Optimal Batch Scheduling of a Multiproduct Dairy Process using a combined Optimization/Constraint Programming Approach

T. Escobet[a], V. Puig[a], J. Quevedo[a], P. Palà-Schönwälder[b], J. Romera[a], W.Adelman[c]

[a]*Research Center for Supervision, Safety and Automatic Control (CS2AC). Rambla Sant Nebridi, s/n, 08022 Terrassa (Spain).*

[b]*Communication Circuits and Systems Research Group (CIRCUIT). Avda. Bases de Manresa, 61-73, 08242 Manresa (Spain).*

[c]*Schwarzwaldmilch GmbH Freiburg, Freiburg, Germany*

Abstract

This paper presents the optimal batch scheduling of a multi-product dairy process using an approach that combines optimization and constraint programming techniques. A suitable model describing the subprocesses and production rules is developed allowing to obtain scheduling constraints relating the production process and the machines available together with their relative efficiencies. After the scheduling problem has been formulated, the batch scheduling of a real powder milk/yogurt process is obtained in an optimal manner using the proposed approach with the objective of meeting customers' deadlines considering the efficiencies/costs of available alternative machines. Results using real consumer orders on some representative scenarios corresponding to the dairy production plant used as a case study are provided. This application shows a formulation closer to the engineering problem description thanks to the constraint-based language that facilitates the adaptation of the optimization objectives and constraints to real applications.

Keywords:
Production batch-scheduling, constraint programming, food processing, multi-objective optimization, cost analysis

November 16, 2018

1. Introduction

Multiproduct multistage batch plants are commonly used in the production of end products in the chemical, pharmaceutical, and food processing industries. In such plants, different products are produced with common equipment and shared resources. As discussed in Hazaras et al. (2013), the utilization of the equipment and resources over time is specified through a production schedule, which specifies the timing and batch size of production tasks. The associated complexities make the generation of production schedules neither an easy nor intuitive task. Compared with other industrial processes, the food industry and, in particular, the dairy industry exhibits specific characteristics such as divergent product structures, seasonal product demand, high demand variability, multiple intermediate products with limited storage capacity feeding many finished goods, high consumer-driven demand variability, long lead times for some packaging materials, and a high level of complexity in the production process(Bilgen and Dogan, 2015). The problem becomes even more complex when the production schedule also has to take into account other aspects such as guaranteeing the required equipment is cleaned at the right time or aiming at energy consumption reduction, among others. For these reasons, the efficient scheduling and operation of such multiproduct (as e.g. milk/yogurt powder) process plant requires a decision support system including consumption monitoring and solution algorithms.

In this paper, the scheduling problem to be solved is stated as a job-shop scheduling where the objective function to be minimized is the schedule span plus the energy/cleaning costs and the main constraints are the sequence of stages required for each order, the machine where each processing stage must be carried out, the delivery data and the cleaning rules. To solve the resulting job-shop scheduling problem, a combined optimization/constraint programming approach is used. Finally, this problem is solved using the solvers available in the IBM ILOG Optimization Suite, initially developed by (Van Hentenryck, 1999) and recently updated by (Laborie, 2009). This optimization suite has a module specifically intended to solve scheduling problems by including in the modeling language elements such as activities, temporal constraints, resources, resource constraints and transitions.

The main contributions of this paper are: (i) A constraint programming formulation for the production scheduling of a multistage multiproduct batch plant considering both production and cleaning tasks; (ii) A multi-

objective optimization formulation considering the production and cleaning costs. And, (iii) the application of the proposed combined approach to a real dairy process plant. The application of the proposed approach to the considered case study shows a formulation closer to the engineering problem description thanks to the constraint-based language that facilitates the adaptation of the optimization objectives and constraints to consider practical situations.

The research presented in this paper has been developed in the context of the European research project EnReMilk[1]. The EnReMilk project proposes an integrated engineering approach aiming at the reduction of water and energy consumption in milk processing. This project focuses on the optimization of emerging and novel engineering technologies in key dairy unit operations to provide significant savings of water and energy, while ensuring food quality and safety. One way for reducing water and energy consumption is finding the optimal scheduling of the production that minimizes the cost while maximizes the production, as presented in this paper.

The structure of the reminder of the paper is as follows: Section 2 presents the state-of-the-art in the scheduling of the type of batch processes considered in the paper. Section 3 presents the case study process description and the scheduling problem statement. Section 4 introduces the proposed solution based on the combined optimization/constraint programming approach. Section 5 describes the results obtained when applying the proposed approach in the considered case study. Finally, Section 6 presents the conclusions and future work.

2. State-of-the-art

It is well known that the generation of the optimal production schedule for dairy processes involving multiproduct batch plants is not an easy task. Several optimization techniques, which are able to perform well in problems of such complexity, providing efficient solutions, have been described in the literature. Some of the most common solution methods are mixed-integer linear programming (MILP) methods, mixed-integer non-linear programming (MINLP) methods, constraint programming (CP) and Heuristic and Metaheuristic methods. Harjunkoski et al. (2014) presents an overview on existing

[1]http://www.enremilk.eu/index.html

modeling and scheduling methodologies and focuses on their industrial applicability. A survey on the application of metaheuristics for optimization in food manufacturing industry can be found in Wari and Zhu (2016b), where logistic planning and scheduling problems represent the majority of the applications. A review of the most relevant literature considering production planning and scheduling applied to dairy industry can be found in Sel and Bilgen (2015) and Sel et al. (2015). The former intends to provide a critical review on quantitative supply chain models within the dairy industry. On the other hand, the latter reviews the most relevant literature considering planning and scheduling problems in yogurt production. Both papers propose the use of CP, offering a more flexible modeling framework.

Nowadays, as is pointed out in Novara et al. (2016), there are still some challenges aimed at capturing more realistic aspects of the scheduling problem and trying to solve bigger size case-studies. Some of the challenges currently being addressed are related to: (i) minimizing the energy consumption, the production cost or the number of cleaning operations; (ii) dealing with large number of product orders, (iii) searching a simultaneous solution of batching and scheduling problems, (iv) the integration of planning and scheduling, and (v) the integration of scheduling and control activities.

