The car sequencing problem: overview of state-of-the-art methods and industrial case-study of the ROADEF'2005 challenge problem

Christine Solnon, Van-Dat Cung, Alain Nguyen, Christian Artigues

The car sequencing problem: overview of state-of-the-art methods and industrial case-study of the ROADEF'2005 challenge problem

Christine Solnon[1], Van Dat Cung[2], Alain Nguyen[3] and Christian Artigues[4]

[1]LIRIS CNRS UMR 5205, University Lyon I
Nautibus, 43 Bd du 11 novembre, 69622 Villeurbanne cedex, France

[2]GILCO, ENSGI-INPG,
46, avenue Félix-Viallet, 38031 Grenoble Cedex 1, France.

[3]RENAULT – RENAULT - Optimization Group for Sales and Logistics, IT department
Renault Novadis, 13 avenue Paul LANGEVIN, 92359 Le Plessis Robinson, France.

[4]Université de Toulouse, LAAS-CNRS
31077 Toulouse cedex 4, France

christine.solnon@liris.cnrs.fr, Van-Dat.Cung@gilco.inpg.fr, alain.nguyen@renault.com,
christian.artigues@laas.fr

Abstract

The ROADEF challenge is organized every two years by the French Society of Operations Research and Decision-Making Aid. The goal is to allow industrial partners to witness recent developments in the field of Operations Research and Decision Analysis, and researchers to face up a decisional problem, often complex, occurred in industry.

In 2005, the subject of this challenge has been proposed by the car manufacturer RENAULT and concerned a car sequencing problem. This problem involves scheduling cars along an assembly line with hard and soft capacity constraints. The industrial problem considered in the challenge differs from the standard problem since, besides capacity constraints imposed by the assembly shop, it also introduces paint batching constraints to minimize the consumption of solvents in the paint shop.

We reviews the exact and heuristic methods of the literature proposed to solve the standard problem and we present the industrial context and the specificities of the challenge problem. We describe the process of the ROADEF'2005 challenge and the methods proposed by the competing teams. We also analyse the results of these methods on the car sequencing instances provided by RENAULT. The final ranking of the candidates is reported and directions for future research based on the results are drawn.

1 Introduction

The ROADEF challenge is organized every two years by the French Society of Operations Research and Decision-Making Aid. The goal is to allow industrial partners to witness recent developments in the field of Operations Research and Decision Analysis, and researchers to face up a decisional problem, often complex, occurred in industry.

In 2005, the subject of this challenge has been proposed by the car manufacturer RENAULT. It concerned a car sequencing problem which is solved since 1993 in the RENAULT factories by an optimization in-house software based on simulated annealing. However, following the

strategic choices of the company to enforce an absolute respect of the precomputed baseline schedule, this problem has become particularly crucial these last years.

Outline of the paper. The problem considered in the challenge is a special case of a classical car sequencing problem, the goal of which is to schedule cars along an assembly line while satisfying capacity constraints. Section 2 defines this standard problem and discusses complexity issues.

This standard problem has been widely studied since its first introduction in 1986, and it is a classical benchmark problem. Section 3 describes exact approaches, that solve the problem to optimality, while Section 4 describes heuristic approaches, that solve the problem approximately.

The industrial problem considered in the challenge differs from the standard problem since, besides capacity constraints imposed by the assembly shop, it also introduces paint batching constraints to minimize the consumption of solvents in the paint shop. Section 5 describes the industrial context of the problem and defines it as a lexicographic multiobjective optimization problem.

Twenty-five teams (a record of participation !), coming from all over the world, have competed for solving this problem. Section 6 describes the challenge process —that allows the organizers to decide between the teams— and gives an overview of the different proposed methods and their results.

Concluding remarks and directions for further research are given in Section 7.

2 Definitions and complexity issues

The car sequencing problem has been first described by Parello et al. in 1986 [PKW86]. This problem involves scheduling cars along an assembly line, in order to install options (e.g., sunroof, radio, or air-conditioning) on them. Each option is installed by a different station, designed to handle at most a certain percentage of the cars passing along the assembly line, and the cars requiring this option must be spaced such that the capacity of every station is never exceeded.

This requirement may be formalized by p/q ratio constraints: each option is associated with a p/q ratio constraint that states that any subsequence of q vehicules must comprise at most p vehicules requiring this option.

Definition of a problem instance. An instance of the car sequencing problem is defined by a tuple (V, O, p, q, r), where

- $V = \{v_1, .., v_n\}$ is the set of vehicules to be produced;
- $O = \{o_1, .., o_m\}$ is the set of different options;
- $p : O \rightarrow \mathbb{N}$ and $q : O \rightarrow \mathbb{N}$ define the capacity constraint associated with each option $o_i \in O$; this capacity constraint imposes that, for any subsequence of q_i consecutive cars on the line, at most p_i of them may require o_i;
- $r : V \times O \rightarrow \{0, 1\}$ defines options requirements, i.e., for each vehicule $v_i \in V$ and for each option $o_j \in O$, $r_{ij} = 1$ if o_j must be installed on v_i, and $r_{ij} = 0$ otherwise.

Note that two different cars of V may require a same configuration of options; all cars requiring a same configuration of options are clustered into a same *car class*.

Solution of a problem instance. The car sequencing problem involves finding an arrangement of the vehicules of V in a sequence, thus defining the order in which they will pass along the assembly line. The *decision problem* consists in deciding whether it is possible to find a sequence that satisfies all capacity constraints, whereas the *optimization problem* involves finding a minimum cost sequence, where the cost function evaluates constraint violations.

In the original formulation of the problem proposed in [PKW86], the cost depends on the options concerned by constraint violations, the number of vehicules exceeding capacity constraints and how close these vehicules are sequenced together. However, in most work, this cost function has been simplified and is defined as the sum, for every option $o_i \in O$, of the number of subsequences of q_i consecutive cars such that the number of cars requiring option o_i in the subsequence is greater than the capacity p_i of the option. A still different definition of the cost of a sequence has been proposed in [HP94, PS04, PSF04], where it is defined as the number of "empty" cars (requiring no option) that should be inserted in the sequence so that all capacity constraints are met.

Complexity issues. The decision problem has been shown to be NP-hard by Gent [Gen98], by a transformation from the Hamiltonian path problem, and by Kis [Kis04], by a transformation from the exact cover by 3-sets problem. Kis also shown that the problem is NP-hard in the strong sense and does not belong to NP in the general case. This comes from the fact that an instance of the car sequencing problem may be encoded by giving, for each different car class, the number of cars belonging to this class and the subset of options required by these cars. With such an encoding, and provided that the number of cars within a same car class can be arbritrarily large, the length of a solution (which is a sequence of $|V|$ car classes) is not bounded by a polynomial in the length of the instance.

