COMBINATORIAL OPTIMIZATION IN COMMUNICATION NETWORKS
Maggie Cheng, Yingshu Li and Ding-Zhu Du (Eds.) pp. 1 - 41

Quasi-Optimal Resource Allocation in Multi-Spot MFTDMA Satellite Networks

Sara Alouf
INRIA, the National Institute for Research in Computer Science and Control
06902 Sophia Antipolis, France
E-mail: Sara.Alouf@sophia.inria.fr

Eitan Altman
INRIA, the National Institute for Research in Computer Science and Control
06902 Sophia Antipolis, France
E-mail: Eitan.Altman@sophia.inria.fr

Jérôme Galtier
INRIA, the National Institute for Research in Computer Science and Control
06902 Sophia Antipolis, France
E-mail: Jerome.Galtier@sophia.inria.fr
France Telecom Research and Development,
06921 Sophia Antipolis, France
E-mail: Jerome.Galtier@rd.francetelecom.com

Jean-François Lalande
INRIA, the National Institute for Research in Computer Science and Control
06902, Sophia Antipolis, France
E-mail: Jean-Francois.Lalande@sophia.inria.fr

Corinne Touati
Institute of Information Sciences and Electronics
University of Tsukuba, Japan
E-mail: corinne@osdp.is.tsukuba.ac.jp

Contents

Abstract

This chapter presents an algorithm for resource allocation in satellite networks. It deals with planning a time/frequency plan for a set of terminals with a known geometric configuration under interference constraints. Our objective is to maximize the system throughput while guaranteeing that the different types of demands are satisfied, each type using a different amount of bandwidth. The proposed algorithm relies on two main techniques. The first generates admissible configurations for the interference constraints, whereas the second uses linear and integer programming with column generation. The obtained solution estimates a possible allocation plan with optimality guarantees, and highlights the frequency interferences which degrade the construction of good solutions.

1 Introduction

We consider a multi-spot geostationary satellite system for which a manager assigns satellite uplink MFTDMA (Multi-Frequency Time-Division Multiple Access) slots to service providers (operators). The service providers themselves operate a park of terminals distributed on the satellite area of cover. Concerning the radio channel, the satellite divides the time and frequency spectrum into time slots. Geographically, the terminals are distributed on zones, themselves being included in *spots*. A spot is an area covered by a satellite beam, as illustrated in Figure 1.

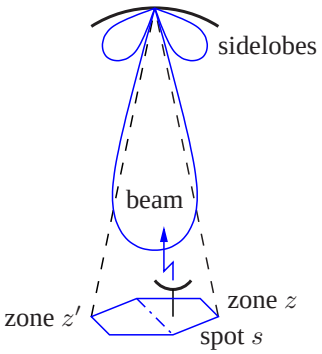


Figure 1: Illustration of a spot and an antenna beam.

Radio interferences impose constraints on the slots that can simultaneously be assigned in different spots that have the same frequency. A slot cannot be assigned simultaneously to more than one zone in a spot. Spots are given colors (bands of frequencies) and spots of different colors do not interfere, but spots of the same

color do, and a slot can be assigned to an operator in a given zone only if the interference it experiences with the other active zones is below a given threshold. Slot assignment is static but can be changed once per hour (due to changes in demands, on the one hand, and to changes in atmospheric conditions, on the other hand). Every hour, the demand of the service providers is re-evaluated and a new allocation could be generated. Due to real-time constraints, solutions are needed within a few minutes.

Our goal is to maximize the throughput of the system. A simplified version of our problem can be modeled as a so called "k-colorable induced subgraph" problem where one considers a graph $G = (V, E)$ consisting of finitely many nodes and directional links [32]. A valid coloring of the graph consists of coloring nodes such that no nodes with a common link have the same color. We look for a subset of nodes $V' \subset V$ and edges $E' \subset E$ such that the induced subgraph is k-colorable, i.e., there is a coloring for the subgraph (V', E') of cardinality at most k. The problem consists in finding such a graph with the maximal number of nodes. This problem turns out to be NP complete [13, GT20]. Actually, our problem is even more complex since arcs have weights (arc weights are related to the amount of interference) and exclusion constraints are more complex. This is in contrast to slot allocation in Satellite Switched TDMA systems that have polynomial solutions as they correspond to coloring problems with a simple bi-partite graph topology [6]. In this chapter, instead of using coloring approaches, we propose to solve the problem differently, using a linear and integer programming approach with column generation.

This work[1] is clearly motivated by the cost of the design of satellite antennas: the cost of an antenna is a strong function of its size, roughly speaking, proportional to the diameter cubed [2]. Larger antennas generate small interferences and have better gain, but increase tremendously the cost of the satellite. One of the goals of this approach is to tune precisely the assignment problem given its profile in terms of interference and gain. We will see that in return, our program can derive which interferences are responsible for (sometimes substantial) loss of capacity for a given demand.

In our experiments to evaluate the proposed approach, we will be using two series of data corresponding to 8 and 32 spots per color respectively. We assumed that there are three zones per spot, and four types of carriers[2]. Our work is focused on one of the colors of the bandwidth (recall that spots of different colors do not interfere with each other), so that the complete processing phase should use the

[1]This work is part of research convention A 56918 between INRIA and ALCATEL SPACE INDUSTRIES (contract number 1 02 E 0306 00 41620 01 2).

[2]Carriers have different bandwidths thus providing different slot durations. The use of a specific carrier by a given terminal is determined by the terminal's transmission capability.

same program for each color (if necessary in a parallel way). In our experiments, the total number of (time) slots that can be assigned is set to 3456.

We propose in this chapter a linear and integer programming approach that allows to solve the problem almost optimally. For the 8-spot case, the problem is solved in a minute or so, with a guarantee of consuming at most 1% more bandwidth than the absolute optimum. The dual/primal approach is exploited in a master/slave fashion, where the master program is a heuristic that finds non-interfering zones that are directly translated into valid columns for the primal problem handled by the slave program. This approach can output the interfering configurations that limit the optimization up to a certain threshold. This information is extremely important for the design of antennas since it explains the characteristics of the antennas that lead to performance limitation. In other words, our approach identifies the interfering configurations that are crucial to the optimization, and this information has to be taken into account when designing antennas. Designers have to make sure that the antennas do not impair such configurations. Last, we show that, in the 32-spot case, our program can output solutions that in practice have good performance.

The structure of the chapter is as follows. Related references are briefly discussed in Section 2. The system model and its constraints are presented in Section 3. The resolution of the time slot allocation problem throughout a simple example is detailed in Section 4, whereas the general solution is detailed in Section 5. Numerical results are presented in Section 6, followed by a concluding section.

2 Related Work

In the vast majority of the cases, the related references that have appeared in the past dealt with simpler models which, in some cases, have been solvable using polynomial algorithms. We wish to mention, however, that problems with similar nature but with simpler structure have also been treated in the context of scheduling in ad-hoc networks, see e.g. [14] and the references therein.

In this section, we will focus on algorithmic approaches for solving the slot allocation (or "burst scheduling") problem, that have appeared in the literature. Empirical approaches for burst scheduling has been proposed in [21, 22, 29]. Concerning other aspects of TDMA satellites, we convey the reader to the paper [1] which surveys issues such as architecture, synchronization and some physical layer considerations and discusses papers presenting probabilistic performance evaluation techniques related to TDMA systems.

A few words on the terminology: we use the standard term SS/TDMA for

Satellite-Switched Time-Division Multiple Access. We also frequently find in the literature the expression "burst scheduling". The term "burst" does not refer to a burst in the input traffic (the data) but rather to the fact that traffic is not transmitted continuously but in bursts.

The input to the slot assignment problem in TDMA systems is often a traffic matrix whose ijth element - denoted as $\delta_{i,j}$ - describes the amount of traffic to be shipped from zone i to zone j, or equivalently the time to transfer it at a fixed channel rate.

In [19], the author considers n transponders to switch an $n \times n$ demand matrix. Each terminal can either send or receive with one transponder at a time. The author shows that the minimal time to transfer a complete matrix corresponds to

$$\max\left(\max_{1\leq i\leq n}\sum_{1\leq j\leq n}\delta_{i,j}, \max_{1\leq j\leq n}\sum_{1\leq i\leq n}\delta_{i,j}\right).$$

The author provides an algorithm that achieves this bound, and in order to do so, the frame should be divided into a number of switching modes, that correspond to several assignments of the transponders. This number of switching modes is minimized under the condition that the time to transfer is minimal. He shows that at most $n^2 - 2n + 2$ different switching modes are necessary.

From the algorithmic aspect, this reference can be explained in simple terms. First note that given the maximum row and column sums of the matrix, it can be **greedily completed** into a matrix with constant row and column sums, simply by marking all the *deficient* rows and columns (i.e. those which do not reach the maximum) and increasing at a step an element sharing a deficient row and a deficient column. Then a **maximum bipartite matching** (what they refer to as System of Distinct Representatives in the paper; see also the improvement of [28]) will find a switching mode. It results that less than $n^2 + 2n + 2$ steps are necessary since at least one element of the matrix goes to zero at a time.

In [3], the authors extend the results of Inukai [19] in the case where k **transponders** are present and the demand matrix is $n \times m$. In this case, the minimal time that one could expect to transfer the matrix is equal to:

$$\max\left(\max_{1\leq i\leq n}\sum_{1\leq j\leq m}\delta_{i,j}, \max_{1\leq j\leq m}\sum_{1\leq i\leq n}\delta_{i,j}, \sum_{1\leq i\leq n}\sum_{1\leq j\leq m}\delta_{i,j}/k\right).$$

The authors give an algorithm that achieves this bound, and manage to bound the number of switching mode used to $n^2 - n + 1$ if $n = m = k$, and $nm + k + 1$ otherwise. However experimental results suggest that this number is substantially

lower to that bound in practice. The algorithmic ingredients are essentially the same as before.

In [16], the authors consider again n transponders to switch an $n \times n$ demand matrix, but this time, **interferences** are taken into consideration. The interferences are modeled as constraints both on the uplink and downlink of the system, with respective undirected graphs G_U and G_D. The graph G_U associates a vertex with each terminal, and has an edge between terminals u and v if and only if terminals u and v cannot communicate at the same time. G_D is defined similarly. The authors demonstrate the NP-hardness of this problem and propose a solution in the context of polarization - which is the case when two independent channels are used to transmit the traffic. They accordingly propose a two-step algorithm: (i) divide the matrix into two parts using supposedly planarity properties, minimizing the interference using a **MAX-CUT algorithm** (the algorithm they use is optimal only in the planar case; note, however, that a 0,87-approximation algorithm in polynomial time of this problem in the general case has been since discovered - see [15]), and (ii) in each of the obtained two parts, minimize the number of necessary time slots to transmit without interference - developing various **coloring heuristics** (e.g. brute force, greedy algorithms) that will help to incrementally construct a suboptimal schedule, selecting a "good" interference-free matrix at each step. Note that this approach fails to give a result on the global optimality of the problem. In fact, only the second part of the algorithm really addresses the problem. Indeed, if a minimum number of time slots can be found in the *general* case, polarization (or other types of separate band assignment, such as frequency division) can be efficiently exploited by splitting the final schedule in two parts (or more, in the case of frequencies) and assigning a part to each polarity (or frequency). Note that our method can be easily adapted to this case and can give general optimality guarantees.

