UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Solving the Response Time Variability Problem by means of a psychoclonal approach

Alberto García-Villoria; Rafael Pastor.

EOLI: Enginyeria d'Organització i Logística Industrial

IOC-DT-P-2008-04
Març 2008

Institut d'Organització i Control
de Sistemes Industrials

Solving the Response Time Variability Problem by means of a psychoclonal approach[†]

Alberto GARCÍA-VILLORIA[*] and Rafael PASTOR
Institute of Industrial and Control Engineering
Technical University of Catalonia (UPC)
{alberto.garcia-villoria / rafael.pastor}@upc.edu

Abstract. The Response Time Variability Problem (RTVP) is a combinatorial scheduling problem which has recently appeared in the literature. This problem has a wide range of real-life applications in, for example, manufacturing, hard real-time systems, operating systems and network environment. Originally, the RTVP occurs whenever products, clients or jobs need to be sequenced in such a way that the variability in the time between the instants at which they receive the necessary resources is minimized. Since RTVP is hard to solve, heuristic techniques are needed for solving it. In a previous study, three metaheuristic algorithms (a multi-start, a GRASP and a PSO algorithm) were proposed to solve the RTVP. These three metaheuristic algorithms have been the most efficient to date in solving non-small instances of the RTVP. We propose solving the RTVP by means of a psychoclonal algorithm based approach. The psychoclonal algorithm inherits its attributes from the need hierarchy theory proposed by Maslow and the artificial immune system (AIS) approach, specifically the clonal selection principle. In this paper we compare the proposed psychoclonal algorithm with the other three metaheuristic algorithms previously mentioned and show that, on average, the psychoclonal algorithm strongly improves the obtained results.

Keywords: response time variability, fair sequences, scheduling, psychoclonal algorithm, clonal selection, metaheuristics

1. Introduction

The concept of *fair sequence* has emerged independently from scheduling problems of diverse environments, principally from manufacturing, hard real-time systems, operating systems and networks environments. The common framework for these scheduling problems is defined by Kubiak (2004) as to build a fair sequence using n symbols, where symbol $i = 1,...,n$ is to occur given number d_i of times in the sequence. The fair sequence will be that one that allocates a fair share of positions in any prefix of the sequence to each symbol i. This *fair* or *ideal* share of positions allocated to symbol i in a sequence prefix of length k is proportional to a relative importance (d_i) of symbol i with respect to the total copies of competing symbols (equal to $\sum_{i=1}^{n} d_i$). There is not a universally definition of fairness because several reasonable metrics of fairness can be defined according to the specific problem.

The first problem in which seems to have appeared the idea of fair sequence is the sequencing on the mixed-model assembly production lines at Toyota Motor Corporation under the just-in-time (JIT) production system. One of the most important JIT objectives is to get rid of all kinds of waste and inefficiency and, according to Toyota,

[†] Sponsored by the Spanish Ministry of Education and Science's project DPI2007-61905; co-funded by the FEDER.
[*] Corresponding autor: Alberto García-Villoria, IOC – Institute of Industrial and Control Engineering, Av. Diagonal 647 (Edif. ETSEIB), 11th floor, 08028 Barcelona, Spain; e-mail: alberto.garcia-villoria@upc.edu

the main waste is due to the stocks. To reduce the stock, JIT production systems require to producing only the necessary models in the necessary quantities at the necessary time. To achieve this, one main goal, as Monden (1983) says, is scheduling the units to be produced to keep a constant consumption rates of the components involved in the production process. Miltenburg (1989) deals with this scheduling problem and he assumes that models require approximately the same number and mix of parts. Thus, he considers only the demand rates for the models. Miltenburg proposes four objective functions based on the fairness idea of scheduling the models so that the proportion of model i produced, over each time period, to the total production is as close to its ideal production as possible. That is, if the number of models is n (models $i = 1,...,n$) and the units of model i to be produced is d_i, then the total number of units to be produced (D) is equal to $\sum_{i=1}^{n} d_i$ (time periods $k = 1,...,D$) and the ideal production of model i at the period time k is $\frac{d_i}{D}k$. This problem is known as Product Rate Variation (PRV) problem (Kubiak, 1993). Kubiak and Sethi (1991, 1994) reformulated the PRV problem as an assignment problem and, therefore, it can be solved with an algorithm whose complexity is polynomial in D.

Independently of assembly lines, the fair sequencing idea has appeared in computer multithreaded systems when Waldspurger and Weihl (1995) proposed the *stride scheduling* to resource allocation in these systems. Multithreaded systems (operating systems, network servers, media-based applications, etc.) need to manage the scarce resources in order to service requests of n clients. Resources are allocated in discrete time slices (authors refer to the duration of a standard time slice as a *quantum*). Resource rights are represented by *tickets* and each client i has a given number d_i of tickets. A fair scheduling is obtained when the resources that a client has received (i.e. the number of quanta in which has been assigned) during the first k allocations, k = 1,...,D (where $D = \sum_{i=1}^{n} d_i$), are directly proportional to its ticket allocations. Thus, a client with twice as many tickets as another will receive twice as much of a resource in a given time interval. Waldspurger and Weihl suggest two metrics to evaluate the fairness of a sequence: the throughput error and the response time variability. The throughput error measures the maximum absolute deviation, for each client i and allocation k, between the resources received and the ideal resources $\left(\frac{d_i}{D}k\right)$. The problem of minimizing the throughput error can be efficiently solved using the Earliest Due Date rule defining fictitious earliest and latest due dates (see Kubiak, 2004).

