SCHEDULING OF MULTIPRODUCT MULTISTAGE BATCH PLANTS WITH UNCERTAIN PROCESSING TIMES: AN INNOVATIVE CONSTRAINT PROGRAMMING APPROACH

Franco M. Novara#, Gabriela P. Henning*#
*INTEC (UNL, CONICET), Güemes 3450, Santa Fe, Argentina
#Facultad de Ingeniería Química, UNL, Santiago del Estero 2829, Santa Fe, Argentina

Abstract

Due to the intrinsic uncertainty and dynamics of industrial environments schedulers must continually reconcile what is expected with what actually happens. One of the most common sources of uncertainty encountered at the operational level is the one associated with variable processing times. This work contributes in the area of proactive scheduling by developing an innovative Constraint Programming (CP) model able to cope with uncertain processing times at the decision stage, prior to scheduling and without resorting to the generation of scenarios. The application of the model to various instances of three case studies shows that the approach is computationally efficient. In addition, when the obtained schedules are compared with the agendas that were reached by a deterministic CP formulation, it is shown that they absorb the variability of the processing times better.

Keywords

Scheduling, Uncertain processing times, Constraint programming, Multiproduct multistage batch plants

Introduction

Nowadays, industrial companies operate under the pressures of competitive economy, which pushes them to have a great variety of products manufactured at low prices, to deliver on time, to be flexible, etc. This context calls for an outstanding operational efficiency. However, industrial environments continuously face uncertainties and unexpected events that conspire against the desired efficiency. Given this situation, the interest on scheduling methodologies that can cope with various sources of uncertainty, such as changes in product orders, equipment failures, processing time variability, recipe changes, raw materials late arrival, manpower availability, etc., has increased in the last decade. Several works have been recently reviewed by Gupta et al. (2016). The most common techniques proposed to deal with uncertainty are stochastic programming, robust optimization and robust counterpart optimization (Li and Ierapetritou, 2008). Alternative approaches such as fuzzy programming and parametric programming methods have also been reported.

Two types of methodologies that deal with uncertainties in scheduling environments can be distinguished (Bonfill et al., 2008): reactive and proactive scheduling. Reactive scheduling methodologies are implemented at execution time to face unforeseen events once they have occurred. On the other hand, proactive approaches incorporate the knowledge of uncertainty at the decision stage, prior to scheduling. This contribution focuses on the consideration of a priori uncertainties during the decision stage in order to generate more resilient schedules. Specifically, the short-term scheduling of multiproduct multistage batch plants with uncertain operation times is addressed in a proactive fashion. Variable processing times/rates are one of the most common sources of uncertainty and they can lead to the generation of idle and wait times. While the first ones cause equipment underutilization and reduce plant productivity, wait times can generate order delays and/or batch rejections due to low quality problems associated with material deterioration.

Processing time uncertainty has been treated with various approaches such as fuzzy programming (Balasubramanian and Grossmann, 2003) and genetic algorithms (Bonfill et al., 2008). This work deals with uncertainties in the processing times by means of a novel Constraint Programming (CP) model that does not rely on the generation of scenarios or worst case formulations.

The remainder of this paper is organized as follows. The problem statement is introduced in the next section, which also includes a description of how to capture uncertainties in the processing times. Then, the stochastic CP approach is presented, followed by a discussion of the results of various instances of three case studies. Final remarks and future work conclude the paper.

Problem statement

A set of batches of different products has to be scheduled in order to achieve two conflicting objectives: on-time delivery and plant efficiency. As most multiproduct multistage industrial plants implement already specified recipes, which correspond to batches of predefined size, the lot sizing problem will not be considered in this contribution. Each batch has to be processed at each stage in one of the multiple non-identical units that operate in parallel. The processing environment has banned batch-unit assignments, forbidden sequences and topology restrictions. Regarding the intermediate storage and inter-stage waiting policy, it is assumed a non-intermediate storage with unlimited wait (NIS-UW) one. In addition, changeover tasks, whose duration depends on the sequence and/or the unit, may be required. Processing times are assumed to be independent stochastic variables that take values with probabilities given by a fixed probability distribution. The rest of the problem variables and parameters are assumed to be deterministic.