Search space and utilization rates. The search space of an instance (V, O, p, q, r) of the car sequencing problem is composed of all possible permutations of the vehicules of V requiring different configurations of options. More precisely, if there are k different car classes so that V is partitionned into k subsets $V = V_1 \cup V_2 \cup \ldots V_k$ such that all vehicules within a same subset V_i require a same configuration of options, then the number of different arrangements of the vehicules of V in a sequence is

$$\frac{|V|!}{|V_1|! \cdot |V_2|! \cdot \ldots \cdot |V_k|!}$$

The difficulty of an instance of the car sequencing problem depends on the size of its search space but also on the utilization rates of the different options, as pointed out by Smith in [Smi96]. The utilization rate of an option corresponds to the ratio of the number of cars requiring it with respect to the maximum number of cars in a sequence which could have it while satisfying its capacity constraint. An utilization rate close to 1 indicates that the demand is close to the capacity, i.e., a hardly constrained option.

3 Solving the car sequencing problem with exact approches

3.1 Constraint Programming

Constraint Programming (CP) is a generic tool for solving constraint satisfaction problems (CSPs), i.e., problems modelled by specifying constraints on acceptable solutions, where a constraint is a relation among several unknowns or variables, each variable taking a value from a given domain [Tsa93]. To solve a CSP with a CP language, one only has to specify it by means of variables and constraints, the solution process being handled by generic algorithms that are

integrated within the CP language. These algorithms, called constraint solvers, are often based on a systematic exploration of the search space, until either a solution is found, or the problem is proven to have no solution. In order to reduce the search space, this approach is combined with filtering techniques that narrow variable domains with respect to some partial consistencies such as Arc-Consistency [Tsa93].

The car sequencing problem is a classical benchmark for CP languages, and it is the first problem of CSPLib, the library of test problems for constraint solvers [GW99]. It has been used to illustrate solving capabilities of CP languages such as, e.g., CHIP [DSvH88], cc(FD) [HSD92], and CLP(FD) [CD96].

The CSP formulation of the car sequencing problem usually introduces two different kinds of variables and three different kinds of constraints:

- A *slot variable* X_i is associated with each position i in the sequence of cars. This variable corresponds to the class of the i^{th} car in the sequence and its domain is the set of all car classes.

- An *option variable* O_i^j is associated with each position i in the sequence and each option j. This variable is assigned to 1 if option j has to be installed on the i^{th} car of the sequence, and 0 otherwise, so that its domain is $\{0, 1\}$.

- *Link constraints* specify the link between slot and option variables, i.e., $O_i^j = 1$ if and only if option j has to be installed on X_i. These constraints are stated thanks to *element* global constraints.

- *Capacity constraints* specify that station capacities must not be exceeded, i.e., for each option j and each subsequence of q_i cars, a linear inequality specifies that the sum of the corresponding option variables must be smaller or equal to p_i.

- *Demand constraints* specify, for each car class, the number of cars of this class that must be sequenced. These constraints are stated thanks to *atmost* global constraints.

Most work on solving the car sequencing problem with CP consider the decision problem, and the output is either a solution that satisfies all capacity constraints, or a failure indicating that such a solution does not exist. However, Bergen et al. [BvBC01] have proposed a constraint-based approach for a particular car sequencing problem that contains both hard constraints, that must be satisfied, and soft constraints, that can be violated at a cost.

Constraint programming has been shown to be effective to solve easy or small instances of the car sequencing problem, but it is not competitive with dedicated approaches on harder or larger instances. To improve the solution process of constraint programming, Smith has proposed to add value ordering heuristics [Smi96]. The idea is to schedule the "difficult" cars as early as possible, where the difficulty of a car is defined with respect to the utilization rates of its options.

Also, Régin and Puget have introduced in [RP97] a global sequencing constraint for imposing minimal and maximal bounds on the number of occurrences of values within periods of consecutive time units. They have proposed a filtering algorithm that is dedicated to this sequencing constraint and that exploits its global semantic to narrow more efficiently variable domains. This filtering algorithm has been integrated to Ilog Solver, a commercial CP library, and illustrated on the car sequencing problem. It allows Ilog Solver to solve some hardly constrained feasible instances, or to prove infeasibility of some over-constrained instances. However, on some other instances, it still cannot reduce domains enough to make CP tractable.

3.2 Integer Programming

Drexl and Kimms [DK01] have proposed an integer programming model for the decisional version of the car sequencing problem. The model is based on 0-1 variables C_{ij} to each car class i and each position j —to decide if the car at position j is of class i. Linear constraints ensure that (i) exactly one car class is assigned to each position, (ii) all cars of each class are assigned to a position, and (iii) all p_k/q_k capacity constraint are satisfied for each option k. The car sequencing problem is mixed with a level scheduling problem, which considers the objective of minimizing the sum of deviations of the vehicle scheduled periods from ideal ones. Hence the considered problem is the car sequencing problem with hard capacity constraints and a level scheduling objective. A second integer programming model is proposed, based on an exponential number of 0-1 variables y_k such that $y_k = 1$ if sequence k is selected, where a sequence is related to a class, giving the positions of the vehicles belonging to this class. Lower and upper bounds are computed through a column generation method. Computational results are presented on a set of generated instances.

Gravel et al. [GGP05] have proposed an integer programming model for the car sequencing problem with soft capacity constraints. This model associates the C_{ij} variable already used in [DK01] and a 0-1 variable Y_{kj} to each option k and each position j —to decide if the subsequence of length q_k starting at position j satisfies the p_k/q_k capacity constraint. Linear constraints are defined to ensure that Y_{kj} variables are assigned to one if and only if the corresponding subsequence violates the capacity constraint. The objective is to minimize the sum of all Y_{kj} variables. This integer programming formulation allowed the authors to find feasible solutions to all the instances of the test suite provided by Lee [LLW98] in CSPLib, and 4 instances of the test suite provided by Smith. However, it could not prove optimality for the 5 other instances.

