Project Scheduling Under Resource Constraints: Application of the Cumulative Global Constraint

Mariem Trojet, Fehmi H'Mida, Pierre Lopez

Project Scheduling Under Resource Constraints: Application of the Cumulative Global Constraint

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

[1] UR MSSDT ESSTT, - Tunis - Tunisia (mariemtrojet@hotmail.com, fehmi.hmida@issatso.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. The objective is to find a solution that minimizes the project makespan, while respecting the precedence constraints and the resource constraints. The problem under consideration is modelled as a Constraint Satisfaction Problem (CSP). It is implemented under the constraint programming language CHIP V5. For modelling the resource constraints, we are particularly interested in the application of the cumulative global constraint. 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.

1. INTRODUCTION

The Resource-Constrained Project Scheduling Problem (RCPSP) 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 respecting at the same time the precedence constraints between tasks 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. To solve the RCPSP, various types of methods have been used: linear programming, constraint programming, heuristics and metaheuristics, and tree search ([7], [12], [21]).

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. As in [1], [3], [10], this paper deals with the resolution of the RCPSP using contraint programming. We propose a scheduling model under resource constraints based on the application of the global cumulative constraint proposed in some constraint programming languages like CHIP V5. We then provide schedules 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.

This paper consists of three parts. In the first part, we recall the basis principles of constraint programming and of the cumulative global constraint concept. The second part focuses on modelling the scheduling problem under study in the form of a constraint satisfaction problem. The third part presents an application developed under the constraint programming language CHIP V5.

2. CONSTRAINT PROGRAMMING

Constraint programming refers to the techniques dealing with constraint representation and exploitation. This paradigm combines methods of operational 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) have 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.

2.1. The CSP formalism

A CSP is defined as a triplet *(X, D, C)* [9, 19] 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 ($X1 + X2 \leq 4$) or nonlinear ($X3 \neq X4$).
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 [17], many more powerful versions have been proposed ([4], [18]). 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 ([5], [21]) 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 [6], Limited Discrepancy Search [13], Randomization & Restart [11]). 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 [20], Maintaining Arc-Consistency [22]).

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 sub-structures 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 sub-set of constraints, corresponding to a sub-structure 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.*, $x1 \neq x2 \neq \ldots \neq xn$. 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 separated constraints. In this work, we apply the cumulative global constraint:

Cumulative ($[S_1, ..., S_n]$, $[p_1, ..., p_n]$, $[R_1, ..., R_m]$, R_k).
A set of *n* tasks has to be scheduled; S_i is the start time of activity *i*, p_i its processing time, $r_{i,k}$ 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 ([15]). The associated filtering algorithms ([2], [8], [14]) were the topic of advanced and varied studies. They are, moreover, in constant improvement. The cumulative global constraint found in the RCPSP a particularly favourable 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. 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. Data

The study relates to 33 different tasks to process among a set of 7 resources.

- *Elementary period of time t.* It is the unit of time in terms of scheduling,
- *Decision horizon H, t∈[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 realisation 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.
- *Resource capacity* R_k: It represents the number of each resource *k* available during various 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 times* 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.*

  $Z_k = max_{i,k} (S_i + p_i)$ (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.

  $Z = max (Z_k)$ (2)

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

3.3. Constraints

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

- *Energy consumption*:

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

This corresponds to the *energy* consumption of a task on a resource [17]. Figure 1 shows an example of a task *A1* with a value of $R_{i,k} = 6$. 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.

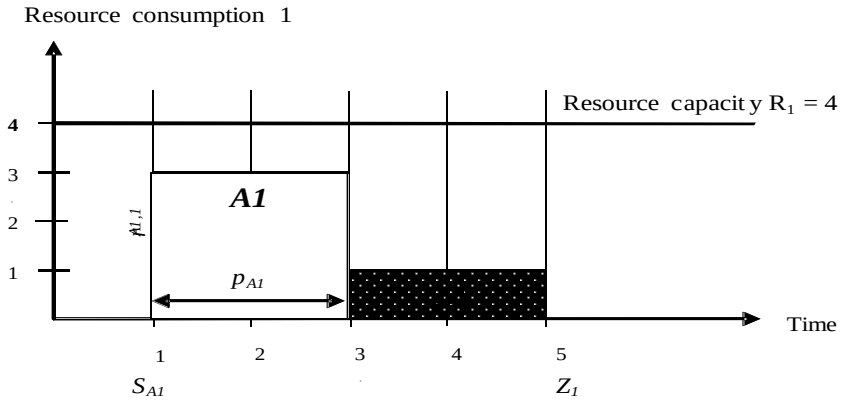

Figure 1. Example of a task A1 ($R_{A1,1} = 6$, $p_{A1} = 2$, $r_{A1,1} = 3$)