In [23] the problem of finding a solution when n transponders are present and an $n \times n$ demand matrix is given is studied under the particular restriction that only a restricted set of switching matrices can be used. In such a case, of course, the authors notice that linear programming can minimize the total transfer time, which means that the solution of the problem can be found efficiently. Rather, they consider some even more specific conditions on the switching matrices and give in that particular case even faster algorithmic solutions.

The type of problem studied in the previous references and the results therein obtained were later extended in [4–6, 26, 30, 31]. It is important to mention that no interference problem is considered in these papers.

In [5], the authors consider the problem of adding some second-priority-traffic to some existing schedule. It is argued to be important when some streaming communications (voice, video) are taken to compute the switching modes, to which

some additional data may be assigned. The authors claim a NP-completeness theorem, and give some heuristics to approach the problem. Note, however, that an alternative solution would then be to recompute the switching matrices from scratch and see whether it increases the total communication time.

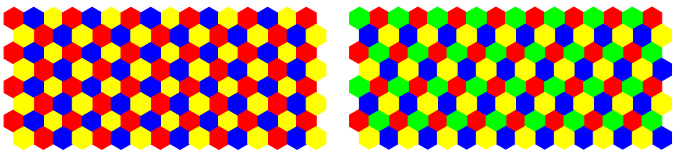
Additionally, note that a considerable amount of work as been done on this topic, see for instance [7, 12, 17, 20, 25, 27, 33].

3 The Model

In this section, we present the model considered in the rest of the chapter. We introduce the spots configurations in Section 3.1 and the interference model in Section 3.2. Section 3.3 presents some practical details concerning the computation of the carrier-to-interference ratio. Last, informations related to the terminals (capacity of transmission, carrier used, demand) are provided in Section 3.4.

3.1 Spatial Reuse

The total satellite bandwidth is subdivided in several equally-large bandwidths. Each one of these will be assigned a *color*. Every spot is assigned a unique fixed color, implying that all terminals of a spot can transmit within the bandwidth corresponding to the spot's color. Every color may be assigned to several spots. This is the concept of spatial reuse (see for instance [10]). Observe that terminals in different spots of the same color will interfere with each other when using the same frequency band within the spots total bandwidth. Multiple terminals will not be allowed to transmit if the global interference generated is too high, as it will impair the correct reception of the data by the satellite. Color assignment is given as an entry of our problem. Examples of color assignment can be seen in Figure 2(a), resp. Figure 2(b), when 3 colors, resp. 4 colors, are used.



(a) 3 colors used (b) 4 colors used

Figure 2: Spatial distribution of spots and optimal reuse of colors.

Since colors do not overlap in bandwidth, they are completely independent from each other. Hence, resource allocation can be done for each color separately.

The original problem has simply to be split in the number of colors used, and each resulting problem can be solved independently from the others. Hereafter, we will consider only the problem of resource allocation within the same color. Without loss of generality, we will consider a spatial reuse of 4 colors. Let N denote the numbers of spots having the same color, and B denote the color bandwidth. We are particularly interested in the case where $N \leq 32$. Figure 3 depicts the spots configuration within one color when 4 colors are used.

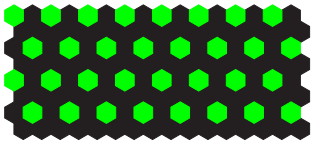


Figure 3: Spatial distribution of spots using the same color (4 colors case).

Different spots of the same color are allowed to transmit only if the overall level of interferences is acceptable and does not impair the correct reception of the transmitted signals at the satellite. In the following section, we will introduce an allocation criterion as a mean to check if it is safe to activate one spot or another. This allocation criterion will condition any frequency reuse between spots of the same color.

3.2 Interference Level

To take into account the real conditions of the radio propagation, it is necessary to account for the position of the terminals within a given spot. The spot is usually large enough to have different channel conditions in different geographical regions. We will therefore divide a spot in a number of *zones* (typically 2 or 3), assuming that each zone exhibits the same propagation conditions in all its area. The radio propagation experienced by a terminal is thus completely characterized by the zone where the terminal is.

If a terminal is transmitting at time t, using carrier f, we will say that its zone/spot is active in (t, f). Whenever a zone is active, its transmission will generate interferences over all other spots using the same carrier at the same time. Note that this interference will be the same over any zone of a given active spot. The importance of the interference is directly affected by the size of the antennas' sidelobes. Figure 4 illustrates well how a transmission can interfere over others. It is clear from Figure 4 that the interference, generated over spot s' by a terminal in spot s, located in a zone other than zone z, will be different. It should also be clear

that the interference generated by an active zone is the same over all zones within the same active spot.

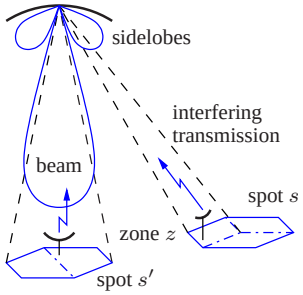


Figure 4: Interference generated by a terminal in zone z over a terminal in spot s'.

Let $G(z)$ denote the minimal antenna gain corresponding to zone z. Let $I(s,z)$ denote the maximal interference generated over spot s by a transmission in zone z. It is the maximal antenna gain in the sidelobes corresponding to zone z, when the main beam is directed to spot s. If zone z belongs to spot s then $I(s,z)=0$. The received signal at the satellite is useful only if its power amplitude is large enough compared to the power of the interfering signals. In other words, the carrier to interference ratio should be beyond a certain threshold σ, otherwise the satellite cannot properly handle the received transmission. Hence, a zone z could be active in (t,f) if and only if the following criteria is satisfied:

$$\frac{C}{I} = \frac{G(z)}{\sum_{z' \text{ active in } (t,f)} I(s(z),z')} \geq \sigma, \tag{1}$$

where $s(z)$ denotes the spot in which zone z is located.

Note that the interferences considered in our model are much more realistic than the ones considered in [16]. Indeed, only two terminals could interfere with each other and in this case, only one of these terminals will be allowed to transmit at a given time. Our model is more complex as multiple interfering communications are possible given that the interference threshold is observed.

3.3 Interference Model in Numerical Results

The power of the interfering signal used in (1) depends on the size of the antenna. Small sidelobes lead to weak interferences. Unfortunately, we do not have data on the power distribution of the interfering signal over all geographical areas, we will therefore assume the following: neighboring spots are the ones generating the

highest interference over each other; remote spots still interfere one on each other but not as significantly. In the results of Section 6, the values in decibels of the gain $G(z)$ (resp. interference $I(s,z)$) are taken randomly in the interval $[40,41]$ (resp. $[11,15]$) decibels. Thus, we use these different quantities:

$$I_1(z) = \sum_{z' \text{ neighbor, active in } (t,f)} I(s(z),z') \tag{2}$$

$$I_2(z) = \sum_{z' \text{ active in } (t,f)} I(s(z),z') \tag{3}$$

$$I(z) = I_1(z) + (1-\gamma)\,(I_2(z) - I_1(z))$$

where γ is a given weight. Equation (1) is replaced with

$$\frac{C}{I} = \frac{G(z)}{I(z)} \geq \sigma. \tag{4}$$

The term $I_2(z) - I_1(z)$ designates the interference generated by all active zones in non-neighboring spots. Therefore, the interferences generated by remote spots are reduced by a factor $1-\gamma$. Observe that taking $\gamma = 0$ is equivalent to considering that all interferences are equally important (Eqs. (1) and (4) will be exactly the same), while having $\gamma = 1$ nullifies the effect of transmissions in non-neighboring spots over the zone at hand.

3.4 Types of Terminals and Demand

Terminals have different capabilities of transmission. A given type of terminals will use a unique frequency band. Hereafter, we will classify terminals according to their capability of transmission, and use the notation $t_k, k = 1,\ldots,\tau$ to refer to a given type of terminals (τ referring to the number of different types of terminals), the ascending order corresponding to the ascending slot duration. Every type of terminals t_k will be assigned a unique bandwidth, denoted by t_k^b. In our problem, the ratio of the bandwidths of any two different types is either an integer or the inverse of an integer and is called the multiplicity. Also, the duration of a slot will be dependent on the terminal's type. The idea is to have the same amount of data transmitted in a slot time whichever the type of terminal at hand: for any type t_k, the product of its bandwidth, t_k^b, and its slot duration, denoted by t_k^t, is a constant: $t_k^b t_k^t = \Delta$. Table 1 reports the values used to evaluate our algorithm. Observe that type t_1 is the smallest in time and largest in bandwidth, whereas type t_4 has the longest time slot duration and the narrowest bandwidth. From the table we can write $t_1^b = 2t_2^b = 8t_3^b = 32t_4^b$, or equivalently, $t_4^t = 4t_3^t = 16t_2^t = 32t_1^t$.

Table 1: Test values of terminals types.

Type	Maximum number of time slots per frame	Maximum number of carriers per spot bandwidth
t_1	192	18
t_2	96	36
t_3	24	144
t_4	6	576

The individual demands of all terminals in a zone are aggregated according to the type of terminals, and hence, the bandwidth used by every type. Let $d(z, t_k)$ denote the demand in time slots in zone z expressed in time slots of type t_k, for any zone z and any type t_k.

4 Solving a Simple Example

In this section, we will consider the simple case where there is only one type of terminals, i.e. all terminals use the same amount of bandwidth to transmit their data. For every carrier, the channel can be accessed simultaneously by multiple terminals/zones according to the Time-Division Multiple Access (TDMA) technique. Solving the resource allocation problem translates then into the following question: which zones are allowed to transmit in a given time slot and using a given carrier?

Consider the example illustrated in Figure 5. There are 3 spots transmitting in the same color, each spot having 2 zones. When active, every zone generates

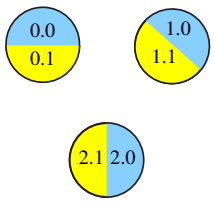


Figure 5: Example with 3 spots.

a certain level of interference over all other spots (gain and interferences can be found in Table 2, values are not in dB). Every spot can have either one of its zones active, or be inactive (recall that only one zone in a given spot can be active at a given time). Hence, there are $3^3 = 27$ possibilities in our simple example.

Table 2: Gain and interferences of the 6 zones in the example.

Zone	Gain	$I(\text{Spot } 0, \cdot)$	$I(\text{Spot } 1, \cdot)$	$I(\text{Spot } 2, \cdot)$
0.0	4	-	5	3
0.1	6	-	5	7
1.0	3	4	-	2
1.1	8	7	-	10
2.0	5	3	7	-
2.1	5	7	3	-

Considering any zone from the example, this zone can be active (*on*) only if its carrier-to-interference ratio is above a certain value. This ratio will naturally depend on whether the other spots are active or not (*on* or *off*). For every zone considered, there are 9 possible situations, as reported in Table 3. Let $\sigma = 0.3$. All of the situations where only two spots are active are valid, since the carrier-to-interference ratio is higher than 0.3 for all zones in every such situation (refer to last column and last row for every zone). Among all $2^3 = 8$ situations where 3 spots are active, only 3 are valid. For instance, if zones 0.0, 1.0 and 2.0 are active, it appears that the carrier-to-interference ratio is above $\sigma = 0.3$ for zones 0.0 and 2.0, but not for zone 1.0. This combination is therefore not valid and should not be used in the allocation procedure. The only 3 combinations with 3 active spots that are valid are illustrated in Figure 6. The reader can check that, for each combination, all zones satisfy the allocation criterion.

