The Use of Exclusion Constraints to Handle Location Continuity Conditions

Helmut Simonis

COSYTEC SA
4, rue Jean Rostand
Parc Club Orsay Université
F-91893 Orsay Cedex
France
simonis@cosytec.fr

1. Abstract

In this report we describe the use of exclusion constraints to handle continuity conditions on arrival and departure locations in transportation problems. The arrival location of a task assigned to a resource must be the same as the departure location of the next task assigned to this resource. In CHIP these constraints are normally handled with the cycle constraint which allows to express conditions on consecutive tasks as a graph problem. We here present an alternative model which uses the diffn constraint in CHIP to express the condition. This model provides less propagation than the cycle constraint, but can be used for very large problem instances, provided the number of distinct locations is small. In addition the model handles the problem of passive transportation, tasks which move resources from one location to another without performing active duty. We also present two examples of the use of the exclusion constraint for a airline personnel assignment problem and a transport scheduling problem from the food processing industry.

2. Introduction

Many transportation problems are subject to the following constraint. Tasks (operations) are assigned to resources (persons, vehicles). Each task starts at some departure location and end at some arrival location. For two consecutive tasks assigned to the same resource, the arrival location of the first task and the departure location of the second task must coincide.

Typical examples of such problems are aircraft rotation assignment and crew rotation [BKC94]. A number of flights (tasks) must be scheduled to either aircraft or crew members in such a way that the arrival station of one flight is the departure station of the next flight performed by this resource. Usually numerous other constraints must be satisfied as well, for example connection times, work hour limits, rest periods, etc.

It is important to note that this constraint on location continuity is “weakly incremental”, i.e. a partial assignment which can be completed to a legal one may be illegal in itself. This behaviour makes it difficult to enforce consistency of the constraint at all times.

3. Conventional Model

The standard way to express this constraint in CHIP [DHS88] [AB93] [VSD92] is to use the cycle constraint [BC94] working on a directed graph of N resource and M task nodes [BKC94]. Each resource and each task activity is represented by a node in the graph. The resource nodes are also called special nodes. Two tasks A and B are linked in the graph if task A can be immediately followed by task B, that is if task B starts after task A finishes and the departure station of task B is equal to the arrival station of task A. Resource nodes are linked to task nodes if the resource can either start or end work with these tasks. Resource nodes are also linked to themselves. The cycle constraint ensures that there are N cycles in the graph; Each resource node belongs to a different cycle. Each cycle may contain a number (perhaps zero) of task nodes, which describe the sequence of operations performed by the resource of the cycle.

