Optimal Schedule of a Dairy Manufactory

R. Adonyi,[a] E. Shopova,[b,*] and N. Vaklieva-Bancheva[b]
[a]Department of Computer Science, University of Pannonia,
Veszprem, Egyetem u. 10, H-8200, Hungary
[b]Bulgarian Academy of Sciences, Institute of Chemical Engineering,
"Acad. G. Bonchev" Str., Bl.103, Sofia, 1113, Bulgaria

Original scientific paper
Received: November 20, 2007
Accepted: September 24, 2008

This paper deals with short-term scheduling of the dairy industry. Two different approaches are proposed for obtaining the minimal makespan schedules. According to the first, S-graph framework is proposed to find the optimal solution of the flow-shop scheduling problem. The problem is solved by applying the branch and bound technique. The second approach uses the integer programming formulation of the scheduling problem and BASIC genetic algorithm has been used to solve the optimization problem. Both approaches take into consideration volumes of units assigned to perform tasks, and respective size factors that affect the size of batches and their number must be produces to achieve production goals and thus on the schedules duration. Manufacturing of two type curds is used as a case study. The results obtained show that both approaches provide comparable solutions. Both approaches could be seen as a good alternative to project manager to find appropriate schedule of the dairy industry.

Key words:
Scheduling problem, S-graph framework, integer programming formulation, dairy industry

Introduction

Traditionally, the dairy industry is well positioned worldwide. Within this sector, the majority of the companies are small- and medium-size enterprises (SMEs). The market demand for dairy products is increasing constantly. For instance, in Bulgaria, the consumption of curds has almost doubled from 2316 t in 2003 to 4302 t in 2004[**]. This growth requires the dairy processing companies to be more agile and adaptive to market changes. Business process re-engineering programs can help in supporting SMEs for introducing new manufacturing technologies or for establishing novel ways of increasing productivity and profits of the existing industrial plants. In this context, short-term schedules can play an important role in efficiently exploiting the plant capacities and improving flexibility of production and delivery.

Scheduling of batch manufactures becomes an extremely complex problem if a large number of batches must be produced. In such cases, the search space could be reduced drastically by eliminating the redundant and combinatorial infeasible solutions. Different approaches are proposed in literature to deal with the problem. They differ in the chosen scheduling type, process representation – e.g., STN,[1–4] RTN,[5–7] S-graphs,[8,9,11] the time domain continuity (discrete or continuous) and in the focus of problem formulation – mathematical programming (MILP[1–4,7] and MINLP[5,6]) and combinatorial driven.[8,9,11]

However, the studies discussing scheduling problems for dairies are limited in literature. For example, to solve a scheduling and sequencing problem for milk tankers arising at a central base for milk collection, Basnet *et al.* have proposed an exact algorithm aiming to minimize a daily makespan. The scheduling problem is formulated as a linear integer programming (IP) and solved by embedding within a branch and bound approach.[12] The study also provides a comparison of proposed exact algorithm with a heuristic one developed for the same case.[13]

More recently, a mixed integer linear programming (MILP) model is proposed by Doganis and Sarimveis[14] to solve optimal production scheduling in a single yoghurt production line. The formulation takes into consideration the specific restrictions of the yoghurt production. In order to represent more realistically the production cost while achieving given production goals, a production sequence-dependent costs function accounting for labor and inventory costs is used as an optimization criterion. The model produces the complete production schedule for a selected horizon and sequence of products that should be manufactured. Further extension of this research proposes a methodology for optimal scheduling in parallel machines.[15]

Marinelli *et al.*[16] have proposed a solution approach for a capacitated lot sizing and scheduling of parallel machines and shared buffers using as a case

[*] All correspondence should be addressed to E. Shopova
E-mail: eshopova@mail.bg

[**] According to the Ministry of Agriculture and Forestry (2005).

study a real problem arising in a packaging company, also, for yoghurt production. Their main idea is to formulate the problem through a hybrid CSLP–CLSP model in order to match different requirements of lot sizing and scheduling in the buffering and processing stage. The goal is to minimize a function of the relevant costs (processing, set-up, holding, etc.).

