Scheduling Dispensing and Counting in Secondary Pharmaceutical Manufacturing

Michele Ciavotta
Instituto Tecnológico de Informática, Universidad Politécnica de Valencia, Ciudad Politécnica de la Innovación, Camino de Vera, E-46022 Valencia, Spain

Carlo Meloni
Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, Via E. Orabona, 4-70125 Bari, Italy

Marco Pranzo
Dipartimento di Ingegneria dell'Informazione, Università di Siena, Via Roma, 56-53100 Siena, Italy

DOI 10.1002/aic.11725
Published online March 30, 2009 in Wiley InterScience (www.interscience.wiley.com).

In this article, we describe a general methodology for operations scheduling in dispensing and counting departments of pharmaceutical manufacturing plants. The departments are modeled as a multiobjective parallel machines scheduling problem under a number of both standard and realistic constraints, such as release times, due dates and deadlines, particular sequence-dependent setup times, machine unavailabilities, and maximum campaign size. Main characteristics of the methodology are the modularity of the solution algorithms, the adaptability to different objectives and constraints to fulfill production requirements, the easiness of implementation, and the ability of incorporating human experience in the scheduling algorithms. Computational experience carried out on two case studies from a real pharmaceutical plant shows the effectiveness of this approach. © 2009 American Institute of Chemical Engineers *AIChE J,* 55: 1161–1170, 2009

Keywords: scheduling, operations management, pharmaceutical industry, rollout algorithm, pilot method

Introduction

The pharmaceutical production systems are called to improve their production planning and scheduling methods to strive for better utilization of resources and reduction of the response time. In fact, declining windows of product exclusivity, competition from generics, and new market entrants from third world countries are increasing the competitive pressure on pharmaceutical companies.[1,2] Pharmaceutical producers need to increase flexibility, and, at the same time, to obtain a noticeable reduction of production costs preserving the quality of products and processes.

Maintaining high growth rates requires to increase the number of new products or the internal efficiency of the organizations. In fact, there are great margins for cost reduction in facing time-to-market opportunity costs, research and development inefficiencies, under-developed processes for matching supply with demand, poor manufacturing cycle efficiency, and discontinuous workflows.[1,3]

In this article, we focus on one of such issues, namely the optimization methods for production scheduling. Common features of the production processes are the small size of the lots, strict requirements on product quality and timing delivery, and the large variety of constraints arising from the shop-floor characteristics or from various technological

Correspondence concerning this article should be addressed to M. Pranzo at pranzo@dii.unisi.it

© 2008 American Institute of Chemical Engineers

issues. Moreover, the quality of a schedule may involve several indices, such as the use of shop-floor resources, production costs in terms of personnel and energy consumption, the attainment of production targets, and so on.

Because of the inherent complexity and variety of pharmaceutical scheduling problems, a large extent of scheduling and related issues are still carried out by human schedulers, who are able to develop feasible schedules based on their past experience and intuition. Developing and implementing effective computerized systems for addressing such operational problems require, therefore, focusing on a number of aspects that are rarely taken into account jointly in the scheduling theory. Among others, we recall the need for general solution algorithms, able to deal with different objectives and constraints to fulfill production requirements, and automated scheduling systems able to easily embed observations and suggestions arising from the scheduling practice.

In pharmaceutical manufacturing, the production strongly depends on the customer demands in terms of products, quantities, and delivery times. In this context, the automation of scheduling activities in noncritical areas allows the management to focalize first on critical areas then propagate solutions and constraints to other areas of the plant. Thus it allows to enlightening promptly possible infeasibilities under on overall view of the plant, as noncritical areas often serve different departments or lines. This contributes to a plant-wide increase of scheduling responsiveness.

In this article, we move in this direction and focus on a general methodology to address scheduling problems arising in noncritical areas of pharmaceutical production systems. Namely, we apply the method for scheduling the production in a parallel machine environment, including lot production with setups, due dates, deadlines, and other realistic constraints. Moreover, we report on two case studies detailing the implementation of the proposed methodology in a real industrial context. The rationale is developing algorithms easily adaptable to deal with different constraints and objectives, and suitable for incorporating human experience in the computerized logic. Such methods are not necessarily based on strong mathematical properties of the particular problem to solve. Rather, the search process is guided by several heuristic procedures, called pilot heuristics, so that it is possible to include new procedures suggested by human experience and/or to modify the behavior of the system by adding, removing or modifying the pilot heuristics. At this aim we apply a metaheuristic strategy known as *rollout* method[4] or *pilot* method.[5,6] These methodologies for the approximate solution of discrete optimization problems have been independently developed by Duin and Voβ[5,6] and Bertsekas et al.[4] The main idea of a rollout algorithm is to include one or more pilot heuristics in a larger framework with the purpose of improving their performance. We also consider general local search techniques for improving the quality of the schedules, and we focus in particular on variable neighborhood descent techniques.[7,8]

The article is organized as follows. The next section introduces the main issues of pharmaceutical manufacturing systems. In the Section ''Problem description'' the case studies are illustrated in detail, and in the Section ''Solution methods'' the proposed algorithmic approaches are described, while the related results are reported and discussed in the successive Section ''Computational results''. Finally, conclusions are drawn in the last section of the article.

Pharmaceutical Manufacturing Systems

The pharmaceutical industrial production mainly consists of at least two manufacturing stages: *primary* and *secondary* manufacturing.[3,9] The former is dedicated to the production of active ingredients and other basic components and production is typically a *push* process (driven by forecast demand) organized in long campaigns to reduce the impact of long cleaning and setup times that are necessary to ensure quality and to avoid cross-contamination. Primary manufacturing is therefore not very sensitive to short-term fluctuations of the customers demand, and the main issue is a careful lot sizing to avoid shortages of active ingredients in successive production steps.[2,3] Secondary manufacturing production is usually a *pull* process, driven by wholesalers' orders, in which active ingredients and other components are dispensed, processed and packed to compose the final products.

