SCHEDULING OPTIMIZATION IN A CAR ASSEMBLY LINE USING GENETIC ALGORITHMS

Fábio C. Malinowski [1]
famalinowski@uol.com.br
[1]RENAULT do Brasil Automóveis S.A
BR 277, km 73, Estrada da Roseira
83090-900 São José dos Pinhais (PR) – BRAZIL

Heitor S. Lopes [2]
hslopes@cpgei.cefetpr.br

Flávio Neves Jr. [2]
neves@cpgei.cefetpr.br
[2]CEFET-PR/CPGEI
Av. Sete de Setembro, 3165
80230-901 Curitiba (PR) – BRAZIL

Abstract
In this work it is described the use of Genetic Algorithms for the car assembly line scheduling optimization problem. The problem is defined in terms of the variables related to the assembly line and requested production. It can be summarized as finding the order in which vehicles of different types with different optional accessories are assembled, respecting the operational restrictions of the workstations imposed by each option type. Four simulations were done in which the GA approach has demonstrated excellent performance, even when compared with an industrial optimization system.

1. Introduction

Nowadays, with the growing diversity of car types, where the customer is free to choose the vehicle and optional accessories (henceforth, "options") that is more appropriated in terms of cost and comfort, the car assembling companies have to be prepared to assemble the requested vehicle as soon as possible and with the smallest possible production cost. To reduce production costs, the sequence in which cars are assembled is very important. As it happens in all assembly lines, the resources related to the technological processes are restricted (time, workers, workstations, etc), and it is not interesting to increase the production costs hiring more workers or investing in supplementary equipments. These restrictions impose several limits to the production in order to assure the quality.

A Genetic Algorithm – GA (Goldberg, 1989) is a search and optimization method broadly used for engineering and computer science problems. In this work, a GA is used to optimize the ordering in which vehicles are assembled, considering the restrictions of the assembly line and the type and options of each vehicle. In a similar problem (Warwick & Tsang, 1996), GA was proved efficient, and this has motivated the present work.

2. Methodology

2.1 – Problem Definition

The first step to apply the methodology is the complete definition of the problem in terms of the basic relationships among variables related to the production line. The current state of the factory and the amount of requested vehicles to be produced are checked. A daily production schedule determines the

number and types of vehicles to be produced. Vehicles are typified according to their optional accessories (options). Using these information, equation 1 computes the total number of vehicles that can be produced.

$$N = \sum_{j=1}^{k} pr[j] \tag{1}$$

where:

k = vehicle type (0...7) produced in this factory;
$pr[j]$ = demand of production of the vehicle type j;
N = total number of vehicles to be produced.

To associate possible options to the different vehicle types, a grouping $O[m,j]$ is used, where m represents the option and j the vehicle type. For instance, $O[m,j]=1$ means that the vehicle type j has the option type m, otherwise $O[m,j]=0$. Next, for every option, its corresponding restrictions are defined. Restrictions are represented by the values of the variables pm:qm that mean that for a given option m, pm vehicles can be assembled with this option respecting a minimum interval of other qm vehicles, without the same option. Using the parameters mentioned before, the following relations can defined:

a) The total number of a given option ($Onum$) requested in the scheduling:

$$Onum(m) = \sum_{j=1}^{k} \left(pr[j] * O[m, j] \right) \tag{2}$$

b) The maximum number of each option ($Omax$) allowed in the scheduling taking into account its respective process restriction:

$$Omax(m) = \left(\frac{pm}{qm} \right) * N \tag{3}$$

c) The resources utilization level (Um) for the workstation for each option:

$$Um(m) = \frac{Onum(m)}{O\max(m)} \tag{4}$$

A necessary, but not sufficient condition for the vehicles sequencing be feasible, it is that all the capacity restrictions for each option have to be satisfied, resulting in an average value for the total resources utilization level $\bar{m} < 1$, that is:

$$\bar{m} = \frac{\sum_{m=1}^{n} Um(m)}{n} \tag{5}$$

where:

n = number of options;

Um(m) = resources utilization level for the option *m*.

For a typical production scheduling, using real-world data, table 1 shows options, restrictions and the previously defined parameters for the assembly line.

Table 1 – Example of a vehicle production schedule.