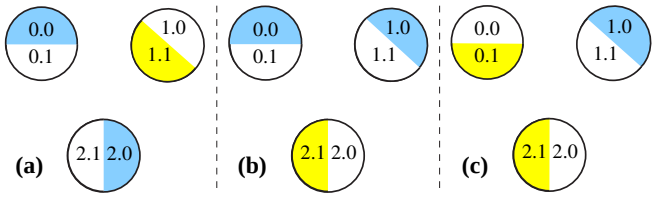


Figure 6: Valid 3-spot combinations for a threshold $\sigma = 0.30$.

Observe that the 3-spot combinations transmit more data, at the same time, than the 2-spot combinations which are less efficient.

Table 3: Values of the carrier-to-interference ratio for all zones in all situations.

C/I for Zone 0.0	Zone 1.0 *on*	Zone 1.1 *on*	Spot 1 *off*
Zone 2.0 *on*	0.57	0.40	1.33
Zone 2.1 *on*	0.36	0.29	0.57
Spot 2 *off*	1.00	0.57	-
C/I for Zone 0.1	Zone 1.0 *on*	Zone 1.1 *on*	Spot 1 *off*
Zone 2.0 *on*	0.86	0.60	2.00
Zone 2.1 *on*	0.55	0.43	0.86
Spot 2 *off*	1.50	0.86	-
C/I for Zone 1.0	Zone 0.0 *on*	Zone 0.1 *on*	Spot 0 *off*
Zone 2.0 *on*	0.25	0.25	0.43
Zone 2.1 *on*	0.38	0.38	1.00
Spot 2 *off*	0.60	0.60	-
C/I for Zone 1.1	Zone 0.0 *on*	Zone 0.1 *on*	Spot 0 *off*
Zone 2.0 *on*	0.67	0.67	1.14
Zone 2.1 *on*	1.00	1.00	2.67
Spot 2 *off*	1.60	1.60	-
C/I for Zone 2.0	Zone 0.0 *on*	Zone 0.1 *on*	Spot 0 *off*
Zone 1.0 *on*	1.00	0.56	2.50
Zone 1.1 *on*	0.38	0.29	0.50
Spot 1 *off*	1.67	0.71	-
C/I for Zone 2.1	Zone 0.0 *on*	Zone 0.1 *on*	Spot 0 *off*
Zone 1.0 *on*	1.00	0.56	2.50
Zone 1.1 *on*	0.38	0.29	0.50
Spot 1 *off*	1.67	0.71	-

4.1 Case of a Simple Demand

Assuming that there is a demand of 100 time slots per *zone*, it is clear that the minimum number of time slots necessary to fulfill the demand is 200, since only one zone per spot can be active at any time. For the first 100 time slots, the combination in Figure 6(a) can be used to satisfy the demand of zones 0.0, 1.1 and 2.0, and for the second 100 time slots, the combination in Figure 6(c) can be used to satisfy the demand of zones 0.1, 1.0 and 2.1, which solves the problem.

4.2 Case of a More Complex Demand

Consider here a demand slightly more complex than in the previous case, as can be seen in Table 4. The demand per *spot* is 200 time slots, as in the previous

Table 4: Demand (in time slots) of the different zones.

Zone	0.0	0.1	1.0	1.1	2.0	2.1
Demand	50	150	50	150	150	50

case, but more than 200 time slots are needed to satisfy all zones, because the 3 combinations of Figure 6 cannot be used as efficiently as before. It is clear that the combination in Figure 6(a) can still be used for 50 time slots to satisfy the demand of zone 0.0, and zones 1.1 and 2.0 are left with 100 time slots demand to satisfy. Also, the combination in Figure 6(c) can be used for 50 time slots to satisfy the demand of zones 1.0 and 2.1, and zone 0.1 is left with an unsatisfied demand of 100 time slots. To complete the allocation problem, we can use combinations with only two active zones, allocating 50 time slots to each one of the following combinations: (i) zones 0.1 and 1.1; (ii) zones 0.1 and 2.0; and (iii) zones 1.1 and 2.0. Observe that the allocation procedure consists mainly in allocating 250 time slots to combinations of zones, provided that these combinations are valid.

Looking at Figure 6, we see that combinations (b) and (c) differ only on spot 0. They both include zones 1.0 and 2.1, but while the first combination includes zone 0.0, the second includes zone 0.1. It is therefore possible to merge these combinations into one, composed of *any* zone of spot 0 and zones 1.0 and 2.1. Hereafter, we will use the term "family" to refer to such combination of zones/spots. Observe that it is possible to use a given family when allocating slots, even though not all zones within this family need to be active. This observation will add flexibility to the solution. Using the same amount of time slots as before, that is 250, the allocation to satisfy the demand of Table 4 could now be satisfied as expressed in Table 5. In this solution, zone 0.0 will be assigned 50 extra time slots.

Table 5: A more efficient solution to the example.

Number of time slots	Family to use	Active zones
100	Zones 0.0, 1.1, 2.0	Zones 0.0, 1.1, 2.0
50	Spot 0, Zones 1.0, 2.1	Zones 0.1, 1.0, 2.1
50	Spots 0, 1	Zones 0.1, 1.1
50	Spots 0, 2	Zones 0.1, 2.0

5 Solving the General Case

As seen in the previous section, to solve the allocation problem in the simple case where there is only one type of terminals, we have first computed the carrier-to-interference ratio for all zones which let us identify the valid combinations, or families, of zones that are allowed to transmit simultaneously. Second, we have allocated a certain number of time slots for some families in order to satisfy the demand of all zones. To solve the allocation problem in general (arbitrary number of zones/spots, arbitrary demand and multiple types of terminals) we will have to (i) generate families of spots/zones that are valid (see Section 5.1), (ii) identify the number of time slots of each type to allocate to which families in order to satisfy the demand (see Section 5.3), and (iii) allocate the required number of time slots by placing the carriers in the radio channel and the time slots in the corresponding time frames (see Section 5.2). The details of step (ii) are presented through Sections 5.4 - 5.8. Section 5.9 presents a wrap-up of our approach.

5.1 Solving Interference Problems

Our approach is mainly based on the following key observation: for any time t and any frequency f, there exists at least one family of zones that can be simultaneously active. Let Z denote one such family, we therefore have:

$$\frac{G(z)}{\sum_{z' \in Z} I(s(z), z')} \geq \sigma \quad \forall z \in Z. \tag{5}$$

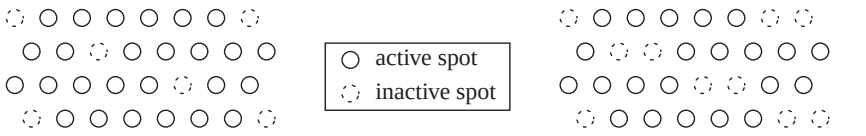
Naturally, there could be in family Z no more than one (active) zone per spot. This concept of concurrent transmissions is somehow similar to graph coloring [18], where families of independent edges are used to solve the problem. Here, we will use families of zones allowed to transmit at the same time (and at the same frequency).

In practice, there is a very large number of families checking this criterion. It is possible to have families that differ only by one spot, according to which zone in the spot is active (see the example in Section 4). As already said, such families can be merged in a single family, keeping in mind that, for that particular spot, several zones could be allowed to be active. This merging will add flexibility to the use of the resulting family. To solve the interference problem, we will generate a certain number of families, that will be used later on in the time slot allocation procedure. It is crucial to generate in the first place the most efficient families, or in other words, the families having the highest possible number of zones that can be active in (t, f), while presenting the highest flexibility.

5.1.1 Generating Generic Families

The threshold of interference σ is given as an input. If σ is very weak (for instance 10dB, which is not very realistic), all spots can be active in (t, f). As σ increases, less spots can be active simultaneously using the same frequency. The difficulty here is to have the maximum number of active spots/zones for a given σ.

Recall the allocation criterion given in (4). It makes the distinction whether the interfering terminal is in a neighboring spot or not. Terminals in the vicinity are considered to interfere more than remote terminals. It then comes out that inactive spot should be geographically distributed for increased efficiency. We consider situations where only a restricted set of spots are inactive. We call a configuration 6/7 (resp. 5/7, 4/7) when at most 6 (resp. 5, 4) spots over a vicinity of 7 are active. We illustrate in Figure 7 such possible configurations. We translate the illustrated patterns (that have maximality properties on the infinite grid) to obtain a limited but efficient series of families.



(a) a possible configuration 6/7 (b) a possible configuration 5/7

Figure 7: Example of configurations 6/7 and 5/7.

It is obvious that there are 7 distinct configurations 6/7 as there are 7 possible positions for the inactive spots in a line. Also, there are $7 \times 3 = 21$ distinct configurations 5/7, since every configuration 6/7 generates 3 possible configurations 5/7 according to whether there are 0, 1 or 2 active spots between two inactive spots in any horizontal line. In a similar way, there are in total $7 \times 5 = 35$ configurations 4/7, as every configuration 6/7 generates 5 possible configurations 4/7.

5.1.2 Status of a Spot

In the previous section, we have introduced efficient spatial configurations of active/inactive spots that distribute homogeneously the inactive spots. We believe that these configurations are more efficient than others as they will allow a larger number of spots to be active given the same threshold σ. Each one of these configurations yields several families of active zones. Indeed, spots are usually divided into few zones (typically 2 or 3), and there are several possibilities for having a spot active. As (i) the power gain depends on the geographical zone within a spot, and

(ii) the interferences generated over the spot depend on which zones have transmitted the interfering signals, it is quite possible that one zone in a spot does not check the allocation criterion (4) while another zone in the very same spot does. Therefore, every spot will be assigned a *status* describing which zones can potentially be active. If a spot s has $nbZones(s)$ zones, then its status takes value in the interval $[0, 2^{nbZones(s)} - 1]$. For instance, the status of a 3-zone spot could take on one of the following values (a 2-zone spot could take on one of the first 4 statuses in the list):

0: the spot is inactive;

1: zone 0 checks (4), hence it could transmit;

2: zone 1 checks (4), hence it could transmit;

3: zones 0 and 1 check (4); hence either one could transmit;

4: zone 2 checks (4), hence it could transmit;

5: zones 0 and 2 check (4); hence either one could transmit;

6: zones 1 and 2 check (4); hence either one could transmit;

7: all zones check (4); hence either one could transmit;

Instead of generating families of zones, we will generate families of spots and assign to each spot the convenient status given the allocation threshold σ. Allocating time slots to a 3-zone spot with status 7 would actually be done by allocating the time slots to *either* one of its 3 zones, which increases freedom and improves the efficiency of our approach.

5.1.3 Simplifying the Computation of the Allocation Criterion

At the beginning of Section 5.1, we have defined a family of zones Z satisfying (5). In this section, we will derive a similar equation for families of spots. Instead of checking the allocation criterion (4) for every zone, we will have to check it for every spot. To be able to check if a spot could be active and decide which status it could have, we assign to every spot a gain and an interference over other spots.

