HAL
open science

Project Scheduling Under Resource Constraints: Application of the Cumulative Global Constraint in a Decision Support Framework

Mariem Trojet, Fehmi H'Mida, Pierre Lopez

To cite this version:

Mariem Trojet, Fehmi H'Mida, Pierre Lopez. Project Scheduling Under Resource Constraints: Application of the Cumulative Global Constraint in a Decision Support Framework. Computers & Industrial Engineering, 2011, 61 (2), p. 357-363. 10.1016/j.cie.2010.08.014 . hal-00491796

HAL Id: hal-00491796

https://hal.science/hal-00491796

Submitted on 14 Jun 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Submitted to *Computers and Industrial Engineering*

Project Scheduling Under Resource Constraints: Application of the Cumulative Global Constraint in a Decision Support Framework

Mariem Trojet[1], Fehmi H'Mida[1], Pierre Lopez[2,3]

[1] UR MSSDT ESSTT, Tunis - Tunisia (mtrojet@laas.fr, fehmi.hmida@esti.rnu.tn)
[2] CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse, France (lopez@laas.fr)
[3] Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France

ABSTRACT

This paper concerns project scheduling under resource constraints. Traditionally, the objective is to find a unique solution that minimizes the project makespan, while respecting the precedence constraints and the resource constraints. This work focuses on developing a model and a decision support framework for industrial application of the cumulative global constraint. For a given project scheduling, the proposed approach allows the generation of different optimal solutions relative to the alternate availability of outsourcing and resources. The objective is to provide a decision-maker an assistance to construct, choose, and define the appropriate scheduling program taking into account the possible capacity resources. The industrial problem under consideration is modeled as a Constraint Satisfaction Problem (CSP). It is implemented under the constraint programming language CHIP V5. The provided solutions determine values for the various variables associated to the tasks realized on each resource, as well as the curves with the profile of the total consumption of resources on time.

Keywords: Production scheduling, constraint satisfaction problem, constraint programming, cumulative global constraint.

[*] Corresponding Author: Mariem Trojet, Phone: (+216) 98 287 375, E-mail: mtrojet@laas.fr

1. INTRODUCTION

In the context of operation scheduling, the type of resource is traditionally considered to be *a machine* that can perform at most one operation at a time (disjunctive resource). A natural extension of the basic problems involves the presence of *additional resources*, where each resource has a limited capacity (cumulative resource) and each operation requires the use of a part of each resource during its execution. This leads us to the area of Resource-Constrained Project Scheduling Problem (RCPSP) which consists in scheduling activities on renewable resources available in limited quantities. A classical objective function consists in optimizing the end date of the project, while satisfying at the same time the precedence constraints between activities and the resource constraints, that is, at every time, the sum of the resource consumptions for the activities in process should not exceed the resource capacity. Also, Blazewicz *et al.* (1983) proved that the RCPSP, as a generalization of the job shop scheduling problem, is NP-hard in the strong sense.

To solve the RCPSP, various types of methods have been used: linear programming, constraint programming, heuristics and metaheuristics, and tree search. For applications and classification of the RCPSP and its extensions, we refer to the surveys written in (Herroelen *et al.*, 1998; Brucker *et al.*, 1999; Kolisch and Padman, 2001; GOThA, 2006; Özdamar and Ulusoy, 1995).

The constraint programming paradigm has been designed to express and solve problems involving constraints. That consists in finding among the possible values of a set of variables those which satisfy all the constraints simultaneously. Several published research (Baptiste and Le Pape, 1997; Beldiceanu *et al.*, 1996; Dorndorf *et al.*, 1999) deal with the resolution of the RCPSP using constraint programming.

The aim of this paper is to provide the decision-maker more flexibility and reactivity to define a scheduling program. Thus, we propose a decision support framework fitted to an industrial context.

The scheduling model developed integrates the application of the global cumulative constraint. It provides the optimal end date schedules for each capacity scenario, delivered in numerical form and in the form of curves tracing the profile of the total consumption of resources over time.

This paper consists of four parts. In the first part (Section 2), we recall the basis principles of constraint programming and of the cumulative global constraint concept. Section 3 describes the industrial problem and formulates the scheduling model under resource constraints. On the basis of this model, Section 4 develops the decision support framework. In Section 5, the industrial case is used to implement, under the constraint programming language CHIP V5, the feasibility of the proposed approach. Finally, Section 6 is devoted to the concluding remarks and some future research directions.

2. CONSTRAINT PROGRAMMING

Constraint programming refers to the techniques dealing with constraint representation and exploitation. This paradigm combines methods of operations research (*e.g.*, graph theory, mathematical programming, combinatorial optimization methods) with tools resulting from artificial intelligence (*e.g.*, filtering algorithms, instantiation heuristics, search schemes). Research carried out on constraint satisfaction problems (CSPs) has resulted in the development of effective models which are now widely used in various domains such as computer vision, robot or agent planning, scheduling, human resources management, design, agronomy, diagnosis, or others (Gyssens *et al.*, 1994; Van Hentenryck *et al.*, 1992; Vargas, 1995).

