Calculating Lower Bounds on a Resource Scheduling Problem

Helmut Simonis

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

1. Abstract

In this paper we describe the modelling of lower bounds for resource scheduling problems with limited working time for disjunctive resources. We first describe a model of the original problem based on global constraints in CHIP. We then present four different lower bounds which are defined by relaxation of some conditions in the original problem. The lower bounds are obtained by using the cumulative constraint in CHIP and can be used both as a priori estimate and incrementally during problem solving. We also give an example from a real-life application, where these lower bounds are used to classify the difficulty of scheduling problems.

2. Introduction

Many scheduling problems contain the following sub-problem. Tasks with variable duration and start-dates must be scheduled in time and assigned to disjunctive resources. These resources often are personnel assigned to the tasks. The start time of the resource may be chosen (within limits) as part of the scheduling process. Each resource will be available for a fixed time period from its start-time and can be assigned to tasks during this period. For personnel this time period is usually defined by contract and will be the same for all workers. One resource may work on one task at a time or may be waiting (idle) between two tasks. It is often interesting to obtain a lower bound on the number of resources required for a particular scheduling period without actually generating the schedule. If such lower bounds can be obtained easily, they can be used a priori to classify problems before scheduling is attempted. Another use is the incremental update of the bound during the solution process as part of an optimisation procedure.

There are two obvious lower bounds which are quite often used for estimating the resource consumption. One bound is based on the total surface of all tasks to be scheduled. If this surface is divided by the amount of work which can be performed by a resource in one work period, a lower bound is obtained. Another bound is found by treating the problem

as a cumulative scheduling problem and obtaining a lower bound on the cumulative resource requirements.
If the scheduling problem is not constrained and tasks can be scheduled to any resource at any time, the problem becomes a bin-packing instance, for which very good lower bounds are known [CL91][MT90]. In general it is possible to look at the bin-packing relaxation of the scheduling problem to obtain another lower bound.
In this note we discuss yet another lower bound which can be obtained by using the cumulative constraint [AB93] in CHIP [DHS88] [VSD92] [FHK92] [JM94]. This bound often gives better results than the first two obvious methods, and is influenced by constraints on time periods.
The different lower bounds are obtained from the original problem by selectively relaxing some aspect of the constraints. We briefly discuss this taxonomy of lower bounds.
We then present an application of the method to a real-life scheduling/assignment problem. In the TACT [Sim95b] system, the lower bound is used to estimate the number of drivers required in a transportation problem. This lower bound is used to classify the difficulty of finding schedules on different days and also serves as part of an explanation component if the problem solver fails to find a schedule.

3. Problem statement

We start with a description of the problem that we want to tackle. We have to schedule a number of tasks in time and assign them to disjunctive resources. The range of possible start dates may be restricted a priori and is normally influenced by other constraints. The duration of the tasks may be fixed or may depend on the resource assignment. Each resource can handle one task at a time. The time between the beginning of the first task on a resource and the end of the last task is called the work period. We assume that the length of the work period of any resource is restricted to a maximal value called *W*.

3.1 Constraint expression

We will now first explain how the problem itself can be expressed in CHIP. This model is based on variants of the diffn [BC94] constraint.

We describe tasks with a tuple of domain variables

- *Start*
- *Duration*
- *Resource*

which have appropriate initial domains.

3.1.1 Assignment Problem

The resource constraint will be expressed with a 2-dimensional diffn constraint, where the x-dimension denotes time and the y-dimension denotes the resource assignment. A task is represented as a rectangle with

start point *(Start, Resource),* length *Duration* and height 1. A graphical interpretation of the constraint is given in Figure 1.