The gain of a spot is defined as the minimum value of the gains of its zones which are active (information available from the status of the spot). Let $G(s)$ denote the spot gain, we can write

$$G(s) = \min_{z \text{ in } s, \text{ active}} G(z).$$

The interference generated over spot s by spot s' is defined as the maximum value of the interferences generated by all zones of spot s' that could potentially be active. It will be denoted as $I(s, s')$. We have

$$I(s, s') = \max_{z' \text{ in } s', \text{ active}} I(s, z').$$

Recall the sums $I_1(z)$ and $I_2(z)$ introduced in (2)-(3). They represent the overall interference generated by active zones in neighboring spots and in all spots, respectively. Let $I_1(s)$ and $I_2(s)$ be their equivalent at the spot level:

$$I_1(s) = \sum_{s' \text{ neighbor, active}} I(s, s'), \qquad I_2(s) = \sum_{s' \text{ active}} I(s, s')$$

Similarly to what we did at the zone level, the total level of interference generated over a spot s will be computed as:

$$I(s) = \gamma\, I_1(s) + (1 - \gamma)\, I_2(s)$$

Thus, a spot is said to be *valid* if it checks the following criterion

$$\frac{G(s)}{I(s)} \geq \sigma. \tag{6}$$

The advantage of using (6) rather than using (4) will be clear from the following example. Consider a spot whose status is 7. This means that it has 3 zones that could all be active (of course, not together). To check this hypothesis, one would have to check if each zone satisfies the criterion (4). It is definitely more advantageous to use instead the criterion (6) as the computation time would be greatly reduced. Observe that (6) implies (4). For any active zone z in spot s:

$$\begin{aligned} \frac{G(s)}{I(s)} &= \frac{G(s)}{\gamma\, I_1(s) + (1 - \gamma)\, I_2(s)} \\ &\leq \frac{G(z)}{\gamma\, I_1(z) + (1 - \gamma)\, I_2(z)} = \frac{G(z)}{I(z)}. \end{aligned}$$

Thus, if a spot with a given status is valid, then all of its zones corresponding to its status are valid.

For maximum flexibility, we would like to have all spots in a family have a status equal to $2^{nbZones(s)} - 1$. To that purpose, we will first generate families of spots, all having the highest status, and then test their validity. That can be done by checking the allocation criterion (6) for all spots in a family.

5.1.4 Heuristics for Generating Valid Families

We want to maximize the number of active zones, we start by generating the 7 families 6/7 in which any active spot s has the status $2^{nbZones(s)}-1$ while inactive ones have status 0. We then successively test the validity of these families and separate them in two pools, one for valid families and the other for non-valid families. We do the same with families 5/7, 4/7, etc.

To make a non-valid family become valid, some of its active zones should be deactivated. For instance, if a 3-zone spot having status 7 (any one of its 3 zones could be active) is not valid, then we should test the validity of its family when its status is 3, 5 or 6 (zone 2, zone 1 or zone 0 are deactivated). The following heuristic is used:

1. randomly choose a non-valid family from the pool of non-valid families;
2. as long as the family is not valid, do:
   (a) randomly choose a spot,
   (b) if its status is non-null and the spot is non-valid, deactivate at random one of the active zones; keep a record of the spot identifier;
3. try, for a certain number of times, to reactivate zones which were deactivated in step 2 and test the validity of the resulting family after each try: an amendment is adopted only if the family is valid;
4. compare the valid family obtained in step 3 with those in the pool of valid families. In case of redundancy, increment a counter of redundancies and reject the family; otherwise, add the family to the pool of valid families. Return to step 1 to generate another family.

This algorithm stops either when the desired number of valid families is reached, or when the counter of redundancies has reached a given maximum value. At this point, we have generated valid families of spots. In every spot s of a valid family, $0, \ldots, nbZones(s)$ zones are candidates in the time slot allocation procedure.

5.2 Placing the Carriers in the Radio Channel

The constraints on the radio channel deal with the spot bandwidth B and the time frame duration T. When planning the allocation of a time slot from a given carrier to a given type of terminal, one schematically uses a rectangle of a *fixed* surface equal to Δ in the time-frequency space (recall Section 3.4). See for instance zone 0.1 in Figure 11 in which two different types of terminals are used.

Thus, if the types of terminals are denoted by subscripts from 1 to τ (ordered by decreasing bandwidth), and if x^{t_k} denotes the number of time slots of type t_k used in the spot, we then have:

$$\sum_{k=1}^{\tau} x^{t_k} \leq \frac{BT}{\Delta}. \tag{7}$$

In other words, the maximal surface, in the time-frequency space, that can be allocated to a spot is equal to the product BT, yielding an upper bound equal to BT/Δ on the number of time slots that can be allocated.

The following lemma is used to establish the properties of a filling of time slots:

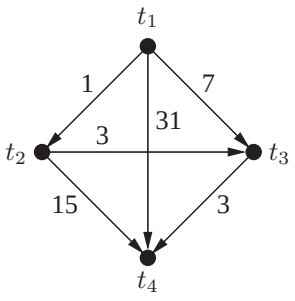
Lemma 5.2.1 *Let $G = (V, E)$ be a directed graph with $V = \{t_1, \ldots, t_\tau\}$ and $E = \{(t_j, t_k) : j < k\}$. Define $w(t_j, t_k) = w_{(j,k)} = \frac{t_j^b}{t_k^b} - 1$. Then any path in G from t_i to t_j has a weight less than $w_{(i,j)}$. In particular, any path in G from t_1 to t_τ has a weight less than $w_{(1,\tau)}$.*

Proof. Note that G is transitive. Thus, if $(t_i, t_j) \in E$ and $(t_j, t_k) \in E$, then $(t_i, t_k) \in E$. Observe that for any numbers x and y such as $x > 1$ and $y > 1$, we have

$$(x-1) + (y-1) = xy - 1 - (x-1)(y-1) < xy - 1. \tag{8}$$

The weight of the path $t_i \to t_j \to t_k$ is equal to $w_{(i,j)} + w_{(j,k)} < w_{(i,k)}$ (take $x = t_i^b/t_j^b$ and $y = t_j^b/t_k^b$ in (8)), which concludes the proof. □

Example 5.2.2 *Figure 8 illustrates the graph $G(V, E)$ corresponding to the data in Section 3.4.*



$V = \{t_1, t_2, t_3, t_4\}$
$E = \{(t_1, t_2), (t_1, t_3), (t_1, t_4),$
$(t_2, t_3), (t_2, t_4), (t_3, t_4)\}$
$w_{(1,2)} = 1$
$w_{(1,3)} = 7$
$w_{(1,4)} = 31$
$w_{(2,3)} = 3$
$w_{(2,4)} = 15$
$w_{(3,4)} = 3$

Figure 8: Graph G according to Table 1.

Thereafter, we show that a path in this graph corresponds to losses due to the geometrical structure of the problem. Any change in type during the placement process will incur a waste in space in the time-frequency space. Changing from type t_i to type t_j ($j > i$) will cause *at most* an unused space equal to $w_{(i,j)}$. To minimize the space that could be lost, the best thing to do is to place the types monotonically. We have opted to fill the time-frequency space from left to right and top to bottom using the ascending order of types. The maximum number of unused time slots with this policy is given by the weight along a path in G that goes from t_1 to t_τ. We know from Lemma 5.2.1 that this maximum is less than $w_{(1,\tau)}$. To be more precise, this maximum (obtained in the worst case) is exactly the sum of the weights along the path followed in graph G to go from t_1 to t_τ.

Result 5.2.3 *It is feasible to place, in the time-frequency space, x^{t_k} time slots of type t_k, for $k = 1, \ldots, \tau$ if*

$$\sum_{k=1}^{\tau} x^{t_k} \leq \frac{BT}{\Delta} - w_{(1,\tau)}. \tag{9}$$

Eq. (9) is therefore a sufficient condition for a placement algorithm.

Proof. We give the sketch of the proof. To prove that Eq. (9) is sufficient to find a placement, we will have to devise a placement policy that will succeed in placing all time slots in the space $B \times T$ without wasting more than the theoretical maximum waste $w_{(1,\tau)}$. We start by subdividing the total space $B \times T$ in rectangles whose "frequency" dimension is the maximum bandwidth of a time slot and whose "time" dimension is the maximum duration of a time slot. Such rectangles have a bandwidth t_1^b and a duration t_τ^t, and will be referred to as "rectangles $(1, \tau)$".

The rectangles $(1, \tau)$ are filled according to the ascending order of types. To evaluate the "waste" within a single rectangle $(1, \tau)$ induced by this filling policy, two cases have to be considered whether a change of type forces the filling of the next rectangle $(1, \tau)$, or not.

Case 1 If the bottom-right corner of a rectangle $(1, \tau)$ is reached, there will be no waste when switching to the following rectangle. The only waste (if any) will be "internal" to the rectangle $(1, \tau)$ at hand. If this rectangle contains types t_i and t_j, it is allowed to "waste" a space equal to $w_{(i,j)}$. Indeed, it subdivides itself in rectangles (i, j) of types increasing from t_i to t_j. In the worst case, we lose the sum of the weights in a path of the graph of Lemma 5.2.1 going from type t_i to type t_j. Rectangles containing only one type of time slots should not cause any waste in space.

Case 2 If a change in type forces the switching to the following rectangle whereas the bottom-right corner of the current rectangle has not been reached, then there will be an additional waste apart the internal one computed in Case 1. If the current rectangle $(1, \tau)$ contains types t_i and t_j and the following one has a slot of type t_k in its top-left corner, then the current rectangle could potentially have a waste larger than $w_{(i,j)} + w_{(j,k)}$, but it will be at most equal to $w_{(i,k)}$.

Over all rectangles $(1, \tau)$ in the space $B \times T$, the total waste is equal to the sum of the waste within every rectangle. We now from Lemma 5.2.1 that this sum will be at most equal to $w_{(1,\tau)}$, yielding Result 5.2.3. □

Observe that time slots of types t_1 and t_τ could never be placed simultaneously in a rectangle $(1, \tau)$ as can be seen in the example in Figure 9. The same observation holds for types t_i and t_j and rectangles (i, j) (see rectangle I in Figure 9). Observe also that a rectangle (i, j) can have exactly t_i^b/t_j^b time slots exclusively of type t_i or of type t_j (see rectangles II and III in Figure 9). The surface of a rectangle (i, j) is exactly $(t_i^b/t_j^b)\Delta$. Therefore, the space $B \times T$ could be subdivided in exactly $BT/((t_1^b/t_\tau^b)\Delta)$ rectangles $(1, \tau)$.

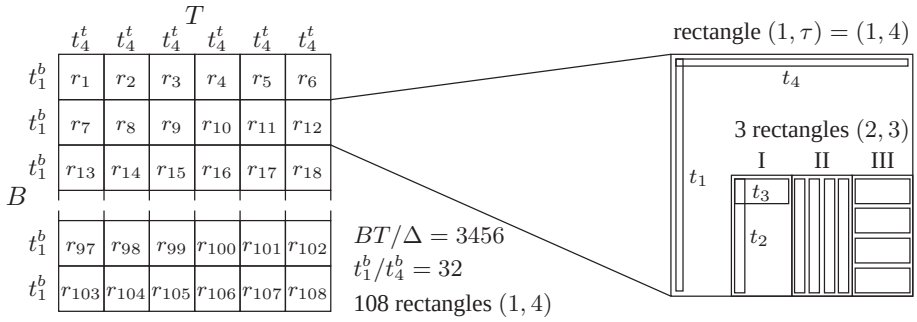


Figure 9: Subdivision of $B \times T$ corresponding to Table 1 and rectangles (i, j).

