Available online at www.sciencedirect.com
ScienceDirect
Journal of Cleaner Production 16 (2008) 225−232
Journal of Cleaner Production
www.elsevier.com/locate/jclepro

Effective scheduling of a large-scale paint production system

Róbert Adonyi, Gábor Biros✠, Tibor Holczinger, Ferenc Friedler*

Department of Computer Science, University of Pannonia, Veszprem, Egyetem u. 10, H-8200, Hungary

Received 1 January 2004; received in revised form 1 January 2006; accepted 18 August 2006
Available online 24 October 2006

Abstract

Due to the large variety of options offered to customers, batch production schemes are highly accepted in the paint industry implying that scheduling plays an important role in optimal allocation of plant resources among multiple products. Since in a batch process, the cleaning of equipment units is the major source of waste, waste minimization is also to be taken into account in determining the schedule.

The formerly developed *S*-graph framework [Sanmartí E, Holczinger T, Puigjaner L, Friedler F. Combinatorial framework for effective scheduling of multipurpose batch plants. AIChE Journal 2002;48(11):2557−70.] proved to be highly effective in solving multipurpose batch scheduling; it has now been specialized for solving paint production scheduling problems including waste minimization. The efficacy of the new approach is illustrated with the solution of large-scale paint production scheduling problems.

Keywords: Paint production; Batch process scheduling; *S*-graph; Waste minimization

1. Introduction

The importance of the paint and coatings industry can simply be expressed by the number of painting and coatings manufacturing facilities. For example, in the US their number is over one thousand. The paint and coatings industry produces a huge variety of products that protect, preserve, and also beautify the objects to which they are applied. Typical products include architectural coatings (e.g. house paints), industrial coatings (e.g. automotive finishes, wood furniture and fixture finishes), and special purpose coatings (e.g. traffic paints, roof coatings).

Paint production usually consists of three major operations: grinding and dispersion, mixing and coloring, and finally, discharging and packaging (see e.g. Orcun et al. [2]). Paints and coatings are typically produced in batches. They are made in stationary and portable equipment units such as high-speed dispersion mixers, rotary batch mixers, blenders, sand mills, and tanks. Raw materials include solvents, resins, pigments, and additives comprising inorganic and organic chemicals. In general, paint manufacturing does not involve chemical reactions between the raw materials; thus, the finished paint consists of a mixture of the different raw materials. Since several dozens of products are to be produced in a painting and coating manufacturing site, the corresponding scheduling problem is usually highly complex. Because of the importance of paint production, it is essential to determine the optimal or near optimal schedule for the operation.

The cleaning of the equipment units is the main source of waste generation in paint production. Since cleaning is required when the product is changed, the number of changes is also to be minimized. Furthermore, all engineering aspects of cleaner production cannot be represented in the mathematical model of scheduling [3], it is advantageous to generate several optimal and near optimal solutions for selection on the basis of further examination.

Numerous approaches are known for scheduling. The mathematical programming methods, such as mixed integer linear

* Corresponding author. Fax: +36 88 428 275.
E-mail address: friedler@dcs.vein.hu (F. Friedler).
✠ Deceased.

0959-6526/$ - see front matter
doi:10.1016/j.jclepro.2006.08.021

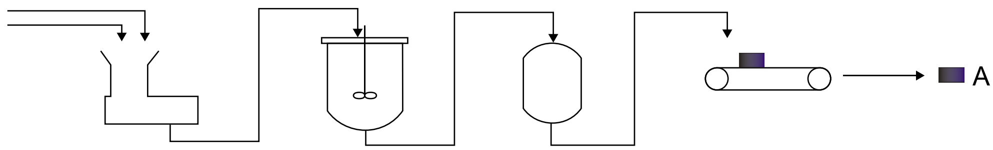

Fig. 1. Typical paint production process.

programming (MILP) [4–7] and mixed integer non-linear programming (MINLP) [8,9], are enumerative techniques that can, in principle, generate the optimal solution. In practice, however, they require an unaffordable amount of computation time. Search methods, such as tabu search [10] and simulated annealing [10,11] may generate a solution with appropriate effort, however, the quality of the solution is unknown.

Orcun et al. [12] developed an MILP model similar to the state-task network [7] for planning and scheduling of a batch paint production plant. They considered a general plant; the products can run through different production stages and follow different manufacturing routes.