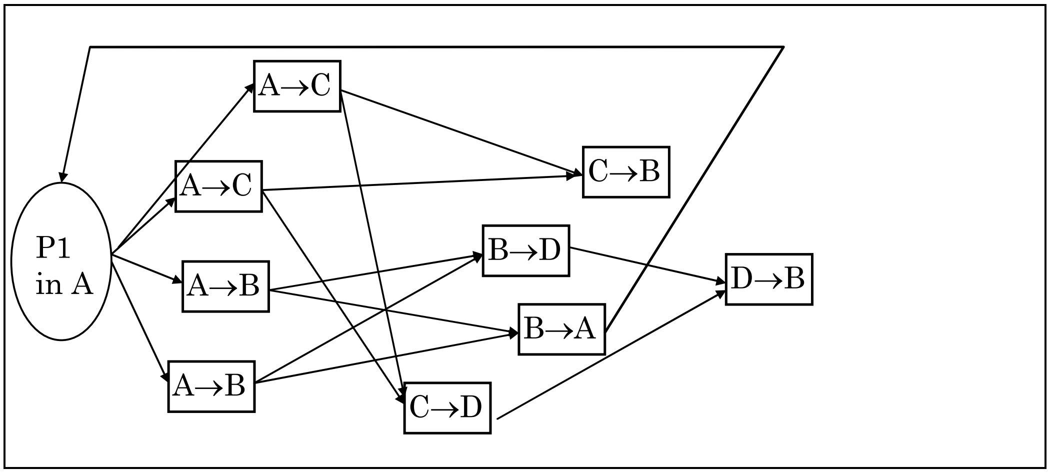

Figure 1: Task and Resource Nodes

The cycle constraint uses advanced constraint reasoning techniques to check whether certain nodes can belong to certain cycles. This reasoning is quite complex and sometimes time consuming. Therefore the cycle constraint is best used for difficult problems of small to medium size (several hundred nodes). In addition, it can be quite complex to set up the cycle constraint initially, as the connectivity inside the graph must be pre-computed as domains of the node variables. Since each nodes has (typically) a different domain, the calculation of the initial domains can be quite expensive in the order of $O((N+M)^2)$ for N resource nodes and M task nodes.

An additional problem arises if we allow passive transports [BKC94]. A passive transport is an additional operation which is used to move a resource from one location to another. The passive transport may be required to get a resource to a location so that a task starting from this location can be handled by the resource. In general, the number of passive transport required is not known in advance, but must be kept small since they incur significant costs. In the constraint model presented above, passive transportation is represented as additional nodes which are linked into the graph by connecting task nodes ending and starting in different locations. The main problem in this model is to limit the number of extra nodes which are introduced. If all passive transport possibilities are included, the graph size will explode. If only a few extra nodes are added, we may not be able to construct all cycles successfully.

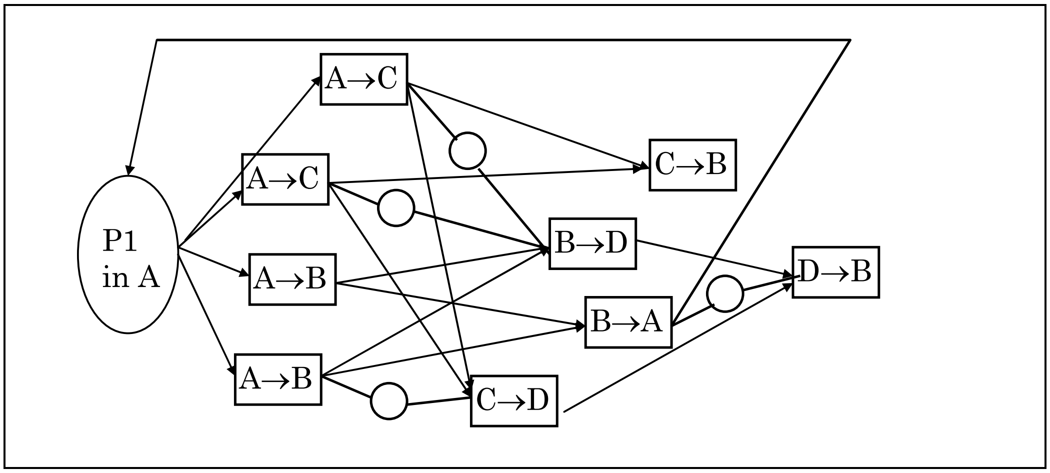

Figure 2: Adding passive transport nodes

We also have to introduce strategies to avoid using passive transportation excessively during the assignment phase. This involves adding cost variables to each node (via element constraints) and penalising connections to passive transport nodes with a higher cost.

In general we see that the standard modelling with the cycle constraint involves rather complex pre-processing steps which are warranted only for hard and very constrained problems. In the next section, we discuss an alternative, which is computationally cheaper, but also less powerful.

4. Alternative Model

We now present an alternative model which avoids some of the problems of the standard model, and uses a simpler, but still powerful propagation. This alternative model will be useful if many resources and tasks must be scheduled and only a very limited number of locations need to be taken into account. The model is a generalisation of the scheduling problem with one location, which we explain now.

