Hybrid Modelling for Robust Solving

Brahim Hnich[1], Zeynep Kiziltan[2], Ian Miguel[3], and Toby Walsh[1*]

[1] Cork Constraint Computation Centre
University College Cork
Cork, Ireland
{brahim, tw}@4c.ucc.ie

[2] Computer Science Division
Department of Information Science
Uppsala University
Uppsala Sweden
Zeynep.Kiziltan@dis.uu.se

[3] Department of Computer Science
University of York
Heslington, York United Kingdom
ianm@cs.york.ac.uk

* We would like to thank C. Castro and S. Manzano for providing us with the real-life instances they used in their experiments. The first and the last authors are supported by Science Foundation Ireland. The third author is supported by UK-EPSRC grant number GR/N16129.

Abstract. We study a balanced academic curriculum problem and an industrial steel mill slab design problem. We show that these problems can be modelled in different ways, using both Integer Linear Programming (ILP) and Constraint Programming (CP) techniques, and consider the utility of each model. We also propose integrating the models to create hybrids that benefit from the complementary strengths of each model. Experimental results show that, especially in the case of hybrid CP/ILP models, the integration significantly increases the domain pruning, and decreases the run-time on many instances. Furthermore, a CP/ILP hybrid model gives a more *robust* performance in the face of varying instance data.

Keywords: Application, Modelling, Integration, Constraint Programming, and Integer Linear Programming.

Introduction

Many real-life problems can be modelled as constraint satisfaction problems (CSPs). For a given problem, many models can be developed, each having a different problem representation and employing a different solution method to solve the problem, as well as a different formulation of the constraints. This may make a model better or worse than any of the other models. It may also be the case that different models have complementary strengths. In such a case, alternative models of a problem can be integrated so as to obtain a new model that overcomes the disadvantages of one model with the advantages of the other one, and vice versa.

Integration of different models of a problem has been studied by Cheng *et al.* [Cheng et al. (1999)] and Smith [Smith (2001)], and a similar idea was previously suggested by Geelen [Geleen (1992)]. By integrating different models, the domain pruning carried out in each model may be improved significantly, giving a more powerful model than any of the participating models. However, this may increase the run-time due to the increase in the number of variables and constraints. Such an integration is achieved by introducing *channelling constraints* that link the variables of the participating models.

In this paper, we study two problems. The first problem is a balanced academic curriculum problem (BACP) proposed in [Castro et al. (2001)] and is prob030 in CSPLIB (www.csplib.org). The problem is to design an academic schedule by assigning periods to courses such that the academic load of each period is balanced. The second problem we consider is an industrial steel mill slab design problem (SMSDP) [Frisch et al. (2001a),Frisch et al. (2001b)]. The objective in this problem is to pack orders onto slabs such that the total slab capacity (and therefore wastage) is minimised. We show that these problems can be modelled in different ways, and argue why each model is useful. We then propose integrating the models so as to benefit from the complementary strengths of each model. Experimental results show that the integration significantly increases the domain pruning, and decreases the run-time on many instances. An additional benefit of the model integration is an increased *robustness* in the face of varying instance data.

The rest of this paper is organised as follows. In Section 1, we describe the problems. In Section 2, we study Integer Linear Programming (ILP) models of the BACP and SMSDP, followed by two Constraint Programming (CP) models for each problem in Section 3. Section 4 shows why and how some models are integrated. Then, in Section 5, we present the performance of the models on several instances of each problem. Section 6 reviews related work and Section 7 provides a summary and conclusion.

1 Problem Descriptions

The BACP proposed in [Castro et al. (2001)] is to design a balanced academic curriculum by assigning periods to courses in a way that the academic load of each period is balanced, i.e., as similar as possible. The curriculum must obey the following administrative and academic regulations:

- *Academic curriculum*: an academic curriculum is defined by a set of courses and a set of prerequisite relationships among them.
- *Number of periods*: courses must be assigned within a maximum number of academic periods.
- *Academic load*: each course has associated a number of credits or units that represent the academic effort required to successfully follow it.
- *Prerequisites*: some courses can have other courses as prerequisites.
- *Minimum academic load*: a minimum amount of academic credits per period is required to consider a student as full time.
- *Maximum academic load*: a maximum amount of academic credits per period is allowed in order to avoid overload.
- *Minimum number of courses*: a minimum number of courses per period is required to consider a student as full time.
- *Maximum number of courses*: a maximum number of courses per period is allowed in order to avoid overload.