Options	**Vehicle types**								**Process restrictions**			
	0	1	2	3	4	5	6	7	*p:q*	*Omax*	*Onum*	*Um*
1. ABS	0	1	0	0	0	0	1	0	1:3	27	16	0.60
2. Leather seats	0	0	1	0	0	0	0	0	1:5	20	10	0.50
3. Diesel Engine	0	0	0	1	0	0	0	0	1:4	27	13	0.48
4. Air conditioner	0	1	0	0	0	1	1	0	2:5	53	20	0.37
5. Electric pack	0	0	1	0	1	0	0	0	1:2	40	27	0.67
6. Engine 1.0 16V	0	0	0	0	0	0	1	0	1:4	27	11	0.41
7. Airbag	0	0	0	0	0	0	0	1	1:1	80	20	0.25
Requested production	0	5	10	13	17	4	11	20	$\mu = 0.5$			

2.2 Penalty function

The next step for applying the methodology is the definition of a penalty function (*P*). This function penalizes a vehicle that violates the process restrictions. This violation occurs when a given vehicle of the sequence demands the same option (*m*), without respecting the required interval (*qm*), exceeding the capacity (*pm*) of the workstation. For every option (*m*), a corresponding penalty value for the vehicle in the *i*-th position is calculated as follows:

$$cost(i,m) = P\left(m, \left[O[m,S_i] * \sum_{j=i+1}^{i+(qm-1)\le N} O[m,S_j] \right] \right) \quad (6)$$

where:

i = vehicle position in the assembly sequence (*i* = 1... 80);

S_i = the *i*-th vehicle type of the scheduling;

$O[m, S_i]$ =is 1 when vehicle type (S_i) requests the option (*m*) and is 0 otherwise.

If a smaller proximity interval for each option is defined (smaller than *qm*) it is possible to improve the positioning of penalized vehicles in the assembly line. This can be done maintaining at least one vehicle between vehicles with same options. For every vehicle not respecting this minimal interval, an additional penalty value $F[m]$ will increase its option penalty value (equation 6), according to:

$$Tcost\ (i,m) = cost\ (i,m) + F[m] \quad (7)$$

Therefore, the total scheduling cost of the (*N*) vehicles to be produced is defined as:

$$Scost = \sum_{i=1}^{N} \sum_{m=1}^{n} Tcost(i,m) \quad (8)$$

3. Genetic Algorithm Application

3.1 Chromosome representation

The chromosome length is defined according to the number of vehicles to be produced (daily) in the assembly line. For a typical real-world situation, this number is 290. Genes in the chromosome are represented using integers, in the range [0..7], indicating the vehicle type. The position of the gene in the chromosome represents the order in which the corresponding vehicle should be produced, there is, the first gene represents the first vehicle, the second gene, the following and so on until the 290th gene, which is the last vehicle to be produced.

3.2 Selection method

In the computational experiments the tournament selection method was used in association with elitism. The tournament selection method is preferred to the classical "roulette-wheel" (fitness proportionate) since it enforces less selective pressure and thus avoids premature convergence. When using elitism, the best solution of a generation is always copied to the new generation ensuring a monotonically increase of the fitness function until the convergence.

3.3 Objective function

The objective function is determined by the maximum number of penalties that can be contained a sequence of vehicles to be produced. For this problem, its computation is determined by the equation:

$$Pmax = N * m * (qmax - 1) + \sum_{j=1}^{k} \left| pr[j]_{requested} - pr[j]_{produced} \right| \quad (9)$$

where:

- N = total number of vehicles to be produced;
- m = total number of options;
- k = vehicle type (0...7) produced in this factory;
- $qmax$ = qm value of the option with highest restriction;
- $pr[j]$ = number of vehicles of type j requested/produced.

3.4 Fitness function

The fitness function is what the GA will try to optimize (by default, maximize). Since the objective function defined by equation 9 gives higher values for worse schedules, the fitness function is defined as the maximum estimated penalty subtracted from $Pmax$. In words, it simply evaluates how much vehicles of the scheduled sequence were penalized. Therefore, if no vehicle encoded in the chromosome violates the process restrictions, the optimum value obtained is the maximum estimated penalty itself. It is not necessary to normalize penalties, since all variables considered are number of vehicles.

3.5 Genetic operators

The two classical genetic operators of crossover and mutation were used. The one-point crossover (with probability p_c=1) was used to preserve the chromosome structure, avoiding building blocks breaking and inducing local search. Single-bit mutation (with probability p_m=0.07 per bit) allows the algorithm to maintain suitable population diversity in order to more evenly explore the search space.

