Scheduling commercial advertisements for television[1]

Abstract. The problem of scheduling the commercial advertisements in the television industry is investigated. Each advertiser client demands that the multiple airings of the same brand advertisement should be as spaced as possible over a given time period. Moreover, audience rating requests have to be taken into account in the scheduling. This is the first time this hard decision problem is dealt with in the literature. We design two mixed integer linear programming (MILP) models. Two constructive heuristics, local search procedures and simulated annealing (SA) approaches are also proposed. Extensive computational experiments, using several instances of various sizes, are performed. The results show that the proposed MILP model which represents the problem as a network flow obtains a larger number of optimal solutions and the best non-exact procedure is the one that uses SA.

Keywords: advertising, scheduling, optimisation, mathematical programming, heuristics

1. Introduction

Commercial advertising is the main source of revenue for television (TV) channels (Alaei and Ghassemi-Tari, 2011). For instance in 2003, 100 billion dollars were spent on U.S. TV advertisements alone (Fleming and Pashkevich, 2007). The two main agents in the advertising campaigns are the TV channels and the advertisers, both are usually faced with solving several difficult decision problems. First the TV channel needs to schedule the programmes into the appropriate time slots or spots (advertising breaks) for the next scheduling horizon (e.g., Reedy *et al.*, 1998). Then the marketing department of such a TV channel sells the spots based on an associated (forecasted) audience rating. The selling mechanism can vary depending on the TV channel business structure and price. For instance, some satellite TV channels in France prepare *packages* of spots where each package has a price, a number of spots and an estimated minimum level of forecasted audiences (Benoist *et al.*, 2007). On the other hand, the advertisers (or advertising agencies acting as intermediaries) buy the spots in order to obtain the most efficient media plan (advertising campaign) while being limited to a maximum budget. The problem of obtaining the optimal media plan for an advertiser involves basically two subproblems. The first is to decide which TV channel to choose and at what time and how often to broadcast the advertisements of the brands (Mihiotis and Tsakiris, 2004; Ghassemi-Tari and Alaei, 2013). The second subproblem is about forecasting the impact of the media plan (Sissors *et al.*, 2002).

Since the offer of spots is usually less than its demand, an auction is commonly conducted for the sales of the spots (Alaei and Ghassemi-Tari, 2011; Ghassemi-Tari and Alaei, 2013) and the TV channel has to decide which bids to accept in order to maximize its revenue (Kimms and Müller-Bungart, 2007). For instance, Jones (2000) describes the case in which the advertisers bids are characterised by certain requirements that the set of spots has to satisfy instead of specifying the spots to buy.

Finally, when the channel has sold the spots, each advertising client sends its order. The order defines the number of times that the advertisements of the advertiser' brands has to be aired. Obviously, the total number of airings of the order is the same as the number of spots that the advertiser has bought. Usually the order does not define the

[1] Supported by the Department of Innovation, Universities and Enterprise of Generalitat de Catalunya under grant BE-DGR-2011.

airdates of its advertisements but some general rules (Kimms and Müller-Bungart, 2007). These rules may include audience rating requirements (e.g., Fleming and Pashkevich, 2007; Pereira *et al.*, 2007) and *regularity* in the multiple airings of the advertisements of the same brand (e.g., Bollapragada *et al.*, 2004). Thus, the problem that the TV channel faces is to determine for each advertiser, the schedule of advertisements (i.e., which advertisement will air in each spot) that best satisfies the order of the advertiser.

In this paper we focus on the latter problem where the satisfaction of the advertisers' orders is important as it can increase the revenue of the TV channel as well. For instance, Bollapragrada *et al*. (2004) describe the case for the U.S. National Broadcasting Company (NBC), in which the improvement of the orders satisfaction resulted in an increase of revenues by over 15 million dollars annually.

To the best of our knowledge, the audience rating and the airing regularity requirements have not been considered simultaneously in the literature. We aim in this study to tackle this decision problem. The contributions of this paper include:

(i) The study of a new scheduling problem which arises in the TV industry
(ii) The development of two novel mathematical formulations
(iii) The design of constructive heuristics, local searches as well as SA-based meta-heuristics to solve the problem
(iv) New benchmark solutions for future research including lower and upper bounds

The remainder of the paper is organised as follows. Section 2 describes the problem and presents an illustrative example. A review of the related literature is provided in Section 3 followed by two mixed integer linear programming (MILP) models which we developed in Section 4. Section 5 presents two constructive heuristics, local searches and three simulated annealing (SA) variants. The computational experiments are given in Section 6 and our conclusions with a brief on future research are covered in Section 7.

2. Definition of the problem

The advertiser usually requests that the multiple airings of each advertisement should be as regular as possible. In other words, any two consecutive airings of the advertisement should be as evenly spaced in time as possible during the airing period. On the other hand, each purchased spot has its audience rating which can be either high, medium or low. The advertiser requests a minimum number of times each advertisement to be aired in high and in medium (or higher) audience rating spots.

Thus, the problem is to obtain the schedule of advertisements that best satisfies the advertisers' requests and which increases the revenues and productivity associated with the television network's sales.

More formally, the problem is defined as follows. An advertiser has C brands to be advertised and for each brand there is one advertisement. Each advertisement c ($c = 1,...,C$) has to be aired n_c times in at least $nh_c + nm_c$ spots with medium or high audience ratings and ensuring that a minimum of nh_c airings is performed in high audience spots. The advertiser purchases N spots such that $N = \sum_{c=1}^{C} n_c$. For each spot s, it is given its air date, t_s, and its (forecasted) audience rating $r_s \in \{H, M, L\}$ ($s = 1, ..., N$), where *H*, *M* and *L* represent high, medium and low audience rating, respectively. Without loss of generality, we can assume that $t_1 = 0$ and $t_s > t_{s-1}$ ($s = 2, ..., N$). The horizon of the airing period is T ($T = t_N$) and the desired or ideal time interval between any two consecutive airings of advertisement *c* is $q_c = T/n_c$.

The aim is to find a feasible broadcast scheduling (that is, an assignment of advertisements into spots that satisfies the audience rating constraints) that minimises the irregularity. As it is done in Bollapragada *et al*. (2004), the objective function consists in minimising the sum of absolute deviations from the ideal timing between each pair of consecutive airings of each advertisement; that is: $Z = \sum_{c=1}^{C} \sum_{k=2}^{n_c} \left| \tau_{ck} - T/n_c \right|$, where τ_{ck} is the time interval between the kth and $(k-1)$th airings of advertisement *c* with $n_c \geq 2$.

An illustrative example

Consider an example based on the one shown in Bollapragada *et al*. (2004), which consists of an advertiser that has bought 17 spots for airing 6 advertisements. Here we also include, for each advertisement, the minimum number of high and medium audience rating spots to be aired. Table 1 shows the advertisement names, the number of airings of each brand advertising and the minimum number of high (#H) and medium (or higher) (#M) audience rating spots in which they must be aired. For instance, for advertisement 'A' it is requested 3 airings with at least one medium (or higher) rating spot.

[Insert Table 1 here]

A feasible solution for this example is given in Table 2, which shows the spot number (#), its corresponding audience rating, the air date, air time and instant (T), in hours, of the spots, and the chosen advertisement to be aired in that spot (Solution). Here the 3 types of audience rating correspond to the following air time intervals: H (9:00 PM - 11:30 PM), M (6:30 PM - 8:00 PM) and L (3:00 PM - 6:00 PM).Thus, the objective function value is computed as:

$Z = Z_A + Z_B + Z_C + Z_D + Z_E + Z_F = 357.5$, where:

$$Z_A = |(191-68)-383/3| + |(383-191)-383/3| = 4.67 + 64.33 = 69$$

$$Z_B = |(115-49)-383/5| + |(167-115)-383/5| + |(265-167)-383/5| + |(381-265)-383/5| = 10.6 + 24.6 + 21.4 + 39.4 = 96$$

$$Z_C = |(138-0)-383/2| = 53.5$$

$$Z_D = |(314-93.5)-383/2| = 29$$

$$Z_E = |(122.5-24)-383/3| + |(284.5-122.5)-383/3| = 29.17 + 34.33 = 63.5$$

$$Z_F = |(193-48)-383/2| = 46.5$$

[Insert Table 2 here]

3. A brief review on related work

Most of the research about scheduling problems in the TV industry has focused on the schedule of programmes (e.g., Reddy *et al*., 1998) but very little research has been explored on the schedule of advertisements from the point of view of the TV channel.