The problem of minimizing the response time variability is known as Response Time Variability Problem (RTVP). Waldspurger and Weihl define the response time as the elapsed time from a client's completion of one quantum up to including its completion of next. Since the quantum duration is fixed, this definition is equivalent to the number of quanta between a client's two consecutive quantum allocations plus one. Thus, the response time variability for a client is the variance of its response times.

This metric is not exclusively useful on computer system environments. For example, in our experience with practitioners of manufacturing industries, we noticed that practitioners usually refer to a good mixed-model sequence not in terms of ideal production as it is usual in the literature (Miltenburg, 1989), but in terms of having distances between the units for the same model as regular as possible (i.e. there should not be variance in the response times of the models). Herrmann (2007) found the RTVP while was working with a healthcare facility that needed to schedule the collection of waste from waste collection rooms throughout the building. Given data about how often a trash handler needs to visit each room, the facilities manager wanted these visits to occur as regularly as possible so that excessive waste would not collect in any room. For instance, if a room needs four visits per eight-hour shift, then, ideally, it would be visited every two hours. The problem is difficult because different rooms require a different number of visits per shift. Herrmann proposed a heuristic algorithm based on the stride scheduling for solving it.

The RTVP is, unfortunately, NP-hard (Corominas et al., 2007). To solve the RTVP, Waldspurger and Weihl (1995) proposed the stride scheduling, which is a greedy heuristic algorithm. Corominas et al. (2007) proposed a mixed integer linear programming (MILP) model and five greedy heuristic algorithms. In Corominas et al. (2006), an improved MILP model is proposed (the practical limit to obtain optimal solutions is 40 units to be scheduled). Finally, García et al. (2006) proposed three metaheuristic algorithms to solve the problem.

In this paper, a psychoclonal algorithm based approach is proposed to solve the RTVP. Psychoclonal is a very new population-based metaheuristic that was first proposed by Tiwari et al. (2005). This metaheuristic inherits its attributes from the need hierarchy theory of Maslow (1954) and the artificial immune system (AIS) approach, specifically the clonal selection principle (Gaspar and Collard, 2000; de Castro and von Zuben, 2002). There are five levels of needs arranged in the Maslow's hierarchy, named physiological needs, safety needs, social needs, growth needs and self-actualization needs. Clonal selection explains the response of immune systems to non-self antigens. The cells (lymphocytes) that produce antibodies that can recognize the intruding antigens are selected to proliferate by cloning and further are undergone to an affinity maturation process that consists in hypermutations in order to obtain cells that produce antibodies that can improve their affinities to the non-self antigens. The worst cells are undergone receptor editing: cells are deleted and replaced by new ones. The whole process continues until the self-actualization level is reached.

The psychoclonal metaheuristic has yielded very good results when it has been used to solve several scheduling and combinatorial optimization problems (Prakash and Tiwari, 2005; Tiwari et al., 2005; Kumar et al., 2006a; Kumar et al., 2006b; Singh et al., 2006). The proposed psychoclonal algorithm for solving the RTVP is compared with the more efficient procedures for solving non-small instances published to date: three metaheuristic procedures presented by García et al. (2006), which are a multi-start, a GRASP (Greedy Randomized Adaptive Search Procedure) and a PSO (Particle Swarm Optimization) algorithm. On average, the psychoclonal algorithm improves strongly on previous results.

The rest of the paper is organized as follows. Section 2 presents a formal definition of the RTVP and briefly exposes the three metaheuristic procedures presented by García et al. (2006) for its solution. Section 3 describes the basic scheme of the psychoclonal metaheuristic. Section 4 proposes a psychoclonal algorithm based approach for solving the RTVP. Section 5 provides the computational experiments and the comparison with the other metaheuristics. Finally, some conclusions are given in Section 6.

2. The Response Time Variability Problem (RTVP)

The aim of the Response Time Variability Problem (RTVP) is to minimize variability in the distances between any two consecutive copies of the same symbol.

The RTVP is formulated as follows. Let n be the number of symbols, d_i the number of copies to be scheduled of the symbol i (i = 1,…,n) and D the total number of copies ($D = \sum_{i=1}^{n} d_i$). Let s be a solution of an instance in the RTVP that consists of a circular sequence of copies ($s = s_1 s_2 \ldots s_D$), where s_j is the copy sequenced in position j of sequence s. For all symbol i in which $d_i \geq 2$, let t_k^i be the distance between the positions in which the copies k + 1 and k of the symbol i are found (i.e. the number of positions between them, where the distance between two consecutive positions is considered equal to 1). Since the sequence is circular, position 1 comes immediately after position D; therefore, $t_{d_i}^i$ is the distance between the first copy of the symbol i in a cycle and the last copy of the same symbol in the preceding cycle. Let $\bar{t}_i$ be the average distance between two consecutive copies of the symbol i ($\bar{t}_i = D/d_i$). For all symbol i in which $d_i = 1$, t_1^i is equal to $\bar{t}_i$. The objective is to minimize the metric Response Time Variability (RTV) which is defined by the following expression:

$$RTV = \sum_{i=1}^{n} \sum_{k=1}^{d_i} (t_k^i - \bar{t}_i)^2 \qquad (1)$$

For example, let $n = 3$, $d_A = 2$, $d_B = 2$ and $d_C = 4$; thus, $D = 8$, $\bar{t}_A = 4$, $\bar{t}_B = 4$ and $\bar{t}_C = 2$. Any sequence is a feasible solution. For example, the sequence (C, A, C, B, C, B, A, C) is a solution, where $RTV = \left((5-4)^2 + (3-4)^2\right) + \left((2-4)^2 + (6-4)^2\right) + \left((2-2)^2 + (2-2)^2 + (3-2)^2 + (1-2)^2\right) = 2 + 8 + 2 = 12$.