Some authors have already provided solutions to these challenges. For example, a mixed-integer linear programming model, based on the definition of families of products, is proposed in Kopanos et al. (2011). In this paper, timing and sequencing decisions are made for product families rather than for products, thus reducing significantly the model size. A hybrid MILP formulation, which combines elements from discrete and continuous time representation, is presented in Silvente et al. (2014). This formulation is proposed for determining optimal decisions in terms of both energy production and consumption of a smart grid. A hybrid MILP/CP approach is proposed in Sel et al. (2015) for solving the integrated production planning and distribution problem for a two-stage semi-continuous set type yogurt production. Okubo et al. (2015) presents two models based on integer programming and constraint programming to solve the scheduling problem considering the energy consumption. In Doganis and Sarimveis (2007), a MILP model is proposed targeting the optimal production scheduling in a single yogurt production line while considering production time and production sequence cost. Deka and Datta (2017) propose an evolutionary algorithm for solving the scheduling problem of a network of heat exchangers under milk fouling as a multi-objective optimization problem for minimizing not only the cleaning cost but

also to minimize both the excess energy consumed on overheating milk and flow rate of heating medium. Wari and Zhu (2016a) uses MILP to solve a multi-week scheduling problem for an ice cream facility. The proposed solution includes the clean-up sessions, weekend breaks and semi-processed products in the model.

As discussed in the introduction, in this paper, a combined optimization/constraint programming approach is applied for solving the problem of determining an optimal batch scheduling of a real multi-product milk/yogurt powder process. Constraint programming is a programming paradigm wherein relations between variables are stated in the form of constraints. Constraints differ from the common primitives of imperative programming languages in that they do not specify a step or sequence of steps to execute, but rather the properties of a solution to be found (Marriott and Stuckey, 1998). In recent years, the constraint programming paradigm has successfully automated the solution of complex combinatorial problems in many domains as diverse as planning, routing, allocating resources, managing time, organizing personnel, cutting materials, blending mixtures, and many others (Ceberio and Kreinovich, 2017; Zeballos et al., 2011). A major advantage of the constraint programming paradigm is that it enables to use the statement of a problem directly to develop scheduling model. When a constraint-based model for describing a problem is developed, the solution of the problem is guided by the constraints themselves that help choosing variables with values that guarantee the constraint satisfaction. Since the model of the problem derives directly from the problem representation, there is much less chance of error or incongruence between the semantics of the problem representation and the semantics of the problem solution.

For batch scheduling problems, as the one presented in this paper, the new constraint programming solvers as the CP Optimizer available in IBM ILOG Optimization Suite (Laborie, 2009), presents some advantages compared with classical mixed-integer optimization approach, as indicated in the studies presented by Ham and Cakici (2016), Ku and Beck (2016) or in Ham et al. (2017). Among other facts regarding the CP approach, the authors highlight the ability to write a more natural formulation of the problem, the flexibility and scalability of the code, and the high level description of the problem close to the engineering one where the search algorithms are automatically embedded in the CP solver. This is the reason for selecting the CP Optimizer in this paper.

3. Case study and problem formulation

In this section, the considered case study is introduced and the problem statement is presented.

3.1. Case study description

The case study is based on a real milk/yoghurt powder production plant that can produce skim milk powder(SMP), whole milk powder (SVP), cream powder (SSP), yogurt and two types of quark (quark and UF-quark).

Milk/yogurt powder production consists in removing the water content at the lowest possible cost under stringent hygienic conditions while retaining all the desirable natural properties of the milk/yogurt, such as: color, flavor, solubility and nutritional value (Pearce, 2000). Milk/yogurt powder can be produced using skim milk or skim milk concentrate. Skim milk contains, typically 91-92% of water, while in the case of skim milk concentrate, water content is about 64%. During powder milk production, this water is removed in two stages, namely an evaporation phase followed by a spray drying process. Figure 1 shows one part of the different processing paths involved in the considered case study.

In the evaporation process, part of the water is removed by boiling the milk under reduced pressure at low temperature. In our case study, two types of evaporators are available, named as ED1 and ED2, respectively. Note from Figure 1 that ED1 can produce all products having two operation ways (named way 1 and way 2 for milk and yogurt/quark powder respectively) while ED2 only can produce milk powder. They differ in the capacity of product to process, ED1 way 1 can process between 5800 to 11000 kg/h, ED1 way 2 can process 11000 kg/h and ED2 can process 13000-20000 kg/h, depending on the product being produced. Our case study includes also two dryers, identified as Tower 1 (TW1) and Tower 2 (TW2), the difference between them being their capacity. Tower 1 can process 700-770 kg/h of milk, whereas Tower 2 can process 3400-4000 kg/h.

Yogurt and quark must be pasteurized and fermented before drying. During these processes, which are critical to guarantee product quality, the pasteurized product is formed, and its textural characteristics and distinct flavor are acquired. In case of yogurt, fermentation time is about 5 hours, whereas in case of quark and UF-quark fermentation, time ranges are 15 h and 18 h, respectively.

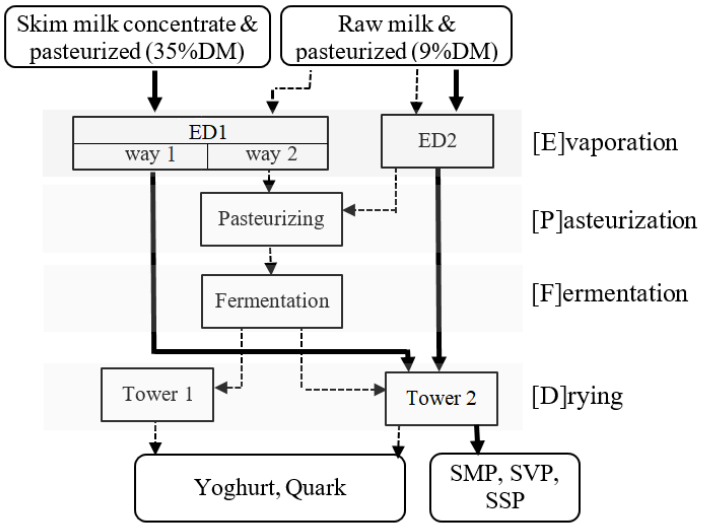

Figure 1: Processing paths and involved production units of the case study process where dashed line corresponds to yoghurt and quark production and continuous line corresponds to SMP, SVP and SSP production.

All the processes in the production units use two types of energy: thermal energy for heating, provided by steam, and electrical energy for pumping and the milk atomizer in the drying process.

As in other food manufacturing processes, milk processing plants have cleaning systems, called Clean-In-Place (CIP), which make use of hot cleaning solutions, usually caustic, combined with acid solutions. During the production processes of milk/yogurt powder, proteins, organic materials and other contaminants are deposited on the surface of vessels, pipework and other associated equipment. The removal of these contaminants is critical for producing an uncontaminated saleable product. In general, the CIP process has its own set of rules, which depend of products to be produced. These rules impose limits on the maximum hours of continuous operation. In the case of evaporators, the optimal production period between CIP cycles is 24 hours and 120 hours for the towers. The duration of each CIP cycle is 4 hours.