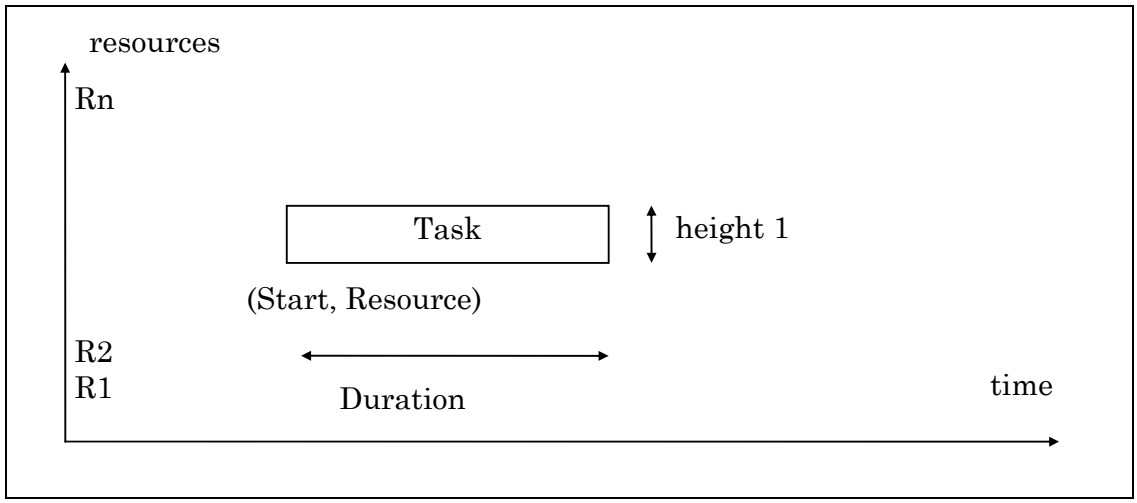

Figure 1: Diffn constraint for machine assignment

The diffn constraint is called in the following form

```
diffn([[Start1, Resource1, Duration1, 1],
      [Start2, Resource2, Duration2, 1], ...],
      unused, unused, unused, unused, unused)
```

The first argument of the diffn constraint is a list of n-dimensional rectangles, each represented as a list. In our 2-dimensional case, we have to give the four values x- and y-starting point, length and height. The remaining five arguments of the diffn constraint allow to express additional constraint, which are explained in [BC94].

3.1.2 Work periods

We will now use one of the additional parameters, the region constraints [BC94], to implement the work period limits. For the region constraints we give a set of rectangles which constitute the regions on which we express some constraints. For each region, we also give a projection dimension and a domain variable which is constrained. The region constraints work as follows. For all task rectangles which intersect with a region rectangle, the constraint computes the distance between the first and the last use in the projection direction for that region. In our example we create one region for each resource. The regions have height one and stretch over the complete scheduling horizon. The projection direction is the x-axis (direction 1). This means that the value calculated by the region constraint is the range between the first and the last use of the resource by any task. This is just the work period as we have defined it above. Figure 2 shows a graphical representation, the mathematical formulation of the constraint is given in [BC94].

By limiting the domain of the range variable, we enforce a limit on the working period for each resource.