As has been explained, the best procedures to date for solving the RTVP are three metaheuristic algorithms. Therefore, next these algorithms are briefly explained (for more details of the three algorithms, see García et al., 2006).

The multi-start method is based on generating initial random solutions and on improving each of them to find a local optimum, which is usually done by means of a local search procedure (Martí, 2003). Random solutions are generated as follows. For each position, a symbol to be sequenced is chosen at random. The probability of each symbol is equal to the number of copies of this symbol that remain to be sequenced divided by the total number of copies that remain to be sequenced. The local search procedure used is applied as follows. A local search is performed iteratively in a neighborhood that is generated by interchanging each pair of two consecutive symbols of the sequence that represents the current solution; the best solution in the neighborhood is chosen; the optimization ends when no neighboring solution is better than the current solution.

GRASP, designed by Feo and Resende (1989), can be considered as a variant of the multi-start method in which the initial solutions are obtained using directed randomness. The solutions are generated by means of a greedy strategy in which random steps are added and the choice of elements to be included in the solution is adaptive. The random step in the GRASP proposed by García et al. (2006) consists of selecting the next symbol to be sequenced from a set called candidate list; the probability of each candidate symbol is proportional to the value of its Webster index, which is based on the parametric method of apportionment with parameter $\delta = 1/2$ (Balinski and Young, 1982). The Webster index for the symbol i ($i = 1,\ldots,n$) is evaluated as $\frac{d_i}{\left(x_{ik} + \delta\right)}$, where x_{ik} is the number of copies of the symbol i in the sequence of length $k = 0,\ldots,D$ (assume $x_{i0} = 0$). The candidate list is composed by the symbols with greater value of their Webster index. The local search procedure applied to the initial solutions is the same local search that is applied by the multi-start method.

PSO is a population-based metaheuristic algorithm designed by Kennedy and Eberhart (1995), which is based on an analogy of the social behaviour of flocks of birds when they search for food. The population or swarm is composed of particles (birds), which have a multi dimensional real point (which represents a feasible solution) and a velocity (the movement of the point in the n-dimensional real space). The velocity of a particle is typically a linear combination of three types of velocity: 1) the inertia velocity; 2) the velocity to the best point found by the particle; and 3) the velocity to the best point found by the swarm. The PSO algorithm iteratively modifies the point and the velocity of each particle as it looks for the optimal solution. García et al. (2006) propose four PSO variations; the best of them is called *PSO-M1F*. Although the PSO algorithm was originally designed for working in a multi dimensional real space, *PSO-M1F* is adapted to work with a sequence of integer numbers that represents the solution. In this adaptation of the PSO algorithm, a point is now the sequence of integer numbers that represents a solution and the velocity is an ordered list of transformations that must be applied to the particle so it changes from its current point to another point; each transformation consists of a pair of positions of the point (sequence) to be swapped. In the case of the velocity to the best point found by the particle, this velocity is a list of transformations needed to obtain the best particle point from the current position; the case is analogous for the velocity to the best point found by the swarm. The initial points are generated as in the multi-start method.

3. The psychoclonal metaheuristic

The psychoclonal metaheuristic has been recently proposed by Tiwari et al. (2005) which has been successfully used for solving the following scheduling and combinatorial problems: the disassembly line balancing problem (Prakash and Tiwari, 2005), the assembly configuration problem (Tiwari et al., 2005), the flow shop problem (Kumar et al., 2006a), the make-to-stock inventory deployment problem (Kumar et al., 2006b) and the product mix decision problem (Singh et al., 2006).

This metaheuristic is inspired by the need hierarchy theory of Maslow (1954) and the clonal selection principle developed by Gaspar and Collard (2000). First, the Maslow's theory is explained in Section 3.1. Next, the clonal selection principle is presented in Section 3.2. Finally, the psychoclonal metaheuristic scheme is described in Section 3.3.

3.1. Maslow's need hierarchy theory

Psychologists have investigated the motivations of people behavior during their lifetime. Maslow proposed a theory, known as need hierarchy theory, which hypothesizes that the people behavior is motivated for satisfying their needs. These needs are grouped into five sets that are hierarchically arranged according to the degree of necessity. Therefore, there are five levels of needs which are represented by the Maslow's pyramid (see Figure 1). These levels are: physiological needs, safety needs, social needs, growth needs and self-actualization needs. In order to satisfy the upper levels, first the lowest levels have to be satisfied.

