

Modelling Producer/Consumer Constraints

Helmut Simonis
COSYTEC SA
4, rue Jean Rostand
Parc Club Orsay Université
F-91893 Orsay Cedex
France

Trijntje Cornelissens
Beyers & Partners
Michielssendreef 42
B-2930 Brasschaat
Belgium

1. Abstract

In this paper we describe the modelling of producer/consumer constraints with the CHIP system. Producer/consumer constraints arise in scheduling problems with consumable resources like raw materials or money, in particular for batch based processing. The constraint assures that at each time point enough consumable resources are available. The modelling with CHIP uses the cumulative constraint to express conditions in a very declarative way, yet obtains very good propagation due to the reasoning power build into the cumulative constraint. We show that with producer/consumer constraints many resource scheduling problems can be easily expressed and give examples of its industrial use.

2. Introduction

Constraint logic programming (CLP) [FHK92][JM94] is increasingly used to solve hard scheduling and planning problems [BKC94] [BCP92] [BS92] [BDP94] [CDF94] [DS91] [Ev92] [FS90] [Wa94]. Special constraints like cumulative [AB93] and diffn [BC94] have been introduced to express constraints on renewable resources [Go92], like machines or personnel. These resources can be used over a time period up to a certain level of availability. In this paper we show how to model constraints on consumable resources with the same basic mechanisms. Consumable resources often are raw materials or amounts of money, which are available in limited amounts over certain time periods. A producer/consumer constraint on consumable resources expresses that at any time point more resources must have been produced than consumed. We use CHIP [DHS88] to express these constraints. CHIP is a constraint logic programming system originally developed at the ECRC in Munich [VH89][HSD92], and further developed by COSYTEC. We express producer/consumer constraints in the finite domain solver of CHIP, using the global cumulative constraint [AB93].

The paper is structured as follows:

In section 3, we introduce the producer/consumer constraint with several variations. Section 4 gives the declarative semantics of the constraint. Section 5 briefly introduces the cumulative constraint in CHIP. The next section describes the modelling of the producer/consumer constraint with the help of the cumulative. In section 7, we present different extensions of the constraint. Finally, we discuss the use of the constraint for the ATLAS scheduling system in section 8.

3. Producer/Consumer Constraints

Producer/consumer constraints arise in many different variants. We now discuss some typical examples.

3.1 Raw Materials

Consider a scheduling problem where each task requires certain amounts of different raw materials. The raw materials must be available at the beginning of each task in order to start the task. Some initial stock of the different raw materials is available, more becomes available with receivings of raw materials in given quantities at certain time points. The problem consists in scheduling the task in such a way that for each task all raw materials are available at its start. In this case, the production of the resource is fixed (known), only the consumption can be varied by scheduling tasks at different time points. We call this type of constraint a *consumer constraint*.

3.2 Finished Products

The opposite problem occurs when scheduling with hard delivery dates of finished products. Each task produces a certain quantity of finished products at the end of the task. A certain initial stock of each finished product may be available. There are orders for defined quantities of finished products for fixed dates, which must be satisfied. The problem consists in scheduling the tasks in such a way that all orders can be satisfied. Since in this case the consumers are fixed and only the producers (tasks) are variable over time, we call this situation a *producer constraint*.

Note that this model is a generalisation of the usual production scheduling problem[Ba74][Go93][CC88][Fr82], where orders are matched one-to-one with production tasks. In this case, the producer constraint simply consists in scheduling all tasks before their due-date, which can be easily expressed by inequality constraints. This simplified model can be used in many situations where orders are few and known well in advance. It can not be used if many orders require the same finished product and there is no direct link between orders and production tasks.

3.3 Intermediate Products

This situation arises in multiple step scheduling problems, where some tasks create an intermediate product in certain quantities and other tasks use that intermediate product. Typical examples are batch based production in the chemical and food industry. We call this type of constraint the general *producer/ consumer constraint*.

