Multiobjective Evolutionary Optimization of Batch Process Scheduling Under Environmental and Economic Concerns

Elisabet Capón-García
Dept. of Chemistry and Applied Biosciences, ETH Zürich, Zürich 8093, Switzerland

Aarón D. Bojarski, Antonio Espuña, and Luis Puigjaner
Dept. of Chemical Engineering, CEPIMA, Universitat Politècnica de Catalunya, ETSEIB, Barcelona 08028, Spain

DOI 10.1002/aic.13841
Published online June 1, 2012 in Wiley Online Library (wileyonlinelibrary.com).

The simultaneous consideration of economic and environmental objectives in batch production scheduling is today a subject of major concern. However, it constitutes a complex problem whose solution necessarily entails production trade-offs. Unfortunately, a rigorous multiobjective optimization approach to solve this kind of problem often implies high computational effort and time, which seriously undermine its applicability to day-to-day operation in industrial practice. Hence, this work presents a hybrid optimization strategy based on rigorous local search and genetic algorithm to efficiently deal with industrial scale batch scheduling problems. Thus, a deeper insight into the combined environmental and economic issues when considering the trade-offs of adopting a particular schedule is provided. The proposed methodology is applied to a case study concerning a multiproduct acrylic fiber production plant, where product changeovers influence the problem results. The proposed strategy stands for a marked improvement in effectively incorporating multiobjective optimization in short-term plant operation. © 2012 American Institute of Chemical Engineers *AIChE J,* 59: 429–444, 2013
Keywords: LCA, multiobjective optimization, evolutionary algorithms, multiproduct batch scheduling, product changeover, sequence dependency

Introduction

Tighter economic demands and increasingly stringent safety and environmental policies in process industries require more sophisticated decision-making tools. Thus, shorter periods of time for decision-making are imposed by market and process uncertainty affecting all enterprise levels. Specifically, the scheduling function, which is related to the organization of the human and other resources used in a company and directly linked to the satisfaction of customer demands, significantly affects the company results.

Environmental, safety, and economic concerns are the basis for most of the criteria that can be used to analyze the scheduling problem. In practice, the decision maker often faces the situation of choosing a given solution under different and conflicting objectives such as cost, performance, reliability, safety and productivity among others.[1] The absence of a universal and unique criteria to establish the goodness of a given schedule increases the complexity of the decision-making process, because a large number of solutions may be suitable. Therefore, the goal of management is frequently multifold and prefers to be guided through a set of good alternatives available rather than a single "best one" proposal. The analysis of the decision maker's alternatives under conflicting objectives is performed by means of multicriteria decision analysis. Therefore, efficient multiobjective optimization strategies, which can generate different alternatives for decision analysis in production scheduling, stand a great chance of adding competitive advantage to business.[2]

In the literature, several authors present alternative methodologies to account for engineering, economic, and environmental trade-offs arising in batch process scheduling. However, most of existing contributions are problem oriented. Two different approaches are usually adopted, namely, (1) techniques based on mathematical programming and (2) techniques based on heuristics and metaheuristics.

Mathematical programming approaches are extensively used for solving scheduling problems; remarkable examples are reviewed by Mendez et al.[3] The most important feature provided by these approaches is that optimal solutions can be proved global under certain problem circumstances (e.g., linearity, convexity, casting the problem into a MILP). However, the combinatorial nature of the scheduling problem poses serious computational difficulties when using mathematical rigorous approaches. Therefore, it is widely recognized that full-space mathematical programming approaches cannot tackle long scheduling time horizons or with many batches to be assigned to many resources, that is, problems of important size.[4] Even more, the integration of multiobjective issues at the operational level increases the problem complexity. Therefore, the generation of multiobjective optimal solutions for scheduling problems, which entail almost immediate results for day-to-day decisions, can only be

Additional Supporting Information may be found in the online version of this article.

Correspondence concerning this article should be addressed to L. Puigjaner at luis.puigjaner@upc.edu.

provided for models of limited complexity given the capabilities of the current software–hardware systems.[5]

Despite the former issues, several authors discussed possible approaches to tackle economic and environmental considerations associated with the scheduling problem. In practice, different methods can be applied to study a given multiobjective problem. Methods such as optimizing a single criterion at a time, goal programming, and physical programming have been proposed to deal with this type of problem.[6] However, the most accepted technique for tackling with multiobjective problems consists of the Pareto-based evaluation,[7] where a vector containing all the objective values represents the solution fitness. The generally accepted solution of a multiobjective problem is known as the Pareto front (PF), which encompasses a set of solutions, for which any improvement in one objective can only take place if at least one other objective worsens.[6] Some authors[8] identify their Pareto optimal solutions reformulating their mathematical problem as a parametric programming problem; however, in this work, the strategy of filtering of the solutions obtained from the optimization has been adopted.

The work of Song et al.[9] considers the scheduling problem, modeled by a MILP formulation, of a refinery process taking into account the environmental impact. The ε-constraint method is used to obtain a set of Pareto solutions for the multiobjective optimization that considers global environmental impacts by means of the critical surface-time 95 assessment methodology. Berlin et al.[10] consider a case study of the dairy industry, where the production sequencing affects the environmental impact from a life-cycle perspective. They developed a heuristic method to minimize production waste based on production rules. Their methodology is further applied by Berlin and Sonesson[11] to a case study with two dairy products. The authors conclude that the environmental impact of processing cultured milk products can be greatly reduced by adopting sequences with fewer changes of product. Park et al.[12] present a goal constrained programming algorithm for the multiobjective optimization with priority for the scheduling of cutting papers, and various optimal schedule sets are provided. Our former work[5] aimed at providing a robust algorithm for the generation of reliable Pareto fronts while considering different metrics.

Due to the former considerations for the rigorous optimization regarding limitations of software–hardware systems and the inclusion of multiobjective formulations, the application of evolutionary optimization presents itself as an alternative to efficiently solve large-scale problems at the cost of not ensuring the assessment of global optimality. Many works have used different evolutionary algorithms, for example, Coello-Coello and Landa-Becerra[13] provide with a review of these methods. Specifically, different works tackle the batch scheduling problem using metaheuristic multiobjective optimization techniques. For example, the ant colony optimization (AC) was applied by Jayaraman et al.[14] for the optimal design and scheduling of batch chemical processes of a multiproduct batch plant. Arnaout et al.[15] have also used AC optimization for minimizing makespan on unrelated parallel machines with sequence-dependent setup times; they divided the scheduling problem into two subproblems: assignment and sequencing, each of which was solved using different ant trails collaboratively. The use of other metaheuristics such as simulated annealing (SA)[16] and particle swarm optimization[17] for solving the scheduling problem has also been exemplified.

One of the most used techniques in metaheuristic optimization, which is well suited to address any type of problem regardless of its structure, consists of the genetic algorithm (GA). This technique has been applied for solving the problem of unrelated parallel machine scheduling, in some cases including planning[18] or other features such as job sequence- and machine-dependent setup times.[19] He and Hui[20] present a heuristic approach based on GA for solving large-size multistage multiproduct scheduling problem in batch plants suitable for different scheduling objectives, such as total process time and total flow time. They present techniques that greatly reduce unnecessary search space and increase the search speed using a penalty method for handling the constraints in the problem, which avoids the infeasibility during the GA search and further greatly increases the search speed.

Other authors combine different optimization heuristics, as for example in the work of Ponnambalam and Mohan-Reddy[21] where SA is combined with GA, where SA serves as a local optimizer. In this context, it is important to note the inclusion of discrete-event simulation (DES) model to represent dynamically the production system behavior as in Azzaro-Pantel et al.,[22] or the replacement of such DES by an artificial neural network.[23,24] Other developments include fuzzy logic[25] or probabilistic[26] representations for different variables in the treatment of uncertainty.

While mathematical programming approaches coupled with rigorous optimization solvers profit of model derivatives and/or higher-order model information, derivative-free optimization (DFO) approaches such as evolutionary algorithms and others,[27] jointly known as metaheuristics, rely mainly on the model's objective function evaluation. Consequently, in these approaches the scheduling problem can be considered as a black box model and no information regarding first- or higher-order derivatives is required, and only the objective function value and some constraints satisfaction is checked. Even more, many metaheuristic formulations are inherently multiobjective, including the Pareto efficiency concept in the way new solutions are tested and gathered. However, one important drawback on metaheuristic optimization is the fact that the solution method is based on procedural search techniques, and the violation of constraints is handled through penalty functions. Therefore, it may be difficult to obtain actual feasible solutions for highly constrained problems, because the penalty functions may lead to accept solutions that do not fulfill the problem constraints.

This work is intended to assist in the systematic and efficient production scheduling under consideration of economic and environmental impact, proposing a global general algorithm computationally efficient. The detailed problem statement is referred to the one described in Ref. 5. The scheduling problem is formulated using mathematical programming but solved using a multiobjective genetic algorithm on top of a classical optimization software, which allows the implementation of an efficient optimization procedure. This second layer optimization problem will be different depending on the objective functions being studied and on the degree of freedom that the GA allows for the integer solutions. Therefore, if the binary variables are fixed by the multiobjective optimization problem, then problem constraints are checked as a feasibility problem of the mathematical formulation and continuous variables, such as processing times and makespan, are obtained from the resulting second layer problem. It is also proposed to allow some freedom for

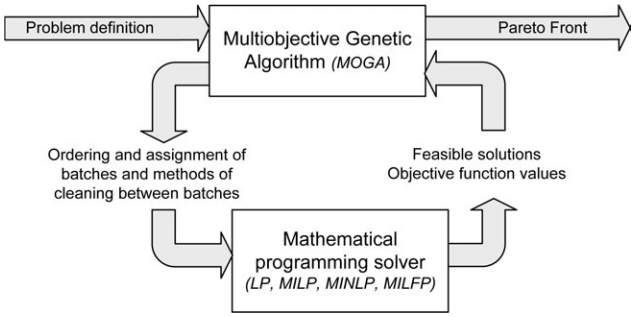

Figure 1. Overall algorithm outline.

changing the integer decisions, because the second layer acts as a local search optimizer using a bit change method. The proposed methodology is illustrated through a case study based on a multiproduct batch facility producing acrylic fibers.

Methodology

This work represents a comprehensive step over the previously presented approaches by systematically assisting in the production scheduling under different economic and environmental impact considerations. The following sections include a brief discussion of the objective functions adopted in this work and the features of the proposed multiobjective genetic algorithm.

Overall algorithm

The solution strategy involves the use of the multiobjective genetic algorithm (MOGA) to create and improve a set of possible solutions defined by the integer decisions related to batch assignment, sequencing, and cleaning method. Therefore, such decision variables are properly coded to implement the MOGA. Next, the resulting solutions are transformed into binary variables to be passed to the mathematical programming formulation, so that the continuous variables such as processing times and makespan are evaluated for those decisions. This step also allows for the problem constraints to be checked by the mathematical solver. Moreover, the mathematical programming model provides with the objective function values, that the GA uses for its selection operator. Thus, a rigorous local search may be applied, to improve the solution quality, see Figure 1.