In this article, we focus on secondary manufacturing systems only, which consist of a set of multi-purpose production facilities that produce a variety of intermediate and finished products through multi-stage production processes. Facilities are linked by supplier-customer relations, and each of them may interact with other external (e.g. suppliers) or internal (e.g. warehouses) entities. The production processes are commonly devoted to produce solid, liquid, aerosol or powdered items according to a family of similar recipes.[9] They are typically organized with three or four main departments (Figure 1). These departments, to a certain extent, can operate independently from each other because intermediate products can be stored in sealed bins.

The preliminary activities of materials preparation are performed in the *dispensing* department.[9] These operations attain to weighing, dosage and preparation of each ingredient in a recipe and represent one of the most important steps in

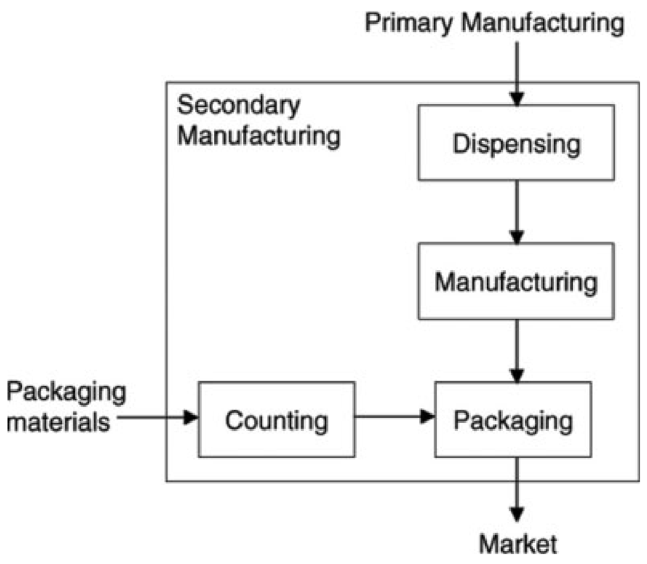

Figure 1. The main production phases in the secondary pharmaceutical manufacturing.

the pharmaceutical process. All recipes must be strictly observed and only approved materials can be used in the operation.

Weighing materials is a time consuming operation requiring workers to pay particular attention for: containment and separation of product from operators; possible cross-contamination; cleaning of booth and equipment and gowning and washing for the operators and separation from the other areas. The main function of the dispensing department is to test materials and release them to production lines. The essential requirement is that material released has been tested and approved by quality controllers for production purposes, and that only this material is used for manufacturing products. The dispensary handles several different classes of material some of which require particular care in the preparation because of their active or dangerous properties. In addition, material may also have to be passed back to the warehouse when only a partial quantity is required from a large container. Finally, the dispensing department provides records that enable a complete audit trail of all the materials dispensed.

The specific production process activities, such as binder preparation, bulk materials granulation and blending, tablet or capsule production, are performed in the *manufacturing* department, whereas activities of counting and packaging are in charge of the *packaging* department.[10] However, when counting activities are relevant, *counting* and packaging activities are performed in different departments. This situation frequently arises in Europe because of the many different national specific rules and languages requiring to handle a huge number of different packages in the same plant.[1,3,9]

Problem Description

In this article, we report our experience with a practical implementation of a scheduling system at a pharmaceutical production plant located in Italy. We consider the production scheduling of two departments: *dispensing* and *counting*. They are the simplest and least critical (from a scheduling point of view) departments in secondary pharmaceutical manufacturing. The plant supplies different European countries and the production flow is organized with the four main phases of Figure 1. However, although there is only one dispensing and one counting department in the plant, manufacturing and packaging activities are organized with several departments. Late and urgent orders are managed as orders with strict deadlines, to be processed as soon as possible. Clearly, deadlines make it difficult to organize long campaigns, and therefore tend to reduce the actual capacity of the departments. This reduction may cause, in turn, late deliveries at the end of the week and such negative effects may propagate over several weeks. All these problems motivate the need for more coordination among the departments and with the planner.

Dispensing department

In the dispensing department the availability of all raw materials required by each recipe is checked.

Raw materials, picked up from a warehouse, are weighed and prepared in sealed bins which are sent to buffers waiting for processing in the manufacturing department. The exceeding raw materials, if any, are sent back to the warehouse unless the following job requires the same components. The weighing operations are performed in two independent rooms in parallel.

Cross-contamination issues require one product at a time being processed in a room and the room cleaned when switching from one product to another. *Minor cleaning* is sufficient when two consecutive products need the same raw materials. The production is organized in groups of products requiring the same raw materials to be scheduled consecutively, which are called *campaigns*. *Major cleaning* is, however, necessary after a maximum number of products of the same type, called the *size* of a campaign.

From the scheduling point of view each room acts as a single machine with sequence-dependent setup times and campaigns. For each production order there may be a release time and a due date or a deadline, when there is a risk of stock-out for customers. There may be planned temporary room unavailability, mainly due to lack of personnel, which must be taken into account when scheduling the production. Note that, while a setup operation can start before unavailability and complete after the interruption, ordinary processing operations cannot be interrupted.

A discussion with the plant managers led to the following objectives. Feasibility is considered in pharmaceutical industry as the first priority because avoiding stock-outs at final customers is of the utter importance. A schedule is considered feasible by the dispatchers when all the deadlines, if any, are respected. The main objective for a feasible schedule is the minimization of the maximum lateness, i.e., the maximum over all products of the difference between the product completion time and its due date. The maximum lateness minimization aims to avoid economical and legal implications of late deliveries. Among these solutions they look for schedules having smaller values of the makespan (in order to grant an higher utilization of the plant) and, in case of further ties, they prefer schedules minimizing the number of late jobs (so to reduce perturbations in the other departments of the plant).

Counting department

The counting department prepares materials required for packaging. Packages, labels and information leaflets are taken from a warehouse, counted and prepared for the subsequent packaging operations. This department typically deals with a much larger number of lots with respect to the dispensing department, because of the number of different packages that must be used in different countries and the presence of different packaging departments of the plant.

The counting department is composed of three independent rooms, although a room may be temporarily unavailable, and there is no significant setup between two consecutive operations.

Also in the counting department there are release times (when the packages become available), deadlines and due dates (propagated backward from the packaging departments).

As in the dispensing department the main goal is the deadline constraints satisfation and the objectives are in lexicographic order minimizing maximum lateness, makespan and number of late jobs.