- *Cumulative 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* ($S_i \leq t \leq S_i + p_i$) do not exceed a certain capacity R_k. It is used to model the cumulative resource constraint.

  For $k = (1,...,7)$, $\forall t \in [\min (S_i), \max (S_i + p_i)]$

$$\sum_{S_i < t < S_i + p_i} r_{i,k} \leq R_k \quad (4)$$

- *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:

$$G \rightarrow E: S_{E3} \geq S_{G1} + p_{G1}.$$

4. EXPERIMENTS

4.1. Model implementation with CHIP V5

Different 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 modelling of a set of classical constraints in a more concise way. Thus, 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 (S1, P1, Q1, unused, r1, R1, Z1)

The variables*: S1 = [SA1, SB1, SC1, SD1, SG1, SW1], P1 = [PA1, PB1, PC1, PD1, PG1, PW1], R1 = [RA1, RB1, RC1, RD1, RG1, RW1]* corresponding respectively to start times S_i, durations p_i and resource numbers $r_{i,k}$ of tasks *A1, B1, C1, D1, G1, W1*. The cumulative constraint imposes to satisfy the number of resources $r_{i,k}$ consumed $r_1 = [r_{A,1}, r_{B,1}, r_{C,1}, r_{D,1}, r_{G,1}, r_{W,1}]$ respective to tasks *A1, B1, C1, D1, G1, W1* and the capacity of resource *R1*. *Z1* is relative to the calculation of the completion time of the five tasks on resource 1.

4.2. Illustration of global cumulative constraint

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 [16]. The second has the same goal, but it gets on the basis of constraints called timetable, based on the concept of mandatory part [15], interval of time when the task is necessarily carried out.

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

Task i	S_i Start times	$R_{i,k}$ Resource amount	p_i Duration task	$r_{i,k}$ Resource number
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

The cumulative curve of resource 1 is presented in Figure 2. 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 2. Cumulative curve resource 1

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 (resources) gives the profile of the total consumption over time (Figure 2). It corresponds to the distribution presented in Figure 3.