The scheduling problem is modeled by a mathematical formulation based on the immediate precedence concept,[28] which has been adequately extended to consider product batching, different interbatch cleaning methods and additional objective functions (e.g., makespan, productivity, and environmental impact). The resulting mathematical formulation is thoroughly described by Capón-García et al.[5] In the model, a single unit per stage is assumed, and the major decisions concern: (1) the batch allocation, that is, whether a batch i of product p should be performed, represented by the binary variable W_i, producing the quantity specified by the batch size parameter $\overline{\mathrm{BS}_i}$; and (2) the batch sequencing and interbatch cleaning method represented by the binary variable $X_{ii'c}$, which is 1 if batch i precedes batch i' and the cleaning method c is used.

Objective functions

Different objective functions have been used in this work to measure different aspects of the problem at hand, specifically economic and environmental metrics. However, the proposed methodology can be applied considering other typical scheduling objective functions based on time metrics, such as makespan, earliness, and tardiness.

Environmental Impact. Environmental aspects are being incorporated into the design of chemical processes due to pressures from regulation policies and a global trend toward sustainability in businesses.[29] A wide range of process design frameworks including environmental considerations have been proposed: the methodology for obtaining minimum environmental impact process (MEI methodology),[30] the waste reduction algorithm[31] proposed by the United States Environmental Protection Agency, which uses the pollution balance concept and the environmental fate and risk assessment tool,[32] are only some representative examples. Most of these methodologies embed the concepts of life cycle assessment (LCA).[33] Within LCA, the overall life cycle of a process or product is analyzed, taking into account upstream and downstream flow in the process from cradle to grave. Consequently, the LCA technique is also selected in this work as the best suited environmental tool for process development and optimization.

The implementation of a LCA for a given process or product requires the gathering of data regarding process environmental interventions (e.g., raw material consumption, uncontrolled emissions, and waste generation). This set of data is organized in a life cycle inventory (LCI), which is the basis for the environmental impact calculation. Specifically, if only scheduling decisions are considered, the LCI is directly related to product recipes and product changeover procedures. Within this model, and to avoid emissions double counting, raw material emissions are not aggregated to product manufacturing, similarly cleaning environmental interventions are considered separately.

The total environmental impact, which includes both the batch i production process environmental impact ($\overline{\mathrm{EnvIm}_i}$) and batch changeover between i and i' at stage k using cleaning method c environmental impact ($\overline{\mathrm{EnvIm}_{ii'kc}}$), is expressed by means of Eq. 1, whereas relative environmental impact can be obtained by dividing the total environmental impact by the total produced quantity Eq. (2).

$$z^{\mathrm{ei}} = \sum_{i,i',c|i\neq i'} X_{ii'c} \sum_{k} \overline{\mathrm{EnvIm}_{ii'kc}} + \sum_{i} W_i \overline{\mathrm{EnvIm}_i} \tag{1}$$

$$z^{\mathrm{rei}} = \frac{z^{\mathrm{ei}}}{\sum_i W_i \overline{\mathrm{BS}_i}} \tag{2}$$

Economic Throughput. Economic criteria are of utmost importance in the process industry. Hence, multiple economic objectives can be adopted in process scheduling, depending on the decision maker preferences, which stem from industrial demands. For this work, the total profit, which considers batch i product benefits ($\overline{\mathrm{BP}_i}$) and changeover costs between batches i and i' using cleaning method c at each stage k ($\overline{\mathrm{ChCost}_{ii'kc}}$), is defined by Eq. 3. Thus, the profitability Eq. (4), which results from dividing the total profit by the production schedule makespan (Mk), is considered.

$$z^{\mathrm{profit}} = \sum_{i} \overline{BP_i} W_i - \sum_{i,i',c|i\neq i'} X_{ii'c} \sum_{k} \overline{\mathrm{ChCost}_{ii'kc}} \tag{3}$$

$$z^{\mathrm{profitability}} = \frac{z^{\mathrm{profit}}}{\mathrm{Mk}} \tag{4}$$

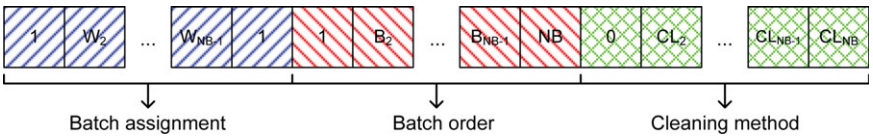

Figure 2. String representation of a solution.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

The former objective functions are either linear (Eqs. 3 and 1) including also the makespan calculation or fractional (Eqs. 4 and 2). Therefore, insomuch as the mathematical scheduling model is formulated using linear constraints, the objective function will determine the problem type. Specifically, if a linear objective function is considered, the resulting scheduling problem is a mixed-integer linear programming (MILP). In contrast, for a fractional objective, a mixed-integer non-linear programming (MINLP) problem must be solved.

It must be pinpointed that, in this case, the nonlinearity is only associated to the objective functions and not to the scheduling model, whose constraints are all linear. Thus, given the fact that the nonlinear objective functions are the ratio of two linear functions, such MINLP problems may be further classified as a special type of MINLP, namely mixed-integer linear fractional program (MILFP). Because of the nonlinear nature of the MILFP problems and the combinatorial complexity of the scheduling problem, the resulting model may result computationally intractable specially for large-size instances. Therefore, You et al.[34] extend the Dinkelbach's algorithm, which originally exploits the relationship between nonlinear fractional programming and nonlinear parametric programming to solve convex nonlinear fractional problems, to obtain the global optimal solution of an MILFP problem by solving a sequence of MILP subproblems. As a result, large-scale MILFP problems can be easier handled than with standard MINLP methods.

The former objective functions (OFs) are the phenotypes of our genetic algorithm, in all cases we are using two of the former OFs to describe any of the solutions proposed, rendering the problem multiobjective.

Multiobjective genetic algorithm

As mentioned before, the scheduling problem is formulated using mathematical programming tools, but it is solved using a multiobjective genetic algorithm, considered as an efficient optimization strategy. The proposed solution strategy involves the creation and improvement of a set possible solutions defined by the integer decisions related to: batch assignment, batch sequencing, and cleaning method. Unit assignment decisions are not presented in this work, because the case of a single unit per stage is studied. However, such feature may be directly considered by introducing an additional part to the solution representation corresponding to the unit assignment. Therefore, such decision variables have been properly coded to implement the multiobjective genetic algorithm and encompass the solution genotype.

Next, the resulting decisions are transformed into binary variables and passed to the mathematical program, so that the continuous variables, such as processing times and makespan are evaluated for those decisions, and the problem constraints, such as demand satisfaction and time horizon, are consequently checked. The inner level mathematical program optimization can be run in two different modes: (1) one where integer variables are fixed rendering an LP using the values that the GA has set, and using a LP solver, such as CPLEX, as feasibility solution tester; and (2) other where a local search strategy based on a bit-code heuristic is used to improve the solutions by changing the value of the binary variables as explained in the following sections. In this last case if linear functions are used then a MILP is solved for each objective function, whereas, if a nonlinear objective function is selected then a MINLP or a sequence of MILPs are solved using the algorithm proposed for MILFPs.

The main features of the proposed multiobjective genetic algorithm are discussed next.

MOGA Solution Representation. According to the genetic algorithm heuristic technique, each possible solution to the problem is referred to as an individual of the population, which represents the whole set of solutions to be evaluated, that is, its phenotype is to be calculated. The information regarding each individual genotype is characterized in its chromosome formed by different genes. In this work, the production schedule is encoded as a chromosome that consists of a string divided into three genes represented by vectors (see Figure 2). The length of each gene (i.e., vector) corresponds to the total number of batches (NB) associated with the final products that may be performed including the initial and final still-state. The maximum number of batches that can be assigned is provided by the decision maker as an upper bound related to the maximum product demand and the batch size, based on previous knowledge or by the problem constraints. The first and last vectors represent the ordered set of batches, whereas the second vector contains the permutation of the number of batches, defining the order in which batches are processed.

Specifically, the first vector represents the decision of the batches to be performed, and it directly corresponds to the binary variable W_i of the mathematical formulation. Therefore, if a given batch i is processed, its position in the vector (i) contains a value of 1 ($W_i = 1$); otherwise, its value is 0, consequently its alleles are 0 or 1. In the chromosome second vector, the first and last entries correspond to the initial and final batches, representing the still-state. The other positions in the vector represent the sequence in which batches are performed, consequently its values that is, alleles are integers from 1 to NB-1. The last vector of the chromosome contains the interbatch cleaning method (CL_i) that precedes each batch, and its alleles range from 1 to the number of possible interbatch cleaning methods available. It must be noted that the batch representing the initial still state does not have any precedence cleaning method.

As a result, if a number of NB batches are to be produced then the chromosome representing a given set of decisions will have NB · NB · NB values, see Figure 2. The second and third vectors of the chromosome correspond to the binary variable $X_{ii'c}$ of the mathematical formulation, which stands for the assignment of cleaning method c to changeover when batch i is produced immediately before batch i'.

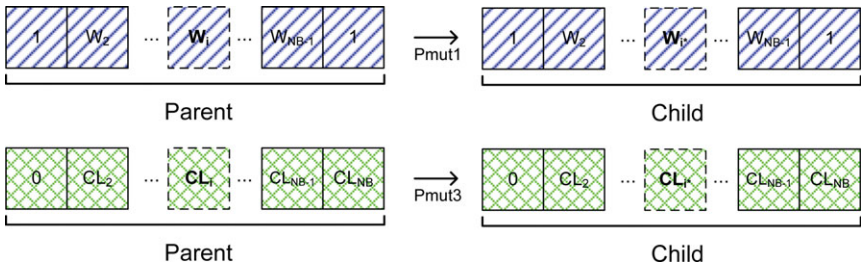

Figure 3. Mutation operation for a given individual.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Feasible Integer Solution Generation. A simple algorithm is adopted for generating solutions that constitute feasible sequences and satisfy the minimum demand. In the first vector of the chromosome, the first and last batches corresponding to the still state as well as a minimum number of batches to fulfill the demand are compulsory and are fixed to have value $W_i = 1$. The remaining batches are performed or not, using a random assignment criterion. The second vector of the chromosomes, which can contain any integer value from 1 to NB is constructed by assigning random values to each batch position and ordering them according to such value. The value assigned is the order that each of them receives when sorted, so this procedure allows for generating sequences where integer values would not repeat themselves. Anyhow, the first and last positions of this vector are fixed to the batches representing the still state. The final vector of the individuals is initialized by randomly assigning a cleaning method to precede each batch.

The former guidelines for generating the chromosomes allow for disregarding a large amount of possible solutions, which will otherwise be generated: solutions where minimum demand is not achieved, or where batches have the same sequence order. Therefore, the adopted representation and its generation creates sequence feasible solutions of the scheduling problem, because the binary variables of the mathematical formulation are completely defined. Thus, the batches allocation, sequence, and cleaning method of each solution can be directly introduced in the mathematical formulation, so that timing constraints may be checked and the value of the continuous variables, such as starting and finish times of the operations, makespan, and costs, can be obtained by solving a LP problem.