2.1. The CSP formalism

A CSP is defined as a triplet *(X, D, C)* (Montanari, 1974; Dechter, 2003) with:

• $X = (X1, X2, ..., Xn)$ is the set of variables of the problem;

• $(D = D1, D2, ..., Dn)$ is the set of domains. Each variable $Xi \in X$, $1 \leq i \leq n$, is associated with one domain $Di \in D$ which represents all the possible values for Xi. These domains are finite, but of any kind, symbolic or numerical;

• $C = (C1, C2, ..., Ck)$ is the set of constraints. Each constraint Cj is a relation between some variables of X. These constraints are of any kind, linear (*e.g.*, $X1 + X2 \leq 4$) or non-linear (*e.g.*, $X3 \neq X4$).

Given a CSP (X, D, C), its resolution is to instantiate all the variables, so that all constraints are simultaneously satisfied.

2.2. CSP solving

General CSPs belong to the class of NP-complete problems. Their solving is based on the application of constraint propagation techniques (filtering phase) and on tree search (resolution phase).

Filtering phase: It consists in removing the values of the variables which have no chance to be among a solution. Within the filtering algorithms, the arc-consistency (AC) is probably the most used. This algorithm checks the consistency of a constraint between two variables of a CSP. Since the seminal AC-3 (Mackworth, 1977), many more powerful versions have been proposed (Mohr and Henderson, 1986; Bessière, 1994). However, the easy implementation of AC-3, its adaptability to broader frameworks than the classical CSP, as well as recent improvements of the basic version (Zhang and Yap, 2001; Bessière *et al.*, 2005) make this algorithm a widely used filtering technique.

Resolution phase: It consists in finding a complete instantiation (a solution) respecting all constraints. The resolution is carried out by means of various algorithms based on tree search (*e.g.*, Backtrack (Bitner and Reingold, 1975), Limited Discrepancy Search (Harvey and Ginsberg, 1995), Randomization & Restart (Gomes *et al.*, 1998). Some of them are hybrid algorithms in the sense

they perform a certain level of filtering on each variable instantiation in the tree expansion (Forward-Checking, Real-Full-Look-Ahead (Nadel, 1989), Maintaining Arc-Consistency (Sabin and Freuder, 1994).

Last, let us mention that scanning the search space can be improved by ordering heuristics (order of the variable instantiations and choice of a value for a given variable).

2.3. The global cumulative constraint

Combinatorial problems generally present independent substructures easily identifiable, all of which being formulated by a group of constraints. This is the reason for introducing the concept of "*global constraint*". A global constraint is a subset of constraints, corresponding to a substructure of the original problem. Several types of global constraints have been developed: alldifferent, diffn, cycle, sequence, cumulative, etc. In the example of the global constraint "alldifferent $(X1, ..., Xn)$", each of the variables $X1, ..., Xn$ must take a different value, *i.e.*, $X1 \neq X2 \neq .. \neq Xn$. To each global constraint, at least one specific filtering algorithm is associated. A global constraint takes into account the group of constraints as a whole, in a more effective manner than standard propagation techniques applied to separate constraints. In this work, we apply the cumulative global constraint:

$$Cumulative\ ([S_1, ..., S_n], [p_1, ..., p_n], [E_{1,k}, ..., E_{m,k}], [r_{1,k}, ..., r_{m,k}], R_k, Z_1).$$

A set of n tasks (activities) has to be scheduled; S_i is the start time of task i, p_i its processing time, $E_{i,k}$ the *energy required*, $r_{i,k}$ the number of units of resource k needed for its execution, R_k the number of resource k available at every time.

The origin of cumulative global constraint results from the work of Lahrichi (1982). The associated filtering algorithms (Caseau, 1994; Hooker, 2000; Baptiste *et al.*, 2001) were the topic of advanced and varied studies. They are, moreover, in constant improvement. The cumulative global constraint found in the RCPSP a particularly favorable field of application. It consists in the evaluation, for a

given interval, of the number of resources required to perform the activities; if this number exceeds the number of available resources then a contradiction is detected.

3. THE SCHEDULING MODEL UNDER RESOURCE CONSTRAINTS

The RCPSP under study is composed of m resources. As already mentioned, R_k is the number of resource k available. A set of n tasks is to be scheduled. Each task i has a duration p_i per unit of resource and requires for its realization a number $r_{i,k}$ of resource k.

We distinguish the time constraints that are mainly defined by the precedence relations between tasks, and the resource constraints which require that, at every time and for each resource, the total demand does not exceed the availability.

The proposed model consists in building the production planning starting from the identification of the parameters, the variables and the constraints translating the various characteristics related on times, the products and the physical system. A solution consists in assigning a start time to each task by satisfying all the constraints.

3.1. Industrial problem description

The study is related to a workshop composed of 4 sections: cutting, assembly, painting, and finishing. The various products are prepared in the cutting section, then transferred to the assembly section. Once the products are assembled, they are transferred to the painting section and finally to the finishing section. The passing of products through the cutting machines depends on their specific process plan. The overall flow of production is shown below.

<Figure 1 to be inserted here>

In this study, we are particularly interested to define a scheduling program under resource constraints in the cutting section. This section contains 7 resource types. For the 11 products traversing the cutting section, we have finally 33 different tasks to process.