The studies mentioned above solve different scheduling problems for dairy products accepting respective technological lines as single or parallel machines. However, dairies, as the most multipurpose batch plants, have a flexible structure. They employ sets of batch equipment units of different types (such as milk-separators, pasteurizers, vat-reactors, drainers, cream-ripening, butter-churning, packing machines etc.) with different volumes suited to manufacture various sets of products. Plants allow each processing task of a product recipe to be carried out in different subsets of the suitable equipment units that leads to existence of multiple potential production routes for product manufacturing. Depending on the volumes of equipment units assigned to the tasks and tasks size factors these production routes result in different batch sizes and consequently in different number of batches to be produced to achieve given production demands. If a group of dairy products have to be produced compatibly in a time horizon using existing equipment units, the most important question for dairy managers is which production routes must be chosen and how to schedule the products in order to fulfill given production demand with minimal makespan.

Taking into account the importance of the problem for a dairy manufacture, the goal of this paper is to present two independent approaches for minimal makespan schedules creation. The first one is based on the S-graph framework representation and solution is obtained applying branch and bound technique. The second approach uses an integer programming formulation to model the scheduling problem and a genetic algorithm is applied for its solution.

Motivating example

The study has been inspired by the need of one small dairy to organize efficiently a compatible manufacturing of two types of curd – low fat (w = 0.3 %) (further called product P1) and high fat (w = 1 %) (named product P2), so as their demands be fulfilled in the shortest term possible.

The technologies for both products are similar. They comprise 3 main tasks (Fig. 1) where: task 1 is the pasteurization of skimmed milk carried out by pasteurizer units; task 2 is the milk acidification and curds by-product processing that has to be performed by vat-reactors; and task 3 is draining of the remaining whey and target products processing, which is performed by drainers.

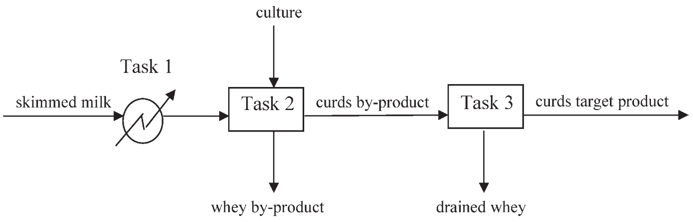

Fig. 1 – *Production technology*

For product P1, skimmed milk with w = 0.076 % milk fat is used as raw material, which leads to the following values of task size factors: $v_{1,1} = 3.506$; $v_{1,2} = 3.984$; $v_{1,3} = 1.1$ dm^3 kg^{-1}, while for product P2 the milk used has w = 0.234 % fat fraction and results to next size factors: $v_{2,1} = 4.351$; $v_{2,2} = 4.944$; $v_{2,3} = 1.1$ dm^3 kg^{-1}.

Processing times for both products are equal: task 1 – 30 min; task 2 – 240 min and task 3 – 30 min.

The production demands for both products are equal $m_{Q_1} = 1400$ kg, $m_{Q_2} = 1400$ kg.

The suitable units of the diary to perform tasks and their volumes are given in Table 1.

Table 1 – *Plant data*

Type	Pasteurizers				Vat reactors			Drainers			
No	1	2	3	4	5	6	7	8	9	10	11
V/dm^3	300	250	150	100	300	400	250	80	60	60	100

A rough estimation for a number of feasible production routes for each product shows that in case of one unit assignment to each task, the number of feasible production routes is 80. If the model is more complicated by assigning different number of the available units (see Fig. 2) the production routes could rise up to 1372.

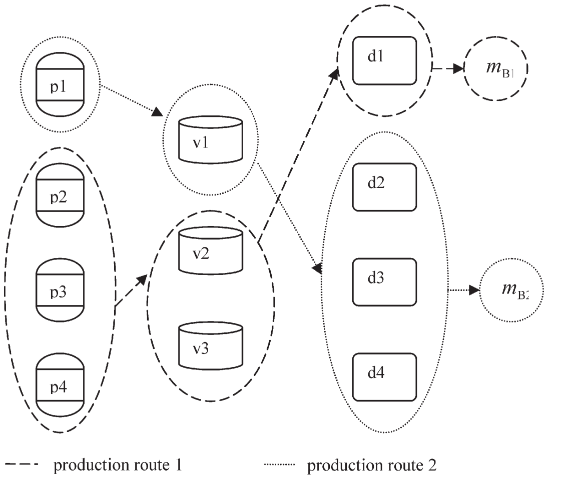

Fig. 2 – *Two feasible production routes for curds manufacturing – p – pasteurizers, v – vat reactors, d – drainers; m*$_B$ *– respective batch sizes*

The estimation above illustrates that even such a simple process as curd production would cost efforts of the plant manager to determine the most appropriate combination of the production routes needed to achieve the set production goals at the minimal makespan.