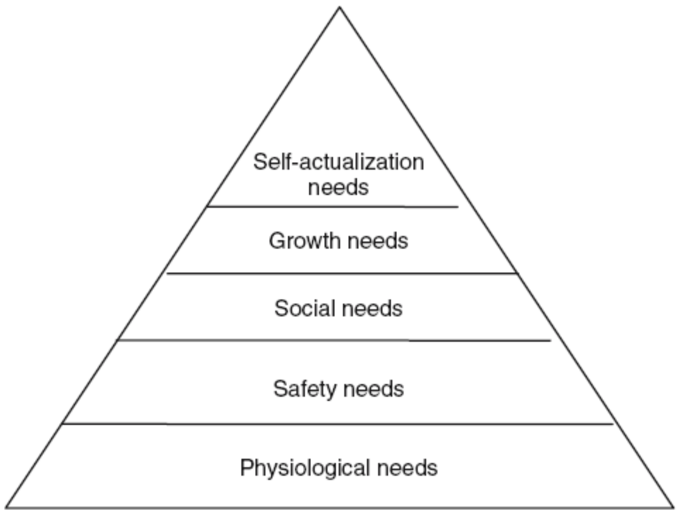

Figure 1. Maslow's pyramid

Tiwari et al. (2005) resume the five need levels as follows:

A. Physiological needs. This refers to the most basic survival needs for food, water and shelter from the environment to permit continued existence. In optimization, this corresponds to the generation of possible sequences based upon the problem environment.

B. Safety needs. The seconds set of needs has to do with physical and physiological safety from external threats to our well-beings. Constraints are inherent threats to our well-being. Constraints are inherent features of any optimization problem. Thus, external threats in the engineering perspective correspond to constraints imposed on the problem. This is where evaluation of a particular entity or candidate solution is carried out.

C. Social needs. In both the physiological and the safety needs are fairly satisfied, there will emerge the love, affection and belongingness needs. This is the need to belong to warm supportive associations of other people. In engineering, this refers to the selection of the candidate solution and the term social reflects the interaction between candidate solutions.

D. Growth needs. Every individual desires to reproduce entity of its own kind. This adds to a high valuation of personal worth. Here, candidate solutions diversify to extend the search space. This movement toward a local optimum is the basic mechanism of every evolutionary technique, e.g., cross over and mutation in genetic algorithms.

E. Self-actualization needs. The final and highest level in the hierarchy consists of needs for personal growth, for the development of one's full potential and for the fulfillment associated with the realization of an individual's capabilities. This is where the best solution is obtained. Self-actualization needs are unique and they can never be fully satisfied or fulfilled. This is very true for any optimization problem as the emphasis is always on finding the near-optimal solution rather than the global optimum. According to theory, the more self-actualization needs are fulfilled, the stronger they become. This is why a stop condition is required to decide the near-optimal solution.

3.2. Clonal selection principle

Artificial Immune Systems (AIS) are an emerging kind of computational intelligence paradigm inspired by the biological immune system of vertebrate animals. Their applications include optimization, anomaly detection, fault diagnosis and patter recognition (de Castro and Timmis, 2002). Wang et al. (2004) classify the methods based in AIS in three main categories: clonal selection principle-based, Genetic Algorithms (GA)-aided and immune networks-based approaches.

The clonal selection explains the response of the immune system of the vertebrate animals when they are attacked by foreign antigens. Immune system has lymphocytes or white cells that secrete antibodies that neutralize the foreign antigens (since a given lymphocyte only produces a single type of antibodies, in AIS there is no distinction between a lymphocyte and its antibodies). The effectiveness of the immune response depends on the affinity that has the antibodies with the antigens. The first time that the body is exposed to a given antigen, immune system has low affinity antibodies, each one with different affinity. But immune system is able to learn how to produce high affinity antibodies (it is known as reinforcement learning). The antibodies are selected to proliferate according to their affinities (clonal selection). Next, antibody clones are diversified by two mechanisms: hypermutation and receptor editing. This phenomenon is referred as maturation of the immune response. The hypermutation introduces random changes into an antibody inversely proportional to its affinity with the antigen. A large proportion of the hypermuted clones becomes more dysfunctional but, however, occasionally an effective hypermutation improves their affinity. The receptor editing deletes the lowest affinity antibodies and develops new ones through genetic recombination; in AIS, the genetic recombination is modeled creating a new antibody at random. Figure 2 illustrates hypermutation guides the affinity to a local optimum whereas receptor editing escapes from the local optimum.

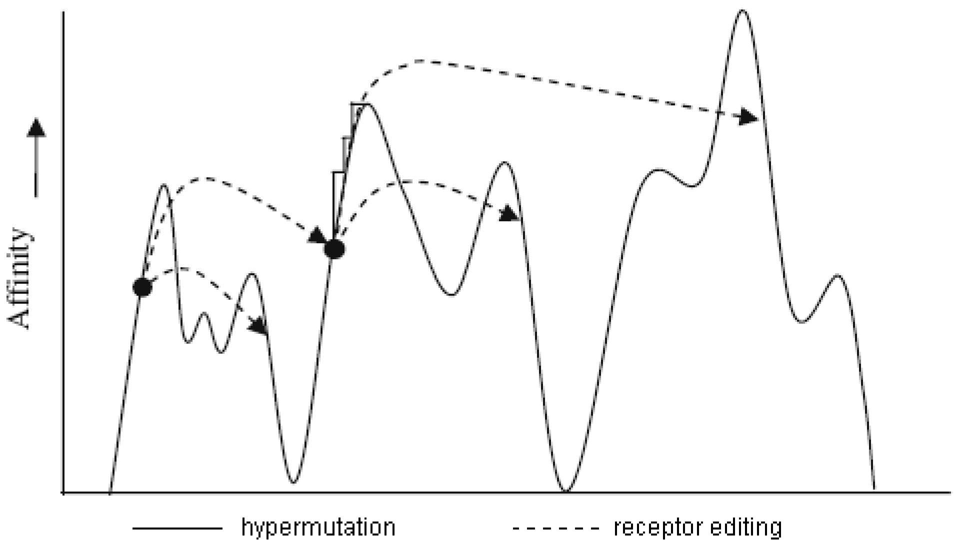

Figure 2. Representation of hypermutation and receptor editing

The reinforcement learning strategy makes that immune system continuously improving its efficiency to block foreign antigens. Since the body would be expected a given antigen more times during its lifespan, immune system keeps clones of the highest affinity antibodies (immune memory). Therefore, immune system ensures speed and accuracy in its responses across the time.