Mendez and Cerda [13] introduced a novel continuous-time MILP formulation for the optimal short term scheduling. They considered different intermediate storage policies such as unlimited intermediate storage (UIS) and no intermediate storage (NIS).

2. Problem to be solved

The recipe of a batch process can be given for each product by the network of tasks. In the paint production, a product is produced by four successive tasks: grinding, mixing, storing the intermediate materials, and packing. Grinding, mixing and storing are batch type operations while packing is continuous. Fig. 1 shows the conventional representation of the production.

A task cannot be performed by a dedicated equipment unit, because there are usually more tasks than equipment units. An equipment unit is assigned to each task for a time interval where the length of the interval must not be shorter than the processing time of the related task. Changeover time is defined for an equipment unit if cleaning is necessary. The whole amount of the intermediate material is used by the successive tasks. Traditionally, such assignment of equipment units to tasks and schedule of tasks is generated that have minimal makespan. This schedule provides the highest efficiency of the production system with the possibility of unnecessarily large waste generation. For determining the schedule of tasks that requires minimal cleaning cost, the objective function of the problem has to be modified. While in the original problem the makespan, in the reformulated problem the cleaning cost must be minimized. This reformulation has minor effect on the solution procedure; therefore, an effective solver for the original problem is useful for the reformulated problem also.

In practice, the speed of the packing lines is much lower than the speed of the mixers and grinders; furthermore, the number of storage tanks is usually not large enough to store the intermediate materials in a dedicated equipment unit. This information should be taken into account during scheduling.

3. *S*-graph framework for scheduling

The *S*-graph framework developed by Sanmartí et al. [1] includes a mathematical model and solver for scheduling. The basic idea is to consider a problem formulation that manifests the unique structure of the class of scheduling problems and a solution procedure that exploits the specific structure of the problem. This approach can result in an enormous acceleration relative to general purpose solvers.

The *S*-graph framework consists of the representation of the scheduling problem [14], the basic algorithm [1], and the acceleration tools. This section summarizes the *S*-graph representation and the basic algorithm.

Each task given in the recipe is represented by a node in the *S*-graph. Moreover, an additional node is assigned to each product (see Fig. 2). The set of those equipment units that can perform task *i* is denoted by Si. The processing orders of the tasks are given by the arcs of the graph. The processing time of a task may depend on the selection of the equipment unit. A weight is assigned to each arc; it is the minimum of the processing times of the plausible equipment units. In this

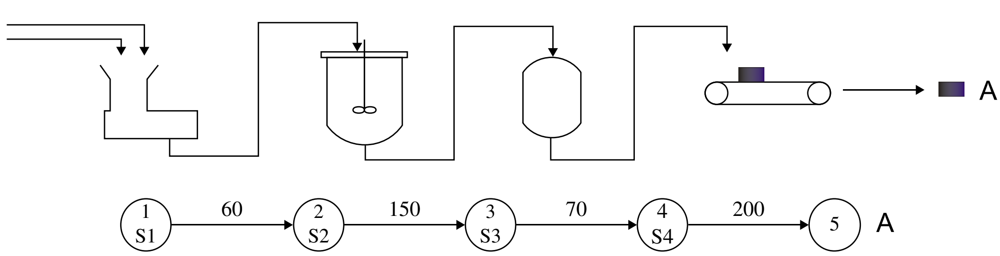

Fig. 2. Conventional and *S*-graph representation of the recipe of product A.

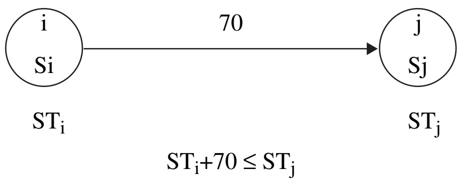

Fig. 3. The relation between the starting times of tasks connected by an arc.

representation, the value assigned to an arc expresses a lower bound for the difference of the starting times (ST_i, ST_j) of the two related tasks (see Fig. 3).

For generating multiple batches of the products, the appropriate part of the *S*-graph of the production of one batch of product is repeated according to the number of batches to get the recipe-graph of multiple batches as shown in Fig. 4.

The *S*-graph can represent the different storage policies including non-intermediate storage (NIS), unlimited intermediate storage (UIS), zero waiting (ZW) and common intermediate storage (CIS) policies. In the following, NIS will only be considered.

