Soft Car Sequencing With Colors: Lower Bounds and Optimality Proofs

Thierry Benoist

Bouygues e-lab, 32 av. Hoche,
75008 Paris, France
tbenoist@bouygues.com

Abstract. This paper is a study of the car sequencing problem, when feature spacing constraints are soft and colors of vehicles are taken into account. Both pseudo-polynomial algorithms and lower bounds are presented for parts of the problem or family of instances. With this set of lower bounds, we establish the optimality (up to the first non trivial criteria) of 54% of best known solutions for the benchmark used for the Roadef Challenge 2005. We also prove that the optimal penalty for a single ratio constraint N / P can be computed in $\mathcal{O}(P)$ and that determining the feasibility of a car sequencing instance limited to a pair of simple ratio constraints can be achieved by dynamic programming. Finally, we propose a solving algorithm exploiting these results within a local search approach. To achieve this goal, a new meta-heuristic (*star-relinking*) is introduced, designed for the optimization of an aggregation of criteria, when the optimization of each single criterion is a polynomial problem.

1. Introduction

Car sequencing is a well known combinatorial optimization problem, faced by car manufacturers. It consists in ordering cars on the production line, according to features spacing constraints of the form "*never more that 2 sunroofs in any sequence of 5 vehicles*", called *ratio constraints*. These constraints model the necessity for cars requiring special operations to be evenly distributed along the line. In its standard[1] form it is a decision problem, whose NP-completeness is proven in (Gent 1998). Solving algorithm reported in the literature include Constraint Programming (Dincbas et al. 1998), Linear Programming (Gravel et al. 2005) Ant Colonies (Solnon 2000) and Local Search (Puchta & Gottlieb 2002).

Since cars are produced on demand, the set of vehicles for a day cannot necessarily satisfy those ratio constraints. However, cars have to be built even when the corresponding combinatorial problem is infeasible. Indeed minimizing constraints violations means minimizing bottlenecks or minimizing the size of the mobile team (polyvalent workers able to work on different assembly stations). That is the reason why results on infeasible problems are often reported as a number of constraint violations. In this paper we evaluate solutions with a finer penalty function designed by the

[1] Problem #1 of the CSP lib: http://www.cse.unsw.edu.au/~tw/csplib/prob/prob001/index.html

French automobile manufacturer Renault[2]: on each constraint window, each supernumerary feature yields one penalty point. Besides, high priority and low priority constraints are distinguished and denoted HPRC and LPRC respectively.

The second novelty of this problem is that it takes into account the color of each vehicle. Indeed the paint shop is a very special station in the production line, with specific requirements. First, the paint shop has to minimize the consumption of paint solvent. The paint solvent is used to wash spray guns each time the paint color is changed between two consecutive scheduled vehicles, therefore the minimization of color changes is another optimization criteria. On the other hand, long sequences of cars of the same color tend to make visual quality controls inaccurate, thus an upper limit is defined on the length of these color batches.

This *Soft Car Sequencing Problem with Colors,* combining paint changes minimization and two levels of ratio constraints, in an instance-dependant lexicographic order, was proposed by Renault to the OR community for the ROADEF Challenge 2005.

The main contribution of this paper is to produce optimality proofs for a large part of this real-world benchmark. To achieve this goal, this paper proposes an analysis of this problem, defined in section 2, describing optimal algorithms for special cases (section 3) and lower bounds adapted to each family of instances (section 4).

The second contribution of this work is the presentation of a new meta-heuristic (named *star-relinking*), designed for the optimization of an aggregation of criteria, when the optimization of each single criterion is a polynomial problem. This meta-heuristic is introduced and applied to this car sequencing problem in section 5. It proved to be a simple an efficient way to take advantage of optimal algorithms of section 3, reaching the final stage of the Roadef Challenge.

2. Problem definition

2.1. Data and constraints

An instance of the SCSPC (*Soft Car Sequencing Problem with Colors*) is defined by[3]:

- C, the number of colors
- B, the "batch limit", namely the maximum allowed number of consecutive vehicles of identical color.
- R the number of ratio constraints, with $H \subset R$ the subset of high-priority ratios.
- E the number of relevant vehicles of the previous day
- T the number of cars to be produced on the considered day
- $\forall\, j \in [0,R\text{-}1]$ two integers N_j and P_j with $P_j > N_j > 0$ (defining constraints N_j/P_j).
- $\forall\, i \in [-E, T\text{-}1]$, $c_i \in [0,C\text{-}1]$ is the color of car i and f_i is a bitvector of length R (vector of features of car i).
- A permutation of triplet (*paint, high, low*) where high is necessarily before low.

[2] http://www.prism.uvsq.fr/~vdc/ROADEF/CHALLENGES/2005/

[3] In the whole paper, notation [a,b] refers to a closed interval of *integers* (finite set)

A solution of the problem is a permutation σ of [0, *T*-1] such that, among cars from position 0 to *T*-1, the longest sequence of cars of identical color is smaller or equal to *B*. This batch constraint is the only imperative constraint of the problem.

The objective of the problem is to find a permutation σ of minimum valuation. This valuation function is defined in the next section.

2.2. Valuation function

Let φ_t be the vector of features of the vehicle in position *t* in permutation σ, and γ_t its color (defining these values for $t \geq T$ will be convenient when counting violation penalties for the end of the day):

$$\varphi_t = \left\{ f_t \;\; if \;\; t \in [-E,-1], \;\; f_{\sigma^{-1}(t)} \;\; if \;\; t \in [0, T-1], \;\; 0 \;\; if \;\; t \geq T \right\} \quad (1)$$

$$\gamma_t = \left\{ c_t \;\; if \;\; t \in [-E,-1], \;\; c_{\sigma^{-1}(t)} \;\; if \;\; t \in [0, T-1], \;\; 0 \;\; if \;\; t \geq T \right\} \quad (2)$$

When the considered ratio constraint is non ambiguous (say *j*) we will use notation Φ_t when referring to φ_{tj}.

Paint changes are defined as positions $t \in [0,T-1]$ such that the color of car in position *t*-1 is different from the color of the car in position *t* (that is to say $\gamma_{t-1} \neq \gamma_t$). The number of such positions in [0,*T*-1] is the color valuation Ω_{paint}.

Violation penalties aim at measuring the violation of each ratio constraint N_j/P_j. For all *w* in $[-P_j+1, T-1]$ we considerer window $[w, w+P_j-1]$ and count the number of vehicle with option *j* in this window: each supernumerary vehicle (with respect to N_j) costs one point (see **Fig. 1**).