3.3. The psychoclonal metaheuristic scheme

The aim of Tiwari et al. (2005) when they developed the psychoclonal metaheuristic was to obtain a generic scheme based on the need of explotation (i.e. the local search) and the exploration (i.e. the global search) of the search space. The need hierarchy theory of Maslow and the clonal selection principle were used by Tiwari et al. (2005) to develop the psychoclonal metaheuristic, which scheme is as follows:

Need level A (physiological needs). Each antibody represents a solution. Thus, an affinity function has to be defined based on the objective function. It is also required an initial population of antibodies generated at random depending upon the environment of the problem.

Need level B (safety needs). The antibodies are exposed to the antigen, i.e. the value of their affinity function is calculated.

Need level C (social needs). An interaction is carried out between the antibodies to identify the best antibodies of the population. The best antibodies are selected and cloned proportionally to their affinity function value.

Need level D (growth needs). The generated clones are submitted for hypermutation with a rate inversely proportional to their affinity function value. After satisfaction of need level D, it is necessary to check the needs of level B (i.e. to calculate the affinity function values of the clones after hypermutation).

Need level E (self-actualization needs). The best clones are selected to be part of the new population generation. In addition, new antibodies generated at random are added to the new generation (receptor editing). The process repeats until the self-actualization is reached (e.g. a maximum number of generation or a maximum computing time).

4. The psychoclonal algorithm based approach for solving the RTVP

In Section 4.1 we design an algorithm based on the Pychoclonal metaheuristic for solving the RTVP. The algorithm has several parameters that influence in its efficiency. The selection of their values is discussed in Section 4.2.

4.1. Design of the psychoclonal algorithm

In this paper we propose an algorithm for solving the RTVP based on the psychoclonal metaheuristic.

The first consideration is the choice of the antibody representation for a solution. For the RTVP, the more intuitive representation consists in a D-length sequence of the symbols (where $D = \sum_{i=1}^{n} d_i$). The design of algorithm is explained below:

A1. The affinity function f for the antibody *ab* is defined as $f(ab) = 1/RTV(ab)$, where RTV(*ab*) is the RTV value of the solution represented by *ab*.

A2. The initial population is set by antibodies that are generated as in the multi-start algorithm. That is, for each position of the sequence (antibody), a symbol to be sequenced is chosen at random. The probability of each symbol is equal to the number of copies of this symbol that remain to be sequenced divided by the total number of copies that remain to be sequenced. The total number of antibodies that form the population is N.

B. For each antibody *ab* of the current population, $f(ab)$ is evaluated.

C. The best n antibodies according to their affinity value are selected to be cloned. The number of clones (*NC*) that are generated for each selected antibody is calculated with the following expression:

$$NC(ab_i) = \text{round}\left(\frac{\beta.N}{i}\right),\ i = 1,\ldots,n \tag{2}$$

where ab_i is the ith best antibody of the current population, *round* is an operator that rounds its argument toward the closest integer and β is a multiplying factor.

D1. The clones are submitted for hypermutation with a rate inversely proportional to their affinity value. The hypermutation rate (σ) for a clone *ab* is calculated with the following expression:

$$\sigma(ab) = \max\left(1, \text{round}\left(D.e^{-K\frac{f(ab)}{f^*}}\right)\right) \tag{3}$$

where K is the control factor of decay and f^* is the affinity value of the best antibody of the current population. The hypermutation rate indicates how many simple mutations are applied to the cloned antibodies. A simple mutation consists in choosing randomly two positions of the sequence that represents the antibody and swapping them. In order to maintain the best antibodies, we keep one original (parent) antibody unhypermutated.

D2. For each cloned antibody *ab*, *f(ab)* is evaluated.

E1. The current population is set with the (N - d)th best cloned antibodies.

E2. The current population is completed adding d new antibodies generated at ramdon as explained in step *A2*.

E3. Until the computing time of the algorithm does not reach a preset time go to step *B*.

The way that the number of clones (Equation 2) and the hypermutation rate (Equation 3) are evaluated is based on the CLONALG algorithm (de Castro and von Zuben, 2002), which is one of the most widely applied method based on AIS (Wang et al., 2004).

The algorithm that we propose has 5 parameters: N (size of the population), n (number of the best antibodies to be cloned), β (multiplying factor to calculate the number of clones of a given antibody), K (control factor of decay of the hypermutation rate) and d (the number of new generated antibodies to be added into the population). Their suitable values are discussed in the next section.

4.2. Fine-tuning of the psychoclonal algorithm parameters

Fine-tuning the parameters of a metaheuristic algorithm is almost always a difficult task. Although the parameter values are extremely important because the results of the metaheuristic for each problem are very sensitive to them, the selection of parameter values is commonly justified in one of the following ways (Eiben et al., 1999; Adenso-Díaz and Laguna, 2006): 1) "by hand" on the basis of a small number of experiments that are not specifically referenced; 2) by using the general values recommended for a wide range of problems; 3) by using the values reported to be effective in other similar problems; or 4) by choosing values without any explanation.