3.3 Specific method

Drexl *el al.* [DKM06] have proposed a specific branch-and-bound method to solve the car sequencing and level scheduling problem they already considered in [DK01] (see Section 3.2). Considering only the car sequencing part, the method is based on an original branching scheme based on the concept of Car Sequencing (CS) state. A CS state is associated to each node of the branch and bound tree. Besides the partial sequence corresponding to the node, the CS state is characterized by a matrix $(m_{ij})_{O\times\{1,\ldots,|V|\}}$ where $m_{ij} = 0$ if option i is present in position j, $m_{ij} = 1$ if option i may be planned in period t and $m_{ij} = -1$ if it cannot be planned in position j because of branching decisions or because it would violate a constraint. Necessary conditions and learning mechanisms based on the CS states are then used to prune the search. Computational results are presented on the set of instances proposed in [DK01].

4 Heuristic approaches for the car sequencing problem

Different incomplete approaches have been proposed, that leave out exhaustivity, trying to quickly find approximately optimal solutions in an opportunistic way, e.g., greedy search, local search, genetic algorithms, and ant colony optimization.

4.1 Greedy approaches

Given a car sequencing problem, one can build a sequence in a greedy way, starting from an empty sequence, and iteratively adding a new car at the end of the sequence with respect to some greedy heuristic function. A first greedy approach has been proposed by Hindi and Ploszaski in 1994 [HP94]. In 2003, Gottlieb et al. [GPS03] have proposed and compared six different

greedy heuristics for the car sequencing problem. The best performing heuristic, among the six considered heuristics, is based on the "dynamic sum of utilization rates", i.e., at each step one adds the car that maximizes the sum of the required options utilization rates, these utilization rates being dynamically updated each time a new car is added at the end of the sequence. Gottlieb et al. have shown that this kind of greedy construction, when combined with a mild amount of randomization and multiple restarts, can solve very quickly all instances of the test suite provided by Lee [LLW98] in CSPLib.

4.2 Local Search approaches

The idea of local search is to improve a sequence by locally exploring its "neighborhood", i.e., the set of sequences that may be obtained from the current sequence by performing an elementary transformation, called a "move". From a given initial sequence, the search space is explored from neighbour to neighbour until an optimal sequence is found or until a maximum number of moves have been performed. Many different local search approaches have been proposed for solving the car sequencing problem. Most of them are generic approaches the performance of which has been illustrated, among other problems, on the car sequencing problem [DTWZ94, LLW98, DT99, MH02, NTG04, PSF04]. However, some local search approaches have been specifically dedicated to this problem [PG02, GPS03, PS04].

Performances of these different local search approaches have been illustrated on benchmark instances of the CSPLib [GW99]. All the recent approaches proposed in [MH02, GPS03, NTG04, PS04, PSF04] have been shown to be very effective on all these instances.

Local search approaches for solving the car sequencing problem mainly differ with respect to (i) the way the initial sequence is constructed, (ii) the neighborhood considered at each move, and (iii) the (meta)heuristic considered to choose a move within the neighborhood.

Construction of the initial sequence. In most cases, the initial sequence, from which the local search is started, is a random permutation of the set of vehicules to produce. However, Gottlieb et al. [GPS03] have proposed to construct this initial sequence in a greedy way, and they have shown that this significantly improves the solution process.

Neighborhood. Different kinds of moves, giving rise to different neighborhoods, may be considered. Many approaches only consider *Swap* moves that exchange pairs of cars requiring different option configurations.

However, Puchta et al. [PG02, GPS03] introduced five other move types:

- *Forward/backward Insert*, that removes the vehicule at a position i and inserts it before/after position i;
- *SwapS*, that exchanges two cars which option requirements are different for one or two options;
- *SwapT*, that exchanges two consecutive cars;
- *Lin2Opt*, that reverses the order of the cars in a subsequence;
- *Random*, that randomly shuffles a subsequence.

Also, Perron *et al.* have investigated the capabilities of Large Neighborhood Search (LNS) for solving the car sequencing problem. In [PS04], two types of neighborhoods are considered, that both rely on a size parameter s: the first one is defined by all the permutations of $4s$ cars

that are randomly and independently chosen; the second one is defined by all the permutations of a subsequence of s consecutive cars. In [PSF04], a third large neighborhood is considered, that is defined by looking at which variables are affected through constraint propagation when one variable is frozen.

Search strategies. Given a neighborhood, different (meta)heuristics have been considered for choosing the next move to perform at each iteration.

Both Lee et al. [LLW98] and Davenport et al. [DTWZ94, DT99] have considered the min-conflict hill-climbing heuristic [MJPL92] and have proposed to escape from local minima by increasing the weight of the violated constraints.

Michel and van Hentenryck [MH02] have considered a reactive tabu search approach, where the size of the tabu list is reactively adapted with respect to the need for diversification.

In [PG02, GPS03], a randomly chosen move is evaluated, and the move is accepted and applied, if it does not deteriorate the solution quality. Otherwise the move is rejected and another one is tried on the current solution.

In the LNS approaches proposed in [PS04, PSF04], the large neighborhoods are explored thanks to constraint programming, and the first non deteriorating move is applied.

Neveu et al. [NTG04] have proposed a new metaheuristic, called *IDWalk*. This approach introduces only one parameter, called *Max*, that determines the maximum number of neighbours that are considered before performing every move. At each iteration *IDWalk* chooses the first non decreasing neighbour, and if all the *Max* considered neighbours deteriorate the current solution, *IDWalk* chooses the best one over them. The *Max* parameter is automatically determined at the beginning of the search by performing a few short walks with different possible values.

4.3 Genetic Algorithms

Warwick and Tsang [WT95] have proposed a genetic algorithm for solving the car sequencing problem. This approach takes inspiration from natural evolution and explores the search space through selection, cross-over and mutation operators upon a population of sequences. At each generation, selected sequences are combined by cross-over operations; as the created offsprings may not satisfy the global permutation constraint, they are greedily repaired; after repair, each offspring is hill-climbed by a swap function (similar to the one used in local search approaches). Experiments reported in [WT95] show that this approach is able to solve easy instances of the car sequencing problem, with low utilization percentages. However, with higher utilization percentages, the number of successful runs is severely decreased.

4.4 Ant Colony Optimization (ACO)

Solnon [Sol00] has proposed a first ACO algorithm dedicated to permutation constraint satisfaction problems —the solution of which is a permutation of a given tuple of values. Performances of this algorithm have been illustrated, among other problems, on the car sequencing problem.

The basic idea of Ant Colony Optimization (ACO) [DS05] is to model the problem to solve as the search for a minimum cost path in a graph, and to use artificial ants to search for good paths. The behavior of artificial ants is inspired from real ants: they lay pheromone trails on components of the graph and they choose their paths with respect to probabilities that depend on pheromone trails that have been previously laid by the colony; these pheromone trails progressively decrease by evaporation. Intuitively, this indirect stigmergetic communication means aims at giving information about the quality of path components in order to attract ants, in the following iterations, towards the corresponding areas of the search space.