Scheduling models

The scheduling environment of interest in both departments is the parallel machines case, i.e., the problem is to decide when to start an operation and on which machine under a number of realistic constraints. In this context, industrial examples can be found in Ovacik and Uzsoy (1997),[11] whereas a survey on the parallel machine scheduling research is available in Cheng and Sin (1990).[12] The parallel machine scheduling problem with multiple objectives is discussed by T'kindt and Billaut (2002).[13] The version of the problem with sequence dependent setup times is addressed by Lee and Pinedo (1997)[14] and Tang (1990)[15] which considers also the case with minor and major setups. The latter case is particularly frequent in the pharmaceutical industry in which minor setups are related to successive processing of jobs belonging to the same family, while major setups occur when the families of two successive jobs are different and a more accurate and time/work consuming changeover operation is required.

A formal model for these scheduling problems can be formulated adopting the three fields ($\alpha|\beta|\gamma$) classification scheme of Graham et al.,[16] where α indicates the scheduling environment, β describes the job characteristics or restrictive requirements, and γ defines the objective function to be minimized.

With this notation, the dispensing department can be classified as

$$P_2|r_i, d_i, D_i, s_{ij}, MCS, \text{unavail}|\text{Lex}(L_{\max}, C_{\max}, U),$$

in which, P_2 indicates identical parallel machines production environment with 2 machines; r_i, d_i, and D_i indicate that jobs have release times, due dates, and deadlines, respectively; s_{ij} indicates the presence of sequence-dependent setup times; *MCS* represents the constraints on the maximum campaign size; *unavail* represents the possible room unavailability constraint; the objectives are in lexicographic order: $L_{\max}$ minimization of maximum lateness, $C_{\max}$, makespan minimization, U, minimization of the number of tardy jobs.

Similarly, the counting department can be described as:

$$P_3|r_i, d_i, D_i, \text{unavail}|\text{Lex}(L_{\max}, C_{\max}, U).$$

We notice that, although the dispensing and counting departments are considered simple departments from the operations practice point of view, because they are not critical and by far less complicated than manufacturing and packaging departments, the resulting scheduling problems are considered quite difficult NP-hard problems in the scheduling literature.[17,18] Moreover, some constraints as the maximum size of a campaign or the particular machine unavailability that allows to resume only cleaning operations are not frequently addressed in the scheduling literature.[2]

Solution Methods

In this section we describe a general approach to design scheduling algorithms that are easily adaptable, modular and suitable for incorporating human experience in the computerized methods. Simple heuristics are more easily accepted and trusted by the human schedulers, who can understand their principles and suggest modifications to improve performance over time. In these methods, the search process is performed by several heuristic procedures guided by a general optimization strategy. The method is applied to the cases under study, and the following subsections are devoted to describe in detail the characteristics of the considered algorithms.

```
Modified Jackson Schedule (MJS)
Input: a set $P$ of production orders;
$t = \min\{r_i : i \in P\}$, $S = \emptyset$
repeat
    $R = \{i \in P : r_i \leq t\}$
    if there is a job in $R$ subject to a deadline constraint then
        select a job $j \in R$ with the smallest deadline
        in case of ties, select a job with the smallest setup
    else    if there is a job in $R$ subject to a due date constraint then
                select a job $j \in R$ with the smallest due date
                in case of ties, select a job with the smallest setup
            else $t = \min\{r_i : i \in P\}$, $R = \{i \in P : r_i \leq t\}$
    if a job has been selected then
        assign $j$ to the machine able to complete it first, taking into account (if any)
        setups, campaigns and machine availability, $S = S \cup \{j\}$, $P = P \setminus \{j\}$
        update $t$ to the smallest completion time of all available machines
until $P = \emptyset$
```

Figure 2. Algorithmic scheme of the Modified Jackson Schedule.

Constructive algorithms for dispensing and counting

We focus on greedy algorithms to produce a solution. Greedy algorithms typically sort the operations according to a given *criterion* and then, starting from the empty solution, build a complete schedule by adding to the partial schedule one operation at a time, according to the order induced by the adopted criterion.

We introduce two greedy algorithms developed for scheduling the production in the dispensing and counting departments. We developed a modified version of the Jackson Schedule[19] algorithm and we refer to it as *MJS*. Details on the MJS algorithm are illustrated in Figure 2. The heuristic criterion sorts all the jobs that are available to be scheduled first according to their priority (i.e., the presence of a deadline) and second by the smallest due date value. The dispatching rule assigns the selected job to the machine which is able to complete it first. Clearly, the completion time of a job takes into account the presence of sequence-dependent setup times, campaigns and machine unavailabilities. We observed that, the schedules produced by hand in the counting department were quite similar to that of the MJS algorithm.

Algorithm *Delta* extends the MJS algorithm by decomposing the processing horizon into intervals of length Δ, and by replacing the due dates (and deadlines) in the interval [$k\Delta$, $(k+1)\Delta$] with $k\Delta$. Then, it schedules jobs according to the MJS algorithm. This means that in the preprocessing step the production orders having similar due dates are grouped

```
Algorithm Delta ($\Delta$)
Input: a set $P$ of production orders;
Forall jobs $i \in P$ compute modified due dates and deadlines: $\bar{d}_i = \lfloor \frac{d_i}{\Delta} \rfloor \Delta$; $\bar{D}_i = \lfloor \frac{D_i}{\Delta} \rfloor \Delta$
$t = \min\{r_i : i \in P\}$, $S = \emptyset$
repeat
  $R = \{i \in P : r_i \leq t\}$
  if there is a job in $R$ subject to a deadline constraint then
    select a job $j \in R$ with the smallest modified deadline,
    in case of ties, select a job with the smallest setup
  else    if there is a job in $R$ subject to a due date constraint then
            select a job $j \in R$ with the smallest modified due date,
            in case of ties, select a job with smallest the setup
          else $t = \min\{r_i : i \in P\}$, $R = \{i \in P : r_i \leq t\}$
  if a job has been selected then
    assign $j$ to the machine able to complete it first, taking into account setups,
    campaigns and machine availability, $S = S \cup \{j\}$, $P = P \setminus \{j\}$
    update $t$ to the smallest completion time of all available machines
until $P = \emptyset$
```

Figure 3. The scheme of the algorithm Delta.

together. Hence, a larger value of Δ favors the formation of larger campaigns, whereas setting $\Delta = 0$ corresponds to the MJS algorithm. In Figure 3 we show the details of the algorithm Delta.