Adenso-Díaz and Laguna (2006) proposed a new technique called CALIBRA for fine-tuning the parameters of heuristic and metaheuristic algorithms. CALIBRA is based on Taguchi's fractional factorial experimental designs coupled with a local search procedure.

CALIBRA has been chosen for fine-tuning the psychoclonal algorithm parameters using a set of 60 representative training instances (generated as explained in Section 5). The following parameter values are obtained: $N = 25$, $n = 3$, $\beta = 1.3$, $K = 7.6$ and $d = 3$.

The GRASP and PSO algorithms (the multi-start algorithm has not parameters) are also fine-tuned with CALIBRA and the same 60 training instances. GRASP has only one parameter, which is the size of the candidate list; the obtained value is 3. *PSO-M1F* has four parameters and the following values are obtained: size of the population = 24, coefficient that weights the inertia velocity (ω) = 0.87, coefficient that weights the velocity to the best particle point (c_1) = 0.87 and the coefficient that weights the velocity to the best swarm point (c_2) = 0.37.

5. Computational experiment

The multi-start, GRASP and PSO algorithms developed by Garcia et al. (2006) are the most efficient algorithms published to date for solving non-small instances. Therefore, we compare our proposed psychoclonal algorithm with them.

The computational experiment for the four metaheuristic algorithms was carried out for the same instances and conditions used in García et al. (2006). That is, the algorithms ran 740 instances which were grouped into four classes (185 instances in each class) according to their size. The instances in the first class (*CAT1*) were generated using a random value of D (number of copies) uniformly distributed between 25 and 50, and a random value of p (number of symbols) uniformly distributed between 3 and 15; for the second class (*CAT2*), D was between 50 and 100 and p between 3 and 30; for the third class (*CAT3*), D was between 100 and 200 and p between 3 and 65; and for the fourth class (*CAT4*), D was between 200 and 500 and p between 3 and 150. For all instances and for each type of symbol $i = 1,\ldots,p$, a random value of d_i (number of copies of the symbol i) was between 1 and $\left|\frac{D-p+1}{2.5}\right|$ such that $\sum_{i=1}^{p} d_i = D$. All algorithms were coded in Java and the computational experiment was carried out using a 3.4 GHz Pentium IV with 512 Mb of RAM.

For each instance, the four metaheuristics were run for 50 seconds. Table 1 shows the averages of the RTV values to be minimized for the global of 740 instances and for each class of instances (*CAT1* to *CAT4*).

	Psychoclonal	**Multi-start**	**GRASP**	**PSO-M1F**
Global	**235.68**	**21,390.39**	**14,168.83**	**8,502.83**
CAT1	14.92	12.08	15.47	66.44
CAT2	44.25	44.36	88.48	424.60
CAT3	137.07	226.90	510.44	3,000.52
CAT4	746.48	85,278.25	56,060.92	30,519.76

Table 1. Averages of the RTV values for 50 seconds

For the global of all instances, the psychoclonal algorithm is 97.23% better than *PSO-M1F* (which is the best metaheuristic algorithm proposed by García et al., 2006), 98.34% better than the GRASP algorithm and 98.90% better than the multi-start algorithm. Observing the results in Table 1 by class, we can see that a simple algorithm such as the multi-start algorithm obtains good averages for small instances (*CAT1* and *CAT2*) but a very poor average for large instances (*CAT4*). On the other hand, *PSO-M1F* produces bad results for small and medium instances (*CAT1*, *CAT2* and *CAT3*) and better results for large ones. Finally, the psychoclonal algorithm works very well for all class of instances. For the smallest instances (*CAT1*) it is the second best shortly overcome by the multi-start algorithm. For the remaining classes, the psychoclonal algorithm obtains the best RTV averages. For *CAT2* instances, it is 0.25%, 49.99% and 89.58% better than the multi-start, GRASP and PSO algorithms, respectively. For *CAT3* instances, it is 39.59%, 73.15% and 95.43% better than the multi-start, GRASP and PSO algorithms, respectively. Finally, for *CAT3* instances, it is 99.12%, 98.67% and 97.55% better than the multi-start, GRASP and PSO algorithms, respectively.

In Table 2 we compare the number of times that each algorithm reaches the best RTV value obtained with all four algorithms. The results are shown for the 740 instances overall and for each class of instances.

	Psychoclonal	Multi-start	GRASP	PSO-M1F
Global	**456**	**242**	**196**	**0**
CAT1	61	148	102	0
CAT2	80	80	50	0
CAT3	149	14	25	0
CAT4	166	0	19	0

Table 2. Number of times that the best solution is reached

As we expect from the results in Table 1, Table 2 shows that the psychoclonal algorithm reaches the best solution the greatest number of times (in 61.62% for the global of all instances). Observing the results by class, we can see that for *CAT3* instances, although the multi-start algorithm obtains a better RTV average than the GRASP, the GRASP algorithm reaches mores times the best solution. And for *CAT4* instances the GRASP algorithm reaches more times the best solution than *PSO-M1F*, although *PSO-M1F* is better according to the RTV value average.

To complete the analysis of the results, their dispersion is observed. A measure of the dispersion (let it be called σ) of the RTV values obtained by each algorithm alg = {psychoclonal, multi-start, GRASP, *PSO-M1F*} for a given instance, ins, is defined as follows:

$$\sigma(alg, ins) = \left(\frac{\mathrm{RTV}_{ins}^{(alg)} - \mathrm{RTV}_{ins}^{(best)}}{\mathrm{RTV}_{ins}^{(best)}} \right)^2 \tag{5}$$

where $\mathrm{RTV}_{ins}^{(alg)}$ is the RTV value of the solution obtained with the algorithm alg for the instance ins, and $\mathrm{RTV}_{ins}^{(best)}$ is, for the instance ins, the best RTV value of the solutions obtained with the four metaheuristics. Table 2 shows the average σ dispersion for the global of 740 instances and for each class of instances.