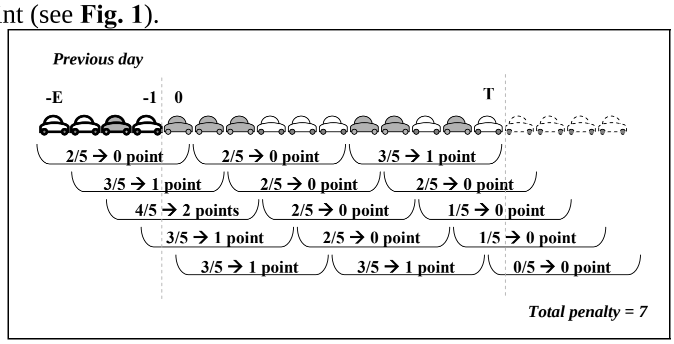


Fig. 1. Penalties computation for a ratio constraint 2/5

In other words ratio valuations Ω_{high} (penalties on high level constraints) and Ω_{low} (penalties on low level constraints) are defined as follows.

$$\Omega_{high} = \sum_{j \in H} \omega_j \,, \quad \Omega_{low} = \sum_{j \notin H} \omega_j \quad \text{with} \quad \omega_j = \sum_{w=-P_j+1}^{T-1} \max(0, \sum_{t=w}^{w+P_j-1} \varphi_{tj} - N_j) \quad (3)$$

The valuation function is a triplet (Ω_{paint}, Ω_{high}, Ω_{low}) that should be minimized with respect to the lexicographical ordering defined by the given permutation of (*paint, high, low*).

Notations: a car featuring option j (f_{ij} = 1) will be called a j-car, and a car without option j (f_{ij} = 0) will be called a j-blank. Besides a set of T cars including Q j-cars (and $T - Q$ j-blanks) will be denoted by $(Q,T)_j$.

3. Pseudo-polynomial sub-problems

In this section we prove that finding the minimal penalty for a single ratio constraint N/P can be achieved in $O(P)$ and that the minimum number of paint changes can be computed in $O(C)$. These results will be applied to our global problem in the following sections: first to compute accurate lower bounds, and then to design a local search strategy.

3.1. The uncoloured single ratio problem

Let us consider the case of a problem restricted to a single ratio constraint N_j/P_j (R=1, C=1, B=T). We denote by Q the total number of cars featuring this option ($Q = \sum_{i\geq 0} f_{i0}$). Any vector Φ of length T and weight Q ($\sum \Phi_t = Q$) is a valid solution. We define $Y_t = \sum_{i=t}^{t+P-1} \Phi_i$ and $Z_t = \max(0, Y_t - N)$ (counter and penalty, respectively). In what follows we prove that inserting j-cars (cars featuring the option) from left to right, avoiding penalties until impossible is an optimal greedy algorithm. The first property states that postponing the insertion of an "acceptable" j-car is never useful. The second property states that anticipating the insertion of a "supernumerary" j-car is never useful.

Property 1 : *If $Q>0$ and ($Q=T$ or $\sum_{i=-P+1}^{-1} \Phi_i < N$) then there is some optimal solution such that : $\Phi_0=1$.*

Proof : The case Q=T is trivially true, so we will focus on the case $Q < T$. Let Φ be an optimal solution with Φ_0=0. We have $Y_{-P+1} < N$. Let t_1=min{t | $t > 0 \wedge \Phi_t$=1} (exists since $Q > 0$). Since $Y_{-P+1} < N$ and Φ_t=0 $\forall\ t \in[0, t_1$-1], we have $Y_{t1-P} < N$. Let Φ' be the solution which differs only from Φ by Φ'_{t1-1}=1 and Φ'_{t1}=0. This solution abides: $Y'_{t1-P} \leq N$ (no penalty) and $Y'_{t1}=Y_{t1}$-1 (no other window is modified). Therefore the valuation of Φ' is smaller or equal to that of Φ, that is to say that Φ' is optimal too. Repeating this operation until Φ_0=1 proves the property. ■

Property 2 : *If Q=0 or ($Q < T$ and $\sum_{i=-P+1}^{-1} \Phi_i \geq N$) then there is some optimal solution such that : Φ_0=0.*

Proof : The case Q=0 is trivially true, so we will focus on the case $Q > 0$. Let Φ be an optimal solution with Φ_0=1. We have $Y_{-P+1} > N$ hence Z_{-P+1}>0. Let t_1 = min{t | $t > 0 \wedge \Phi_t = 0$} (exists since $Q < T$). Since $Y_{-P+1} > N$ and Φ_t=1 $\forall$ t $\in$[0,t_1-1], we have $Y_{t1-P} > N$. Let Φ' be the solution which differs only from Φ by Φ'_{t1-1}=0 and Φ'_{t1}=1. This solution abides $Y'_{t1-P} = Y_{t1-P}$ -1 and $Y'_{t1}=Y_{t1}$+1 (no other window is modified). Since $Y_{t1-P} > N$, this swap decreases penalty associated to window [t_1-P, t_1-1] by one. On the other hand, the increase of the penalty associated to window [t_1, t_1+P-1] cannot exceed one. Therefore the valuation of Φ’

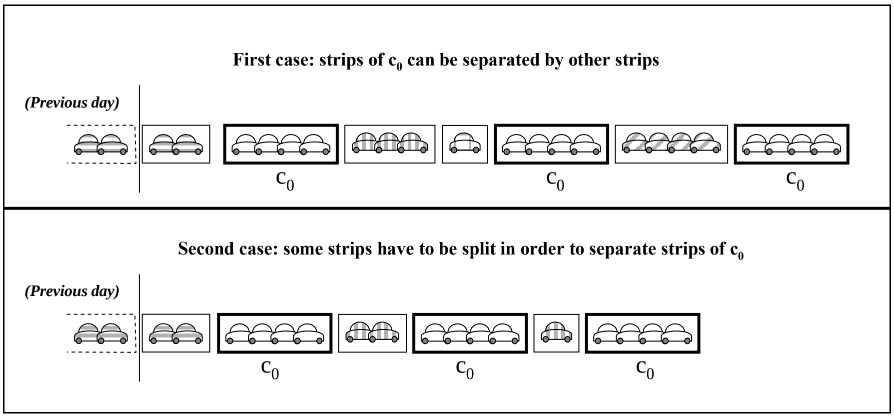


Fig. 2. Paint changes minimization with B = 4

- **First case $\sum_{c \neq c0} s_c \geq s_{c0}$:**
  - Since a color change occurs just before each strip but the first one (provided that we chose color c_{-1} for the first one), a *lower bound* of the number of paint changes is $\sum_c s_c - 1$.
  - This lower bound *can be reached* as follows. We insert one c_{-1} strip first, and one c_0 strip just after (if $c_0 \neq c_{-1}$). Then we have $s_{c0} - 1$ chinks between c_0 strips. Since $\sum_{c \neq c0} s_c \geq s_{c0}$ enough other strips are available to fill these chinks. Besides since $\forall\ c\ s_c \leq s_{c0}$, we can avoid inserting two strips of identical colors in a single chink (when $s_c = s_{c0}$, one c strip can be inserted after the last c_0 strip).
- **Second case $\sum_{c \neq c0} s_c < s_{c0}$:**
  - Two color changes occurs around each c_0 strip but the last one (and the first one if $c_0 = c_{-1}$). Therefore a *lower bound* of the number of paint changes is $2s_{c0} - 2$ if $c_0 = c_{-1}$ and $2s_{c0} - 1$ otherwise.
  - This lower bound *can be reached* as follows. We start and end with a c_0 strip. We have $s_{c0} - 1$ chinks between these c_0 strips. Since $s_{c0} - 1 \geq \sum_{c \neq c0} s_c$, these chinks can be filled using a single color per chink provided that $\sum_{c \neq c0} q_c \geq s_{c0} - 1$ (otherwise the problem is infeasible).