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 2 and consecutively to task 5 in the graph in Fig. 4, then, an arc is established from all the consecutive tasks of task 2 to task 5 as shown in Fig. 5. The weight of the arc is equal to the length of the changeover.

Because of the combinatorial characteristics of scheduling, a branch-and-bound (B&B) 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.

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 (see Appendix I for details).

4. Adapting the *S*-graph framework to paint production

Acceleration tools of the *S*-graph framework have been developed for solving large-scale scheduling problems including paint production problems.

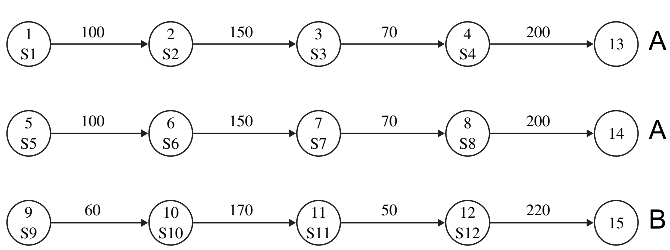

Fig. 4. Recipe-graph of the illustrative example: two batches of product A and one batch of product B.

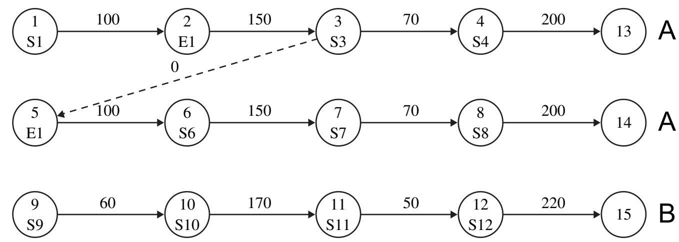

Fig. 5. *S*-graph representation of task sequence 2–5 for equipment unit E1 with NIS policy: new arc from node 3 to node 5 with 0 weight (dotted line).

4.1. Elimination of redundancy in the search space

An industrial scheduling problem may contain a large number of batches of the same product. The basic algorithm of the *S*-graph framework considers every batch as an individual product that implies a significant redundancy in the search space. To exclude the redundancy during the search, Holczinger et al. [15] inserted additional arcs into the recipe-graph that resulted in sufficient acceleration for solving large-scale problems.

4.2. Scheduling of the storage tanks

In the paint production problem, the materials can be stored in the grinders, mixers and tanks. Since a packing line itself has no storing capacity, it requires storing its feed in some equipment units. This requirement can simply be expressed by introducing an additional arc. For example, the arc from node 9 to node 7 in the *S*-graph in Fig. 6 ensures that the intermediate material can be stored in tank E7 during packing. Note that in this specific example, arc from node 4 to node 7 becomes redundant and therefore, it can be eliminated.

4.3. LP model for bounding

The longest path algorithm may not give a sharp lower bound, especially near to the root of the enumeration tree. An appropriate, linear programming (LP) model can sharpen the bound with the help of the longest path.

Suppose that L_j denotes the length of the longest path leading to node j in the *S*-graph of a partial problem. Let c_i denote the lower bound of the finishing time of equipment unit i. The value of c_i can be determined by formula (1).

$$c_i = \max\left(\max_{j\in M_i}\left(L_j + t_{ij}\right), \min_{j\in \overline{M}_i}\left(L_j\right)\right) + \sum_{j\in \overline{M}_i} t_{ij} \quad (1)$$

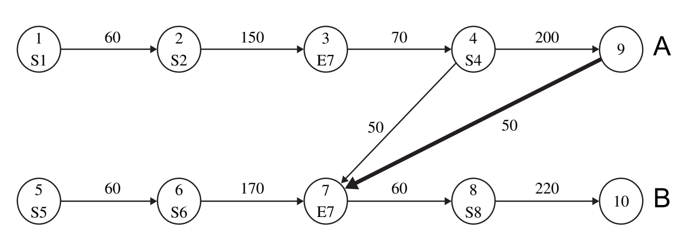

Fig. 6. Arc from node 9 to node 7 ensures that tank E7 is available until task 4 (packing) is finished.