After each shutdown, a start-up phase before production is required. The start-up phase takes between 1 and 2 hours and consists in heating the machinery involved in the production process. When the operational tempera-

ture is reached, the production starts.

The factory receives periodically orders from customers. Table 1 shows an example of the orders received where $orderId$ identifies the order, $prod$ is the ordered product, m is the order volume, tr is the release date when the order arrives to the company and td is the due date when the product should be completed to deliver the order on time. Orders must be scheduled at least two weeks in advance. The number of orders during this period is between 80 to 120.

Table 1: Orders received.

$orderId$	$prod$	Family	tr	m [kg]	td
709365	7281	SSP	2/1/2014	16.900,00	10/3/2014
714985	7234	Yoghurt	20/1/2014	1.500,00	12/3/2014
724732	7282	SSP	17/2/2014	5.250,00	17/3/2014
723164	7253	Yoghurt	12/2/2014	2.250,00	7/3/2014
731127	6042	SVP	6/3/2014	7.172,00	14/3/2014

3.2. Problem statement

This research focuses on the production facility presented in the previous section, but it can be easily adapted to other similar production processes.

Each order (Table 1) is described by the tuple

$$O = \langle orderId, prod, tr, m, td \rangle \tag{1}$$

Let $\mathbb{O} = \{O_1, ..., O_{N_o}\}$ be the list of all N_o orders to be scheduled. It is assumed that these N_o data are used to generate the production schedule for a given period, in our case two weeks. It could happen that multiple orders for a given product appear in this list.

The facility produces about N_p different products belonging to a product family set ($\mathbb{F}$) defined as:

$$\mathbb{F} = \{pf_1, pf_2, \cdots, pf_{N_{fp}}\}, \tag{2}$$

with $N_{fp} < N_p$.

Each product family, $pf_i \in \mathbb{F}$, is identified by a code ($pCode$) that has a specific production recipe and is obtained by performing a set of activities (or tasks). Each activity or raw material transformation (act) is supposed to use one or more machines ($machine$) for a given time period or processing time ($pTime$) and a given operation ($pStage$) in a given sequence. Each machine

has an identifier ($machineId$) and an associated cost weight ($wCost$). In some machines, the processing time ($pTime$) depends on the product order quantity. In that case, the resource is characterized by its processing flow ($pFlow$) and material input characteristics or concentration ($pConc$). The model of each process unit (U) is defined by the 10-tuple

$$U = \langle pf, pCode, act, pStage, machine, machineId, pFlow, pConc, pTime, wCost \rangle . \tag{3}$$

Let $\mathbb{U} = \{U_1, , \cdots, U_{N_u}\}$ the list of all the N_u activities to perform to produce all products $\mathbb{F}$.

The process unit characteristics, $\mathbb{U}$, are presented in Table 2, where $w_1, ..., w_7$ are the cost by hour associated to each machine and operation path. Note that $pFlow$ and $pTime$ are mutually excluded, if a machine is characterized by $pFlow$ the time duration $pTime$ is not considered, and conversely.

Table 2: Process unit's characteristics.

pf	$pCode$	act	$pStage$	$machine$	$machineId$	$pFlow$ [to/h]	$pConc$ [%]	$pTime$ [h]	$wCost$ [cost/h]
SMP	1	E	1	ED1w1	1	6,8	30		w_1
SMP	1	E	1	ED2	2	22	9		w_2
SMP	1	D	2	TW2	3	4	44		w_3
SVP	2	E	1	ED1w1	1	5,8	30		w_1
SVP	2	E	1	ED2	2	19	9		w_2
SVP	2	D	2	TW2	3	4	44		w_3
SSP	3	E	1	ED1w1	1	11	9		w_1
SSP	3	E	1	ED2	2	16	9		w_2
SSP	3	D	2	TW2	3	4	44		w_3
Yoghurt	4	E	1	ED1w2	1	11	9		w_4
Yoghurt	4	E	1	ED2	2	13	9		w_2
Yoghurt	4	P	2	Past.	5	8	21		w_5
Yoghurt	4	F	3	Fermt.	6		21	5	w_6
Yoghurt	4	D	4	TW2	3	3,4	21		w_3
Yoghurt	4	D	4	TW1	4	0,5	21		w_7
Quark	5	E	1	ED1w2	1	11	9		w_4
Quark	5	E	1	ED2	2	13	9		w_2
Quark	5	P	2	Past.	5	8	21		w_5
Quark	5	F	3	Fermt.	6		21	15	w_6
Quark	5	D	4	TW2	3	3,4	21		w_3
Quark	5	D	4	TW1	4	0,75	21		w_7
UF-Quark	6	P	2	Past.	5	8	21		w_5
UF-Quark	6	F	3	Fermt.	6		21	18	w_6
UF-Quark	6	D	4	TW2	3	3,4	21		w_3
UF-Quark	6	D	4	TW1	4	0,75	21		w_7

Machinery must be cleaned periodically, using the corresponding CIP

procedure. During CIP, machinery can not be used for production tasks. Each machine has its own cleaning period requirement ($cipPeriod$), cleaning duration time ($cipTime$) and cleaning cost ($cipCost$) as can be seen in Table 3. Thus, the model of each machine cleaning operation (C) is defined by the 4-tuple:

$$C = \langle machine, cipPeriod, cipTime, cipCost \rangle . \tag{4}$$

Let $\mathbb{C} = \{C_1, , \cdots, C_{N_m}\}$ be the list of all the N_m machines with the CIP information.

Table 3: Process unit's characteristics.

machine	*cipPeriod* [h]	*cipTime* [h]	*cipCost* [cost/h]
ED1	24	4	w_{10}
ED2	24	4	w_{11}
TW1	120	4	w_{12}
TW2	120	4	w_{13}
Past	120	4	w_{14}
Fermt	120	4	w_{15}

The optimal scheduling problem is formulated to minimize energy and cleaning costs while satisfying the customer's delivery due data. The scheduling problem considered in this paper is defined as follows. Given

- a time horizon,
- the available process units (machines) and their capacities,
- the production recipe,
- the material orders with the delivery data and product family,
- sequence-depending production activities,
- plant wide cleaning duration and requirements,

determine

- the optimal sequence, grouping and timing of all active production activities,
- the timing of cleaning activities minimizing the number,
- minimizing the electrical and thermal consumption of each process unit.

4. Proposed solution

In this section, the proposed approach to solve the scheduling problem formulated in the previous section is presented.

The use of the constraint programming approach entails two relatively distinct activities:

- *Problem Representation.* A problem representation consists of the declaration of the unknown (decision) variables and the constraints of the problem constraining them. This representation is specific to the problem domain under consideration and requires a very expressive programming language to capture that specificity.