Algorithm Delta surrogates the schedules produced by hand in the dispensing department. In fact, in the dispensing department the schedulers strive to obtain large campaigns other than respecting the due dates.

The results of MJS and Delta heuristics were considered feasible, although not particularly performing, by the schedulers. We used the rollout method of Figure 4 to improve the performance of these basic heuristics. It is well known that greedy heuristics may exhibit an erratic behavior, possibly because (*i*) locally promising decisions may not lead to good global solutions, and (*ii*) wrong choices can not be changed anymore in the solution process. The main idea of the *Rollout* algorithm[4,20] or *Pilot* method[6,21] is to overcome or, at least, mitigate these limitations by means of a look-ahead strategy. More specifically, the partial solution is iteratively enlarged by using one or more greedy algorithms as a look-ahead strategy. This approach can be viewed also as an approximated dynamic programming method.[22,23] These algorithms have been applied to different problems, such as stochastic vehicle routing problems,[24] test sequencing for fault diagnosis applications,[25] general versions of job shop scheduling problems.[26]

We next illustrate the rollout/pilot method for a parallel machine scheduling problem. A solution S is a schedule of m machines and n jobs. Starting from the empty solution (with no fixed components), the k-th iteration consists in evaluating a *scoring function* $p(j)$ for each job j which can be added to the current partial solution $S_{k\text{-}1}$ and choosing the one having the smallest scoring function. The iteration is repeated, for $k = 1$ to n, when a complete solution is found. Given a partial solution $S_{k\text{-}1}$, and let j be a possible job to be added to the schedule, a pilot heuristic $H(\cdot)$ is a constructive algorithm that, starting from the partial solution S_k in which job j has been added according to the heuristic dispatching rule, produces a complete solution with objective function value $p(j) = H(S_{k\text{-}1},j)$. At the end of the k-th iteration the job j_0 associated to the smallest $H(S_{k\text{-}1},j)$ is permanently added to the partial solution. Figure 4 shows a sketch of the rollout algorithm using a single pilot heuristic applied to the problem of sequencing a set J of jobs and m machines.

```
Algorithm Pilot/Rollout
begin
for $k = 1, \ldots, |J|$ do
  begin
  $p(j_0) = +\infty$
  for all $j \in J$ do
    if $(p(j) = H(S_{k-1}, j)) < p(j_0)$ then $j_0 = j$
  $S_k = \{S_{k-1} \cup j_0\}$, $J = J \setminus \{j_0\}$
  end
end
```

Figure 4. Algorithmic scheme of Pilot/Rollout

Local search procedures

Another well known technique to improve a given solution is based on the *local search* principle. Local search algorithms are based on a neighborhood structure N. More specifically, we consider three neighborhood structures associated with the following moves, aiming at improving the single machine schedules independently from each other.

- SWAP: Two adjacent jobs assigned on the same machine M_x, say j_k and j_{k+1}, are swapped. The rest of the schedule remains unchanged.
- INSERT: A job j_h is removed from its current position in the schedule of M_x and it is inserted on the same machine, before or after another operation j_k.
- EXCHANGE: Two (nonadjacent) jobs assigned on the same machine M_x, say j_h and j_k, are swapped, i.e., j_h replaces j_k and vice versa. The swap move is therefore a particular exchange move.

Figure 5 provides an illustration of the proposed moves.

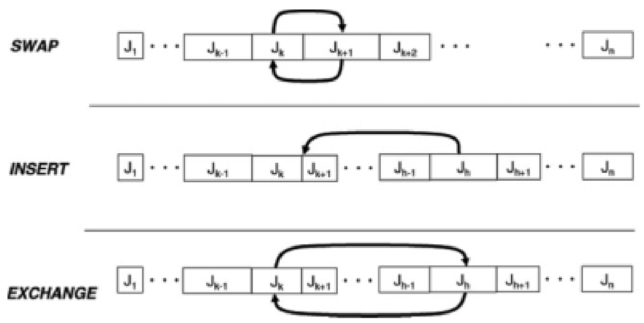
SWAP
J1 · · · Jk-1 Jk Jk+1 Jk+2 · · · · · · Jn
INSERT
J1 · · · Jk-1 Jk Jk+1 · · · Jh-1 Jh Jh+1 · · · Jn
EXCHANGE
J1 · · · Jk-1 Jk Jk+1 · · · Jh-1 Jh Jh+1 · · · Jn

Figure 5. An illustration of the three moves.

Variable Neighborhood Descent (VND)

```
**Input:** a set of neighborhoods: $\{\mathcal{N}_1 \ldots \mathcal{N}_k\}$, and an initial solution;
$i := 1$
**repeat**
    search for a profitable move in $\mathcal{N}_i$
    **if** an improving move is found **then**
        apply the move
        $i := 1$
    **else** $i := i + 1$
**until** $i \geq k$
```

Figure 6. Algorithmic scheme of Variable Neighborhood Descent (VND).

Local search algorithms converge to a local minimum, which may be of poor quality. To contrast this drawback, a variety of strategies have been proposed to escape from local minima. The *Variable Neighborhood Descent* (VND) is a local search technique focused on systematic neighborhood changes.[7,8] It is based on the observation that a local minimum for a given neighborhood is not necessarily a local minimum when other neighborhoods are considered. More specifically, let $\{N_1 \ldots N_k\}$ be a set of neighborhood structures. The VND starts the search process from an initial solution in the first neighborhood $i = 1$, i.e., N_1. In the generic iteration, the algorithm scans the i-neighborhood looking for an improving move; once an improving move is detected it is performed and the VND starts searching again in the first neighborhood (N_1). If no such improving move exists, the algorithm starts searching in the (i+1)-neighborhood. The search process terminates when no improving moves are available in all the considered neighborhoods. The sketch of the algorithm is given in Figure 6.

In our implementation the three neighborhoods are ordered according to their size, i.e., SWAP, INSERT and EXCHANGE.