When $q_{c-1} = 0$, the optimum is $\sum_c s_c$ in the first case and $2s_{c0} - 1$ in the second case.

Finally both feasibility and minimum number of paint changes can be determined in $O(C)$.

4. Lower bounds

The benchmark provided by Renault is made of 80 real world instances[7] issued of various factories. We have defined three families of instances:

[7] Involving up to 1500 vehicles and 26 ratio constraints (averages are 760 and 11 respectively)

1. PAINT instances are those whose first optimization criteria is the minimization of color changes (22 instances)
2. EASY instances are those whose best known solution satisfies all ratio constraints (zero penalty) whose importance is greater than color changes minimization (26 instances)
3. Other instances are said to be HARD (32 instances)

In the following sections, each case is considered separately; lower bounds are presented and compared with best known solutions. Throughout this paper, the term “best know solution” will always refer to the best solution found by competitors during the challenge.

4.1. Paint first, ratios second

When the first optimization criterion is paint changes minimization, better lower bounds that those of section 3.1 can be obtained for penalties attached to each ratio constraint. Indeed the known optimality of Ω_{paint} provides upper bounds on the number of strips of each color: in most cases (when $\sum_{c\neq c0} s_c \geq s_{c0}$) these upper bounds τ_c equal lower bounds $s_c = \lceil q_c / B \rceil$ exposed in section 3.2. Then considering penalties within strips of a color can be judicious since features are not evenly dispatched among colors, what can make penalties unavoidable even for features whose global density is low. The client behavior can cause correlations between features and colors: expensive colors may be more frequent on cars with expensive features. Besides, some colors are only available for certain types of cars.

A first lower bound can be derived from lower bound ***PQ – NT - N(P-N)*** defined in section 3.1. Let q_c be the number of cars of color c and q_c^j be the number of such vehicles featuring option *j*. If these q_c cars are split into τ_c strips this lower bound would apply to each of these strips and the sum of these terms is a lower bound of ω_j^c (the penalty for ratio constraint *j within strips of color c*). Finally we have $\omega_j^c \geq Pq_c^j - Nq_c - \tau_c N(P-N)$.

The following property can be used to take into account the trailing sequence, what is very important when using these bound as a guide for a greedy algorithm for instance, evaluating the consequences of the choice of the next color.

Property 3 : *When no batch limit is set, let* $(Q_1,T_1)_j$ ... $(Q_\tau,T_\tau)_j$ *be a dispatch of cars of color* c *into* τ *strips* ($\sum Q_i = q_c^j$ *and* $\sum T_i = q_c$), *minimizing penalties for ratio* j *within strips of color* c, *including penalties caused by inserting* (Q_1,T_1) *just after the trailing sequence. Then there is a sequence of equal cost such that* $\forall\ i \geq 2$ $(Q_i,T_i)_j=(x,x)_j$ *with* $x \leq N$.

Proof. Each strip $(x,y)_j$ with $x \leq N$ can be turned into an $(x,x)_j$ pair just by moving all *j*-blanks from this strip to the first one.

Otherwise, supposing (without loss of generality) that *(x,y)* is arranged in normal form (cf. section 3.1), then the whole sequence following the starting *N* consecutive *j*-cars can be moved as is at the end of the first strip without adding a single penalty:

- if $y–x \geq P–N$, then this displaced sequence starts with *P-N j*-blanks and no penalty can occur on overlapping windows to the left of this sequence.

- if $y–x < P–N$, then each displaced j-car cannot belong to a greater number of saturated windows in its new position. Indeed, with i' the start of the (x,y) strip and i'' the position of the first displaced j-car: all windows starting before i' and including i'' are saturated in their original position, and windows starting after i' cannot become saturated if they are not in their original position (since (x,y) starts with N consecutive j-cars).

All cases are covered, hence the property is proven. ■

Therefore, if c is chosen as the next color we know that ω_j^c will be greater than the minimum cost of inserting $q_c^j–(\tau_c–1)N$ j-cars in a strip of length $q_c–(\tau_c–1)N$, just after the current trailing sequence.

Another way of improving our lower bound is to take into account the trailing limit. It can be done solving the following MIP. Note that this last lower bound is computed in around one second for a problem and thus cannot be used during search as opposed to previous formulas that can be computed during a branch and bound or greedy search.

$$\text{Minimize} \sum_{i=1}^{\tau_c} PQ_i - NT_i - N(P-N) \qquad \textbf{(4)}$$

$$\text{Subject to} \quad \begin{aligned} &\forall i, \quad 0 \le Q_i \le T_i \le B \\ &\sum_{i=1}^{\tau_c} Q_i = q_c^j \\ &\sum_{i=1}^{\tau_c} T_i = q_c \end{aligned} \qquad \textbf{(5)}$$

with Q_i and T_i non negative integers

The following table compares obtained lower bounds (each computed in less than one second) with best known results. For each instance the first letter A, B or X denotes the test set it belongs to in the considered benchmark[8]. The min number of paint changes (PCC) is only reported for information since its optimality is known (cf. section 3.2). Therefore the “first non trivial criterion” is the penalty associated to high-priority ratio constraints (HPRC): the average gap[9] on this criterion is 3.7% with 50% of optimality proofs. Note that solving the above MIP improved the bound on 6 instances and that a “color-unaware” application of bounds of section 3.1 yields an average gap of 85%. The gap attached to low-priority ratio constraints is only reported when it makes sense that is to say when optimality on HPRC is proven.