- *Elementary period of time t*. It is the unit of time in terms of scheduling,
- *Decision horizon H, $t \in [1, H]$*. It is the period over which the scheduling considers data production and makes decisions,
- *The set of products P*: *P = (A, B, C, D, E, F, G, M, W, Y, K),*
- *The set of tasks T: T = (A1, A4, A5, A7, B1, B7, C1, D1, D4, D5, D7, E3, E4, E5, E7, F3, F7, G1, M3 , M4, M5, M7, W1, W4, W5, O3, O4, O5, O7, K3, K4, K5, K7),*

  The achievement of each product requires the processing of a set of tasks. Each task is performed on a different resource. Example: *Product B requires the realization of 2 tasks: (B1, B7). B1 and B7 are respectively carried out on resource 1 and resource 7.*
- *Energy required* $E_{i,k}$: It denotes the energy required to perform the task *i* on resource *k.*
- *Resource capacity* R_k: It represents the number of each resource *k* available during periods *t*.

  $R_1 = 4 / R_2 = 2 / R_3 = 4 / R_4 = 3 / R_5 = 3 / R_6 = 1 / R_7 = 3.$

3.2. Variables

The variables involved in the model are:

- *Start time* S_i : corresponds to the start time of task *i*.
- $r_{i,k}$: corresponds to the number of resource *k* allocated to task *i*.
- *Duration* p_i : corresponds to the duration of task *i*.
- *Completion times* Z_k : corresponds to the completion times of all tasks on resource *k*.

$$Z_k = \max_{i,k} (S_i + p_i) \quad (1)$$

Example:

$$Z_1 = \max (S_{A1} + p_{A1}, S_{B1} + p_{B1}, S_{C1} + p_{C1}, S_{D1} + p_{D1}, S_{G1} + p_{G1}, S_{W1} + p_{W1}).$$

- *Total completion time Z (makespan)*: it is the completion time for all tasks on all resources.

$$Z = \max (Z_k) \quad (2)$$

The objective is to determine a schedule with minimum makespan *Z*.

3.3. Constraints

In this section, we present the three types of constraints to satisfy, in order to produce feasible solutions:

- *Task energy constraints*:

$$p_i * r_{i,k} = E_{i,k} = Constant \quad (3)$$

This corresponds to the *energy* consumption of a task on a resource (Lopez *et al.*, 1992). Figure 2 shows an example of a task *A1* with $E_{i,k} = 6$. In this case, the possible values of p_i and $r_{i,k}$ are: *((2.3), (3.2), (6.1))*. The solution *(2.3)* means that task *i* requires *3* resources during *2* time units.

<Figure 2 to be inserted here>

- *Capacity resource constraints*: For each resource *k*, they express the fact that, at every time, the total number of resources used by a set of tasks processed at time t $(S_i \leq t \leq S_i + p_i)$ does not exceed a certain capacity R_k. It is used to model the cumulative resource constraint. For a given scheduling, the whole tasks that consume resource *k* at *t* are such that:

$T_k(t) = \{i\}_{i=1..n}$ with $t \in [t_i, t_i + p_i[$

$$\sum_{i \in T_k(t)} r_{i,k} \leq R_k \quad (4)$$

Precedence Constraints: The precedence constraints between tasks are of two types. Those corresponding to the process plan of a product and those from the delivery of some products. They have the form $i \rightarrow j$, prohibiting the start of the second task, *j*, before the end of the first, *i*.

Process plan constraints: for each product a process plan sets a sequence of tasks necessary for its realization.

$$S_j \geq S_i + p_i \quad (5)$$

For example, the process plan of product *A* is: *(A1, A4, A5, A7).* The associated precedence constraint states: $A4 \rightarrow A5$: $S_{A5} \geq S_{A4} + p_{A4}$.

Delivery constraints: some products must be processed before others (constraints imposed by the order of delivery products). These inter-products constraints are expressed as follows:

$$G \rightarrow E: S_{E3} \geq S_{G1} + p_{G1} \quad (6)$$

4. DECISION SUPPORT FRAMEWORK

The objective of the proposed model is to find an optimal solution that minimizes the project makespan. However, if the resolution did not generate any solution respecting the delivery deadline, the project manager may modify his problem and then consider some constraint relaxations. These relaxations will be taken into account in the resolution phase with the display of a new solution. If the project manager accepts this solution, the modifications realized will be considered, otherwise and if it is possible, other relaxations are carried out (Lizarralde, 2007).

Considering the problem of resource capacity, the project manager will be able, and according to the availability of subcontracting for each type of resource, to relax the corresponding cumulative constraint (Figure 3). Thus, he can classify the cumulative constraints by ascending order of importance. This process of decision support is presented in the following figure.

<Figure 3 to be inserted here>

The relaxation is performed on the first constraint on the list until the definition of a consistent solution. When a feasible solution is found, the project manager will undertake the necessary action for the new scheduling program. If the new solution is not accepted, the planned relaxation should be reduced and the process continues by the relaxation of the next cumulative constraint until the definition of an optimal scheduling program taking into account the possibilities of outsourcing.