- *Solution Search.* Solving the problem consists of selecting a value in the domain of each constrained variable, so that all the constraints are satisfied. In many cases, it also is necessary to search for a solution that optimizes a given criterion thus combining the constraint programming approach with optimization, as proposed in this paper.

These two activities are described in the next subsections for the considered case study.

4.1. Problem representation

The representation of the scheduling problem presented in Section 3.2 is based on the development of two models, namely, the task and the machine models.

On the one hand, the task model is a list with the collection of all operations (tasks) required for producing each ordered product including the CIP tasks. Each task is characterized by the 6-tuple

$$T = \langle taskId, orderId, stepNumber, releaseDate, deliveryDate, cip \rangle \,. \quad (5)$$

where $taskId$ corresponds to a task identifier, the value $stepNumber$ corresponds to the production step, $releaseDate$ and $deliveryDate$ provide the manufacturing range, and the variable $cip \in \{0, 1\}$ is used to indicate if it is a CIP task ($cip = 1$) or a production task ($cip = 0$).

Let $\mathbb{T} = \{T_1, , \cdots, T_{N_t}\}$ be the task model that includes the list of all the N_t tasks to be scheduled. This list is built combining the set of customers orders $\mathbb{O}$, the set of product families $\mathbb{F}$, the unit model $\mathbb{U}$ and the machine

Algorithm 1: Model task $\mathbb{T}$

input : The set of customers orders $\mathbb{O}$, the set of product families $\mathbb{F}$, the unit model $\mathbb{U}$ and the machine cleaning model $\mathbb{C}$

output: The task model $\mathbb{T}$

set $i = 1$ and $j = 1$;

`// Task model corresponding to production activities`

while $prod_j \in \mathbb{O}! = \emptyset$ **do**

  choose in $\mathbb{F}$ the family of $prod_j$ and set $pfId_j = family\,(prod_i)$;

  choose the subset $\mathbb{U}_s \in \mathbb{U}$ with the same pf but different $pStage$;

  set $n_s = max\,(\mathbb{U}_s)$;

  for $k \leftarrow 1$ **to** n_s **do**

    write a task identificator named $taskId$;

    set $reliasDate = tr_j - di$;

    set $deliveryDate = td_j - di$;

    set $T_i = \{taskId, orderId_j, k, reliaseDate, deliveryDate, 0\}$;

    set $i = i + 1$;

  end

  set $j = j + 1$;

end

`// Task model corresponding to CIP activities`

set j=1;

while $machine_j \in \mathbb{C}! = \emptyset$ **do**

  set $ncip_j = ceil\,((df - di)\,/cipPeriod_j)$;

  for $k \leftarrow 1$ **to** $ncip_j$ **do**

    write a task identificator named $taskId$;

    write a CIP task codification named pd;

    set $releaseDate = k \times cipPeriod_j$;

    set $T_i = \{taskId, pd, 1, releaseDate, releaseDate, 1\}$;

    set $i = i + 1$;

  end

  set $j = j + 1$;

end

cleaning model $\mathbb{C}$. The pseudocode of the algorithm for obtaining the task model $\mathbb{T}$ is presented in *Algorithm* 1, where *di* is a given scheduling initial time.

On the other hand, the machine model ($\mathbb{M}$) is a list of N_m elements with all the resources or machines needed to perform each task, $\mathbb{M} = \{M_1, \cdots, M_{N_m}\}$. Each machine is characterized by the 4-tuple

$$M = \langle taskId, machineId, duration, cost \rangle , \tag{6}$$

where $machineId$ identifies the machine used to carry out each task, *duration* is the time needed to perform the task which is computed using the information presented in Table 2 as in (7) and *cost* is the cost related to the machinery use as in (8) where w_i the cost associated to the machine operation.

$$duration = \begin{cases} (100 \times m/pConc)/(1000 \times pFlow) & \text{if } pFlow \neq \emptyset \\ pTime & \text{if } pFlow = \emptyset \end{cases} , \tag{7}$$

$$cost = duration \times w_i, \tag{8}$$

The machine model is built using *Algorithm* 2.

Both models, $\mathbb{T}$ and $\mathbb{M}$, directly provide the information needed for the constraint programming/optimization solver.

4.2. Constraint Programming/Optimization Approach

As discussed in Section 3.2, the goal is to determine the optimal scheduling of customer orders with the aim of minimizing energy and cleaning costs while maximizing the production taking into account the task and machine models, $\mathbb{T}$ and $\mathbb{M}$ respectively, by using the Optimization Programming Language (OPL) (IBM ILOG, 2015).

The OPL model developed has the following components: data declaration, decision variables declaration, objective function, constraints and script statement.

The declaration of the data allows the user to easily name them in the model. In the developed OPL model three data elements are used: *machines* is a bounded integer (9a) (we have six machines according to Figure 1) and $\mathbb{T}$ and $\mathbb{M}$ are data structures constructed from the tuples (9b) and (9c), respectively.

Algorithm 2: Machine model $\mathbb{M}$

input : The set of customers orders $\mathbb{O}$, the set of product families $\mathbb{F}$, the unit model $\mathbb{U}$ and the machine cleaning model $\mathbb{C}$

output: The machine model $\mathbb{M}$

set $i = 1$ and $j = 1$;

`// Machine model corresponding production activities`

while $prod_j \in \mathbb{O}! = \emptyset$ **do**

  choose in $\mathbb{F}$ the family of $prod_j$ and set $pfId_j = family\,(prod_i)$;

  choose the subset $\mathbb{U}_m \in \mathbb{U}|pfId_j == pf$ and set $n_m = max\,(\mathbb{U}_m)$;

  for $k \leftarrow 1$ **to** n_m **do**

    write a task identificator named $taskId$;

    write a machine identificator named $machineId$;

    compute $duration$ using equation (7);

    compute $cost$ usign equation (8);

    set $M_i = \{taskId, machineId, duration, cost\}$;

    set $i = i + 1$;

  end

  set $j = j + 1$;

end

`// Machine model corresponding to CIP activities`

set j=1;

while $machine_j \in \mathbb{C}! = \emptyset$ **do**

  for $k \leftarrow 1$ **to** $ncip_j$ **do**

    write a task identificator named $taskId$;

    write a CIP task identificator named $cipId$;

    set $duration = cipTime_j$;

    set $cost = cipCost_j$;

    set $M_i = \{taskId, cipId, duration, cost\}$;

    set $i = i + 1$;

  end

  set $j = j + 1$;

end

```
range machines = 1..6;                                   (9a)
tuple T{                                                 (9b)
 key string taskId;
 key int orderId;
 key int stepNumber;
 int releaseDate;
 int deliveryDate;
 int cip;
};
tuple M{                                                 (9c)
 string taskId;
 int machineId;
 int duration;
 int cost;
};
```

In the constructions, of the OPL model for solving the scheduling problem three decision variables (dvar in OPL language) have been considered: *tasks*, *modes* and *mchs*, the first two are defined as interval decision variables, `dvar interval`, and the last as `dvar sequence`

```
dvar interval tasks[t in T] in 0..10000000;
dvar interval modes[md in M] optional size md.duration;
dvar sequence mchs[m in machines]
    in all (md in M : md.machineId == m) modes[md];
```

An interval decision variable represents an unknown of a scheduling problem, in particular an interval of time where an activity is carried out. An interval sequence decision variable is defined on a set of interval variables where the value of an interval sequence variable represents a total ordering of the interval and that are used to represent the batch schedule Laborie (2009).