	Best known solution			Lower bound		Gaps	
instance	**PCC**	**HPRC**	**LPRC**	**HPRC**	**LPRC**	**HPRC**	**LPRC**
A/022_3_4_RAF_EP_ENP	11	**39**	1	**39**	1	**0.0%**	0.0%
A/039_38_4_RAF_EP_ch1	68	**155**	0	**131**	0	**15.5%**	

[8] Available at http://www.prism.uvsq.fr/~vdc/ROADEF/CHALLENGES/2005/
[9] Gaps are computed as (Bestknown – lowerbound)/bestknown

A/064_38_2_RAF_EP_ENP_ch1	63	**423**	782	**413**	725	**2.4%**	
A/064_38_2_RAF_EP_ENP_ch2	27	**367**	52	**367**	49	**0.0%**	5.8%
B/022_RAF_EP_ENP_S22_J1	13	**22**	148	**22**	146	**0.0%**	1.4%
B/023_RAF_EP_ENP_S23_J3	50	**1327**	31	**1264**	15	**4.7%**	
B/024_V2_RAF_EP_ENP_S22_J1	132	**2022**	1158	**1817**	823	**10.1%**	
B/025_RAF_EP_ENP_S22_J3	126	**122**	5589	**122**	5512	**0.0%**	
B/028_ch1_RAF_EP_ENP_S22_J2	38	**98**	188	**85**	177	**13.3%**	
B/028_ch2_RAF_EP_ENP_S23_J3	4	**0**	71	**0**	70	**0.0%**	1.4%
B/029_RAF_EP_ENP_S21_J6	52	**709**	2171	**707**	2150	**0.3%**	
B/035_ch1_RAF_EP_ENP_S22_J3	6	**156**	90	**146**	84	**6.4%**	
B/035_ch2_RAF_EP_ENP_S22_J3	7	**651**	671	**550**	450	**15.5%**	
B/039_ch1_RAF_EP_ENP_S22_J4	55	**45**	96	**45**	83	**0.0%**	13.5%
B/039_ch3_RAF_EP_ENP_S22_J4	59	**214**	671	**214**	663	**0.0%**	1.2%
B/048_ch1_RAF_EP_ENP_S22_J3	64	**115**	670	**115**	643	**0.0%**	
B/048_ch2_RAF_EP_ENP_S22_J3	58	**282**	1180	**279**	1170	**1.1%**	
B/064_ch1_RAF_EP_ENP_S22_J3	62	**95**	288	**84**	266	**11.6%**	
B/064_ch2_RAF_EP_ENP_S22_J4	31	**52**	178	**52**	178	**0.0%**	0.0%
X/022_RAF_EP_ENP_S49_J2	12	**2**	3	**2**	3	**0.0%**	0.0%
X/035_CH1_RAF_EP_S50_J4	5	**10**	0	**10**	0	**0.0%**	
X/035_CH2_RAF_EP_S50_J4	6	**56**	0	**56**	0	**0.0%**	

Table 1. Lower bounds for PAINT instances

4.2. “Easy” ratios first, paint second

For EASY instances the optimization of Ω_{paint} amounts to minimizing the number of paint changes *with ratios*[10] *considered as hard constraints*. It is the opposite situation to that studied in section 4.1. This section describes lower bounds of Ω_{paint} in this case, based on improvements of the computation of the minimum number or strips of each color.

For each color c and for each ratio constraints j we can consider the minimum number of strips of color c consistent with the no penalty constraint $\omega_j = 0$. The linear bound presented in section 4.1 reads $\omega_j \geq \omega_j^c \geq PQ_c^j - NT_c - \tau_c N(P\text{-}N)$ and $\omega_j = 0$ requires $\tau_c \geq (PQ_c^j - NT_c) / N(P\text{-}N)$. This first bound does not take into account the batch limit B, whereas it can have an important impact. For instance we have not only $\tau_c \geq \lceil T_c / B \rceil$ but also $\tau_c \geq \lceil Q_c^j / Q_{\max}(B) \rceil$, since at most $Q_{\max}(B)$ j-cars can fit in each strip. Finally the exact $\tau_c^{\min}$ can be computed as follows.

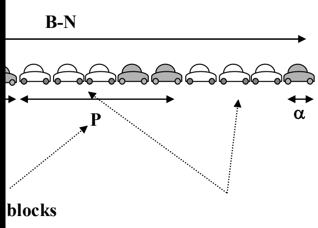
Without loss of generality we can suppose that each strip will be in normal form (alternate sequence of blocks of j-cars separated by $P\text{-}N$ consecutive j-blanks). Consequently the maximum number of j-cars that can fit in τ_c strips only depends on the number of available blank blocks $X = \lfloor (T_c\text{-}Q_c^j) / (P\text{-}N) \rfloor$. In each strip (see **Fig. 3**), N j-cars can fit without any j-blank block[11]; any additional block (up to $\beta = \lfloor (B\text{-}N) / P \rfloor$) allows N additional j-cars; if a last additional block can be added, it allows α additional j-cars with $\alpha = \max(0, B - (N + \beta P - (P\text{-}N))$. Note that if no last additional block can be added then $\alpha = 0$.

---

[10] More precisely: ratios whose importance is greater than Ω_{paint} minimization.

[11] If $N > B$, then the basic bound T_c/B is optimal.

[redacted]



B/025_EP_RAF_ENP_S22_J3	**167**	0	5180	**157**	**6.0%**
B/028_ch2_EP_RAF_ENP_S23_J3	**4**	0	71	**4**	**0.0%**
B/039_ch1_EP_RAF_ENP_S22_J4	**78**	0	89	**78**	**0.0%**
B/039_ch3_EP_ENP_RAF_S22_J4	**197**	0	0	**194**	**1.5%**
B/039_ch3_EP_RAF_ENP_S22_J4	**189**	0	103	**189**	**0.0%**
B/048_ch1_EP_ENP_RAF_S22_J3	**200**	0	0	**193**	**3.5%**
B/048_ch1_EP_RAF_ENP_S22_J3	**161**	0	378	**161**	**0.0%**
B/064_ch1_EP_ENP_RAF_S22_J3	**182**	0	0	**173**	**4.9%**
B/064_ch1_EP_RAF_ENP_S22_J3	**130**	0	187	**130**	**0.0%**
B/064_ch2_EP_RAF_ENP_S22_J4	**130**	0	69	**127**	**2.3%**
X/023_EP_RAF_ENP_S49_J2	**192**	0	66	**176**	**8.3%**
X/024_EP_RAF_ENP_S49_J2	**337**	0	6	**298**	**11.6%**
X/028_CH2_EP_ENP_RAF_S51_J1	**3**	0	0	**3**	**0.0%**
X/029_EP_RAF_ENP_S49_J5	**110**	0	98	**109**	**0.9%**
X/034_VP_EP_RAF_ENP_S51_J1_J2_J3	**55**	0	794	**54**	**1.8%**
X/039_CH1_EP_RAF_ENP_S49_J1	**69**	0	239	**69**	**0.0%**
X/039_CH3_EP_RAF_ENP_S49_J1	**231**	0	30	**231**	**0.0%**
X/048_CH1_EP_RAF_ENP_S50_J4	**196**	0	1005	**192**	**2.0%**
X/064_CH2_EP_RAF_ENP_S49_J4	**37**	0	0	**37**	**0.0%**
X/655_CH1_EP_RAF_ENP_S51_J2_J3_J4	**30**	0	0	**30**	**0.0%**

Table 2. Lower bounds for EASY instances

4.3. "Hard" ratios

In the typology defined page 6, HARD instances are those for which no known solution manages to strictly satisfy all ratios constraint of greater importance than Ω_{paint}.