4.1 Single Location Problem

Consider the following problem. A number of activity tasks *Ti*, *1<=i<=n*, must be assigned to a number of resources *Rj, 1<=j<=m*. A resource may handle one task at a time, i.e. the tasks assigned to one resource may not overlap in time. In CHIP, we express this constraint with the diffn constraint [BC94]. The diffn constraint expresses in its most simple form an n-dimensional non-overlapping condition. We can describe our current problem in two dimensions, one being time, the other being resources. Tasks are modelled as rectangles with origin (*si, ri*) duration *di* and height one.

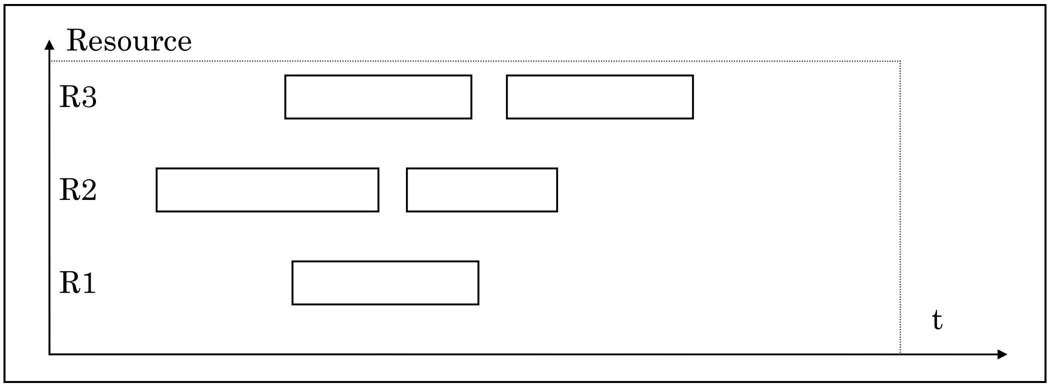

Figure 3: Single Location Problem

The diffn constraint assures that the rectangles do not overlap, so that tasks assigned to the same resource are scheduled to non-overlapping time periods and that tasks overlapping in time are assigned to different resources. Note that this constraint is a necessary condition also for the multiple location problem. In fact, we include this constraint as the first part of the model for multiple locations. We now want to extend the constraint to cope with multiple locations.

4.2 Multiple Locations

For each task *Ti* we define departure and arrival locations *LDi* and *LAi*, which belong to a finite set *L* of all locations. We also need a transport matrix which defines the minimum time required to reach one location from another. We express this matrix as a function

$$f: L \times L \rightarrow N$$

The value corresponds to the time required for a passive transport between two locations. The function *f* must satisfy the triangular inequality condition

$$f(L_1, L_3) \leq f(L_1, L_2) + f(L_2, L_3)$$

and the conditions

$$f(L, L) = 0$$
$$f(L_1, L_2) > 0 \quad if \quad L_1 \neq L_2$$

We now introduce one diffn constraint per location. We will identify the constraint by the location for which it stands. The x-axis of the diffn constraints represents time, the y-axis represents resources with each resource occupying a range of *k>=2* values. The meaning of the value *k* will be explained below.

Each task is represented in each diffn constraint. by a combination of three alternative rectangles. If a task starts at a location *Li* we place a *start marker* in the diffn constraint of that location. The start marker is a rectangle in position *si, k*ri* with a duration one and a height *k*. An *end marker* is placed in the diffn constraint of the arrival location of each task *Ti*. The marker is placed at location *si+di-1, k*ri* again with a duration of one and a height *k*.

We also place *exclusion markers* in the diffn constraints for all locations if we don't put a start marker or an end-marker. The exclusion markers are placed before and after the task period. They ensure that no other tasks starting from or ending at the wrong location will be assigned to the resource close to the task which is already placed there.