Processing times having a stochastic behavior

Provided that the processing times uncertainty directly affects the end time of each processing task, the proposed methodology associates a stochastic variable with the completion time of each batch. The value of this variable, which is named *end time subject to deviation* (*et-StD*$_b$), has a *P* probability of occurrence. For any given batch *b*, *et-StD*$_b$ cannot be described accurately, with mathematical precision, without a significant CPU effort. Nevertheless, it could be estimated in a practical way. The variable that approximates *et-StD*$_b$ in the proposed CP model is *eet-Batch*$_b$, the *estimated end time of batch b*. *eetBatch*$_b$ is calculated by means of the expected end time plus *n* times the estimated end time standard deviation, with *n* being the *z* value associated with a normal cumulative probability *P*. The expected end time is computed by adding processing times that are assumed to have normal probability distributions, as well as idle times and changeover/setup times, which are supposed to have a deterministic behavior. Therefore, it is also expected to have a normal distribution.

Estimation of end time standard deviation

The end time standard deviation of a batch *b* included in a given agenda can be estimated as *n* times the maximum standard deviation associated with the start times of the tasks of this batch in the set of assigned units (*devStart*$_b$), plus *n* times the standard deviation of the processing time associated with the whole set of tasks demanded by batch *b* (*devBatch*$_b$). While *devStart*$_b$ takes into account the role of processing time uncertainty of those tasks that precede the execution of batch *b* in all the assigned units, *devBatch*$_b$ captures the uncertainty in the execution of the tasks that pertain to batch *b* itself. Thus, one of the underlying ideas is to push forward in the agenda the whole set of tasks corresponding to batch *b* a value equal to *devStart*$_b$ in order to take into account the uncertainties in the processing times of those tasks that precede batch *b*. Similarly, by means of *devBatch*$_b$ the expected end time is pushed forward due to the variations of the processing times associated with batch *b* itself. Figure 1 shows the conceptual interpretation of these variables.

Two conservative simplifications have been made: (i) *devStart*$_b$ and *devBatch*$_b$ are linearly added; (ii) the idle-times and/or wait times that may be part of the agenda and could act as buffers, compensating certain possible delays, are not considered with the role of bumpers.

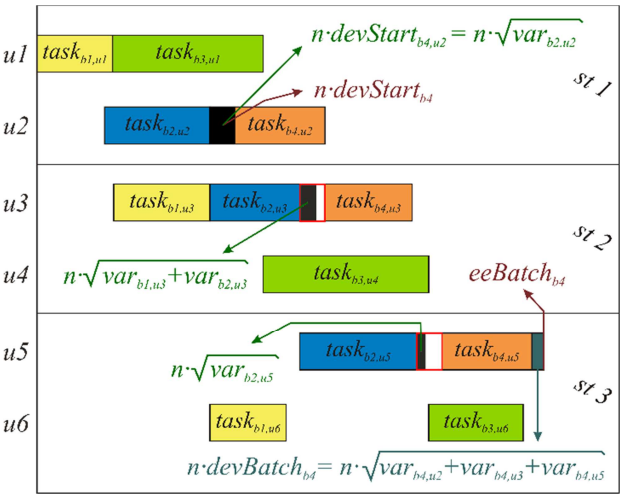

Figure 1. Interpretation of the main stochastic variables that participate in the proposal

Innovative CP stochastic model

The proposed methodology handles uncertainties in the processing times by creating two associated schedules that are simultaneously built. One of them, the production schedule, is generated using nominal processing times. No buffer times neither overestimated (conservative) processing times are included a priori to compensate possible delays that may result from the realizations of the uncertainties. Nevertheless, the assignment and sequencing decisions would be performed in order to better anticipate positive variabilities (processing times greater than the

nominal duration of the tasks) and their impact on total tardiness. The second schedule is not an operational agenda. It is employed to capture the variances of the task set that is assigned to each equipment unit. The structure of both agendas is similar (same task-unit assignments and task sequencing). However, in the variance agenda, instead of processing times, the span of the tasks is equal to the variance sizes. Another difference is that all the tasks assigned to a given unit are scheduled from the beginning of the planning horizon, without any intermediate idle time.