Computational Results

In what follows, we study four different algorithms obtained by combining the three building blocks previously described:

- Algorithm H consists in the application of a simple greedy heuristic, either MJS or Delta.
- Algorithm RH is the rollout algorithm using H as pilot heuristic.
- Algorithm H+LS corresponds to improving the solution of the greedy algorithm by applying the VND procedure.
- Finally, we call RH+LS the rollout algorithm in which the final solution is used as initial solution by the VND algorithm.

For each department these algorithms have been tested on 6 sets of 15 randomly generated realistic instances, each instance representing 2 weeks of planned production. The instances have been generated respecting a typical production mix of the plant. Each test instance in the dispensing department has a number of jobs ranging from 40 to 60. The number of jobs in the counting department, for each instance, ranges from 400 to 480. These sizes represent the typical number of jobs produced in the two departments in a two-week production horizon. The instances can be grouped according to the presence of only one or more machine planned unavailabilities in the planning horizon. Note that these interruptions are known in advance. For each group we consider three sets of 15 instances having different number of jobs with higher priority, i.e., having a deadline. More specifically, we consider instances *(i)* having no deadline, *(ii)* in which 20% of the jobs have a deadline constraint, and *(iii)* the most constrained set in which 40% of the jobs have a deadline. All the remaining jobs have a due date. The time in which a given job must be completed, i.e., the difference between due date (deadline) and release time $d_i - r_i$, ranges from one to three production time shifts (each production time shift is set to 16 consecutive hours). All the algorithms are written in C and run on a AMD X2 processor with 2.2 GHz with Linux operating system.

In Tables 1–4 (5–8), we present the results of the experimental campaign on the dispensing (counting) department, respectively. Each row of the tables represents the results obtained by applying a different greedy heuristic. Namely, we report on the results of MJS (which corresponds to setting $\Delta = 0$ in the Delta algorithm), and two versions of the Delta algorithm obtained by setting Δ equal to 8 hours (i.e., half time shift) and to the average processing time of all the jobs in the instance. We refer to them as Δ_{8h} and Δ_{avg}, respectively. For each algorithm we present the computation time (Time) in seconds, the objective functions values (maximum lateness L_{max}, makespan C_{max} and number of tardy jobs U) and the percentage of feasible instances (%feas), i.e., instances in which the deadline constraints are not violated. All the values in each row of the tables are the average over 15 instances all having the same amount of deadline

Table 1. Dispensing Department: Single Interruption (H and H+LS)

		H					H+LS				
Heuristic	%Deadline	Time	L_{max}	C_{max}	U	%feas	Time	L_{max}	C_{max}	U	%feas
MJS	0.0	<0.01	3742.13	20,672.47	10.47	1.00	<0.01	3664.87	20,595.20	9.67	1.00
Δ_{8h}	0.0	<0.01	3762.60	20,692.93	10.67	1.00	<0.01	3685.33	20,615.67	9.80	1.00
Δ_{avg}	0.0	<0.01	3762.60	20,692.93	10.67	1.00	<0.01	3685.33	20,615.67	9.80	1.00
MJS	0.2	<0.01	4313.87	20,813.53	14.47	1.00	<0.01	4134.47	20,725.40	12.13	1.00
Δ_{8h}	0.2	<0.01	3965.47	20,794.87	11.60	0.67	<0.01	3799.87	20,643.93	8.47	0.73
Δ_{avg}	0.2	<0.01	3965.47	20,794.87	11.60	0.67	<0.01	3799.87	20,643.93	8.47	0.73
MJS	0.4	<0.01	4292.13	20,684.00	16.53	1.00	<0.01	3977.80	20,556.87	14.20	1.00
Δ_{8h}	0.4	<0.01	3789.00	20,667.20	8.20	0.73	<0.01	3672.20	20,561.20	7.06	0.73
Δ_{avg}	0.4	<0.01	3789.00	20,667.20	8.20	0.73	<0.01	3672.20	20,561.20	7.06	0.73

Table 2. Dispensing Department: Single Interruption (RH and RH+LS)

		RH					RH + LS				
Heuristic	%Deadline	Time	L_{max}	C_{max}	U	%feas	Time	L_{max}	C_{max}	U	%feas
MJS	0.0	0.53	2625.27	19,568.07	5.27	1.00	0.53	2625.27	19,568.07	5.27	1.00
Δ_{8h}	0.0	0.54	2708.93	19,651.73	6.67	1.00	0.54	2708.93	19,651.73	6.07	1.00
Δ_{avg}	0.0	0.54	2708.93	19,651.73	6.67	1.00	0.54	2708.93	19,651.73	6.07	1.00
MJS	0.2	0.53	3192.53	19,878.40	7.20	1.00	0.53	3140.13	19,878.40	6.73	1.00
Δ_{8h}	0.2	0.54	2972.13	19,879.73	4.73	1.00	0.54	2972.13	19,879.73	4.67	1.00
Δ_{avg}	0.2	0.53	2972.13	19,879.73	4.73	1.00	0.53	2972.13	19,879.73	4.67	1.00
MJS	0.4	0.51	3082.80	19,758.33	9.26	1.00	0.51	3050.13	19,779.60	9.00	1.00
Δ_{8h}	0.4	0.51	3047.60	19,895.06	5.26	0.80	0.51	3044.00	19,902.80	5.20	0.80
Δ_{avg}	0.4	0.56	3047.60	19,895.06	5.26	0.80	0.56	3044.00	19,902.80	5.20	0.80

Table 3. Dispensing Department: Multiple Interruptions (H and H+LS)

		H					H+LS				
Heuristic	%Deadline	Time	L_{max}	C_{max}	U	%feas	Time	L_{max}	C_{max}	U	%feas
MJS	0.0	<0.01	3336.73	20,296.40	15.80	1.00	<0.01	3258.27	20,224.07	13.80	1.00
Δ_{8h}	0.0	<0.01	3336.73	20,296.40	15.80	1.00	<0.01	3258.27	20,224.07	13.80	1.00
Δ_{avg}	0.0	<0.01	3336.73	20,296.40	15.80	1.00	<0.01	3258.27	20,224.07	13.80	1.00
MJS	0.2	<0.01	3796.53	20,124.53	16.00	0.93	<0.01	3438.20	20,061.33	13.60	1.00
Δ_{8h}	0.2	<0.01	3297.33	19,984.53	9.80	0.73	<0.01	3240.33	19,911.46	8.46	0.73
Δ_{avg}	0.2	<0.01	3297.33	19,984.53	9.80	0.73	<0.01	3240.33	19,911.46	8.46	0.73
MJS	0.4	<0.01	4765.33	20,297.46	19.80	0.93	<0.01	4311.40	20,224.40	17.60	0.93
Δ_{8h}	0.4	<0.01	3350.06	20,280.40	14.13	0.33	<0.01	3253.00	20,173.20	11.53	0.40
Δ_{avg}	0.4	<0.01	3350.07	20,280.40	14.13	0.33	<0.01	3253.00	20,173.20	11.53	0.40

constraints, as indicated in the second column (%Deadline) of the Tables.