For a task *Ti* starting at location *LDi* at time *si*, we place exclusion markers in all locations *LD* different from *LDi* as rectangles starting at *si-f(LD, LDi)* with duration *f(LD, LDi)* and height 1. The domain in y dimension is *k*ri..k*ri+k-1* if the task is assigned to resource *ri*.

For a task *Ti* ending at location *LAi*, we place exclusion markers in all locations *LA* different from *LAi* as rectangles starting at *si+di* with duration *f(LAi, LA)* and height 1. The domain in y dimension is *k*ri..k*ri+k-1* if the task is assigned to resource *ri*.

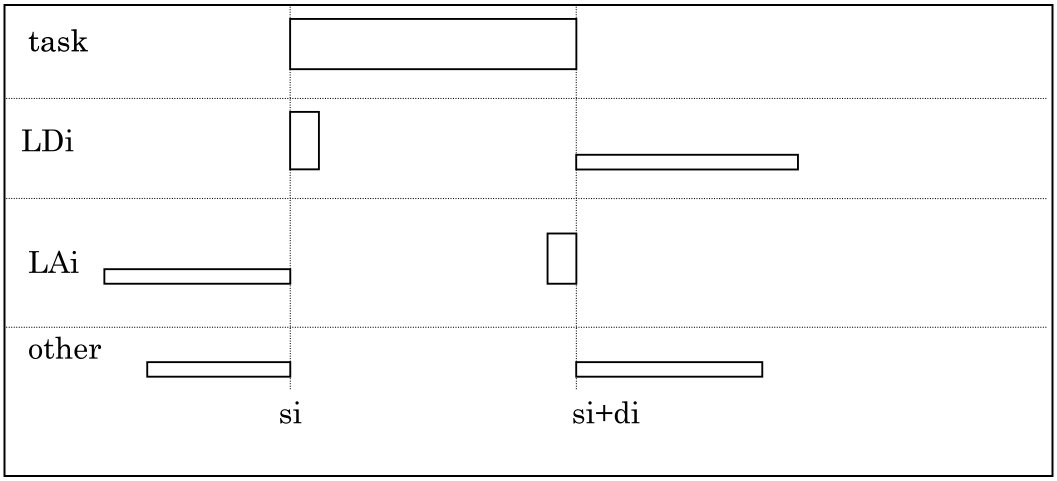

Figure 4: Start, End and Exclusion Markers

Note that the exclusion markers can have different length, depending on the structure of the function *f*. The markers do not extend into the period *si..si+di*. This period on resource *ri* is excluded from the domain of any other task due to the single location diffn constraint. The height *k* of the exclusion markers and their position in the y-dimension is chosen such that exclusion markers of different tasks may be assigned to different y values and therefore do not overlap, but that exclusion markers prohibit start or end-markers in the time period for the resource. The exact value

of *k* can be computed from the minimum and maximum values of *f*, but a value of 10 will be correct for most situations.
Intuitively, the diffn constraint means that if a task requires a resource at some location *Li* then the resource can not be at any other location within the time period defined by the travel time matrix *f*. Together with the basic diffn constraint the location constraint ensures that consecutive tasks for one resource either have compatible locations or are sufficiently far apart to allow for passive transportation between them.

5. Example

We now present a very simple example with four tasks assigned to a single resource. In the first case, the continuity condition is satisfied as the arrival location of the previous task always is the departure location of the next task. Figure 5 shows the task rectangles in the main diffn constraint at the top and in the three diffn constraints for locations *A*, *B* and *C*.