Model implementation

The proposed CP model was developed using the OPL programming language, supported by the IBM ILOG CPLEX Optimization Studio environment (IBM ILOG, 2013). The IBM ILOG OPL language, combined with the IBM ILOG CP Optimizer constraint programming engine, provides some specific scheduling constraints, functions, as well as different types of variables, aimed at describing scheduling problems properly (Novara et al., 2016). Among these features, a specific construct that manages sequence and unit depended changeovers in an efficient way is employed. In addition, a warm-start mode is available, which allows specifying an initial point to reduce the computational effort, especially for big size models.

Nomenclature

Sets/Indexes
B/b: batches to be produced within the planning horizon
C_u/-: units of stage *s+1*, which are unconnected to unit *u*, belonging to stage *s*
F_p/-: products that are forbidden as successors of product *p* when assigned to the same unit
P/p: products to be manufactured
S/s: processing stages
U/u: equipment units

Parameters
co: <*u,p,p'*> triplets containing the changeover time between products *p* and *p'* in unit *u*
dd_b: due-date of batch *b*
n: number of standard deviations associated with a probability P that is established by the scheduler in relation to certain stochastic variables
$pt_{p,u}$: nominal processing time required by a batch of product *p* in unit *u*

Variables
$devBatch_b$: float variable that represents the standard deviation of the processing time associated with the whole set of tasks demanded by batch *b*
$devSeq_u$: sequence variable defined for each unit *u*. It represents an ordering of interval variables $var_{b,u}$ on *u*. Each of these interval variables is characterized by an attribute defining the product *p* associated with batch *b*
$devStart_b$: given the units in which the tasks of batch *b* are carried out, this float variable captures the maximum of the processing time standard deviations of the set of tasks that precede the execution of batch *b* in each of such units.
$eetBatch_b$, float variable capturing the estimated end time of batch *b*
$stTask_{b,s}$: interval variable representing the processing task of batch *b* at stage *s*
$task_{b,u}$: interval variable representing the processing task of batch *b* in unit *u*
tardiness: float variable that captures the expected total tardiness.
$unitBatchSeq_u$: sequence variable defined for each unit *u*. It represents an ordering of processing task interval variables associated with unit *u*. Each variable is characterized by the *p* product associated with batch *b*
$var_{b,u}$: interval variable representing the variance of the processing time of batch *b* in unit *u*

Constraints

Expression (1) enforces each batch to be assigned to just one processing unit at each stage. Constraint (2) ensures precedence relationships between adjacent processing tasks of any batch *b*.

$$alternative\left(stTask_{b,s}, all(u \in U_s)\ task_{b,u}\right), \quad \forall s \in S, \forall b \in B \tag{1}$$

$$endAtStart\left(stTask_{b,s}, stTask_{b,s'}\right) \quad \forall b \in B, \forall s, s' \in S, s \neq Card(S), s' = s+1 \tag{2}$$

Topology restrictions are captured by means of expression (3). Constraint (4) avoids overlapping the execution of tasks in any unit *u* and simultaneously inserts changeover times between consecutive tasks assigned to such unit. Expression (5) avoids forbidden sequences by resorting to sequence variables and the *typeOfNext* construct. See details in Novara et al. (2016) and IBM ILOG (2013).

$$minl\left(endOf\left(task_{b,u}\right), endOf\left(task_{b,u'}\right)\right) = 0 \quad \forall b \in B, \forall u' \in C_u, \forall u, u' \in U \tag{3}$$

$$noOverlap(unitBatchSeq_u, co), \qquad \forall u \in U \tag{4}$$

$$typeOfNext\left(unitBatchSeq_u, task_{b,u}\right) \neq p' \quad \forall (p, p') \in Fp, \forall u \in U, \forall b \in B_p \tag{5}$$