For these problems, section 3.1 provides a first lower bound, computed as the sum of minimal penalties for each single ratio constraint. This simple sum proves the optimality of the best known Ω_{high} (or Ω_{low} when best Ω_{high} is 0) on 17 instances out of 32. Four additional optimality proofs (and three lower bounds improvements) can be obtained with a linear model of the problem. This classical model (Gravel et al. 2005, Estellon et al. 2005) is based on the identification of families of vehicles: two vehicles belong to the same family if they share exactly the same options (ignoring low priority ratio constraint when focusing on Ω_{high}). Within this formalism, a solution is an assignment of a unique family to each position (binary variables), and the definition of penalty counters is straightforward, using intermediate[13] binary variables detecting features on each position. We refer the reader to papers cited above for an exact formulation of this linear program. For large scale problems like those in this benchmark, only the continuous relaxation of this model can be solved (in a few seconds). Nonetheless, this relaxation is never smaller than the sum of minimal penalties because, as proven in appendix, for each single ratio it has the convex hull property.

With this linear program, the strict feasibility of some problems remains undetermined that is to say that the obtained lower bound is 0. In this section we focus on pairs of ratio with a dynamic programming approach and finally prove that ratio constraints cannot be strictly satisfied for these instances.

[13] Interestingly, (Estellon et al. 2005) point out that requiring the integrality of these intermediate variables is sufficient to ensure the integrality of all binary variables of the model.

Let us define a *simple* ratio constraint as a ratio constraint N/P with N=1 or N=P-1, which is the case for 95% of ratio constraints in the considered benchmark. For a pair of such simple ratios N_1/P_1 and N_2/P_2, we consider the problem of placing n cars with both features, n_1 cars with feature 1 and n_2 cars with feature 2, in a strip of length T, without any penalty. This problem can be solved with a dynamic program based on a 5-dimensional state $(\nu,\nu_1,\nu_2,\tau_1,\tau_2)$ where ν,ν_1,ν_2 are counters initialized to (n,n_1,n_2) and τ_1,τ_2 code for the minimum number of blanks to be inserted before placing a car with the feature (for 1/P ratios) or for the maximum number of vehicles that can be inserted before placing a blank (for P-1/P ratios). Initial values t_1, t_2 of τ_1,τ_2 depend on vehicles of the previous day. With Λ recursively defined by equations **(8)**, **(9)**, **(10)**, the minimum required length for placing (n,n_1,n_2) without penalty is $\Lambda(n,n_1,n_2,t_1,t_2)$.

Property 4 : *The difference between* $\Lambda(n,n_1,n_2,t_1,t_2)$ *and the available length (total number of vehicles) is a lower bound of the penalty for ratios* N_1/P_1 *and* N_2/P_2.

Proof. Given an optimal solution with a total penalty equal to x, there is necessarily a ratio N_j/P_j and a window of length smaller than P_j with at least N+1 j-cars, starting and ending with a j-car. Inserting a j-blank just before the rightmost car of this window decreases violation in this window without increasing penalties anywhere. In other words adding one j-blank can at least decrease penalty by one. Finally if the optimal solution with length T has a penalty equal to x, then the minimum length without violation is at least T+x. ■

Condition for the insertion of a car with feature 1:

$$F(\tau_1) = TRUE \ \ if \ \ (N_1 = 1 \wedge \tau_1 = 0) \vee (N_1 = P-1 \wedge \tau_1 > 0), \ \ FALSE \ \ otherwise$$

Transition rules for τ_1:

$$f(\tau_1) = \begin{cases} P_1 - 1, & if \ \ N_1 = 1 \\ \max(0, \tau_1 - 1), & if \ \ N_1 = P_1 - 1 \end{cases} \quad \text{(insertion of a car with feature 1)} \tag{8}$$

$$b(\tau_1) = \begin{cases} \max(0, \tau_1 - 1), & if \ \ N_1 = 1 \\ P_1 - 1, & if \ \ N_1 = P_1 - 1 \end{cases} \quad \text{(insertion of a car without feature 1)}$$

***F'*,*f'* and *b'* are the equivalent functions for ratio N_2/P_2**

$$\forall \tau_1, \tau_2, \ \ \Lambda(0,0,0,\tau_1,\tau_2) = 0 \tag{9}$$

$$\Lambda(\nu,\nu_1,\nu_2,\tau_1,\tau_2) = 1 + \min \begin{cases} \Lambda(\nu-1,\nu_1,\nu_2,f(\tau_1),f'(\tau_2)), & provided \ \ that: \nu > 0 \wedge F(\tau_1) \wedge F'(\tau_2) \\ \Lambda(\nu,\nu_1-1,\nu_2,f(\tau_1),b'(\tau_2)), & provided \ \ that: \nu_1 > 0 \wedge F(\tau_1) \\ \Lambda(\nu,\nu_1,\nu_2-1,b(\tau_1),f'(\tau_2)), & provided \ \ that: \nu_2 > 0 \wedge F'(\tau_2) \\ \Lambda(\nu,\nu_1,\nu_2,b(\tau_1),b'(\tau_2)), & provided \ \ that: \neg(F(\tau_1) \wedge F'(\tau_2)) \end{cases} \tag{10}$$

The time complexity of this algorithm is $4nn_1n_2P_1P_2$. As for space complexity it can be bounded by $2nn_1n_2P_1P_2/\max(n,n_1,n_2)$ using an appropriate evaluation ordering, such that states $(\nu,.,.,.,.)$ can be garbage collected before reaching states $(\nu$-2,.,.,.,.)

for instance. In practice, when N_1=1 we have $n_1P_1<T$ (otherwise infeasibility is sure) and it turns out that $P\leq10$ for all (P-1)/P ratios. Finally complexity can be approximated to O(T^3) in our benchmark and computation time never exceeds 90 seconds on a 3GHz PC.

As detailed in the following table, we obtain an average gap of 11% with 72% of optimality proof on HPRC (or LPRC when best known HPRC is 0). On most problems, no better bound than the sum of individual minimal penalties is reported and this simple bound is optimal for 19 instances.