Again this is a generalisation of a well known problem. In the job-shop scheduling problem [CC88][Pi88], each job consists of tasks which have to be scheduled in some order. This order often is caused by producer/consumer constraints between the tasks. The difference is that producers and consumers are matched one-to-one, this is not the case for the general producer/consumer constraint, where the sequence of tasks is not predetermined.

4. Mathematical Formulation

A mathematical formulation of the producer/consumer constraint is quite simple. Given an initial quantity Q_0 at time 0, and given a set of producers P_i which produce amounts Q_{P_i} at times T_{P_i} and a set of consumers C_j which consume amounts Q_{C_j} at times T_{C_j} , the following inequalities must hold:

Let

$$a = \min \left(T_{C_i}, T_{P_i} \right)$$

$$b = \max \left(T_{C_i}, T_{P_i} \right)$$

$$\forall t \in [a, b] \quad Q_0 + \sum_{T_{P_i} \leq t} Q_{P_i} - \sum_{T_{C_j} \leq t} Q_{C_j} \geq 0 \quad (1)$$

Unfortunately, these constraints can not be easily expressed as sets of inequality constraints, except in the case that T_{P_i} and T_{C_j} are fixed or a total (temporal) order is given for the P_i and C_j events. In that case, the constraint can be expressed as a set of recursive equalities and inequalities on the stock level S_t at time point t :

$$S_a = Q_0$$

$$\forall t \in [a, b-1] \quad S_{t+1} = S_t + \sum_{T_{P_i}=t} Q_{P_i} - \sum_{T_{C_j}=t} Q_{C_j} \quad (2)$$

$$\forall t \in [a, b] \quad S_t \geq 0$$

If the time points T_{P_i} or T_{C_j} are not fixed (resp. no total order given on the events), it is very difficult to express the constraints without adding many additional decision variables.

Fortunately, an alternative solution exists to this problem: using the cumulative constraint in CHIP.

5. Cumulative Constraint

Originally [AB93], the cumulative constraint was introduced in CHIP to tackle complex scheduling problems which could not be solved efficiently with current constraint logic programming systems. Also, experiments in solving complex decision making problems have shown the possibility of extending the use of the cumulative constraint in order to solve placement problems [BC94]. We now briefly describe the declarative semantics and the interpretation of the cumulative constraint

`cumulative([S1, ..., Sn], [D1, ..., Dn], [R1, ..., Rn], L)`

where $[S_1, \dots, S_n]$, $[D_1, \dots, D_n]$ and $[R_1, \dots, R_n]$ are non-empty lists of domain variables that have the same length n , and where L is a natural number. For a domain variable V , we define respectively $\min(V)$ and $\max(V)$ as the smallest and the greatest value of the domain of the variable V .

Let:

$$a = \text{minimum}(\min(S_1), \dots, \min(S_n))$$

$$b = \text{maximum}(\max(S_1) + \max(D_1), \dots, \max(S_n) + \max(D_n))$$

The cumulative constraint holds if the following condition is true:

$$\forall t \in [a, b]: \sum_{j \text{ such that } S_j \leq t \leq S_j + D_j - 1} R_j \leq L \quad (3)$$

Procedurally, the implementation of the cumulative constraint corresponds to a specialisation of the lookahead [VH89] declaration. From an interpretation point of view the cumulative constraint matches directly the single resource scheduling problem [Go93] [CC88], where S_1, \dots, S_n correspond to the start of the tasks, D_1, \dots, D_n correspond to the duration of the tasks, and R_1, \dots, R_n to the amount of resources used by each task. The natural number L is the total amount of available resource which must be shared at any instant by the different tasks. The cumulative constraint states that, at any instant t of the schedule, the sum of the amounts of resource for all tasks that are active at time t does not exceed the upper limit L .