Constraints (6) and (7) describe the calculation of variables $devStart_{b,u}$ and $devBatch_b$, respectively. As previously mentioned, they are the two stochastic variables that are employed to estimate the end time standard deviation of batch *b*. The role of the fictitious tasks representing the task variances, $var_{b,u}$, appears precisely in constraints (6) and (7). In (6) the start time of each variance interval variable, captures the variance associated with the processing times of the set of activities that are predecessors of batch *b* in the same equipment unit. Expression (8) shows how the estimated end time of batch *b* is calculated by means of deterministic and stochastic variables. Finally, constraint (9) describes the expected total tardiness, which is the objective function to be minimized.

$$devStart_b = \sqrt{\max_{\forall u \in U}\left(startOf\left(var_{b,u}\right)\right)}, \forall b \in B \tag{6}$$

$$devBatch_b = \sqrt{\sum_{\forall\, u \in U} sizeOf\left(var_{b,u}\right)}, \forall\, b \in B \quad (7)$$

$$eetBatch_b = endOf\left(stTask_{b,s}\right) + n \cdot devStart_b + n \cdot devBatch_b \quad \forall\, b \in B, \forall\, s \in S, s = card(s) \quad (8)$$

$$tardiness = \sum_{\forall\, b \in B} maxl(0, eetBatch_b - dd_b) \quad (9)$$

Constraints (10)-(12) link the operational agenda with the variance or auxiliary schedule. If the interval variable $task_{b,u}$, representing the processing task of batch b in unit u is included in the solution, the corresponding interval variable representing the variance of this task has to be included too, as shown in (10). Expression (11) avoids the overlapping of interval variables representing variances. Finally, constraint (12) enforces the interval variables representing processing and variance tasks to follow the same sequence in each unit.

$$presenceOf\left(task_{b,u}\right) = presenceOf\left(var_{b,u}\right) \quad \forall\, b \in B, \forall\, u \in U \quad (10)$$

$$noOverlap(devSeq_u), \qquad \forall\, u \in U \quad (11)$$

$$\min\{ endOf\left(task_{b',u}\right) - endOf\left(task_{b,u}\right), endOf\left(var_{b,u}\right) - endOf\left(var_{b',u}\right)\} \leq 0 \quad \forall\, b, b' \in B, \forall\, u \in U \quad (12)$$

Parameters such as unit ready times, batch release times and nominal processing times of the batch activities, can be taken into account without resorting to special constraints. This is done by declaring the domain of each interval variable that represents the execution of a task belonging to a batch recipe. The same reasoning applies to the duration of interval tasks representing variance tasks.

Case studies and results

The methodology was tested by means of three deterministic examples available in the literature that were slightly modified. In all the cases, a NIS-UW intermediate storage/inter-stage waiting policy was adopted. In addition, total tardiness was the objective function. Each case study was solved under various uncertain conditions. Stochastic processing times were captured by means of different asymmetrical triangular distributions that try to resemble what happens in realistic production environments, where more delays than anticipations occur. Each distribution was generated assuming the deterministic processing time as the mode. The lower value was randomly created by subtracting the mode the result of multiplying its value by an aleatory number belonging to the [0, *inf*] interval. Similarly, the upper limit was generated by adding the mode the result of multiplying its value by a random number belonging to the [0, *sup*] interval. By adopting different values for the *inf* and *sup* parameters, various case studies instances have been generated, as seen in Table 1.

Case study 1 (C1): It is based on Example 4 of Marchetti and Cerdá (2009), which corresponds to a facility having 5 processing stages and 12 non-identical units. Sequence dependent changeovers are considered, but set-up times are avoided. Other modifications from the original example are: (i) NIS-ZW policy instead of a UIS one; (ii) topology constraints are added; (iii) limited availability of discrete resources, such as electricity or manpower, are ignored; (iv) due-dates were modified in order to obtain a deterministic solution having zero total tardiness.

Case study 2 (C2): It is based on the multiproduct batch plant having 5-stages, 25 non-identical units and topological constraints, which was studied by Zeballos et al. (2011). Product orders and processing units are characterized by release and ready times, respectively. The due-dates correspond to the set DD2 proposed by the authors. In addition, some orders cannot be processed in certain units and there are forbidden processing sequences.