Pereira *et al*. (2007) studies the advertising scheduling problem faced by a Portuguese TV station in which the total viewing for each advertisement is maximized while satisfying several audience, operating and legal constraints as well as fulfilling the audience targets. The authors propose a decision support systems to deal with the problem.

In particular, the regularity on the advertisement airings has been largely ignored in the literature, with the exception of Bollapragada *et al*. (2002, 2004) and Brusco (2008).

Bollapragada *et al*. (2002) introduce an advertising scheduling problem faced by the National Broadcasting Company. It is solved in Bollapragada *et al*. (2004) and later in Brusco (2008) and Gaur *et al*. (2009). These studies focus on the regularity aspect of the schedule only without taken into account the audience rating requirement. Moreover, it is assumed that the time intervals between two consecutive spots are constant. Thus, under this assumption, the procedures proposed in Bollapragada *et al*. (2004) and Brusco (2008) are designed so the airings of the same advertisement are spread as evenly as possible in the sequence of the available spots. However, for example, in Table 2 we can see that the time interval between spots #1 and #2 is 24 hours whereas between spots #3 and #4 it is only 1 hour. Therefore, this assumption is too simplistic in practice. In fact, Bollapragada *et al*. (2004) highlighted the idea that "*the commercials with the same ISCI code* [the airings of each advertisement] *to be as evenly spaced in time as possible*" (p. 688). Gaur *et al*. (2009) introduced some of these ideas by using different weights between pairs of commercials.

Even with these aforementioned simplifications, the problem introduced in Bollapragada *et al*. (2002) is NP-complete and hence ours is also NP-complete (proof is given in the Appendix). Bollapragada *et al*. (2004) propose two mixed integer linear programming (MILP) models, one branch and bound (B&B) algorithm and four heuristics to solve their problem. Brusco (2008) enhances the previous B&B algorithm and presents a simulated algorithm (SA). Gaur *et al*. (2009) addressed the problem by formulating it as a generalisation of the max k-cut problem.

Another problem which is related to ours is the response time variability problem (RTVP). The RTVP is a sequencing optimisation problem that arises whenever products, clients, jobs, etc, need to be sequenced with the aim to minimise the variability in the time between the instants at which they receive the necessary resources. This can be found in a variety of real-life contexts such as mixed-model assembly lines (e.g., Corominas *et al*., 2010), computer multithreaded systems (e.g., Waldspurger and Weihl, 1994; Kubiak, 2009), periodic machine maintenance (e.g., Anily *et al*., 1998) and waste collection (Herrmann, 2011). This problem is defined as follows.

Let C be the number of symbols (products, clients, jobs, etc.) and n_c the number of copies of symbol c (c = 1,...,C) to be sequenced in a sequence $s = s_1 s_2 ... s_N$ of length N ($N = \sum_{c=1}^{C} n_c$). For all types of symbols with $n_c \geq 2$, let the variables d_{ck} be the distance between the position of the kth and $(k-1)$th copies of symbol c (i.e., the number of positions between them, where the distance between two consecutive positions is considered equal to 1). Let also assume that s_1 immediately follows s_N (i.e., it is a circular sequence). Therefore, d_{c1} is the distance between the first copy of symbol c in a cycle and the last copy of the same symbol in the preceding cycle. Note that for all symbols c in which n_c = 1, the variables $d_{c1} = N$. The objective is to minimise $RTV = \sum_{c=1}^{C} \sum_{k=1}^{n_c} \left(d_{ck} - N/n_c \right)^2$.

The above problem has analogies with the schedule of advertisements, where the symbols would be the advertisements, the copies referring to their number of airings and the positions representing the spots. However, our problem differs from the RTVP in the followings:

- the RTVP is cyclic
- the time distances between two consecutive positions are constant
- the differences of the real distances with respect to the ideal distances are squared penalised
- the RTVP is unconstrained

These differences add extra complexity to the problem including especially the inclusion of the new constraints. However, the two problems have an important common characteristic in that both have a fitness function that depends on the relative distances between the positions of each pair of consecutive copies of the same symbol.

The RTVP is shown to be NP-complete (Corominas *et al.*, 2007). The RTVP has been solved by means of MILP (Corominas *et al.*, 2007, 2010), and recently by a B&B algorithm (García-Villoria *et al.*, 2013), adaptive constructive heuristics (Salhi and García-Villoria, 2012), metaheuristics (Corominas *et al.*, 2013; García-Villoria and Pastor, 2013) and hyper-heuristics (García-Villoria *et al.*, 2011).

4. MILP formulations

We present two MILP models (*M1* and *M2*). The first one is a straightforward model whereas the second is based on network flow formulation. The following additional notation is given throughout:

E_{ck}, L_{ck}	Earliest and latest spot in which the kth airing of advertisement c can be assigned, respectively: $E_{ck} = k$, $L_{ck} = N - n_c + k$ ($c = 1,...,C$; $k = 1,...,n_c$).

4.1. Model M1

A first intuitive (non linear) mathematical model, *M1*, is formulated as follows:

Variables

y_{cks} is 1 if the kth airing of advertisement *c*, $c=1,...,C; k=1,...,n_c$, is assigned into spot *s*, $s=E_{ck},...,L_{ck}$; and 0 otherwise.

Model

$$\text{Minimise } Z=\sum_{c=1}^{C}\sum_{k=2}^{n_c}\left|\left(\sum_{s=E_{ck}}^{L_{ck}} t_s\cdot y_{cks}\right)-\left(\sum_{s=E_{c,k-1}}^{L_{c,k-1}} t_s\cdot y_{c,k-1,s}\right)-q_c\right| \qquad (1)$$

subject to

$$\sum_{s=E_{ck}}^{L_{ck}} s\cdot y_{cks}\geq\left(\sum_{s=E_{c,k-1}}^{L_{c,k-1}} s\cdot y_{c,k-1,s}\right)+1 \qquad c=1,...,C;\ k=2,...,n_c \qquad (2)$$

$$\sum_{s=E_{ck}}^{L_{ck}} y_{cks}=1 \qquad c=1,...,C;\ k=1,...,n_c \qquad (3)$$

$$\sum_{c=1}^{C}\sum_{\substack{k=1|\\E_{ck}\leq s\leq L_{ck}}}^{n_c} y_{cks}=1 \qquad s=1,...,N \qquad (4)$$

$$\sum_{k=1}^{n_c}\sum_{\substack{s=E_{ck}|\\r_s=H}}^{L_{ck}} y_{cks}\geq nh_c \qquad c=1,...,C \qquad (5)$$

$$\sum_{k=1}^{n_c}\sum_{\substack{s=E_{ck}|\\r_s=H}}^{L_{ck}} y_{cks}-nh_c+\sum_{k=1}^{n_c}\sum_{\substack{s=E_{ck}|\\r_s=M}}^{L_{ck}} y_{cks}\geq nm_c \qquad c=1,...,C \qquad (6)$$

$$y_{cks}\in\{0,1\} \qquad c=1,...,C;\ k=1,...,n_c;\ s=E_{ck},...,L_{ck} \qquad (7)$$

The objective function (1) is to minimise the sum over all advertisements of the absolute deviations between the actual and ideal lengths of the time interval between two successive airings of the same advertisement. Constraints (2) ensure the natural order between airings of the same advertisement. Constraints (3) guarantee that each airing of each advertisement is assigned to one spot, whereas constraints (4) state that only one advertisement airing is assigned into each spot. Constraints (5) and (6) ensure that each advertisement is aired a minimum number into high and medium (or higher) audience rating spots, respectively. Regarding constraints (6), note that the number of spots with a high audience rating in which advertisement *c* is assigned ($\sum_{k=1}^{n_c}\sum_{\substack{s=E_{ck}|\\r_s=H}}^{L_{ck}} y_{cks}$) minus the number of airings needed in high audience rating spots (nh_c) plus the number of spots with a medium audience rating in which advertisement *c* is assigned must be at least nm_c. Finally, (7) define the binary variables.

The model presented above is non linear due to the objective function. This can however be linearized by introducing the new continuous non negative variables π_{ck}^{+} and π_{ck}^{-} ($c=1,...,C; k=2,...,n_c$):

Model M1 (linearized)

$$\text{Minimise Z} = \sum_{c=1}^{C}\sum_{k=2}^{n_c}\left(\pi_{ck}^{+} + \pi_{ck}^{-}\right) \tag{8}$$

subject to

constraints (2), (3), (4), (5), (6) and (7)

$$\pi_{ck}^{+} - \pi_{ck}^{-} = \left(\sum_{s=E_{ck}}^{L_{ck}} t_s \cdot y_{cks}\right) - \left(\sum_{s=E_{c,k-1}}^{L_{c,k-1}} t_s \cdot y_{c,k-1,s}\right) - q_c \qquad c = 1,...,C;\ k = 2,...,n_c \tag{9}$$

$$\pi_{ck}^{+} \geq 0, \pi_{ck}^{-} \geq 0 \qquad c = 1,...,C;\ k = 2,...,n_c \tag{10}$$

M1 has $N^2 + N - \sum_{c=1}^{C} n_c^2$ binary variables, $2\cdot(N - C)$ continuous variables and $5\cdot N - C$ constraints.