Genetic Algorithm Operators. The optimization strategy based on the genetic algorithm consists of improving the individuals of the population at each iteration, that is, generation. Therefore, for a given population, a number of individuals is selected, and the operators are applied to these individuals, so-called parents, to generate their off-spring. According to Coello et al.,[35] three operators are applied: mutation, recombination, and selection. As for the aforementioned solution representation, different operators are appropriate to be applied to different chromosome sections. In the first vector, the mutation has been adopted to choose the assignment of those batches that are optional to fulfill the minimum demand, otherwise integer solutions will be generated, which are *a priori* known to be unfeasible. The second vector of the chromosome is replicated by means of several variations of the recombination operator: switching between any two genes, inversion of the genes order between any two points, and the crossover between two parents. The third part of the chromosome evolves singly using the mutation operator. A proportion of P_{mut} individuals of the total population is obtained by the application of the mutation operator, and another proportion (P_{ran}) is generated using the feasible integer solution generator and is considered to be random, while the remaining individuals are generated using recombination (P_{rec}). Thus, the amount of individuals generated by each operator is defined *a priori* by setting the former proportions ($P_{\text{ran}} + P_{\text{mut}} + P_{\text{rec}} = 100\%$). The operators applied to the parents solutions pool are explained next:

- *Mutation*. A random gen of the chromosome is selected to be mutated, and its value is changed (see Figure 3). The chromosome genes that are susceptible for mutation are those related to the first and third gene vectors. Mutation only affects a single gen.
- *Switch of two batches*. This operator is applied to the gene coding the batches sequence. A batch is switched position with other providing a new sequence as shown in Figure 4. Two randomly selected genes of the vector are exchanged, and consequently, only two batches are modified in sequence. The amount of population individuals generated using this operator is P_{swi}.
- *Inversion of batches between two points*. This operator consists of inverting the order of the genes between two randomly chosen positions (see Figure 5). It is applied to obtain a percentage P_{pos} of the whole population, and the number of batches modified in sequence using this operator depends on the chosen sequence positions.
- *Chromosomes crossover*. This operator is applied using two parents. The children are generated by choosing a random position in the chains (i_1 and i_2) of the respective parents. Consequently, four possibilities arise where each

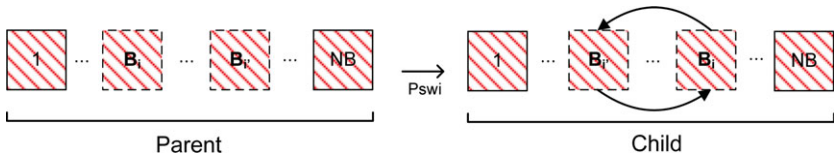

Figure 4. Switch of two positions operation.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

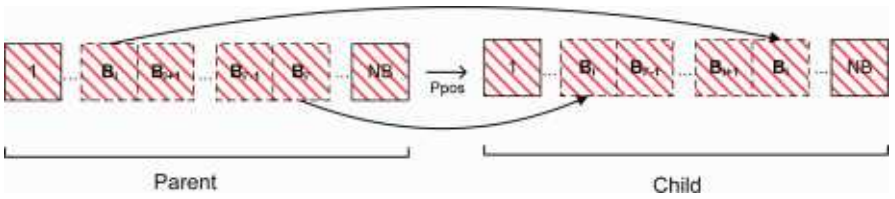

Figure 5. Reverse operator between two positions operation.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

child preserves half of the parent sequence without changes (B_{i_1} and B_{i_1+1} for Parent 1, whereas B_{i_2} and B_{i_2+1} for Parent 2). The rest of the sequence chain for each child is fulfilled with the lacking batches in the order that appear in the other parent (see Figure 6). In this sense, R_{i_1+1} contains the sequence of batches from Parent 2 that are not considered in B_{i_1}. The former way of performing the sequence crossover guarantees that feasible sequences will be generated. Only two randomly chosen children are selected and the amount of individuals total proportion of P_{crx} individuals is generated by choosing the same amount of pairs of individuals of the total population.

Finally, the selection operator is used to generate the mating pool for the next generation. In this case, instead of allowing all the population to be selected as parents based on roulette selection rules, only the solutions that belong to the Pareto front and are closer to it are used. The individuals are classified according to the proximity to the Pareto frontier. The individuals pertaining to the Pareto front are known as PF^1. To obtain the set of second best individuals, a filter is performed disregarding the individuals belonging to PF^1, thus obtaining PF^2. Similarly, the other ith best individuals can be classified by disregarding during the filtering procedure the $(i-1)$th previously selected individuals, which in general terms can be considered of higher rank. From these different sets of solutions (PF^i), the mating pool is generated, by selecting the number of PFs (N_{PFs}) that are included or not. It must be noted that in all cases PF^1 is the problem solution and is the one that is the most interesting to generate.

Multiobjective Genetic Algorithm Procedure. The description of the steps of the genetic algorithm procedure applied in this work are detailed next.

Initialization. The very first step consists of initializing the population. Hence, the construction of a given set of individuals is performed using the criteria previously described for integer feasible solution generation.

Objective functions evaluation and ranking criterion. Once the individuals of the population are defined, the objective functions must be calculated and possible unfeasibilities regarding timing constraints must be checked. Next, the goodness of the individuals must be measured by feeding each individual from the entire solution pool to the mathematical program and solve the problem considering the integer set fixed. This step serves two purposes, checks the feasibility of each solution, and allows for assigning to each individual their different objective function values. In the case of a single objective optimization, the value of the objective function could be used as performance metrics. However, in multiple objective optimization, the criterion of proximity to the Pareto frontier is selected to rank the individuals.

Rigorous Local Search. Based on a local branching strategy, the solutions of the mating pool could be further improved using mathematical optimization. Therefore, the solutions corresponding to those individuals are sent as initial solution to a mathematical scheduling problem solver. Specifically, the values of the binary variables W_i and $X_{ii'c}$ are introduced as parameters pW_i and $pX_{ii'c}$, respectively. There, a maximum number of changes in these integer variables is allowed by means of Eqs. 5 and 6. Due to the mathematical formulation any change in the batch performance (W_i) entails at least three changes in the sequence and cleaning variable values ($X_{ii'c}$). Consequently, two integer parameters are added for constraining the amount and type of changes allowed regarding the solution sent by the GA: N_X^{changes} is the number of changes allowed to the sequence binary variables $X_{ii'c}$, while N_W^{changes} refers to the batching binary variables.

$$\sum_{i|pW_i=0}(W_i - pW_i) + \sum_{i|pW_i=1}(pW_i - W_i) \le N_W^{\text{changes}} \tag{5}$$

$$\sum_{i,i',c|pX_{ii'c}=0}(X_{ii'c} - pX_{ii'c}) + \sum_{i,i',c|pX_{ii'c}=1}(pX_{ii'c} - X_{ii'c}) \le 3 * N_W^{\text{changes}} + N_X^{\text{changes}} \tag{6}$$

Specifically, it is proposed to analyze the eventual improvement of the solution by optimizing these problems considering all objective functions, one at a time. As a result, as many new solutions as objective functions are obtained from each individual stemming from the mating pool. These new solutions are considered together with their parents (which in this case are the initial individuals that started the local search) and filtered to obtain the improved mating pool. As this optimization may be time consuming, this local search is applied once every N_{ls} number of generations and only to the mating pool solutions. Furthermore, the amount of time that is left for the optimizer to solve the local search is also fixed to $T_{\text{lim}}^{\text{ls}}$.

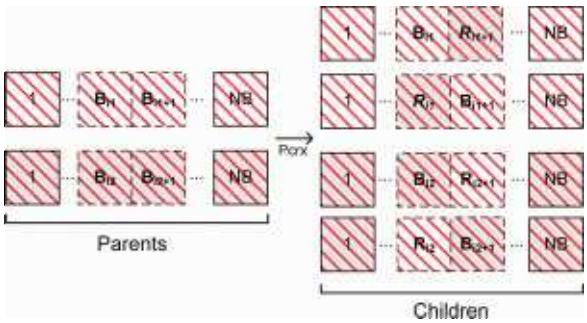

Figure 6. Crossover of two individuals operation.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

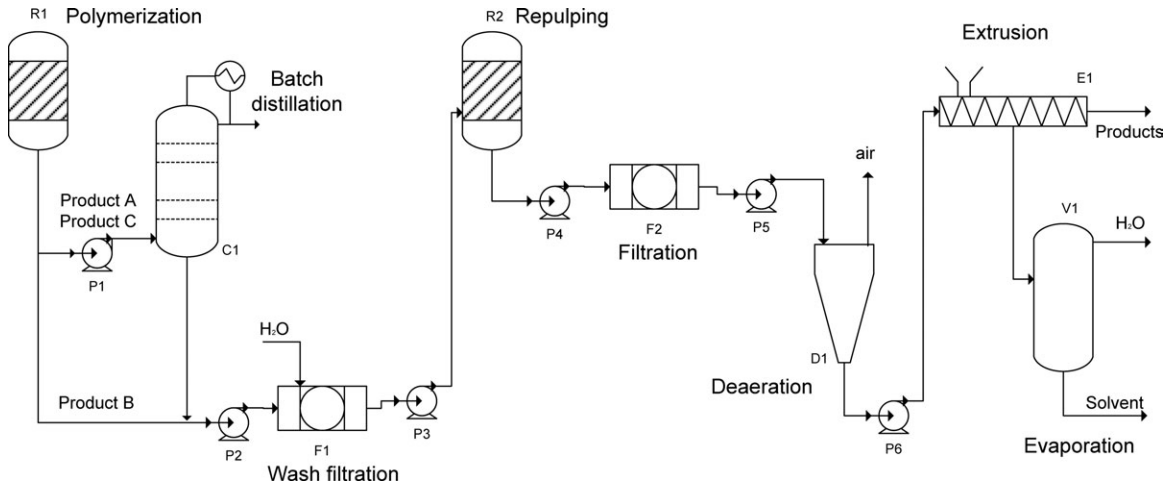

Figure 7. Flowsheet of the production process of acrylic fibers manufacturing.

Replication. To obtain the next generation, the mating pool individuals serve as parents. A certain percentage of the next generation, P_{ran}, is created following the procedure explained in the initialization procedure, disregarding any information from the mating pool, and allows for keeping the population diverse enough while reducing the risk of providing local solutions.

The rest of the population is created applying the previously discussed operators to randomly selected individuals of the mating pool. Specifically, P_{mut} is the total percentage of the individuals of the next population created by mutation, whereas the remaining is created using recombination operators: switch (P_{swi}), crossover (P_{crx}), and inversion (P_{pos}); note that $P_{rec} = P_{swi} + P_{crx} + P_{pos}$. In the case of mutation $P_{mut} = P_{mut1} + P_{mut3}$, where P_{mut1} is the percentage of the individuals of the next population created by mutation of the first chromosome part, whereas P_{mut3} represents the percentage created by mutation of the third chromosome part.

Algorithm 1: MOGA hybrid algorithm.