- On the contrary lower bounds for instances with prefix B/035_ are one point larger (and optimal) when solving the linear program.
- All four first instances (with prefix A/024_) have no better bound than 0 when dynamic programming is not enabled
- Finally the lower bound 9 for instance A/039_38_4_EP_RAF_ch1 is obtained using a Russian Dolls Strategy (Benoist & Rottembourg 2004), whilst pure LP yields lower bound 7. Lower bounds on pairs of ratios are computed[14] first (in particular the minimum penalty for the second and third ratios cannot be smaller than 9) and the global continuous relaxation is solved, taking this cut into account. This strategy gave no result on other instances.

	Best known solution			**Lower bound**		**Gaps**	
instance	**PCC**	**HPRC**	**LPRC**	**HPRC**	**LPRC**	**HPRC**	**LPRC**
A/024_38_3_EP_ENP_RAF	302	4	0	**2**	0	50%	
A/024_38_3_EP_RAF_ENP	249	4	83	**2**	0	50%	
A/024_38_5_EP_ENP_RAF	309	4	34	**2**	1	50%	
A/024_38_5_EP_RAF_ENP	280	4	79	**2**	1	50%	
A/025_38_1_EP_ENP_RAF	720	0	99	0	70	-_	29%
A/039_38_4_EP_RAF_ch1	129	13	0	**9**	0	31%	
A/048_39_1_EP_ENP_RAF	290	0	61	0	3	-_	95%
B/022_EP_ENP_RAF_S22_J1	109	0	3	0	3	-_	**0%**
B/023_EP_ENP_RAF_S23_J3	316	48	0	48	0	**0%**	
B/023_EP_RAF_ENP_S23_J3	310	48	8	48	0	**0%**	
B/024_V2_EP_ENP_RAF_S22_J1	430	1074	850	973	754	9%	
B/024_V2_EP_RAF_ENP_S22_J1	298	1074	1068	973	754	9%	
B/025_EP_ENP_RAF_S22_J3	479	0	3912	0	3912	-_	**0%**
B/028_ch1_EP_ENP_RAF_S22_J2	76	54	3	54	3	**0%**	
B/028_ch1_EP_RAF_ENP_S22_J2	49	54	124	54	3	**0%**	
B/028_ch2_EP_ENP_RAF_S23_J3	6	0	70	0	70	-_	**0%**
B/029_EP_ENP_RAF_S21_J6	167	35	2150	35	2150	**0%**	
B/029_EP_RAF_ENP_S21_J6	165	35	2170	35	2150	**0%**	
B/035_ch1_EP_ENP_RAF_S22_J3	49	67	52	**67**	50	**0%**	
B/035_ch1_EP_RAF_ENP_S22_J3	36	67	61	**67**	50	**0%**	
B/035_ch2_EP_ENP_RAF_S22_J3	205	385	341	**385**	338	**0%**	
B/035_ch2_EP_RAF_ENP_S22_J3	187	385	351	**385**	338	**0%**	

[14] These bounds are optimal integer solution (obtained in 3 hours by a complete tree search) of the MIP restricted to a pair of ratios, where the number of integrality constraints is controlled as follows: "$\forall t\in[T_1, T_2]$, $X_t\in\{0,1\}$" (for some variables X_t) is replaced by the approximation "$\Sigma_{t\in[T1, T2]}X_t \in \{0, 1, 2, \ldots. T_2\text{-}T_1\text{+}1\}$", with 32 [$T_1$, T_2] intervals by ratio constraint.

B/039_ch1_EP_ENP_RAF_S22_J4	117	0	29	0	29	**-_**	**0%**
B/048_ch2_EP_ENP_RAF_S22_J3	337	3	0	3	0	**0%**	
B/048_ch2_EP_RAF_ENP_S22_J3	93	3	1029	3	0	**0%**	
B/064_ch2_EP_ENP_RAF_S22_J4	130	0	69	0	69	**-_**	**0%**
X/025_EP_ENP_RAF_S49_J1	407	0	160	0	160	**-_**	**0%**
X/028_CH1_EP_ENP_RAF_S50_J4	95	36	341	36	278	**0%**	
X/034_VU_EP_RAF_ENP_S51_J1_J2_J3	87	8	35	8	23	**0%**	
X/048_CH2_EP_RAF_ENP_S49_J5	76	31	1116	31	1	**0%**	
X/064_CH1_EP_RAF_ENP_S49_J1	187	61	29	61	0	**0%**	
X/655_CH2_EP_RAF_ENP_S52_J1_J2_S01_J1	34	153	0	153	0	**0%**	

Table 3. Lower bounds for HARD instances

5. Local Search and Star Relinking

In this section we use our analysis of the problem (section 3 and 4) in order to design an efficient solving algorithm for this *Car Sequencing* problem. First we define a chronological constructive algorithm, based on the impact of each decision on the different lower bounds. Then we use the efficient algorithms identified in Section 3 as guides in a local search strategy. The resulting meta-heuristic can be generalized to combinatorial problems whose objective function is an aggregation of polynomial sub-objectives.

5.1. Greedy algorithm and local search

Using exact algorithms of section 3 and lower bounds of section 4, we can design a greedy algorithm. This chronological constructive approach consists in always selecting the next vehicle so as to minimize the current lower bound of the objective function, breaking ties with two secondary criteria: a first one inciting to insert as many car options as possible (never postponing without reason) and a second one avoiding changing the current color. This first solution can be improved looking for pairs of vehicles whose respective positions can be profitably exchanged. The impact of this kind of swap can be computed incrementally: the only involved penalties are those associated to options featured by exactly one of both vehicles and to windows covering exactly one of both positions. Besides updating the color change counter only requires considering previous and next vehicles. Finally the batch limit constraint can be checked in *O(B)* but this test is only required when a pair of vehicle is about to be selected as the best improving one. A descending exploration of this neighborhood yields significant improvements, even if this first local search approach cannot escape from local optima or plateaus.

Trying to escape from plateaus with random horizontal[15] moves turned out to be time consuming. On the contrary we found out that using our pseudo polynomial of section 3.1 to find a sequence of moves improving one ratio constraint without degrading others was more efficient. Following up this idea we decided to generalize this intensification principle to the case of local optima. Finally we obtained a new meta-heuristic similar to Tabu (Glover 1986) taking advantage of the fact that opti-

[15] Moves with no impact on the objective function

mizing any single part of the objective function (a single ratio) is a polynomial problem. In the next subsection, we describe this strategy in the general case before applying it to Soft Car Sequencing in section 5.3.

5.2. Star Relinking

Let P be a combinatorial optimization problem in the form $min\{f(x) \mid x \in X\}$, with X a finite set and f a function on X taking value in an ordered set Y. We suppose that f is a combination of n "sub-objectives" $f_1, f_2, \ldots. f_n$ that is to say that $\forall\ x \in X$, $f(x) = \oplus(f_1(x), f_2(x), \ldots. f_n(x))$ with $\oplus$ a non decreasing function from Y^n to Y, that is to say that for any vector a and b of Y^n, if $\forall\ i,\ b_i \geq a_i$ then $\oplus(b) \geq \oplus(a)$.