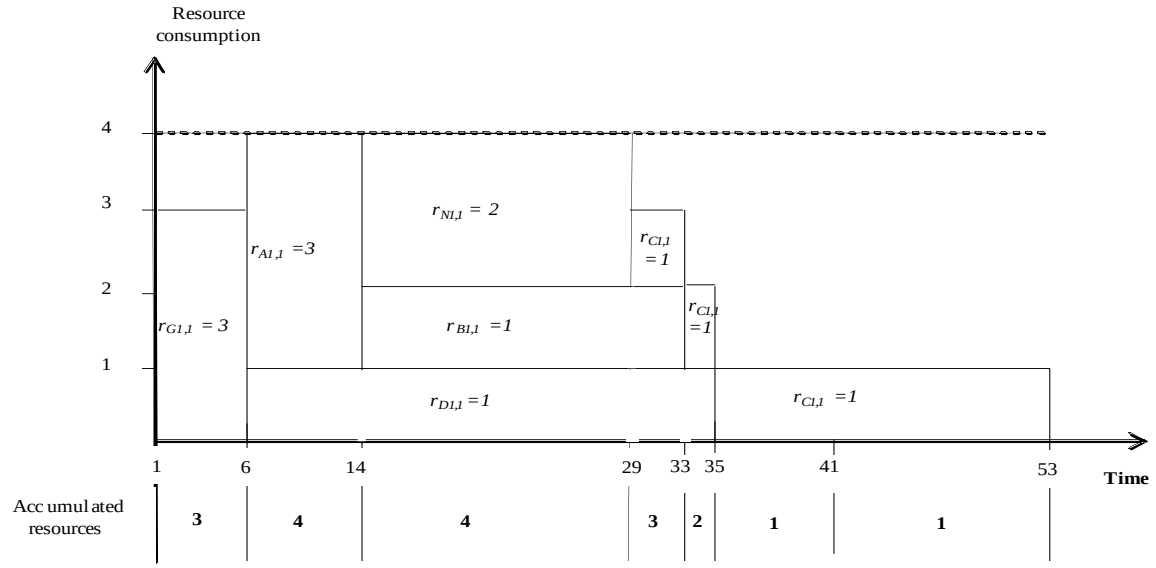

Figure 3. Detailed cumulative curve

Task *C1* presents a particular case. Indeed, without changing the optimal total completion date of planning the five resources $max\ (Z_k) = 78 = Z_5$, the processing of task *C1* offers the possibility to choose one or two resources:

C1 consumes 1 resource,
$R_{C1,1}$=24h→p_{C1}=24h→Z_1=53h<Z_5
C1 consumes 2 resources,
$R_{C1,1}$=24h→p_{C1}=12h→Z_1=41h<Z_5

By default, the proposed solution is that minimizing the number of resources (*C1* consumes 1 resource).

The cumulative curve in Figure 2 depicts the second alternative, by indicating the right vertical (Z_1 = 41) for the choice of two resources (p_{C1} = 12). In this case, the profile of the cumulative curve takes the following form (Figure 4):

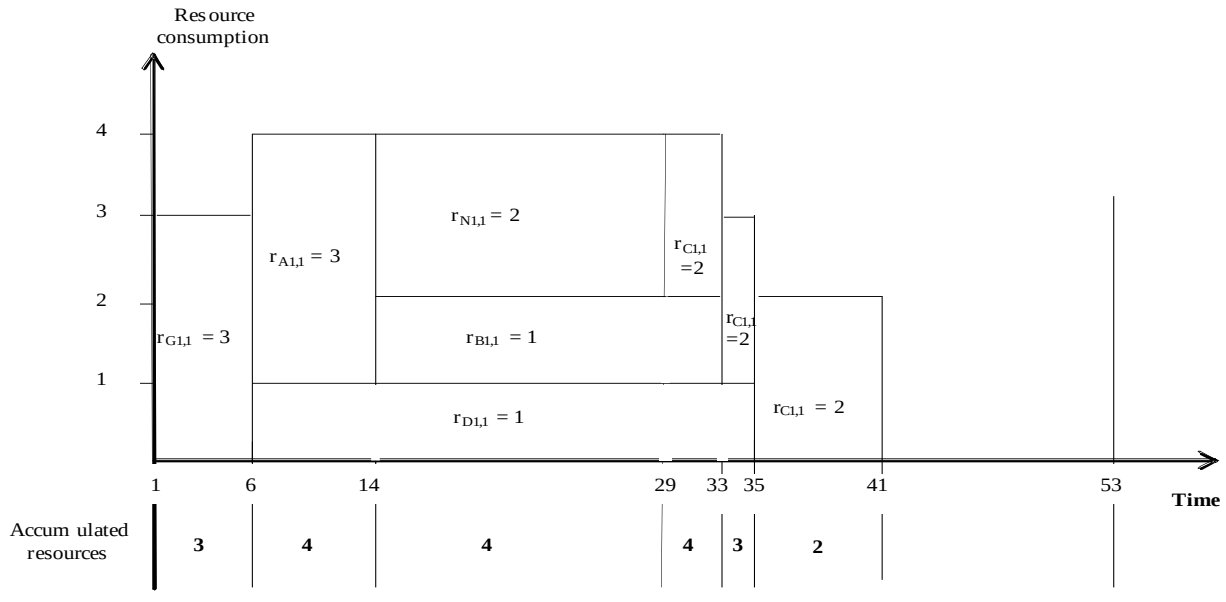

Figure 4. Cumulative curve for $R_{C1,1} = 2$

The consumption profile of a resource *k* presents the number of resources *k* occupied over all the period. The optimal scheduling solution obtained on the remainder involved resources offers for each of these resources a similar cumulative curve with a different profile.

5. CONCLUSION

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 energy consumptions. 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 the resource R_k. It also provides a decision-aid under these constraints as a margin of cooperation/negotiation with other decision centres.