Data: Input parameters for the MOGA (N_{pop}, N_{gen}, N_{rep}, N_{ls}, N_{PFs}, P_{ran}, P_{mut1}, P_{mut3}, P_{swi}, P_{pos}, P_{crx}, and T_{lim})
Result: A reliable Pareto frontier estimate PF^*
begin
  solve the batching problem;
  generate the initial population POP_0 with N_{pop} individuals;
  evaluate the objective function of POP_0, calling the feasibility test math program;
  $j \leftarrow 1$;
  while ET $\leq T_{lim}$ or $j \leq N_{gen}$ **do**
    obtain the Pareto frontier estimate, $PF_j^l \xleftarrow[\text{Pareto filtering}]{} [POP_{j-1}]$;
    check the end criteria for N_{rep} consecutive PF;
    gather the mating pool $MatPool_j \leftarrow [PF_j^l \mid \text{and selected } N_{PFs}]$, from POP_{j-1};
    if $\text{mod}(\frac{j}{N_{ls}}) = 0$ **then**
      obtain PF_j^{1ls} and the N_{PFs}^{ls} from the $MatPool_j$ using bit-change local search;
      gather $MatPool_j \xleftarrow[\text{Pareto filtering}]{} [PF_j^l \mid PF_j^{1ls} \mid \text{selected } N_{PFs} \text{ and } N_{PFs}^{ls}]$;
    generate the off-spring population POP_j using as parents $MatPool_j$;
    evaluate the objective function of POP_j, calling the feasibility test math program;
    $j \leftarrow j + 1$;
  $PF^* \leftarrow PF_j^l$
end

End criteria. Finally, three possible end criteria are proposed for the algorithm, namely:

- *Maximum time*. A maximum computing time limit T_{lim} is given. In case the elapsed time (ET) exceeds such limit, the algorithm is stopped.
- *Maximum number of generations*. In case that a maximum number of generations, N_{gen}, is exceeded the algorithm is stopped.
- *Number of consecutive similar PF estimations*. When the estimation of the PF is the same for a N_{rep} number of consecutive generations, the algorithm finishes.

Results

To test the former algorithm, many of its parameters must be selected. The discussion about this selection as well as its application to large-size problems is done in this section.

The proposed methodology is illustrated in a case study that was originally posed by Grau et al.[36] It consists of a multiproduct batch process plant that produces three acrylic fiber formulations by a suspension polymerization process (Figure 7) requiring 14 processing stages. Due to minimization of inventory costs, the possible storage of polymer (considered as intermediate product) after deaeration has been disregarded, and polymer extrusion is performed right after polymer deareation. Production recipes contain a detailed description of the product batch sizes,[36] as well as operational times and energy demands[36] of each of the production stages. Production costs, sales price, environmental impact as well as the costs, times and environmental impact are detailed in a previous work[5] (Appendix). Thus, the characterization of the batch fibers and cleaning methods is detailed in the Supporting Information.

The MOGA algorithmic strategy (Algorithm 1) and Pareto filtering of the solutions have been implemented and solved in Matlab,[37,38] and the whole solving process automated using Matgams.[39] The mathematical formulation and local search have been implemented in GAMS, and solved using CPLEX 11.2 in a 4 thread 3 GHz computer.

Algorithm tuning

As discussed by Conn et al.,[27] tuning the parameters of a DFO algorithm can itself be thought as an optimization problem, where different criteria are to be met and the algorithm parameters are optimization variables. To analyze the results

Table 1. Parameters of the Hybrid MOGA

Parameter	Definition
N_{pop}	Number of individuals of the population
N_{gen}	Maximum number of generations
N_{rep}	Number of equal PF to meet end criterion
T_{lim}	Maximum time for the algorithm implementation
N_{ls}	Number of generations between two consecutive local search procedures
T_{lim}^{ls}	Maximum time available for local search optimisation
$N_X^{changes}$	Number of changes of the binary variables $X_{i'c}$ the local search procedure
$N_W^{changes}$	Number of changes of the binary variables W_i the local search procedure
N_{PFs}	Number of PFs that are included in the mating pool
P_{ran}	Percentage of random individuals of the population
P_{mut1}	Percentage of the population obtained by mutation in the first vector in the chromosome
P_{mut3}	Percentage of the population obtained by mutation in the third vector in the chromosome
P_{swi}	Percentage of the population obtained by switching positions of the second vector in the chromosome
P_{pos}	Percentage of the population obtained by inverting the chain of genes between two positions of the second vector in the chromosome
P_{crx}	Percentage of the population obtained by crossover of two parents in the second vector of the chromosome

of a given set of parameters different criteria were analyzed: (1) number of model runs (N_{mruns}), (2) the fraction of Pareto solutions that the last generation contains compared to a "true" PF ($F_{PF} = \frac{N_{PF}^{solutions}}{N_{PF}^{true}}$), and (3) the time elapsed for its execution (ET). In the first case, the number of generations and the number of model runs in both modes (feasibility/OF evaluation and local search) are considered. For calculating the fraction of PF solutions, an estimate of the "true" PF is required (PF^{true}); this estimation is done by considering all the numerical experiments that were run, or optimizations that were done using directly the MILP mathematical solver.

Instead of using an optimization approach to tune the algorithm parameters, we focus on different selections of those parameters based on a design of experiments (DOE) and check the optimization results of such selections. This approach is similar to the one adopted in Ref. 15, for tuning ACO algorithm parameters.

The parameters that must be decided and fixed beforehand are specified in Table 1. However many of them were predefined using widely accepted heuristics, thus minimizing the amount of parameters to consider. According to the GA toolbox from Matlab,[37] the number of individuals of a population, N_{pop}, must be equal or larger than 15 times the number of variables, consequently we have set this value to $N_{pop} = 15 * N_{vars}$. Given that the variables considered are the number of parameters in the string that can be changed, namely sequence, cleaning type, and batch allocation, then the N_{vars} value depends on the problem size. Specifically, for the current model, the number of variables involved in an individual is the sum of three variables (Eq. 7): (1) the noncompulsory batches (according to the product demand), (2) the sequence in which the batches must be processed (number of batches), and (3) the cleaning method (as there is cleaning method between processing and still state, the number of variables is equal to NB + 1).

$$N\text{vars} = \text{NB}_{\text{notcompulsory}} + \text{NB} + (\text{NB} + 1) \tag{7}$$

The maximum number of generations, N_{gen}, is the maximum limit that the algorithm performs; consequently, the number of generations will be related to T_{lim} and N_{rep}, according to the end criteria presented in the previous paragraphs. These three parameter values were fixed to: $N_{gen} = 1000$, $T_{lim} = 3600$ s, and $N_{rep} = 10$, which provide with adequate solution times in this case.

Regarding the mating pool, three possibilities were analyzed $N_{PFs} = 1$, where the mating pool only considers PF^1, whereas the other possibilities considered $N_{PFs} = 2$ or 3, where the mating pool consisted of the PF^1 and the second and third best Pareto fronts (PF^2 and PF^3).

To choose the percentage of the new individuals that are derived from the mating pool using each operator (P_{ran}, P_{mut1}, P_{mut3}, P_{swi}, P_{pos}, P_{crx}), different numerical experiments were performed. It has been found that when a single operator is used, that is, $P_{any} = 1$, the results are worse than when combinations of them are used and so a given combination has to be set. Given that the total fraction must add to one, only five of the previous parameters are independent. To select the best combination of such percentages, a statistical analysis has been performed based on five parameters that allow for calculating the former six, namely, they have been modeled as: $P_{ran} = r$ $P_{mut1} = (1 - r) * \beta * m$ $P_{mut3} = (1 - r) * \beta * (1 - m)$ $P_{swi} = (1 - r) * (1 - \beta) * c_1$ $P_{pos} = (1 - r) * (1 - \beta) * (1 - c_1) * c_2$ $P_{crx} = (1 - r) * (1 - \beta) * (1 - c_1) * (1 - c_2)$, where the experiment parameters are: r, β, m, c_1, and c_2.

In addition, it is necessary to define the parameters related to the local search, namely, $N_X^{changes}$, $N_W^{changes}$, N_{ls}, and T_{lim}^{ls}.

To select the best values of each parameter, three different levels for each parameter are considered (Table 2). Each experiment was run using the same three different random generator seeds to avoid eventual "lucky draws". As running the total number of experiments (^{10}) would be computationally very expensive, it has been decided to divide the tuning of the parameters in an iterative loop considering two steps, namely the tuning of the parameters related to the genetic algorithm (r, β, m, c_1, c_2, N_{PFs}) and the tuning of the parameters related to the local search (N_{ls}, $N_W^{changes}$, $N_X^{changes}$, T_{lim}^{ls}). This loop is repeated until no improvement is found.

A case study consisting of the demand shown in Table 3, where a minimum of 60% of the demand must be satisfied, is used for tuning the algorithm parameters. The optimal

Table 2. Values for the Combinations of Parameters That Result in Alternative Parameters Tuning

	Combination		
r	0.15	0.25	0.40
β	0.7	0.8	0.9
m	0.25	0.50	0.75
c_1	0.25	0.50	0.75
c_2	0.25	0.50	0.75
N_{PFs}	1	2	3
N_{ls}	1	3	5
$N_W^{changes}$	0	1	2
$N_X^{changes}$	4	6	20
T_{lim}^{ls}	5	10	20

Table 3. Product Batch Sizes and Demand for Algorithm Tuning

Fiber	Batch Size (ton/batch)	Demand (ton)
A	2.5	15
B	1.8	5.4
C	1.5	6

Table 4. Mean Values for Evaluation Metrics of the Best Resulting Solutions (Out of the 10^3 Combinations Explored) Along the Different Iterations

Iteration	r	beta	m	c1	c2	$N_{PF}^{solutions}$	N_{ls}	$N_X^{changes}$	$N_W^{changes}$	T_{ls}^{lim}	$\overline{F_{PF}}$	$\overline{\text{Generation}}$	$\overline{\text{ET}}$ (sCPU)	$\overline{F_{PF}/\text{ET}} \times 10^4$
1	0.15	0.8	0.75	0.75	0.25	2	–	–	–	–	0.111	402.3	137.7	9.489
	0.25	0.9	0.25	0.5	0.75	3	–	–	–	–	0.102	426.3	150.5	6.1959
	0.25	**0.7**	**0.75**	**0.75**	**0.25**	**3**	–	–	–	–	**0.102**	**138.3**	170.0	**6.072**
2	0.25	0.7	0.75	0.75	0.25	3	**5**	**4**	**0**	**10**	**0.991**	**36.67**	**1145.0**	**8.787**
	0.25	0.7	0.75	0.75	0.25	3	5	4	0	20	0.991	41.7	1267.0	7.945
	0.25	0.7	0.75	0.75	0.25	3	1	4	0	10	0.981	19.0	3254.0	3.067
	0.25	0.7	0.75	0.75	0.25	3	3	4	0	20	0.963	24.7	1233.0	7.973
3	0.25	0.7	0.75	0.75	0.75	3	5	4	0	10	0.981	37.7	1127.0	8.949
	0.25	**0.7**	**0.75**	**0.25**	**0.75**	**3**	5	4	0	10	**0.981**	27.7	789.7	**12.849**
	0.15	0.9	0.75	0.25	0.75	3	5	4	0	10	0.972	30.0	795.3	13.047
4	**0.25**	**0.7**	**0.75**	**0.25**	**0.75**	**3**	**5**	**4**	**0**	**10**	**0.991**	**30.3**	**959.6**	**10.443**
	0.25	0.7	0.75	0.25	0.75	3	3	4	0	5	0.991	28.0	1519.0	6.583

Pareto frontier is found as a result of all the solutions obtained from all the experiments and contains 36 solutions. The experimental results were compared considering the different values of F_{PF}, N_{mruns}, ET, and $F_{PF}/ET \times 10^4$. In this sense, the average value across three random seeds was considered for each combination of parameters. The first iteration contemplates all the combinations derived from considering the six parameters (3^6) disregarding local search. The second iteration selects the best combination of the previous six parameters, fixes them, and considers all combinations the local search parameters. Next, the values of the local search parameters are fixed, and the following iteration corresponds to the first step of the iterative loop. The results of the best combinations at the different iterations of the repeating loop are shown in Table 4. The values of parameters selected at each iteration are presented in bold. The selection criterion is based on the comparison metrics. In that table, it can be observed that the time elapsed when the solver is running an optimization run (iteration > 1) is sensitively higher than when the solver does not perform local search at all (iteration 1). However, the results obtained when applying local search are much better.