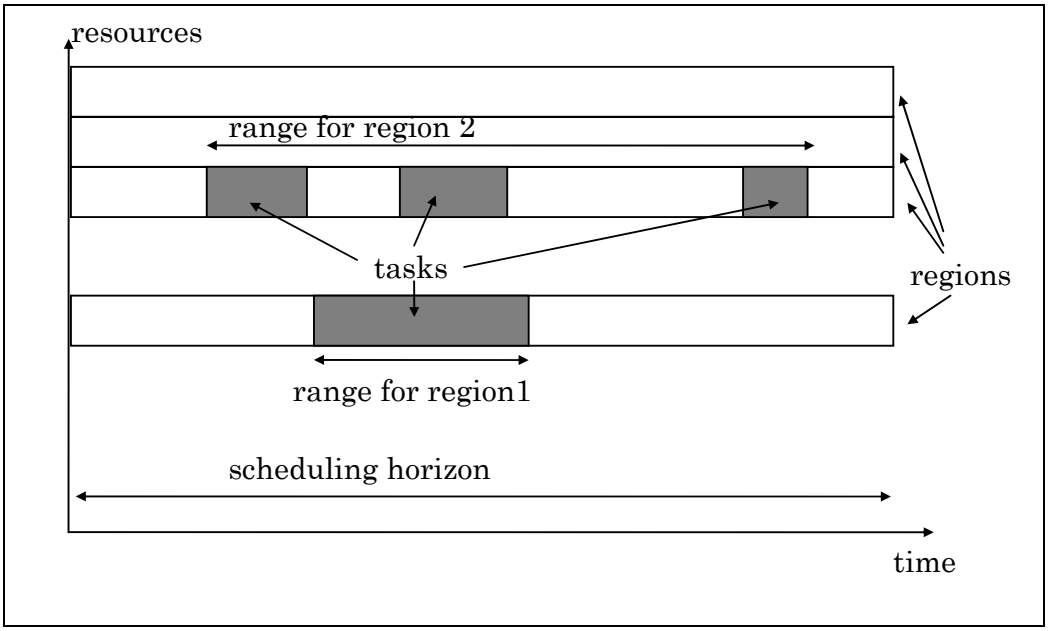

Figure 2: Region constraints

The extended diffn constraint is now called in the following form

```
Range1 :: 0..W,
Range2 :: 0..W,
...
diffn([[Start1, Resource1, Duration1, 1],
       [Start2, Resource2, Duration2, 1], ...],
     unused, unused, unused, unused,
     [[[0,1,Period,1],1,Range1 ],
      [[0,2,Period,1], 1, Range2 ],... ])
```

where *Period* is the overall scheduling horizon and the variables *Range1*, *Range2*, etc describe the working time for the different resources.

4. Lower bounds

Given this model we can now present the different lower bounds and show how they are expressed with the cumulative constraint of CHIP.

4.1 Surface

A very simple lower bound is obtained by calculating the overall amount of work to be scheduled and to divide this by the maximum length of the work period. In principle we can simply sum up all durations of the tasks and divide this number by the length of the work period. If the duration of the tasks is not fixed, it is interesting to express this lower bound via a

constraint which is automatically updated as soon as some duration changes. We introduce a domain variable *Nr* for the lower bound and create a linear constraint, which expresses the condition

$$Nr :: 0..NrTasks$$
$$Dummy :: 0..W-1$$
$$W * Nr = \sum_i Duration_i + Dummy$$

This type of lower bound will be quite good if there are few constraints on the sequence of tasks and the task duration is small compared to the length of the work period. The lower bound calculated is just the number of resources needed to perform all tasks, if no resource is kept waiting and all resources work during the full span of the work time. If the start times of the tasks are constrained a lot or the tasks can not be combined into groups with the duration of the work period, this lower bound will be too optimistic.

4.2 Peak

Another type of constraint is obtained by the cumulative relaxation of the assignment problem. Instead of assigning tasks to multiple disjunctive resources, we handle one cumulative resource to which we assign tasks with a resource usage of 1. The limit is obtained by the peak use of the cumulative resource at any one time point.

This lower bound can be easily calculated using the cumulative constraint in CHIP. As the domain variables for the tasks are more and more constrained, the lower bound obtained increases. We again define a domain variable for the lower bound and express the constraint

```
Nr :: 0..NrTasks
cumulative([Start1, Start2, ...],
     [Duration1, Duration2, ...],
     [1, 1, ...],
     unused, unused, Nr, unused, unused)
```

We use only some of the parameters of the cumulative constraint. A more detailed description is found in [AB93]. The first parameter is the list of start dates, the second parameter is the list of durations, the third parameter is a list of resource use, which is set to 1 for each task in our case. The *Nr* parameter constrains the variable to the peak utilisation of the resource.

It is often useful to visualise the cumulative constraint with a resource utilisation profile, which shows how many resources are used at any time point. This profile changes as the domains of tasks are more and more restricted.

Figure 3 shows the cumulative constraint for four tasks together with the resource profile drawn with a thick line.