From the problem statement and task/machine model, the following constraint programming/optimization model is formulated in the following using OPL language (see Appendix for details about the meaning of the keywords).

$$\text{minimize}(\alpha_1 J_1 + \alpha_2 J_2 + \alpha_3 J_3)$$

$$J_1 = \max(\texttt{endOf}(task\,[t_j])), \forall(t_j \in \mathbb{T}), \quad t_j.cip = 0 \tag{11a}$$

$$J_2 = (\sum \texttt{presenceOf}(modes\,[md]) * md.cost), \forall(md \in \mathbb{M}) \tag{11b}$$

$$J_3 = \sum(\underline{\varepsilon}_j^2 + \overline{\varepsilon}_j^2), \forall(t_j \in \mathbb{T}), \quad t_j.cip = 1 \tag{11c}$$

subject to

$$\texttt{startAtstart}(task\,[t_j], task\,[t_i], dt), \quad \forall t_j, t_i \in \mathbb{T},$$
$$t_j.stepNumber + 1 = t_i.stepNumber, \quad t_j.orderId = t_i.orderId \tag{11d}$$

$$\texttt{alternative}(task\,[t], (modes\,[md], \forall md \in \mathbb{M}, md.taskId = t.taskId),$$
$$\forall(t \in \mathbb{T}) \tag{11e}$$

$$\texttt{noOverlap}(mchs[m]) \quad \forall m \in machine \tag{11f}$$

$$t_j.start \geq t_j.releaseDate \quad \forall t_j \in \mathbb{T}, \quad t_j.cip = 0 \tag{11g}$$

$$t_j.end \leq t_j.deliveryDate \quad \forall t_j \in \mathbb{T}, \quad t_j.cip = 0 \tag{11h}$$

$$t_j.start \geq (t_j.releaseDate - \underline{\varepsilon}_j) \quad \forall t_j \in \mathbb{T}, \quad t_j.cip = 1 \tag{11i}$$

$$t_j.end \leq (t_j.deliveryDate + \overline{\varepsilon}_j) \quad \forall t_j \in \mathbb{T}, \quad t_j.cip = 1 \tag{11j}$$

The optimization criteria include three objectives:

- J_1 defined in (11a) aims to minimize the production span. The function $\texttt{endOf}(task\,[t])$ returns the end time of a given interval variable $task[t]$.
- J_2 defined in (11b) computes the cost of resource md required by each processing task. The expression `presenceOf` is true if the interval variable $modes\,[md]$ is present in the solution, either is false.
- J_3 defined in (11c) aims to mimimize the relaxation in the starting ($\overline{\varepsilon}$) and ending time ($\underline{\varepsilon}$) of the CIP tasks according to the rules established by each machine.

Weight tuning ($\alpha_i, i = 1, \cdots, 3$) allows to prioritize and establish the trade-off between the different objectives (Deb and Deb, 2014). In this paper, the tuning procedure presented in Logist et al. (2011) based on computing

the Pareto front is used. This procedure is based on defining the desired management point and determine the weights that find the solution of the Pareto front closest to the management point.

On the other hand, the constraints describe the production rules to be fulfilled. Following the nomenclature presented in Zeballos et al. (2011), t_j models the processing of task j described in $\mathbb{T}$ and md_i models the product process i described in $\mathbb{M}$. The problem has been solved considering seven kinds of constraints that are obtained from the production rules:

- Constraint (11d) defines the production sequence according to the type of product. As our process is semi-continuous, it is not necessary that one task finishes before starting the next one. The subsequent task can start with a given time delay dt, which, in our case, has been considered of 1 hour. This constraint is enforced with `startAtstart` constraint.

- In the studied process, it is possible to produce one product using alternative machines. The allowed alternatives are formulated in (11e) using `alternative` constraint. The process has six machines which can work simultaneously but only on one task at a time. This is avoided thanks to the `noOverlap` constraint (11f).

- Constraints (11g) and (11h) place the lower and the higher bounds regarding when a specific task should be performed. The first one does not allow any task to start before the time assigned (the creation date of the consumer order). The second constraint enforces the end of processing activity (the delivery date agreed with the consumer).

- Constraints (11i) and (11j) allow starting and ending the CIP activity in the interval of $[\underline{\varepsilon}, \overline{\varepsilon}]$ hours around the predetermined time according the CIP rules of each machine. The relaxation values ($\underline{\varepsilon}$ and $\overline{\varepsilon}$) are introduced to allow a small degree of freedom when scheduling the production tasks allowing to initiate the CIP tasks a bit later or earlier of the pre-established regular cleaning times according the production needs.

5. Results

In this section, we present the results obtained in real production planning scenarios in the considered milk/yogurt powder production plant described

in Section 3.1. The production time horizon considered in this example was an uninterrupted period of 15 full days where demand quantities and due dates are based on real sequences of product orders.

To illustrate how the proposed approach facilitates the adaptation of the optimization objectives to the practical situations (as discussed in the introduction), the effect of changing the electrical and thermal consumption cost of the units operating in parallel (evaporators ED1 and ED2, see Figure 1) will be presented.

Using real data obtained from the real plant, the electrical and thermal consumption costs of each evaporator has been determined. Table 4 shows the mean values of the energy and water consumptions of the evaporators in production and in CIP operation. Note that the consumption of electrical energy is bigger in ED1 while the consumption of thermal energy is bigger in ED2. Thermal energy is obtained from steam which is produced by a steam generator based on a gas boiler. However, since the price of gas and electricity can vary with time according the market, three cases are considered according the relative costs of both evaporators

- Scenario 1: Both evaporators present the same cost ($w_1 = w_2$)
- Scenario 2: The cost of ED1 is larger than the cost of ED2 ($w_1 > w_2$)
- Scenario 3: The cost of ED2 is larger than the cost of ED1 ($w_1 < w_2$)

This allows to assess that the optimization of the production scheduling using the proposed approach is able to adapt according the energy prices. In all these scenarios, the weights for objectives (11a)-(11c) are respectively $\alpha_1 = 10$, $\alpha_2 = 1$ and $\alpha_3 = 1$, i.e., the minimization of the time span of the production is prioritized with respect to the energy costs and relaxation of the starting/ending of the CIP tasks. As already discussed in Section 4, the tuning of these weights has been done using the procedure proposed in Logist et al. (2011) by establishing the desired trade-off among the considered objectives.