In the ACO algorithm of [Sol00], pheromone is laid on couples of consecutive cars in order to learn for promising sub-sequences of cars. This first algorithm has been improved in [GPS03] by integrating greedy heuristics, and it has been experimentally compared with the local search approach of [PG02], showing that local search is slightly inferior to ACO for small CPU time limits, whereas for larger limits both approaches yield comparable solution quality.

Gravel et al. [GGP05] have proposed another ACO algorithm for the car sequencing problem, that integrates a local search procedure that is used to improve the solutions constructed by ants.

5 The ROADEF'2005 car sequencing problem

The car sequencing problem proposed by RENAULT for the ROADEF'2005 challenge differs from the standard problem defined in Section 2. Indeed, besides capacity constraints imposed by the assembly shop, it also introduces paint batching constraints, and it considers two categories of capacity constraints to take into account their priority. These new requirements that are specific to vehicule production planning and sequencing at RENAULT are described in Section 5.1, whereas Section 5.2 formally defines the problem.

5.1 Vehicle production planning and sequencing at RENAULT

Client orders are sent in real time to the vehicle plants (see Figure 1). The daily task of each plant is (1) to assign a single-day manufacturing period to each ordered vehicle, taking assembly line capacity constraints and client due dates into account. The next task is (2) to sequence the vehicles inside each production day while satisfying at best the requirements of the production workshops: body, paint and assembly workshops. The resulting vehicle sequence is the baseline sequence sent to the workshops.

Figure 1: Vehicle plants at RENAULT

For the challenge, the following assumptions are made. Only the paint and assembly worhshops are considered, assuming the body shop is not critical for the schedule of a production day. The assignment decisions made at step (1) cannot be overruled.

This planning and scheduling process is currently performed at RENAULT by a software which uses linear programming for step (1) and simulated annealing for step (2).

Paint workshop requirements. The main paint shop objective is to minimize the consumption of a solvent which is used to clean the paint equipment at each color change.

This is achieved by sequencing contiguously the same-colored vehicles. Indeed, gathering the vehicles having the same color into batches and sequencing the batches in any order minimizes the number of paint cleanings.

However, each batch of vehicles has to be of limited size since paint equipment has to be cleaned periodically, even if no color change occured. This **batch size limit** is a hard constraint, i.e., that cannot be violated by any feasible solution.

Assembly workshop requirements. The main objective of the assembly workshop is the work load balancing of the different work units on the assembly lines. To achieve this purpose, the vehicles requiring complex operations have to be sufficiently distant in the sequence. In other words, the density of these "difficult" vehicles has to be limited to prevent overloading the work units that assemble them.

This is precisely achieved by the p/q ratio contraints, see Section 2. In many cases, there is no sequence allowing to satisfy all the ratio constraints for a given manufacturing period. Hence, as precised in Section 2, ratio constraints are tackled as "soft" constraints and the objective of optimization is to minimize the number of violated ratio constraints.

Priority classes for assembly workshop requirements. While the standard car sequencing problem considers only ratio constraints (see Section 2), RENAULT has always considered both the number of color changes and soft ratio constraint objectives, for two major reasons. First, according to the RENAULT supply chain strategy, the paint and assembly workshops process the same vehicle sequence. Second, there is a compromise to find between these two objectives.

As a matter of fact, depending on the labor cost, it can be advantageous either to best satisfy the ratio constraints, which allows to limit the workforce level requirements of the assembly line, or to optimize the runs of cars of the same color when savings in solvent compensate the overcost due to workload increase in the assembly shop.

To make the balancing between assembly and painting easier, two categories of ratio constraints are defined, the priority ratio constraints and the nonpriority ratio constraints. The priority ratio constraints correspond to critical operations in the assembly shop whereas the nonpriority ones correspond to less critical operations and are defined mostly for workload smoothing.

Factories with high labor costs, while putting the emphasis on color change minimization, can also define a subset of priority ratio constraints of higher importance, and another subset of nonpriority ratio constraints, satisfied at no expense of additional color changes.

5.2 Optimization problem formulation of the challenge problem

Counting the ratio constraint violations. The evaluation of ratio constraint violations is slightly different from the standard definition, in order to express the fact that, when an option is overconstrained, one should sequence the vehicules requiring this option as evenly as possible. This purpose is reflected by computing the ratio constraint violations on gliding subsequences throughout the sequence. If many vehicules are too close with respect to ratio constraints, many violations will be observed inside each gliding subsequence (instead of only one violation as defined in the standard car sequencing problem), and these violations will also be counted several times for all gliding subsequences they appear in.

Another difference with the standard car sequencing problem is that, for the last gliding subsequence of the current day, the computation for each ratio is done assuming that the first cars of the subsequent day are not concerned by the ratio.

Let us consider for example a 1/5 ratio and the sequence of cars `--XX-X`, where an '`X`' (resp. '`-`') denotes a vehicule that requires (resp. does not require) the option. To evaluate this sequence, we first consider the two gliding subsequences of size five, i.e., `--XX-`, which is evaluated to 1, and `-XX-X`, which is evaluated to 2. Then, we consider the subsequences `XX-X-`, `X-X--`, and `-X---`, which are obtained by completing the last gliding subsequence by cars that are not concerned by the ratio, and which are respectively evaluated to 2, 1, and 0. Hence, the whole sequence `--XX--X` is evaluated to 6 with respect to a 1/5 ratio constraint.

More formally, we define the number of violations in a sequence $(\sigma_u)_{u\in\{1,\ldots,n\}}$ of n vehicules, for an option o_i that have a capacity ratio p_i/q_i, as follows

$$\begin{aligned}\#\text{violations}((\sigma_u)_{u\in\{1,\ldots,n\}}, o_i) &= \sum_{j\in\{1,\ldots,n-q_i+1\}} \max(0, (\sum_{k\in\{j,\ldots,j+q_i-1\}} r_{\sigma_j i}) - p_i) \\ &+ \sum_{j\in\{n-q_i,\ldots,n-1\}} \max(0, (\sum_{k\in\{j,\ldots,n\}} r_{\sigma_j i}) - p_i)\end{aligned}$$

Lexicographic multiobjective optimization. The objective is to build a sequence of vehicules optimizing the requirements of paint and assembly shops, which leads to the minimization of three possibly conflicting objective functions: the number of violated priority ratio constraints (#PRC), the number of violated non priority ratio constraints (#NPRC), and the number of color changes (#CC). As priority constraints have a higher priority than non priority constraints, the three following objective hierarchies are observed in the different RENAULT factories:

minimize #PRC, then #CC, and then #NPRC

or

minimize #PRC, then #NPRC, and then #CC

or

minimize #CC, then #PRC, and then #NPRC

The lexicographic optimization is garanteed by penalties 10^6, 10^3, and 1 assigned to the first, second, and third objectives respectively.