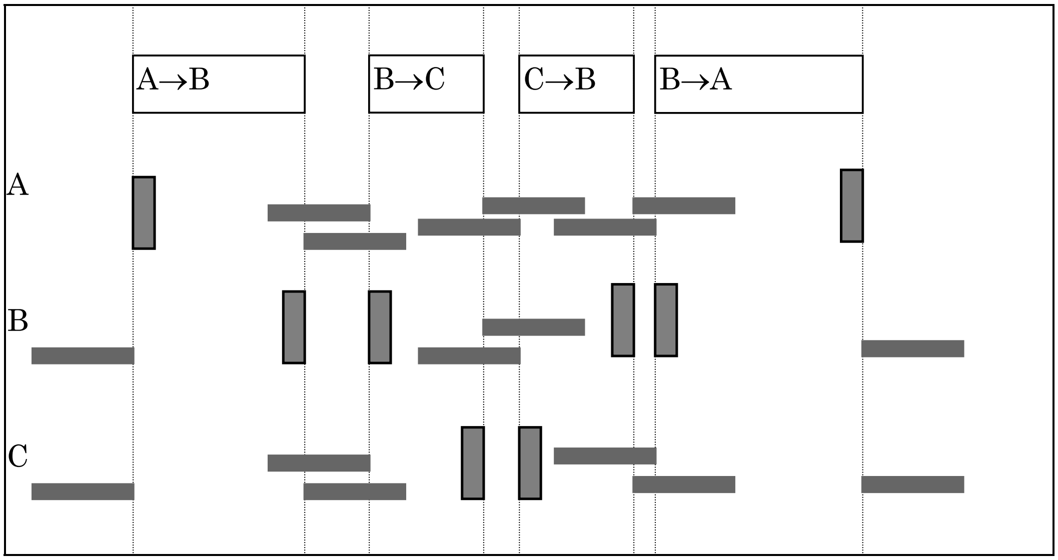

Figure 5: Simple Example

In the second example we require a passive transport between the third and the fourth task, which must be separated atleast by the time required for the transport. Figure 6 shows the graphical representation of the different diffn constraints.

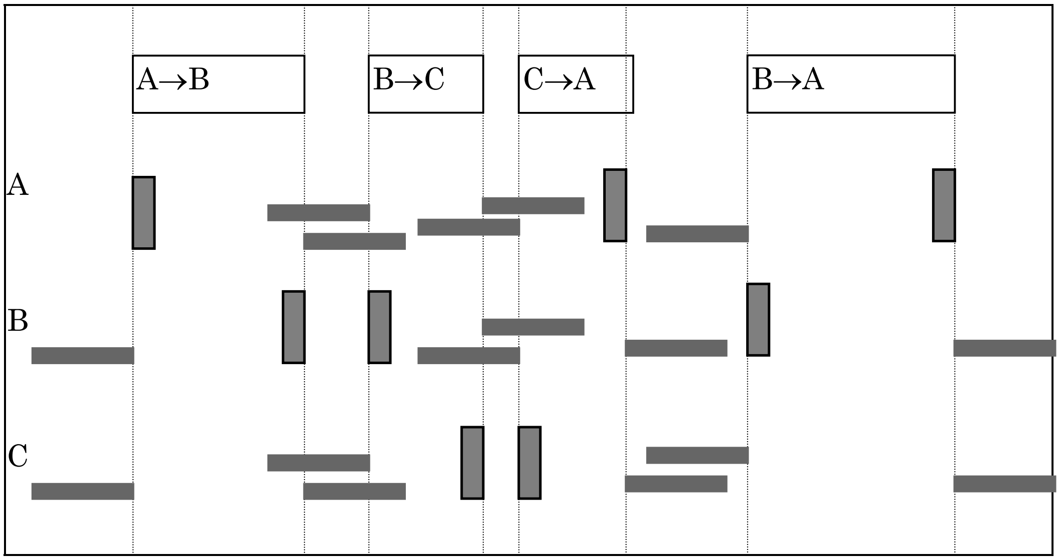

Figure 6: Passive Transport

In the last example we see what happens if the constraint is violated. In this example the third and fourth task are too close to allow a passive transport between locations *A* and *B*. The diffn constraints in location *A* and *B* are violated. Figure 7 depicts this situation.