The goal is to assign a period to every course in a way that the minimum and maximum academic load for each period, the minimum and maximum number of courses for each period, and the prerequisite relationships are satisfied. An optimal balanced curriculum minimises the maximum academic load for all periods. Note that we could consider other types of balance criterion such as minimising the sum of the academic load of all periods.

In the SMSDP, proposed in [Frisch et al. (2001a),Frisch et al. (2001b)], steel is produced by casting molten iron into slabs. A finite number, σ, of slab *sizes* is available. Each of the d input orders has two properties, a *colour* corresponding to the route required through the steel mill and a *weight*. An order cannot be split between slabs. The problem is to pack orders onto slabs such that the total weight of steel produced is minimised. There are two types of constraints:

1. **Capacity constraints.** The total weight of orders assigned to a slab cannot exceed the slab capacity.
2. **Colour constraints.** Each slab can contain at most p of k total colours (p is usually 2). This constraint arises because it is expensive to cut the slabs up in order to send them to different parts of the mill.

2 ILP Models

We present ILP models of the BACP and the SMSDP and discuss the similarities between the proposed models.

2.1 BACP

The inputs to the BACP are the integer sets *courses* and *periods* giving the courses and periods respectively, the integers a and and b giving the minimum and maximum allowed academic loads per period respectively, the integers c and d giving the minimum and maximum allowed number of courses per period respectively, the function *credit* giving the number of credits for each course, and the set *prereq* of pairs of courses $\langle i, j\rangle$ such that course i is a prerequisite of course j.

The ILP model that is already proposed in [Castro et al. (2001)] and shown in Figure 1 is as follows. The assignment of periods to courses is represented by a 2d 0/1 matrix of decision variables (CUR_{2d}). The meaning of $CUR_{2d}[i, j] = 1$ is that course i is assigned period j. The academic load is represented by a 1d matrix of decision variables ($LOAD$). The maximum academic load of all periods is represented by the decision variable C. The objective function simply minimises C. The first constraint in Figure 1 enforces that every course is assigned only one period because a 2d 0/1 matrix on its own does not ensure this property. The second constraint uses a weighted column sum expression to compute the academic load of each period. The third constraint guarantees that C is the maximum academic load of all periods. Enforcing the prerequisites constraint is however tricky. If course i is a prerequisite of course j, the posted constraint implies a strict lexicographical ordering between the i^{th} row and the j^{th} row of the 2d 0/1 matrix. Hence, this disallows the course j to be assigned a period that has lower index than the one assigned to course i. Enforcing the academic load and the amount of courses allowed per period is achieved through a set of inequalities.

Outputs:	$C \ in \ 0..maxint$ $LOAD[periods] \ in \ 0..maxint$ $CUR_{2d}[courses, periods] \ in \ 0..1$
Minimise:	C
Constraints:	% row sum to enforce that every course appears exactly in one period $\forall i \in courses \, . \, \sum_{j \in periods} CUR_{2d}[i,j] = 1$ % weighted column sum to compute the academic load for each period $\forall j \in periods \, . \, LOAD[j] = \sum_{i \in courses} credit(i) * CUR_{2d}[i,j]$ % C is the maximum academic load $C = max_{j \in periods} LOAD[j]$ % partial row sum to enforce the prerequisite constraints $\forall \langle i,j \rangle \in prereq \, . \, \forall k \in periods : k > 1 \, .$ $CUR_{2d}[j,k] \leq \sum_{r=1}^{k-1} CUR_{2d}[i,r]$ % set of inequalities restricting the academic load $\forall j \in periods \, . \, a \leq LOAD[j] \leq b$ % column sum to restrict the number of courses for each period $\forall j \in periods \, . \, c \leq \sum_{i \in courses} CUR_{2d}[i,j] \leq d$
Advantages:	all constraints are linear ease of statement of the academic load constraints (weighted column sum)
Disadvantages:	difficulty to state the prerequisite constraints

Fig. 1. BACP: An ILP model.

In this model, the academic load constraint of a period can easily be stated by a weighted column sum on the 2d 0/1 matrix. Despite the difficulty of stating the prerequisite constraints, all constraints of the model are linear. Therefore, ILP methods can easily be employed to solve this model. We refer to the model in Figure 1 as ILP_{BACP} when an ILP solver is used to solve the model.

2.2 SMSDP

We present an initial pure ILP model, then discuss how to improve the model through adding symmetry-breaking constraints and an implied constraint.

