USING EVOLUTIONARY ALGORITHMS AND SIMULATION FOR THE OPTIMIZATION OF MANUFACTURING SYSTEMS

L.Tautou and H. Pierreval

Laboratoire de Recherche en Systèmes de Production
Institut Français de Mécanique Avancée, Campus des Cézeaux, B.P. 265, F-63175 Aubière Cedex
Phone: (33) 73 28 80 00   Fax: (33) 73 28 81 00
e-mail: tautou@ifma.ifma.fr and pierreva@ifma.ifma.fr

Abstract

This paper is concerned with an optimization method for manufacturing systems. This method is able to optimize problems with any type of variables (variables from a real set e.g. conveyor speed, an integer set e.g. size of buffer or any general set e.g. dispatching rules). It is based on the association of an evolutionary algorithm (EA) and a simulation model. Extensions of Michalewicz's genetic operators and algorithm are proposed to tackle manufacturing system problems. This method is applied to an example: the configuration of a workshop producing plastic yoghurt pots. The criterion to optimize is the cost of the workshop and the three variables are the size of a silo, the size of a warehouse and a choice between two manufacturing methods. The application has been realized by connecting a modified version of Michalewicz's Genocop software (for EA) and a simulation language.

1 Introduction

Due to the present tough economic competition, modern manufacturing systems need to provide high performance at the lowest cost. Thus, particular attention has to be paid to the selection of values for the different factors which influence costs and performances. Those factors can be relative to the configuration of the physical system (e.g. a number of machines, a size of a warehouse, a choice between several machines or a flexible one) or management parameters (e.g. dispatching rules, storage policies, a number of Kanban). This can be addressed by optimizing a criterion, that is, to accurately choose the values of the n variables, X_i, of a vector $X=(X_1, X_2, ..., X_n)$, where the X_i variables can take values from a part of the real set (e.g., speed of AGV), a part of the integer set (e.g., number of places in a warehouse) or in any general set E (e.g., choice between several dispatching rules). Such an optimization approach requires the development of a method able to take into account the different types of variables, the complexity of the system and the fact that the various variables to optimize are not independent.

In this paper, we propose to choose X_i values by optimizing a simulation model of the system studied with an evolutionary algorithm. The method proposed here allows various types of variables (quantitative or qualitative variables). We will first introduce several optimization simulation methods, then we will introduce evolutionary algorithms (EA) and propose an adaptation of these algorithms to the optimization of manufacturing systems. This approach is then illustrated with the example of the configuration of a workshop producing plastic yoghurt pots, in order to minimize its cost. We will finally describe the application of the method and the results we obtained.

2 Simulation Optimization Methods

A simulation optimization problem is an optimization problem where the objective function is a response evaluated by simulation. It may be formulated as follows:

$$\min_{X \in D}(f(X))$$

where f is the criterion evaluated by simulation, $X=(X_i)_{i=1..n}$ is the vector of variables, such as each variable X_i is an element of a real, integer or qualitative domain ($D = \underset{i=1..n}{\otimes} D_i$).

Several research studies have been carried out on simulation optimization. In tutorials on simulation optimization, such as Azadivar (1992), Fu (1994), Kleijnen (1994) four major classes of approaches can be distinguished: gradient based search methods, stochastic approximation methods, response surface methodology and heuristic methods. Such methods have been applied in various application fields, such as manufacturing systems (Rosenblatt and al, 1994) or economic systems (Kleijnen, 1988). Basically, the aim of each of these approaches is to propose a strategy to explore the solution space D, with a limited number of simulation experiments. Most of the existing techniques are adapted to the cases where the domains D_i are real intervals, due for example, to the computation of a gradient, of a response surface or a regression metamodel. Therefore, such methods can not handle qualitative variables (e. g., queuing strategies), which limits considerably their potential applications. Moreover, in many cases several of these methods can be sensitive to local extremums, due to the exploration strategy they use (e. g., design of experiments and gradient methods). Finally, certain of these methods are not easy to use or to implement in simulation packages (e. g., restriction of D to a smaller D' where a metamodel is valid, factorial design of experiments and building of a response surface). They may require expertise in statistics.

To cope with those drawbacks, we proposed to use EA , which have not been mentioned by Azadivar (1992) and Fu (1994). The use of EA has been quoted by Szczerbicka (1994) or Biethahn (1993) as a possible solution for simulation optimization.