	Psychoclonal	Multi-start	GRASP	PSO-M1F
Global	**0.23**	**12,562.53**	**27,928.91**	**1,433.77**
CAT1	0.79	0.05	0.71	74.30
CAT2	0.10	0.08	5.59	227.79
CAT3	0.02	0.75	17.74	903.40
CAT4	0.01	50,249.23	111,691.60	4,529.60

Table 3. Average σ dispersion regarding the best solution found

For the global of all instances, the psychoclonal algorithm has the least average σ dispersion very far from the dispersion of the other algorithms. Observing the results in Table 3 by class, we see that the behavior of the dispersions is almost analogous to the behavior of the RTV values. For the smallest instances (*CAT1* and *CAT2*), the multi-start algorithm gives the smallest average dispersion and it is near followed by the psychoclonal algorithm. For the medium and big instances (*CAT3* and *CAT4*), the psychconal algorithm shows clearly the least dispersion, followed by the multi-start algorithm for the medium instances. Although the multi-start algorithm gives the worst RTV solutions for the *CAT4* instances, it has less dispersion than the GRASP algorithm; this indicates that multi-start algorithm is more stable than the GRASP algorithm in this case. To summarize, the results in Tables 1-3 show that the psychoclonal algorithms has a very good performance in terms of the RTV values and also has a very stable behavior for all classes of instances.

The bad results for the larger instances (*CAT4*) obtained by the multi-start, GRASP and PSO algorithms may occur because 50 seconds might not be enough time for them to converge. Table 4 shows the averages of the RTV values for the global of all instances and for each class of instances (*CAT1* to *CAT4*) obtained with the four algorithms when they are run for 1,000 seconds.

	Psychoclonal	**Multi-start**	**GRASP**	**PSO-M1F**
Global	**161.60**	**1,378.58**	**1,495.12**	**6,619.34**
CAT1	14.90	10.93	13.59	66.44
CAT2	39.90	35.48	75.08	424.54
CAT3	122.38	160.67	428.86	3,000.52
CAT4	469.23	5,307.25	5,462.95	22,985.85

Table 4. Averages of the RTV values for 1000 seconds

With 1000 seconds of execution time, which seems time enough for the convergence of the four algorithms (see Figure 3), the psychoclonal algorithm is for the global of all instances 88.28%, 88.19% and 99.30% better than the multi-start, GRASP and PSO algorithms, respectively. *PSO-M1F* converges and remains trapped in a bad local optimum at the second 140 whereas the psychoclonal, multi-start and GRASP algorithms improves their RTV average during all the computing time. Although the multi-start and GRASP algorithms improve a lot their average results, the psychoclonal algorithm is clearly better. Indead, the results obtained with the psychoclonal algorithm for 50 computing seconds are much better than the results obtained with the other algorithms for 1000 computing seconds.

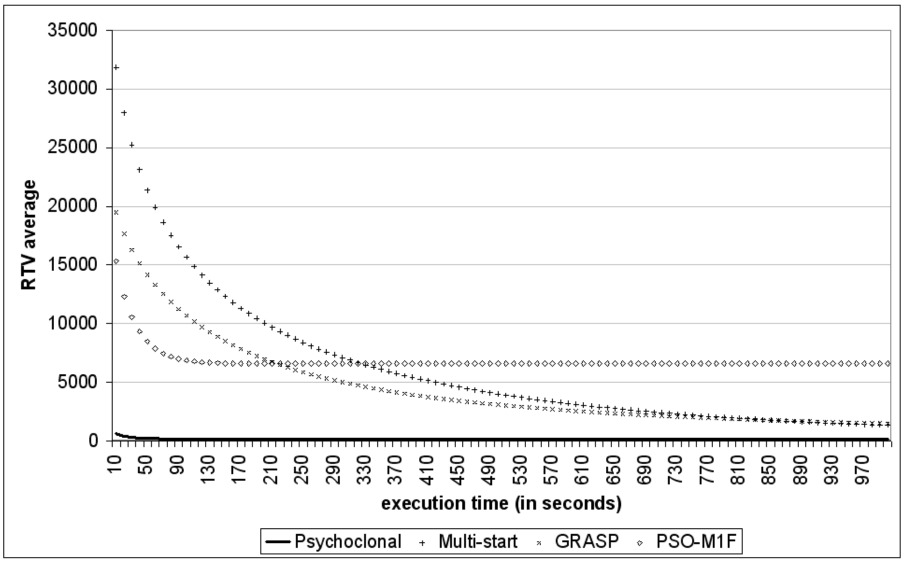

Figure 3. Average of the RTV values obtained over the computing time

6. Conclusions

In this paper, the Response Time Variability Problem (RTVP) is solved. This problem is an NP-hard scheduling problem that proposes a new metric to measure the *fairness* of a solution according to the relative importance of the different symbols to be sequenced. In the RTVP, the aim is to minimize variability in the distances between any two consecutive copies of the same symbol, i.e. to distribute the symbols the more *regular* as possible.