Problem statement and assumption

In the considered dairy industry problem, set P denotes the available equipment units of different types with volumes V_p, where $p \in P$.

Set I denotes the milk products to be produced in the plant and respective demands are mentioned by m_{Q_i}, $\forall i, i \in I$.

Sets L_i present the tasks that must be carried out to manufacture products i, for $i \in I$. The processing times and size factors of tasks $l \in N_{L_i}$ are denoted by $t_{i,l}$ and $v_{i,l}$, respectively ($\forall l$, $l \in N_{L_i}$, $\forall i$, $i \in I$).

The information concerning the equipment units suitable to perform each task of each product is known *a priori*.

Additionally following assumptions are posed:

1. Processing times are constant and do not depend on the batch size;

2. Change-over times are neglected.

3. Cleaning times also not taken into account.

The S-graph framework

The S-graph framework was introduced to represent and efficiently solve production scheduling problems. This work is based on the S-graph representation[11] and a corresponding general framework,[9] and the acceleration tools.[8]

The general S-graph framework

A batch process scheduling problem is defined by the recipe, the amount to be produced from each product, and the plausible tasks to equipment units assignments. The aim is to give a schedule together with a task-equipment unit assignment so that we achieve minimal makespan.

The S-graph framework is a graph representation and algorithm designed for the NIS case (Non Intermediate Storage), and is able to solve industrial size scheduling problems. The S-graph can be represented by directed graph $G(N, A_1, A_2)$, where N is the set of nodes, sets A_1, A_2 contain the recipe-arcs and the schedule-arcs, respectively. The nodes of the S-graph represent the tasks (task node) and the products (product node) of the recipe; and its arcs represent the precedence relationships among the tasks (recipe-arc) and the order of application of the equipment units (schedule-arc). Since the arcs of an S-graph represent precedence in time, an S-graph that represents a recipe or a schedule of a problem, always acyclic. There are two specific S-graphs, the recipe-graph for the recipe and the schedule-graph for the solution.

The recipe-graph is based on the recipe. It describes the inputs of the scheduling problem. It gives the order of the tasks, the material transfers among them, and the set of plausible units for each task. Fig. 3 illustrates the recipe-graph of product A and B. The task nodes are labeled by 1 through 8, and the product nodes by 9 and 10. Sets S1 through S8 contain the equipment units that can be assigned to the corresponding task of the recipe.

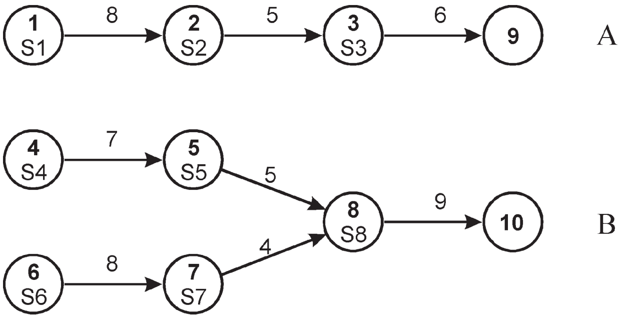

Fig. 3 – *Recipe-graph for two products*

The schedule-graph describes a single solution of the scheduling problem, i.e., all tasks represented in the recipe-graph have been scheduled by taking equipment-task assignment into account. The S-graph framework can handle the NIS policy effectively. According to the NIS case an equipment unit is not applicable for its next task until the intermediate material of its current task has been transferred to another equipment unit. This constraint on the equipment units can be expressed by the way the schedule-arcs are added to the S-graph. The S-graph representation ensures that the intermediate materials of a schedule are always stored in the corresponding equipment unit. If equipment unit E1 is assigned to task 6 and consecutively to task, then, a schedule-arc is established from all the consecutive tasks of task 6 to task 1 as shown in Fig. 4.