3 Optimization Using Evolutionary Algorithms

3.1 General Principles

Genetic algorithm (GA) concepts, developed by John Holland (1975), are inspired by natural evolution phenomena. One of the essential principles of those phenomena is that only the strongest individuals, i.e. the best adapted to their environment, survive. These algorithms begin their search for the optimal solution from a set of potential solutions (individuals or chromosomes), which is called the population. Each chromosome is a vector whose components (genes) are binary elements. GA makes an initial population evolve towards a population, which is expected to contain the best solution. For that, they use the following reproduction-evaluation cycle: for each iteration called a generation, individuals (represented by chromosomes) from the current population are selected with a given probability and copies from these individuals are created. Selection of individuals is based on their fitness relative to the current population, i.e. the strongest individuals will have a higher probability of reproducing. Fitness is determined by the objective function we wish to optimize. New individuals, produced by this process are submitted to the mutation and the crossover genetic operators that we will describe later on. This process is called reproduction. From generation to generation, the algorithm converges towards a population which contains the optimal solution to the problem (more details can be found in Michalewicz, 1992).

Several authors have extended GA, which are henceforth classified as evolutionary algorithms (EA). These extensions concern in particular, for example, coding, genetic operators or selection techniques (Janikow and Michalewicz, 1992), Mühlenbein, 1993). Axelsson (1993) proposes to use EA for sizing a turbine with real value variables. We propose subsequently to use EA for production systems optimization. The algorithm proposed is based on the following general schema (Mühlenbein, 1993).

Step 0. define a genetic representation of the problem.

Step 1. create an initial population $P(0)=X^1, X^2,..., X^N$.

Step 2. evaluation: compute the fitness $f(X^i)$ for each individual X^i, i=1..N.

Step 3. selection: select individuals from P(t), this give the set S(t). The same individual from P(t) can appear several times in S(t).

Step 4. recombination: pair individuals of S(t) and for each pair of individuals:
- apply crossover to the pair with probability p_{cross} and copy the offspring in set S(t+1) (the pair of individuals is eliminated and replaced by its offspring),
- copy the pair of individuals in set S(t+1) with probability $1-p_{cross}$.

For each individual of S(t+1):
- apply mutation to the individual with probability p_{mut} and copy transformed individual in P(t+1),
- copy the individual in P(t+1) with probability $1-p_{mut}$.

Step 5. set t=t+1 and return to step 2 until we reach our stopping criterion.

3.2 Application of EA to Simulated Production Systems.

In production systems, the nature of the variables (or genes) X_i, which make up chromosomes are varied because a chromosome corresponds to a feasible configuration of the production system. Thus, we will consider that a chromosome can contain integer genes (representing, for example, a buffer size, a number of AGVs), real value genes (e.g., a silo size, a duration) and qualitative genes (e.g., a choice between stock management modes: SPT, EDD, manufacturing solution). The fitness of a chromosome, computed by simulation, will be the value of the optimization criterion corresponding to the configuration coded in the chromosome. To take into account the variety of gene types, we propose to extend the crossover and the mutation genetic operators defined by Michalewicz (1992).

For the crossover and the mutation operators, we randomly chose an integer *pos* between 1 and *n*, n being the number of genes in a chromosome.
Reproduction with the crossover operator is made as follows: let two chromosomes b and c be submitted to the crossover:

$$b = b_1\, b_2 \ldots b_{pos} \mid b_{pos+1} \ldots b_n$$
$$c = c_1\, c_2 \ldots c_{pos} \mid c_{pos+1} \ldots c_n$$

After the crossover, the chromosomes *b* and *c* become *b* ' and *c* ':

$$b' = b_1\, b_2 \ldots b_{pos}\, c_{pos+1} \ldots c_n$$
$$c' = c_1\, c_2 \ldots c_{pos}\, b_{pos+1} \ldots b_n$$

Reproduction by the mutation operator is made as follows: let a chromosome *b* be submitted to the mutation:

$$b = b_1\, b_2 \ldots b_{pos}\, b_{pos+1} \ldots b_n$$

After the mutation, the chromosome b becomes b':

$$b' = b_1\, b_2 \ldots b'_{pos}\, b_{pos+1} \ldots b_n$$

The definition of the element b'_{pos} is different depending on pos values, three cases can appear:
(1) pos corresponds to a real value gene, b'_{pos} is a real value randomly chosen in the corresponding domain D_{pos}.
(2) pos corresponds to an integer gene, b'_{pos} is an integer randomly chosen in the corresponding domain D_{pos}.
(3) pos corresponds to a qualitative gene, we randomly chose a position pos' in the domain of qualitative elements, b'_{pos} is the element corresponding to pos' in the corresponding domain D_{pos}.