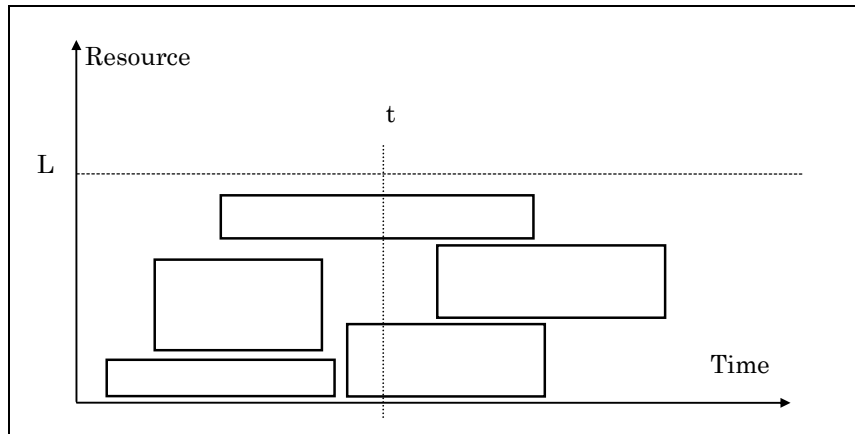


Figure 1: Cumulative Constraint

6. Modelling

We now face the question on how to express the producer/consumer constraint with the help of a cumulative constraint. We will describe the intuition of the modelling first, and then show its correctness formally.

The cumulative constraint models *tasks*, which have a start and an end date, a duration and a resource use, and a *resource availability* which constrains the amount of resource available at any given *time point*. The producer/consumer constraint models *events*, which occur at a *time point* with a given quantity and constrains the overall resource consumption over a *time period*.

In order to express the producer/consumer constraint with the cumulative constraint, we have to convert the events at a time point into tasks stretching over a time period and convert the constraints over resource use in a time period into constraints on individual time points. In a sense, the producer/consumer constraint is the dual of the cumulative constraint, in that time points and periods change their meaning.

The basic idea of the conversion is as follows:

A producer events makes some amount of resource available at a time point. We express this by a task which block a resource from the start until this time point.

A consumer event uses up some resource at its time point. This resource is no longer available and thus blocked until the end.

The consumer events can consume the initial stock and all production finished before them, the resource limit in the cumulative constraint is then the sum of all producer quantities plus the initial stock.

Graphically, we can express the situation with the following diagram:

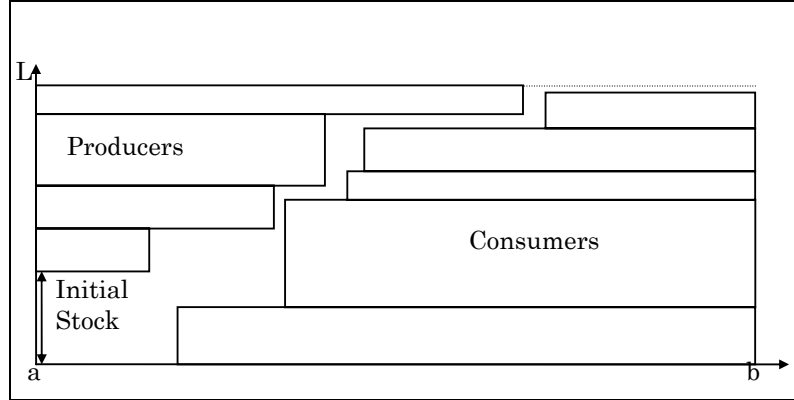


Figure 2: Producer/consumer constraint as cumulative

As long as the producer and consumer areas don't overlap, the constraint is satisfied.

Formally, we define *consumer tasks* and *producer tasks* to express the producer/consumer events.

The producer tasks for the P_i events are defined with a start date a , a duration T_{P_i-a} and a resource use Q_{P_i} .

The consumer tasks for the C_j events are defined with a start date T_{C_j} , a duration $b-T_{C_j}$ and a resource use Q_{C_j} .