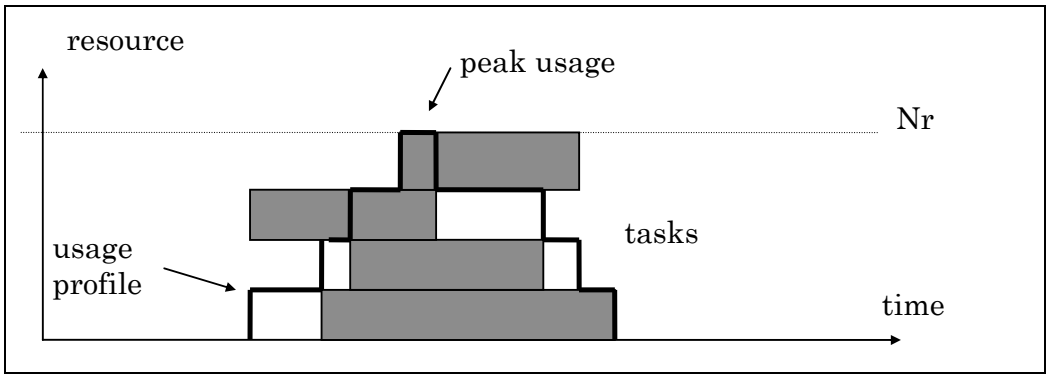

Figure 3: Peak usage limit

This *peak lower bound* takes the limits on the start dates into account. The smaller the domains of the start variables, the better the estimation on resource use. This lower bound will therefore be more interesting during the generation of a schedule, rather than as an a priori lower bound. The calculation also completely ignores the length of the work period and so will give rather poor initial estimates if the work period length is small compared to the scheduling period.

4.3 Combination

In passing we want to mention that we can extend the peak cumulative model to calculate the surface lower bound as well. For this we use the additional last parameter of the cumulative constraint which has two parts, an intermediate level of resource use and a domain variable which is constrained to the surface of all tasks above the intermediate level. In the general case, this parameter can be used for constraint relaxation or for allowing a limited amount of overtime on a cumulative resource.

```
Nr :: 0..NrTasks
Surface :: 0..NrTasks*W
cumulative( [Start1, Start2, ...],
     [Duration1, Duration2, ...],
     [1, 1, ...],
     unused, unused, Nr, unused, [0,Surface])
```

As the intermediate level we specify 0, so that the total surface of all tasks is calculated in *Surface*. Since the tasks all have height 1, the total surface corresponds to the total duration of all tasks. From the surface a simple equality constraint allows to calculate the number of resources required.

4.4 Bin packing

As mentioned before, a bin packing lower bound can be obtained by taking the duration of the tasks, the work period and the resource assignment into account, while ignoring the start dates. Each resource is a bin with

fixed size *W*. Each task is an item that should be put in one bin and which uses *Duration* space in the bin. We are interested in a lower bound of the number of bins required for a given set of items. Normally, we do not want to actually solve the bin-packing problem exactly, but are more interested in a lower bound which can be obtained without too much effort. The mathematical background of this problem can be found for example in [CL91] [MT90].
With CHIP, we can express the bin packing problem with a cumulative constraint [AB93] [BDP94]. The x-axis on this constraint will be the bin number, the y-axis (the cumulative resource) will be the time. Each task is represented by a rectangle with length 1 and height *Duration*, which starts at location *Bin*. This domain variable denotes to which bin the rectangle will be assigned. The resource limit of the cumulative constraint will be set to *W*, the maximal work period length. We use one additional parameter in the cumulative constraint which constrains the resource use in the x-axis.
The constraint takes the form

```
Limit :: 0..W,
End :: 1..NrTasks,
[Bin1, Bin2, ...]:: 1..NrTasks,
cumulative([Bin1, Bin2, ...],
     [1, 1, ...],
     [Duration1, Duration2, ...],
     unused, unused, Limit, End, unused),
```

The *End* value will be a lower bound on the number of resources required according to the formula

$$End = \max(Bin_i + 1) \leq Bound + 1$$

Figure 4 shows this constraint in a graphical form.