Figures 2, 3 and 4 show machine task assignment of the resulting optimal scheduling corresponding to the Scenarios 1, 2 and 3, respectively. In these figures, the machine used by each task, the starting time (start), the ending time (end) as well as the processing time are presented. Colors blue and green are used in alternative manner to better distinguish to consecutive tasks. Note that when both evaporators have the same cost (Scenario 1), they are

Table 4: Energy and water consumption of ED1 and ED2 in both operation states: production and CIP.

	Electrical energy		Thermal energy		Water consumption
	Prod. [kWh]	CIP [kWh]	Prod. [kWh]	CIP [kWh]	CIP [to/h]
ED1	91,2	106,7	345,6	551,5	4,2
ED2	72,7	81,5	2063,2	1274,0	8,9

used in balanced way, each one processing 45 tasks. However, when the cost of one of them is larger than the other (Scenario 2 or 3), the most used is the cheaper one, as expected. Figures 3 and 4 show that in Scenario 2 ED1 performs 27 tasks and ED2 performs 63 tasks while in Scenario 3, the results are reversed. The remaining machines are distributed in the scheduling time as needed to produce all the customer orders giving in the three scenarios the same distribution.

Table 5 presents a part (because of space limitations) of the optimal solution obtained in the three considered scenarios. This table shows how the orders presented in Table 1 have been scheduled sorted by the variable *modes#*. Note that the orders that belong SSP or SVP family only have two stages: the first one can be assigned to the machine 1 (ED1) or 2 (ED2), depending of the scenario, while the second one can only be done in machine 3 (TW2). In contrast, the orders that belong to the yogurt family have four stages. The first one can be assigned to the machine 1 (ED1) or 2 (ED2), depending of the scenario, the second and the third ones to be machines 5 (Pasteurization) and 6 (Fermentation) respectively, while the last one to machines 3 (TW2) or 4 (TW1), depending on occupation of each one of the machines. $deliveryDate$, $start$ and $stop$ are relative time in hours with respect the scheduling initial time, in our case it is one o'clock of May, 1st. Each task is associated to a $mode\#i$ where $i = 0 \cdots N_t$ which is used to identify the task in the scheduling graph. Note that in all these results $end \leq deliveryDate$ is satisfied.

Figure 5 shows the accumulated electrical consumption of ED1 and ED2 in normal operation in the 3 scenarios. These results are related with the evaporator activities presented in Figures 2, 3 and 4. As expected, the consumption is related with the hours of usage of each evaporator. A more interesting analysis is the electrical and thermal power consumption of the evaporators both in normal operation and CIP phase (see Figure 6). Note that in Scenario 2 the electrical energy consumption decreases about 7.25%

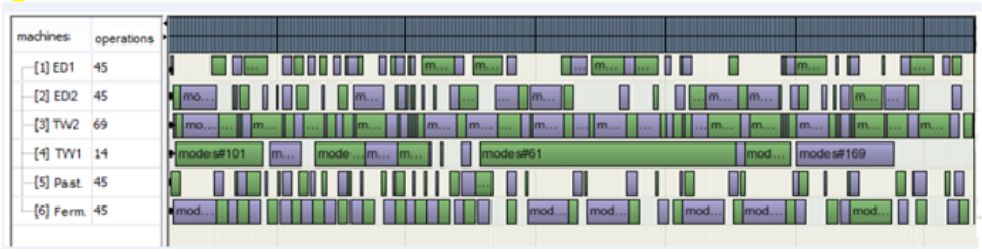

Figure 2: Optimal scheduling for Scenario 1 ($w_1 = w_2$).

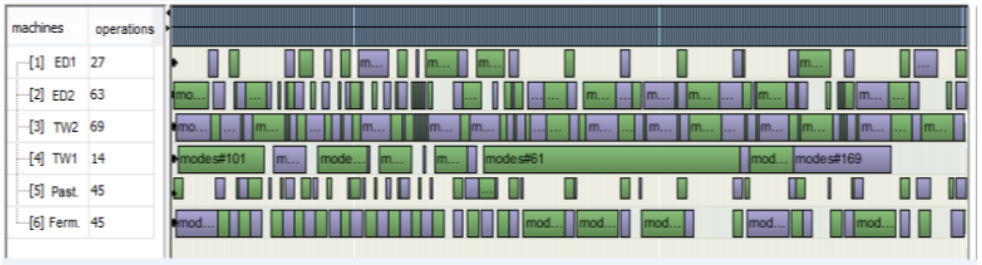

Figure 3: Optimal scheduling for Scenario 2 ($w_1 \gg w_2$).

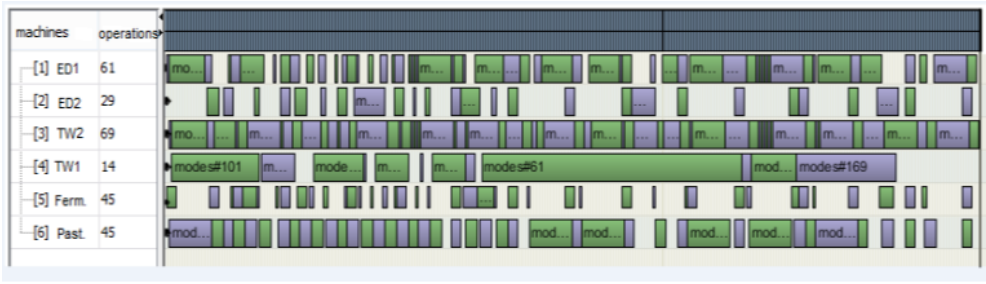

Figure 4: Optimal scheduling for Scenario 3 ($w_1 \ll w_2$).

Table 5: Scheduling comparison in the three Scenarios of the orders provided in Table 1.