5. EXPERIMENTS

5.1. Model implementation with CHIP V5

Various propagation and resolution algorithms of the CSP have been integrated into Constraint Programming languages. Many environments have been developed and distributed. Among these environments, some are commercial (*e.g.*, CHIP, ILOG-Solver), other are academic tools (*e.g.*, Oz, Gnu-Prolog, CIAO SICStus). The scheduling model we developed was implemented with CHIP V5. The programming consists to declare the domain variables (decision), to post the constraints, and finally to enumerate and/or optimize the solutions. The variables are declared as domain variables that take their values in finite sets of integers. The constraints implemented are of two types. The first, relative to precedence constraints, are written in form of arithmetic linear constraints. The second *cumulative global constraint* provides a significant level of abstraction. It allows the modeling of a set of classical constraints in a more concise way. Therefore, it was used to integrate two constraint types: the cumulative constraints and the energy consumptions.

Thus, we associated to each resource a corresponding cumulative global constraint. We present this mechanism below relating to the tasks processed on resource 1.

Cumulative $(S_i, p_i, E_{i,1}, r_{i,1}, R_1, Z_1)$

The variables: $S_i = [S_{A1}, S_{B1}, S_{C1}, S_{D1}, S_{G1}, S_{W1}]$, $p_i = [p_{A1}, p_{B1}, p_{C1}, p_{D1}, p_{G1}, p_{W1}]$, $E_{i,1} = [E_{A1,1}, E_{B1,1}, E_{C1,1}, E_{D1,1}, E_{G1,1}, E_{W1,1}]$ corresponding respectively to start times S_i, durations p_i, and energy required $E_{i,1}$ of tasks *A1, B1, C1, D1, G1, W1*. The cumulative constraint imposes to satisfy the number of resources $r_{i,1}$ consumed $r_{i,1} = [r_{A1,1}, r_{B1,1}, r_{C1,1}, r_{D1,1}, r_{G1,1}, r_{W1,1}]$ respective to tasks *A1, B1, C1, D1, G1, W1* and the capacity of resource R_1. Z_1 is relative to the calculation of the completion time of the five tasks on resource 1.

According to its commercial description, the processing of cumulative global constraints in CHIP V5 is performed by about twenty of methods. Among the known and effective methods, we can particularly note the energy reasoning and the timetable constraints. The first allows, on the basis of

balance between the consumptions of a resource by activities over a time interval and the energy offered by this resource on the same interval, to determine a lower bound on the amount of the resource which can be used (Lopez *et al.*, 1992). The second has the same goal, but it gets on the basis of constraints called timetable, based on the concept of mandatory part (Lahrichi, 1982), interval of time when the task is necessarily carried out.

5.2. Illustration of the decision support

For any solution, the visualization of the consumption on each resource is information making it possible to follow the satisfaction of cumulative global constraints. The presentation of the results wants to be didactic; we then present as an example only the cumulative curve of resource 1. The curve and analysis presented in this section relate to the optimal solution.

The capacity of resource 1 is equal to 4. It realizes the 6 tasks: *A1, B1, C1, D1, G1,* and *W1*. The following table presents the values of the optimal solution.

<Table 1 to be inserted here>

The cumulative curve of resource 1 is presented in Figure 4. The X-axis presents the time (in hours) and the Y-axis the number of resources. The profile visualizes the evolution of resource consumption over time. The horizontal line indicates the maximum capacity of the resource ($R_1=4$).

<Figure 4 to be inserted here>

The curve is composed by stacked rectangles associated to tasks *A1, G1, W1, B1, D1,* and *C1* realized on resource 1. Each rectangle is characterized by one duration p_i and a number of consumed resource $r_{i,k}$. The combination of these rectangles gives the profile of the total consumption over time. It corresponds to the distribution presented in Figure 5.

<Figure 5 to be inserted here>

To allow the manager to choose an optimal solution favoring the maximal and balanced occupation of the resources, the cumulative curve indicates, with a vertical white line (see Figures 4 & 5), the possibility to smooth the resource assignment without exceeding available resources capacity and minimum makespan Z found for the scheduling. In the case of resource 1, task *C1* presents this alternative provided to the manager. Indeed, the processing of task *C1* offers the possibility to choose one or two resources consumption, equivalent to Z_{1max} = 53 and Z_{1min} = 41:

- *If C1* consumes 1 resource:

  then $r_{C1,1} = 1 \rightarrow E_{C1,1}$ = 24h $\rightarrow S_{C1} = 29 \rightarrow p_{C1}$ = 24h $\rightarrow Z_1$ = 53h (Figure 5);

- *If C1* consumes 2 resources:

  then $r_{C1,1} = 2 \rightarrow E_{C1,1}$ = 24h $\rightarrow S_{C1} = 29 \rightarrow p_{C1}$ = 12h $\rightarrow Z_1$ = 41h (Figure 6).

<Figure 6 to be inserted here>

The cumulative curve in Figure 6 depicts the second alternative corresponding to the choice of two resources (p_{C1} = 12), and the distribution of all the tasks on resource 1 for this alternative.

Another interesting issue would be to consider the resource flexibility in project scheduling, which could permit to propose multiple assignments and switch possibilities. To that aim, one way is to consider tasks that may be processed according to several modes; the modes determine different amounts of resources required and corresponding processing time. An alternative, which occurs when considering staff members, is to associate the notion of skill to resources. Since this is not the subject of our paper, we will not discuss it much further. We refer the interested reader to (Bellenguez-Morineau and Néron, 2008; Heimerl and Kolisch, 2010) to know more about Multi-Mode and Multi-Skill RCPSPs.