Table 1
Recipes of the Example

Task	Product A		Product B		Product C	
	Eq.	Time (h)	Eq.	Time (h)	Eq.	Time (h)
1	E1	8	E1 E2	9 11	E1 E2	7 7
2	E2 E3	15 5	E3	5	E3	4

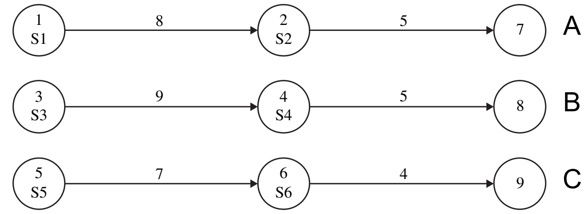

Fig. 7. Recipe-graph of Example.

where M_i denotes the set of scheduled nodes of those tasks that are to be performed by equipment unit i, $\overline{M}_i$ denotes the set of those unscheduled nodes that can be performed only by equipment unit i, and t_{ij} is the operating time of task j if it is performed by equipment unit i.

Let $\overline{N}$ denote the set of those nodes of the *S*-graph that are unscheduled and have optional equipment units to perform. For equipment unit i and task j, non-negative continuous variable x_{ij} ($i = 1, 2, \ldots, n, j \in \overline{N}$) denotes the duration of the assignment of equipment unit i to task j. Therefore, the finishing time of the activity of equipment unit i is $c_i + \sum_{j\in\overline{N}} x_{ij}$ ($i = 1, 2, \ldots, n$), it is a lower bound to makespan X. Every task from set $\overline{N}$ has to be performed by some equipment units, i.e. $\sum_{i\in S_j} x_{ij}/t_{ij} \geq 1$, where $j \in \overline{N}$.

The solution of the LP problem (2)−(5) gives a lower bound, X, for the partial problem:

$$\min X \tag{2}$$

s.t.

$$c_i + \sum_{j\in\overline{N}} x_{ij} \leq X, \quad i = 1, 2, \ldots, n \tag{3}$$

$$\sum_{i\in S_j} \frac{x_{ij}}{t_{ij}} \geq 1, \quad j \in \overline{N} \tag{4}$$

$$x_{ij} \geq 0, \quad i = 1, 2, \ldots, n, \ j \in \overline{N} \tag{5}$$

where n is the number of equipment units, S_j is the set of equipment units that can perform task j.

4.3.1. Example

Three equipment units, E1, E2, and E3, are available to generate three products, A, B, and C. The recipes of the products are given in Table 1. The recipe-graph of the example is given in Fig. 7 for one batch of each product.

An LP model is solved at the root of the enumeration tree of the branch-and-bound algorithm for the initial bound. Since $\overline{M}_1 = \{1\}$, $\overline{M}_2 = \emptyset$, $\overline{M}_3 = \{4, 6\}$, $c_1 = 8$, $c_2 = 0$, and $c_3 = 16$, the LP problem can be formulated as given by Eqs. (6)−(13).

$$\min X \tag{6}$$

s.t.

$$8 + x_{13} + x_{15} \leq X \tag{7}$$

$$x_{22} + x_{23} + x_{25} \leq X \tag{8}$$

$$16 + x_{32} \leq X \tag{9}$$

$$\frac{x_{13}}{9} + \frac{x_{23}}{11} \geq 1 \tag{10}$$

Table 2
Recipes for products A, B, C, D, E, and F

Task	Product A		Product B		Product C		Product D		Product E		Product F	
	Eq.	Time (min)	Eq.	Time (min)	Eq.	Time (min)	Eq.	Time (min)	Eq.	Time (min)	Eq.	Time (min)
1	E1	60	E1	60	E2	60	E3	60	E4	40	E5	40
2	E6	310	E7 E8 E9	240 120 240	E8	120	E7 E9	240 240	E6 E8	300 120	E7 E8	240 120
3	E10 E11 E13 E15 E17 E19 E20	60 120 60 120 60 120 60	E11 E13 E15 E17 E19 E20	120 60 120 60 120 60	E11 E12 E13 E14 E16	120 70 70 60 50	E10 E11 E13 E14 E15 E17 E18 E19 E20	60 120 60 90 120 60 90 120 60	E10 E12 E14 E16 E18 E20	60 90 90 90 90 60	E10 E15 E16 E17 E18 E19 E20	60 120 90 60 90 120 60
4	E21 E22	720 540	E22 E23	540 720	E21 E22	720 540	E22 E23	540 720	E21	720	E21 E22 E23	720 540 720

Table 3
Number of batches of the products

Product	A	B	C	D	E	F
Number of batches	3	5	1	3	9	3

$$\frac{x_{15}}{7}+\frac{x_{25}}{7}\geq 1 \tag{11}$$

$$\frac{x_{22}}{15}+\frac{x_{32}}{5}\geq 1 \tag{12}$$

$$x_{13},x_{15},x_{22},x_{23},x_{25},x_{32}\geq 0 \tag{13}$$

The resultant lower bound is $X=17.4$. It is a sharper bound than that given by the longest path algorithm, i.e. 14.