Introduce $\delta = w_{(1,\tau)}$. From now on, we will consider the following constraint on the number of time slots to be used

$$\sum_{k=1}^{\tau} x^{t_k} \leq \frac{BT}{\Delta} - \delta.$$

We know from Result 5.2.3 that the placement is feasible if this constraint is respected. Observe that for the data in Table 1, this constraint allows to solve the problem of the placement by sacrificing less than $w_{(1,4)}/3456 = 0.897\%$ of the bandwidth. This ratio depends on the data of the problem and cannot be guaranteed for any instance of the problem. The only guarantee is that the fraction of lost

Algorithm 1 Placement algorithm.

Input: An ordered heap H of types $\{t_1, \ldots, t_1, t_2, \ldots, t_2, \ldots, t_\tau, \ldots, t_\tau\}$. An ordered set R of $(1, \tau)$ rectangles ordered from left to right and top to bottom
Ouput: A placement on $B \times T$

Let r be the first $(1, \tau)$ rectangle in R
Set $t_{old} = head(H)$
while H is not empty **do**
  Dequeue t_k from H
  if t_k cannot be placed in r **then**
    Fill empty space in r (if any) with empty types t_{old} {waste}
    Select next rectangle r in R
  else
    if t_k and t_{old} are different **then**
      Jump to next multiple of t_k^b/t_{old}^b filling with empty types t_{old} {waste}
    end if
  end if
  Set $t_{old} = t_k$
  Put t_k in r with leftmost, topmost policy
end while

bandwidth will be less than $w_{(1,\tau)}\Delta/(BT)$. Nevertheless, the given example of Table 1 is representative of the possible instances, and a loss of roughly 1% is definitely satisfying regarding the complexity of the problem. It might be possible to do even better than that by adopting a lower value of δ, assuming that the arrangement will still be feasible. In practice, one can carry out the placement according to many other policies, which may lead to a waste smaller than that of the preceding proof.

It is not trivial to write the placement algorithm. We will therefore give just its simplest version[3] in Algorithm 1. Algorithm 1 consists in filling the space from left to right and "jumping" to the order of multiplicity when there is a change in the type. The set R of rectangles divides the space $B \times T$ and is ordered as illustrated in the example of Figure 9.

Figure 10 depicts a sample output of the placement algorithm working with 4 types of terminals, according to the data in Table 1. This placement is obtained when using Algorithm 1. There are 9 time slots of type t_1, 3 time slots of type t_2, 11 time slots of type t_3 and 4 time slots of type t_4 to place in the time-frequency space of size $B \times T$. The orders of multiplicity are 2 between types t_1 and t_2,

[3] We will consider here the case where only terminals of the same type can transmit together. We will see later on in Section 5.3 that it is possible to assign different types of carriers to distinct spots.

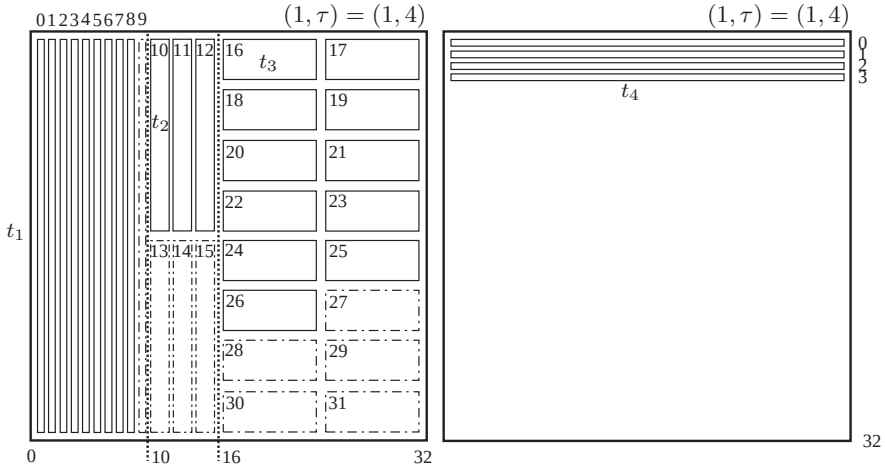


Figure 10: Sample output of the placement algorithm.

and 4 between types t_2 and t_3, and between types t_3 and t_4 (see Section 3.4). The rectangles drawn in dotted lines are "lost spaces" whereas the rectangles in continuous features are time slots of different types placed on the time-frequency space. The vertical dotted lines correspond to the "jumps" occurring at the changes in types. The dotted line at number 10 (resp. 16) corresponds to the change from type t_1 to t_2 (resp. from type t_2 to t_3). One time slot of type t_1 and three time slots of type t_2 were lost in these jumps. To place the first time slot of type t_4, a new rectangle had to be used. The empty space in the first one was filled with five empty time slots of type t_3. The placement of Figure 10 incurred a total loss of 1+3+5=9 time slots ($9 < w_{(1,4)} = 31$). The example of Figure 10 illustrates well the two cases considered in the proof of Result 5.2.3: the change from type t_1 to t_2 is considered in Case 1 (internal waste) and the change from type t_3 to t_4 is considered in Case 2.

The lost space in this example may seem very significant compared to the total amount of time slots to be placed (exactly 27), but this is because of the little demand. As explained before, the losses are inevitable when changing types. The advantage of this policy is that all lost spaces are nicely formatted: they can be used if the demand for certain types of slots is larger.

To conclude this section, we give an example of placement in which the maximum waste allowed is attained. It is the case when there are only 2 time slots to place, the first of type t_1 and the second of type t_τ. Two rectangles $(1,\tau)$ are needed here, and each will have exactly one time slot. There will be $w_{(1,\tau)}$ time slots lost in the first rectangle.

5.3 Satisfying the Global Demand

Instead of allocating time slots of a certain type to a spot, we propose to allocate slots to typified families, *i.e.*, simultaneously in all spots. In a typified family, distinct spots can be assigned different types. If family F_i assigns type t_k to spot s, we will note $F_i^T(s) = t_k$.

Initially, we will consider families with only one type. Thus, for a family F_i, we can choose a type of terminal t_k which will be used on all concerned spots (another family $F_{i'}$ would use another type $t_{k'}$). In other words, $\forall s,\ F_i^T(s) = t_k$. Such families will be denoted as 1-typified families. We place this 1-typified family, in the time-frequency space, at exactly the same place for all concerned spots, implying that all spots would use the same frequency band. In this way, we are sure that the allocation criterion is respected, because of the definition of a family. Over other frequency bands, another family could be used to satisfy another (or the same) demand.

Figure 11 shows a possible placement of the radio resources. If $F_i^T(s) = t_k$, we will note (F_i, t_k) in the rectangle concerned. Thus, this notation is found in all active zones of a family (for instance, zones 0.1 and 2.0 for family F_2). The

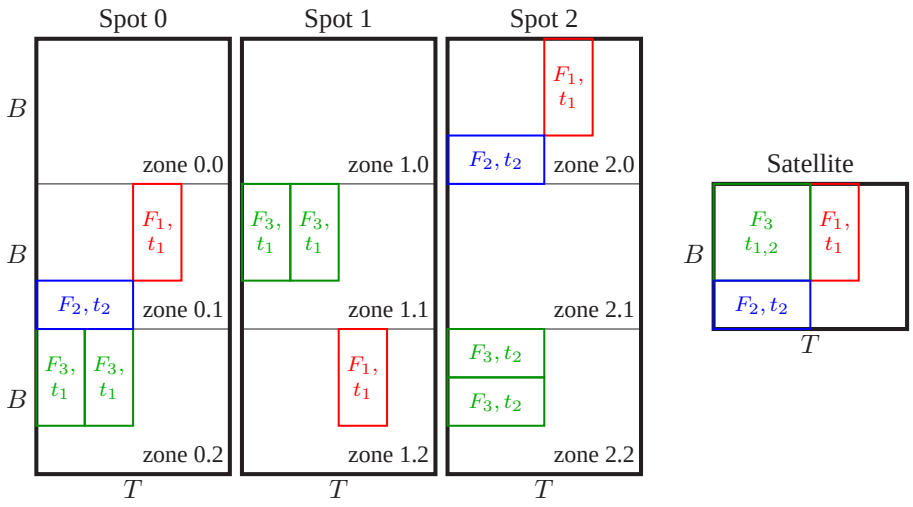


Figure 11: A global example of arranging families.

constraints of capacity on each zone, in terms of bandwidth and time frame are ensured by the constraint of surface of a rectangle (F_i, t_k) on the rectangle $B \times T$.

A family can possibly have several types of terminals according to its different spots. It is the case, for example, for the rectangles (F_3, t_1) in zones 0.2 and 1.0

and (F_3, t_2) in zone 2.2. We will say that family F_3 is 2-typified and its type will be denoted $t_{1,2}$, as can be seen in the diagram at the right of Figure 11 (satellite point of view). These families have a specific order of multiplicity. If t_k is the type in the family having the larger bandwidth and $t_{k'}$ that with the narrower bandwidth, then the order of multiplicity of the family is

$$F_i^M = \frac{t_k^b}{t_{k'}^b} \in \mathbb{N}^* - \{1\}.$$

5.4 Linear Program $\mathcal{P}$

In this section, we define the linear program used to compute a solution, based on the typified families described earlier. Without loss of generality, we consider the case where each spot has three zones. We model the constraints for satisfying demands with equations (11)-(13). Equation (10) provides the time-frequency space constraint of Result 5.2.3.

The variables of the linear program, denoted $\mathcal{P}$, are the x_{F_i}, which represent the number of times that the typified families are used. They must be integer variables. Let $\mathcal{I}$ be the current set of typified families used to solve $\mathcal{P}$. Recall that $d(z, t_k)$ is the demand for type t_k, as defined in Section 3.4. Let $F_i^A(z) = \textit{on}$ denote if zone z could be active, and $F_i^A(z) = \textit{off}$ otherwise. $\mathcal{P}$ is then defined as $\min\ J$ where

$$J = \sum_{i \in \mathcal{I}} F_i^M x_{F_i} \leq \frac{BT}{\Delta} - \delta \tag{10}$$

$$\forall k \in [1, \tau],\ \forall z \in s,\ \sum_{i \in \Gamma(z,k)} F_i^M x_{F_i} \geq d(z, t_k) \tag{11}$$

$$\forall k \in [1, \tau],\ \forall z, z' \in s,\ \sum_{i \in \Gamma(z,z',k)} F_i^M x_{F_i} \geq d(z, t_k) + d(z', t_k) \tag{12}$$

$$\forall k \in [1, \tau],\ \forall\ s,\ \sum_{i \in \Gamma(z,z',z'',k)} F_i^M x_{F_i} \geq d(z, t_k) + d(z', t_k) + d(z'', t_k) \tag{13}$$

with:

$$\begin{aligned}
\Gamma(z,k) &= \left\{ i \in \mathcal{I} / F_i^T(s) = t_k,\ F_i^A(z) = \textit{on} \right\} \\
\Gamma(z,z',k) &= \left\{ i \in \mathcal{I} / F_i^T(s) = t_k,\ (F_i^A(z) = \textit{on}\ \text{ or }\ F_i^A(z') = \textit{on}) \right\} \\
\Gamma(z,z',z'',k) &= \left\{ i \in \mathcal{I} / F_i^T(s) = t_k,\ \exists z \in s / F_i^A(z) = \textit{on} \right\}.
\end{aligned}$$

It is obvious that if (10) is not satisfied, no integer solution can be found. Therefore, we choose to consider the occupied surface as the objective function to minimize.