Because of the combinatorial characteristics of scheduling, a branch and bound procedure may be useful for generating the optimal schedule of a scheduling problem. The recipe-graph with no equipment unit assignment serves as the root of the enumeration tree of the B&B procedure. At any partial problem, one equipment unit is selected and then all child partial problems are generated through the possible assignments of this equipment unit to unscheduled nodes.[9]

The bounding procedure tests the feasibility of a partial problem. If this test is positive, it determines the lower bound for the makespan of all solutions that can be derived from this partial problem simply by using the well-known longest path algorithm, but in some cases better lower bound can be derived with using a linear programming model.

Fig. 4 illustrates the optimal schedule in the form of a schedule-graph. The inserted schedule-arcs represent the optimal schedule. The makespan, determined by the longest path algorithm, is 30.

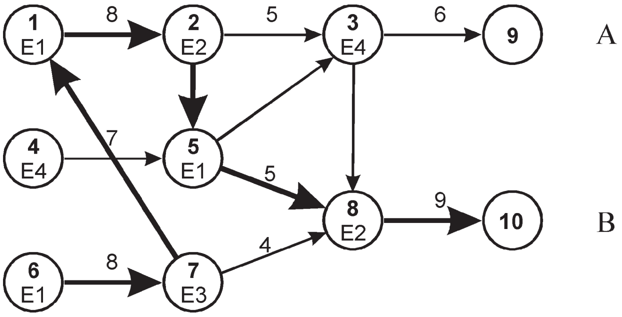

Fig. 4 – *Schedule-graph of the optimal solution. Bold line shows the longest path.*

Flow shop scheduling with the S-graph framework

In the present work the S-graph framework is extended for the flow shop scheduling dairy production problems. The amounts of products are calculated by introducing the volume for the equipment units and the size factors for the tasks. Let the equipment unit p is assigned to task ts of S-graph G, where $ts \in N$, and V_p and v_{ts} represent the volume and size factor of equipment unit p and task-ts. The size of the task ts is determined by $m_{\mathrm{BT}_{ts}} = V_p/v_{ts}$. The size of batch b is determined by

$$\min_{ts \in B_b} \{ m_{\mathrm{BT}_{ts}} \},$$

where set B_b contains the task nodes of batch b. If more than one equipment units are scheduled for a task, the volumes of these equipment units are added. Moreover, the S-graph framework is extended to schedule the equipment units to the products and not to the tasks, as the flow shop scheduling problems require.

To generate such a minimal makespan schedule that fulfils the product requirement, the bounding procedure is extended. The original bounding procedure checks the feasibility of the schedule, if the feasibility is true the longest path algorithm or a linear programming model alternatively gives the lower bound of the makespan. The bounding procedure of the S-graph framework for the dairy production problem is extended with calculation of the feasibility of batch sizes. If the amount of products of the partial problem will be less than the required amounts then the feasibility of the partial problem fails.

Integer programming formulation

According to the integer programming formulation a mathematical model is used to describe scheduling problem. The goal is to find such production routes for all products that ensure compatible manufacturing of products in a common campaign and result in minimal makespan schedules.

However to ensure product processing in a common campaign the number of units of each type must be equal or greater than the number of tasks that must be carried out with these type of units.

Additionally, it is supposed that the products are manufactured with cycle overlapping. The cycle time, denoted by T_i, is limited by the longest processing time, i.e., $T_i = \max_{l \in L_i} \{ t_{i,l} \}$, $\forall i,\ i \in I$.

Information for units suitable to process each task of each product is given by binary matrices named $\boldsymbol{UID(i)}_{l,p}$, where:

$$\boldsymbol{UID(i)}_{l,p} = \begin{cases} 1- \text{ if unit } p \text{ is suitable to perform} \\ \quad\ \text{task } l \text{ of product } i; \\ 0- \text{ otherwise} \end{cases}$$

Mathematical description

Variables

To identify the different production routes for the products, a set ζ of binary variables is introduced. Variables $\zeta_{p,i}$, $i \in I$, $p \in P$, are used to control this process, as follows:

$$\zeta_{p,i} = \begin{cases} 1- \text{ if unit } p \text{ is used for product } i \\ 0- \text{ otherwise} \end{cases} \qquad (1)$$