As shown, the LP model can generate sharper lower bound than the longest path algorithm. In practice, however, the construction and solution of the LP model for every partial problem may need too much computational effort. Furthermore, approaching the leaves of the enumeration tree, the difference between the bound from the LP and the longest path algorithm is usually reduced. Hence, it is valuable to use the LP model for those partial problems that are near the root of the enumeration tree.

5. Application

Twenty-three equipment units, E1–E23, are available to generate six products, A, B, C, D, E, and F. The recipes of the products are given in Table 2. The changeover time is 70 min for equipment units E6, E7, E8, and E9, and 100 min for equipment units E1–E5 and E10–E20. All other changeover times are supposed to be zero. The number of batches to be produced is given in Table 3.

To get the minimal makespan solution, an MILP model based on the work of Mendez and Cerda [13] has been solved by GAMS/CPLEX 7.5 and required 34 s CPU time on an AMD Athlon XP 2200 MHz. The same problem has been solved by the proposed algorithm that required 0.9 s CPU time on the same PC. The resultant makespan is 6700 min. Figs. 8 and 9 show the schedule-graph and the Gantt-chart of the optimal solution, respectively.

The cleaning of the equipment units is one of the most polluting and costly operations. The minimal makespan schedule of Figs. 8 and 9 contains 11 cleaning operations. The cleaning