Formal definition of the challenge problem. The RENAULT car sequencing problem is defined by:

- a standard car sequencing problem instance (V, O, p, q, r), as defined in Section 2;
- two subsets O_P and O_{NP} of O such that O_P (resp. O_{NP}) contains the set of options having priority (resp. non priority) ratio constraints;
- a set of colors C and a function $c : V \to C$ that associates a color c_i to each vehicule $v_i \in V$;
- a batch size limit $B \in \mathbb{N}$;
- the weights w_{CC}, w_{PRC}, and w_{NPRC} of color changes, priority and nonpriority ratio constraint violations respectively, where $\{w_{CC}, w_{PRC}, w_{NPRC}\}$ is a permutation of $\{1, 10^3, 10^6\}$ such that $w_{PRC} > w_{NPRC}$;
- a sequence $(\sigma_u)_{u\in\{-k+1,\ldots,0\}}$ that contains the last k vehicules sequenced during the previous day, where $k = \max_{o_i\in O} q_i - 1$. This enables to take into account the previous production day when computing the number of constraint violations and color changes.

A feasible solution is a sequence $\sigma = (\sigma_u)_{u\in\{1,\ldots,|V|\}}$ that is a permutation of V, and that satisfies the batch size limit, i.e., that does not contain more than B consecutive cars with a same color.

The cost of a feasible solution σ is the weighted sum

$$\text{cost}(\sigma) = w_{CC} \cdot \#\text{CC}(\sigma) + w_{PRC} \cdot \#\text{PRC}(\sigma) + w_{NPRC} \cdot \#\text{NPRC}(\sigma)$$

where

- $\#\mathrm{CC}(\sigma)$ is the number of color changes in σ. When evaluating this number, the last car sequenced during the previous day (σ_0) is considered, i.e.,

$$\#\mathrm{CC}(\sigma) = |\{t \in \{0, \ldots, n-1\}, c_{\sigma_t} \neq c_{\sigma_{t+1}}\}|$$

- $\#\mathrm{PRC}(\sigma)$ is the number of violated priority ratio constraints in the sequence composed of the last k vehicules of the previous day followed by the vehicules of σ, i.e.,

$$\#\mathrm{PRC}(\sigma) = \sum_{o_i \in O_{PR}} \#\mathrm{violations}((\sigma_u)_{u \in \{-k+1,\ldots,|V|\}}, o_i)$$

- $\#\mathrm{NPRC}(\sigma)$ is the number of violated nonpriority ratio constraints in the sequence composed of the last k vehicules of the previous day followed by the vehicules of σ, i.e.,

$$\#\mathrm{NPRC}(\sigma) = \sum_{o_i \in O_{NPR}} \#\mathrm{violations}((\sigma_u)_{u \in \{-k+1,\ldots,|V|\}}, o_i)$$

6 Description and results of the ROADEF'2005 Challenge

The above-described Car Sequencing Problem was already solved in a satisfactory manner in the RENAULT factories through a sofwtare based on simulated annealing [CDNT92]. However, during the last years, this problem became critical since the company made the strategic decision of enforcing at the workshop level a strict respect of the baseline sequence which is computed 6 days before the actual production.

Before this policy, it was possible to sort the vehicles at the paint shop entrance to improve color runs without considering the ratio constraints. The sequence was then rearranged at the paint shop entrance to minimize the ratio constraint violations. Such local rearrangements are no more allowed and the optimization of the baseline sequence is now a key issue.

In such a context the ROADEF challenge represented for RENAULT a real opportunity for evaluating several possible car sequencing algorithms.

6.1 Data instance sets and challenge phases

The data provided to the challenge participants are all issued from RENAULT factories. However, to test new possible configurations, the ratio constraints of some instances have been tightened and the objective hierarchies of some others have been changed.

All candidates were also provided with the solutions found by the simulated annealing software used by RENAULT.

Three instance sets were used for the challenge:

A set: The first set (made of 16 instances) served during the challenge qualification phase. It was built from the data of 6 different factories (from 1 to 2 instances per factory). Based on the results of the teams on this first set, an intermediate ranking was issued. Each candidate team was informed of all these results.

B set: The second set (made of 45 instances) was provided to the qualified teams so they would be able to improve and tune their methods. The set was built from data coming from 10 factories. Each instance has to be solved for each of the 3 possible objective hierarchies. The interest for RENAULT was here to test the algorithms in the context of both current and targeted shop configurations. Furthermore, some required ratios were overconstrained to test the ability of the methods to smoothen the violations of unsatisfiable contraints.

Inst.	#cars	#ratio	#colors	#config.	p/q ratios
022	704	3+9	13	84	1/(3,4,5,6,10), 2/3, 5/6, 10/15
023	1260	5+7	13	202	1/(2,3,5,6,7,8,10,20)
024	1319	7+11	15	275	1/(2,3,4,6,7,10,13,15,20), 3/5, 6/7
025	996	6+14	19	235	1/(2,3,4,6,7,9,10,18,20,28,76), 2/3, 5/8, 17/30
028_1	325	9+17	15	150	1/(2,3,4,6,7,8,10,12,13,16,18,20,25,39), 1/(46,91,136), 3/5, 3/4, 4/5
028_2	65	1+5	4	10	1/(2,3,5,6,14)
029	780	4+3	14	73	1/(2,3,6,7), 3/4, 39/43
034_VP	931	3+5	10	24	1/(2,3,5,6,7,10,29,65)
034_VU	231	6+2	7	64	1/(2,4,5,13,17,22), 3/5
035_1	90	1+0	6	11	1/2
035_2	376	2+0	7	19	1/2
039_1	1247	1+11	15	328	1/(2,4,6,7,8,13,19,20,56,297)
039_3	1037	2+10	17	156	1/(3,5,8,14,16,19,25,38,47,247), 12/60, 41/60
048_1	519	6+16	13	209	1/(1,2,3,4,5,6,7,8,9,10,18,137,274), 2/3
048_2	459	8+12	13	141	1/(2,3,4,5,6,7,8,9,10,20,25,50)
064_1	875	9+2	13	156	1/(2,4,5,8,9,30,10,100), 2/3, 4/9
064_2	273	6+0	12	42	1/(2,5,10,20,100), 8/10
655_1	264	4+1	9	19	1/(2,5,15)
655_2	219	4+0	7	18	1/(6,9,16), 2/3

Each row displays successively the instance name, the number of vehicles, the total number of priority and nonpriority ratio contraints, the number of colors, the number of configurations, i.e., the number of classes of vehicles of different color and/or option requirements, and the different p/q ratio values.