4 An Illustrative Example

4.1 Problem formulation

This problem is related to the design of a workshop which manufactures printed plastic yoghurt containers. Containers are extruded and cast with granulated plastic material. Trucks supply the silo with this raw material when the level in the silo is under a given threshold. Then the silo supplies the raw material necessary to produce plastic sheet by a system of pipes leading to the extrusion machine for manufacturing a batch of rolls of plastic sheet. Before containers are cast, rolls must be stocked if the machine is currently being used. Two operations are necessary to cast containers. Those operations can be made in two ways: the first one (dedicated solution) uses two machines separated by a buffer, the second (flexible solution) uses a single flexible machine which is able to carry out both operations. Two types of products (P1 and P2) are processed. The P1 and P2 products are both stocked in a warehouse called S before being sent for printing. If S is full, new products wishing to enter are crushed into small grains until places in the warehouse become idle. The P1 and P2 product printing is done by specific machines. A description of the manufacturing process is given in figure 1.

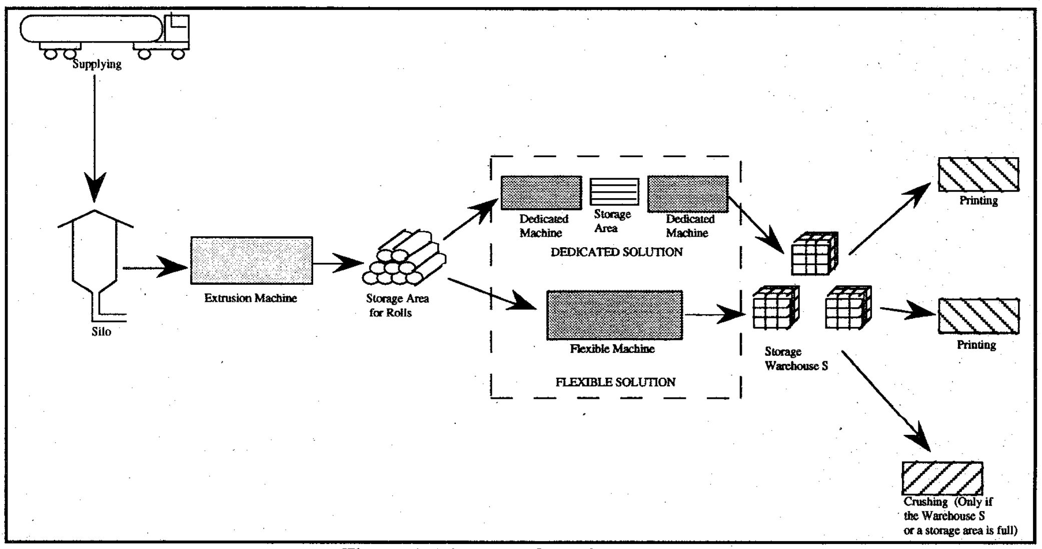

Figure 1: The manufacturing process

The first problem to address is to determine the best silo capacity for the problem. In fact, if the silo capacity is too high, it is too expensive to purchase. On the other hand, if it is too small, this would necessitate a lot of deliveries (which are expensive because of the purchase of raw material and the price of the travel of a truck). The second problem is to choose the most advantageous solution between the dedicated one and the flexible one. In fact, the dedicated solution is more expensive than the flexible one, but gives better performance. Then, the final problem is to size the warehouse S where the P1 and P2 products are stored. Printing machines must work permanently (no stock outage) but stock must also be not too large, otherwise storage cost will be too high. The warehouse S must have a sufficient size so as not to be saturated too often and to avoid crushing the surplus of pallets produced.

4.2 Cost Optimization

Our purpose is to minimize the cost of this workshop. The cost considered here is the sum of the following elements:

(1) cost of machines: we only consider the purchase price of machines which make up the dedicated solution and the flexible solution,
(2) cost of silo: we consider that the price of the silo only depends on its capacity,
(3) raw material consumption: that is, the cost of truck travels added to the raw material cost,
(4) jobs in process: we assume the cost of the jobs in progress to be the same at each stage of the production process,
(5) cost of the warehouse S: we assume that this cost is proportional to its size,
(6) crushing: we will distinguish between different crushing types: we consider that each stock corresponds to a crushing type. The total cost of crushing will be given by the following formula:

$$\text{total cost of crushing} = \sum_{i=1,2} N(i)C_b(i)$$

with i=the number representing a specific stock inventory,
N(i)=the total number of crushed pieces for the crushing i,
$C_b(i)$=unit cost of crushing i.