The RTVP occurs in diverse environments as manufacturing, hard real-time systems, operating systems and networks environments. Since it is a NP-hard problem, metaheuristic methods are needed for solving non-small instances. García et al. (2006) have proposed three metaheuristic algorithms for solving the RTVP (a multi-start, a GRASP and a PSO algorithm), which are the most efficient algorithms published to date. In order to improve the published results, a psychoclonal algorithm based approach is proposed for solving the RTVP.

The psychoclonal metaheuristic inherits its attributes from the need hierarchy theory of Maslow (1954) and the artificial immune system (AIS) approach, specifically the clonal selection principle (Gaspar and Collard, 2000). The main features of this metaheuristic are various levels of needs, affinity maturation to guide the solution to a local optimum and receptor editing to escape from local optima and to explore new regions of the solution search space.

A computation experiment was carried out and its results show that the psychoclonal algorithm that we propose improves strongly the previous results obtained with the other three algorithms on average. In addition, the psychoclonal algorithm has a very stable behavior for small, medium and large instances.

REFERENCES

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

Balinski, M.L. and Young, H.P. (1982) 'Fair Representation: meeting the ideal of one man, one vote', *Yale University Press*, New Haven CT.

Corominas, A., Kubiak, W. and Pastor, R. (2006) 'Solving the Response Time Variability Problem (RTVP) by means of mathematical programming', *Working paper IOC-DT*, Universistat Politècnica de Catalunya, Spain.

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

De Castro, L.N. and Timmis, J. (2002) 'Artificial Immune Systems: A New Computational Intelligence Approach', *Springer-Verlag*, London, United Kingdom.

De Castro, L.N. and von Zuben, F.J. (2002) 'Learning and optimization using the clonal selection principle', *IEEE Transactions on Evolutionary Computation* , Vol. 6, No. 3, pp. 239-251.

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

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

García, A., Pastor, R. and Corominas, A. (2006) 'Solving the Response Time Variability Problem by means of metaheuristics, *Special Issue of Frontiers in Artificial Intelligence and Applications on Artificial Intelligence Research and Development*, Vol. 146, pp.187-194.

Gaspar, A. and Collard, P. (2000) 'Two models of immunization for time dependent optimization', in *Proceeding of the IEEE International Conference on Systems Manufacturing and Cybernetics*, pp. 113-118.

Herrmann, J.W. (2007) 'Generating Cyclic Fair Sequences using Aggregation and Stride Scheduling', *Technical Report*, University of Maryland, USA.

Kennedy, J. and Eberhart, R.C. (1995) 'Particle swarm optimization', *IEEE International Conference on Neural Networks*, Australia.

Kubiak, W. (1993) 'Minimizing variation of production rates in just-in-time systems: A survey', *European Journal of Operational Research*, Vol. 66, No. 3, pp. 259-271.

Kubiak, W. (2004) 'Fair Sequences', In *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, Chapman and Hall.

Kubiak, W. and Sethi, S.P. (1991) 'A Note on Level schedules for mixed-model assembly lines in just-in-time production systems', *Management Science*, Vol. 37, pp. 121-122.

Kubiak, W. and Sethi, S.P. (1994) 'Optimal just-in-time schedules for flexible transfer lines', *The International Journal of Flexible Manufacturing Systems*, Vol. 6, pp. 137-154.

Kumar, A., Prakash, A., Shankar, R. and Tiwari, M.K. (2006a) 'Psycho-Clonal algorithm based approach to solve continuous flow shop scheduling problem', *Expert Systems with Applications*, Vol. 31, No. 3, pp. 504-514.

Kumar, A., Prakash, A., Tiwari, M.K. and Chan, F.T.S. (2006b) 'Stochastic make-to-stock inventory deployment problem: an endosymbiotic psychoclonal algorithm based approach', *International Journal of Production Research*, Vol. 44, No. 11, pp. 2245-2263.

Martí, R. (2003) 'Multi-start methods', *Handbook of Metaheuristics*, Glover and Kochenberger (eds.), Kluwer Academic Publishers, pp. 355-368.

Maslow, A.H. (1954) *Motivation and personality*, New York: Harper & Bros.

Miltenburg, J. (1989) 'Level schedules for mixed-model assembly lines in just-in-time production systems', *Management Science*, Vol. 35, No. 2, pp. 192-207.

Monden, Y. (1983) 'Toyota Production Systems', *Industrial Engineering and Management Press*, Norcross, GA.

Prakash, A. and Tiwari, M.K. (2005) 'Solving a dissassembly line balancing problem with task failure using a psycho-clonal algorithm', *International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, California, USA.

Singh, R.K., Prakash, Kumar, S. and Tiwari, M.K. (2006) 'Psycho-clonal based approach to solve a TOC product mix decision problem', *International Journal of Advanced Manufacturing Technology*, Vol. 29, pp. 1194-1202.

Tiwari, M.K., Prakash, A., Kumar, A. and Mileham, A.R. (2005) 'Determination of an optimal sequence using the psychoclonal algorithm', *ImechE, Part B: Journal of Engineering Manufacture*, Vol. 219, pp. 137-149.

Waldspurger, C.A. and Weihl, W.E. (1995) ‘Stride Schedulling: Deterministic Proportional-Share Resource Management’, *Technical Report MIT/LCS/TM-528*, Massechusetts Institute of Technology, MIT Laboratory for Computer Science.

Wang, X., Gao, X.Z. and Ovaska, S.J. (2004) ‘Artificial Immune Optimization Methods and Applications – A Survey’, *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 4, pp. 3415-3420.