Case study 3 (C3): Based on the example of Castro et al. (2009). The facility has 5 stages with 20 dissimilar processing units. Fifty product orders are to be scheduled. Sequence dependent changeovers are considered.

The results of the deterministic and stochastic approaches have been compared. The deterministic agendas have been obtained with the CP model proposed by Novara et al. (2016). All the deterministic solutions corresponding to Cases 1-3 are optimal schedules having a total tardiness equal to zero. However, the stochastic ones (the operational schedules of this approach) have performance values that are not that far appart: Examples C1-a to C1-d have a total tardiness of zero, whereas the C1-e to C1-j instances a value equal to 3.2. Similary, C2-a and C2-b have total tardiness values of 17.3 and 0, respectively. Finally, both C3-a and C3-b have total tardiness of zero. If Makespans are compared, the Makespans of the C1 set of instances range from 96 to 129.3, whereas the deterministic value is of 113.3 time units. For cases C2-a and C2-b the stochastic makespan values are 336.4 and 334.5, quite close to the 335.5 deterministic one. Finally, for the C3-a and C3-b examples the stochastic makespan values are 59.590 and 58.700, close to the 58.970 deterministic one.

Then, the different agendas have been contrasted by means of simulation to test their resilience to uncertain processing times. Table 1 and Figures 2-3 allow comparing the results of the simulations for various variability conditions (see *inf* and *sup* parameter values in Table 1). The different performance indicators (Total Tardiness, Makespan, etc.) reported in Table 1 correspond to the average values obtained from 50.000 simulations that have been executed for each agenda. During such simulations, when tasks had a delay, a right-slide rescheduling policy was applied to accommodate the successor activities. On the contrary, no schedule correction was done when tasks finished earlier than predicted.

An analysis of the results presented in Table 1 reveals that, with the exemption of the C3-a example, all the solu-

tions obtained with the stochastic proposal exhibit a better behavior than the deterministic ones. The exception corresponds to a large-scale example which demands extra CPU time or a warm-start mode (initial solution) to render a good quality solution. Table 1 allows concluding that the deterministic solutions become more affected by the processing time uncertainty than the stochastic ones.

Furthermore, simulations results for the extreme values of the distributions, which are not discussed here due to space limitations, show a very robust behaviour and are of considerably better quality than the deterministic ones.

Figure 2 shows a plot of the total tardiness values associated with various instances of the C1 problem that correspond to increasing processing times variability scenarios, for both deterministic and stochastic methodologies. The stochastic approach is more stable, performing better than the deterministic one, especially for values of *sup* greater than 0.30. Figure 3 shows a similar behavior for the total start time delay performance indicator, which is the sum of the start time postponement with respect to the scheduled start time of all the tasks. Once again, the stochastic approach renders lower values of this indicator, which could be considered as a robustness measure.

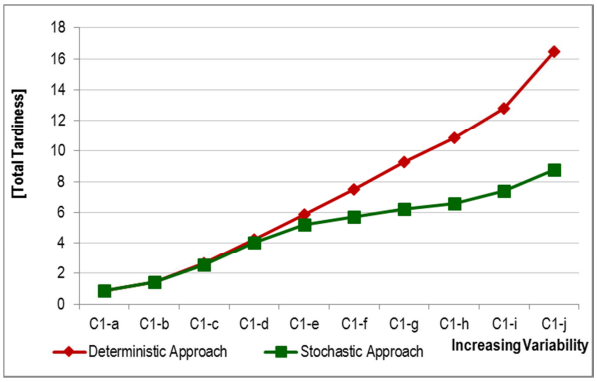

Figure 2. Case 1 total tardiness for increasing processing times variability scenarios

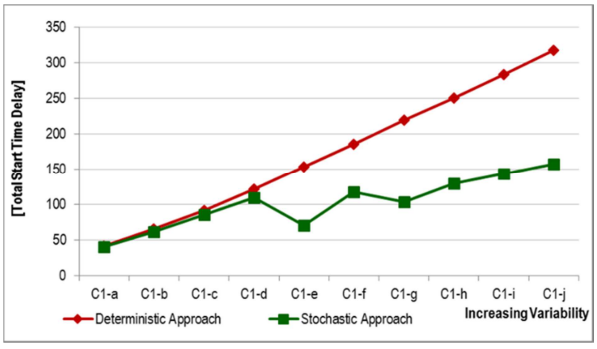

Figure 3. Case 1 total start time delay for increasing processing times variability scenarios

Finally, Figures 4 and 5 show the Gantt charts corresponding to example C1-h. Both associated agendas, the operational and the variance one, are displayed, showing that they have exactly the same structure.

