condition. When connecting the sets of 8 channels the average time is 1.04 seconds, with a maximum of 6.85 seconds, whereas when 23 channels are connected the average required time is 20.76 seconds with a maximum of 377.95 seconds.

In order to evaluate the influence of the initial payload status, Table 8 provides the results for each of the 30 instances of 23 channels to connect on the payload switch matrix with 50 switches. The columns present respectively the average and standard deviation of the fitness, the minimum and maximum fitness, the average and standard deviation of the computational time and the minimum and maximum computational time for the 30 different payload statuses. A remarkable standard deviation is observed in both fitness and computational time. The minimum and the maximum difference in the required time is 33.36 seconds and 376.17 seconds respectively with an average of 100.754 sec. The minimum difference in the value of the objective function is 6 changes and the maximum 13, with an average of 8 switch changes.

Furthermore, the most difficult instances, i.e. the instances 18 and 26 that required the highest computational time (46.52 seconds and 45.80 seconds respectively) are the ones with the highest fitness values (21.4 and 21.26 switch changes respectively). The easiest problem instance, i.e. instance 28 solved in average after 11.50 seconds, is the one with the minimum average fitness of 18.56 changes. This correlation is illustrated in Figure 9.

5.3.1.2 100 switches payload.

For the payload with 100 switches, as can be seen from Table 7, the hit rate is 100% for the sets of 8 and 13 channels to connect, whereas for the cases of 18 and 23 channels the hit rate is is 97% and is 43.22% respectively. The average fitness is for the sets of 8 channels 6.69 and for the sets of 23 channels 21.87 switch changes.

The average time when connecting 8 and 13 channels is only 1.52 and 13.56 seconds
For the sets of 23 channels to connect, the time reaches 215.76 seconds with maximum value 582.32 seconds.

5.3.2 Comparison of the Exact Bi-Objective Algorithms

For both exact algorithms, the number of switch changes was chosen to be the objective to optimise, while constraining the length of the longest path. This choice is justified by the fact that the switch changes is an easier objective to solve, as has been shown in the subsection 5.3.1. In all the experiments, the termination condition was set to 10 minutes as this problem occurs during the in-orbit case. For the $\epsilon$-constraint method we chose to set $\delta = 1$ to iteratively vary the constraint bound of the second objective. This is justified by the integer nature of the variable, the small number of Pareto optimal solutions in

![Average Fitness vs. Average Time](image)

Figure 9. Correlation between average fitness and average time (in seconds) for each instance of 23 channels to connect on the 50 switches payload.
Table 9. Hit Rate comparison of the exact methods and average and maximum number of Pareto solutions found in the fronts for the 50 switches payload.

<table>
<thead>
<tr>
<th>Channels</th>
<th>Hitrate</th>
<th>Avg. Pareto</th>
<th>max Pareto</th>
</tr>
</thead>
<tbody>
<tr>
<td>epsilon</td>
<td>8</td>
<td>73.11%</td>
<td>1.643 ± 0.720</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>3.44%</td>
<td>1.733 ± 0.520</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>2.66%</td>
<td>1.391 ± 0.583</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>adaptive</td>
<td>8</td>
<td>97.37%</td>
<td>1.658 ± 0.71</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>72.22%</td>
<td>1.622 ± 0.707</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>4.11%</td>
<td>1.416 ± 0.603</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>0.22%</td>
<td>1 ± 0</td>
</tr>
</tbody>
</table>

Table 10. Time(sec) comparison of the exact methods for the 50 switches payload.

<table>
<thead>
<tr>
<th>Channels</th>
<th>Time</th>
<th>min Time</th>
<th>max Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>epsilon</td>
<td>8</td>
<td>87.90 ± 135.517</td>
<td>0.640</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>176.15 ± 157.032</td>
<td>3.330</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>148.98 ± 144.297</td>
<td>7.78</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>adaptive</td>
<td>8</td>
<td>49.75 ± 96.717</td>
<td>0.210</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>58.80 ± 116.799</td>
<td>0.210</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>151.26 ± 189.427</td>
<td>4.03</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>501.02 ± 31.084</td>
<td>479.04</td>
</tr>
</tbody>
</table>

our instances and the low spread of solutions.