4.2 Model M2

Other scheduling problems in TV have been modelled as network flow problems (e.g., Reddy *et al.*, 1998; Bollapragada *et al.*, 2004). Here we propose a new model, *M2*, based on representing the assignment of the advertisement airings into spots as a network flow problem.

Variables

$f_{cs\sigma}$	is 1 if advertisement *c*, $c = 1,...,C$, is shipped from spot *s*, $s = 1,...,N-1$, to spot σ, $\sigma = s+1,...,N$; and 0 otherwise.
p_{cs}	is 1 if advertisement *c*, $c = 1,...,C$, is assigned into spot *s*, $s = 1,...,N$; and 0 otherwise.
α_{cs}	is 1 if the first airing of advertisement *c*, $c = 1,...,C$, is assigned into spot *s*, $s = 1,..., N - n_c + 1$; and 0 otherwise.
ω_{cs}	is 1 if the last airing of advertisement *c*, $c = 1,...,C$, is assigned into spot *s*, $s = n_c,...,N$; and 0 otherwise.

Model

$$\text{Minimise Z} = \sum_{c=1}^{C}\sum_{s=1}^{N-1}\sum_{\sigma=s+1}^{N} \left|t_\sigma - t_s - q_c\right| \cdot f_{cs\sigma} \tag{11}$$

subject to

$$\sum_{s=1}^{N} p_{cs} = n_c \qquad c = 1,...,C \tag{12}$$

$$\sum_{c=1}^{C} p_{cs} = 1 \qquad s = 1,...,N \tag{13}$$

$$\sum_{s=1}^{N-n_c+1} \alpha_{cs} = 1 \qquad c = 1,...,C \tag{14}$$

$$\alpha_{cs} + \sum_{\sigma=1}^{s-1} f_{c\sigma s} = \sum_{\sigma=s+1}^{N} f_{cs\sigma} \qquad c = 1,...,C;\ s = 1,...,\min\left(N - n_c + 1, n_c - 1\right) \tag{15a}$$

$$\alpha_{cs} + \sum_{\sigma=1}^{s-1} f_{c\sigma s} = \omega_{cs} + \sum_{\sigma=s+1}^{N} f_{cs\sigma} \qquad c=1,...,C;\ s=n_c,...,N-n_c+1 \qquad (15b)$$

$$\sum_{\sigma=1}^{s-1} f_{c\sigma s} = \sum_{\sigma=s+1}^{N} f_{cs\sigma} \qquad c=1,...,C;\ s=N-n_c+2,...,n_c-1 \qquad (15c)$$

$$\sum_{\sigma=1}^{s-1} f_{c\sigma s} = \omega_{cs} + \sum_{\sigma=s+1}^{N} f_{cs\sigma} \qquad c=1,...,C;\ s=\max\left(N-n_c+2,n_c\right),...,N \qquad (15d)$$

$$\sum_{\sigma=s+1}^{N} f_{cs\sigma} = p_{cs} \qquad c=1,...,C;\ s=1,...,n_c-1 \qquad (16a)$$

$$\omega_{cs} + \sum_{\sigma=s+1}^{N} f_{cs\sigma} = p_{cs} \qquad c=1,...,C;\ s=n_c,...,N \qquad (16b)$$

$$\sum_{\substack{s=1| \\ r_s=H}}^{N} p_{cs} \geq nh_c \qquad c=1,...,C \qquad (17)$$

$$\sum_{\substack{s=1| \\ r_s=H}}^{N} p_{cs} - nh_c + \sum_{\substack{s=1| \\ r_s=M}}^{N} p_{cs} \geq nm_c \qquad c=1,...,C \qquad (18)$$

$$f_{cs\sigma} \in \{0,1\} \qquad c=1,...,C;\ s=1,...,N-1;\ \sigma=s+1,...,N \qquad (19)$$

$$p_{cs} \in \{0,1\} \qquad c=1,...,C;\ s=1,...,N \qquad (20)$$

$$\alpha_{cs} \in \{0,1\} \qquad c=1,...,C;\ s=1,...,N-n_c+1 \qquad (21)$$

$$\omega_{cs} \in \{0,1\} \qquad c=1,...,C;\ s=n_c,...,N \qquad (22)$$

The objective function (11) is to minimise the sum of the discrepancies between the ideal and real time intervals for each airing of each advertisement. Constraints (12) state that the requested number of airings of each advertisement is used, whereas constraints (13) state that only one airing is assigned to each spot. Constraints (14) ensure that there is one first airing for each advertisement. Constraints (15a-d) together with constraints (16a-b) guarantee the flow conservation for each advertisement. Finally, constraints (17) and (18) ensure that each advertisement is aired a minimum number into high audience rating spots and a minimum number into medium (or higher) audience rating spots, respectively. Constraints (19) to (22) refer to the binary variables. Note that (15a-d) can be unified in constraints 15b for all $c=1,...,C; s=1,...,N$ by defining new α_{cs} and ω_{cs} variables and setting them to zero, but we split them into 4 type of constraints for the sake of clarity; a similar observation can be done for (16a-d).

M2 has $4 \cdot N \cdot C + 2 \cdot C - 2 \cdot N$ binary variables and $N \cdot (2 \cdot C + 1) + 4 \cdot C$ constraints.

The model *M2* is less intuitive than *M1* but it is more efficient as it produces more optimal solutions as will be shown in Section 6.3 where optimality can be guaranteed for instances up to around 150 spots only. These models are used to validate our heuristic methods that will be presented in the following section, by providing either optimal solutions when possible, or lower and upper bounds if found within a limited CPU time.

5. Heuristic methods

The proposed MILP models are not appropriate for larger instances as will be shown in the computational results section. The best way forward is therefore to propose efficient heuristic methods. We put forward two constructive heuristics, some local search procedures and SA-based algorithms.

5.1. Constructive heuristics

We present two greedy type heuristics which we refer to as *H1* and *H2* both having the following common two steps. In step 1, a trial solution, not necessarily feasible, is obtained. Here, we do not take into account the constraints about the audience rating for the advertisements. In step 2, we aim to transform the trial solution if found infeasible in step 1 into a feasible one by applying a repair mechanism on this trial solution so the audience constraints are satisfied.

The difference between the two heuristics lies in the first step only. These two steps of *H1* and *H2* are described briefly as follows:

H1 first step

This is based on the minimum contribution idea among the advertisements that have pending airings. For each spot *s* = 1,...,*N*, the advertisement that yields the minimum contribution to the objective function based on the current partial solution is selected to be aired in spot *s* (the lexicographical order is used as the tie breaker).

Let us first introduce the following nomenclature:

seq_p:	The sequence of advertisements obtained at step *p*; $p = 0, \ldots, N$. Initially seq_0 is a void sequence.
$\hat{n}_{cp}$:	The number of times left for advertisement *c* to be assigned in seq_{p-1}; $c = 1, \ldots, C$, $p = 1, \ldots, N$.
C_p^1:	The set of advertisements that have to be aired only once and have not been sequenced in seq_{p-1}; $p = 1, \ldots, N$. That is, $C_p^1 = \left\{ c = 1, \ldots, C \mid n_c = \hat{n}_{cp} = 1 \right\}$.
C_p^2:	The set of advertisements that have to be aired more than once and have not been sequenced in seq_{p-1}; $p = 1, \ldots, N$. That is, $C_p^2 = \left\{ c = 1, \ldots, C \mid n_c = \hat{n}_{cp} \geq 2 \right\}$.
C_p^3:	The set of advertisements that have to be aired more than once and have been sequenced in seq_{p-1} at least once; $p = 1, \ldots, N$. That is, $C_p^3 = \left\{ c = 1, \ldots, C \mid \left(n_c \geq 2 \right) \wedge \left(\hat{n}_{cp} < n_c \right) \right\}$.
lsp_{cp}:	The last spot in which advertisement *c* has been sequenced in seq_{p-1}; $c \in C_p^3$, $p = 1, \ldots, N$.

$$\Delta(c,p)=\begin{cases} 0 & \text{, if } c\in C_p^1\cup C_p^2 \\ \left(t_p-t_{lsp_{cp}}\right)-q_c & \text{, if } c\in C_p^3 \end{cases};\ c=1,\ldots,C,\ p=1,\ldots,N.$$

The pseudo-code of the *H1* first step is shown in Figure 1.