For each i, we also suppose that minimizing f_i alone is a polynomial problem, and we define $\Delta_i(x) = \{x' \in X \mid f_i(x') < f_i(x)\}$. If x is not a minimum of f_i then $\Delta_i(x) \neq \emptyset$. With δ_i a function[16] in X such that $\delta_i(x) \in \Delta_i(x)$ and x_0 a starting point in X, our Star Relinking procedure reads as follows.

```
starRelinking(x0) →
S := {x0}                                  // list of candidates
T := ∅                                     // tabu list
while S ≠ ∅                                // (in practice a time limit is defined)
   let y := argmin_S f in                  // selection of the best candidate
   | T := T ∪ {y}                          // increase the tabu list
   | S := S – {x}                          // remove x from the list of candidates
   | for i in [1,n]                        // scan all improvement directions
   |    let x := y in
   |       | while Δi(x) ≠∅                // while x is not optimal for fi
   |       |    x := δi(x)                 // move to a better x w.r.t. fi
   └       └    if x∉T then S := S ∪ {x}   // save x as a new candidate
```

Fig. 4. Star Relinking algorithm

This local search meta-heuristic consists in exploring (from a starting point x_0) n paths corresponding to the n components of the global objective. Each path i can be seen as a relinking (Glover 2000) from the current central point to a minimum of f_i. This approach can also be compared to RINS (Danna et al. 2005) where the current continuous relaxation (optimal w.r.t. the objective function) and the incumbent solution (perfect w.r.t. integrality constraints) are combined[17].

Intermediate points encountered on these paths are kept in a candidate list, and the best candidate with respect to the global objective f will be the next central point from which n paths will be explored. The goal of this approach is to take advantage of the polynomiality of sub-objectives. This property ensure the existence of a polynomial function δ_i but the choice of this selection function is critical for the efficiency of the method. It must return a value as close as possible to $argmin_{\Delta i(x)} f$ that is to say that $f(\delta_i(x))$ must be as small as possible. Indeed this movement should decrease f_i without causing a too large increase of other sub-objectives. The next section describes the

---

[16] δ_i is defined on $\{x \in X \mid \Delta_i(x) \neq \emptyset\}$ and takes values in X.

[17] However, RINS is a pure MIP approach where the combination of both solutions is not based on local moves.

choices of selection function δ_i (and the management of the tabu list) for our car sequencing problem.

5.3. Selection function , tabu list and results

In our case *X* is the set of a permutations of [0, *T*-1], and each ratio constraint defines a polynomial sub-objective. The number of color changes is another sub-objective but in what follows we will consider that this number is either a constraint or completely ignored (depending of the ordering of optimization criteria). With ***N*** the set of natural numbers, the valuation set is $\boldsymbol{N}^3$ with the lexicographic ordering (cf. section 2.2) and the aggregation function ⊕ is the addition of penalties (each ratio constraint contributing to its level).

Let σ_0 be a permutation of [0, *T*-1]. We define $lb_i(\sigma_0, t)$ as the minimum penalty for ratio *i* under the constraint $\forall\ t' < t,\ \sigma^{-1}(t') = \sigma_0^{-1}(t')$. This function can be computed in *O(P)* as explained in section 3.1, taking cars placed in positions $t' < t$ for granted. In particular $lb_i(\sigma_0, 0) = \omega_i{}^*$ (optimal solution for this ratio) and $lb_i(\sigma_0, T) = \omega_i(\sigma_0)$ (penalty for ratio *i* in solution σ_0). If σ_0 is not optimal for ratio *i,* then there is at least one "breaking point" *t* such that $lb_i(\sigma_0, t+1) > lb_i(\sigma_0, t)$ and a non empty set of transpositions $\tau = (t,t')$ with $t'>t$ (exchange of car in position *t* with a car in a position $t'>t$) such that $lb_i(\tau\sigma_0, t+1) = lb_i(\sigma_0, t)$. We can select in this set the transposition minimizing $\omega_i(\tau\sigma_0)$. We call this exchange a repairing move for breaking point *t*.

Scanning σ_0 from left to right and applying all repairing moves leads to an optimal permutation for ratio *i*. Our selection function δ_i is based on this principle, with three improvements:

1. we stop this procedure as soon as the resulting permutation σ satisfies $\omega_i(\sigma) < \omega_i(\sigma_0)$
2. we defined a maximum increase ε for the global objective and all transpositions leading to such an increase are ignored. However if ε is too small, many breaking points become non repairable, therefore we repeat this process with increasing values of ε, starting with ε=0 (in order to detect "pure" improvements if possible) and ending with ε=+∞ (in order to always find a permutation improving ω_i)
3. the resulting permutation σ is improved by the local search descent defined in section 5.1, under the constraint $\omega_i(\sigma) < \omega_i(\sigma_0)$.

As for the tabu list, since a solution is a trade off between different ratio constraints, it stores penalty vectors of permutation $\omega(\sigma)$ instead of storing permutation themselves in order to ensure better diversification. The results of this star relinking local search are presented in **Table 4** (details on base A and B can be found on http://www.prism.uvsq.fr/~vdc/ROADEF/CHALLENGES/2005/final_results.html).

Instance	**Star-relinking solution**	**Best known solution**	**Gap**
X/022_RAF_EP_ENP_S49_J2	12 002 003	12 002 003	0.0%
X/023_EP_RAF_ENP_S49_J2	204 896	192 466	6.5%
X/024_EP_RAF_ENP_S49_J2	27 046 418	337 006	∞
X/025_EP_ENP_RAF_S49_J1	188 119	160 408	17.3%
X/028_CH1_EP_ENP_RAF_S50_J4	43 400 089	36 341 495	19.4%

X/028_CH2_EP_ENP_RAF_S51_J1	3	3	0.0%
X/029_EP_RAF_ENP_S49_J5	112 029	110 298	1.6%
X/034_VP_EP_RAF_ENP_S51_J1_J2_J3	55 995	55 995	0.0%
X/034_VU_EP_RAF_ENP_S51_J1_J2_J3	8 097 049	8 087 036	0.1%
X/035_CH1_RAF_EP_S50_J4	5 010 000	5 010 000	0.0%
X/035_CH2_RAF_EP_S50_J4	6 056 000	6 056 000	0.0%
X/039_CH1_EP_RAF_ENP_S49_J1	69 441	69 239	0.3%
X/039_CH3_EP_RAF_ENP_S49_J1	232 343	231 030	0.6%
X/048_CH1_EP_RAF_ENP_S50_J4	202 946	197 006	3.0%
X/048_CH2_EP_RAF_ENP_S49_J5	31 117 940	31 077 916	0.1%
X/064_CH1_EP_RAF_ENP_S49_J1	61 258 070	61 187 230	0.1%
X/064_CH2_EP_RAF_ENP_S49_J4	37 000	37 000	0.0%
X/655_CH1_EP_RAF_ENP_S51_J2_J3_J4	30 003	30 000	0.0%
X/655_CH2_EP_RAF_ENP_S52_J1_J2_S01_J1	280 033 000	153 034 000	83.0%
Total on base X	**Average gap :7.3%, Mean gap: 0.13%**		
Total on base A	**Average gap :20.4%, Mean gap: 1.62%**		
Total on base B	**Average gap :2.2%, Mean gap: 0.02%**		

Table 4. Star-relinking results

Obtained solutions are in average 7% above best known values and this algorithm ranked 12 among 55 in the Roadef Challenge. In other words it produces good solution but cannot compete with fast local search approaches like those developed by (Estellon et al. 2005) or (Ribeiro & et al. 2005). It may suggest that in this problem the choice of the next movement is not worth spending too much time, even with a polynomial algorithm.