In our industrial case, the problem is solved to obtain the optimal solution under internal resource constraints. The results for resource 1 are presented above ($Z_{1max} = 53$, $Z_{1min} = 41$). The decision support framework presented in Section 4 provides more flexibility and reactivity to find the appropriate scheduling program. In practical situations, this optimal solution was generally used as

a starting point of an appropriate solution. In the case where the delivery deadline could not be respected, the project manager can relax one by one (resource by resource) the cumulative constraints according an ordered list of outsourcing availabilities. The application of this procedure for the case problem at resource 1 level, passing from a capacity of 4 to a capacity of 5 (outsourcing one unit capacity), provides the cumulative curve of resource presented in Figure 7 ($Z_{1max} = 48$, $Z_{1min} = 36$).

<Figure 7 to be inserted here>

To reduce the total completion time Z, the project manager can simulate the outsourcing possibilities for the different resources. In real-world situations, the values of the capacity subcontracted can be adjusted subjectively based on the partnership with the subcontractors.

6. CONCLUSIONS

The proposed scheduling model is based on a constraint satisfaction approach. It formalizes a set of decision variables to be managed and a set of constraints to be satisfied. In this framework, we have implemented the concept of *cumulative global constraint*. With a concise formulation, it allowed to combine two types of constraints: the cumulative constraints of and the task energy equations. Thus, the solutions obtained take into account the existing alternatives to the duration p_i of a task i according to the number of resources $r_{i,k}$ used for its realization, while respecting the capacity limitation of resource R_k. It also provides a decision support framework under these constraints as a margin of cooperation/negotiation with subcontractors. The different elements presented in the model were implemented through the constraint programming system CHIP V5. The results are delivered in numerical form and in the form of curves tracing, for a given resource, the profile of the total consumption of resources over time.

Further works plan to take account of the dynamic features of scheduling. That leads us to adopt an approach which is based on the *Dynamic* Constraint Satisfaction Problem formalism. The objective

is to avoid starting again research from zero after each change of the problem. Moreover, the more or less precise knowledge of the decision variables implies their controllability. This leads us to consider *Conditional* Constraint Satisfaction Problems with variables and constraints dependent on a situation and having conditions of presence.

REFERENCES

1. Baptiste, P. & Le Pape, C., (1997). Adjustments of release and due dates for cumulative scheduling problems. *Proceedings of the International Conference on Industrial Engineering and Production Management*, Lyon, France.
2. Baptiste, P., Le Pape, C., & Nuijten, W., (2001). *Constraint-Based Scheduling: Applying Constraint Programming in Scheduling Problems*, Kluwer.
3. Beldiceanu, N., Bourreau, E., Rivreau, D., & Simonis, H., (1996). Solving resource-constrained project scheduling with CHIP. *Proceedings of the 5th International Workshop on Project Management and Scheduling* (pp. 35-38), EURO PMS, Poznan, Poland.
4. Bellenguez-Morineau, O., & Néron, E., (2008). Multi-mode and multi-skill project scheduling problem. In *Resource-Constrained Project Scheduling: Models Algorithms, Extensions and Applications*, C. Artigues, S. Demassey, E. Néron (Eds). ISTE/Wiley, 149-160.
5. Bessière, C., (1994). Arc-consistency and arc consistency again. *Artificial Intelligence*, 65(1), 179-190.
6. Bessière, C., Régin, J.-C., Yap R., & Zhang, Y., (2005). An optimal coarse-grained arc consistency algorithm. *Artificial Intelligence*, 165(2), 165-185.
7. Bitner, J. & Reingold, E., (1975). Backtracking programming techniques. *Communications of the ACM*, 18(11), 651-656.
8. Blazewicz, J., Lenstra, J. K., & Rinnooy Kan, A.H.G., (1983). Scheduling projects to resource constraints: Classification and complexity. *Discrete Applied Mathematics*, 5(1), 11-24.

9. Brucker, P., Drexl, A., Möhring, R., Neumann, K., & Pesch, E., (1999). Resource-constrained project scheduling problem: Notation, classification, models and methods. *European Journal of Operational Research*, 112(1), 3-41.

10. Caseau, Y., (1994). Constraint Satisfaction with an object-oriented knowledge representation language. *Journal of the Applied Intelligence*, 4(2), 157-184.

11. Dechter, R., (2003). *Constraint Processing*. Morgan Kaufmann, San Francisco.

12. Dorndorf, U., Huy Phan, T., & Pesch, E., (1999). A Survey of Interval Capacity Consistency Tests for Time- and Resource-Constrained Scheduling. In *Project Scheduling – Recent Models, Algorithms and Applications*, J. Weglarz (Ed). Kluwer, 213-238.

13. Gomes, C., Selman, B., & Kautz, H., (1998). Boosting Combinatorial Search Through Randomization. *Proceedings of the 15th National Conference on Artificial Intelligence* (pp. 431-437), Madison, WI, USA,.