Constraints

Feasibility constraints. The production routes are feasible if at least one appropriate equipment unit is assigned to each task. To support searching process the information provided in $\boldsymbol{UID}(i)_{l,p}$ is used and the following constraints are introduced:

$$\prod_{l=1}^{L_i} \left(\sum_{p=1}^{P} \boldsymbol{UID}(i)_{l,p} \zeta_{p,i} \right) \geq 1,\ \forall i,\ i \in I. \qquad (2)$$

Compatibility constraints. Production routes are compatible if there are not products sharing the same equipment units. Fulfillment of this condition ensures by following set of constraints:

$$\sum_{i=1}^{I} \zeta_{p,i} \le 1, \ \forall p, \ p \in P. \tag{3}$$

Batch sizes calculations. When the production routes are constructed the corresponded batch sizes have to be calculated. The particular batch size of each task depends on the volumes of units assigned for its performance and its size factor. They are calculated by:

$$m_{\mathrm{BT}_{i,l}} = \frac{\sum_p V_p \boldsymbol{UID}(i)_{l,p} \zeta_{p,i}}{v_{i,l}}, \forall l, \ l \in L_i, \forall i, \ i \in I. \tag{4}$$

The batch size for each product is limited by the task with minimal particular batch sizes, i.e.,

$$m_{\mathrm{B}}(\zeta)_i = \min_{l \in L_i} \{m_{\mathrm{BT}_{i,l}}\}, \ \forall i, \ i \in I. \tag{5}$$

Number of batches calculation. Numbers of batches that must be performed to produce the required demands depend on ζ and could be determined by:

$$N_{\mathrm{B}}(\zeta)_i = \left\lfloor \frac{m_{Q_i}}{m_{\mathrm{B}}(\zeta)_i} \right\rfloor, \ \forall i, \ i \in I, \tag{6}$$

where by $\lfloor \ \rfloor$ the greater whole part of the relation in is mentioned.

Objective function

Because of cyclic manufacturing, the times needed to produce the demands for the products following already constructed production routes are calculated by:

$$\mathrm{ST}(\zeta)_i = N_{\mathrm{B}}(\zeta)_i T_i + \left(\sum_{l=1}^{L_i} t_{i,l} - T_i \right), \ \forall i, \ i \in I. \tag{7}$$

The expression in brackets accounts for the queue times, determined from the first and last batches.

Having in mind that all products process compatibly the schedule duration is determined by the product for which the value of eq. (7) is maximum. What follows the optimal schedule is determined by this combination of feasible and compatible production routes – $\zeta_{p,i}$ that leads to:

$$\min_{\zeta} \left[\max_{i \in I} \{\mathrm{ST}(\zeta)_i\} \right]. \tag{8}$$

Expression (8) is used as the objective function for creation of minimal makespan schedules.

BASIC genetic algorithm – a tool for problem solution

To solve the formulated IP scheduling model, the genetic algorithm called BASIC GA[10] is applied. It follows all common steps of the genetic algorithms. The continuous search space [0, 1] and real representation schemes are exploited for both, real and integer variables. It works with a predefined constant size of population. At the first generation, BASIC GA initializes a population of randomly created individuals. Applying morphogenesis functions their phenotypes are determined. Afterwards they are used to calculate the values of the objective function and to determine respective fitness functions.

At the next steps the evolutionary operators take place to create the offspring. Firstly, a biased selection for reproduction is carried out. The algorithm operates with the fitness function values to provide the most prospective samplings for crossover. They gather in a sampling pool. Then, the samplings go to recombination. Randomly chosen individuals form couples, which number is equal to the number of samplings. If, for a given couple, a predefined crossover probability has happened, their chromosomes recombine providing two children, otherwise, parents pass directly in the offspring. Finally, the mutation takes place. Each gene of each offspring's chromosome goes to mutation if a predefined mutation probability has happened. In the last stage, selection for replacement carries out to produce a new population for the next generation. The offspring decode to obtain the respective solutions. Both, the parents' and children's chromosomes collect in a replacement pool. The elite individual, corresponding to the best solution in the pool passes to the new population. Further, selection goes unbiased randomly drawing chromosomes from the pool till next population is completed. At the end of this phase, the number of generation increases.

BASIC uses the generation number as a stop criterion. It checks for the stop criterion fulfillment. If it is met, the obtained best solution is proposed as a problem solution. In the opposite case, the loop is closed through the fitness functions calculation for the new population.