The selected combination of parameters consists of the following values: $r = 0.25$, $\beta = 0.7$, $m = 0.75$, $c_1 = 0.25$, and $c_2 = 0.75$, which corresponds to the following probabilities/percentages: $P_{ran} = 0.25$, $P_{mut1} = 0.525$, $P_{mut3} = 0.175$, $P_{swi} = 0.075$, $P_{pos} = 0.16875$, and $P_{crx} = 0.05625$. Additionally, N_{ls} was set to 5 (one local search runs and four consecutive feasible runs), while $T_{lim}^{ls} = 10$ sCPU. These last two values were selected based on a $T_{lim} = 3600$ s; longer algorithm runs might enable different values for both parameters. As for N_{PFs}, the selected value is 3. In the first iteration, the selected combination has slightly worse mean values of the selection criteria than for the two first parameter combinations shown in the table. However, for the selected combination, in none of the three seeds, the maximum number of generations was reached, whereas for the other two slightly better combinations of parameters reached the maximum number of generations for some of the seeds without meeting any other end criterion. For this reason, the third parameter combination was selected instead of the two former combinations. Thus, the values of parameters $N_X^{changes}$ and $N_W^{changes}$ are 4 and 0, respectively, which do not change from the second iteration of the algorithm, although other combinations have been explored at each algorithm iteration (refer to Table 5). Regarding the zero value for $N_W^{changes}$, it may be explained by the fact that the individual variability regarding the number of batches originated from the genetic algorithm is enough to describe the whole solution space, and it is not necessary to include the changing of the number of batches to substantially improve a given solution.

Table 5. Values of the Tuned Parameters of the GA

Parameter	Value
N_{gen}	10^3
N_{rep}	10
N_{ls}^{lim}	10
N_{ls}	5
N_{PF}	3
$N_X^{changes}$	4
$N_W^{changes}$	0
N_{ran}	0.15
N_{mut}	0.1
N_{mut2}	0.1
N_{swi}	0.1
N_{pos}	0.1
N_{crx}	0.1

Algorithm application

The case study was solved in Capón-García et al.[5] using a rigorous mathematical approach, and considering a demand of two batches of each product, and that a minimum of 50% of the demand (i.e., 1 batch) of each product must be satisfied. However, the computational times reported for the application of the rigorous algorithm was highly dependent on the solving time required for the optimization of each constrained problem. MILPs were easier to solve, whereas MINLPs needed longer times. Given that the bottleneck of that algorithm resides in the optimization step, any method or technique for decreasing this time will improve the overall algorithm solution time. Clearly the applicability of the rigorous model and algorithm, as they were presented, to practical day-to-day operation decisions was far from being optimal due to the excessive computational time required.

Therefore, this former case study has been solved using the hybrid optimization approach. Specifically, two biobjective

Table 6. Optimization Results Considering the Rigorous Approach and Hybrid Approach for Cases (i), Total Profit vs. Environmental Impact, and (ii), Profitability vs. Environmental Impact

	Rigorous Approach				Hybrid Approach — Average values		
	$N_{PF^{true}}$	Problems Solved	$N_{PF}^{solutions}$	ET (sCPU)	N_{mruns}	$N_{PF}^{solutions}$	ET (sCPU)
Case (i)	24	2000	24	1753	22.6	24	122.3
Case (ii)	51	51	28	274640	43.0	36	2985.4

Table 7. Product Batch Sizes and Example of Industrial Demand

Fiber	Batch Size (ton/batch)	Demand (ton)
A	2.5	30.0
B	1.8	9.0
C	1.5	10.5

optimization problems containing linear and nonlinear objective functions, namely (i) profit and environmental impact and (ii) profitability and environmental impact, have been considered. The parameters tuned in the previous section were used, and only the number of individuals at each generation has been adjusted according to the number of problem variables (16 variables, according to Eq. 7), resulting in a total of 240 individuals. Thus, the solution procedure has been repeated for three different seeds to provide average values for the number of Pareto solutions and the solution time.

The comparison in terms of solution quality and time between the rigorous solution method[5] and the average values for the hybrid approach are shown in Table 6. An improved estimation of the actual PF has been obtained after solving case (ii) for the three seeds with the hybrid approach. The solutions of each of the resulting PFs have been merged, and the Pareto filter applied, resulting in 51 solutions of an improved PF. Only a subset of these solutions was obtained in the previous work using the rigorous approach; and the individual runs of the hybrid approach only contain a subset of the actual PF. From the results in Table 6, it is clear that the hybrid optimization approach reaches the same optimal PF than the rigorous optimization method for case (i) in all three seeds and reaches such solution with much less time (about 90% less). Even more, for case (ii), where nonlinear problems have to be solved, the hybrid optimization approach reaches at each individual run a higher number of Pareto solutions contained in the optimal PF than the rigorous approach, though two of the solution seeds stopped at the maximum solution time (3600 s CPU).

Furthermore, the scheduling of this same plant when facing an industrial size demand has been optimized (Table 7). A minimum of 80% of the demand of each product must be satisfied. On the one hand, a biobjective problem is posed, considering profit maximization and environmental impact minimization under two different time horizons, namely (iii) unrestricted time horizon and (iv) restricted time horizon. On the other hand, case (v) takes into account the biobjective problem of profitability maximization and environmental impact minimization with unrestricted time horizon.

The parameters of the MOGA for these three cases are those tuned in the previous phase (Table 5). Nevertheless, the total number of variables is 53 according to Eq. 7, and the population has been adjusted to $N_{pop} = 15 \times 53 = 795$ individuals as indicated in the algorithm tuning. Thus, the maximum computational time of the algorithm has been extended to 12 h CPU, because the problem is more complex. Even more, in case iv, as there is a strict time horizon limit, many unfeasible individuals appear (ca. 80%). Hence, in this case, the population has been increased to three times the suggested value.

Case iii. The trade-offs that arise considering profit and total environmental impact for a large demand are studied. Figure 8 presents the PF for this two-objective optimization. Solutions representation contain the ordered sequence of batches, where the capital letters A–C, stand for the single product campaigns separated by the number of the interproduct cleaning method (namely, 1–3), superscripts indicate the number of batches in the campaign and subscripts show the total number of cleanings of each type within the product campaign ordered in terms of cleaning method type: 1|2|3. The estimated PF in this case results in 36 nondominated solutions (circles). Computational effort and number of

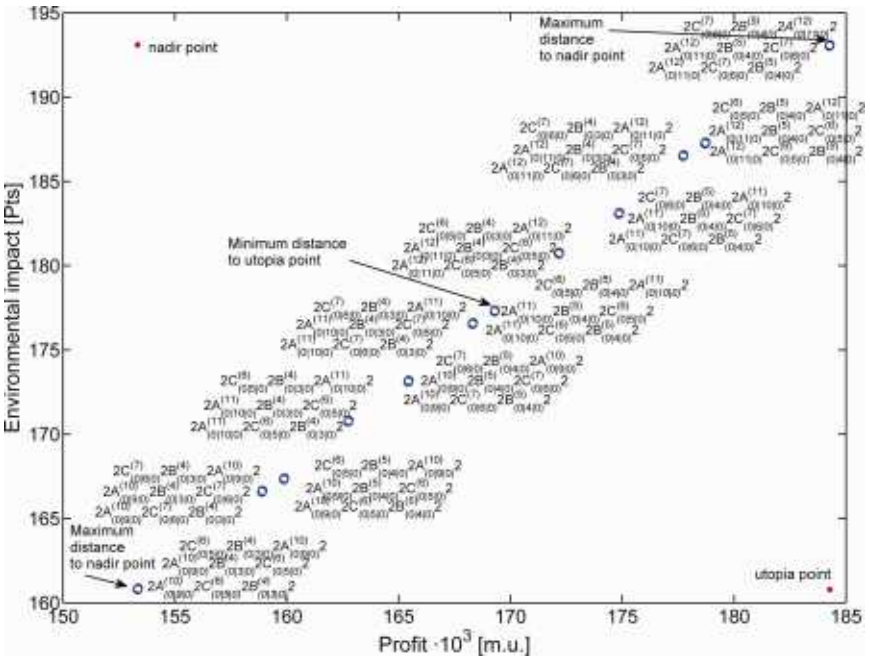

Figure 8. Case (iii). Solutions for two-objective optimization considering profit and environmental impact.

Nondominated solutions are encircled (Pareto frontier); stars represent nadir, utopian points; and sequences in italics represent compromise solutions shown in Table 9. Note that for the sake of clarity some sequences are not explicitly shown in the figure. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table 8. Case (iii)—Number of Pareto Solutions, Generations, Elapsed Time, and Fraction of Actual Pareto Frontier (36 Solutions) for a Large Demand Without Restricted Time, for Three Random Seeds

	Seed 1	Seed 2	Seed 3
$N_{PF}^{solutions}$	34	36	30
N_{mruns}	35	35	28
ET $\times$ 10^3 (sCPU)	8.61	8.40	4.65
F_{PF} (%)	97.2	100.0	83.3

generations for the three seeds are presented in Table 8. The PF estimations for all three seeds are quite close to the actual PF. Even more, the solution time is below 2.5 h CPU.

The solution with the highest profit satisfies the total demand, whereas the most environmentally friendly option only processes the minimum required amount of each product (a minimum of 80% of the demand of each product). In any case, the same changeover cleaning method 2 is selected in all solutions, because it is the most economic and environmental advantageous, despite the time required, which is not an active constraint in this problem. Pareto solutions are distributed in 12 clusters between the two anchor points. The clusters basically differ in the number of batches of each product.