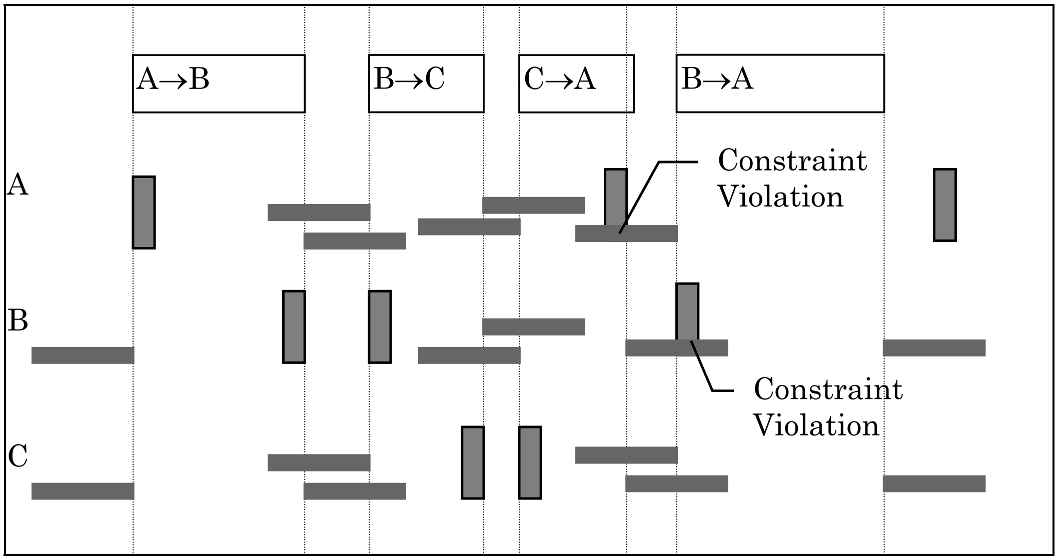

Figure 7: Constraint Violation

6. Applications

We will now discuss two possible applications of the exclusion constraints for real-life problems in different areas.

6.1 Airline Crew Rotation

The first problem is a crew assignment problem [BKC94] for an airline with multiple home bases. Home bases are locations where personnel of the airline are based and where these persons start and end their work periods. The problem consists of assigning flight personnel to trips, a sequence of flight legs which starts at one home base and ends at the same or another home base. The number of bases to be handled is quite small (5-6 locations). The arrival and departure times of all flights are already known, so that we face an assignment rather than a scheduling problem. The number of trips to be covered can be quite large (several thousand). A solution based on the cycle constraint not feasible due to the large number of trips and the complex connectivity at the home bases connecting all flights arriving to a large number of flights departing within a certain time period. In addition the handling of passive transportation between home bases requires the introduction of a large number of auxiliary nodes in the cycle graph, thereby increasing even more the complexity of the graph generation phase.

We can note that the generated graph is very wide, i.e. most nodes have many possible successor nodes, so that alternatives are not sufficiently pruned for good constraint propagation.

A solution based on exclusion markers offers a simple model of the required continuity of location and at the same time handles the problem of passive transport in a very simple way without introducing additional tasks. A heuristic in the assignment routine minimises the use of passive transports by packing tasks on a resource as tightly as possible.

This part of the model is of course only one aspect of the overall constraint model for the problem, a multitude of other constraints must be satisfied.

6.2 Livestock transport problem

The second example of the use of exclusion markers is a problem of road transport [SBCK95]. Lorries of different types are used to transport livestock from farms to processing factories. This transport is organised in tours starting and ending at a factory. Empty transport between factories is possible, but should be avoided if possible.