BASIC GA involves a different number of schemes in the genetic operators, which makes it adaptive to various optimization problems. Thus, it includes three schemes for the selection for recombination (roulette wheel, rank based and tournament); five crossover operators (N-points, uniform, arithmetic, blend and simulated binary crossovers); and three mutation strategies (uniform, non-uniform and breeder mutations). Moreover, two types of ranking (linear and square) are included in the rank-based selection, which additionally increases its flexibility. BASIC GA can be easily adjusted to the concrete problems by fitting its parameters, which are divided to two groups – global and local parameters. The global parameters are: the population size, number of generations, number of samplings and crossover and mutation probabilities, and concern the whole algorithm. They do not depend on the chosen schemes of genetic operators. The local parameters refer to some of the schemes involved and are used for the fine-tuning of the system.

BASIC GA is designed to deal with constrained optimization problems, using both static and dynamic penalization techniques, and is tested on a large number of examples from the literature, with comparable results.

Results and discussions

Solution with the S-graph framework

The first solution, found during the branch-and-bound algorithm has 245 h makespan. This value is reduced systematically until the optimal solution is found. The minimal makespan schedule of the curd production is 61 h. The product requirement is fulfilled with 15 and 14 batches for product P1 and P2. The batch size of product P1 is 99.8 kg and the batch size of product P2 is 103.4 kg. Table 2 contains the optimal equipment unit allocation to the products that leads to the optimal flow shop scheduling. The optimal equipment unit allocation was found in 3.9 seconds on a 3 GHz Intel Pentium PC.

Table 2 – *Productions units used in the optimal schedule*

Type	Pasteurizers				Vat reactors			Drainers			
No	1	2	3	4	5	6	7	8	9	10	11
Product 1	0	1	0	1	0	1	0	1	0	0	1
Product 2	1	0	1	0	1	0	1	0	1	1	0

Integer programming formulation

The makespan minimization problem is solved by using BASIC-GA at following settings: pop size – 240; generations – 50; samplings – 120; crossover probability– 0.75; mutation probability – 0.1; square rank selection, simulated binary crossover and non-uniform mutation. Seven runs were carried out. Three of them have resulted in the makespan equal to 61 h, three – to 65 h and one to 71 h. The progress of best solutions using BASIC-GA during generations is shown on the Fig. 5.

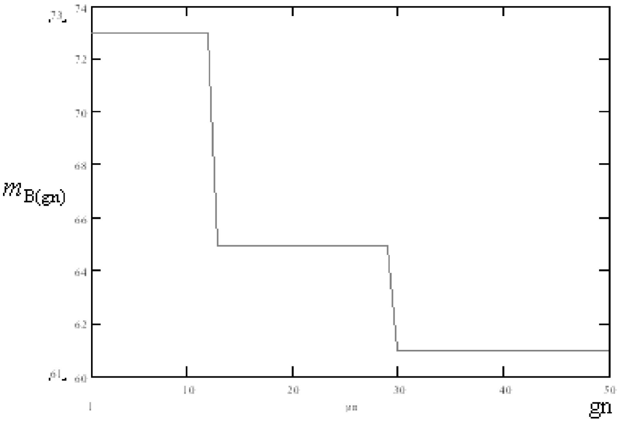

Fig. 5 – *BASIC-GA. The progress of best solutions during generations ($m_{B(gn)}$ – are values of best solution obtained and generation – gn).*

It is obvious that the minimal makespan value is equal to that obtained with S-graph approach. The corresponding batch sizes and number of batches also are the same: 99.8 kg for P1 and 103.4 kg for P2; and 15 and 14 batches respectively. Comparing the units assigned in the optimal solutions obtained by IP formulation with this by S-graph framework the different schedules can be seen. The difference comes from the units assigned to perform task 3 (draining) (see Tables 3 and 4).

Table 3 – *Process unit assignments in the first optimal schedule*

Type	Pasteurizers				Vat reactors			Drainers			
No	1	2	3	4	5	6	7	8	9	10	11
Product 1	0	1	0	1	0	1	0	1	0	1	0
Product 2	1	0	1	0	1	0	1	0	1	0	1

Table 4 – *Process unit assignments in the second optimal schedule*

Type	Pasteurizers				Vat reactors			Drainers			
No	1	2	3	4	5	6	7	8	9	10	11
Product 1	0	1	0	1	0	1	0	0	1	0	1
Product 2	1	0	1	0	1	0	1	1	0	1	0

However, the batch sizes of both products are determined from units assigned to perform pasteurization tasks. In the solutions obtained by both, S-graph approach and IP formulation they are same.