The optimization problem is given in the following form: find the vector X° in domain D such that X° minimizes the function f(X) with the following definitions:

$$f(X) = C_{mach}(X_2) + C_s X_1 + nb(X)(C_c + C_{mat} X_1) + C_e T(X) + \sum_{i=1,2} C_b(i, X_2) N(i) + C_{stock} X_3$$

with the notation which follows:

1- *The problem variables* are:
- X_1=the silo capacity, X_1 is a real value in the interval [cap_mini;cap_maxi],
- X_2=the name of the employed solution, X_2 is an element of the element set {the dedicated solution, the flexible solution},
- X_3=the size of warehouse S, X_3 is an integer in the interval [taille_mini..taille_maxi].

2- *The search domain* is:
- D=[cap_mini;cap_maxi]x{the dedicated solution, the flexible solution}x[taille_mini..taille_maxi] where x denotes the cross product.

3- *The problem constants* are:
- $C_m(X_2)$=the purchase price of machines of solution X_2,

- C_s=the purchase price of a silo with an unit capacity,
- C_c=the cost of a truck's travel,
- C_r=the unit purchase price of raw material,
- C_e=the storage cost of jobs in processes,
- $C_b(i,X_2)$=the unit cost of crushing of type i with solution X_2,
- C_{st}=the cost of a place in storage S,

4- *The values determined by the simulation* are:
- n(X)=the number of times we must fill the silo,
- T(X)=quantifies the stock evolution during τ (the time necessary to carry out production). T(X) is computed as follows:

$$T(X) = \int_0^{\tau} Y(t)dt$$

where Y(t) is the number of pieces in the total stock at time t.

5 Application to the Example

In the problem studied , a chromosome corresponds to a possible configuration of the workshop. It is a vector made up of three elements, the first one is the silo capacity (real value), the second one is the size of buffer S (integer value) and the third one is the chosen manufacturing solution (qualitative value). The fitness of each chromosome, that is, the cost of the proposed configuration, would be very difficult to be analytically evaluated because of the problem nature (e. g., interaction between variables, blocking phenomena). To evaluate this cost, we give three values X_1, X_2 and X_3 as input to a simulation, whose response is the cost for the production of 10 000 pallets. The total time of the simulation τ is the time needed to produce 10 000 pallets.

Results given in the following section have been obtained with the method presented in the preceding sections. We have used a small population size in order to reduce the computing time. The initial population is randomly chosen in the domain D. We have noted that the minimum found in our tests is reached for populations of at least 15 individuals. However, when we increase population size, the minimum is globally reached faster, as can be seen in table 1.

Population size	Mutation probability	Crossover probability	Generation number of mini. cost
15	0,15	0,15	498
15	0,20	0,10	111
15	0,10	0,20	500
30	0,15	0,15	290
30	0,20	0,10	30
30	0,10	0,20	112

Table 1: The application of the EA

We note that the result is not improved after iteration number 500 for the examples given here. Increasing the frequency of crossover (or mutation) applications does not lower further the minimum cost, but it does modify the convergence speed as shown in figure 2. These remarks are verified for populations of 15 or 30 individuals.