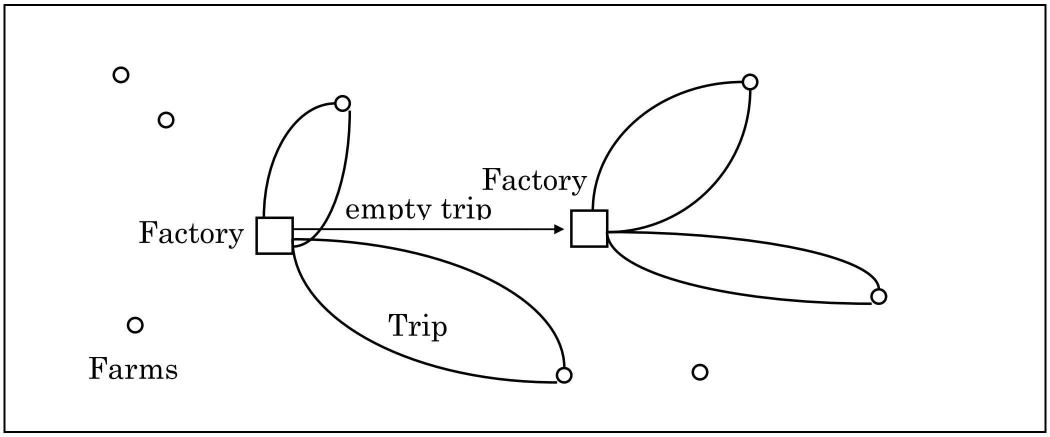

Figure 8: Life Haul trips

The normal duration of a trip is known, and start and end-dates have to be scheduled within certain limits to satisfy producer/consumer constraints [SC94] of the factories. The objective of the problem is to either minimise the number of lorries required to cover a given transportation problem for a day, or to balance the workload for a fixed given number of lorries available at a day. The exclusion constraints are well suited to this problem since the number of factories is quite small (2-3) and the number of trips may be quite large.

In difference to the crew allocation problem here we have to solve a combined scheduling and assignment problem, since the trips can be moved in time in addition to being assigned to different drivers.

7. Summary

In this paper we have introduced the concept of exclusion markers in order to handle constraint on locations in transport and assignment problems. This modelling provides an alternative to the use of the cycle constraint for solving such problems. The exclusion constraint should be used if only a limited number of different locations have to be handled and if the number of tasks to be assigned is large.

The exclusion markers use multiple diffn primitives of CHIP as their basic building block. The amount of propagation is restricted to local exclusion of resources from the domain of tasks. In contrast, the cycle constraint performs deeper (but also more expensive) propagation based on graph algorithms.

A major difference in the two alternative models is the handling of passive transports. In the cycle constraint, all possible passive transports have to be included *a priori* in the graph as additional nodes, increasing the problem complexity. With exclusion constraint, passive transport is handled *implicitly*. This is an advantage if passive transport is used on a regular basis, but does not suffice to express the problem where no passive transport is allowed at all.

We have presented two possible example uses of the exclusion constraint in two real-world problems. One is a crew assignment problem for an airline with multiple home bases, the other a transport problem in the food processing industry. In both instances the exclusion constraints can form a small, but important part of the overall constraint model. These examples show the usefulness of this constraint for a variety of scheduling and assignment problems as an alternative to the cycle constraint in CHIP.

Another use of exclusion markers for expressing set-up time constraints in scheduling problems is described in another paper [Sim95].

8. 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

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

[BC94] N. Beldiceanu, E. Contejean
Introducing Global Constraints in CHIP
Journal of Mathematical and Computer Modelling, Vol 20, No 12, pp 97-123, 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.

[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

[JL87] J. Jaffar J.L. Lassez
Constraint Logic Programming
Proc 14th ACM Symposium on Principles of Programming Languages, Munich, Germany, 1987

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

[SC94] H. Simonis, T. Cornelissens
Modelling Producer/Consumer Constraints
Technical Report COSY/TR/
October 1994

[SBCK95] H. Simonis, N. Beldiceanu, P. Charlier, P. Kay
A Model of TACT
Technical Report COSY/TR/
March 1995

[Sim95] H. Simonis
Modelling Machine Set-up Times in CHIP
Technical Report COSY/TR/
March 1995

[VSD92] 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