[Insert Figure 1 here]

H2 first step

This step is based on a recently developed heuristic by Salhi and García-Villoria (2012) for the RTVP. The heuristic is modified and adapted accordingly to solve this new but related scheduling problem. The idea is to avoid, when selecting the next advertisement to be aired, accumulating an excessive future increase in the time between the next airing of an advertisement and its last airing. In other words, this is similar to the regret cost usually used in the transportation and the allocation literature.

The pseudo-code of the *H2* first step is shown in Figure 2. We can distinguish two phases in the algorithm. Let *R* be the number of steps used by the algorithm to sequence at least one airing of all advertisements that have to be aired more than once. That is, step *R* is the one in which $C_{R+1}^2=\varnothing$ and $C_R^2\neq\varnothing$. The first phase is applied during the first *R* steps (lines 2 to 4 of the pseudo-code) whereas the second phase consists of the remaining $N-R$ steps (lines 5 and 6). In lines 3 and 6, the criterion to select the advertisement to be sequenced at position p is the advertisement c with the maximum $\Delta(c,p)$ value; in case of tie, the criterion shown in Figure 3 is used. In line 4 the criterion is to select the advertisement that has to be aired more times; in case of tie, the lexicographical order is used.

[Insert Figure 2 here]

[Insert Figure 3 here]

The repair mechanism for both H1 and H2

The repair mechanism of the trial solution obtained in the first step is based on the following fact. One advertisement (say c_1) may have been assigned to more high audience rating spots than the minimum required. Using the same token, another advertisement (say c_2) may have been assigned to a fewer than the minimum required. This observation is also applied to the medium audience rating spots constraints. To overcome this problem, we implement an interchange between a pair of spots in which the scheduling of c_2 is closer to fulfil the audience rating constraints while c_1 remains feasible. The trial solution is repaired by applying iteratively this kind of interchanges. A pseudo-code of the repair mechanism is given in Figure 4 where the high audience rating requirement is first repaired (lines 1 to 5 of the pseudo-code) followed by the medium audience rating requirement (lines 6 to 10).

[Insert Figure 4 here]

We observed empirically that the second step not only repairs the trial solution but usually improves its fitness. The reason is that the trial solution of the first step is obtained heuristically (i.e., its optimality is not guaranteed); otherwise, the quality of the solution will either remain unchanged or get worse.

The solutions of the illustrative example given in Section 2 that are obtained with *H1* and *H2* are, respectively, A-B-C-D-A-B-A-F-E-B-F-D-E-C-B-B-E (Z = 456.37) and B-A-E-C-B-D-C-A-B-E-B-F-D-A-B-E-F (Z = 390.7)

5.2. Local search procedures

We propose several local search procedures which are based on generating and exploring the neighbourhood of the current solution.

(i) Neighbourhood structures
The four following neighbourhoods are considered:

N_1: Neighbours obtained by interchanges of each consecutive pair of spots. Only the interchanges that obtain feasible solutions are considered (i.e., solutions in which the audience rating constraints are satisfied). The time complexity for this neighbourhood is $O(N)$.

N_2: Neighbours obtained by interchanges of each consecutive and non-consecutive pair of spots. Only feasible interchanges are considered. This neighbourhood has a $O(N^2)$ time complexity.

N_3: Neighbours obtained by insertions of each advertisement airing in each position of the sequence. Only feasible insertions are considered. This neighbourhood run in $O(N^2)$.

N_{23} : $N_2 \cup N_3$. This neighbourhood has the same time complexity as above namely $O(N^2)$.

(ii) Exploration strategies
The two following exploration strategies of the neighbourhood are proposed:

NEE (non exhaustive exploration): The neighbourhood is generated until a neighbour better than the current solution is found (if it exists), which is selected to be the new current solution. This is also known as the first improvement in the literature.

EE (exhaustive exploration): The entire neighbourhood is generated and the best neighbour which improves the current solution the most is selected. This is also known as the best improvement in the literature.

In total, nine local search procedures are proposed: *LS-N1*, *LS-N2* and *LS-N3* use the *NEE* strategy together with the neighbourhoods N_1, N_2 and N_3, respectively. *LS-N4* and *LS-N5* both use N_{23} with *NEE* except that *LS-N4* explores first the neighbours of N_2 and then the neighbours of N_3 whereas *LS-N5* does the opposite. Note that when the *EE* strategy is used, the order in which the neighbours are explored is irrelevant as the overall best is selected. Also note that *LS-E1*, *LS-E2*, *LS-E3* and *LS-E4* use the *NEE* strategy together with the neighbourhoods N_1, N_2, N_3 and N_{23}, respectively.

5.3. Simulated annealing algorithms

The simulated annealing metaheuristic (SA) was first proposed in Kirkpatrick *et al.* (1983) to solve complex combinatorial optimisation problems. SA has been successfully applied for solving a wide range of combinatorial optimisation problems (Nikolaev and Jacobson, 2010). In particular, SA-based algorithms gave, on average, the best solutions when solving the RTVP (García-Villoria and Pastor, 2013).

SA can be seen as a local search where moves to non-improving solutions are allowed with a certain probability. The objective of accepting non-improving solutions is to avoid being trapped into a local optimum. The metaheuristic starts from an initial solution, which is initially the current solution. Then, at each iteration, a new solution selected at random from the neighbourhood of the current solution is considered. If the neighbour is not worse than the current solution, then the neighbour becomes the current solution; in the case that it is worse, the neighbour can become also the current solution with a probability that depends on: 1) how worse is the neighbour, and 2) the value of a control parameter called *temperature*, which is usually a non-increasing function of the number of iterations.

In practice, standard SA algorithms may be trapped in a local optimum after a certain amount of computational time. This occurs when the temperature (one of the main cooling schedules) is too low and then the probability of accepting worse neighbours is negligible. To reduce the risks of such drawbacks, we propose to reset the temperature of the SA to the initial temperature when the current temperature is lower than the final temperature (Dowsland and Adenso-Díaz, 2003).

Based on the aforementioned idea, we propose three SA algorithms. We refer to these as *SA1*, *SA2* and *SA3*. All three SA algorithms are based on the common scheme whose pseudo-code is shown in Figure 5. Iteratively the temperature is reset to the initial temperature t_0 and the SA procedures obtain an initial solution with the function *obtainInitialSolution*(), whose definition is different in each SA algorithm (line 2 of Figure 5) and defined in Sections 5.3.1 to 5.3.3. Then, the solution jumps iteratively to a neighbour selected at random from the neighbourhood N_{23} (lines 5 to 8 of Figure 5) and the temperature is decreased after *itt* iterations by a cooling factor α, $0 < \alpha < 1$ (lines 3 and 4 of Figure 5). When the temperature is lower than the final temperature t_f (line 3 of Figure 5) then the temperature is reset to t_0 and the process continues until the maximum allowed runtime is reached (line 1 of Figure 5).

The neighbourhood used is N_{23} (line 5 of Figure 5). We have empirically found that this neighbourhood provides the best performance (see Section 6.3, Table 5). With respect to the reduction of the temperature (line 3 of Figure 5), the most popular way is the geometric reduction (Dowsland and Adenso-Díaz, 2003, Henderson *et al.*, 2003). This scheme was also found to be promising when addressing similar type of problems by Brusco (2008) and recently by García-Villoria and Pastor (2013)

Therefore, the SA procedures have the following four parameters: t_0, t_f, *itt* and α. The setting of their values, for each procedure, is discussed in Section 6.2.

[Insert Figure 5 here]

5.3.1. SA1

The solution that returns *obtainInitialSolution*(), which obtains the initial solution each time that the temperature is reset (line 2 of Figure 5), in the *SA1* algorithm is always the same solution found by *H2* as this showed to be the best performer (see Section 6.3, Table 5).

5.3.2. SA2

The solution that returns *obtainInitialSolution*() in the *SA2* algorithm is found by a greedy randomized version of *H2*. The latter is based on the greedy randomized adaptive search procedure (GRASP) as presented by Feo and Resende (1989).

Specifically, the *H2* first step (Figure 2) is randomized as follows.

- The selection of the advertising to be sequenced (lines 3, 4 and 6 of Figure 2) is performed at random from a restricted candidate list (RCL).
- In lines 3 and 6 of Figure 2, the RCL is defined by the *L1* advertisings with the highest $\Delta(c, p)$ values (in case of tie, the tie breaker of Figure 3 is used).
- Also in line 4 of Figure 2, the RCL is defined by the *L2* advertisings with the highest n_c values. In case of tie, the lexicographical order is used.