However we are still convinced that this Star Relinking meta-heuristic could be appropriate for other problems.

6. Conclusion

In this paper we have studied *Soft Car Sequencing With Colors*. After having identified polynomial algorithms for the minimization of each sub-objective (color changes or penalties of a single ratio constraints), we have computed lower bounds for three families of instances.

- When paint changes minimization is the first criteria, we used the dispatch of car features among colors to obtain lower bounds of penalties attached to each ratio constraint.
- On the contrary when only “easy” ratio constraints are more important than colors, we found three simple lower bounds for the minimum number of strips for each color.
- Finally when the first criteria is made of “hard” ratios constraints, we reused the classical LP formulation of the problem, explained the quality of its relaxation by a polyhedral argument (see appendix) and detected infeasibility of some pairs of ratio constraints by dynamic programming.

In summary we have proven that more than 50% of the instances of the considered benchmark are closed with respect to their first non trivial criteria.

In addition to this lower bounds we have proposed a new meta heuristic named *Star Relinking,* yielding interesting results on this problem and which may apply on others problems whose objective is an aggregation of polynomial sub-objectives.

Appendix

Analytical solution for the single ratio problem when E=0 (no-trail case)

With $\eta = \lfloor(T-Q)/(P-N)\rfloor$, $Q' = Q-\eta N$, $T' = T-\eta P$, the minimum penalty is:

- When $T' \leq P$: $(Q'-N)(P-T'+1) + (Q'-N)(Q'-N-1) + (T'-Q')\max(0, Q'-2N)$
- When $Q' = T'$: $(H-N)(|Q'-P|+1) + (H-N)(H-N-1)$ with $H = \min(P,Q')$
- When $Q' < P + N$:
  - When $P < T' < P + N$:
    $(T-P+1)(P-(T'-Q')-N) + (T'-Q')\max(0, Q'-2N) +$
    $(P-(T'-Q')-N)(P-(T'-Q')-N-1)$
  - When $T' = P + N$: $(T'-Q'+1+N)(Q'-2N) + (Q'-2N)(Q'-2N-1)$
  - When $P+N < T'$:
    $(T'-Q'+1)(Q'-2N) + N(P-(T'-Q')-N) + (Q'-2N)(Q'-2N-1)$
- Otherwise : $PQ' - NT' - N(P-N)$

The classical LP formulation is totally unimodular for the single ratio problem

For the single ratio problem, the linear program evoked in 4.3 takes a simple form (**11**), with X_t=1 if and only if the car in position t has the feature. For $t \notin [1,T]$, X_t is not a variable but a data, determined by the previous day for t<1 and equal to 0 for t>T. The change of variable leading to system (12), similar to that of (Bartholdi et. Al 1980), shows that the matrix is totally unimodular (dual of a flow problem[18]).

$$\min \sum_{t=-P}^{T+P} Z_t \quad \text{such that} \quad \forall t, Z_t \geq \sum_{i=t-P+1}^{t} X_i \quad \text{and} \quad \sum_{t=-P}^{T} X_t = q \qquad \textbf{(11)}$$

With $Y_t = \sum_{i=-P}^{t} X_i$ constraints can be rewritten as $Y_T = q$ and

$$\forall t \begin{cases} Z_t \geq Y_t - Y_{t-P} \\ Y_t - Y_{t-1} \leq 1 \\ Y_{t-1} - Y_t \leq 0 \end{cases} \qquad (12)$$

References

J.J. Bartholdi, J.B. Orlin, and H.D. Ratliff (1980), *Cyclic scheduling via integer programs with circular ones.* Operations Research, 28:110–118.

T. Benoist and B. Rottembourg (2004), *Upper bounds for revenue maximization in a satellite scheduling problem*, 4OR: A Quarterly Journal of Operations Research, Volume 2, Issue 3, Oct 2004, Pages 235 - 249

E. Danna, E. Rothberg, C. Le Pape (2005), *Exploring relaxation induced neighborhoods to improve MIP solutions*, Mathematical Programming, Vol. 102, Issue 1. pp 71 – 90.

[18] Notice that once transposed, Z columns become unary constraints (capacities in the flow)

M. Dincbas, H. Simonis, and P. van Hentenryck (1988). *Solving the car-sequencing problem in constraint logic programming.* In Y. Kodratoff, editor, Proceedings ECAI-88, pp. 290—295.
B. Estellon, F. Gardi, K. Nouioua (2005), *Ordonnancement de véhicules: une approche par recherche locale à grand voisinage*. In Journées Françaises de Programmation par Contraintes, Angers.
I. Gent (1998), *Two Results on Car-sequencing Problems.* APES Report APES-02-1998: April 1998. Available from http://www.dcs.st-and.ac.uk/~apes/apesreports.html
F. Glover (1986), *Future paths for integer programming and links to artificial intelligence.* Computers and Operations Research, 13, 533-549.
F. Glover, M. Laguna, and R. Martl (2000), *Fundamentals of scatter search and path relinking.* Control and Cybernetics, 39:653-684.
M. Gravel, C. Gagné et W.L. Price (2005), *Review and comparison of three methods for the solution of the car sequencing problem*. Journal of the operational research society, 56: 1287-1295.
M. Puchta et J. Gottlieb (2002). *Solving car sequencing problems by local optimization.* In Applications of Evolutionary Computing, LNCS 2279,p. 132 142. Springer-Verlag.
C. Ribeiro, D. Aloise, T. Noronha,C. Rocha, S. Urrutia (2005), *A heuristic for a real-life car sequencing problem with multiple requirements*. In MIC 2005, Vienna.
C. Solnon (2000), *Solving Permutation Constraint Satisfaction Problems with Artificial Ants*. In Proceedings of 14th European Conference on Artificial Intelligence (ECAI 2000), 118 - 122, IOS Press, 2000.