14. GOThA, Working group on "Gestion de projet sous contraintes de ressources". *Problème de gestion de projet à contraintes de ressources : approches et méthodes de résolution.* http://www.ocea.li.univ-tours.fr/eocea/rcpsp, 2006 (in French).

15. Gyssens, M., Jeavons, P.G., & Cohen, D.A., (1994). Decomposing constraint satisfaction problems using database techniques. *Artificial intelligence*, 66(1), 57-89.

16. Harvey, W. & Ginsberg, M., (1995). Limited Discrepancy Search. *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (pp. 607-613), Montreal, Canada.

17. Heimerl, C. & Kolisch, R., (2010). Scheduling and staffing multiple projects with a multi-skilled workforce. *OR Spectrum*, 32 (2), 343-368.

18. Herroelen, W., De Reyck, B., & Demeulemeester, E., (1998). Resource-constrained project scheduling: A survey of recent developments. *Computers & Operations Research*, 25(4), 279-302.

19. Hooker, J., (2000). *Logic-Based Methods for optimization: Combining Optimization and*

Constraint satisfaction. Wiley, USA.

20. Kolisch, R. & Padman, R., (2001). An integrated survey of deterministic project scheduling. *Omega*, 29(3), 249-272.
21. Lahrichi, A., (1982). Scheduling: the notions of hump, compulsory parts and their use in cumulative problems. *C. R. Acad. Sci.*, Paris, 294, 209-211.
22. Lizarralde, I., (2007). *Steering engineering activities for the distributed development of a complex system*. PhD thesis, Institut National des Sciences Appliquées de Toulouse, Toulouse, France. http://tel.archives-ouvertes.fr/tel-00163853
23. Lopez, P., Erschler, J., & Esquirol, P., (1992). Ordonnancement de tâches sous contraintes : une approche énergétique. *Automatique, Productique, Informatique Industrielle*, 26, 453-481.
24. Mackworth, A. K., (1977). Consistency in networks of relations. *Artificial Intelligence*, 8(1), 99-118.
25. Mohr, R. & Henderson, T., (1986). Arc and path consistency revisited. *Artificial Intelligence*, 28(2), 225-233.
26. Montanari, U., (1974). Network of constraints: Fundamental properties and applications to picture processing. *Information Science*, 7, 95-132.
27. Nadel, B., (1989). Constraint Satisfaction Algorithms. *Computational Intelligence*, 5, 188-299.
28. Özdamar, L. & Ulusoy, G., (1995). A survey on the resource constrained project scheduling problem. *IIE Transactions*, 27, 574-586.
29. Sabin, D. & Freuder, E., (1994). Contradicting conventional wisdom in constraint satisfaction. *Proceedings of the 11th European Conference on Artificial Intelligence* (pp. 125-129), Amsterdam, The Netherlands.
30. Van Hentenryck P., Simonis, H., & Dincbas, M., (1992). Constraint satisfaction using constraint logic programming. *Artificial intelligence*, 58, 113-159.
31. Vargas, C., (1995). Modélisation du processus de conception en ingénierie dessystèmes

mécaniques : Mise œuvre basée sur la propagation de contraintes, Ph. D. Thesis. Ecole Normale Supérieure de Cachan.

32. Zhang, Y. & Yap, R., (2001). Making AC-3 an Optimal Algorithm. *Proceedings of the 17th International Joint Conference on Artificial Intelligence* (pp. 316-321), Seattle, WA, USA.

33. Gargeya, V. B. and Deane, R. H., (1996). Scheduling research in multiple resource constrained job shops: A review and critique. *International Journal of Production Research*, 34, 2077-2097.

Task i	Start times S_i	Energy required $E_{i,k}$	Task duration p_i	Resource number $r_{i,k}$
A1	6	24	8	3
B1	14	19	19	1
C1	29	24	24	1
D1	6	29	29	1
G1	1	15	5	3
W1	14	30	15	2

Table 1. Values of the variables tasks realized on resource 1 (optimal)