Minimizing J results in the maximization of reuse of the resources and thus in the maximization of the system throughput.

Result 5.4.1 *Eqs.* (11)-(13) *guarantee the satisfaction of the demand in type* t_k.

Proof. The satisfaction of the demand in type t_k can be computed on a flow from a source s, while passing by 3 arcs (or $nbZones(s)$, if there are $nbZones(s)$ zones) of respective capacities $d_0 = d(z_0, t_k)$, $d_1 = d(z_1, t_k)$, and $d_2 = d(z_2, t_k)$, as seen in Figure 12. The capacities of the other arcs, denoted by $C[z_0, z_1, z_2]$,

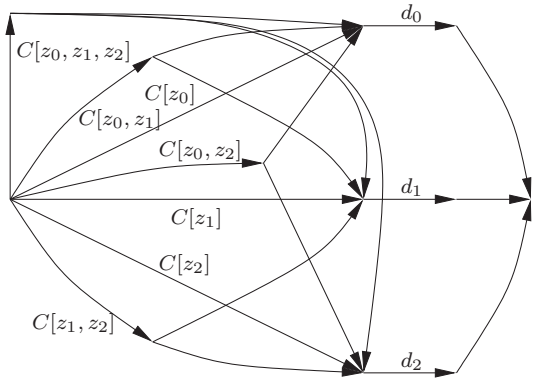


Figure 12: Modeling the constraints of zones as flows.

$C[z_j, z_{j'}]$ for $j \neq j'$, $\{j, j'\} \subset \{0, 1, 2\}$ and $C[z_j]$, $j \in \{0, 1, 2\}$, are given by:

$$\begin{aligned} C[z_0, z_1, z_2] &= \sum_i x_{F_i}^{t_k} \times U[Fi, \{z_0, z_1, z_2\}] \\ C[z_j, z_{j'}] &= \sum_i x_{F_i}^{t_k} \times U[Fi, \{z_j, z_{j'}\}] \\ C[z_j] &= \sum_i x_{F_i}^{t_k} \times U[Fi, \{z_j\}] \end{aligned}$$

where $U[Fi, Z]$ is equal to 1 when F_i could activate either one of the zones of the set Z, and to 0 otherwise. The capacities of all other arcs in the figure are assumed infinite. Indeed, by the theorem of Ford Fulkerson [11] (or in its version of Menger [24]), there is a maximum integer flow from the source to the sink, which is equal to the cardinality of a minimal cut. However, there are 8 cuts of finite size (or $2^{nbZones(s)}$ in the case of $nbZones(s)$ zones), according to the choice of the arcs of capacity d_0, d_1 and d_2. One of these equations is trivial since it stipulates

that the flow of the zones must be less than $d_0 + d_1 + d_2$. The 7 others are checked by our linear program. □

5.5 Optimal Typification of Families

There exists τ^N different ways of typifying a given non-typified family F_i. As it is too much to include them all in $\mathcal{P}$ we will use the concept of *generation of columns* [9]. A column corresponds to one valid typified family. The optimal float solution is obtained when $\mathcal{I}$ is the set of *all* valid typified families, a set that is too large to be used in practice. Actually, the process initializes $\mathcal{I}$ as the set of homogeneously typified families. However, given a restricted $\mathcal{I}$, dual properties allow to identify new columns to be added to $\mathcal{I}$ to improve the solution. We show in the following that dual properties characterize non-typified families, which greatly simplifies the problem of identifying an optimal $\mathcal{I}$.

Let $\mathcal{P}$ be rewritten as follows:

$$\begin{array}{ll} \text{Minimize} & f = c \cdot x \\ \text{Such that} & \left\{ \begin{array}{lcl} Ax & = & b \\ x & \geq & 0 \end{array} \right. \end{array}$$

Let A_B denote the matrix extracted from the corresponding system of equations, and x_B be the vector of the associated families. Let x_N denote the vector of the other families, and A_N be the corresponding matrix. In the same way, we subdivide c in c_B and c_N. We can write

$$A_B x_B + A_N x_N = b \quad \text{and} \quad f = c_B x_B + c_N x_N.$$

It comes then

$$\begin{array}{rcl} x_B & = & A_B^{-1} b - A_B^{-1} A_N x_N, \\ f & = & c_B A_B^{-1} b + (c_N - c_B A_B^{-1} A_N) x_N. \end{array}$$

The equations above return a basic solution to the system with $x_N = 0$. The system is optimal if and only if

$$c_N - c_B A_B^{-1} A_N \geq 0.$$

Thus, the system is improvable if and only if a negative coefficient can be found in the above vector. We further decompose A_N by writing $A_N = [A_{\alpha_1} \cdots A_{\alpha_m}]$ where m is the number of columns of A_N, each column corresponding to a family with subscript α_j. In particular, we have $c_{\alpha_j} = F^M_{\alpha_j}$.

Result 5.5.1 *For any non-typified family $\mathcal{F}$, there exist a constant $K_{\mathcal{F}} > 0$ and a function $\kappa_{\mathcal{F}}$, mapping pairs $(type, \text{Spot})$ to positive real numbers, such that for all typified families deriving from $\mathcal{F}$, we have*

$$c_i - c_B A_B^{-1} A_i = F_i^M \left(K_{\mathcal{F}} - \sum_{s \text{ spot}} \kappa_{\mathcal{F}}(F_i^T(s), s) \right).$$

Proof. Observe that, for a given line of A corresponding to a family F_i, denoted as A_i, all coefficients are either 0 or F_i^M. In addition, if F_i and F_j are typified families deriving from the same non-typified family, then $A_i/F_i^M = A_j/F_j^M$. Also, if c_i and c_j are the coefficients of c corresponding to F_i and F_j, then $c_i/F_i^M = c_j/F_j^M$. Observe that a spot s corresponds specifically to certain lines of A, given by $P_s A$ where P_s is the corresponding projection. If F_i and F_j derive from the same non-typified family and $F_i^T(s) = F_j^T(s)$, then $P_s A_i/F_i^M = P_s A_j/F_j^M$. Last, defining the following constants $K_{\mathcal{F}} := c_i/F_i^M$ and $\kappa_{\mathcal{F}}(F_i^T(s), s) := c_B A_B^{-1} P_s A_i/F_i^M$ yields the result. □

The optimal solution of our program is obtained when $\mathcal{I}$ is the set of *all* typified valid families. Since this set is too large to be used for a computation, we simply start with a restricted $\mathcal{I}$ which is progressively augmented to reach the optimum.

Result 5.5.2 *The program $\mathcal{P}$ with the restricted set of families $\mathcal{I}$ is improvable with respect to the set of all valid families if there exists a non-typified family $\mathcal{F}$ such that*

$$K_{\mathcal{F}} - \sum_{s \text{ spot}} \max_{t \ type} \kappa_{\mathcal{F}}(t, s) < 0. \tag{14}$$

If we find one or several non-typified families which show that the system is improvable, we can strictly improve the solution by introducing the corresponding typified families (with the types found by the above maximization) into the linear program. This property considerably reduces the number of searches to be made in order to reach the optimal solution. In practice, as long as it is assumed that the solution is improvable, it will be possible to restrict the search by choosing a type for all spots in a subset of $\{t_1, \ldots, t_\tau\}$, reducing thereby the coefficient of multiplicity of the derived families and thus, the difficulty of the integrity constraints.

5.6 The Slave Program

Given a set of non-typified valid families, the slave program assigns the types to the families and returns the exact solution of $\mathcal{P}$ among all possible types. At first, the families are 1-typified with all possible types. The solution returns a dual which allows to derive the improving 2-typified families according to Section 5.5. Then the

linear program is solved again and eventually the dual will generate new 2-typified families. The process is iterated until no new 2-typified families are obtained, which means that we have reached the optimal solution given (i) the current set of non-typified families and (ii) the fact that only 2-typified families are used. The same process is done until τ-typified families are considered.

5.7 The Master Program

In this section, we show how we exploit the properties derived in Section 5.5 to find new valid families that will eventually lead to one $\mathcal{I}$ having the optimal solution.

A spot s being either inactive, or either one of its $nbZones(s)$ zones being active, it will have $nbZones(s)+1$ possible states. Hence, for N spots, all having the same number of zones, there will be $(nbZones(s)+1)^N$ combinations to test. For instance, there will be $4^8 = 65536$ combinations to test for an 8-spot configuration in which each spot has exactly 3 zones, which is very reasonable. However, when the number of spots increases, it will no longer be reasonable to generate all families, which makes it difficult to find the optimal float solution.

Fortunately, for moderate numbers of spots, we will still be able to derive an optimal solution in a relatively small time, thanks to a pruning technique described hereafter.

- A "pruner" selects zones within a spot. If several zones have the same gain, then only one of these is selected for an exhaustive search. This step is called "pruning".
- The p families with the highest "improvement potential" are selected. These are the ones having the highest sum in (14).
- Every selected family is "reaugmented" whenever possible. In other words, if there are zones satisfying the allocation criterion without invalidating the family, then these are incorporated in the family.

The valid families generated by this technique are added to $\mathcal{I}$ and used in the next iteration to solve the linear program. This methodology is depicted in Figure 13.