Regarding the most environmentally friendly solution cluster located at about 153.3 $\times$ 10^3 m.u. total profit and 160.8 Pts environmental impact, product C offers higher increment in profit and lower environmental impact, resulting in the next cluster located around 158.9 $\times$ 10^3 m.u. total profit and 166.6 Pts environmental impact. The following sequence of minor environmental benefit with higher gain in profit (around 159.9 $\times$ 10^3 m.u. and 167.3 Pts) includes an additional batch of product B instead of C; and then, a batch of A instead B or C (around 162.7 $\times$ 10^3 m.u. and 170.8 Pts). Next, two additional batches are considered in the production sequence resulting in one cluster located at about 165–172 $\times$ 10^3 m.u and 173–181 Pts, and the other one at about 175–179 $\times$ 10^3 m.u and 183–187 Pts. Finally, compliance with total demand reaches the greatest economic benefit, which is 184.3 $\times$ 10^3 m.u. and produces an environmental impact of 193.1 Pts. In every cluster, solutions differ in the production sequences. To start producing with fiber C is slightly more environmentally friendly and less economically profitable than with fiber A. In all cases, batches are produced in single product campaigns, as inventory costs have not been considered.

Table 9 shows that the compromise solution according to the minimum distance to the utopian point consists of sequence $2C_{(0|5|0)}^{(6)}2B_{(0|4|0)}^{(5)}2A_{(0|10|0)}^{(11)}2$, which is located approximately in the middle of the whole range of both objective functions. If the maximum distance to the nadir point was selected as decision criterion, there would be two

Table 9. Case (iii)—Utopian, Nadir, and Solutions of Compromise According to the Different Metrics Considering Profit and Environmental Impact

z^{profit} $\times$ 10^3 (m.u.)	z^{ei} (Pts)	Sequence	Distance Utopian	Distance Nadir
184.2802*	193.1041†	$2A_{(0\|12\|0)}^{(13)}2C_{(0\|6\|0)}^{(7)}2B_{(0\|4\|0)}^{(5)}2$	1.000	**1.000**
169.2880	177.2768	$2C_{(0\|5\|0)}^{(6)}2B_{(0\|4\|0)}^{(5)}2A_{(0\|10\|0)}^{(11)}2$	**0.704**	0.711
153.3351†	160.7842*	$2C_{(0\|5\|0)}^{(6)}2B_{(0\|3\|0)}^{(4)}2A_{(0\|9\|0)}^{(10)}2$	1.000	**1.000**

In bold, value of the minimum distance to utopia and maximum distance to nadir. Distances are reported normalized.
*Utopia
†Nadir.

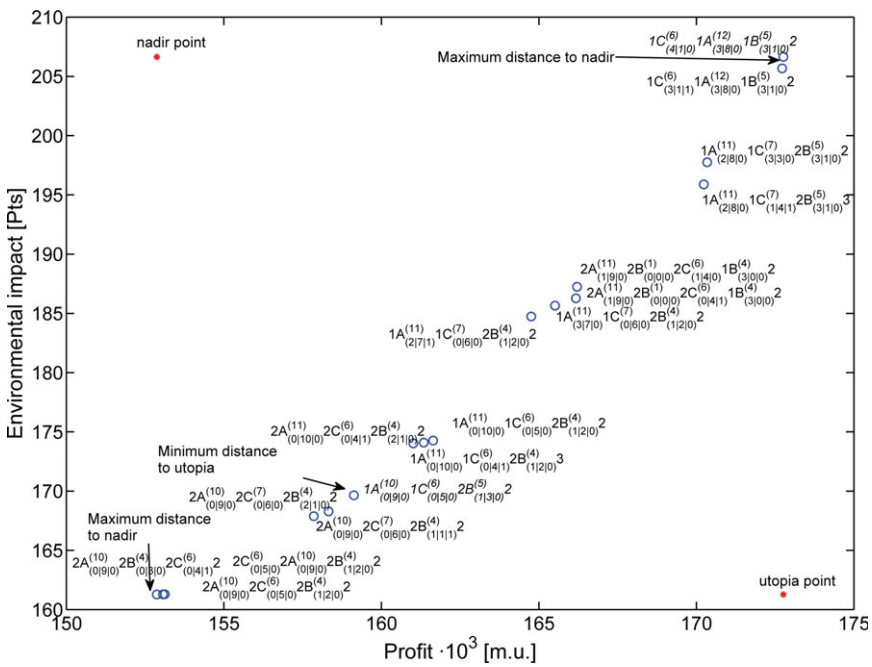

Figure 9. Case (iv). Solutions for two-objective optimization considering profit and environmental impact.

Notation as Figure 8. Sequences in italics represent compromise solutions shown in Table 11. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table 10. Case (iv) — Number of Pareto Solutions, Generations, Elapsed Time and Fraction of Actual Pareto Frontier (17 solutions) for a Large Demand with Restricted Time, for Three Random Seeds

	Seed 1	Seed 2	Seed 3
$N_{PF}^{solutions}$	11	9	16
N_{mruns}	25	25	29
ET $\times$ 10^4 (sCPU)	3.73	3.67	4.43
F_{PF} (%)	52.6	47.1	41.2

possibilities: either the solution of maximum profit or the solution of minimum environmental impact, because both of them have the same maximum normalized distance to the nadir solution.

Case iv. Figure 9 presents the PF for the two-objective optimization of profit and total environmental impact considering a restricted time horizon (140 h). The estimated PF in this case results in 17 nondominated solutions (circles), which has been estimated from the combination of three different seeds. Table 10 shows the summarized results for the three cases. In each seed, the PF estimation contains about one half of the actual PF solutions estimation. The maximum computing time allowed is reached without meeting the end criteria of finding 10 consecutive identical solutions. Anyhow, the solution quality regarding the computational effort is very reasonable to schedule the production plan for a period of time of a full week with strict time horizon constraints. Therefore, the evolutionary approach proves to be an efficient method for finding good Pareto estimations with reasonable computational effort, specially for highly timely constrained problems.

The PF contains at least one interbatch cleaning method 1 or 3 in all sequences (note that such method may be applied within the product campaign, written as a subscript in the solution notation) to fulfill the time horizon restriction, because these methods are less time consuming than method 2. Thus, cleaning method 2 is combined with cleaning method 1 to obtain Pareto optimal production sequences from both economic and environmental points of view. Again, single product campaigns are generated due to the problem features (changeover costs and impacts, and negligible storage costs) and the selected objective functions. Thus, the solution with the highest profit in this case has lower profit and higher environmental impact than the corresponding highest profit in case iii (note that the axes in Figures 8 and 9 have different scales). The main reasons consist of two facts: (1) one batch of product C cannot be produced within the scheduling time horizon; and (2) cleaning method 1, which has higher environmental impact than method 2, has to be used to reduce the total production time within the time horizon. It is also noteworthy to mention that the solution with the lowest environmental impact in case (iii) cannot be reached in case (iv). Therefore, slightly different schedules considering

Table 11. Case (iv) — Utopian, nadir, and solutions of compromise according to the different metrics considering profit and environmental impact with restricted time horizon

$z^{profit} \times 10^3$ (m.u.)	z^{ei} (Pts)	Sequence	Distance Utopian	Distance Nadir
152.872†	161.278*	$2A^{(10)}_{(0\|9\|0)}2B^{(4)}_{(0\|3\|0)}2C^{(6)}_{(0\|4\|1)}2$	1.000	**1.000**
161.643	174.266	$1A^{(10)}_{(0\|9\|0)}1C^{(6)}_{(0\|5\|0)}2B^{(5)}_{(1\|3\|0)}2$	**0.628**	0.839
172.761*	206.643†	$1C^{(6)}_{(4\|1\|0)}1A^{(12)}_{(3\|8\|0)}1B^{(5)}_{(3\|1\|0)}2$	1.000	**1.000**

In bold, value of the minimum distance to utopia and maximum distance to nadir. Distances are reported normalized.
*Utopia
†Nadir.

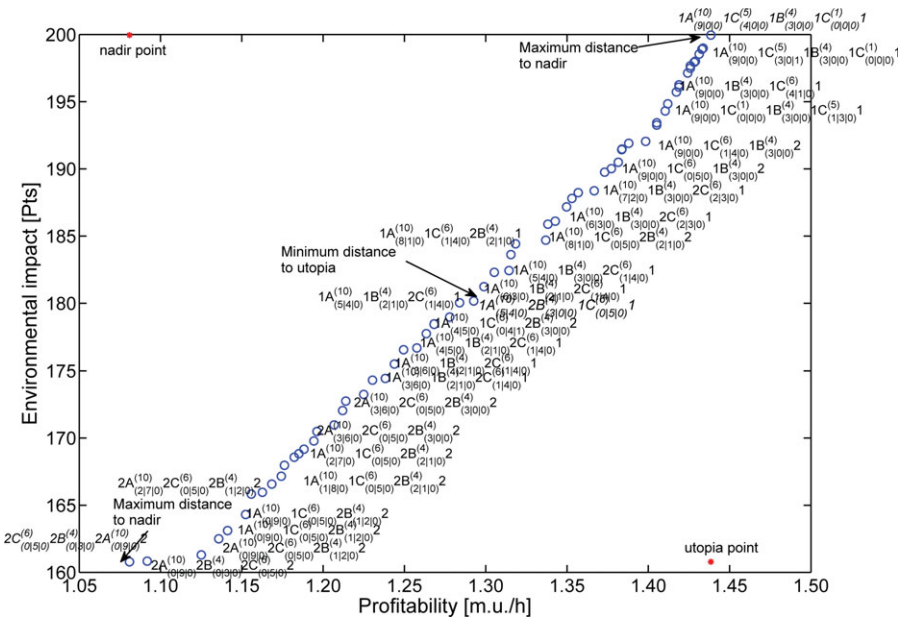

Figure 10. Case (v). Solutions for two-objective optimization considering profitability and environmental impact.

Notation as Figure 8. Sequences in italics represent compromise solutions shown in Table 13. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table 12. Case (v) — number of Pareto Solutions, Generations, Elapsed Time, and Fraction of Actual Pareto Frontier (67 Solutions) for a Large Demand Considering Profitability and environmental Impact, for Three Random Seeds

	Seed 1	Seed 2	Seed 3
$N_{PF}^{solutions}$	64	70	58
N_{mruns}	45	40	45
ET $\times$ 10^4 (sCPU)	4.62	4.57	4.69
F_{PF} (%)	46.3	47.8	59.7

cleaning methods 1 and 3, which are less time consuming, are introduced in those solutions with the lowest environmental impact in case iv.

Table 11 shows that the compromise solution according to the minimum distance to the utopian point consists of sequence $1A^{(10)}_{(0|9|0)}1C^{(6)}_{(0|5|0)}2B^{(5)}_{(1|3|0)}2$, which is located approximately in the middle of the whole range of both objective functions. If the maximum distance to the nadir point was selected as decision criterion, there would be two possibilities: either the solution of maximum profit or the solution of minimum environmental impact.

Case v. The estimated PF for the two-objective optimization of profitability and total environmental impact considering unrestricted time horizon contains 67 nondominated solutions, estimated from the results of three different seeds. Figure 10 shows the solutions, which are not grouped in clusters as in the previous cases, but present a more homogeneous distribution along both objective functions. Such behavior is similar to that in case (ii), which consists of the same objective functions, but a much smaller demand .