Conclusions and future work

An innovative CP stochastic scheduling methodology was proposed to address processing time uncertainty proactively. It has been tested by means of several examples with a satisfactory CPU performance, since the approach does not have the computational load of most stochastic proposals. The attained operational schedules are more resilient and better prepared to absorb the effects of uncertainty. This conclusion was reached after performing simulations that compared the agendas obtained with this proposal and the ones reached with a deterministic CP model.

One of the inherent limitations of the proposal is the normal distribution assumption of processing times, as opposed to real industrial settings, where they are better modeled by asymmetric distributions. However, simulation tests, which employed these asymmetric distributions, rendered very good results. Future work will extend the approach to address other intermediate storage/inter-stage waiting policies: UIS (unlimited intermediate storage), NIS-ZW and NIS-FW (non-intermediate storage, zero and finite wait, respectively). The impact of a proper warm-start (good quality initial solution) will also be analyzed.

Acknowledgments

The authors wish to acknowledge the financial support received from CONICET (PIP 112 20110101145) and UNL (CAI+D 2011 50120110100473LI).

References

Balasubramanian, J., Grossmann, I.E. (2003). Scheduling optimization under uncertainty - An alternative approach. *Comput. Chem. Eng.*, *27*, 469–490.

Bonfill, A., Espuña, A., Puigjaner, L. (2008). Proactive approach to address the uncertainty in short-term scheduling. *Comput. Chem. Eng.*, *32*, 1689–1706.

Castro, P.M., Harjunkoski, I., Grossmann, I.E. (2009). Optimal short-term scheduling of large-scale multistage batch plants. *Ind. Eng. Chem. Res.*, *48*, 11002–11016.

ILOG (2013). CPLEX - IBM ILOG - 12.5.1 http://www.ibm.com/support/knowledgecenter/SSSA5P_12.5.1/ilog.odms.studio.help/Optimization_Studio/topics/COS_home.html?lang=es.

Gupta, D., Maravelias, C., Wassick, J. (2016) From Rescheduling to Online Scheduling. *Chem. Eng. Res. Des.* In Press.

Li, Z., Ierapetritou, M.G. (2008). Robust optimization for process scheduling under uncertainty. *Ind. Eng. Chem. Res.*, *47*, 4148–4157.

Marchetti, P. A., Cerdá, J. (2009). A general resource-constrained scheduling framework for multistage batch facilities with sequence-dependent changeovers. *Comput. Chem. Eng.*, *33*, 871–886.

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. *Comput. Chem. Eng.*, *93*, 101–117.

Zeballos, L.J., Novas, J.M., Henning, G.P. (2011). A CP formulation for scheduling multiproduct multistage batch plants. *Comput. Chem. Eng.*, *35*, 2973–2989.

Table 1. Comparison of deterministic and stochastic approaches by means of simulation