4. Computational Simulations and Results

Four computational simulations were done to evaluate the performance of the GA system in different situations. The GA system developed here was based in the GALOPPS freeware system, version 3.2 (Goodman, 1986).

In the first simulation, the algorithm was tested in an extreme situation, where all vehicles of the schedule have some option that represents restrictions to the assembly line, and also the resources utilization level is high (μ = 0.7). As expected, it was observed that the algorithm tended to favor vehicles with lower number of restrictions than those with higher. This behavior decreases the overload in the workstations.

In the second simulation, it was included a vehicle type that doesn't present any restriction (basic model), reducing the resources utilization level to μ = 0.5. An optimal solution was obtained with all process restrictions respected.

In the third simulation, the number of vehicles to be produced was significantly increased. This experiment aimed to simulate the ideal sequence situation with ratio 2:3 (see section 2.1). There is, for each two vehicles with restrictions, three vehicles should be produced without restrictions. As result, the algorithm succeeded to schedule vehicles without violating any restriction, but the amount of produced vehicles was smaller than the requested.

In the last simulation, the AG was compared with an industrial system currently in use in the production plant, named OLGA. At this moment there is no available information about OLGA's optimization methodology. Real-world data of a typical production day (where schedule presented some process restrictions) were used and both systems were compared according their ability to obtain a schedule with the smallest possible number of violations. Table 2 shows details of the requested schedule for this simulation. Other system parameters were: N=290, k=8, m=7, m=0.33.

Table 2 – Production schedule for a usual day where 24 penalties was registered.

Options	**Vehicle types**								**Process restrictions**			
	0	1	2	3	4	5	6	7	*p:q*	*Omax*	*Onum*	*Um*
1. ABS	1	0	0	0	0	1	0	0	1:3	97	77	0.80
2. Diesel Engine type MN	0	1	0	0	0	0	0	0	1:6	48	5	0.10
3. Diesel Engine type M2	0	0	1	0	0	0	0	0	1:5	58	29	0.50
4. "Clio" type L65	1	0	0	0	1	1	0	0	1:2	145	46	0.32
5. Leather seats	0	1	0	1	0	0	0	0	1:6	48	4	0.10
6. "Clio" basic	0	0	0	0	0	0	1	0	1:1	290	116	0.40
7. "Scénic" basic	0	0	0	0	0	0	0	1	1:1	290	33	0.11
Requested production	68	5	4	18	35	11	116	33	μ = 0.33			

For this simulation, the GA converged around the 300th generation, finding a better solution than that found by OLGA. Table 3 presents a comparison between GA and OLGA.

Table 3 – Comparative results of the 2 systems

Parameter	OLGA	GA
Total number of penalties	24	22
Penalty for ABS	16	18
Penalty for Diesel engine (MN)	0	0
Penalty for Diesel engine (M2/CA)	3	0
Penalty for Clio (Sedan)	5	4
Penalty for leather seats	0	0

5. Conclusions

It was observed that the ability of the GA to find solutions (vehicle schedules for the assembly line) without penalties decreases as the assembly line resources utilization level (μ) increases. In general, the GA ability in finding good solutions it is not necessarily restricted by the size of the search space (amount of vehicles). Such ability is, in fact, precluded by the complexity of interactions among the options contained in the several types of vehicles and by the process restrictions to each option. Besides, an important consequence of increasing the search space is the growth of computational effort (time) necessary to process the algorithm, which is a sort of exponential function of the number of vehicles (N).

Based on the previous results, it was possible to verify the behavior of GA was very satisfactory, since in both situations (with low level and high level of restrictions) it has achieved an excellent performance. An important advantage of the GA over the OLGA system, is that in the case of inclusion of a vehicle that causes a penalty in the schedule, this vehicle will not be positioned soon after to the other vehicle with the same option, what avoids workstation overload. Since it is not requested the optimization takes place in real time (it is done once a day), the substitution the system current in use by a GA-based optimization system could be considered in the near future.

References

GOLDBERG, D.E., (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, Addison-Wesley.

WARWICK T., TSANG E.P.K., (1996) Tackling Car Sequencing Problems Using Generic Genetic Algorithm, *Evolutionary Computation* **3**, 3, 267-298.

GOODMAN E.D., (1996) *Genetic Algorithm Optimized for Portability and Parallelism System*. East Lansing, Michigan State University.