Table 1: The RENAULT instance set X

Inst.	#cars	#options	#config.	p/q ratios
CSP lib [GW99]	100-200	5	17-30	1/(2,3,5), 2/(3,5)
Gravel *at al* [GGP05]	200-400	5	19-26	1/(2,3,5), 2/(3,5)

Table 2: The classical car sequencing instances

X set: The last set (made of 19 instances) remained unknown from the teams until the end of the challenge. It was used by the jury to issue the final ranking. This set reflects the data of all the 19 RENAULT and DACIA assembly lines in Europe and Mercosur, each instance representing a production day on one of these sites.

In table 1, we give the characteristics of the X instances. For comparison purpose, we give in table 2 the characteristics of the CSPLib instances [GW99] and the instances proposed by Gravel *et al* [GGP05] (which do not consider priorities nor colors). The tables show clearly that the industrial instances are much larger and can be useful to reveal the scalability of state-of-the-art methods. Note also that ratio constraints of the X instances are much less regular than the ones of the CPSLib, and that q values are often much larger. This last point is likely to increase considerably the difficulty of the problem, since even counting the number of violated constraints can be a non-trivial task.

6.2 Challenge conditions and ranking process

For both the qualification and final phases, the candidates had to send their software either as an executable file compatible with the target computer architecture (Pentium IV – 1,6 Ghz and 1 Go RAM) or as a C or C++ file. The tests were run by RENAULT allocating for each program 5 runs of 10 minutes and the results for each candidate were obtained by an average over the 5 runs.

The initial ranking method was the following:

- For each instance, set the mark to the obtained objective value.
- For each instance, normalize the mark with the best and the worst marks:

$$\frac{mark_{team} - mark_{worst}}{mark_{best} - mark_{worst}}$$

- Compute for each team the average mark over all instances.

However it appears that the results of the best teams were so close that this method turns out to be inadequate. Hence, it has been decided to refine the mark computation as follows. A "musical chairs" principle has been applied for the mark computation. After each evaluation round, the last ranked team was eliminated for the next round. The procedure stops when only 3 teams remain.

6.3 Synthesis of the proposed methods

We make a synthesis of the methods proposed by the candidate teams, in terms of the families of proposed algorithms, of some common principles used by most methods (greedy algorithm and neighborhood operators), multiobjective optimization, numerical results and implementation issues.

6.3.1 Families of algorithms

Table 3 gives a classification of all the methods proposed by the participants of the qualification stage, and Table 4 gives the methods proposed by the finalist teams, listed in the final ranking order.

The number of methods is larger than the number of candidates since some teams combined several algorithms. The second team in the final ranking implemented 6 different greedy heuristics and several combinations of Variable and/or Iterated Neighborhood Search, depending on the considered objective hierarchy.

The predominance of local search methods has to be underlined. A great attention has been paid by the candidates on the definition of neighborhood operators and on the design of space search strategy such as the balance between intensification and diversification and/or the escape from local optima. Sparcer methods such as AG, AC and ILP have however shown their potential on the car sequencing problem.

The case of the winner team (Estellon, Gardi et Nouioua) is worth mentionning since they proposed a large neighborhood search method (based on ILP) for the qualification phase which obtained limited results due to the short runtime limitation of 10 minutes. For the final phase they switched to a fast local search method.

Algorithm	#teams.
greedy heuristic (CH)	All
Simulated Annealing (SA)	5
Tabou Search (TS)	7
Variable Neighborhood Search (VNS)	4
Large Neighborhood Search (LNS)	2
Local Search (LS)	5
Iterated Local Search (ILS)	2
Genetic Algorithm (GA)	1
Ant Colony Optimization (AC)	1
Integer Linear Programming (ILP)	1
Constraint Programming (CP)	1

Table 3: Methods proposed at the qualification stage

Rg.	Algorithm	Team
1	LS	Estellon, Gardi & Nouioua
2	VNS+ILS with restart	Aloise, Noronha, Rocha, Ribeiro & Urrutia
3	SA	Briant, Naddef & Mounié
4	SA	Bloemen
5	SA	Kuipers
6	LS	Gravranovic
7	TS	Cordeau, Laporte & Pasin
8	VNS+SA	Riesler, Chiarandini, Paquete, Schiavinotto & Stützle
9	TS	Craciunas, Gendreau & Potvin
10	LS	Pawlak, Rucinski, Piechowiak, Plaza
11	AC+LS+ TS/VNS	Gravel, Gagné, Price, Krajecki & Jaillet
12	LS	Benoist

See meaning of acronyms in Table 3

Table 4: Methods and ranking of finalist teams

6.3.2 Greedy heuristics

As expected, all methods build the initial solution with a fast and simple method when the objective to minimize the number of color changes is considered first since the problem is polynomial.

The vast majority of proposed works also developed greedy heuristics to generate good initial solutions when the priority ratio constraint violation objective is considered first. Usually the so-obtained first initial solution was better than the solution proposed by RENAULT. We have to note that one team used Constraint Programming to build such an initial solution and that CP has not been used by any other team in this challenge.

6.3.3 Neighborhood operators

The best results were obtained by using the neighborhood operators described in Section 4.2, and some variants such as (the following list is not exhaustive) :

Group swap : Swap two subsequences.

K swap : Swap K couples of vehicules.

forward/backward Group insert : Insert a subsequence of vehicules after/before a given position j.

Invert same type : Reverse a subsequence where the first and last vehicles either have the same color, are equivalent in terms of priority ratio constraints, or are the first or last vehicles of runs of identical (or distinct) colors.

Taking two extremes, the "Shuffle" operator was used carefully since its evaluation requires exponential times. However it permits a good diversification of the search. On the contrary, the challenge winners (Estellon, Gardi and Nouioua) reported that the simple "SwapT" operator offers a good compromise for solution improvement (while avoiding large perturbations) and CPU time since its evaluation is quite fast.