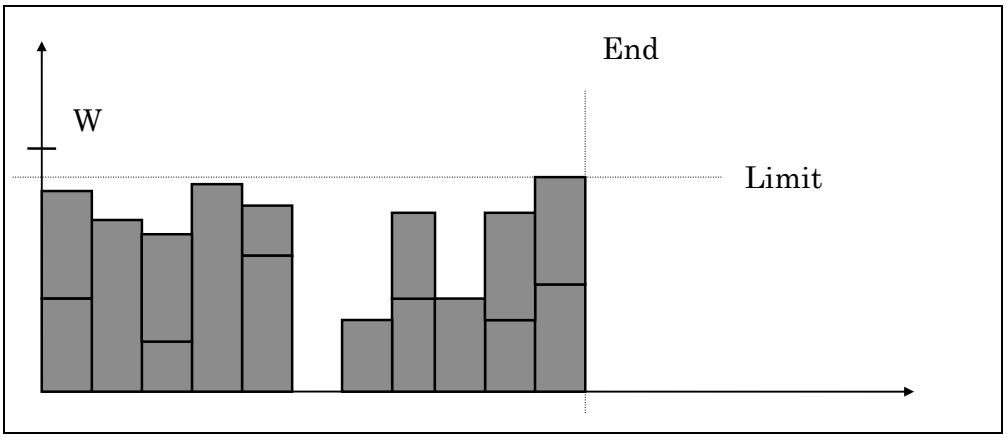

Figure 4: Bin packing model

4.5 Multi-peak

We now present another bound which improves the peak bound by taking the work period limit into account. In Figure 3 we have seen the resource profile generated by a schedule. The peak usage is the point where this profile takes its maximum value. If we look at the profile *W* time points before or after this time point, we may find other tasks scheduled there. Since one resource may only work for *W* units, these tasks must be assigned to resources which are not taken into account in the peak estimation. We can thus add the resource requirements of all points *W* time units apart and obtain a new profile which still gives a lower bound on the resource requirement of the original scheduling problem. An example is shown in Figure 5.

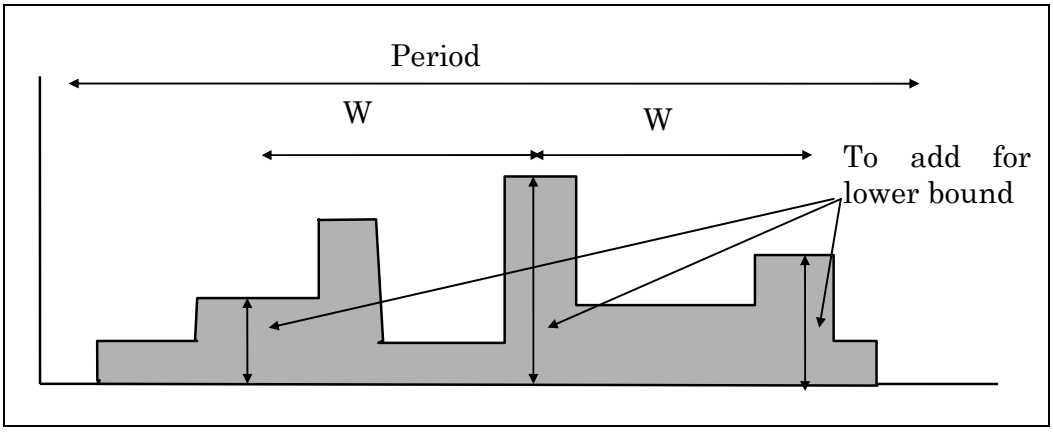

Figure 5: Improvement of peak bound

This is indeed true for any set of time points *W* units apart. A very simple way of creating this new profile for all time points is to shift the original profile by multiples of *W* units and to add the profiles. If the original scheduling period had length *Period*, we have to shift the profile

$$R = \frac{Period}{W}$$

times. The maximum height of this superimposed schedule gives a lower bound for the original problem. This situation is depicted in Figure 6.