Order					Scenario 1			Scenario 2			Scenario 3		
taskId	Name *modes#i*	Product	*release Date* [h]	*delivery Date* [h]	*machineId*	*start* [h]	*end* [h]	*machineId*	*start* [h]	*end* [h]	*machineId*	*start* [h]	*end* [h]
709365_1	#1	SSP	0	216	1	199	209	2	159	169	2	196	206
709365_2	#2	SSP	0	216	3	200	210	3	160	170	3	197	207
714985_1	#18	YOGURT	0	264	2	207	209	2	137	139	2	137	139
714985_2	#21	YOGURT	0	264	5	208	209	5	138	139	5	138	139
714985_3	#22	YOGURT	0	264	6	209	214	6	139	144	6	139	144
714985_4	#20	YOGURT	0	264	3	210	213	3	140	143	3	140	143
724732_1	#106	SSP	0	216	1	212	215	2	49	52	1	155	158
724732_2	#107	SSP	0	216	3	213	216	3	50	53	3	156	159
723164_1	#76	YOGURT	0	144	1	59	62	2	59	65	1	59	62
723164_2	#79	YOGURT	0	144	5	60	62	5	60	62	5	60	62
723164_3	#80	YOGURT	0	144	6	61	63	6	61	66	6	61	66
723164_4	#77	YOGURT	0	144	4	63	66	4	62	83	4	62	83
731127_1	#205	SVP	120	312	2	293	306	2	250	263	1	158	171
731127_2	#204	SVP	120	312	3	294	307	3	251	264	3	159	172

and the thermal energy consumption increases by 21% while on Scenario 3 the electrical energy consumption increases by 3.5% and the thermal energy consumption decreases by 35.5%.

6. Conclusions

In this the paper, the optimal batch scheduling of a milk/yogurt powder production plant is addressed by means of a combined constraint programming/optimization approach. The motivation for using a constraint programming approach lies in the fact that it dissociates the representation of the problem from the search algorithms used to solve it. First, the process and production rules are modeled using a set of constraints. To this aim and to facilitate the creation of constraints, two algorithms have been developed that allow to generate the constraints associated to the task and machine models. Then, an objective function that includes the process optimization criteria (a trade-off between time span and economic cost) is defined. The constraint programming/optimization problem has been formulated and solved using the IBM ILOG CPLEX Optimization Studio. A set of scenarios based on a set real orders are presented to illustrate the validity and performance of the proposed approach. As a future work, the proposed solution will be integrated in the production management system of the dairy plant to be tested in real operation.

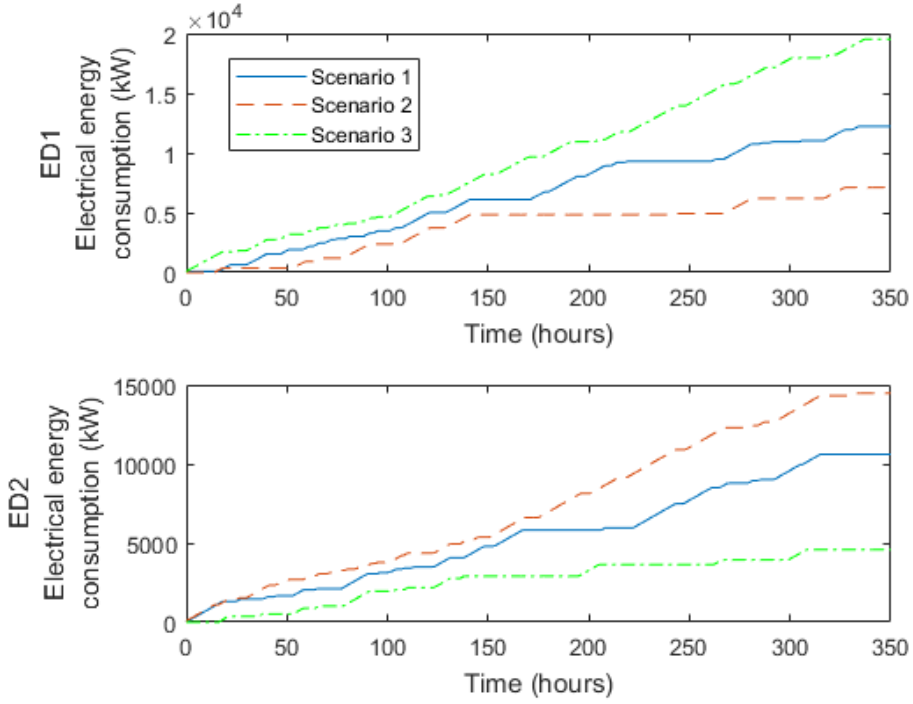

Figure 5: ED1 and ED2 electrical energy consumption in the three scenarios.

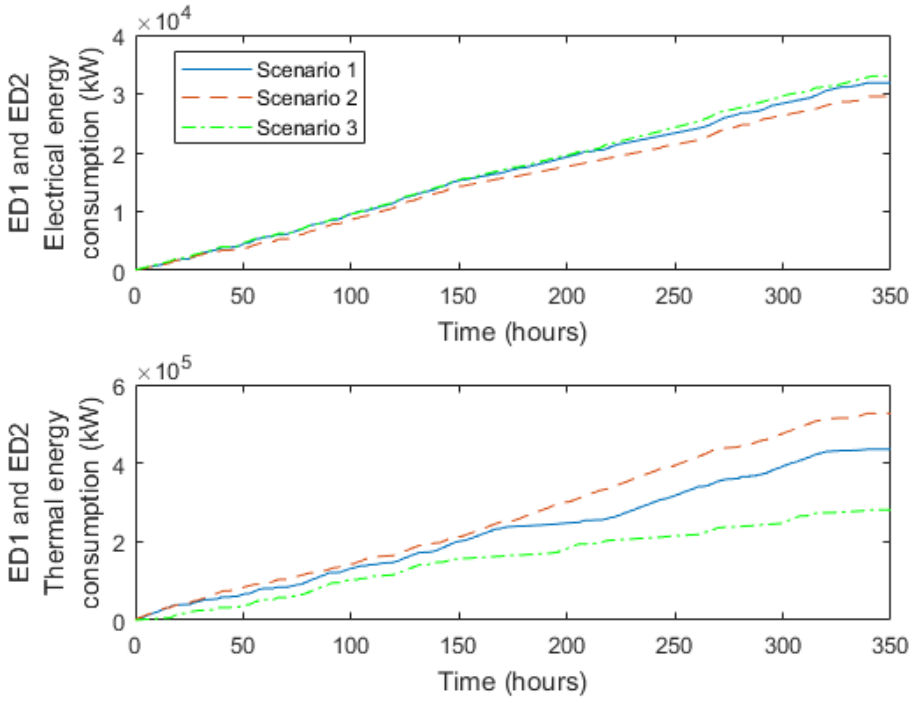

Figure 6: Electrical and themal energy consumption of evaporators in the three scenarios.