The inputs to the SMSDP are the integers j denoting the number of available orders, the integer p denoting the maximum number of colours a slab can accommodate, the function *weight* storing the weight of each order $o \in 1..j$, the function *colour* storing the colour of each order $o \in 1..j$, and the set of available slab sizes *sizes*.

The number of slabs is not fixed, but if we assume that the weight of each order does not exceed the maximum possible slab size, then the maximum number of slabs required is j. Thus, our objective can be viewed as finding an assignment of sizes to slabs and assigning orders to slabs in such a way that the capacity and colour constraints are satisfied and the cost function has its optimal value. Generally, in an optimal solution, a subset of the slabs won't be assigned to any orders. This can be modelled by adding the value 0 to the set of available sizes of the slabs so that when a slab is not necessary to solve the problem, it is assigned the size 0.

There are two immediate alternatives for modelling the assignment of sizes to slabs. First, a one-dimensional matrix can be used, indexed by $1..j$. Each element of this matrix is assigned a value from the set $sizes \cup \{0\}$, specifying a size for one of the available slabs. Initial experimentation with this approach yielded poor results. Hence, we use a two-dimensional matrix 0/1 matrix $Size_{2d}$ indexed by $1..j$ and $sizes \cup \{0\}$. The meaning of $Size_{2d}[s,z] = 1$ is that slab s gets assigned size z. However, we need to add constraints on the rows to make sure that each slab gets assigned exactly one size:

$$\forall s \in 1..j \, . \sum_{z \in sizes} Size_{2d}[s,z] = 1$$

The assignment of orders to slabs can be modelled as a 2d 0/1 matrix $Order_{2d}$ indexed by $1..j$ in both dimensions. The meaning of $Order_{2d}[o,s] = 1$ is that order o is assigned to slab s. Again, we need to enforce that each order gets assigned to exactly one slab by adding the following row-sum constraints:

$$\forall o \in 1..j \; . \sum_{s \in 1..j} Order_{2d}[o, s] = 1$$

The capacity constraints can be expressed as weighted row-sum constraints as follows:

$$\forall s \in 1..j \; . \sum_{o \in 1..j} Order_{2d}[o, s] * weight(o) \leq \sum_{z \in sizes} z * Size_{2d}[s, z]$$

The colour constraints are difficult to express on these variables. To cure this, we explicitly represent the set of colours on each slab. We introduce a 2d 0/1 matrix $Colour_{2d}$ indexed by $1..j$ and the set of available colours $colours$, where $Colour_{2d}[s, c] = 1$ means that there exists an order with colour c assigned to slab s. With these extra variables, the colour constraints can be expressed as:

$$\forall s \in 1..j \; . \sum_{c \in colours} Colour_{2d}[s, c] \leq p$$

However, we have to link the matrices $Order_{2d}$ and $Colour_{2d}$ to make sure that whenever an order o with colour c is assigned to a slab s, then colour c is on slab s. We could achieve this through the following set of channelling constraints:

$$\forall o \in 1..j, s \in 1..j \; . \; Order_{2d}[o, s] = 1 \rightarrow Colour_{2d}[s, colour(o)] = 1$$

However, we use this equivalent linear formulation instead:

$$\forall o \in 1..j, s \in 1..j \; . \; Order_{2d}[o, s] \leq Colour_{2d}[s, colour(o)]$$

Finally, the cost function can be expressed as:

$$Cost = \sum_{s \in 1..j} \sum_{z \in sizes} Size_{2d}[s, z] * z$$

Since, each order has to be assigned a slab, we can add the implied constraint that tightens the lower bound of $Cost$ to be the sum of the weights of all orders:

$$Cost \geq \sum_{o \in 1..j} weight(o)$$

An ILP model of the SMSDP is presented in Figure 2. We refer to this model as ILP_{SMSDP} whenever an ILP solver is used to solve it.