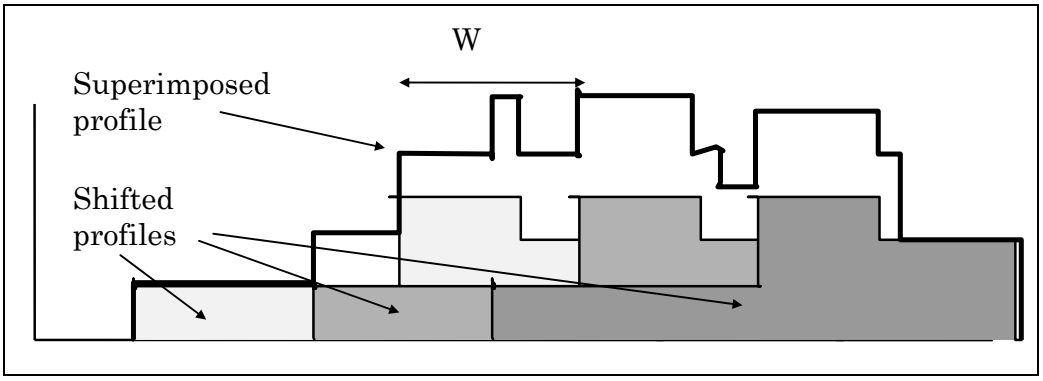

Figure 6: Superimposed profiles

Instead of extracting the profiles from the problem and computing their sum, we can use the cumulative constraint directly to create the superimposed profile. Each tasks is replicated R times with the same duration and resource use, but with different start dates which are shifted W units. Notes that this lower bound can be already calculated even if the start dates are not yet fixed by using domain variables for the start dates of the shifted tasks and link them to the start dates of the original tasks with equality constraints of the form

$$ShiftedStart_k = Start + k * W \quad k \in \{1..R\}$$

We call this new bound *multi-peak bound*, since it combines multiple peaks in the resource utilisation profile. If W is small compared to *Period*, it will give a much better bound than the simple peak bound. It will be useful in particular if the start dates of the tasks are quite restricted from the beginning. But if no constraints are placed on the start dates, the initial bound will be quite weak.

5. Discussion

In the previous section we have explained different lower bounds on the resource requirements in our scheduling problem. The bounds can be calculated incrementally or a priori with a single cumulative constraint. The bounds are obtained by relaxing some aspects of the problem, while keeping others.

- The surface bound takes the duration of the tasks and the length of the work period into account, but completely ignores the start times and the resource assignment.
- The peak bound is based on the start dates and the duration of the tasks, but ignores the work period length and the resource assignment.
- The bin packing bound uses the duration of the tasks, the work period limit and an estimation of the resource assignment, but ignores the start date of the tasks.

- The multi-peak bound uses the start dates, the duration and the work period limits, but ignores the resource assignment problem.

Comparing the different methods, we see the following taxonomy of bounds (shown in Figure 7).

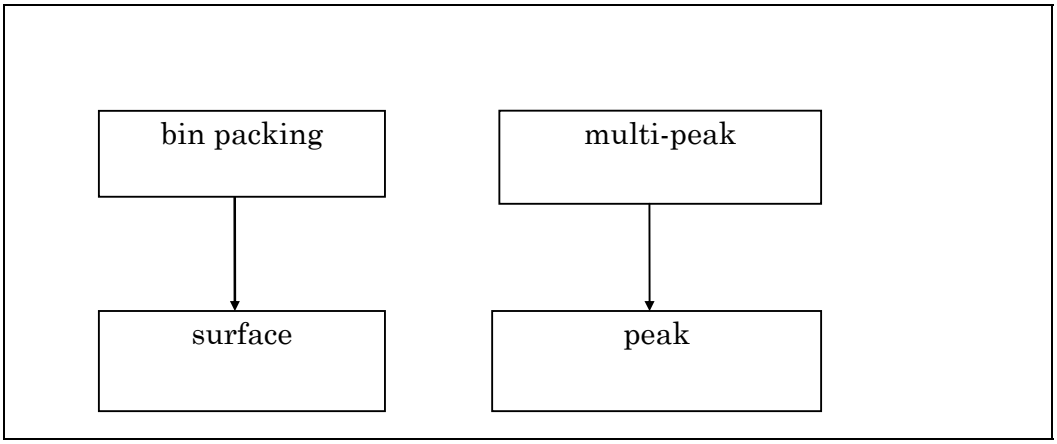

Figure 7: Lower bound taxonomy

The bin packing bound will be always equal to or better than the surface bound, and the multi-peak bound will be equal to or better than the peak bound. The other bounds are not directly comparable.
Depending on the importance of the different aspects of the scheduling problem we can choose one or several of the lower bounds for calculation. If for example the start dates are not constrained by other conditions, then the peak and multi-peak bound will give rather poor a priori bounds. If on the other hand, the problem is close to an assignment problem where the start dates are (nearly) fixed, the bin packing bound may give bad results, since it does not take the start dates into account.