Table 12 summarizes the results of the three algorithm runs. The PF estimation of each run contains about one half of the number of integer solutions of the final PF estimation, which results in a good and reasonable trade-off between solution quality and computational effort to schedule the production plan for a period of time of a full week in a case of industrial size. Therefore, the proposed evolutionary approach is an efficient method to find good Pareto estimations, even for highly constrained and nonlinear problems given that the proposed strategy of local search and the Dinkelbach algorithm for the MILFP problem are incorporated in the algorithm.

The estimated PF solutions contain cleaning methods 1 and 2, which are the most profitable and least contaminant respectively. Thus, the former two cleaning methods are used to obtain Pareto optimal production sequences, from an economic and environmental point of view. It is noteworthy to mention that some PF solutions have multiple campaign for product production. Specifically, product C is divided into two campaigns in the most profitable solution, because the resulting makespan decreases despite of a slight decrease in total profit; as a result, profitability is higher with multiple campaign for product.

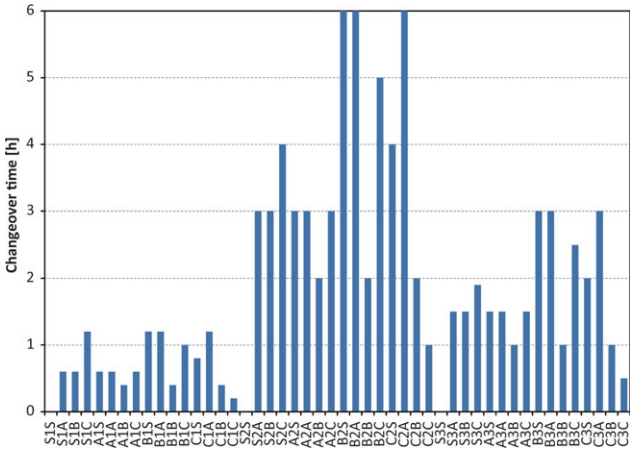

Figure 11. Changeover time between pairs of products (S-still state, A–C) for the three methods (1–3).

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

The compromise solution according to the minimum distance to the utopian point consists of sequence $1A^{(10)}_{(5|4|0)}2B^{(4)}_{(3|0|0)}1C^{(6)}_{(0|5|0)}1$, as shown in Table 13. Such solution is located approximately in the middle of the whole range of both objective functions. If the maximum distance to the nadir point was selected as decision criterion, there would be two possibilities: either the solution of maximum profitability or the solution of minimum environmental impact. This case demonstrates the capability of the proposed approach to also deal with nonlinear objective functions; what is more, it can tackle any scheduling problem with the proposed features regardless of the selected objective functions.

Conclusions

For short-term scheduling of large plants, it is necessary to use optimization strategies that can deal with the problem complexity in adequate computational effort. Such requirement is even stricter when multiobjective optimization is carried out. In a previous work,[5] the results of rigorous optimization for multiobjective scheduling problems was studied, and existing trade-offs were unveiled. However, reported computational times were extremely high. Hence, a hybrid strategy consisting of an evolutionary method, based on the genetic algorithm, combined with a rigorous local search has been developed and effectively applied to obtain reliable Pareto frontiers in small computational times. In the proposed strategy, an iteration loop is performed to adequately tune the GA parameters. As a result, the stochastic strategy allows to obtain better Pareto frontier estimations than the deterministic method even for small cases, where the rigorous mathematical problem entails high computational resource usage.

Table 13. Case (v) — Utopian, Nadir, and Solutions of Compromise According to the Different Metrics Considering Profitability and Environmental Impact with Unrestricted Time Horizon

z^{prod} (m.u./h)	z^{ei} (Pts)	Sequence	Distance Utopian	Distance Nadir
1.0810†	160.784*	$2C^{(6)}_{(0\|5\|0)}2B^{(4)}_{(0\|3\|0)}2A^{(10)}_{(0\|9\|0)}2$	1.000	**1.000**
1.2926	180.164	$1A^{(10)}_{(5\|4\|0)}2B^{(4)}_{(3\|0\|0)}1C^{(6)}_{(0\|5\|0)}1$	**0.641**	0.778
1.4385*	199.955†	$1A^{(10)}_{(9\|0\|0)}1C^{(5)}_{(4\|0\|0)}1B^{(4)}_{(3\|0\|0)}1C^{(1)}_{(0\|0\|0)}1$	1.000	**1.000**

In bold, value of the minimum distance to utopia and maximum distance to nadir. Distances are reported normalized.
*Utopia
†Nadir.

Even more, for large-size problems or highly constrained problems, and either linear or MILFP problems, the proposed hybrid strategy allows to find high-quality Pareto frontier estimations in a reduced amount of computational effort. Thus, the MOGA is specially useful for cases with time horizon constraints, in which problem optimality is highly difficult to guarantee.

Acknowledgments

Financial support received from the Spanish Ministerio de Ciencia e Innovación and the European Community (ERDF; DPI2006-05673; DPI2009-09386) is fully appreciated. In addition, financial support received from the Agencia de Gestió d'Ajuts Universitaris i de Recerca (AGAUR) from Generalitat de Catalunya and European Social Fund as well as the FPU program from the Spanish Ministerio de Educación y Ciencia are also fully acknowledged. The authors also thank the insightful discussions regarding Dinkelbach's and local search algorithms with Professors Ignacio Grossman and Sven Leyffer during the Exploratory Workshop on MINLP in Seville (Spain), 2010.

Notation

Sets and subsets

c = cleaning modes between products
g = objective functions
i = batches
k = stages
p = products (product S simulates plant "still" state)
dynI = set of batches i that have been assigned to a product

Parameters

P_{Mut} = probability of mutation
P_{SwPos} = probability of switching between two positions
P_{SwCha} = probability of switching the order of all batches between two positions
P_{CrxOv} = probability of cross-over between two chromosomes
$\overline{\text{BP}_i}$ = price resulting from the production of batch i
$\overline{\text{BS}_i}$ = batch size of batch i
$\overline{\text{ChCost}_{ii'kc}}$ = changeover cost between batches i and i' for stage k using changeover type c
$\overline{\text{EnvIm}_i}$ = environmental impact resulting of producing a batch i
$\text{EnvIm}_{ii'kc}$ = environmental impact associated with changeover type c between batches i and i' for stage k
H = time horizon
pW_i = parameter which holds value 1, if batch i is processed; and 0, otherwise
$pX_{ii'c}$ = parameter which takes values 1 or 0, depending on the assignment of cleaning changeover method c, if batch i is produced immediately before batch i'

Continuous variables

Mk = objective function that aims at minimizing the makespan
z^{ei} = objective function that aims at minimizing the environmental impact
z^{prod} = objective function that aims at maximizing productivity
z^{profit} = objective function that aims at maximizing profit
z^{rei} = objective function that aims at minimizing the relative environmental impact

Binary variables

W_i = production of batch i
$X_{ii'c}$ = Assignment of cleaning method c to changeover, if batch i is produced immediately before batch i'

Algorithm notation

B_i = order in which batch i is processed
CL_i = cleaning method that precedes batch i
ET = elapsed time
F_{PF} = fraction between the number of individuals in the estimation of the PF and the total number of individuals belonging to the best estimation of the PF
j = iteration counter
MatPool_j = mating pool at iteration j
NB = total number of batches associated with the final products that may be produced
$\text{NB}_{\text{notcompulsory}}$ = total number of batches associated with the final products that are not compulsory according to production demand constraints
N_{gen} = maximum number of generations
N_{ls} = number of generations between two consecutive local search procedures
N_{mruns} = number of generations of the genetic algorithm
N_{PFs} = number of PFs that are included in the mating pool
$N_{\text{PF}}^{\text{solutions}}$ = number of solutions contained in the PF of the last generation of the algorithm
$N_{\text{PF}}^{\text{true}}$ = number of solutions contained in the best estimation of the actual PF of the problem
N_{pop} = number of individuals of the population
N_{rep} = number of equal PF to meet end criterion
Nvars = number of problem variables
N_W^{changes} = number of changes of the binary variables W_i the local search procedure
N_X^{changes} = number of changes of the binary variables $X_{ii'c}$ the local search procedure
P_{crx} = percentage of the population obtained by crossover of two parents in the second vector of the chromosome
PF^* = Pareto frontier solutions estimated by the proposed algorithm
PF_j^i = Pareto frontier solutions at generation j belonging to the i Pareto front
P_{mut} = percentage of individuals obtained by mutation
P_{mut1} = percentage of the population obtained by mutation in the first vector in the chromosome
P_{mut3} = percentage of the population obtained by mutation in the third vector in the chromosome
POP0 = initial population
POP_j = population at iteration j
P_{pos} = percentage of the population obtained by inverting the chain of genes between two positions of the second vector in the chromosome
P_{ran} = percentage of random individuals of the population
P_{rec} = percentage of individuals obtained by recombination
P_{swi} = percentage of the population obtained by switching positions of the second vector in the chromosome
T_{lim} = maximum time for the algorithm implementation
$T_{\text{lim}}^{\text{ls}}$ = maximum time available for local search optimization
tol = tolerance value as termination criterion

Literature Cited

1. Wiecek MM, Ehrgott M, Fadel G, Figueira JR. Multiple criteria decision making for engineering. *Omega-Int J Manage S*. 2008;36:337–339.
2. Bojarski AD. Life cycle thinking and general modelling contribution to chemical process sustainable design and operation, PhD Thesis. Barcelona, Spain: Universitat Politècnica de Catalunya, 2010.
3. Mendez CA, Cerda J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng*. 2006;30:913–946.
4. Castro PM, Harjunkoski I, Grossmann IE. Greedy algorithm for scheduling batch plants with sequence-dependent changeovers. *AIChE J*. 2011;57:373–387.
5. Capón-García E, Bojarski AD, Espuña A, Puigjaner L. Multiobjective optimization of multiproduct batch plants scheduling under environmental and economic concerns. *AIChE J*. 2011;57:2766–2782.
6. Messac A, Ismail-Yahaya A, Mattson C. The normalized normal constraint method for generating the Pareto frontier. *Struct Multidiscip Optimization*. 2003;25:86–98.
7. Silva JDL, Burke EK, Petrovic S. *An introduction to multiobjective optimisation in scheduling and timetabling*. In: Gandibleux X, Sevaux M, Sörensen K, T'Kindt V. editors *Metaheuristics for Multiobjective Optimisation*. Germany: Springer, 2004:91–131.
8. Grossmann IE, Guillén-Gosálbez G. Optimal design and planning of sustainable chemical supply chains under uncertainty. *AIChE J*. 2009;55:99–121.