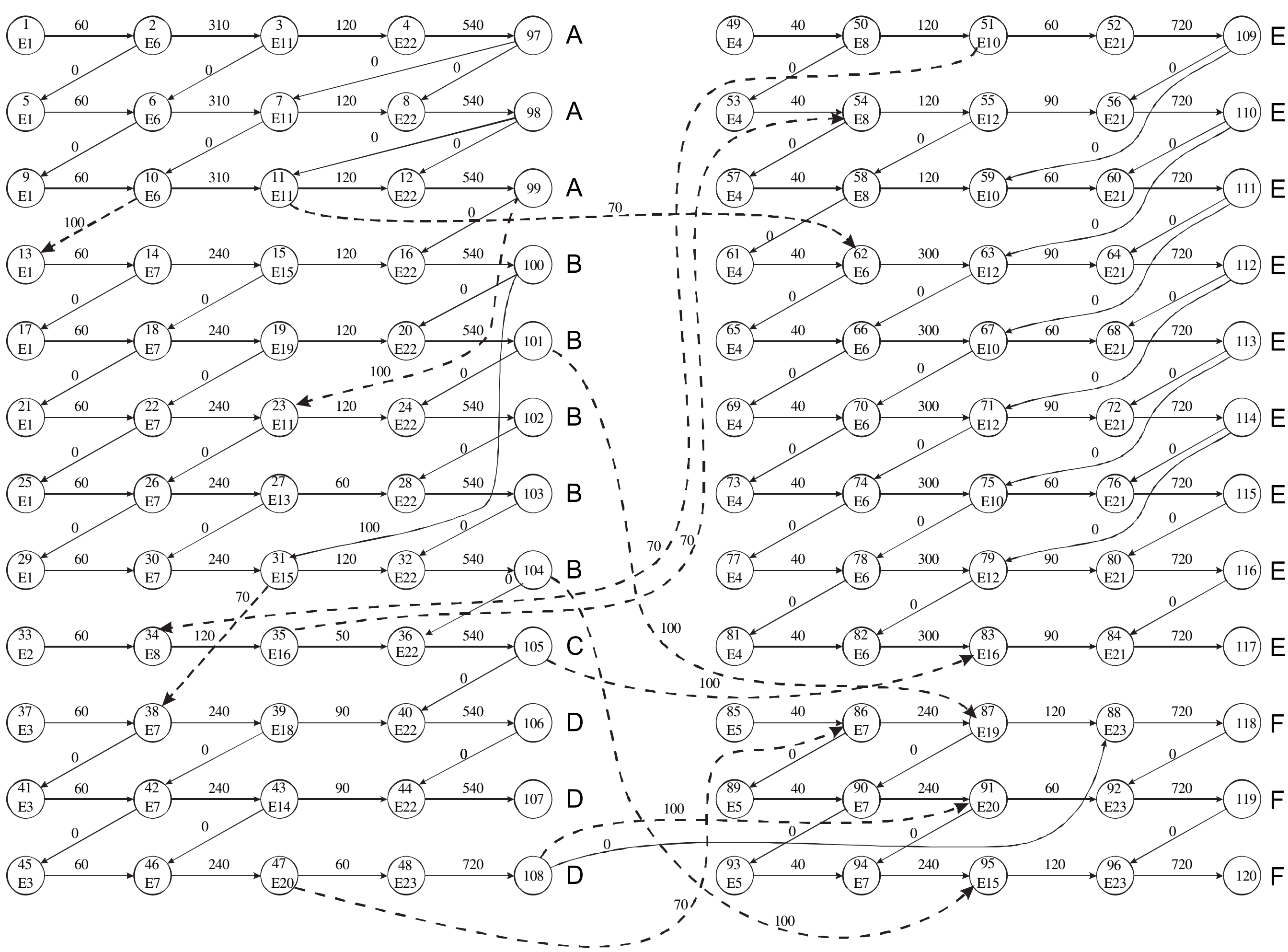

Fig. 8. Schedule-graph of the solution with minimal makespan (dotted arcs represent changeovers with cleaning costs).

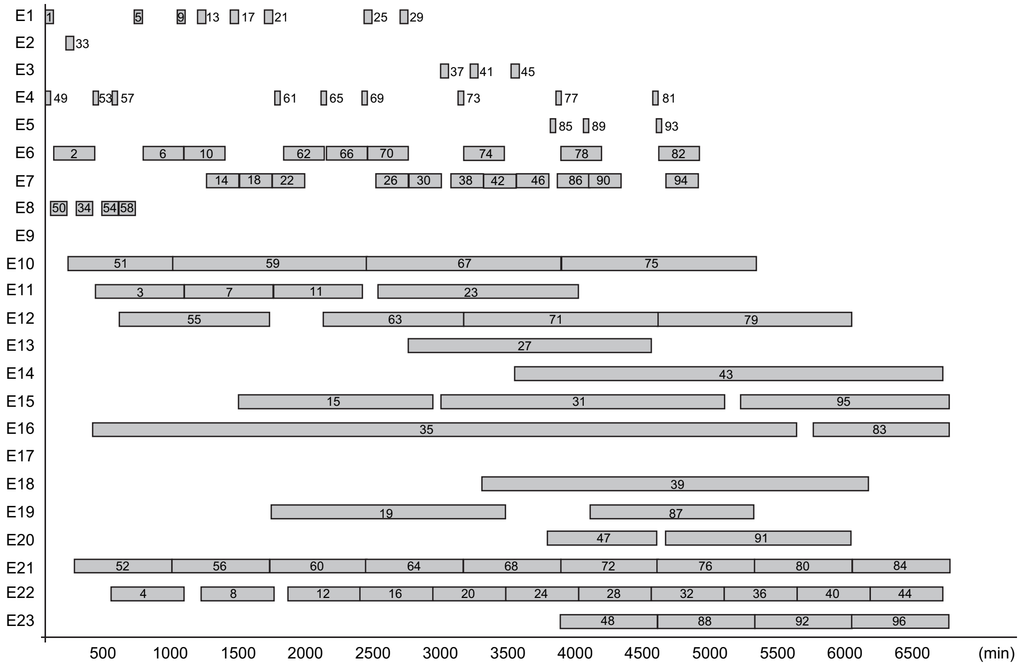

Fig. 9. Gantt-chart of the solution presented in Fig. 8.