The overall availability level for the cumulative constraint is given by the equation:

$$L = Q_0 + \sum_{\text{All } P_i} Q_{P_i} \quad (4)$$

Using the declarative semantics defined for the cumulative constraint above, we can show the correctness of our modelling. For each time point t , the cumulative constraint expresses the inequality

$$\forall t \in [a, b]: \sum_{j \text{ such that } S_j \leq t \leq S_j + D_j - 1} R_j \leq L \quad (5)$$

Using our descriptions for producer and consumer tasks, we obtain

$$\begin{aligned}
\sum_{T_{P_i} > t} Q_{P_i} + \sum_{T_{C_j} \leq t} Q_{C_j} &\leq \sum_{\text{All } P_i} Q_{P_i} + Q_0 \\
\sum_{T_{C_j} \leq t} Q_{C_j} &\leq \sum_{T_{P_i} \leq t} Q_{P_i} + Q_0 \\
Q_0 + \sum_{T_{P_i} \leq t} Q_{P_i} - \sum_{T_{C_j} \leq t} Q_{C_j} &\geq 0
\end{aligned} \tag{6}$$

which is the definition of the producer/consumer constraint.

7. Extensions

In this section we discuss some extensions of the basic constraint which often occur in real-life situations. We show that the same modelling can be used after some simple transformations.

7.1 Margins

The basic constraint above states that at each time point the stock level of the consumable resource must be greater or equal to zero. In reality, this constraint will be strengthened to hold a minimum operational stock level l , which must be available at all times. The minimum stock is a safeguard against unexpected delays or requirements of the consumable resource. The actual amount for l will be carefully chosen as a compromise between the inventory cost of this stock and the possibility of running out of stock.

Handling this safety margin in our constraint modelling is trivial, we just reduce the initial stock by the value l .

7.2 Variable Resource Consumption

In our modelling above we have assumed fixed quantities for producer and consumer events. We can easily extend the modelling to cope with variable quantities by using an extension of the cumulative constraint which allows variable resource limits.

A possible alternative can be used in special cases of batch based production. In some batch production systems, the size of one batch can be adjusted between a minimum level Min and a maximum level Max . In order to produce an overall quantity Qty of a product, we can then find the minimum number N of batches required to fill the order given by:

$$N = \left\lceil \frac{Qty}{Max} \right\rceil \tag{7}$$

This number of batch tasks is then scheduled with a producer/consumer constraint, which controls the start and end of each batch. After this schedule has been obtained, a fine tuning of the batch sizes can be performed independently using the recursive equations 2. This fine

tuning will minimise the batch sizes in such a way as to obtain the smallest possible inventory cost. This is done by reducing batch sizes at the beginning of the time period, while satisfying overall stock constraints.

This approach is worthwhile if the number of batches to be produced should be minimised, i.e. if the batch processing is a critical resource. If not, it may be better to split the production into more batches, so that the stock level can be further reduced.

7.3 Splitting Resource Use

Until now we have assumed that consumer events require the full quantity of the consumable resource at one time point, and that producer events make the total amount of resource available at one time point. This assumption works well for batch based production, where a given quantity of resource is either required at the beginning or becomes available at the end of a batch production step. This type of production is a *single step* producer/consumer.

This type of resource use is often coupled with other production steps, which continuously consume or produce resources. A typical example is a packaging line in food industry, which produces a constant flow of finished products and consumes a constant flow of raw materials. This type of production corresponds to a *continuous flow* producer/consumer constraint. As the flow rate is constant over time, we can treat this type of problem with the cumulative resource constraint, specifying that at each time point the quantities produced and consumed must be balanced. A third type of constraint occurs if a process produces/consumes resources neither in a single step nor in a continuous flow, but rather in *multiple steps*. This often happens if (limited) buffers are available in front of a continuous line, or the finished product from a continuous line is stored in large containers. A typical example would be the packaging of many bottles in crates and palettes. Depending on the time scale, finished palettes may become available in discrete steps, not in a continuous flow.