6.3.4 Multiobjective characteristics and numerical results

Most of the teams adopted the standard strategy for lexicographic multiobjective optimization (At step 1 optimize the first objective. At each step $s > 1$ optimize objective s while fixing the obtained values for objectives $s - x$, $x \geq 1$). The allocated CPU time was equally distributed among the 3 steps.

The differences between the best results were significant only on the second and third objectives. However, these differences are important in practice especially when the second or third objective is the number of color changes which yields significant savings in solvent.

Table 5 gives the results of the first three teams of the final ranking on the X instance set. The objective order is given, where P denotes the priority ratio constraints, NP denotes the non priority ratio constraints, and C denotes the number of colour changes. Objective values are averaged over 5 runs for each instance. All algorithms are robust and yield low standard deviations.

One has to observe that the results are very close. They all obtain the same value for the first objective and, for some instances, reach the same values for the 3 objectives. As said above, this led to use the musical chairs selection method to obtain the final ranking.

Note that the nonfunctionning of the 3rd team code on instance 024 was highly penalizing for the final ranking.

Inst./Obj.	Rank	P	NP	C
022	1	2,00	3,00	12,00
C_P_NP	2	2,00	3,00	12,00
	3	2,00	3,00	12,00
023	1	0,00	66,00	192,40
P_C_NP	2	0,00	77,20	193,00
	3	0,00	68,20	192,80
024	1	0,00	6,00	337,00
P_C_NP	2	0,00	12,00	352,80
	3	n.a.	n.a.	n.a.
025	1	0,00	160,00	407,60
P_NP_C	2	0,00	160,00	602,20
	3	0,00	160,00	437,80
028_1	1	36,00	370,00	94,00
P_NP_C	2	36,00	341,40	95,40
	3	36,00	360,00	92,40
028_2	1	0,00	0,00	3,00
P_NP_C	2	0,00	0,00	3,00
	3	0,00	0,00	3,00
029	1	0,00	42,60	110,40
P_C_NP	2	0,00	66,00	111,80
	3	0,00	3,00	112,00
034_VP	1	0,00	586,80	55,80
P_C_NP	2	0,00	643,60	58,00
	3	0,00	582,00	56,60
034_VU	1	8,00	37,00	87,00
P_C_NP	2	8,00	35,80	87,00
	3	8,00	36,40	87,00
035_1	1	10,00	0,00	5,00
C_P	2	10,00	0,00	5,00
	3	10,00	0,00	5,00

Inst./Obj.	Rank	P	NP	C
035_2	1	56,00	0,00	6,00
C_P	2	56,00	0,00	6,00
	3	56,00	0,00	6,00
039_1	1	0,00	239,00	69,00
P_C_NP	2	0,00	479,60	69,00
	3	0,00	221,20	69,80
039_3	1	0,00	30,20	231,00
P_C_NP	2	0,00	2162,60	231,00
	3	0,00	30,00	231,00
048_1	1	0,00	1044,80	196,00
P_C_NP	2	0,00	1016,00	196,00
	3	0,00	1053,80	196,00
048_2	1	31,00	1116,20	76,80
P_C_NP	2	31,00	1128,40	79,00
	3	31,00	1118,60	77,60
064_1	1	61,00	29,80	187,20
P_C_NP	2	61,00	81,40	190,80
	3	61,00	29,00	190,40
064_2	1	0,00	0,00	37,00
P_C_NP	2	0,00	0,00	37,00
	3	0,00	0,00	37,00
655_1	1	0,00	0,00	30,00
P_C_NP	2	0,00	0,00	30,00
	3	0,00	0,00	30,00
655_2	1	153,00	0,00	34,00
P_C_NP	2	153,00	0,00	34,00
	3	153,00	0,00	34,20

Table 5: Results of the first 3 teams on instance set X

6.3.5 Implementation issues

Due to the short time limit and the large size of some instances, the implementation issues were crucial in this challenge. The winner team implemented for that purpose optimized evaluation techniques of the neighborhood for each operator which allow them to perform on average 170 millions neighborhood evaluations during the 10 minutes of allocated CPU time. The team that has been ranked second (Aloise, Noronha, Rocha, Ribeiro et Urrutia) defined special data structures to evaluate each exchange move in $O(m)$ where m is the number of ratio constraints.

7 Conclusion

The ROADEF'2005 Challenge was particularly successful and brought a comprehensive insight on the proposed methods. The heuristic approaches described in Section 4, i.e. greedy aproaches, local seach methods, genetic algorithms, and ant colony optimization methods, were all successfully adapted to the specificity of this problem.

It could be surprising at first sight that no constraint programming based method were competing after the qualification phase. Indeed, Car Sequencing was a traditional benchmark problem for constraint programming as stated in Section 3. This is partially due to the short runtime requirement (10 minutes) and the large-sized real-life industrial instances, compared to the much smaller CSPlib instances. Some candidates report that they have tested CP and ILP-based approaches that could not obtain satisfactory results after several hours of computational times. Actually, a promising research direction could be to integrate exact and heuristic approaches, that have complementary capabilities: exact approaches such as CP and ILP may be used to prove new bounds whereas heuristic approaches may be used to provide exact approaches with good initial solutions that may be used to prune the search space more efficiently.

The expectations of RENAULT were fully met : because of the great potential savings, the winner team's algorithm has been integrated in RENAULT operational software and rolled out in European pilot factories in November 2005. The roll out in all RENAULT factories worldwide will be completed by the first semester of 2006.

Interesting issues are also raised on the potential balance between paint and assembly shops. Some assembly constraints could be indeed further relaxed to obtain large solvent savings.

Also, practical extensions, which were not presented during the chalenge for sake of simplications, will be studied on the near future. These are:

- predefined positional windows for some vehicles. Such hard constraints are necessary for quality control.
- "crossed ratio contraints". Such ratios concern two different technical features (A) and (B). For instance, ratio constraint " A B 1/N " means that any vehicle with feature A must be distant of at least N-1 positions from any vehicle with feature B.
- "Assembling several sequences into a single one". In some factories several paint shops work in parallel before the corresponding vehicle sequences merge for the assembly shop. It is therefore necessary to compute a sequence per paint shop. Moreover, some vehicles have to be processed by a specialized paint shop whereas others can be processed on any paint shop, which introduces assignment problems.
- Decision support for operators. Which ratio constraints should be relaxed to improve significantly the number of color change objective?

The 4th ROADEF challenge confirms and strengthens the international scope of this competition. 55 inscriptions were received from 15 different countries. No less than 27 programs were submitted at the end of the qualification phase. For the final stage, 12 teams of 7 countries (Bosnia-Hercegovina, Brazil, Canada, France, Germany, The Netherlands, Poland) were invited to present their results during the ROADEF'2005 conference in Tours (France).