5.3.2.1 50 switches payload.

The summary results are provided in Table 9 and Table 10. In the first table, the hit rate, that represents the percentage of instances where the full Pareto front was found within the termination condition, is presented. In addition, the number of Pareto points found on the fronts is provided. The computational time required by the two algorithms is compared in Table 10.

As can be seen in Table 9, in terms of hit rate, the adaptive $\epsilon$-constraint method performs better than the $\epsilon$-constraint method. For the cases of 8 channels to connect, the adaptive method reached hit rate 97.77% compared to 73.11% obtained by the $\epsilon$-constraint method. When 13 channels are connected, the hit rate of the adaptive method remains relatively high, i.e. 72.22%, whereas it drops to 3.44% for the $\epsilon$-constraint. For the cases of 18 channels to connect, the adaptive $\epsilon$-constraint performs still better than the $\epsilon$-constraint with a hit rate of 4.11% compared to 2.66%. Lastly, when 23 channels are connected, none instance was solved by the $\epsilon$-constraint method, whereas the adaptive $\epsilon$-constraint method solved 0.22% of the instances.

Concerning the number of found Pareto solutions among the solved instances, it is relatively low. As an example, for the sets of 8 channels to connect, in average 1.643 and 1.658 optimal solutions have been found by the $\epsilon$-constraint method and the adaptive $\epsilon$-constraint method respectively. In all cases, the instance with the maximum number of Pareto points is solved by the adaptive $\epsilon$-constraint method. For the cases of 13 channels, the adaptive method managed to solve instances with 4 Pareto points in the front whereas for the $\epsilon$-constraint method the maximum number of Pareto points among the solved instances is 3. An example of a generated front with 4 Pareto optimal solutions is displayed in Figure 10.

The reason of the poor performance of the $\epsilon$-constraint method is the necessity of calculating the extreme points. This requirement implies calculation the length of the longest channel path individually. On the other hand, the adaptive $\epsilon$-constraint method performs faster. The number of switch changes is an easier objective to solve and the lexicographic optimisation, required by this method, does not influence significantly the
5.3.2.2 100 switches payload.

For the 100 switches payload, the hit rate as well as the average and maximum number of Pareto solutions found on the fronts are presented in Table 12. The better performance of the adaptive \(\epsilon\)-constraint method is again denoted from these results. The hit rate when connecting 8 channels is 52.11% for the adaptive \(\epsilon\)-constraint method whereas for the \(\epsilon\)-constraint method is 28.44%. The hit rate decreases very fast, indeed when 13 channels are connected, the \(\epsilon\)-constraint method solved 0.55%, and the adaptive method 2.88%. Similarly, for 18 channels the \(\epsilon\)-constraint method did not solve any instance whereas the adaptive \(\epsilon\)-constraint method solved 0.22% of the instances. Finally none of the larger instances, i.e. 23 channels to connect, could be solved within the 10 minutes deadline.
Table 12. Hit Rate comparison of the exact methods and average and maximum number of Pareto solutions found in the fronts for the 100 switches payload.

<table>
<thead>
<tr>
<th>Channels</th>
<th>Hitrate</th>
<th>Avg. Pareto</th>
<th>max Pareto</th>
</tr>
</thead>
<tbody>
<tr>
<td>epsilon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>28.44%</td>
<td>1.046 ± 0.211</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>0.55%</td>
<td>1 ± 0</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>23</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>adaptive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>32.11%</td>
<td>1.059 ± 0.246</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>2.88%</td>
<td>1 ± 0</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>0.22%</td>
<td>1 ± 0</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

The average number of Pareto solutions found in the front is low, with an average of 1.046 for the $\epsilon$-constraint method and 1.059 for the adaptive $\epsilon$-constraint method for 8 channels to connect. The adaptive $\epsilon$-constraint method solves the instances with the higher number of Pareto solutions. For example, for the cases of 8 channels, the maximum number of Pareto points found by the $\epsilon$-constraint method is 2 whereas with the adaptive $\epsilon$-constraint it is 3. An illustration of this front with 3 optimal solutions is shown in Figure 11.