Thus, *SA2* has two additional parameters, *L1* and *L2*, whose values need to be set and discussed in Section 6.2.

5.3.3. SA3

The solution that returns *obtainInitialSolution*() in the *SA3* algorithm is the solution found by *H2* the first time that *obtainInitialSolution*() is called. In the following calls, *obtainInitialSolution*() returns the best solution found so far during the search of *SA3*.

6. Computational experiments

For our computational testing, we solved the MILP models using IBM ILOG CPLEX 12.2. Regarding the heuristic methods, we implemented them in Java SE 1.6.21. The experiments were run on a PC 3.16 GHz Pentium Intel Core 2 Duo E8500 with 3.46 GB RAM. As schedulers need to solve the problem for several hundred advertiser clients and in an interactive mode, there will be, in practice, a time limit for solving the problem of each client. As noted by Bollapragada *et al*. (2004) and Brusco (2008), this time is set to 600 seconds.

6.1. Generation of the data set

We constructed our data sets based on those 40 instances given by Bollapragada *et al*. (2004) and Brusco (2008) where $N \in [8,50]$, $C \in [2,5]$ and $n_c \in [1,229]$. We set the time intervals ($t_s - t_{s-1}$), the minimum numbers of airings in high and medium audience rating spots (nh_c and nm_c, respectively), and the audience rating of each spot (r_s) as follows. For each of the 40 original instances, we considered 3 scenarios namely low, medium and high diversity of the time intervals between spots. Thus, 120 test instances in total are obtained. The time interval values were randomly generated using a continuous uniform distribution:

- Low diversity: $(t_s - t_{s-1}) \in U(1,2)$, $s = 2,...,N$.
- Medium diversity: $(t_s - t_{s-1}) \in U(1,10)$, $s = 2,...,N$.
- High diversity: $(t_s - t_{s-1}) \in U(1,50)$, $s = 2,...,N$.

We also set $nh_c = \max(1, round(0.1 \cdot n_c))$ and $nm_c = \max(1, round(0.2 \cdot n_c))$, $c = 1,...,C$. The only exception is for the 3 instances generated from the original instance #13, in which n_5 is 1, so nh_5 and nm_5 were set to 1 and 0, respectively. The number of spots with high and medium audience rating follows the distribution $U\left(\sum_{c=1..C} nh_c, 1.1 \cdot \sum_{c=1..C} nh_c\right)$ and $U\left(\sum_{c=1..C} nm_c, 1.1 \cdot \sum_{c=1..C} nm_c\right)$ respectively. In addition, the type of audience rating of each spot (H, M or L) is set at random.

The complete set of instances is available at https://www.ioc.upc.edu/EOLI/research/.

The parameter setting of the SA-based algorithm is given next followed by the results of the proposed procedures.

6.2. Parameters setting

Fine-tuning the parameters of an algorithm is almost always a difficult task. Although the parameter values may have a very strong effect on the performance of the algorithm for each problem, they are often selected using one of the following methods, which are not sufficiently thorough (Eiben *et al*. 1999; Adenso-Díaz and Laguna, 2006). The setting is chosen "by hand", based on a small number of experiments that may or may not be referenced; using the general values recommended for a wide range of problems; using the values reported to be effective in other similar problems; or with no apparent explanation.

In this study, we use a tool known as CALIBRA (Adenso-Díaz and Laguna, 2006). This is a tool specifically designed for fine-tuning the parameters of algorithms and is based on using conjointly Taguchi's fractional factorial experimental designs and a local search procedure. A detailed explanation of CALIBRA can be found in Adenso-Díaz and Laguna (2006) and the tool can be downloaded at http://coruxa.epsig.uniovi.es/~adenso/file_d.html.

We applied CALIBRA to a training set of 40 instances. These instances were generated as the test instances (explained in Section 6.1) but using only the medium diversity scenario. Table 3 shows the parameter values returned by CALIBRA for this particular scheduling problem.

[Insert Table 3 here]

6.3. Results of the proposed procedures

MILP models

We have obtained several optimal solutions with the MILP models *M1* and *M2*. Table 4 summarises the overall results. Column *#Opt* shows the number of optimal solutions found, *#Fea* refers to the number of feasible solutions found (excluding optimal ones) and *#NoSol* denotes the number of instances for which no solution was found within 600 seconds of computational time.

[Insert Table 4 here]

M2 is able to obtain more optimal solutions than *M1* besides having relatively fewer instances without finding a solution. It is also worth noting that all 43 instances that are optimally solved by *M1* were also solved by *M2*. Regarding the computational times, *M1* needs 35.25 seconds on average to solve the 43 instances whereas *M2* needs less than half of a second only, when tested on the same 43 instances. It is therefore clear that our proposed model *M2* is relatively much more superior than *M1* while being approximately 90 times faster.

Among the 84 instances optimally solved by *M2*, 25, 29 and 30 instances belong to low, medium and high time intervals diversity, respectively. We performed the paired t-test (with a confidence level equal to 95%) to evaluate whether the probability of success rate of an instance to be solved optimally is the same in each diversity scenario. The null hypothesis was not rejected for any pair of scenarios. Thus, it can be claimed that this parameter is not necessarily influential in the hardness of the problem. On the other hand, the most influential parameter is found to be the number of spots. *M2* obtained optimal solutions for instances with up to 150 spots with the only exception of 2 instances with 300 spots. For instances between around 200 and 400 spots, *M2* may fail to obtain a feasible (non-optimal proven) solution and always failed when the number exceeds 400.

Additionally, the average value of the 15 upper bounds found by *M2* whose optimality is not demonstrated (see Table 4) is 3,137.41, whereas *SA1* and *SA3* (which are the best procedures as explained below) produced, for the same 15 instances, a much smaller average value of 2,455.86 and 2,443.39, respectively. Moreover, *SA1* and *SA3* dominate *M2* over the 15 instances as they always found a better solution in each case.

Because of these limitations, the use of heuristic methods can be considered to be the best way forward for solving medium and specially large instances.

Non exact procedures

Table 5 shows the results of the heuristic and local search procedures. These include the average value of the solutions (*Z*); the average (*Av*) and maximum (*Max*) deviations of the value of the solution found by each procedure with respect to the best known solution value; the average and maximum computational time (in seconds); and *#LO* the number of local minima found within the 600 elapsed seconds. It is worth noting that the minimum deviation for all our procedures is found to be 0 and the minimum computational times is negligible amounting to less than 1 hundredth of a second only.

Constructive heuristics
Both heuristics are very fast requiring less than half a second computational time on average. On the other hand, the performance of *H2* is clearly better since the average *Z* value and average and maximum deviations are much smaller (see Table 5). Additionally, we observed that *H2* achieves solutions that are of the same or of a better quality than the ones found by *H1* for 114 instances.

[Insert Table 5 here]

Effect of the local searches
Here, we applied the 9 local search procedures, *LS-N1* to *LS-N5* and *LS-E1* to *LS-E4*, to the solutions found by the best constructive heuristic namely *H2*. Note that the local search procedures may obviously not find a local minimum within the cpu time limit of 600 seconds.

The average *Z* value and deviation of the solutions obtained with the *NEE* exploration strategy are very similar to those obtained with the *EE* strategy. The only exception appears when the neighbourhood N_{23} is used; then the *NEE* strategy is found to obtain a slightly better value on average but a slightly worse average deviation than *EE*. However, the paired t-test was performed and the differences between the *NEE* and *EE* strategies are found to be statistical significant at the 5% level. On the other hand, the order in which the neighbours are explored has an important effect in terms of computational time. For instance, the time savings of *LS-N4* with respect to *LS-N5* is nearly 35% (86.54 s vs. 122.53 s).

Regarding the neighbourhood structure, it seems that its influence in the performance is greater. The worst average *Z* value and deviation are obtained with the neighbourhood N_1, whereas the best averages are obtained with N_{23}. Specifically, *LS-N1* obtains the greatest average *Z* value and deviation (1,251.82 and 22.65%, respectively), whereas the smallest average Z value and deviation are obtained with *LS-N4* (1,212.89) and with *LS-E4* (15.30%), respectively. The paired t-test confirmed that the quality of the solutions obtained with *LS-N4* and *LS-E4* with respect to those obtained with *LS-N1* are statistical significant. On the other hand, it should be noted that *LS-N1* is much quicker than *LS-N4* and *LS-E4*.