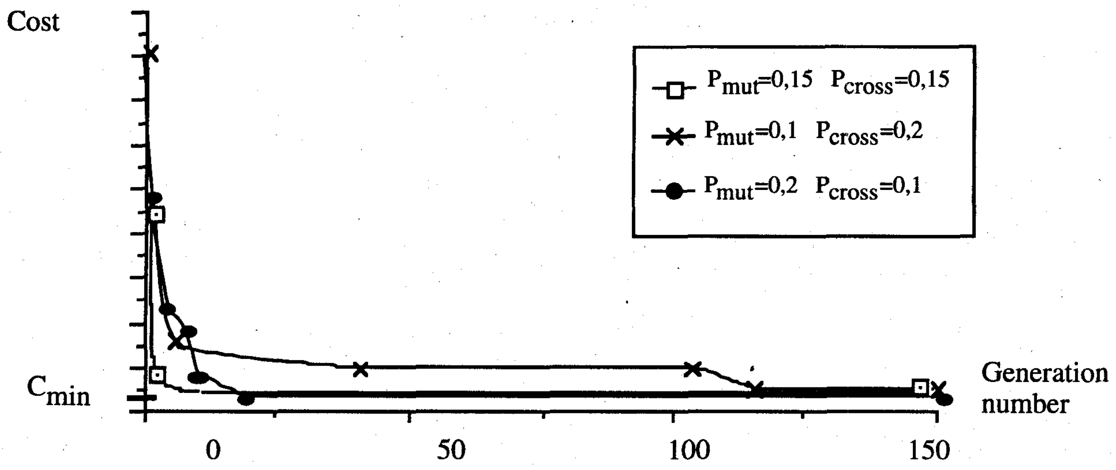

Figure 2: The convergence of the EA

This example is based on a deterministic simulation model. Using a stochastic simulation model is possible but requires statistical tests to compare the objective function (see Mebarki, 1995 for more details).

6 Conclusion

In this paper, we have proposed a new approach in the simulation optimization of manufacturing systems. This approach, based on evolutionary algorithms, allows the optimization of variables of several types of problem because we are not limited by the types of variables. This technique is suited to a wide range of application areas, and therefore appears to be a very interesting tool to optimize and configure simulated systems. This approach is also interesting because it is easy to parallelize. We are now interested in a parallel version of this approach (Pierreval and al, 1995) in order to speed up the search for the optimum value.

References

AXELSSON, J., MENTH, S., SEMMLER, K., 1993, Genetic algorithms in industrial design, *Tools with artificial intelligence TAI'93*, 64 - 67.

AZADIVAR, F., TALAVAGE, J.J., 1980, Optimization of stochastic simulation models, *Mathematics ad computers in simulation*, **22**, 231 - 241.

AZADIVAR, F., 1992, A tutorial on simulation optimization, *Proceedings of the 1992 Winter Simulation Conference,* 198-204.

BIETHAHN, J., NISSEN, V.,1993, Combining simulation and evolutionary algorithms for applications in complex economic systems, *Proceedings of modelling and simulation conference*, Lyon june 7-9, 351-356.

CAUX, C., 1993, Analyse et spécification de systèmes de production pour l'évaluation des performances de la recherche en ordonnancements rapport de thèse n°D.U. 512.

CAUX, C., PIERREVAL, H., PORTMANN, M.C., 1995, Les algorithmes génétiques et leur application aux problèmes d'ordonnancement, à paraître dans APII.

FU, M.C., 1994, Optimization via simulation a review, à paraître dans *Annals of operational research.*

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

HOLLAND, J.H., 1975, *Adaptation in Natural and Artificial Systems* (Univ. of Michigan Press, Ann Arbor).

JANIKOW, C.Z., MICHALEWICZ, Z., 1991, An experimental comparison of binary and floating point representations in genetic algorithms, *Proceedings of the Fourth International Conference on Genetic Algorithms*, 31 - 36.

MEBARKI, N., 1995, Une approche d'ordonnancement temps réel basée sur la sélection dynamique de règles de priorité, Rapport de thèse de doctorat, Université Claude Bernard Lyon I.

MICHALEWICZ, Z., 1992, *Genetic Algorithms + Data Structures = Evolution Programs* (Springer - Verlag).

MÜHLENBEIN, H., 1993, Evolutionary Algorithms: Theory and Applications, soumis à Local search in combinatorial optimization, EHL Narb, JK Lenstia (ed.Wiley).

PIERREVAL, H., TAUTOU, L., BZEZNIK, B., 1995, Parallel evolutionary algorithms for the simulation optimization of manufacturing systems, Eurosim Congress'95, September 11-15, Vienna, Austria.

PORTMANN, M.C., 1994, Part I: Genetic Algorithms in Scheduling Using Genetic Algorithms and Neural Networks,

PFLUG, G.C., 1984, Optimizing simulated systems, *Simuletter*, **15** (4), 6 - 9.

ROSENBLATT, M.J., ROLL,Y., ZYSE,V., 1993, A combined optimization and simulation approach for designing automated storage/retrieval systems, *IIE Transactions*, **25** (1), 40 - 50.

SYRJAKOW, M., SZCZERBICKA, H., 1994, Optimization of simulation models with REMO, *Proceedings of the Conference on Modelling and Simulation 1994*, (eds. Guash and Huber), 274-281.

SZCZERBICKA, H., 1994, Are Genetic Algorithms a panacea for optimization of simulation model ?, *Proceedings of the Conference on Modelling and Simulation 1994*, (eds. Guash and Huber), 38-46.