5.8 Integer Solution to $\mathcal{P}$

The resolution of the slave programs enables the generation of the columns giving the best floating solution in each case. All these columns are then introduced into a new *integer* linear program, and are candidates to return the best possible *integer* solution. We stress that a solution exists with a number of non-zero variables x_{F_i}

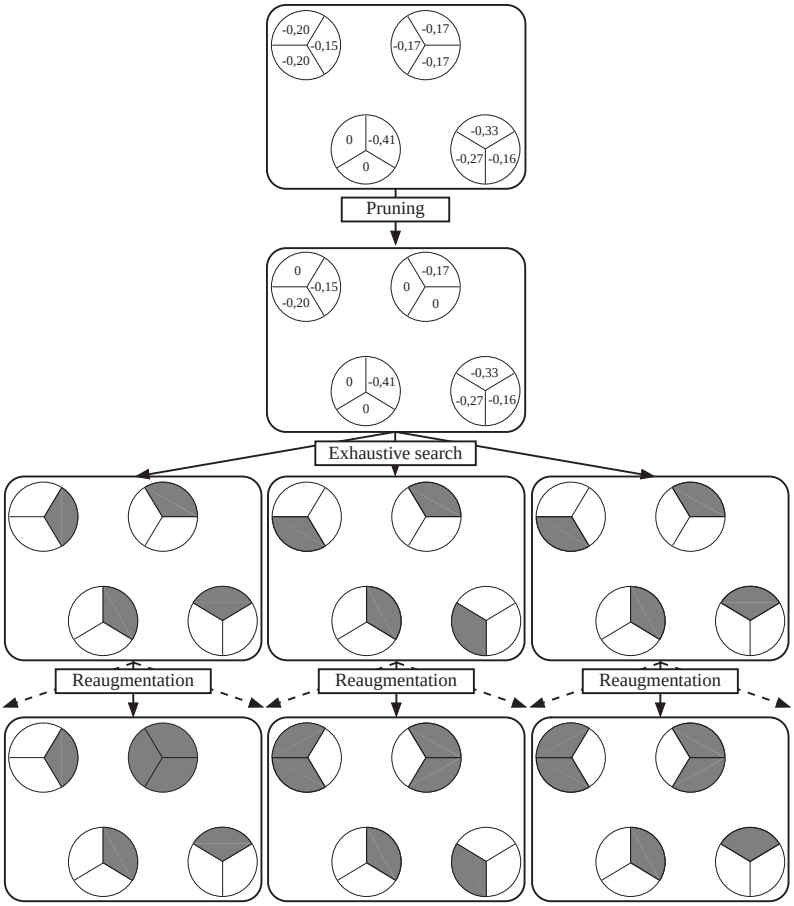


Figure 13: Pruned search of improving families.

at most equal to the number of lines [8, Theorem 9.3, page 145]. For instance, in the case of 8 spots, we know that at most 224 floating variables will be used (896 in the 32 spots case), and therefore a simple ceiling of the variables will give a solution with all variables integer and multiple of 32 at less than 2.1% of the float solution (8.3% in the 32 spots case).

In practice, the resolution of the linear program, using the software Cplex CONCERT 8.0, returns an integer solution, which we arbitrarily fix at 1% of the optimal solution of the float problem. Note that solving completely the problem $\mathcal{P}$, using the columns candidates, cannot be achieved in a reasonable time.

5.9 Algorithm Wrap-up

This part sums up the whole behavior of our algorithm. Each part is represented in Figure 14 by a rectangle (resp. an oval) corresponding to a part of the process

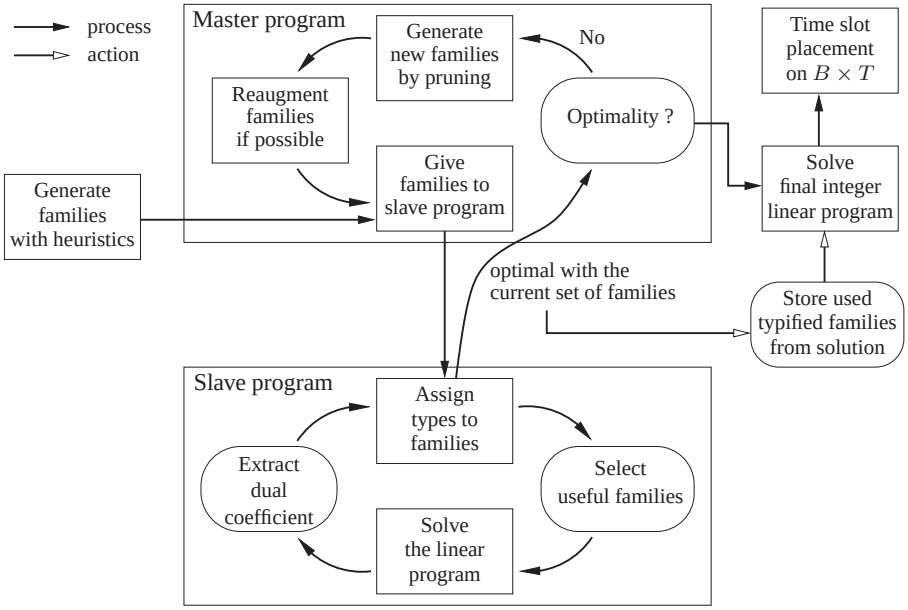


Figure 14: Algorithm overview.

(resp. an action or a decision). We also show the interaction between the master and the slave explained in Sections 5.7 and 5.6. The algorithm starts in the left-most rectangle. We first generate valid but non-typified families as described in Section 5.1. Then, the master program gives directly these families to the slave. The slave program operates as described in Section 5.6: the families are typified, the linear program $\mathcal{P}$ is solved and the slave iterates until reaching optimality. The families involved in the solution are stored for the final integer computation. Afterwards, the master program checks the optimality of the solution given by the slave using the criterion (14). If the optimality is not reached, the pruning technique described in Section 5.7 is performed, generating new families. The master program then calls again the slave, giving it the new families generated. The master/slave process continues until optimality is reached. Next, the final integer linear program is solved as explained in Section 5.8. Finally, we achieve the placement of the resulting number of typified time slots as described in Algorithm 1 in Section 5.2.

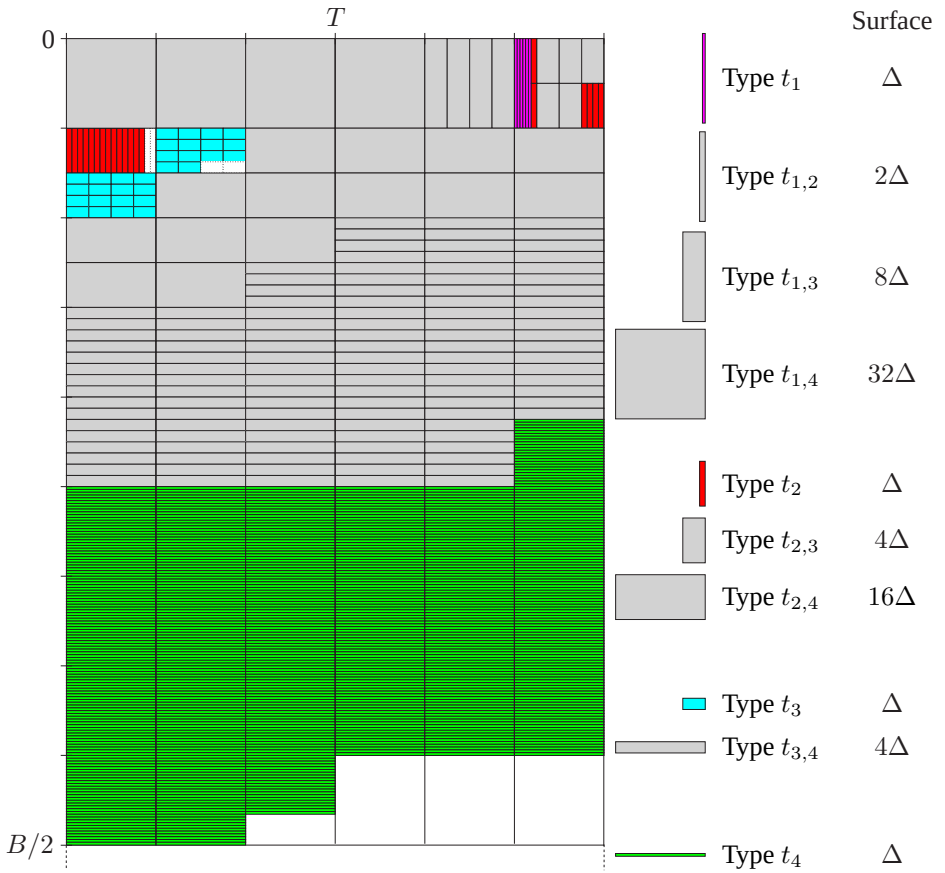


Figure 15: A sample resource allocation.

6 Numerical Results

This section provides some numerical results returned by our approach. We have considered two configurations, the first consisting of 8 spots and the other of 32 spots. The zones demand has been generated according to examples previously provided by ALCATEL SPACE INDUSTRIES. The interferences (in dB) as well as the gains (also in dB) were drawn from uniform distributions, according to specifications provided by ALCATEL SPACE INDUSTRIES. The global interference was considered to be generated mostly by the spots in the vicinity, as the interference generated by remote spots was reduced by 15% ($\gamma = 0.85$).

Our program outputs a time-frequency plan showing the slots allocated, as it can be seen in Figure 15. The time-frequency space therein depicted shows results

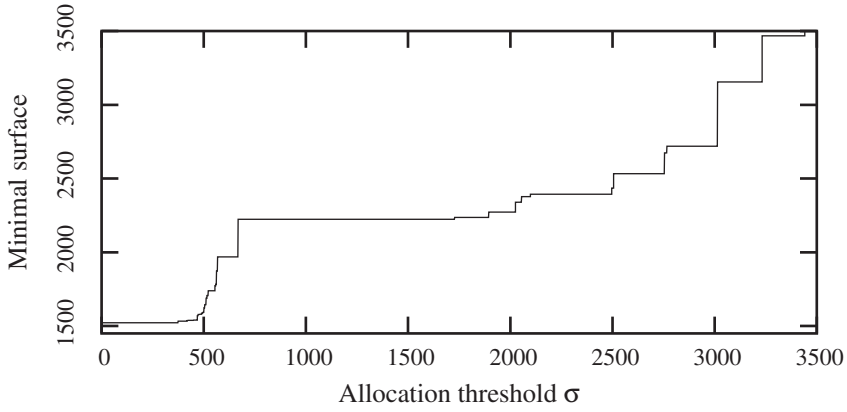


Figure 16: Minimal surface required to satisfy the demand vs. σ.

in the same way as in Figure 10 and Figure 11 (the diagram at the right showing the satellite point of view). Real data, provided by ALCATEL SPACE INDUSTRIES, were used as input to our program and the results are drawn to scale. Figure 15 illustrates well how combinations of types can be used together. For instance, the surface of a block where both types t_1 and t_4 (denoted type $t_{1,4}$ in Figure 15) could be used is 32Δ, where Δ is the surface of a time slot (recall Section 3.4). Observe that $B = 18t_1^b = 36t_2^b = 144t_3^b = 576t_4^b$ and $T = 6t_4^t = 24t_3^t = 96t_2^t = 192t_1^t$ as indicated in Table 1. The lost space here consists of 2 time slots of type t_2 and 2 others of type t_3 (4 white "holes" in Figure 15).

6.1 Results for 8 Spots

In the case where there are only 8 spots per color, our program succeeds in computing the optimal floating solution in about one minute when running on Pentium III machines. This case is particularly interesting as it enables a precise analysis of the effect of the allocation threshold.

We have computed the minimal surface, in the time-frequency plan, that is needed to satisfy the demand, for several values of the allocation threshold σ. The results are plotted in Figure 16. This figure clearly highlights the fact that the minimal surface increases abruptly around certain values of the threshold. Indeed, at some point, the threshold becomes too high impairing the use of some families that will no longer be valid at the considered threshold. The "loss" of these families degrades the solution, yielding a larger minimal surface. Table 6 reports which families become no longer valid at some threshold values. As a consequence, one is able to highlight the configurations of interferences which block the generation

Table 6: Threshold values and families invalidated (X = zone off).