It is worth noting that the best results are obtained with local searches that use the neighbourhood structure N_{23}. This could be due to the fact N_{23} is a larger neighbourhood which includes both N_2 and N_3. The use of such a larger neighbourhood can be considered as special case of very large neighbourhood search which has shown to be a powerful search strategy, see Ahuja *et al.* (2002). This approach may have the handicap of requiring excessive computational effort if not controlled efficiently. In our particular computational experiments, the maximum allowed computational time of 600 sec was found to be large enough to allow the search to converge into a local optimum for most of the instances (from 84.11% to 95.00% of instances depending of the exploration strategies used) and hence render the above limitation of less significance. Therefore, one may expect that the local searches *LS-N4* and *LS-E4* would, on average, be the best local searches.

Our findings also support the above that the best solutions with local searches are obtained with *LS-N4* and *LS-E4*. There is however no significant difference between these two procedures except that *LS-N4* needs on average 86.54 sec whereas *LS-E4* requires nearly 130 sec. Thus, one could claim that *LS-N4* is the most appropriate local search procedure to use.

We also implemented a reduced version of *LS-N*$_4$ where only the neighbours that satisfy $\underset{c=1}{\overset{C}{\forall}} \underset{s=1}{\overset{N-1}{\forall}} \underset{\sigma=s+1}{\overset{N}{\forall}} \left| t_\sigma - t_s - q_c \right| \cdot f_{cs\sigma} \leq \beta \cdot T$ are generated. In this experiment we used $\beta = 0.2$. The reason is that it is expected that solutions with any large discrepancy, for all advertisings, between ideal and real time intervals in consecutive airings would yield a poor quality solution. Although a small cpu time is gained (81.80 s vs 86.54 s), a significant loss in solution quality is recorded (1,264.02 vs 1,212.89). Obviously, better solutions may be obtained if larger values of β were tested but the cpu gain would be even smaller.

Effects of SA
As in our previous experiments, each instance was also solved using a computational time of 600 seconds with the SA-based algorithms. Table 6 shows the average value of the solutions (*Z*); the average (*Av*) and maximum (*Max*) deviations of the value of the solution found by each procedure with respect to the best known solution value; and the average time in which the best solution obtained by the procedure is found (T_{best}), whereas the average solution values found over time are shown in Figure 6. The minimum deviation is 0 for all SA algorithms.

[Insert Table 6 here]

On average, all SA algorithms improve the solutions found by the local search procedures. The smaller average *Z* value, deviation and time to obtain their best solution are obtained with *SA3*, although the results of *SA1* are very close. The paired t-test reveals that at the 5% significant level there are not significant differences between the results of *SA1* and *SA3* but both differs from *SA2*. The average *Z* values obtained over time shown in Figure 6 also illustrate that the convergence of *SA1* and *SA3* are similar while being both quicker than *SA2*. In addition, after 500 seconds, *SA1* and *SA3* experience a moderate average improvement only. As an additional experiment we also doubled the available computational time to 1,200 seconds to see the effect of these approaches. It was found that the average values of the solutions obtained with *SA1*, *SA2* and *SA3* are 1,136.75, 1,152.01 and 1,128.12, respectively showing that all SA algorithms have a good convergence within 600 seconds. Also, it is worth noting that after 250 seconds, *SA3* outperforms on average *LS-N4* by gaining around 7% improvement over *LS-N4* which got stuck at a local minimum much earlier.

[Insert Figure 6 here]

Next we analyse the results of the SA procedures splitting them in the 3 scenarios of low, medium and high diversity of the time intervals between spots. Table 7 shows, for each scenario, the average *Z* value and deviation.

[Insert Table 7 here]

As we can expect, the higher is the diversity between the airing times of the spots, the higher is the average *Z* value (recall that the *Z* value measures the irregularity of the interval times between airings of the same advertisement). It is more indicative the average deviation, which decrease with the diversity. This shows that SA procedures are more robust when solving the high diversity instances. Additionally, we can see that the *SA1* and *SA3* algorithms are clearly better on average than *SA2* regardless the diversity of the time-intervals airings.

6.4. Performance of the best non exact method vs. optimal or lower bound solutions

Optimal solutions
In order to obtain more optimal solutions, we solved the instances with the model *M2* setting the maximum computational time to 7,200 s. However, only 4 new more optimal solutions were founds. That give us a total of 88 known optimal solutions.

On average the best solutions are achieved with *SA1* and *SA3* as shown in the previous subsection. In order to assess the performance of these two SA algorithms we use the 88 optimal solutions for comparison. The average *Z* value of the optimal solutions is 503.57 and that of *SA1* and *SA3* are 526.57 and 522.86, respectively, producing only an average deviation of 1.21% and 1.18%. In addition, *SA1* and *SA3* obtained 77.27% of the optimal solutions (i.e., 68 of the 88 instances).

Lower Bounds-
Regarding the 32 remaining instances, one may be tempted to use appropriate lower bounds if possible. Two lower bounds are explored here. The first (*LB1*) is a crude analytical one whereas the second (*LB2*) is the standard LP relaxation.

LB1 can be derived by considering the individual lower bounds of the contribution of each advertisement *c* to the objective function, *LB1*$_c$, and then compute the sum of the ($n_c - 1$)th lowest values in the set $\{|t_\sigma - t_s - q_c| : s = 1..N-1, \sigma = s+1..N\}$. In other words, this lower bound can be defined as $LB1 = \sum_{c=1..C} LB1_c$.

We computed the average value of *LB1* on the 88 instances where optimality is guaranteed. An average value of 65.26 is recorded which compares rather badly against the average optimal value of 503.57. This result shows that *LB1* seems weak and hence it is not informative enough to rely upon though it can be used as a basis for possible future extension in tightening the lower bound.

A better lower bound can be obtained using $LB = \max(LB1, LB2)$ where *LB2* is the best lower bound returned by CPLEX when solving *M1* and *M2*. The results are based on the 32 instances whose optimal solution is unknown. The average value of *LB* is recorded as 2,397.42 whereas the average value of the *SA1* and *SA3* solutions based on the same 32 instances are 2,831.46 and 2,809.87 yielding an average deviation equal to 40.49% and 40.15%, respectively. Thus, the information provided by the lower bounds about the quality of the solutions is very limited. In addition, it was also noted that *LB* in these particular instances was always dominated by *LB2* supporting our earlier observation on the weakness of *LB1*.

7. Conclusions and future research

A study to solve a scheduling problem arising in the television industry when deciding the airings of the advertisings during the broadcast season is investigated. To the best of our knowledge, this is the first time that this scheduling problem is examined where both the variability of the time intervals between consecutive airings of the same advertising and the audience rating requests are both taken into account.

We propose two MILP formulations including a new network flow-based model (*M2*) which was found to be more efficient in solving optimally instances up to 150 spots. For larger problems, we present constructive heuristics, local search procedures and SA-based algorithms. The best method is the SA algorithm that dynamically updates its initial solution. This variant also proved to be effective as it compares favourably versus the optimal method by producing solutions within 3% of optimality on those relatively small instances.

As the lower bound *LB1* seems to be not tight enough, an interesting line of research would be to develop a tighter lower bound as an important key for developing a B&B algorithm. This algorithm may improve the exact resolution of the problem. Moreover, an effective design of a hybridisation of heuristic methods and the B&B algorithm may also be investigated for larger instances. An overview of heuristic search including hybridization can be found in Salhi (2006).

The problem presented in this work is solved sequentially for each advertiser. However, in order to include other constrains such as to avoid the proximity of airings of advertisements of competing products or service (Brown, 1969), all these problems need to be solved simultaneously. This is an interesting issue that is worth exploring in the future.

Acknowledgments- We would like to thank the editors and the referees for their valuable comments and suggestions that improved both the quality as well as the content of the paper.

Appendix - NP completeness of the problem

Let the problem be called *P1*. *P1* is NP-complete if there is no polynomial time algorithm for solving *P1* unless $P = NP$.

Let *P2* be the problem introduced in Bollapragada *et al*. (2002), which is known to be NP-complete. *P2* is similar to *P1* but there is not audience rating constraints and the time intervals between two consecutive spots are constant. An instance of *P2* is defined by C' and $n_c^{'}$ ($c=1,...,C'$), where C' is the number of advertisements and $n_c^{'}$ is the number of airings of advertisement c. An instance of *P1* can be built in a polynomial time as follows: $C=C'$, $n_c=n_c^{'}$ ($c=1,...,C'$), $nh_c=nm_c=0$ ($c=1,...,C'$), $t_s=s-1$ and $r_s=L$ ($s=1,...,\sum_{c=1}^{C'}n_c$). Since a solution of *P1* is also a solution of *P2* (with the same value of the objective function), this shows that *P1* is also NP-complete.