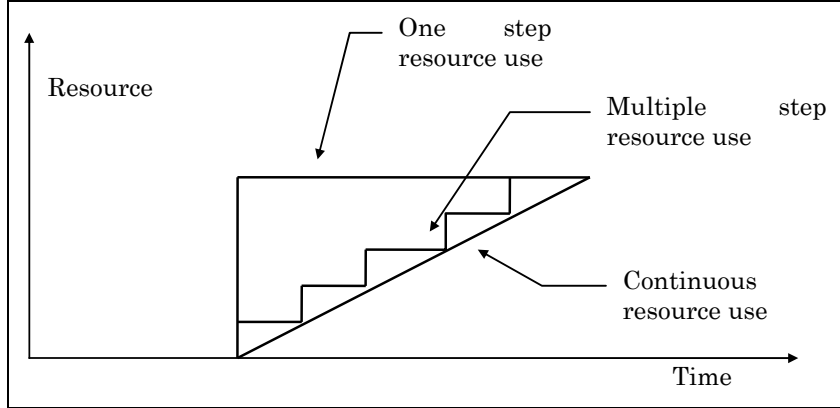


Figure 3: Different Resource Usage Pattern

In the modelling of the producer/consumer constraint, we can introduce multiple step consumption by splitting the producer/consumer tasks into sub tasks with smaller quantity and linked start/end times.

7.4 Limited Storage

The producer/consumer constraint expressed until now only assumes that a minimum limit exists on the stock available of the consumable resource. In practice, the overall storage space will often be limited as well. In this case, we also have to express a constraint on the maximum amount of stock available at each time point. This limit may be given for one resource (for example size of a storage tank) or may be given for the amount taken by several resources together (space in a warehouse). We can express this limit declaratively by the inequality:

$$\forall t \in [a, b] \quad Q_0 + \sum_{T_{p_i} \leq t} Q_{p_i} - \sum_{T_{c_j} \leq t} Q_{c_j} \leq S \quad (8)$$

To express this constraint with the cumulative formalism, we exchange the role of producer and consumer tasks. Consumer tasks now use storage space from the beginning and free storage space at their end, while producer tasks use up storage from their start to the end. The overall resource limit is given by

$$L = S + \sum_{\text{All } C_j} Q_{C_j} - Q_0 \quad (9)$$

The cumulative constraint then expresses

$$\sum_{T_{Pi} \leq t} Q_{Pi} + \sum_{T_{Cj} > t} Q_{Cj} \leq \sum_{\text{All } Cj} Q_{Cj} + S - Q_0$$

$$Q_0 + \sum_{T_{Pi} \leq t} Q_{Pi} + \sum_{T_{Cj} \leq t} Q_{Cj} \leq S$$
(10)

8. Practical Use

We now discuss some examples of the use of consumer/producer constraints for real-life applications. The ATLAS system is a scheduling application for one part of the MONSANTO plant in Antwerp. The program schedules the production of different types of herbicides. The production process is shown in figure 4. Production consists mainly of two steps, a batch formulation part and a continuous packaging (canning) operation.

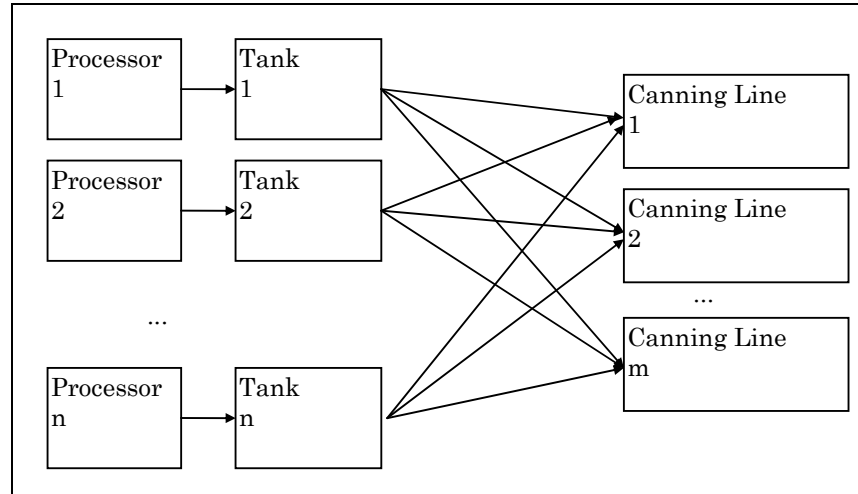


Figure 4: ATLAS production process

Besides many constraints associated with machine choice, set-up times and cumulative manpower constraints, producer/consumer constraints play a major role in the problem solver. The constraints occur in different places:

- Each batch processing step requires different raw materials from a set of available raw materials. Raw materials are stored and delivered at different (fixed) time points in the future. This is a typical consumer constraint where we schedule the use of the raw materials, but not their production.
- The chemicals produced in the batch process are then used to fill bottles and cans of different types and sizes. There is a limited number of storage tanks of fixed sizes which are used to store the

product between production and consumption. The materials for the canning operation must be available at the beginning of each shift. This producer/consumer constraint expresses both minimum and maximum values on the stock levels. Different packaging tasks may use the same intermediate product, which is produced in several batches. The consumption for one task is split in multiples of shift based demands.

- A fine tuning of the batch sizes is performed as a separate step after a schedule has been obtained and accepted. This fine tuning is used to reduce the storage requirements of the intermediate products.
- Besides the chemicals, the canning operations require other packaging materials like bottles, caps, labels, cartons, tape etc. Different canning tasks compete for some of the resources, which can be delivered with a fixed lead time only or by daily call-off from the manufacturer. Again, this is a consumer constraint which splits demands based on the shift pattern. Due to the number of different packaging materials and their storage requirements (empty bottles), only very limited stocks can be held inside the plant. Transfers from and to outside warehouses (also of limited size) must also be taken into account.
- The canning tasks do not correspond directly to orders, one task may serve different orders or several tasks may be needed to fill one order. In addition, some finished product stock may be available. A producer constraint expresses this type of problem.

Not all of these constraints are active in the actual system. Some stock levels (like raw materials) are not critical to the schedule, others (like finished products) are known to be unsatisfiable, as not all orders can be produced in the given time limits.

The actual system works with data sets of several dozen intermediate products and more than one thousand packaging materials, of which roughly one hundred are critical. A typical schedule may contain several hundreds of batches and canning tasks split into many sub-tasks to handle the stock level constraints with an eight hour resolution. In addition, (literally) several thousand other constraints are needed to express other parts of the scheduling problem.

The ATLAS system has been developed jointly by Beyers and Partners and COSYTEC. It is fully operational and in daily use since April 1994. Its overall size is around 30000 lines of CHIP and ORACLE forms/reports code, of which 4000 lines form the actual problem solver. A complete re-run of the problem solver takes some minutes to find a solution of high quality.

The producer/consumer model with the cumulative constraint made it possible to express and effectively solve this scheduling problem in CHIP. But the ATLAS system is not the only application using the producer/consumer modelisation. We just mention two others:

- A scheduling/planning system for crude oil transport in a pipeline network uses producer/consumer constraints for stock levels both for the different sources and for the different sinks.
- A transportation scheduling system for a food processing company uses producer constraints to ensure that raw material transports from many sources arrive in time to keep the different factories operating, while minimising the stock held at each site.

9. Summary

In this paper we have discussed the modelling of producer/consumer constraints for consumable resources with the cumulative constraint in CHIP. This modelling allows to express the constraint in a simple, yet very efficient manner.

Producer/consumer constraints arise in many scheduling and planning problems dealing with stock levels of raw materials and intermediate or finished products. The constraints state that at all time points the available stock level must be between minimum and maximum values. Producer/consumer constraints are a generalisation of sequence constraints arising in flow shop or job shop scheduling problems and occur in many scheduling problems with batch or continuous processing. It is important to note that we could add this important class of constraints to CHIP without adding any new built-in constraint, just by re-using existing constraints in a novel way.

In addition to the formalism we have presented a typical example of the constraint use from the ATLAS system, a real-life application jointly developed in CHIP by Beyers and Partners and COSYTEC.