Case study	inf	sup	Stochastic schedule execution in a simulated environment**					Deterministic schedule execution in a simulated environment				
			Total Tardiness	*Tardy orders*	*Makespan*	*Idle-time*	*Total start time delay*	*Total Tardiness*	*Tardy orders*	*Makespan*	*Idle-time*	*Total start time delay*
C1-a	0.05	0.12	0.9	1	132.3	285.3	40.7	0.9	1	129.6	228.6	42.0
C1-b	0.075	0.18	1.4	1	128.6	270.2	62.0	1.4	1	130.6	233.7	66.1
C1-c	0.1	0.24	2.6	2	133.8	294.9	85.3	2.7	2	131.7	239.8	92.0
C1-d	0.125	0.3	4.0	2	133.0	288.2	1,9.7	4.2	2	132.9	247.1	122.0
C1-e	0.15	0.36	5.2	1	134.3	315.7	70.8	5.9	2	134.2	255.2	153.4
C1-f	0.175	0.42	5.7	1	139.7	265.9	117.6	7.5	2	135.5	262.9	185.1
C1-g	0.2	0.48	6.2	1	130.1	296.0	103.3	9.3	2	136.9	271.9	219.5
C1-h	0.225	0.54	6.6	1	138.5	310.3	129.7	10.9	2	138.3	279.9	250.1
C1-i	0.25	0.6	7.4	2	129.3	279.3	144.0	12.8	3	139.6	288.5	283.7
C1-j	0.275	0.66	8.8	3	140.1	359.1	157.3	16.5	5	141.0	297.1	317.8
C2-a	0.2	0.5	39.6	2	347.0	2311	470.3	71.4	4	346.1	2627	518.7
C2-b*	0.2	0.5	6.0	2	344.0	2376	312.2					
C3-a	0.07	0.2	10.101	5	62.161	293.295	306.621	7.338	4	61.933	301.931	235.093
C3-b*	0.07	0.2	6.726	5	61.130	305.578	251.060					

*Warm start is employed – 4.500 CPU seconds– Computer: Notebook Asus X555LAB, Intel Core i7-5500U processor, 8GB Ram Memory

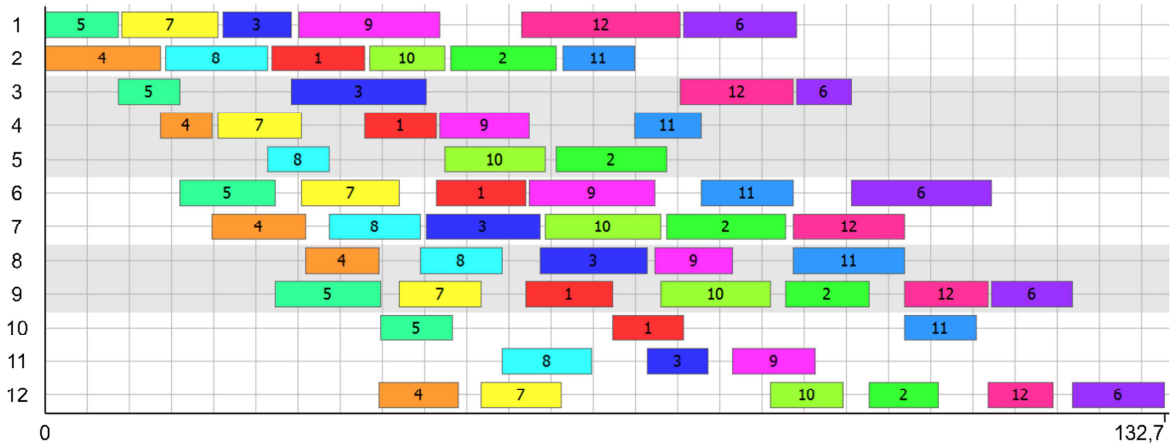

Figure 4. Example C1-h operational schedule

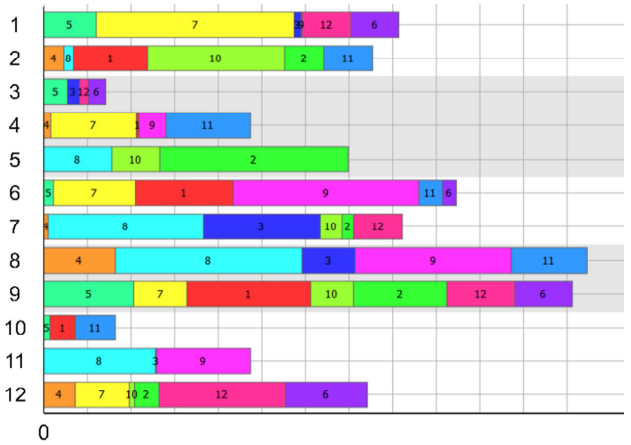

Figure 5. Example C1-h variance schedule