	Spot 0	Spot 1	Spot 2	Spot 3	Spot 4	Spot 5	Spot 6	Spot 7
σ	0 1 2	0 1 2	0 1 2	0 1 2	0 1 2	0 1 2	0 1 2	0 1 2
373	X	X		X		X		
373	X	X		X		X		
418	X	X		X				
423	X	X				X		
450	X	X		X		X		
450	X	X		X		X		
450	X	X		X			X	
469	X	X		X		X		
472	X	X		X		X		
472	X	X		X			X	
472		X		X		X	X	
472		X		X		X	X	
490	X	X		X				
496		X		X		X	X	
501	X	X		X		X		
505	X	X				X	X	
510		X	X			X		

of good solutions. This result has obviously a very strong impact on the design of antennas.

6.2 Results for 32 Spots

For a configuration with 32 spots, we recommend a non-optimal approach using a restricted number of families. We stick to our real-time constraints that consist in obtaining a solution in a few minutes.

Figure 17 depicts the amount of time slots needed to satisfy the demand as a function of the number of valid families used, for several threshold values. Observe that when the pool of families used is larger, the required amount of time slots to satisfy the demand gets smaller. It is therefore more efficient to use a larger pool of families. Observe as well that the solution is more efficient when the allocation threshold σ is smaller, regardless of the number of families used. This observation does not come as a surprise. It is obvious that smaller thresholds would allow a larger number of simultaneous transmissions. Every family would therefore include a larger number of zones that could be active, increasing the efficiency of their use.

As written previously, a larger pool of families improves the solution as it lessens the minimal amount of time slots to be allocated. However, this enhancement comes at the cost of an increased solving time, as it can be seen in Figure 18.

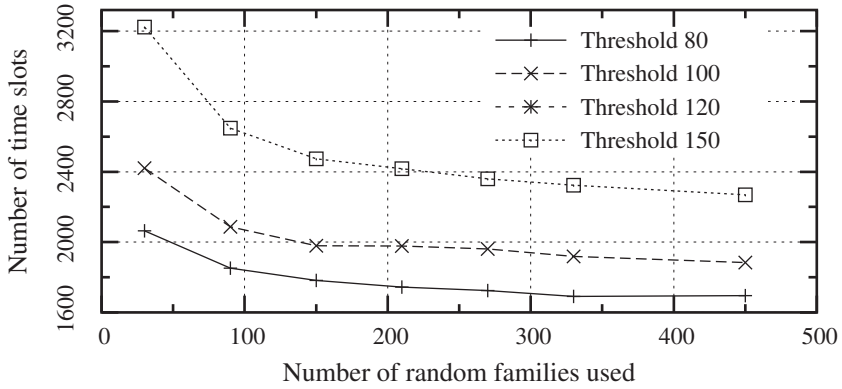


Figure 17: Time slots required to satisfy the demand vs. the number of random families used (32 spots case).

This figure plots the solving time (over Pentium III machines) as a function of the pool size, for several threshold values. Observe that, for the same number of families used, the solving time increases as the threshold values increases. This is mainly due to the time taken for generating the required amount of valid families. For larger thresholds, much more time is needed to generate valid families, as the number of non-valid families gets larger. This is why the difference, between solving times for different thresholds, increases as the number of families to generate increases (see Figure 18).

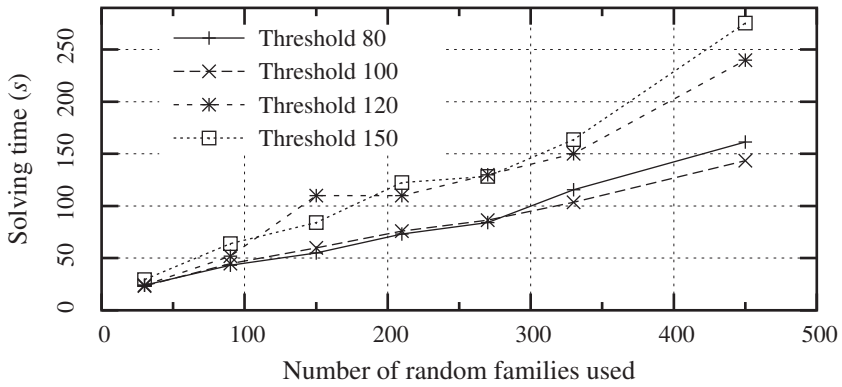


Figure 18: Solving time vs. the number of random families used (32 spots case).

In practice, there is a trade-off between the solving time and the minimal amount of time slots to allocate. For the same number of families used, a small solving time yields a large amount of time slots to satisfy the demand, whereas large solving times yield resource economy. It is then up to the satellite operators to decide for the optimal number of families to use, according to their priorities.

7 Conclusion

In this paper we have devised a novel resource allocation algorithm for MFTDMA satellites. We have considered a more accurate model for satellite communications, first by introducing a realistic modeling of the interferences that are generated by active terminals. Second, we have considered the fact that terminals have specific transmission's capabilities, which is translated into demands of different types of communications. In this context, our model is much more general than the ones presented in Section 2.

We have first introduced the concept of non-concurrent transmissions with the use of families of spots that could transmit simultaneously at the same frequency. These families are then used to allocate time slots to multiple terminals increasing the efficiency of the algorithm. The total demand is satisfied by judiciously placing the different carriers in the radio channel, and the time slots in the corresponding time frames. A linear program is used to compute the number of typified families to use. A column generation process improves these families and selects the good candidates for the last integer programming.

We have shown that with this solution, we can arrange the different carriers in the bandwidth with a less than 1% waste. Our numerical results for a relatively small number of spots have shown that some interference configurations are harmful, in the sense that they impair the use of some families, hence, degrading the efficiency of the solution. For a large number of spots, our results show that a large number of families can improve the efficiency of the solution at the cost of increasing the solving time. Therefore, a trade-off has to be found according to the priorities of the satellite operator.

Acknowledgments

We would like to thank Benoît Fabre, Cécile Guiraud and Isabelle Buret, from ALCATEL SPACE INDUSTRIES, for providing many technical explanations. They have contributed to the modeling of the problem, and have shown a high interest in its resolution.

References

[1] E. Altman, J. Galtier, and C. Touati. A survey on TDMA satellite systems and slot allocation, June 2002. `http://www-sop.inria.fr/maestro/personnel/Eitan.Altman/PAPERS/tdmasurvey.pdf`.

[2] J. Ben-Hur. Technology Summary: Project Nemo. Gaiacomm International Corporation, `http://www.gaiacomminternational.com`, December 2003.

[3] G. Bongiovanni, D. Coppersmith, and C. K. Wong. An optimum time slot assignment algorithm for an SS/TDMA system with variable number of transponders. *IEEE Transactions on Communications*, 29(5):721–726, May 1981.

[4] M. A. Bonuccelli. A fast time slot assignment algorithm for TDM hierarchical switching systems. *IEEE Transactions on Communications*, 37(8):870–874, August 1989.

[5] M. A. Bonuccelli, I. Gopal, and C. K. Wong. Incremental time-slot assignment in SS/TDMA satellite systems. *IEEE Transactions on Communications*, 39(7):1147–1156, July 1991.

[6] S. Chalasani and A. Varma. Efficient time-slot assignment algorithms for SS/TDMA systems with variable-bandwidth beams. *IEEE Transactions on Communications*, 42(2/3/4):1359–1370, Feb/Mar/Apr 1994.

[7] W. Chen, P. Sheu, and J. Yu. Time slot assignment in TDM multicast switching systems. *IEEE Transactions on Communications*, 42(1):149–165, January 1994.

[8] V. Chvatal. *Linear programming*. W. H. Freemann and Company, 1983.

[9] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.

[10] T. ElBatt and A. Ephremides. Frequency reuse impact on the optimum channel partitioning for hybrid wireless systems. In *Proceedings of IMSC '99, Ottawa, Canada*, June 1999.

[11] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, 1962.

[12] N. Funabiki and Y. Takefuji. A parallel algorithm for time-slot assignment problems in TDM hierarchical switching systems. *IEEE Transactions on Communications*, 42(10):2890–2898, October 1994.

[13] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.

[14] H. F. Geerdes and H. Karl. The potential of relaying in cellular networks. In *Proceedings of INOC '03, Evry/Paris, France*, pages 237–242, October 2003.

[15] M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.

[16] I. S. Gopal, M. A. Bonuccelli, and C. K. Wong. Scheduling in multibeam satellites with interfering zones. *IEEE Transactions on Communications*, 31(8):941–951, August 1983.

[17] I. S. Gopal, D. Coppersmith, , and C. K. Wong. Minimizing packet waiting time in a multibeam satellite system. *IEEE Transactions on Communications*, 30(2):305–316, February 1982.

[18] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[19] T. Inukai. An efficient SS/TDMA time slot assignment algorithm. *IEEE Transactions on Communications*, 27(10):1449–1455, October 1979.

[20] Y. Ito, Y. Urano, T. Muratani, and M. Yamaguchi. Analysis of a switch matrix for an SS/TDMA system. *Proceedings of the IEEE*, 65(3):411–419, March 1977.

[21] D. J. Kennedy and *et al.* TDMA burst scheduling within the INTELSAT system. In *Proceedings of GLOBECOM '82, Miami, Florida*, pages 1263–1267, November 1982.

[22] C. King, P. Trusty, J. Jankowski, R. Duesing, and P. Roach. INTELSAT TDMA/DSI burst time plan development. *International Journal of Satellite Communications*, 3(1-2):35–43, 1985.

[23] J. L. Lewandowski, J. W. S. Liu, and C. L. Liu. SS/TDMA time slot assignment with restricted switching modes. *IEEE Transactions on Communications*, 31(1):149–154, January 1983.

[24] K. Menger. Zur allgemeinen kurventheorie. *Fundamenta Mathematicae*, pages 96–115, 1927.

[25] M. Minoux and C. Brouder. Models and algorithms for optimal traffic assignment in SS/TDMA switching systems. *International Journal of Satellite Communications*, 5(1):33–47, 1987.

[26] T. Mizuike, Y. Ito, D. J. Kennedy, and L. N. Nguyen. Burst scheduling algorithms for SS/TDMA systems. *IEEE Transactions on Communications*, 39(4):533–539, April 1991.

[27] T. Mizuike, Y. Ito, L. N. Nguyen, and E. Maeda. Computer-aided planning of SS/TDMA network operation. *IEEE Journal on Selected Areas in Communications*, 9(1):37–47, January 1991.

[28] R. Ramaswamy and P. Dhar. Comments on "An efficient SS/TDMA time slot assignment algorithm". *IEEE Transactions on Communications*, 32(9):1061–1065, September 1984.

[29] A. K. Sinha. A model for TDMA burst assignment and scheduling. *COMSAT Technical Review*, 6:219–251, November 1976.

[30] Y. K. Tham. Burst assignment for satellite-switched and Earth-station frequency-hopping TDMA networks. *Communications, Speech and Vision, IEE Proceedings I*, 137(4):247–255, August 1990.

[31] Y. K. Tham. On fast algorithms for TDM switching assignments in terrestrial and satellite networks. *IEEE Transactions on Communications*, 43(8):2399–2404, August 1995.

[32] B. Toft. Coloring, stable sets and perfect graphs. In R. L. Graham, M. Grötschel, and L. Lovász, editors, *Handbook of combinatorics*, volume 1, chapter 4, pages 233–288. North Holland, 1995.

[33] K. L. Yeung. Efficient time slot assignment algorithms for TDM hierarchical and nonhierarchical switching systems. *IEEE Transactions on Communications*, 49(2):351–359, February 2001.