The different elements presented in the model were implemented through the constraint programming system CHIP V5.

Further works plan to take account of the dynamic features of scheduling. It concerns environments where new constraints are imposed or former constraints are removed due to "*unforeseen*" from rush orders or cancelled, and breakdowns machines.

REFERENCES

[1] Baptiste, P. and C. Le Pape, "Adjustments of release and due dates for cumulative scheduling problems", *Proceedings of the International Conference on Industrial Engineering and Production Management*, Lyon, France, 1997.

[2] Baptiste, P., C. Le Pape and W. Nuijten, "*Constraint-Based Scheduling: Applying Constraint Programming in Scheduling Problems"*, Kluwer, 2001.

[3] Beldiceanu, N., E. Bourreau, D. Rivreau and H. Simonis, "Solving resource-constrained project scheduling with CHIP". *Proceedings of the 5th International Workshop on Project Management and Scheduling*, EURO PMS, Poznan, Poland, p. 35-38, 1996.

[4] Bessière, C., "Arc-consistency and arc consistency again", *Artificial Intelligence*, 65(1), p. 179-190, 1994.

[5] Bessière, C., J. Régin, R. Yap R and Y. Zhang, "An optimal coarse-grained arc consistency algorithm", *Artificial Intelligence*, 2005.

[6] Bitner, J. and E. Reingold, "Backtraking programming techniques". *Communications of the ACM*, 18(11), p. 651-656, 1975.

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

[8] Caseau, Y., "Constraint Satisfaction with an object-oriented knowledge representation language", *Journal of the Applied Intelligence*, p.157-184, 1994.

[9] Dechter, R., "*Constraint Processing"*, Morgan Kaufmann, San Francisco, 2003.

[10] Dorndorf, U., T. Huy Phan, and E. Pesch, "*A Survey of Interval Capacity Consistency Tests for Time- and Resource Constrained Scheduling"*, Project Scheduling - Recent Models, Algorithms and Applications, Kluwer, p. 213-238, 1999.

[11] Gomes, C., B. Selman, and H. Kautz, "Boosting Combinatorial Search Through Randomization", *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, Madison, WI, USA, p. 431-437, 1998.

[12] GOThA, Groupe 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.univtours.fr/eocea/rcpsp, 2006.

[13] Harvey, W. and M. Ginsberg, "Limited Discrepancy Search", *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (IJCAI-95), Montréal, Canada, p. 607-613, 1995.

[14] Hooker, J., "*Logic-Based Methods for optimisation: Combining Optimisation and Constraint satisfaction"*, Wiley, USA, 2000.

[15] Lahrichi, A., "Scheduling: the notions of hump, compulsory parts and their use in cumulative problems", *C. R. Acad. Sci.*, Paris, 294, p. 209-211, 1982.

[16] Lopez, P., J. Erschler, and P. Esquirol, "Ordonnancement de tâches sous contraintes : une approche énergétique", *Automatique, Productique, Informatique Industrielle*, 26:453-481, 1992.

[17] Mackworth, A. K., "Consistency in networks of relations", *Artificial Intelligence*, 8, p. 99-118, 1977.

[18] Mohr, R. and T. Henderson, "Arc and path consistency revisited", *Artificial Intelligence*, 28(2), p. 225-233, 1986.

[19] Montanari, U., "Network of constraints: Fundamental properties and applications to picture processing", *Information Science*, 7, p. 95-132, 1974.

[20] Nadel, B., « Constraint Satisfaction Algorithms », *Computational Intelligence*, 5, p. 188-299, 1989.

[21] Özdamar, L. and G. Ulusoy, "A survey on the resource constrained project scheduling problem", *IIE Transactions*, 27, p. 574-586, 1995.

[22] Sabin, D. and E. Freuder, "Contradicting conventional wisdom in constraint satisfaction", *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI-94)*, Amsterdam, The Netherlands, p. 125-129, 1994.

[23] Zhang, Y. and R. Yap, "Making AC-3 an Optimal Algorithm", *Proceedings of the 17th International Joint Conference on Artificial Intelligence* (*IJCAI-01*), Seattle, WA, USA, p. 316-321, 2001.