Symmetry There are three types of symmetry in the SMSDP [Frisch et al. (2001b)]:

- All slabs are indistinguishable: the slab sizes assigned to each of them may be permuted without affecting a (non) solution. This type of symmetry can be broken in the ILP_{SMSDP} by these set of linear constraints:

$$\forall s_1, s_2 \in 1..j \; . \; s_1 < s_2 \rightarrow \sum_{z \in sizes} Size_{2d}[s_1, z] * z \geq \sum_{z \in sizes} Size_{2d}[s_2, z] * z$$

  These make sure that the first slab gets assigned to the biggest size, the second slab get assigned the next biggest size, and so on. Note that we may impose an increasing ordering, instead.
- Any two identical orders are indistinguishable: the slabs assigned to each of them can be swapped freely without affecting a (non) solution. We can break such symmetries by enforce an ordering on the slabs assigned to symmetric orders using this set of linear constraints:

$$\forall o_1, o_2 \in 1..j \; . \; o_1 < o_2 \wedge weight(o_1) = weight(o_2) \wedge colour(o_1) = colour(o_2) \rightarrow$$
$$\sum_{s \in 1..j} Order_{2d}[o_1, s] * s \leq \sum_{s \in 1..j} Order_{2d}[o_2, s] * s$$

- If two slabs s_1 and s_2 have the same size, then the orders associated with these slabs can be freely swapped leading a symmetric (non) solution. We can use a lexicographic ordering constraint between the columns associated with slabs of the same size. However, these constraints have a high arity and so are not considered here.

Outputs:	$Cost\ in\ 0..maxint$ $Size_{2d}[1..j, sizes]\ in\ 0..1$ $Order_{2d}[1..j, 1..j]\ in\ 0..1$ $Colour_{2d}[1..j, colours]\ in\ 0..1$
Minimise:	$Cost$
Constraints:	% Objective function $Cost = \sum_{s \in 1..j} \sum_{z \in sizes} Size_{2d}[s, z] * z$ % Implied constraint: tightens the lower bound of Cost $Cost \geq \sum_{o \in 1..j} weight(o)$ % row sum to enforce that every order is assigned exactly 1 slab $\forall o \in 1..j\ .\ \sum_{s \in 1..j} Order_{2d}[o, s] = 1$ % row sum to enforce that every slab is assigned exactly 1 size $\forall s \in 1..j\ .\ \sum_{z \in sizes} Size_{2d}[s, z] = 1$ % capacity constraints $\forall o \in 1..j\ .\ \sum_{s \in 1..j} Order_{2d}[o, s] * weight(o) \leq \sum_{z \in sizes} Size_{2d}[s, z] * z$ % colour constraints $\forall s \in 1..j\ .\ \sum_{c \in colours} Colour_{2d}[s, c] \leq p$ % Channelling constraints $\forall o \in 1..j, s \in 1..j\ .\ Order_{2d}[o, s] \leq Colour_{2d}[s, colour(o)]$
Advantages:	all constraints are linear
Disadvantages:	None.

Fig. 2. SMSDP: An ILP model.

3 CP Models

We now explore some models which exploit non-binary (global) constraints, non-linear constraints and other features of CP models. Many scheduling, assignment, routing and other decision problems can be efficiently and effectively solved by CP models consisting of matrices of decision variables (so-called "matrix models" [Flener et al. (2001)]). Both the balanced academic curriculum and steel mill slab design problem can be successfully represented as such matrix models.

3.1 BACP

The ILP model in Figure 1 can be used directly as such a constraint model. We refer to the model in Figure 1 as $CP1_{BACP}$ when a CP solver is used to solve the model. The model $CP1_{BACP}$ has already been proposed in [Castro et al. (2001)]. However, as a CP model, it has constraints of large arity, on which bounds consistency is enforced by current constraint solvers, giving weak propagation. Hence, we consider an alternative matrix model that leads to stronger propagation.

The assignment of periods to courses can also be modelled using a 1d matrix indexed by courses and ranging over periods, which reduces the number of variables from $|courses| * |periods|$ to $|courses|$. However, with this variable modelling we find it difficult to state the academic load constraint. In order to state this constraint we need to enforce that the combined credit of the courses assigned to a particular period equals the load of that period. This would have been possible if CP solvers had a global "weighted occurrence" constraint [Frisch et al. (2001a)]. A weighted occurrence constraint is simply an occurrence constraint [Régin (1996)] (*gcc*) that takes into account the weight of each variable (in this case, the credits of a course variable) that is assigned the target value (in this case a period). In the absence of a weighted occurrence constraint, one can *channel* into the first CP model ($CP1_{BACP}$) where we can easily use weighted sums to express the academic load constraints. The restriction on the number of periods to be assigned a course is captured by the 1d matrix. However, unlike the academic load constraints, the prerequisite constraints are easily stated by enforcing an ordering on the courses that have a prerequisite relationship using binary constraints. Furthermore, the global occurrence constraint *gcc* can be used to enforce the restrictions on the amount of courses allowed per period.