Appendix A. Notation list

Variables	Description
$orderId$	Order identifier
$prod$	ordered product
m	ordered volume
tr	release data
td	due data
pf	product family
$pCode$	product family identification
act	raw material transformation
$machine$	machines
$pTime$	processing time
$pStage$	a given operation
$machineId$	machine identifier
$wCost$	machine cost
$pFlow$	processing flow
$pConc$	naterial input concentration
$cipPeriod$	cleaning period
$cipTime$	cleaning duration time
$cipCost$	cleaning cost
$taskId$	Task identifier
$stepNumber$	Production step
$releaseDate$	Numerical indicator of the order release date relative to initial shqueduling
$deliveryDate$	Numerical indicator of the order dur date relative to initial shqueduling
cip	Boolean cleaning machine indicator
$machineId$	Numerical machinery identificator
$duration$	Time needed to perform a task
$cost$	The cost of the machinery use to perform one task

Sets	Description
$\mathbb{O}$	List of the orders to be sheduledr
$\mathbb{F}$	List of family products
$\mathbb{U}$	List of the activities needed to perform the products
$\mathbb{C}$	List of the machines cleaning operations
$\mathbb{T}$	List of all the task to be scheduled
$\mathbb{M}$	List of all the Machines needed to perform a task

Appendix B. Overview of IBM ILOG CPLEX Commands

Command	Description
`minimize` staticLex (dexpr, dexpr)	Used by CP Optimizer in models with a multi-criteria objective
`max`	The maximal value from a list
dexpr int `endOf`	Returns the end of an interval (scheduling)
boolean `presenceOf`(dvar interval)	Specifies that an interval must be present (scheduling)
`startAtstart`	Constraint on start times of intervals (scheduling)
`alternative`	Create an exclusive alternative amongst intervals (scheduling)
`noOverlap`	Restricts intervals from overlapping in a sequence (scheduling)

Acknowledgement

This work has been supported by the European Union's Seventh Framework Programme under grant agreement EnReMilk-613968 and partly supported by the Spanish MINECO ad FEDER research project SCAV (ref. DPI2017-88403-R). The authors specially thank the support received from Schwarzwaldmilch GmbH Freiburg in the case study described in this paper.

References

Bilgen, B., Dogan, K., 2015. Multistage production planning in the dairy industry: A mixed-integer programming approach. Industrial & Engineering Chemistry Research 54 (46), 11709–11719.

Ceberio, M., Kreinovich, V., 2017. Constraint Programming and Decision Making: Theory and Applications. Springer.

Deb, K., Deb, K., 2014. Multi-objective Optimization. Springer US, Boston, MA, pp. 403–449.

Deka, D., Datta, D., 2017. Multi-objective optimization of the scheduling of a heat exchanger network under milk fouling. Knowledge-Based Systems.

Doganis, P., Sarimveis, H., 2007. Optimal scheduling in a yogurt production line based on mixed integer linear programming. Journal of Food Engineering 80 (2), 445–453.

Ham, A., Fowler, J. W., Cakici, E., Nov 2017. Constraint programming approach for scheduling jobs with release times, non-identical sizes, and incompatible families on parallel batching machines. IEEE Transactions on Semiconductor Manufacturing 30 (4), 500–507.

Ham, A. M., Cakici, E., 2016. Flexible job shop scheduling problem with parallel batch processing machines: MIP and CP approaches. Computers and Industrial Engineering 102, 160 – 165.

Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., Hooker, J., Méndez, C., Sand, G., Wassick, J., 2014. Scope for industrial applications of production scheduling models and solution methods. Computers & Chemical Engineering 62, 161–193.

Hazaras, M. J., Swartz, C. L., Marlin, T. E., 2013. Industrial application of a continuous-time scheduling framework for process analysis and improvement. Industrial & Engineering Chemistry Research 53 (1), 259–273.

IBM ILOG, t. C. O. S. V. d., 2015.
URL `https://www.ibm.com/support/knowledgecenter/en/SSSA5P_12.6.2/ilog.odms.studio.help/Optimization_Studio/topics/COS_home.html`

Kopanos, G. M., Puigjaner, L., Georgiadis, M. C., 2011. Resource-constrained production planning in semicontinuous food industries. Computers & chemical engineering 35 (12), 2929–2944.

Ku, W.-Y., Beck, J. C., 2016. Mixed integer programming models for job shop scheduling: A computational analysis. Computers and Operations Research 73, 165 – 173.

Laborie, P., 2009. IBM ILOG CP optimizer for detailed scheduling illustrated on three problems. In: van Hoeve, W.-J., Hooker, J. N. (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 148–162.

Logist, F., Vallerio, M., Impe, J. V., 2011. Explicit weight selection procedure for optimal control problems with weighted objectives. IFAC Proceedings Volumes 44 (1), 12325 – 12330, 18th IFAC World Congress.

Marriott, K., Stuckey, P. J., 1998. Programming with constraints: an introduction. MIT press.

Novara, F. M., Novas, J. M., Henning, G. P., 2016. A novel constraint programming model for large-scale scheduling problems in multiproduct multistage batch plants: Limited resources and campaign-based operation. Computers & Chemical Engineering 93, 101–117.

Okubo, H., Miyamoto, T., Yoshida, S., Mori, K., Kitamura, S., Izui, Y., 2015. Project scheduling under partially renewable resources and resource consumption during setup operations. Computers & Industrial Engineering 83, 91–99.

Pearce, K., 2000. Milk powder. New Zealand Dairy Research Institute, Palmerston North.

Sel, Ç., Bilgen, B., 2015. Quantitative models for supply chain management within dairy industry: a review and discussion. European Journal of Industrial Engineering 9 (5), 561–594.

Sel, C., Bilgen, B., Bloemhof-Ruwaard, J., van der Vorst, J. G., 2015. Multi-bucket optimization for integrated planning and scheduling in the perishable dairy supply chain. Computers & chemical engineering 77, 59–73.

Silvente, J., Kopanos, G. M., Pistikopoulos, E. N., Espuña, A., 2014. Reactive scheduling for the coordination of energy supply and demand management in microgrids. In: Computer Aided Chemical Engineering. Vol. 33. Elsevier, pp. 493–498.

Van Hentenryck, P., 1999. The OPL optimization programming language. MIT Press.

Wari, E., Zhu, W., 2016a. Multi-week MILP scheduling for an ice cream processing facility. Computers & Chemical Engineering 94, 141–156.

Wari, E., Zhu, W., 2016b. A survey on metaheuristics for optimization in food manufacturing industry. Applied Soft Computing 46, 328–343.

Zeballos, L. J., Novas, J. M., Henning, G. P., 2011. A CP formulation for scheduling multiproduct multistage batch plants. Computers & Chemical Engineering 35 (12), 2973–2989.