Furthermore, the quality of the submitted methods, in particular for junior teams, reached an exceptional level. For the first time two junior teams reached the 1st and 2nd rank in the general final ranking.

The ROADEF challenge demonstrated once again its ability to reach its double objectives promoting the operations research methods for solving industrial NP-hard problems, and allowing conversely academic researchers to confront their ideas with real applications. All the details on the ROADEF challenges can be consulted on the ROADEF web site `www.roadef.org`.

References

[BvBC01] M.E. Bergen, P. van Beek, and T. Carchrae. Constraint-based vehicle assembly line sequencing. In *Proceedings of the 14th Canadian Conference on Artificial Intelligence*, volume 2056 of *LNCS*, pages 88–99, 2001.

[CD96] Ph. Codognet and D. Diaz. Compiling constraints in clp(FD). *Journal of Logic Programming*, 27(3):185–226, 1996.

[CDNT92] T.-L. Chew, J.-M. David, A. Nguyen, and Y. Tourbier. Solving constraint satisfaction problems with simulated annealing : The car sequencing problem revisited. In *12th Int'l Workshop on Expert Systems and their Applications*, pages 405–416, Avignon, France, 1992.

[DK01] A. Drexl and A. Kimms. Sequencing jit mixed-model assembly lines under station load and part-usage constraints. *Management Science*, 47(3):480–491, 2001.

[DKM06] A. Drexl, A. Kimms, and L. Matthiessen. Algorithms for the car sequencing and the level scheduling problem. *Journal of Scheduling (to appear)*, 2006.

[DS05] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, 2005.

[DSvH88] M. Dincbas, H. Simonis, and P. van Hentenryck. Solving the car-sequencing problem in constraint logic programming. In Y. Kodratoff, editor, *Proceedings of ECAI-88*, pages 290–295, 1988.

[DT99] A.J. Davenport and E.P.K. Tsang. Solving constraint satisfaction sequencing problems by iterative repair. In *Proceedings of the first international conference on the practical applications of constraint technologies and logic programming (PACLP)*, pages 345–357, 1999.

[DTWZ94] A. Davenport, E. Tsang, C. Wang, and K. Zhu. Genet: a connectionist architecture for solving constraint satisfaction problems by iterative improvement. In *Proceedings of AAAI'94*, pages 325–330, 1994.

[Gen98] I.P. Gent. Two results on car-sequencing problems. Technical report (http://www.apes.cs.strath.ac.uk/apesreports.html), APES, 1998.

[GGP05] M. Gravel, C. Gagné, and W.L. Price. Review and comparison of three methods for the solution of the car-sequencing problem. *Journal of the Operational Research Society*, 56(11):1287–1295, 2005.

[GPS03] J. Gottlieb, M. Puchta, and C. Solnon. A study of greedy, local search and ant colony optimization approaches for car sequencing problems. In *Applications of evolutionary computing*, volume 2611 of *LNCS*, pages 246–257. Springer, 2003.

[GW99] I.P. Gent and T. Walsh. Csplib: a benchmark library for constraints. Technical report, APES-09-1999, 1999. available from http://www.csplib.org/. A shorter version appears in CP99.

[HP94] K.S. Hindi and M.G. Ploszajski. Formulation and solution of a selection and sequencing problem in car manufacture. *Computers and Industrial Engineering*, 26(1):203–211, 1994.

[HSD92] Pascal Van Hentenryck, Helmut Simonis, and Mehmet Dincbas. Constraint satisfaction using constraint logic programming. *Artificial Intelligence*, 58(1-3):113–159, 1992.

[Kis04] T. Kis. On the complexity of the car sequencing problem. *Operations Research Letters*, 32:331–335, 2004.

[LLW98] J.H.M. Lee, H.F. Leung, and H.W. Won. Performance of a comprehensive and efficient constraint library using local search. In *11th Australian JCAI*, LNAI. Springer-Verlag, 1998.

[MH02] L. Michel and P. Van Hentenryck. A constraint-based architecture for local search. In *OOPSLA '02: Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 83–100, New York, NY, USA, 2002. ACM Press.

[MJPL92] S. Minton, M.D. Johnston, A.B. Philips, and P. Laird. Minimizing conflicts: a heuristic repair method for constraint satistaction and scheduling problems. *Artificial Intelligence*, 58:161–205, 1992.

[NTG04] B. Neveu, G. Trombettoni, and F. Glover. Id walk: A candidate list strategy with a simple diversification device. In *Proceedings of CP'2004*, volume 3258 of *LNCS*, pages 423–437. Springer Verlag, 2004.

[PG02] M. Puchta and J. Gottlieb. Solving car sequencing problems by local optimization. In *EvoWorkshops*, volume 2056 of *LNCS*, pages 132–142. Springer Verlag, 2002.

[PKW86] B.D. Parello, W.C. Kabat, and L. Wos. Job-shop scheduling using automated reasoning: a case study of the car sequencing problem. *Journal of Automated Reasoning*, 2:1–42, 1986.

[PS04] L. Perron and P. Shaw. Combining forces to solve the car sequencing problem. In *Proceedings of CP-AI-OR'2004*, volume 3011 of *LNCS*, pages 225–239. Springer, 2004.

[PSF04] L. Perron, P. Shaw, and V. Furnon. Propagation guided large neighborhood search. In *Proceedings of Principles and Practice of Constraint Programming (CP'2004)*, volume 3258 of *LNCS*, pages 468–481. Springer, 2004.

[RP97] J.-C. Regin and J.-F. Puget. A filtering algorithm for global sequencing constraints. In *CP97*, volume 1330 of *LNCS*, pages 32–46. Springer-Verlag, 1997.

[Smi96] B. Smith. Succeed-first or fail-first: A case study in variable and value ordering heuristics. In *third Conference on the Practical Applications of Constraint Technology PACT'97*, pages 321–330, 1996.

[Sol00] C. Solnon. Solving permutation constraint satisfaction problems with artificial ants. In *Proceedings of ECAI'2000, IOS Press, Amsterdam, The Netherlands*, pages 118–122, 2000.

[Tsa93] E.P.K. Tsang. *Foundations of Constraint Satisfaction*. Academic Press, London, UK, 1993.

[WT95] T. Warwick and E. Tsang. Tackling car sequencing problems using a genetic algorithm. *Journal of Evolutionary Computation - MIT Press*, 3(3):267–298, 1995.