Genetic Algorithms for the scheduling of multiproduct batch plants within uncertain environment

Antonin Ponsich,[a] Anna Bonfill,[b] Antonio Espuña,[b] Catherine Azzaro-Pantel,[a] Serge Domenech,[a] Luc Pibouleau[a]

[a]*Laboratoire de Génie Chimique de Toulouse UMR5503 CNRS/INP/UPS, 5 rue Paulin Talabot BP1301, 31106 Toulouse Cedex1, France, Catherine.AzzaroPantel@ensiacet.fr*
[b]*Department of Chemical Engineering, Universitat Politècnica de Catalunya, ETSEIB, Avenida Diagonal 647, E-08028 Barcelona, Spain, antonio.espuna@upc.edu*

Abstract

This study addresses the problem of batch plant scheduling. In addition, uncertainty on product demands is considered through probabilistic-based methods. In the resulting two-stage stochastic programming problem, the objective is to maximize an Expected Profit Value (EPV) while respecting a constraint forcing the makespan to be lower than a time horizon. A Genetic Algorithm (GA) is proposed for the solution of a multiproduct example. The variable encoding requires special attention. Computational tests are first carried out with a deterministic model to validate the GA efficiency. Then, different runs with different scenario sets highlight the existence of various solution classes, characterized by specific numbers of batches manufactured for each product. Further analysis finally enables to discuss if each schedule is really the best-fitted to the scenario set for which it has been determined.

Keywords batch plant scheduling, stochastic programming, genetic algorithms

1. Introduction

The problem of batch plant scheduling under demand uncertainty is addressed in this work. In nowadays highly dynamic environments, uncertainty and variability have also become inherent characteristics of process systems, which are essential to be considered for modelling purposes. When uncertain

phenomena are taken into account in the preliminary study phases of design/scheduling, a better flexibility is assigned to the system, in order to cope with changes occurring in both technical and economical environments. Uncertainty in industrial systems can be modelled by either fuzzy logic concepts or through probabilistic-based approaches, deriving in stochastic programming problems [1]. The latter relies on scenario sets design and derives in mixed-integer linear models solved by Mathematical Programming techniques. However, metaheuristics may be well-fitted to stochastic integer programming [5] and would allow tackling non-linear models.
In this study, a Genetic Algorithm is proposed to solve the scheduling problem under uncertain market demands. The model and the solution methods are presented in sections 2 and 3. Computational results and analysis are proposed in section 4 and some conclusions are provided in section 5.

2. Model implementation

The adopted formalism describes a typical multiproduct batch plant, following a Zero-Wait policy. The formulation states that some amounts of P products have to be manufactured in J operating stages. Furthermore, uncertainty on product demand is introduced in order to formulate a two-stage stochastic model: in the first stage, the "here-and-now" decision variables are set to determine a particular schedule; this latter is evaluated in the second stage in which uncertainty is implemented and uncertain parameters are known. In the initial formulation [2], the decision variables were:

- the number of batches of each product to be manufactured, in order to satisfy the market demand: $NBatch_i$, $i = \{1,\ldots,P\}$;
- the product sequence of the schedule: $IndBatch_k$, $k = \{1,\ldots,K\}$;
- the corresponding starting and finishing times of all operations J and for all batches K: Tin_{jk}, $j = \{1,\ldots,J\}$, $k = \{1,\ldots,K\}$.

In the resulting Mixed Integer Linear Programming (MILP) problem, the objective function is the Expected Profit Value (EPV), accounting for sales, expenses (production/inventory/unsatisfied demand costs) and some additional terms (penalizing both product changeovers in a sequence and high starting times). The EPV is computed according to s scenarios (defined with normal distribution laws) and their associated probability ($\omega_s = 1/ NScen$). The exact objective function, variables and model equations are defined in [2]. The schedule makespan is constrained to be lower than a horizon time H=168 h.
The complete model was adapted in a simple scheduling simulator for multiproduct batch plants. With a Zero-Wait policy, starting or finishing times of every batches in every processing stage are not necessary to describe completely a solution. So, these dates are not considered as optimisation variables any more and the problem size is reduced.

3. Development of a specific Genetic Algorithm

The used optimisation tool is a classical Genetic Algorithm, implemented in previous works [4]. The method basic principles will not be recalled here. The technique used for selection is the classical roulette wheel, and the fitness is simply the objective function since a maximization case is assumed. The EPV is computed according to *s* scenarios that must be generated at the beginning of the run and kept unchanged during the whole search. Constraints are handled through elimination of the infeasible individuals.

However, the application to the scheduling problem under uncertainty involved some adaptations for variable encoding, and thus for genetic operators (crossover and mutation). Since starting and finishing times are no more considered as decision variables, the chromosome is divided into two zones: the first one encodes the manufactured batch number for each product, while the second one encodes the schedule sequence. The main issue is that the size of the second part depends on the first part values. So, the commonly used permutation-based encoding of a schedule sequence needs further adaptation.