REFERENCES

Ahuja, R., Ergun, O., Orlin, J. and Punnen,A. (2002) 'A survey of very large-scale neighbourhood search techniques', *Discrete Applied Mathematics,* Vol. 123, pp. 75-102.

Alaei, R. and Ghassemi-Tari, F. (2011) 'Development of a Genetic Algorithm for Advertising Time Allocation Problems', *Journal of Industrial and Systems Engineering,* Vol. 4, pp. 245-255.

Adenso-Díaz, B. and Laguna, M. (2006) 'Fine-tuning of algorithms using fractional experimental designs and local search', *Operations Research*, Vol. 54, pp. 99-114.

Anily, S., Glass, C.A. and Hassin, R. (1998) 'The scheduling of maintenance service', *Discrete Applied Mathematics,* Vol. 82, pp. 27-42.

Benoist, T., Bourreau, E. and Rottermbourg, B. (2007) 'The TV-Break Packing Problem', *European Journal of Operational Research,* Vol. 176, pp. 1371-1386.

Bollapragada, S., Cheng, H., Philips, M. and Garbiras, M. (2002) 'NBC's optimzation systems increase revenues and productivity', *Interfaces*, Vol. 32, pp. 47-60.

Bollapragada, S., Bussieck, M.R. and Mallik, S. (2004) 'Scheduling commercial videotapes in broadcast television', *Operations Research*, Vol. 52, pp. 679-689.

Brown, A.R. (1969) 'Selling Television Time: An Optimization Problem', *The Computer Journal*, Vol. 12, pp. 201-207.

Brusco, M.J. (2008) 'Scheduling advertising slots for television', *Journal of the Operational Research Society,* Vol. 59, pp. 1363-1372.

Corominas, A., Kubiak, W. and Moreno, N. (2007) 'Response time variability', *Journal of Scheduling,* Vol. 10, pp. 97-110.

Corominas, A., Kubiak, W. and Pastor, R. (2010) 'Mathematical programming modeling of the Response Time Variability Problem', *European Journal of Operational Research*, Vol. 200, pp. 347-357.

Corominas, A., García-Villoria, A. and Pastor, R. (2013) 'Metaheuristic algorithms hybridized with variable neighbourhood search for solving the response time variability problem', *TOP*, Vol. 21, pp. 296-312.

Dowsland, K.A. and Adenso-Díaz, B. (2003) ‘Heuristic design and fundamentals of the Simulated Annealing’, *Inteligencia Artificial*, Vol. 19, pp. 93-102.

Eiben, A.E., Hinterding, R., and Michalewicz, Z. (1999) ‘Parameter control in evolutionary algorithms’, *IEEE Transactions on Evolutionary Computation*, Vol. 3, pp. 124-141.

Feo, T.A. and Resende, M.G.C. (1989) ‘A probabilistic heuristic for a computationally difficult set covering problem’, *Operations Research Letters*, Vol. 8, pp. 67-81.

Fleming, P.J. and Pashkevich, M.A. (2007) ‘Optimal Advertising Campaign Generation for Multiple Brands Using MOGA’, *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, Vol. 37, pp. 1190-1201.

García-Villoria, A., Salhi, S., Corominas, A. and Pastor, R. (2011) ‘Hyper-heuristic approaches for the response time variability problem’, *European Journal of Operational Research*, Vol. 211, pp. 160-169.

García-Villoria, A. and Pastor, R. (2013) ‘Simulated annealing for improving the solution of the response time variability problem (RTVP)’, *International Journal of Production Research*, Vol. 51, pp. 4911-4920.

García-Villoria, A., Corominas, A., Delorme, X., Dolgui, A., Kubiak, W. and Pastor, R. (2013) ‘A branch and bound algorithm for the response time variability problem’, *Journal of Scheduling*, Vol. 16, pp. 243-252.

Gaur, D.R., Krishnamurti, R. and Kohli, R. (2009) `Conflict resolution in the scheduling in the television commercials’, *Operations Research*, Vol. 57, pp. 1098-1105.

Ghassemi-Tari, F. and Alaei, R. (2013) `Scheduling TV commercials using genetic algorithms’, *International Journal of Production Research*, Vol. 51, pp. 4921-4929.

Henderson, D., Jacobson, S.H. and Johnson, A.W. (2003) ‘The Theory and Practice of Simulated Annealing’, Chapter 10 in *Handbook of Metaheuristics*, Eds. Glover and Kochenberger, Kluwer Academic Publishers.

Herrmann, J.W. (2011) ‘Using aggregation to reduce response time variability in cyclic fair sequences’, *Journal of Scheduling*, Vol. 14, pp. 39-55.

Jones, J.J. (2000) Incompletely Specified Combinatorial Auction: An Alternative Allocation Mechanism for Business to Business Negotiations. PhD Dissertation, University of Florida.

Kimms, A. and Müller-Bungart, M. (2007) ‘Revenue management for broadcasting commercials: the channel's problem of selecting and scheduling the advertisements to be aired’, *International Journal of Revenue Management*, Vol. 1, pp. 28-44.

Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. (1983) ‘Optimization by simulated annealing’, *Science*, Vol. 220, pp. 671-680.

Kubiak, W. (2009). Response Time Variability. Chapter 7 in Proportional optimization and fairness. Springer.

Mihiotis, A. and Tsakaris, I. (2004) ‘A mathematical programming study of advertising allocation problem’, *Applied Mathematics and Computation*, Vol. 148, pp. 373-379.

Nikolaev, A.G. and Jacobson, S.H. (2010) ‘Simulated Annealing’, Chapter 1 in *Handbook of Metaheuristics*, Eds. Gendreau and Potvin, Springer, 2nd edition.

Pereira, P.A., Fontes, F.A.C.C. and Fontes, D.B.M.M. (2007) ‘A Decision Support System for Planning Promotion Time Slots’, *Operations Research Proceedings*, Vol. 2007, pp. 147-152.

Reddy, S.K., Aronson, J.E. and Stam, A. (1998) ‘SPOT: Scheduling programs optimally for television’, *Management Science*, Vol. 35, pp. 192-207

Salhi, S. (2006) ‘Heuristic Search: the Science of Tomorrow’, in Salhi S (Ed), *OR48 Key Note Papers*, London, ORS, 39-58.

Salhi, S. and García-Villoria, A. (2012) ‘An adaptive Search for the Response Time Variability Problem’, *Journal of the Operational Research Society*, Vol. 63, pp. 597-605.

Sissors, J., Baron, R. and Ephorn, E. (2002) *Advertising Media Planning*. McGraw-Hill.

Waldspurger, C.A. and Weihl, W.E. (1994) ‘Lottery Scheduling: Flexible Proportional-Share Resource Management’, *First USENIX Symposium on Operating System Design and Implementation*, Monterey, California.

Table 1. Advertisings to be aired.

Advertising	Airings	#H	#M
A	3	0	1
B	5	2	1
C	2	1	0
D	2	1	1
E	3	1	1
F	2	1	1

Table 2. Schedule of the advertisings in the purchased spots.

Spot #	Rating	Air date	Air time	T (h)	Solution
1	H	5/11/2000	9:00 PM	0	C
2	H	5/12/2000	9:00 PM	24	E
3	H	5/13/2000	9:00 PM	48	F
4	H	5/13/2000	10:00 PM	49	B
5	L	5/14/2000	5:00 PM	68	A
6	M	5/15/2000	6:30 PM	93.5	D
7	L	5/16/2000	4:00 PM	115	B
8	H	5/16/2000	11:30 PM	122.5	E
9	L	5/17/2000	3:00 PM	138	C
10	M	5/18/2000	8:00 PM	167	B
11	M	5/19/2000	8:00 PM	191	A
12	H	5/19/2000	10:00 PM	193	F
13	H	5/22/2000	10:00 PM	265	B
14	L	5/23/2000	5:30 PM	284.5	E
15	H	5/24/2000	11:00 PM	314	D
16	L	5/27/2000	6:00 PM	381	B
17	M	5/27/2000	8:00 PM	383	A

Table 3. Calibration of the SA-based algorithms

	t_0	t_f	*itt*	*α*	*L1*	*L2*
SA1	38.000	0.001	2200	0.990	--	--
SA2	25.000	0.001	1900	0.990	1	2
SA3	38.000	0.001	1900	0.990	--	--