Outputs:	C *in* $0..maxint$ $LOAD[periods]$ *in* $0..maxint$ $CUR_{1d}[courses]$ *in periods* $CUR_{2d}[courses, periods]$ *in* $0..1$
Minimise:	C
Constraints:	% using the 2d matrix to compute the academic load of each period $\forall j \in periods \,.\, LOAD[j] = \sum_{i \in courses} CUR_{2d}[i,j]credit(i)$ % C is the maximum academic load $C = max_{j \in periods} LOAD[j]$ % set of inequalities to enforce the prerequisite constraints $\forall \langle i,j \rangle \in prereq \,.\, CUR_{1d}[i] < CUR_{1d}[j]$ % set of inequalities restricting the academic load of each period $\forall j \in periods \,.\, a \leq LOAD[j] \leq b$ % occurrence constraint to restrict the number of courses for each period $gcc(c..d, periods, CUR_{1d})$ % channelling constraints between the 1d matrix and 2d 0/1 matrix $\forall i \in courses, j \in periods \,.\, CUR_{1d}[i] = j \leftrightarrow CUR_{2d}[i,j] = 1$
Advantages:	ease of statement of all problem constraints use of global constraints (better propagation)
Disadvantages:	increased number of variables and channelling constraints

Fig. 3. BACP: The $CP1_{BACP} + CP2_{BACP}$ integrated model.

Finally, the rest of the constraints and the objective function are the same as in model $CP1_{BACP}$. The model shown in Figure 3 uses this data representation. We refer to this integrated CP model as $CP1_{BACP}+CP2_{BACP}$.

3.2 SMSDP

Similarly to the BACP, the ILP model in Figure 2 can be used directly as a CP model for the SMSDP. When we solve this model with a constraint solver, we shall refer to it as $CP1_{SMSDP}$. However, as a CP model, it has too many constraints of high arity that do not propagate well.

We can use the same transformations that we employed for the BACP to reduce the number of variables in the previous model [Frisch et al. (2001a),Frisch et al. (2001b)]. The assignment of sizes to slabs ($Size_{2d}$) can be modelled as a 1d matrix $Size_{1d}$ indexed by $1..j$ and ranging over the set of sizes. Also, the assignment of slabs to orders ($Order_{2d}$) can be modelled as a 1d matrix $Order_{1d}$ indexed by $1..j$ and ranging over $1..j$. The colour matrix ($Colour_{2d}$), however, defines a relation between the slabs and the colours, as opposed to the previous matrices that define a total function, and hence we cannot replace it by a 1d matrix.

For the colour constraints, we use the same formulation as in the ILP model ILP_{SMSDP}, except that we use the following channelling constraints instead:

$$\forall o \in 1..j \,.\, Order_{1d}[o] = s \rightarrow Colour_{2d}[s, colour(o)] = 1$$

The cost function becomes:

$$Cost = \sum_{s \in 1..j} Size_{1d}[s]$$

We retain the implied constraint on the lower bound of *Cost* from the ILP model.

The capacity constraints are analogous to the academic load constraints of the BACP. Again, a weighted occurrence constraint would be necessary to state these constraints. So, we introduce $Order_{2d}[o,s]$, which is redundant with the matrix $Order_{1d}[o] = s$ and post the following channelling constraints:

$$\forall o \in 1..j, s \in 1..j \,.\, Order_{2d}[o,s] = 1 \leftrightarrow Order_{1d}[o] = s$$

This allows the ease of statement of the capacity constraints on the the 2d 0/1 $Order_{2d}$ matrix by weighted sum constraints as in the ILP model. Unlike the capacity constraints, the symmetry breaking constraints can be efficiently expressed as binary ordering constraints:

Outputs:	$Cost\ in\ 0..maxint$ $Size_{1d}[1..j]\ in\ sizes$ $Order_{1d}[1..j]\ in\ 1..j$ $Order_{2d}[1..j, 1..j]\ in\ 0..1$ $Colour_{2d}[1..j, colours]\ in\ 0..1$
Minimise:	$Cost$
Constraints:	% Objective function $Cost = \sum_{s \in 1..j} Size_{1d}[s]$ % Implied constraint: tightens the lower bound of Cost $Cost \geq \sum_{o \in 1..j} weight(o)$ % stating capacity constraints using $Order_{2d}$ $\forall o \in 1..j\ .\ \sum_{s \in 1..j} Order_{2d}[o, s] * weight(o) \leq Size_{1d}[s]$ % colour constraints $\forall s \in 1..j\ .\ \sum_{c \in colours} Colour_{2d}[s, c] \leq p$ % Channelling constraints $\forall o \in 1..j, s \in 1..j\ .\ Order_{1d}[o] = s \rightarrow Colour_{2d}[s, colour(o)] = 1$ % Symmetry-breaking constraints $\forall s_1, s_2 \in 1..j\ .\ s_1 < s_2 \rightarrow Size_{1d}[s_1] \geq Size_{1d}[s_2]$ $\forall o_1, o_2 \in 1..j\ .\ o_1 < o_2 \wedge weight(o_1) = weight(o_2) \wedge colour(o_1) = colour(o_2)$ $\rightarrow Order_{1d}[o_1] \leq Order_{1d}[o_2]$ % Order Channelling constraints $\forall o \in 1..j, s \in 1..j\ .\ Order_{1d}[o] = s \leftrightarrow Order_{2d}[o, s] = 1$
Advantages:	ease of statement of all problem constraints. binary symmetry-breaking constraints.
Disadvantages:	increased number of variables and constraints.

Fig. 4. SMSDP: the $CP1_{SMSDP} + CP2_{SMSDP}$ integrated model.

- The slab symmetry is broken by this set of constraints:

$$\forall s_1, s_2 \in 1..j\ .\ s_1 < s_2 \rightarrow Size_{1d}[s_1] \geq Size_{1d}[s_2]$$

- The identical orders symmetry can be broken using this set of binary constraints:

$$\forall o_1, o_2 \in 1..j\ .\ o_1 < o_2 \wedge weight(o_1) = weight(o_2) \wedge colour(o_1) = colour(o_2) \rightarrow$$

$$Order_{1d}[o_1] \leq Order_{1d}[o_2]$$

- The symmetry that arises when two slabs are of the same size can be expressed as follows:

$$\forall s_1 < s_2 \in 1..j\ .\ Size_{1d}[s_1] = Size_{1d}[s_2] \rightarrow Order_{2d}[-, s_1] \leq_{lex} Order_{2d}[-, s_2]$$

  where $Order_{2d}[-, s]$ denotes the s^{th} column of matrix $Order_{2d}$ and $\leq_{lex}$ denotes lexicographic ordering. However, we do not use these symmetry-breaking constraints in the CP model since OPL, the solver we use later in our experiments, does not offer a lexicographic ordering constraint between vectors of variables and it would be prohibitively expensive to expand out this ordering constraint into its more primitive parts.

The second CP model, which is an integrated model, is shown in Figure 4 and will be referred to as $CP1_{SMSDP} + CP2_{SMSDP}$.

4 Hybrid ILP/CP Models

Thus far, we have introduced ILP and CP models for both the BACP and SMSDP. We now consider the integration of these models to form hybrids for each problem type, in order to benefit from the complementary strengths of each model.

4.1 BACP

The CP model $CP1_{BACP} + CP2_{BACP}$ has two strong points:

1. The prerequisite constraints are stated using binary constraints where maintaining arc-consistency can be efficiently achieved.
2. The restriction on the number of courses for each period is stated using a single global constraint where generalised-arc consistency is efficiently maintained.

However, to state the academic load constraints we had to introduce a redundant 2d 0/1 matrix (CUR_{2d}) and extra channelling constraints.

The model ILP_{BACP}, on the other hand, suffers from the following:

1. The prerequisite constraints in model ILP_{BACP} are stated using partial row sum constraints. There are more of these constraints than their equivalents in model $CP1_{BACP} + CP2_{BACP}$ and they are of higher arity.
2. To enforce the restriction on the number of courses for each period, we pose, in model ILP_{BACP}, $2*|periods|$ constraints of arity $|courses|$. Whereas, in model $CP1_{BACP} + CP2_{BACP}$ we pose a single constraint.

Nevertheless, the academic load constraints are easily expressed in the model ILP_{BACP} and all constraints are linear, which allows an ILP solver to be used to solve the problem.

In order to benefit from the effectiveness of each model, we propose integrating the models into the hybrid model $ILP_{BACP}+CP2_{BACP}$ (shown in Figure 5) by channelling the variables of the participating models. The disadvantages of this integration are the increased number of variables, and additional channelling constraints to be processed.