In Tables 1–4, we show the results of the proposed algorithms on the test instances for the dispensing department. The computation time never exceeds one second when a rollout algorithm is applied, and it is negligible when H and H+LS algorithms are applied. As far as the solution quality is concerned, we can see that MJS heuristic performs worse than Delta variants when deadlines are present, although tends to find more feasible solutions. We observe that when the Delta heuristic is used it is able to obtain better results than MJS and the two variants (Δ_{8h} and Δ_{avg}) yield exactly to the same solutions. However, when studying the effects of the different algorithmic schemes the performance indicators obtained by RH+LS are greatly improved: The maximum lateness and the makespan are reduced of about 1000 minutes, the number of late jobs is also significantly reduced. A feasible solution is always found by RH and RH+LS algorithms with the only exception of the more constrained instances, i.e., having 40% of jobs with deadline and multiple interruptions. From results reported in Tables 2 and 4 it is clear that the main contribution to these improvements is due to the rollout algorithm.

In Tables 5 and 6, we show the results obtained for the counting department when a single interruption is present, whereas in Tables 7 and 8, we report on the results for instances with multiple interruptions. When used as standalone heuristic (see Tables 5 and 7) Δ_{avg} is the heuristic able to produce better results. However, when tackling instances without deadlines MJS attains similar results. In the counting department, the algorithms are able to find always a solution respecting the deadline constraints. Regarding computation times, we note that more than half an hour is needed to terminate the rollout procedure. This is clearly due to the larger number of jobs in these instances. However, such computation time does not affect the applicability of this approach because the amount of time in which a solution should be produced by the automated scheduling system is usually

Table 4. Dispensing Department: Multiple Interruptions (RH and RH+LS)

		RH					RH + LS				
Heuristic	%Deadline	Time	L_{max}	C_{max}	U	%feas	Time	L_{max}	C_{max}	U	%feas
MJS	0.0	0.51	2852.27	19,852.20	9.87	1.00	0.51	2851.93	19,851.87	9.80	1.00
Δ_{8h}	0.0	0.51	2852.27	19,852.20	9.87	1.00	0.51	2851.93	19,851.87	9.80	1.00
Δ_{avg}	0.0	0.51	2852.27	19,852.20	9.87	1.00	0.51	2851.93	19,851.87	9.80	1.00
MJS	0.2	0.51	2670.46	19,576.06	8.73	1.00	0.51	2664.26	19,575.73	8.40	1.00
Δ_{8h}	0.2	0.51	2643.60	19,591.40	5.60	1.00	0.51	2643.60	19,591.40	5.60	1.00
Δ_{avg}	0.2	0.56	2643.60	19,591.40	5.60	1.00	0.56	2643.60	19,591.40	5.60	1.00
MJS	0.4	0.45	3413.13	19,562.93	11.87	0.93	0.45	3370.13	19,555.13	11.53	0.93
Δ_{8h}	0.4	0.51	2910.93	19,854.60	9.27	0.51	0.60	2908.47	19,852.13	9.20	0.60
Δ_{avg}	0.4	0.57	2910.93	19,854.06	9.27	0.57	0.67	2908.47	19,852.13	9.20	0.60

Table 5. Counting Department: Single Interruption (H and H+LS)

		H					H+LS				
Heuristic	%Deadline	Time	$L_{\max}$	$C_{\max}$	U	%feas	Time	$L_{\max}$	$C_{\max}$	U	%feas
MJS	0.0	0.04	924.47	17,779.67	5.40	1.00	0.04	924.07	17,779.27	5.40	1.00
Δ_{8h}	0.0	0.04	1014.13	17,786.20	6.20	1.00	0.04	1010.80	17,782.87	6.07	1.00
Δ_{avg}	0.0	0.04	924.73	17,779.93	5.27	1.00	0.04	924.33	17,779.53	5.13	1.00
MJS	0.2	0.04	527.93	17,312.73	3.00	1.00	0.07	509.80	17,312.67	3.00	1.00
Δ_{8h}	0.2	0.04	526.27	17,310.47	4.07	1.00	0.06	526.27	17,310.47	4.07	1.00
Δ_{avg}	0.2	0.04	447.00	17,319.13	3.27	1.00	0.06	446.93	17,319.07	3.00	1.00
MJS	0.4	0.03	1064.66	17.549.86	9.73	1.00	0.09	1045.80	17,549.33	9.20	1.00
Δ_{8h}	0.4	0.04	763.33	17,553.33	4.46	1.00	0.07	763.33	17,553.33	4.46	1.00
Δ_{avg}	0.4	0.04	668.66	17,559.00	3.00	1.00	0.07	665.53	17,555.86	2.86	1.00

Table 6. Counting Department: Single Interruption (RH and RH+LS)