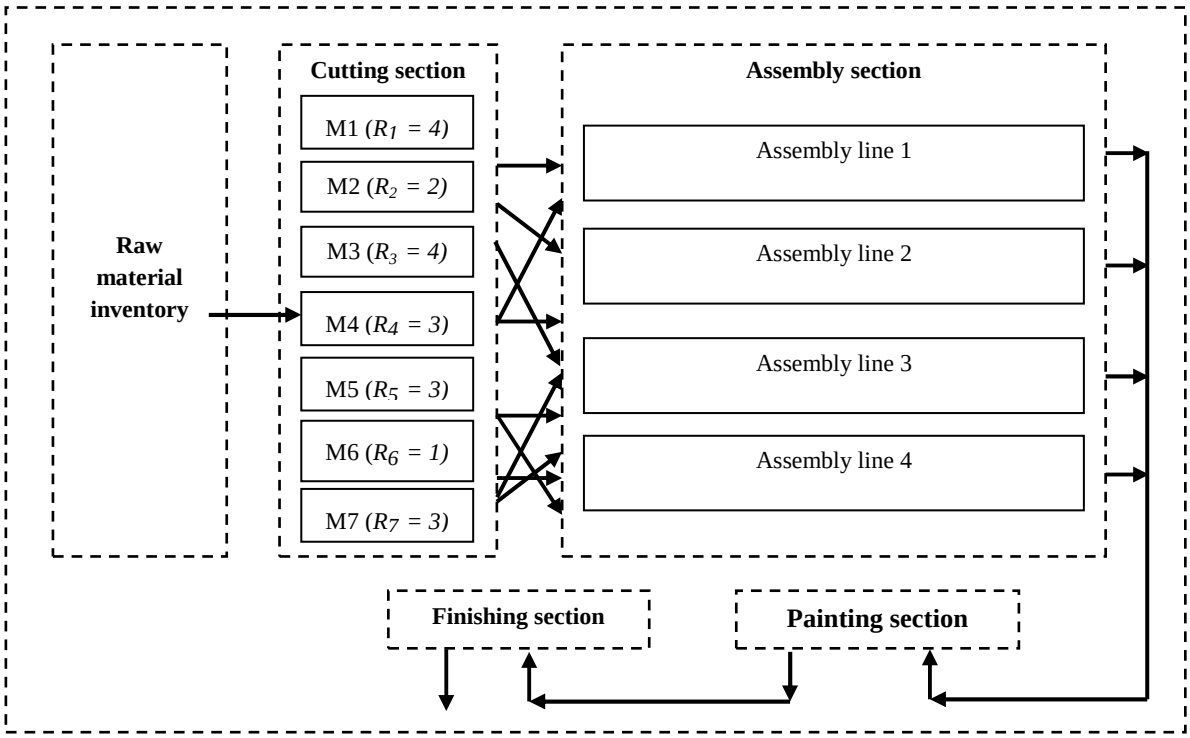

Figure 1. Overall flow of production

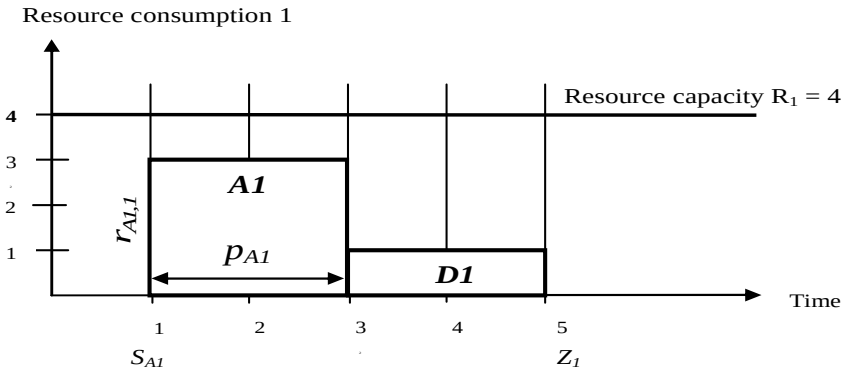

Figure 2. Example of a task *A1* ($E_{A1,1}$ = 6, p_{A1} = 2, $r_{A1,1}$ = 3)