Table 4
Cleaning cost of grinders E1, E2, E3, E4, and E5

From	To					
	Product A	Product B	Product C	Product D	Product E	Product F
Product A	0	500	500	500	500	500
Product B	1000	0	500	500	500	500
Product C	1000	500	0	500	500	500
Product D	1000	500	500	0	500	500
Product E	2000	1000	1000	1000	0	500
Product F	2000	1000	1000	1000	500	0

Table 5
Cleaning cost of mixers E6, E7, E8, E9, and E10

From	To					
	Product A	Product B	Product C	Product D	Product E	Product F
Product A	0	1500	1500	1500	1500	1500
Product B	2500	0	1500	1500	1500	1500
Product C	2500	1500	0	1500	1500	1500
Product D	2500	1500	1500	0	1500	1500
Product E	5000	2500	2500	2500	0	1500
Product F	5000	2500	2500	2500	1500	0

Table 6
Cleaning cost of storage tanks E10–E20

From	To					
	Product A	Product B	Product C	Product D	Product E	Product F
Product A	0	1000	1000	1000	1000	1000
Product B	2000	0	1000	1000	1000	1000
Product C	2000	1000	0	1000	1000	1000
Product D	2000	1000	1000	0	1000	1000
Product E	4000	2000	2000	2000	0	1000
Product F	4000	2000	2000	2000	1000	0

```
procedure main
notation: n: number of equipment units
          N_i (i = 1,2,..., n): set of tasks that can be performed by equipment unit i
          last_node: set of pairs (i, j), where i is an equipment unit and j is a task (node)
          PP = (G (N,A_1,A_2), bound, last_node, SOUN)
input: recipe-graph G (N,A_1,∅) and N_i (i = 1,2,...,n)
begin
    SET = ∅; bound = 0; SOUN = N_1∪N_2∪...∪N_n; last_node = ∅; current_best = ∞;
    put (G (N,A_1,∅), bound, last_node, SOUN) into SET;
    while SET ≠ ∅ do
        begin
        select and remove one element from SET, it is denoted by PP;
        branching(PP);
        end;
    if current_best < ∞ then print solution;
end
```

Fig. 10. The main procedure of the scheduling algorithm.

operations are denoted by the dotted changeover arcs on the *S*-graph. Tables 4–6 include the cleaning costs of the equipment units, where the cleaning costs depend on the difficulty of cleaning, i.e. on the sequence of products of the equipment units.

The cleaning cost of the solution with minimal makespan is 14,000 cost units (CU). However, the minimal cost schedule has only 3500 CU containing only four cleaning operations while its makespan is 6910 min. If the cleaning cost is limited to 5500 CU, the corresponding makespan is reduced to 6700 min.

6. Concluding remarks

The formerly developed *S*-graph framework of batch scheduling has been extended to solve complex paint production problems. The proposed methodology is useful for generating the solution of minimal makespan, the solution of minimal cleaning cost, and the solution of minimal makespan with limited cleaning costs. The examination of an industrial paint production problem illustrates the efficacy of the proposed algorithm.

```
procedure branching(PP)
comment: generates all child partial problem of partial problem PP
notation: graph(PP) = G (N,A_1,A_2)
          bound(PP)= bound
          last_node(PP)=last_node
          SOUN(PP)=SOUN
begin
    let EQ be an equipment unit that can be assigned to an unscheduled task (node);
    let SO = N_EQ ∩ SOUN(PP);
    for all k∈SO do
        begin
        if there is no pair (i, j)∈last_node such that i = EQ then
            begin
            put (graph(PP), bound(PP), last_node(PP)∪{(EQ,k)}, SOUN(PP)\{k})
                into SET;
            end;
        else
            begin
            let G_0(N,A_1,A_2) = graph(PP);
            for all (j, l)∈A_1 do G_0(N,A_1,A_2) = G_0(N,A_1,A_2∪{(l, k)});
            bounding(G_0(N,A_1∪A_2), bound);
            if bound < current_best then
                begin
                if SOUN(PP)\k = ∅ then
                    update current_best, SET, and solution;
                else
                    put (G_0(N,A_1,A_2), bound, last_node(PP)∪{(EQ,k)}\{(EQ,j)},
                        SOUN(PP)\{k}) into SET;
                end;
            end;
        end;
    return;
end;
```

Fig. 11. The branching procedure of the scheduling algorithm.

```
procedure bounding (G (N,A), bound)
begin
    for all s∈N_p do cycle_search (s, {s});
    if no cycle then
        bound = longest_path (G (N,A));
    else
        bound = ∞;
end;
```

Fig. 12. The bounding procedure of the scheduling algorithm.

Acknowledgement

The authors express their appreciation to Dr. T. Majozi for his valuable comments.

Appendix I. Basic algorithms for the *S*-graph framework [1]

The main procedure initializes the values of the variables (see Fig. 10). The branching procedure (see Fig. 11) generates the partial problems. 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 tasks.