		RH					RH + LS				
Heuristic	%Deadline	Time	$L_{\max}$	$C_{\max}$	U	%feas	Time	$L_{\max}$	$C_{\max}$	U	%feas
MJS	0.0	2214.80	605.13	17,571.40	2.40	1.00	2214.80	605.13	17,571.40	2.40	1.00
Δ_{8h}	0.0	2176.31	716.73	17,510.80	3.80	1.00	2176.31	716.73	17,510.80	3.80	1.00
Δ_{avg}	0.0	2163.06	605.13	17,571.40	2.40	1.00	2163.07	605.13	17,571.40	2.40	1.00
MJS	0.2	2148.28	459.13	17,257.53	1.80	1.00	2148.31	457.87	17,257.53	1.80	1.00
Δ_{8h}	0.2	2208.72	453.33	17,256.07	2.67	1.00	2208.74	453.33	17,256.07	2.60	1.00
Δ_{avg}	0.2	2230.41	358.13	17,280.60	1.40	1.00	2230.43	358.13	17,280.60	1.40	1.00
MJS	0.4	2160.33	507.40	17,057.00	4.93	1.00	2160.62	499.80	17,057.00	4.8	1.00
Δ_{8h}	0.4	2291.77	241.13	17,056.80	1.66	1.00	2291.80	241.13	17,056.80	1.66	1.00
Δ_{avg}	0.4	2282.13	139.20	17,089.86	1.13	1.00	2282.16	139.20	17,089.86	1.13	1.00

larger. Moreover, we observe that the LS phase requires always a very short time. On the other hand, the solution quality of the rollout algorithms is significant better than the H+LS. We also observe that a small difference arises when comparing RH and RH+LS, whereas a more significative effect is due to the use of different pilot heuristics.

In Figures 7 and 8, we plot the improvements over the basic stand-alone heuristics due to the proposed improvement schemes for the dispensing and counting department, respectively. The percentage improvement is obtained as follows: $1-(x_v^i/x_H^i)$, where $i \in\{L_{\max}, C_{\max}, U\}$, $v = \{\text{H+LS, RH, RH+LS}\}$ and x_v^i is the solution quality according to the

Table 7. Counting department: Multiple Interruption (H and H+LS)

		H					H+LS				
Heuristic	%Deadline	Time	$L_{\max}$	$C_{\max}$	U	%feas	Time	$L_{\max}$	$C_{\max}$	U	%feas
MJS	0.0	0.04	962.93	17,897.20	8.73	1.00	0.04	961.80	17,896.07	8.53	1.00
δ_{8h}	0.0	0.04	1056.20	17,869.33	9.27	1.00	0.04	1055.00	17,868.13	9.27	1.00
δ_{avg}	0.0	0.04	962.93	17,897.20	8.73	1.00	0.04	961.80	17,896.07	8.53	1.00
MJS	0.2	0.03	1242.06	18,136.80	18.40	1.00	0.06	1239.73	18,136.00	18.26	1.00
δ_{8h}	0.2	0.04	1319.13	18,136.13	13.13	1.00	0.05	1319.13	18,136.13	12.80	1.00
δ_{avg}	0.2	0.04	1188.73	18,135.40	11.60	0.86	0.05	1188.06	18,134.73	11.33	0.86
MJS	0.4	0.04	2124.60	18,610.27	40.13	1.00	0.09	2087.40	18,608.93	39.53	1.00
δ_{8h}	0.4	0.04	2214.67	19,029.73	9.00	1.00	0.07	2212.07	19,027.13	8.87	1.00
δ_{avg}	0.4	0.04	1635.20	18,599.67	5.80	0.93	0.07	1631.13	18,595.60	5.53	0.93

Table 8. Counting Department: Multiple Interruptions (RH and RH+LS)

		RH					RH+LS				
Heuristic	%Deadline	Time	$L_{\max}$	$C_{\max}$	U	%feas	Time	$L_{\max}$	$C_{\max}$	U	%feas
MJS	0.0	2162.51	873.93	17,847.47	5.80	1.00	2162.52	873.93	17,847.47	5.80	1.00
Δ_{8h}	0.0	2227.34	990.60	17,807.40	6.67	1.00	2227.34	990.60	17,807.40	6.67	1.00
Δ_{avg}	0.0	2232.61	873.93	17,847.47	5.80	1.00	2232.61	873.93	17,847.47	5.80	1.00
MJS	0.2	2156.50	962.06	17,841.33	11.60	1.00	2156.52	959.73	17,841.33	11.60	1.00
Δ_{8h}	0.2	2276.41	1014.13	17,828.33	8.73	1.00	2276.43	1014.13	17,828.33	8.73	1.00
Δ_{avg}	0.2	2254.65	895.13	17,864.06	7.93	0.93	2254.67	895.06	17,864.06	7.86	0.93
MJS	0.4	2159.68	1482.87	17,420.80	27.13	1.00	2159.72	1475.13	17,420.80	26.87	1.00
Δ_{8h}	0.4	2289.84	612.73	17,427.73	3.53	1.00	2289.87	612.73	17,427.73	3.47	1.00
Δ_{avg}	0.4	2296.64	658.47	17,633.53	2.20	1.00	2296.67	658.47	17,633.53	2.20	1.00

objective function i obtained by algorithm v. Hence, percentage improvement on $L_{\max}$ possibly gets values $>100\%$ when, starting from $L_{\max} \geq 0$, a new solution with $L_{\max} < 0$ is obtained. In the dispensing department, the main contribution is given by the application of the rollout algorithm, although the use of a VND allows to slightly improve the solution quality without increasing significantly the computation time. For both departments, the algorithms are able to obtain relevant improvements in $L_{\max}$ and U objectives. More specifically, $L_{\max}$ is improved up to 36% on average, and U is improved of more than 40%, whereas the $C_{\max}$ objective improves only up to 3%.

The lack of detailed information on the performance of the previous scheduling process does not allow a direct and complete comparison with the new system. As far as the dispensing and the counting departments are concerned, the assessment of the new system was based on the feedback from the department managers. Specifically, the solutions provided by the stand-alone greedy algorithms were considered not satisfactory, because it was simple for the users to improve the solutions with frequent and tedious local changes in the schedules. On the other hand, the schedules obtained after the rollout and VND phases were rarely improved by users. The increase of capacity for the dispensing department has been estimated up to one hour in a day by the department manager, which approximately corresponds up to the 2% of productivity increase. Similar results were obtained for the counting department.