In conclusion, the producer/consumer constraint enables us to solve new, more complex scheduling problems with the CHIP system in a simple, yet efficient way.

10. References

[AB93] A. Aggoun, N. Beldiceanu
 Extending CHIP in Order to Solve Complex Scheduling Problems
 Journal of Mathematical and Computer Modelling, Vol. 17, No. 7, pages 57-73
 Pergamon Press, 1993

[Ba74] K. R. Baker
 Introduction to Sequencing and Scheduling.
 John Wiley, 1974.

[BKC94] G. Baues, P. Kay, P. Charlier
 Constraint Based Resource Allocation for Airline Crew Management
 ATTIS 94, Paris, April 1994

[BS92] N. Beldiceanu, H. Simonis

Aircraft Maintenance Scheduling
COSYTEC TR, Nov 1992

[BC94] N. Beldiceanu, E. Contejean
Introducing Global Constraints in CHIP
Journal of Mathematical and Computer Modelling, to appear

[BCP92] J. Bellone, A. Chamard, C. Pradelles
PLANE -An Evolutive Planning System for Aircraft Production.
First International Conference on the Practical Application of Prolog. 1-3 April
1992, London.

[BDP94] P. Bouzimault, Y. Delon, L. Peridy
Planning Exams Using Constraint Logic Programming
2nd Conf Practical Applications of Prolog, London, April 1994

[CC88] J. Carlier and P. Chretienne.
Problèmes d'ordonnancement.
Masson, Paris, 1988

[CDF94] A. Chamard, F. Deces, A. Fischler
A Workshop Scheduler System written in CHIP
2nd Conf Practical Applications of Prolog, London, April 1994

[DHS88] M. Dincbas, P. Van Hentenryck, H. Simonis, A. Aggoun, T. Graf and F.
Berthier.
The Constraint Logic Programming Language CHIP.
In Proceedings of the International Conference on Fifth Generation Computer
Systems (FGCS'88), pages 693-702, Tokyo, 1988.

[DS91] M. Dincbas, H. Simonis
APACHE - A Constraint Based, Automated Stand Allocation System
Proc. of Advanced Software Technology in Air Transport (ASTAIR'91)
Royal Aeronautical Society, London, UK, 23-24 October 1991, pages 267-282

[Ev92] O. Evans
Factory Scheduling Using Finite Domains
In Logic Programming in Action LNCS 636, 45-53, 1992

[FS90] M.S. Fox and K. Sycara.
Overview of CORTES: A Constraint Based Approach to Production Planning,
Scheduling and Control.
In Proceedings of the Fourth International Conference on Expert Systems in
Production and Operations Management, 1990.

[Fr82] S. French
Sequencing and Scheduling: an Introduction to the Mathematics of the Job-Shop
Horwood, Chichester, 1982

[FHK92] T. Fruewirth, A. Herold, V. Kuchenhoff, T. Le Provost, P. Lim, M.
Wallace

Constraint Logic Programming - An Informal Introduction
In Logic Programming in Action LNCS 636, 3-35, 1992

[Go93] Gotha
Les Problemes d'Ordonnement
Operations Research vol27, 1, 1993, pages 77-150

[JM94] J. Jaffar M. Maher
Constraint Logic Programming: A Survey
Journal of Logic Programming, 19/20: 503-581, May-July 1994

[Pi88] E. Pinson
Le Probleme de Job Shop
These de Doctorat de Univ Paris VI, 1988

[VH89] P. Van Hentenryck.
Constraint Satisfaction in Logic Programming.
MIT Press, Boston, Ma, 1989.

[HSD92] P. Van Hentenryck, H. Simonis, M. Dincbas
Constraint Satisfaction using Constraint Logic Programming
Journal of Artificial Intelligence, Vol.58, No.1-3, pp.113-161, USA, 1992

[Wa94] M. Wallace
Applying Constraints for Scheduling
In B. Mayoh, E. Tyugu, J. Penjaam (Eds) Constraint Programming, Springer
Verlag, 1994