6. Application Example

We now show how these lower bounds can be used in a real-life context of the TACT [Sim95b] project. This application combines planning, scheduling and assignment aspects of the operational fleet management in the food industry. The scheduling problem is concerned with assigning drivers and lorries to a number of trips between different locations. These trips are nearly fixed in time by producer/consumer constraints [SC94] on the arrival times at the destination sites. The start times can therefore only be changed in a very limited way. We have to assign lorries, drivers and other resources to the trips in order to satisfy a large number of different constraints. The working time of the drivers is restricted both by government regulation and a union contract. The start time for each driver can be changed within some limits. When generating a plan for a new time period, we are interested in obtaining lower bounds on the resource requirements without actually running the complete problem solver. For this purpose we use the multi-peak bounds, since the start

dates are very restricted and the working time limits of the drivers are shorter than the planning horizon. On a number of experiments, the multi-peak bound provided much better results than the other bounds presented above. The actual number of drivers required by the full problem solver will be typically 10% larger than the lower bound. This is not surprising since other constraints like location continuity and passive transport [Sim95] [Sim95a] [BKC94] cause the use of additional resources.
The lower bound is used in two ways. It provides an a priori estimation on the resource requirements before the problem solver is run and it is used as part of an explanation component if the problem solver can not find a solution for a given resource set. Given the complexity of the problem and the interrelation of the different constraints, a failure to find a solution in the constraint solver can not be easily explained. A number of redundant constraints and lower bounds help to explain the problem to the user and give an indication which constraints should be changed in order to make the problem solvable.
In the context of the TACT application, the multi-peak bound provides the best results. But as we explained above, this depends on the relative importance of the different aspects of the scheduling problem.

7. Summary

In this paper we have described several lower bounds for a resource scheduling and assignment problem. These lower bounds are obtained by relaxing some aspects of the problem formulation. The bounds may be used a priori or during solution generation in a optimisation problem. All lower bounds can be quite easily expressed in CHIP, using the global cumulative constraint.
We have also discussed the use of these lower bounds in a real-life scheduling and assignment problem. The problem is to estimate the number of drivers required to cover a number of trips within a complex transportation problem. The maximum working time of the drivers is restricted by regulations and contract. For this problem, the start times of the trips are nearly fixed by other constraints, so that the multi-peak lower bound gives rather good results. For problems with different characteristics, one of the other lower bounds may give better results.
For more general resource scheduling problems, the methods presented here can be easily extended. The methodology of selectively relaxing some problem aspects to obtain a simple lower bound is a well known technique and can be applied also in other situations.
The different programming examples in this paper also show the expressive power and flexibility of the global constraints in CHIP. We are using the diffn and the cumulative primitives to express very different conditions.

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

[BDP94] P. Boizumault, Y. Delon, L. Peridy
Planning Exams Using Constraint Logic Programming
PAP 94, London, April 1994

[CL91] E.G. Coffmann, G.S. Lueker
Probabilistic Analysis of Packing and Partitioning Algorithms
John Wiley & Sone, Chichester, 1991

[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, 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

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

[MT90] S. Martello, P. Toth
Knapsack Problems. John Wiley & Sons, Chichester, 1990

[SC94] H. Simonis, T. Cornelissens
Modelling Producer/Consumer Constraints
Constraint Programming CP95. Cassis, France. September 1995

[Sim95] H. Simonis
Modelling Machine Set-up Times in CHIP
Technical Report COSY/TR/95-2. March 1995

[Sim95a] H. Simonis
The Use of Exclusion Constraints to Handle Location Continuity Conditions
Technical Report COSY/TR/95-1. March 1995

[Sim95b] H. Simonis
A Complex Transportation Problem Solved with CHIP
PACT 95. Paris, France, April 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