Firstly, concerning the chromosome part representing the number of batches for each product ($NBatch_i$, $i=\{1,\ldots,NProducts\}$), a classical binary coding was chosen. The bit number allocated to each product will subsequently define an upper bound for the number of manufactured batches. This upper bound corresponds to the size of the chromosome second part. For this latter part, an integer-gene representation was adopted. Each gene is associated to a possibly existing batch, and its position in the chromosome corresponds to that of the batch position in the sequence. The gene value is equivalent to a product identifier; if equal to zero, then the batch does not exist. This second zone could finally be transformed by shifting all the "zero-genes" towards the end of the chromosome. Figure 1 gives an illustration of the encoding technique.

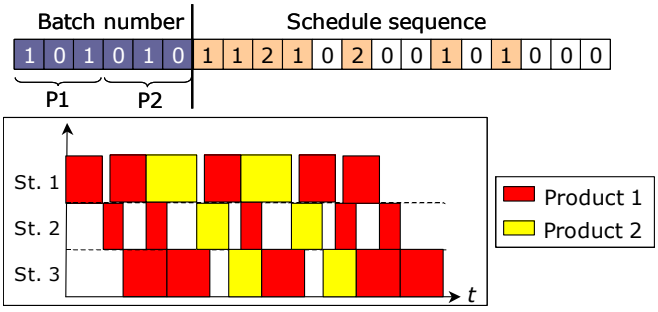

Figure 1. Encoding technique for a two-product, three operating steps schedule

However, this representation mode means that the random generation of an initial individual or the creation of individuals by crossover or mutation may derive in meaningless chromosomes: for instance, the values of the first part are likely to represent a batch number which is not consistent with the second zone

content. Consequently, a repairing method is essential to correct the infeasible chromosomes and make them consistent with respect to the used coding. The implemented method will not be developed here. A random generation of the initial population can thus be carried out. In the same way, since any crossover and mutation method can be performed, very generic strategies were adopted. A classical two-point crossover method was implemented. Concerning mutation:

- for the binary (batch number) zone, inversion of the bit value;
- for the integer (sequence) part, another gene of the same zone is randomly selected and both gene values are exchanged if they are different. If the genes are equal, another gene is selected until an effective change is noted.

4. Computational results

The solution method is applied to a didactic example having 5 products synthesized in 3 operating steps. All the corresponding data are available in [2]. In a first step, the deterministic model (i.e. having the demand for all products set to a nominal value) is solved; then, classical stochastic computations were carried out, simulating 100 scenarios to evaluate the objective function.
It is to note that, since GAs are a stochastic method, 20 runs were carried out for each scenario set. The repeatability of the results will help to prove the statistical quality of the method. The GA's parameters, set on the basis of sensitivity analysis, are the following ones: population size = 200; generation number = 200; survival rate = 60; mutation rate = 40.

4.1. Deterministic operating mode

This first computation enables to validate the good performances of the GA. The found solution is identical to that proposed in [2] with the *CPLEX* solver (from *GAMS* modelling environment [3]): 4508 monetary units. The makespan (152 h.) is quite lower than the time horizon. Furthermore, in both cases, the associated EPV, computed with 100 scenarios decreases to 1686 (*CPLEX*) or 1755 (GA): this proves the low ability of the deterministic schedule to adapt to any kind of demand. The slight difference (=4.09%) between the two EPV values is due to the difference between the involved scenario sets. Under these nominal conditions, the found solution is characterized by a number of batches by product equal to $NBatch_{i=\{P1,\ldots,P5\}} = \{3, 2, 3, 2, 3\}$.
With regard to repeatability, eleven of the twenty GA runs succeeded in finding the optimum located by *CPLEX*. Concerning the nine remaining runs, they all lie very close to the optimal solution (the gap is always lower than 0.5%).

4.2. Stochastic computations

Since computational times were not restrictive (7 s. per run), 20 different scenario sets were independently generated. Furthermore, the GA was run 20

times for each scenario set and the best solution of these 20 runs was recorded as the solution of one global run. This results in 20 *global runs* x 20 *tests* = 400 *runs*. Various solutions were found and the results were classified according to the number of batches synthesized for each product. Six classes of solutions were identified and presented in table 1, but one of them is found by half-part of the 20 global runs. However, another solution is found for 25% of the scenario set (*Stoch2*). Like in deterministic computations, the slight gap between *CPLEX* and *Stoch1* EPV are due to the different scenario sets used to solve the problem, with the consequences on the "second-stage" part of the objective function.