Table 4. Results obtained with the MILP models

	#Opt	*#Fea*	*#NoSol*
M1	43	29	48
M2	84	15	21

Table 5. Average results obtained with the heuristic and local search procedures

		Deviation (%)		*Time (s)*		
	Z	*Av*	*Max*	*Av*	*Max*	*#LO*
H1	4,003.65	200.52	509.56	0.48	7.62	---
H2	1,339.85	31.38	170.81	0.41	4.93	---
LS-N1	1,251.82	22.65	114.31	1.10	12.38	120
LS-N2	1,225.10	16.97	115.73	65.39	600.00	118
LS-N3	1,242.05	20.77	93.30	109.52	600.00	105
LS-N4	**1,212.89**	15.89	115.73	86.54	600.00	114
LS-N5	1,218.50	15.70	70.41	122.53	600.00	102
LS-E1	1,250.52	22.41	114.31	1.36	15.97	120
LS-E2	1,224.23	16.01	70.41	75.11	600.00	114
LS-E3	1,240.23	19.55	84.74	115.14	600.00	102
LS-E4	1,224.11	**15.30**	70.41	129.92	600.00	101

Table 6. Average values obtained with the three SA-based algorithms and instants (in s) in which the best solution is obtained

		SA1	***SA2***	***SA3***
Z		1,141.21	1,166.83	**1,132.73**
Deviation (%) from the best known solution	***Av***	2.50	7.67	**2.45**
	Max	20.99	117.77	20.99
T_{best}		149.83	207.08	156.34

Table 7. Average *Z* value (average dispersion between parenthesis) of the SA-based algorithms.

	SA1	***SA2***	***SA3***
Low diversity	118.96 (3.86%)	161.93 (15.99%)	118.81 (4.08%)
Medium diversity	560.90 (2.18%)	630.63 (5.54%)	560.56 (2.13%)
High diversity	2,743.77 (1.46%)	2,707.94 (1.49%)	2,718.81 (1.13%)

```
0.  Let $seq_0$ be a void sequence
1.  For $(p := 1; p \leq N; p := p+1)$ do:
2.      $c_p^* := \arg\min_{c=1,\dots,C} |\Delta(c,p)|$. In case of tie, use the lexicographical order.
3.      Construct $seq_p$ by sequencing $c_p^*$ in $seq_{p-1}$
4.  End for
5.  Return $seq_N$
```

Figure 1. The pseudo-code of *H1* first step

```
0.  Let $seq_0$ be a void sequence
1.  For $(p := 1; p \leq N; p := p+1)$ do:
2.      If $C_p^2 \neq \varnothing$ then:
3.          If $\exists c : c \in C_p^3 \mid \Delta(c,p) \geq 0$ then $c_p^*$ is the advertisement $c \in C_p^3$ with the
                highest $\Delta(c,p)$ value. In case of tie, use the tie breaker of Figure 3.
4.          Otherwise $c_p^*$ is the advertisement $c \in C_p^2$ with the highest $n_c$ value. If
                there is a tie, use the lexicographical order.
5.      Otherwise ($C_p^2 = \varnothing$):
6.          $c_p^*$ is the advertisement $c \in C_p^1 \cup C_p^3$ with the highest $\Delta(c,p)$ value. In
                case of a tie, use the tie breaker of Figure 3.
7.      Construct $seq_p$ by sequencing $c_p^*$ in $seq_{p-1}$
8.  End for
9.  Return $seq_N$
```

Figure 2. The pseudo-code of *H2* first step

- If there is a tie, select the advertisement with the highest $\hat{n}_{cp}$ value.
- If there is again a tie, select the advertisement with the highest n_c value.
- Finally, if a tie still occurs, use the lexicographical order.

Figure 3. The tie breaker

```
0.  Given a solution $Sol$ of the first step 1, let $xh_c$ and $xm_c$ indicate the number of
    times that advertisement $c$ has been assigned to a spot type $H$ and to a spot type
    $M$, respectively.
1.  While $\exists_{c=1}^{C} xh_c < nh_c$ do:
2.      Compute $UH = \{c = 1..C : xh_c < nh_c\}$ and $OH = \{c = 1..C : xh_c > nh_c\}$
3.      Compute the best interchange (i.e., the one that least increases the objective
        function) between each pair of spots, $s_1$ and $s_2$, such as:
          i) advertisement $c_1$ assigned to $s_1$ belongs to $UH$ and $s_1$ is of type $M$ or $L$
          ii) advertisement $c_2$ assigned to $s_2$ belongs to $OH$ and $s_2$ is of type $H$
4.      Apply the interchange to $Sol$ and update $xm_{c_1}, xh_{c_1}, xm_{c_2}$ and $xh_{c_2}$.
5.  End while
6.  While $\exists_{c=1}^{C} xm_c + (xh_c - nh_c) < nm_c$ do:
7.      Compute $UM = \{c = 1..C : xm_c + (xh_c - nh_c) < nm_c\}$ and
          $OM = \{c = 1..C : xm_c + (xh_c - nh_c) > nm_c\}$
8.      Compute the best interchange between each pair of spots, $s_1$ and $s_2$ such as:
          i) advertisement $c_1$ assigned to $s_1$ belongs to $UM$ and $s_1$ is of type $L$
          ii) advertisement $c_2$ assigned to $s_2$ belongs to $OM$ and $s_2$ is of type $M$ or
            ($s_2$ is of type $H$ and $xh_{c_2} > nh_{c_2}$)
9.      Apply the interchange to $Sol$ and update $xm_{c_1}, xh_{c_1}, xm_{c_2}$ and $xh_{c_2}$.
10. End while
11. Return $Sol$
```

Figure 4. The pseudo-code of the repair mechanism

```
0.  Set the parameters $t_0$ (initial temperature), $t_f$ (final temperature), $itt$ (number of
    iterations during the temperature remains equal), $\alpha$ (cooling factor)
1.  While (current runtime ≤ maximum runtime) do:
2.      $t := t_0$ and $Sol :=$ $obtainInitialSolution()$
3.      For ($t := t_0; t \geq t_f; t := t.\alpha$ ) do:
4.          For ($i := 0; i < itt ; i := i + 1$) do:
5.              Choose at random a solution $Sol'$ from the neighbourhood $N_{23}$ of $Sol$
6.              $\Delta := Z(Sol') - Z(Sol)$
7.              If $\Delta \leq 0$ then $Sol := Sol'$
8.              If $\Delta > 0$ then $Sol := Sol'$ with probability exp(-Δ/t)
9.          End for
10.     End for
11. End while
12. Return the best solution found
```

Figure 5. The pseudo-code of the common scheme of the SA procedures

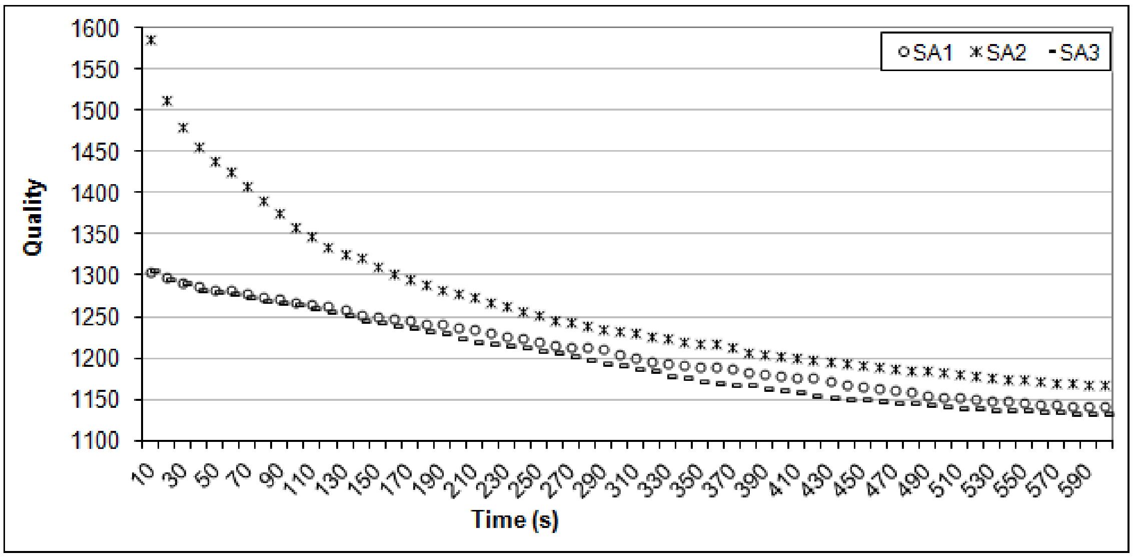

Figure 6. Average value of the best solutions found by the SA algorithms over time.