In Table 13 the computational time required by these methods for the 100 switches payload is displayed. It can be observed that the adaptive $\epsilon$-constraint method performed faster in all cases even if it has higher hit rate. For example, when 8 channels are connected, the average required time for the $\epsilon$-constraint method is 110.43 seconds whereas for the adaptive $\epsilon$-constraint method the corresponding time is 104.14 seconds. When 13 channels are connected, the $\epsilon$-constraint method required 154.24 seconds in average and the computational time for the adaptive $\epsilon$-constraint method is 118.80 seconds.

The computational time required by both methods on the commonly solved instances, is displayed in Table 14. The adaptive $\epsilon$-constraint method performs faster in all cases. The highest difference of 122.05 seconds occurs for the cases of 13 channels to connect.

Table 13. Time(sec) comparison of the exact methods for the 100 switches payload.

<table>
<thead>
<tr>
<th>Channels</th>
<th>Time</th>
<th>min Time</th>
<th>max Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>epsilon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>110.43 ± 162.009</td>
<td>1.110</td>
<td>597.790</td>
</tr>
<tr>
<td>13</td>
<td>154.24 ± 66.287</td>
<td>76.3</td>
<td>240.640</td>
</tr>
<tr>
<td>18</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>adaptive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>104.14 ± 161.528</td>
<td>0.570</td>
<td>597.50</td>
</tr>
<tr>
<td>13</td>
<td>118.80 ± 162.821</td>
<td>1.910</td>
<td>552.52</td>
</tr>
<tr>
<td>18</td>
<td>105.13 ± 138.904</td>
<td>6.91</td>
<td>203.35</td>
</tr>
</tbody>
</table>
6. Conclusion and Perspectives

In this work the problem of optimal communication satellite payload configuration was tackled. Focusing on the initial configuration problem case, an ILP optimisation model for minimising the length of the longest channel path has been proposed. The model is an extension of a previous mathematical model developed for minimising the number of switch changes. Experimental results in realistic payloads, with 50 and 100 switches, demonstrated the validity and the efficiency of the method as well as the limitations in terms of required computational time.

The problem of minimising the length of the longest channel path (LPL) was shown to be a more difficult problem to solve exactly. For instance, on a payload size with 50 switches and 23 amplifiers, considering the planning phase of the initial configuration problem, only 56.66% of the tackled instances were solved when connecting the sets of 18 and 23 channels. On the contrary, minimising the number of switch changes is a significantly easier objective to solve. All the instances of the same size, have been efficiently solved. The computational time when minimising LPL is in average between 1500 and 12000 times higher compared to the required computational time when the number of switch changes is minimised.

A novel multi-objective variant of the initial configuration problem was additionally introduced, where both the longest path length and the number of switch changes have to be minimised. Targeting to the set of Pareto optimal solutions when optimising simultaneously those two objectives, a comparison of two well-known exact multi-objective algorithms was performed, namely the $\epsilon$-constraint method and the adaptive $\epsilon$-constraint method. Exact multi-objective methods are applied for the first time to the considered problem.

The experimental results demonstrated that the adaptive $\epsilon$-constraint method performed better compared to the $\epsilon$-constraint method as it managed to solve more instances within the termination condition. In addition, the adaptive method performed faster on the commonly solved instances. Besides, the instances with the higher number of Pareto optimal solutions on the fronts were solved with the use of the adaptive $\epsilon$-constraint method.

As future work, we plan to further investigate the influence of the selected set of channels to connect on the computational time, as a high standard deviation was observed in all the experiments. Besides, the application of hybrid approaches that will integrate the proposed ILP based exact method with efficient metaheuristics will be addressed for the time critical instances that cannot be solved in an exact manner. Finally, investigating robust solutions for the switch matrix configuration and reconfiguration problem is another important aspect. A robust solution can be defined in this context as a solution which, given any reconfiguration or restoration problem that may arise in the future, can be reconfigured with the minimum operational cost.

Acknowledgements

A. Stathakis acknowledges the support of the National Research Fund of Luxembourg (FNR), with the AFR contract no. 1346481. Experiments presented in this paper were carried out using the HPC facility of the University of Luxembourg.
References