Table 1. Stochastic results ([a] Results from [2])

	CPLEX[a]	*Stoch1*	*Stoch2*	*Stoch3*	*Stoch4*	*Stoch5*	*Stoch6*
EPV	2140	2165	2036	2130	1672	2059	2052
$PV_{nom.}$	3059	3058	3054	2951	3334	3487	2618
$NBatch_i$	4 2 4 3 3	4 2 4 3 3	4 3 4 2 3	3 3 4 3 3	4 3 3 3 3	4 2 4 2 3	4 3 4 3 3
Mks (h)	167	166	166	167	167	167	167
Run rate (%)		50	25	10	5	5	5

Besides, a random effect is introduced by the scenario generation and another one by the optimisation method. In other words, it is difficult to assume that a good EPV is due to a profitable scenario set or to the proper GA efficiency. To overtake this inaccuracy, some selected schedules are evaluated according to the scenario sets associated to the other chosen solutions. This procedure will enable to check out if one of the results found by the GA is really better than the other ones whatever the scenario set, or only for the set it was computed with.
Ten solutions among those previously found were arbitrarily chosen (*Stoch1a* to *Stoch1d*, *Stoch2a*, *Stoch2b*, *Stoch3a*, *Stoch4a*, *Stoch5a*, and *Stoch6a*) and the associated scenario sets were recorded (*Scn1a*,… *Scn6*). Then, each schedule was evaluated according to each scenario set. Computation results were summed up in figure 2: each box with coordinates *(i,j)* corresponds to the schedule $Stoch_j$ evaluated according to the scenario set Scn_i. The reported value is the relative difference (in %) between each new EPV and the initial EPV computed with the solution $Stoch_i$. Thus, a negative value means that schedule *j*, if artificially applied to scenario set *i*, is a worse solution than the schedule determined thanks to the GA ($Stoch_i$). The results clearly show that, except for solution *Stoch6a*, no substantially positive value can be reported. The direct interpretation is that, basically (apart from some exceptions), each schedule is the best-fitted to the scenario set for which it was found. This point proves that:

- the GA is really efficient and manages to find highly adapted solutions to each problem, defined by a particular set of scenarios;
- a good solution for 100 scenarios is not that good for 100 other scenarios. In the treated problem, the five uncertain demand distributions all have a mean value μ such as $120 \leq \mu \leq 300$, and a standard deviation being 50% of μ.

Consequently, the number of 100 scenarios may not be sufficient to provide a reliable basis on which a flexible production planning can be assessed.

5. Conclusions

This study proposed a Genetic Algorithm for the optimisation of the multiproduct batch plant scheduling problem under uncertainty. The adaptation of GA's internal procedures mainly focused on the encoding method. The found results were really satisfying concerning the GA's efficiency that solved accurately the deterministic problem. For the stochastic operating mode, the GA located performing solutions for all the tested scenario sets within reasonable computational times, which proved to be adapted to the associated scenarios. These conclusions also highlighted that 100 scenarios might not be enough to get a really representative sample of the uncertain parameter space.

Figure 2. Relative reliability of the computed solutions on the simulated scenarios

Scenarios \ Schedules	Stoch1a	Stoch1b	Stoch1c	Stoch1d	Stoch2a	Stoch2b	Stoch3a	Stoch4a	Stoch5a	Stoch6a
Scn1a	0.00	0.12	0.12	0.16	-10.01	-10.05	-6.03	-15.47	-8.12	-2.65
Scn1b	-0.21	0.00	0.00	-0.05	-2.59	-2.65	-0.73	-8.30	-3.84	0.83
Scn1c	-0.13	0.00	0.00	0.04	-4.83	-4.87	-1.73	-17.76	-4.83	-0.35
Scn1d	-0.25	-0.05	-0.05	0.00	-4.46	-4.51	-7.14	-13.04	-3.32	-1.59
Scn2a	-0.23	-0.09	-0.09	-0.05	0.00	-0.05	-1.59	-3.27	-3.45	2.99
Scn2b	-0.90	-0.75	-0.75	-0.70	0.05	0.00	-0.65	-0.25	-5.16	4.01
Scn3a	-7.55	-7.39	-7.39	-7.34	-3.34	-3.39	0.00	-15.04	-9.80	-1.33
Scn4a	-2.51	-2.33	-2.33	-2.27	-6.76	-6.82	-5.08	0.00	-6.82	-6.12
Scn5a	-5.78	-5.63	-5.63	-5.59	-0.10	-0.15	-6.07	-15.10	0.00	-2.75
Scn6a	-8.41	-8.27	-8.27	-8.22	-9.43	-9.48	-8.78	-12.55	-13.38	0.00

[2;5]
[0;2]
0
[-2;0]
[-5;-2]
[-10;-5]
[-20;-10]

6. References

1. J.R. Birge and F. Louveaux, Introduction to stochastic programming, Springer (1997).
2. A. Bonfill, M. Bagajewicz, A. Espuña and L. Puigjaner, Ind. Eng. Chem. Eng., 43 (2004) 741-750.
3. A. Brooke, D. Kendrick, A. Meeraus and R. Raman, GAMS User's Guide; GAMS Development Corporation (1998).
4. A. Ponsich, C. Azzaro-Pantel, S. Domenech and L. Pibouleau, MINLP optimisation strategies for batch plant design problems, Ind. Eng. Chem. Res., Under Press.
5. J. Till, G. Sand and S. Engell. 16th European Symposium on Computer Aided Process Engineering, Marquardt and Pantelides (Eds), Elsevier (2007), 1967-1972.