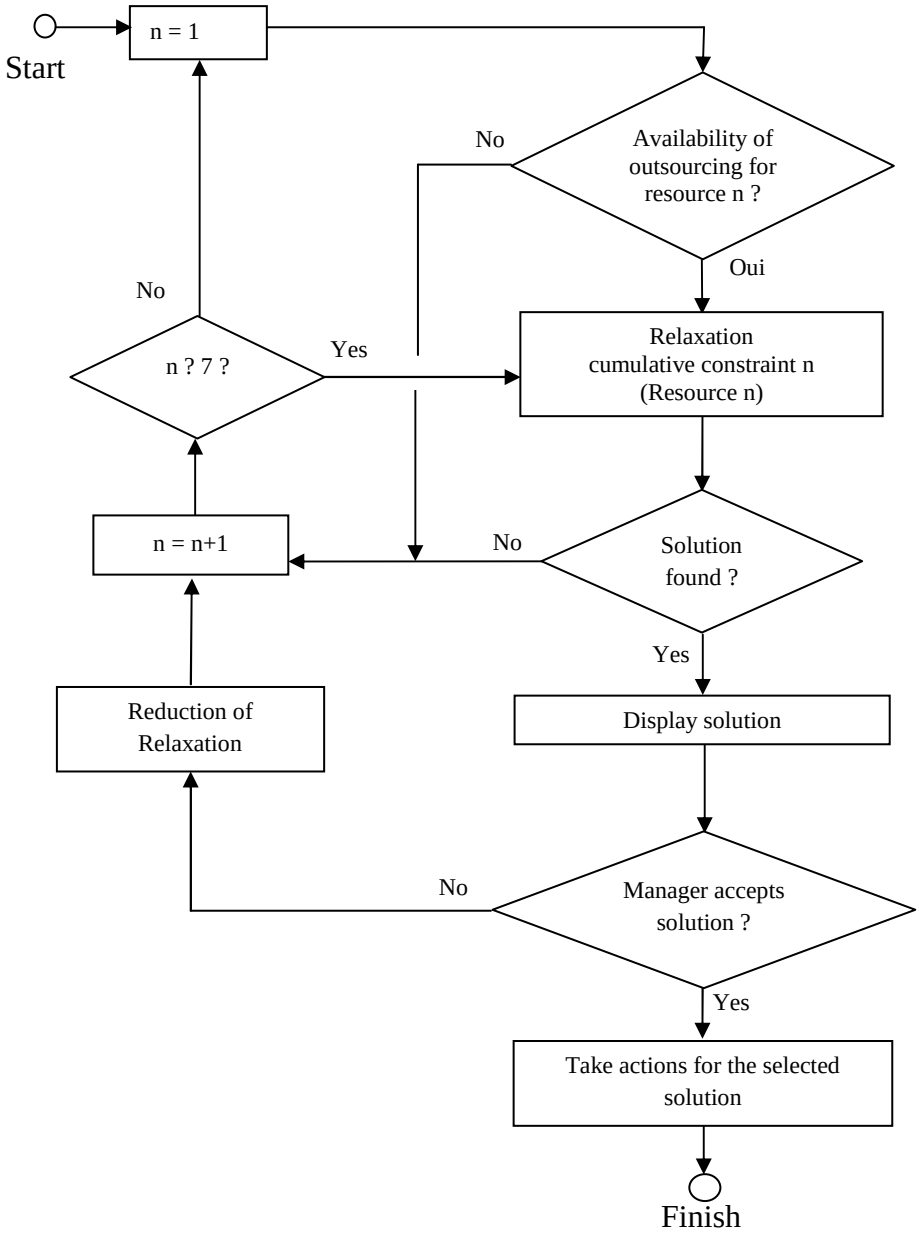

Figure 3. Decision support process

Figure 4. Cumulative curve resource 1 (*R1* = 4)

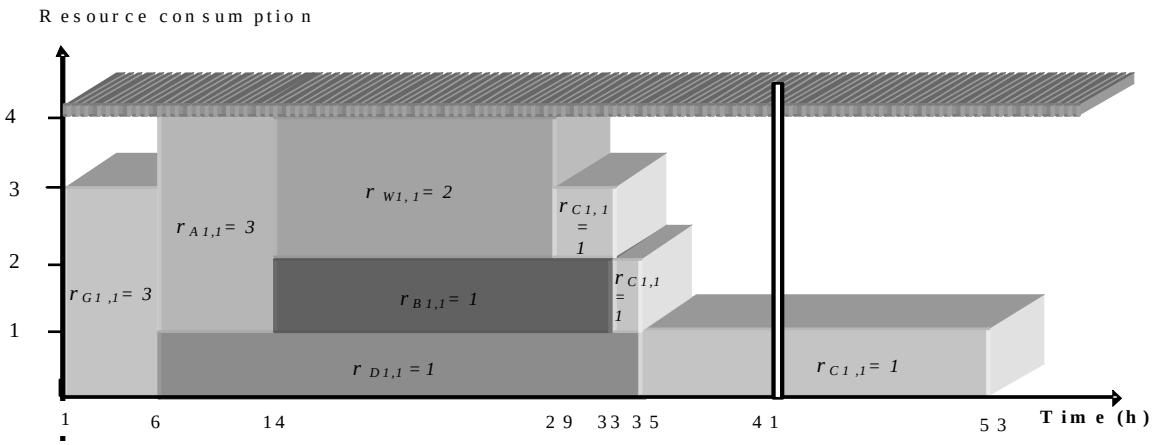

Figure 5. Detailed cumulative curve resource 1 (*R1* = 4, $r_{C1,1} = 1$)

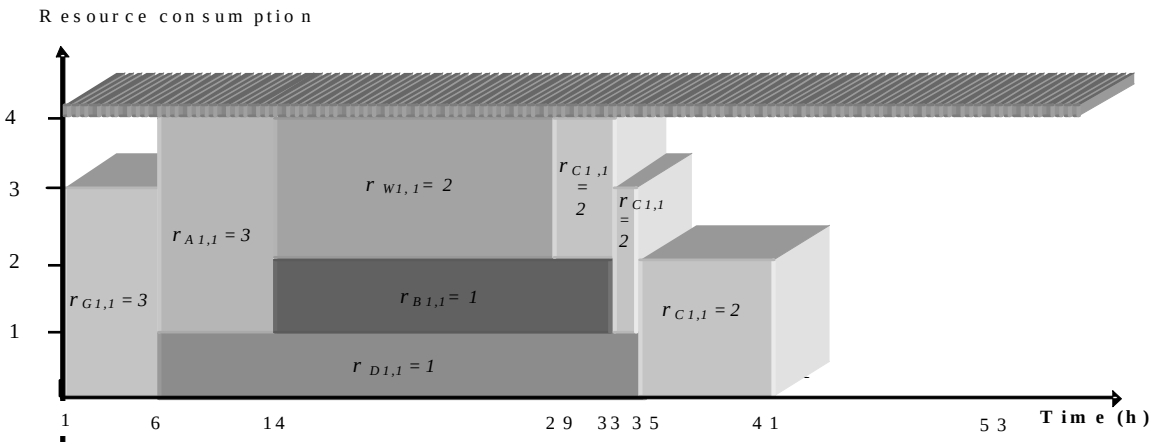

Figure 6. Detailed cumulative curve resource 1 (*R1* = 4, $r_{C1,1} = 2$)

Figure 7. Cumulative curve resource 1 (*R1* = 5)