The bounding procedure (see Fig. 12) tests the feasibility of a partial problem first. If this test is proved positive, it determines a lower bound for the makespan of all solutions that can be derived from this partial problem.

References

[1] Sanmartí E, Holczinger T, Puigjaner L, Friedler F. Combinatorial framework for effective scheduling of multipurpose batch plants. AIChE Journal 2002;48(11):2557–70.
[2] Orcun S, Discioglu A, Kuban Altinel I, Hortacsu Ö. Scheduling of batch processes: an industrial application in paint industry. Computers and Chemical Engineering 1997;21:673–8.
[3] Markowski M, Urbaniec K. Optimal cleaning schedule for heat exchangers in a heat exchanger network. Applied Thermal Engineering 2005;25:1019–32.
[4] Burkard Rainer E, Fortuna T, Hurkens CAJ. Makespan minimization for chemical batch processes using non-uniform time grids. Computers and Chemical Engineering 2002;26:1321–32.
[5] Méndez CA, Henning GP, Cerdá J. An MILP continuous time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. Computers and Chemical Engineering 2001;25:701–11.
[6] Majozi T. An effective technique for wastewater minimisation in batch processes. Journal of Cleaner Production 2005;13:1374–80.
[7] Kondili E, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations. Part 1. MILP formulation. Computers and Chemical Engineering 1993;17:211–27.
[8] Mockus L, Reklaitis GV. Continuous time representation approach to batch and continuous process scheduling. Part 1. MINLP formulation. Industrial and Engineering Chemistry Research 1999;38:197–203.
[9] Sahinidis NV, Grossmann IE. MINLP model for cyclic multiproduct scheduling on continuous parallel lines. Computers and Chemical Engineering 1991;15:85–103.
[10] Bhushan Swarnendu, Karimi IA. Heuristic algorithms for scheduling an automated wet-etch station. Computers and Chemical Engineering 2004;28:363–79.
[11] Tavakkoli-Moghaddam R, Jolai F, Vaziri F, Ahmed PK, Azaron A. A hybrid method for solving stochastic job shop scheduling problems. Applied Mathematics and Computation 2005;170:185–206.
[12] Orcun S, Altinel IK, Hortacsu Ö. General continuous time models for production planning and scheduling of batch processing plants: mixed integer linear program formulations and computational issues. Computers and Chemical Engineering 2001;25:371–89.
[13] Mendez CA, Cerda J. An MILP continuous-time framework for short-term scheduling of multipurpose batch process under different operation strategies. Optimization and Engineering 2003;4:7–22.
[14] Sanmartí E, Friedler F, Puigjaner L. Combinatorial technique for short term scheduling of multipurpose batch plants based on schedule-graph representation. Computers and Chemical Engineering 1998;22(Suppl.):847–50.
[15] Holczinger T, Romero J, Puigjaner L, Friedler F. Scheduling of multipurpose batch processes with multiple batches of the products. Hungarian Journal of Industrial Chemistry 2002;30:305–12.

Róbert Adonyi received his MSc in Information Technology in 2001 from the University of Veszprem, Hungary. His main research area is batch process scheduling. He is currently pursuing his doctoral degree at the Department Computer Science of University of Pannonia (formerly University of Veszprem).

Gábor Biros received his MSc in Information Technology in 2001 from the University of Veszprem, Hungary. He started his PhD work in process systems engineering. The beginning of his promising professional career came to an end by a tragic traffic accident in 2004.

Tibor Holczinger received his MSc in Information Technology in 1997 from the University of Veszprem, Hungary. He received his PhD degree in Information Technology in 2004 from the University of Veszprem. His research area is batch process scheduling.

Professor **Ferenc Friedler** (www.dcs.vein.hu/friedler) is the Dean of Faculty of Information Technology, the Head of Department of Computer Science, and the Chair of Information Technology (IT) PhD program at University of Pannonia (formerly University of Veszprem), Hungary. He received his MSc degree in Mathematics, PhD and DSc in Chemical Engineering, and Habilitation in IT. His research interests include process systems engineering and operations research. He is one of the founders of the *P*-graph framework for process design and synthesis. Dr. Friedler is Hungarian representative in the Computer Aided Process Engineering Working Party of the European Federation of Chemical Engineers. He has been a chair and co-chair of international conferences and honoured with the Knight's Cross Order of Merit of the Republic of Hungary.