Concluding remarks

Dairy industry involves manufacturing of different milk products. They are typical multipurpose batch plants providing opportunity for different scheduling approaches to be applied to their products. Concerning to that, short-term schedules exploit their capability and flexibility. In this paper, two distinct approaches are proposed to get short-term scheduling of dairy productions. Firstly, S-graph approach with branch and bound method has been applied to solve the flow-shop scheduling problem, then the integer programming formulation of scheduling problem has been presented considering simultaneous and cyclic product manufacturing and after that the problem is solved using BASIC-GA.[10] Both approaches are used for optimal scheduling of curds processing and obtained optimal solutions are comparable. Studies described in the paper provide a good tool to the project manager for quick decision making in case of minimal makespan scheduling.

ACKNOWLEDGEMENTS

The study has been conducted with the financial support of the Bulgarian-Hungarian Bilateral Project Chem16 2003/2006, and Bulgarian National Science Fund under the Contract I-1404/04.

Nomenclature

A_1 – recipe arcs
A_2 – schedule arcs
G – graph
H – time horizon, h
I – set of products that be manufactured in the plant
m_{B} – product batch size, kg
m_{BT} – particular batch sizes, kg
m_{Q} – production demand for each product, kg
N – set of nodes of S-graph
N_{B} – number of batches processed for each product
N_{L} – number of production tasks for each product
P – set of different type units
T – cycle time, h
t – processing times, h
ts – label for task in S-graph presentation
UID – binary matrix gives plant units suitable to perform tasks
V – volumes of each unit, dm^3
v – size factors, $\mathrm{dm}^3\ \mathrm{kg}^{-1}$
w – mass fraction, %
ζ – set of binary variables
i – index of product
l – index of production task
p – index of unit

References

1. *Kondili, E., Pantelides, C. C., Sargent, R. W. H.*, Comp. & Chem. Engng. **17** (2) (1993) 211.
2. *Souzani, A. B., Pinto, J. M.*, L.A.A.R. **28** (1–2) (1998) 115.
3. *Giannelos, N. F., Georgiadis, M. C.*, Ind. & Engng. Chem. Res. **41** (9) (2002) 2178.
4. *Maravelias, C. T., Grossmann, I. E.*, Ind. & Engng. Chem. Res. **42** (13) (2003) 3056.
5. *Castro, P. M., Barbosa-Póvoa, A. P., Matos, H. A.*, Ind. & Engng. Chem. Res. **42** (14) (2003) 3346.
6. *Schilling, G., Pantelides, C. C.*, Comp. & Chem. Engng. **23** (4–5) (1999) 635.
7. *Castro, P. M., Barbosa-Póvoa, A. P., Matos, H. A., Novais, A. Q.*, Ind. & Engng. Chem. Res. **43** (1) (2004) 105.
8. *Holczinger, T., Romero, J., Puigjaner, L., Friedler, F.*, H.J.I.C. **30** (2002) 305.
9. *Sanmarti, E., Puigjaner, L., Holczinger, T., Friedler, F.*, AIChE Journal **48** (11) (2002) 2557.
10. *Shopova, E. G., Vaklieva-Bancheva, N. G.,* Comp. & Chem. Engng. **30** (8) (2006) 1293.
11. *Sanmarti, E., Friedler, F., Puigjaner, L.*, Comp. & Chem. Engng. **22** (1998) S847.
12. *Basnet, C., Foulds, L. R., Wilson, J. M.*, Ann. Oper. Res. **86** (1999) 559.
13. *Basnet, C., Foulds, L. R., Wilson, J. M.*, Proceedings of the 32nd Conference of the Operational Research Society of New Zealand, University of Canterbury, NZ, 1996, pp. 131–136.
14. *Doganis, P., Sarimveis, H.,* Journal of Food Engineering **80** (2007) 445.
15. *Doganis, P., Sarimveis, H.,* Ann. Oper. Res. **158** (1) (2008) 315.
16. *Marinelli, F., Nenni, M. E., Sforza, A.*, Ann. Oper. Res. **150** (1) (2007) 177.