Conclusions

In this article, we described a production scheduling problem arising in secondary pharmaceutical manufacturing. More specifically, we addressed the automation of the scheduling process for the dispensing and counting departments. Although these departments are conceived as noncritical by the plant managers they result in a quite complex scheduling problem involving both standard and complex uncommon constraints. Any improvement in these departments is directly translated into a relaxation of the timing constraints for more critical departments. We proposed an algorithmic technique that can be effectively used to rapidly automate the production scheduling process, and we presented a practical implementation in a pharmaceutical manufacturing plant. Results are encouraging and show a substantial improvement over simple heuristic techniques surrogating the schedules produced by hand by the plant schedulers.

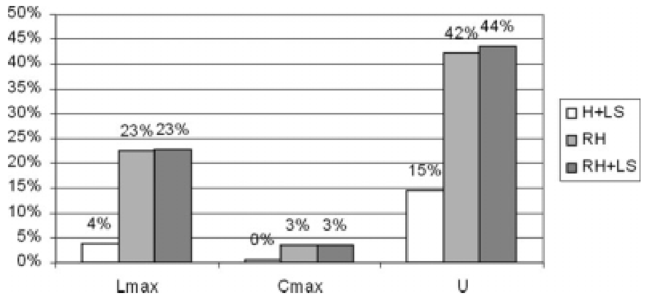

Figure 7. Average improvements in the dispensing department.

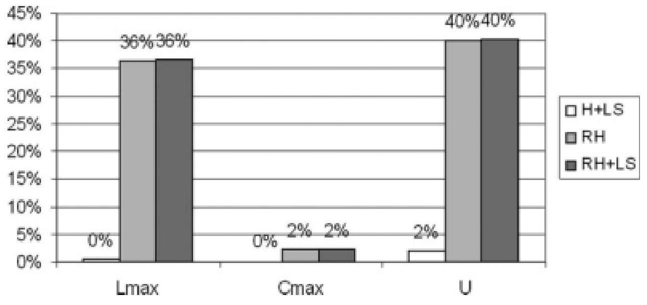

Figure 8. Average improvements in the counting department.

Acknowledgements

Authors wish to thank Dario Pacciarelli (Università Roma Tre) for very fruitful discussions on the research leading to this article. Finally, authors thank an anonymous referee for his comments on the first version, which helped them to improve the overall quality of the manuscript.

Literature Cited

1. Piachaud B. *Outsourcing of R&D in the Pharmaceutical Industry: From Conceptualization to Implementation of the Strategic Sourcing Process*. Houndsmill, Basingstoke, Hampshire, U.K.: Palgrave Macmillan, 2005.
2. Stefansson H, Shah N, Jensson P. Multiscale planning and scheduling in the secondary pharmaceutical industry. *AIChE J*. 2006;52: 4133–4149.
3. Shah N. Pharmaceutical supply chains: key issues and strategies for optimisation. *Comput Chem Eng*. 2004;28:929–941.
4. Bertsekas DP, Tsitsiklis JN, Wu C. Rollout algorithms for combinatorial optimization. *J Heuristics*. 1997;3:245–262.
5. Duin C, Voβ S.*Steiner tree heuristics—a survey*. In: Dyckhoff H, Derigs U, Salomon M, Tijms HC, editors. *Operations Research Proceedings* 1993. Berlin: Springer, 1994:485–496.
6. Duin C, Voβ S. The pilot method: a strategy for heuristic repetition with application to the Steiner problem in graphs. *Networks*. 1999;34:181–191.
7. Hansen P, Mladenović N. Variable neighborhood search: principles and applications. *Eur J Oper Res*. 2001;130:449–467.
8. Hansen P, Mladenović N. A Tutorial on Variable Neighborhood Search. Les Cahiers du GERAD G-2003 46. ISSN 0711-2440. 2003.
9. Cole GC. *Pharmaceutical Production Facilities. Design and Applications*, 2nd ed. Boco Raton, FL: CRC Press, 1998.
10. Venditti L, Meloni C, Pacciarelli D. A tabu search algorithm for scheduling pharmaceutical packaging operations. Technical Report RT-DIA-130, DIA Università Roma Tre, Roma, 2008.
11. Ovacik IM, Uzsoy R. *Decomposition Methods for Complex Factory Scheduling Problems*. Boston, MA: Kluwer, 1997.
12. Cheng TCE, Sin CCS. A state-of-the-art review of parallel-machine scheduling research. *Eur J Oper Res*. 1990;47:271–292.
13. T'kindt V, Billaut J-C. *Multicriteria Scheduling. Theory, Models and Algorithms*. Berlin, Germany: Springer, 2002.
14. Lee YH, Pinedo M. Scheduling jobs on parallel machines with sequence-dependent setup times. *Eur J Oper Res*. 1997;100:464–474.
15. Tang CS. Scheduling batches on parallel machines with major and minor set-ups. *Eur J Oper Res*. 1990;46:28–37.
16. Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG. Optimization and approximation in deterministic machine scheduling: a survey. *Ann Discrete Math*. 1979;5:287–326.
17. Garey MR, Johnson DS. *Computers and Intractability: A Guide to the Theory of NP Completeness*. New York: Freeman, 1979.
18. Pinedo M. *Scheduling: Theory, Algorithms, and Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
19. Jackson JR. Scheduling a production line to minimize maximum tardiness. Research Report 43, Management Science Research Project, University of California, Los Angeles, 1955.

20. Bertsekas DP, Tsitsiklis JN. *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.
21. Voβ S, Fink A, Duin C. Looking ahead with the pilot method. *Ann Oper Res*. 2005;136:285–302.
22. Bertsekas DP. Dynamic programming and suboptimal control: a survey from ADP to MPC. *Eur J Control*. 2005;11:310–334.
23. Choi J, Realff MJ, Lee JH. Approximate dynamic programming: application to process supply chain management. *AIChE J*. 2006;52:2473–2485.
24. Secomandi N. Analysis of a rollout approach to sequencing problems with stochastic routing applications. *J Heuristics*. 2003;9:321–352.
25. Tu F, Pattipati KR. Rollout strategies for sequential fault diagnosis. *IEEE Trans Syst Man Cybern A*. 2003;33:86–99.
26. Meloni C, Pacciarelli D, Pranzo M. A rollout metaheuristic for job shop scheduling problems. *Ann Oper Res*. 2004;131:215–235.

Manuscript received Aug. 17, 2007, revision received Aug. 27, 2008, and final revision received Oct. 8, 2008.