9. Song J, Park H, Lee D, Park S. Scheduling of actual size refinery processes considering environmental impacts with multiobjective optimization. *Ind Eng Chem Res*. 2002;41:4794–4806.
10. Berlin J, Sonesson U, Tillman A. A life cycle based method to minimise environmental impact of dairy production through product sequencing. *J Clean Prod*. 2007;15:347–356.
11. Berlin J, Sonesson U. Minimising environmental impact by sequencing cultured dairy products: two case studies. *J Clean Prod*. 2008;16:483–498.
12. Park M, Kim D, Ko D, Moon I. Multiobjective optimisation for environment-related decision making in paper mill processes. *Int J Environ Pollut*. 2007;29:127–143.
13. Coello-Coello C, Landa-Becerra R. Evolutionary multiobjective optimization in materials science and engineering. *Mater Manuf Process*. 2009;24:119–129.
14. Jayaraman VK, Kulkarni BD, Karale S, Shelokar P. Ant colony framework for optimal design and scheduling of batch plants. *Comput Chem Eng*. 2000;24:1901–1912.
15. Arnaout JP, Rabadi G, Musa R. A two-stage Ant Colony Optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. *J Intell Manuf*. 2010;21:693–701.
16. Li WD, McMahon CA. A simulated annealing-based optimization approach for integrated process planning and scheduling. *Int J Comp Integ Manuf*. 2007;20:85–95.
17. Guo Y, Li W, Mileham A, Owen G. Applications of particle swarm optimisation in integrated process planning and scheduling. *Robot CIM-Int Manuf*. 2009;25:280–288.
18. Dayou L, Pu Y, Ji Y. Development of a multiobjective GA for advanced planning and scheduling problem. *Int J Adv Manuf Technol*. 2009;42:974–992.
19. Chyu CC, Chang WS. A Pareto evolutionary algorithm approach to bi-objective unrelated parallel machine scheduling problems. *Int J Adv Manuf Technol*. 2010;49:697–708.
20. He Y, Hui CW. Genetic algorithm for large-size multi-stage batch plant scheduling. *Chem Eng Sci*. 2007;62:1504–1523.
21. Ponnambalam S, Mohan-Reddy M. A GA-SA multiobjective hybrid search algorithm for integrating lot sizing and sequencing in flowline scheduling. *Int J Adv Manuf Technol*. 2003;21:126–137.
22. Azzaro-Pantel C, Bernal-Haro L, Baudet P, Domenech S, Pibouleau L. A two-stage methodology for short-term batch plant scheduling: discrete-event simulation and genetic algorithm. *Comput Chem Eng*. 1998;22:1461–1481.
23. Senties OB, Azzaro-Pantel C, Pibouleau L, Domenech S. A neural network and a genetic algorithm for multiobjective scheduling of semiconductor manufacturing plants. *Ind Eng Chem Res*. 2009;48:9546–9555.
24. Senties OB, Azzaro-Pantel C, Pibouleau L, Domenech S. Multiobjective scheduling for semiconductor manufacturing plants. *Comput Chem Eng*. 2010;34:555–566.
25. Aguilar-Lasserre A, Pibouleau L, Azzaro-Pantel C, Domenech S. Enhanced genetic algorithm-based fuzzy multiobjective strategy to multiproduct batch plant design. *Appl Soft Comput*. 2009;9:1321–1330.
26. Bonfill A, Espuña A, Puigjaner L. Proactive approach to address the uncertainty in short-term scheduling. *Comput Chem Eng*. 2008;32:1689–1706.
27. Conn AR, Scheinberg K, Vicente LN. *Introduction To derivative-free optimization, 1st ed*. In: *MPS-SIAM Series on Optimization*. Philadelphia: Society for Industrial and Applied Mathematics and Mathematical Programming Society, 2009.
28. Gupta S, Karimi IA. An improved MILP formulation for scheduling multiproduct, multistage batch plants. *Ind Eng Chem Res*. 2003;42:2365–2380.
29. Clift R, Azapagic A. The application of Life Cycle Assessment to process selection, design and operation. *NATO ASI 2*. 1999;62:69–84.
30. Stefanis SK, Livingston AG, Pistikopoulos EN. Environmental impact considerations in the optimal design and scheduling of batch processes. *Comput Chem Eng*. 1997;21:1073–1094.
31. Cabezas H, Bare J, Mallick S. Pollution prevention with chemical process simulators: the generalized waste reduction (WAR) algorithm—full version. *Comput Chem Eng*. 1999;23:623–634.
32. Chen H, Shonnard D. Systematic framework for environmentally conscious chemical process design: early and detailed design stages. *Ind Eng Chem Res*. 2004;43:535–552.
33. ISO14001. *Environmental Management Systems—Requirements with Guidance for use*. 2004.
34. You FQ, Castro PM, Grossmann IE. Dinkelbach's algorithm as an efficient method to solve a class of MINLP models for large-scale cyclic scheduling problems. *Comput Chem Eng*. 2009;33:1879–1889.
35. Coello CAC, Lamont GB, Veldhuizen DAV. *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. New York: Springer, 2007.
36. Grau R, Graells M, Corominas J, Espuña A, Puigjaner L. Global strategy for energy and waste analysis in scheduling and planning of multiproduct batch chemical processes. *Comput Chem Eng*. 1996;20:853–868.
37. Mathworks. The Mathworks Corporation. Matlab 7.8. February 12, 2009. Available at: http://www.mathworks.com/index.html. (Accessed May 31, 2009.)
38. Cao Y. Pareto Front: two efficient algorithms to find Pareto Front. 29 July 2008. Available at: http://www.mathworks.com/matlabcentral/fileexchange/17251 (accessed June 23, 2009).
39. Ferris M. MATLAB and GAMS: Interfacing Optimization and Visualization Software. 2005. Available at: http://www.cs.wisc.edu/math-prog/matlab.html (accessed May 31, 2009).

Appendix: Case Study Description

Table A1 and Figures A1–A4 contain the processing times and economic and environmental data related to the acrylic fiber production plant.

Table A1. Operation Times and Equipment Associated with Each Stage for All Possible Produced Products (h)

		Product A					Product B					Product C				
Stage	Equipment	P	L	O	U	TOT	P	L	O	U	TOT	P	L	O	U	TOT
1	R1	0.2	0	2	0.3	2.5	0.2	0	3	0.75	3.95	0.2	0	1	0.3	1.5
2	P1	0.2	0	0.3	0	0.5	0	0	0	0	0	0.2	0	0.3	0	0.5
3	C1	0.5	0.3	2.5	0.75	4.05	0	0	0	0	0	0.5	0.3	2	0.75	3.55
4	P2	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95
5	F1	0.5	0	0.75	0	1.25	0.5	0	0.75	0	1.25	0.5	0	0.75	0	1.25
6	P3	0.2	0	0.75	0	1.25	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95
7	R2	0.3	0.75	1	0.75	2.8	0.3	0.75	0.75	0.75	2.55	0.3	0.75	0.5	0.75	2.3
8	P4	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95
9	F2	0.5	0	0.75	0	1.25	0.5	0	0.75	0	1.25	0.5	0	0.75	0	1.25
10	P5	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95
11	D1	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95
12	P6	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95
13	E1	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95	0.2	0	0.75	0	0.95
14	V1	0.3	0.75	3.5	0	4.55	0.3	0.75	3	0	4.05	0.3	0.75	1.5	0	2.55

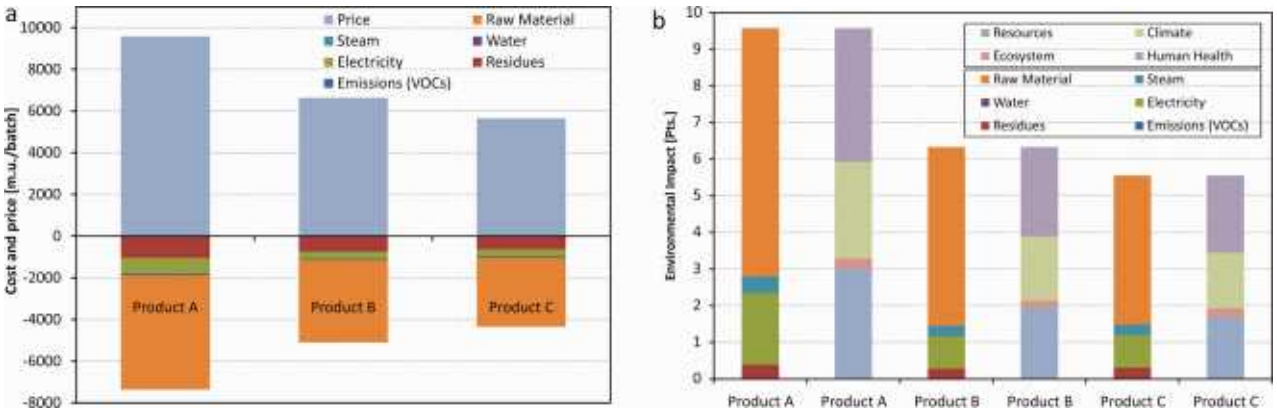

Figure A1. Batch cost and price, and environmental impact for the three acrylic fibers.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

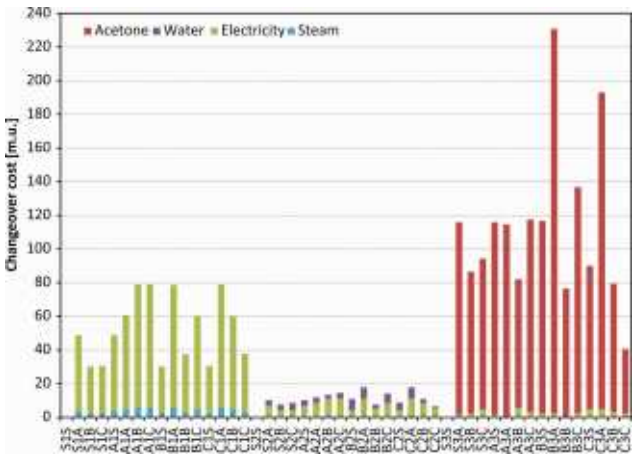

Figure A2. Changeover costs between pairs of products (S-still state, A–C) for the three methods (1–3).

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

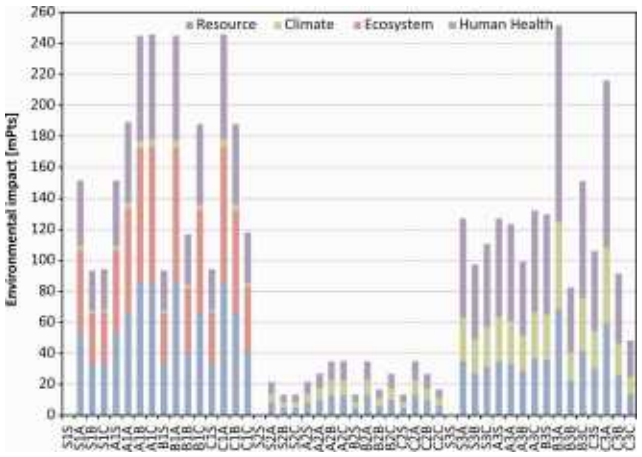

Figure A3. Changeover environmental impacts between pairs of products (S-still state, A–C) for the three methods (1–3).

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Manuscript received Jan. 2, 2012, and revision received May 1, 2012.