In $ILP_{BACP}+CP2_{BACP}$, we specify the prerequisite constraints and the restriction of number of courses per period on the 1d matrix, while we specify the academic load constraints on the 2d matrix. We also decompose the channelling constraints:

$$\forall i \in courses, j \in periods \,.\, CUR_{1d}[i] = j \leftrightarrow CUR_{2d}[i,j] = 1$$

into:

$$\forall i \in courses \,.\, \sum_{j \in periods} CUR_{2d}[i,j] = 1$$

and

$$\forall i \in courses \,.\, CUR_{2d}[i, CUR_{1d}[i]] = 1$$

which is logically equivalent to the following:

$$\forall i \in courses, j \in periods \,.\, CUR_{1d}[i] = j \rightarrow CUR_{2d}[i,j] = 1$$

but, the former poses fewer constraints, which use variable indexing that is more efficient than the implication constraint.

This integration allows ILP_{BACP} to benefit from the power of CP in efficiently handling binary and global constraints. The CP part of the model benefits from ILP_{BACP}'s power in the statement of the academic load constraints. Moreover, this integration allows a hybrid solution method to be employed to solve the problem.

4.2 SMSDP

Similarly to the BACP, the capacity constraints are easily stated in model ILP_{SMSDP} by weighted row sum constraints, whereas we had to introduce extra variables and channelling constraints in the second CP model to state these constraints. The situation is reversed if we consider the symmetry-breaking constraints. In the second CP model, we use binary ordering constraints, whereas on the ILP model we have high arity constraints. In order to benefit from the effectiveness of each model, we propose integrating the models into the hybrid model $ILP_{SMSDP}+CP2_{SMSDP}$ (presented in Figure 6) by channelling the variables of the participating models. The disadvantages of these integrations are the increased number of variables, and additional channelling constraints to be processed. We enforce the capacity constraints on the ILP model, the symmetry-breaking constraints on the CP model and colour channelling constraints on both models. Note that best results are obtained when the cost function is also imposed on both models. This is because one representation gives more propagation in some situation than the other, and vice-versa. We also decompose the order and size channelling constraints as we did for the BACP.

Outputs:	C *in* 0..*maxint* $LOAD[periods]$ *in* 0..*maxint* $CUR_{1d}[courses]$ *in periods* $CUR_{2d}[courses, periods]$ *in* 0..1
Minimise:	C
Constraints:	% using the 2d matrix to compute the academic load of each period $\forall j \in periods\,.\; LOAD[j] = \sum_{i \in courses} CUR_{2d}[i, j] * credit(i)$ % row sum to enforce that every course appears exactly in one period $\forall i \in courses\,.\; \sum_{j \in periods} CUR_{2d}[i, j] = 1$ % C is the maximum academic load $C = max_{j \in periods} LOAD[j]$ % using the 1d matrix to state the prerequisite constraints $\forall \langle i, j\rangle \in prereq\,.\; CUR_{1d}[i] < CUR_{1d}[j]$ % set of inequalities restricting the academic load of each period $\forall j \in periods\,.\; a \leq LOAD[j] \leq b$ % using the 1d matrix to state a global constraint $gcc(c..d, periods, CUR_{1d})$ % channelling constraints between the 1d matrix and 2d 0/1 matrix $\forall i \in courses\,.\; CUR_{2d}[i, CUR_{1d}[i]] = 1$
Advantages:	ease of statement of all problem constraints use of global constraints (better propagation)
Disadvantages:	redundant variables extra channelling constraints

Fig. 5. BACP: the $ILP_{BACP} + CP2_{BACP}$ hybrid model.

5 Experimental results

In order to evaluate the performances of the proposed models, we carried out some experiments by implementing the models in OPL [Van Hentenryck (1999)]. The CP models are solved using Ilog Solver 5.2 and the ILP models are solved using CPLEX 7.5.0, both called via OPL. For the hybrid models a hybrid solution procedure, composed of Solver and CPLEX, is used and is also called by OPL. The hybrid solution procedure works as follows. The search is conducted in a CP style, but at each node of the search tree the objective function and the set of all linear constraints are passed to CPLEX, where linear relaxation is used to produce a lower bound. The lower bound is used by Solver to add a constraint forcing the objective function to take a value less than or equal to the lower bound (assuming a minimisation problem). Channelling constraints (handled by Solver) connect the ILP 0/1 variables participating in the objective function and all linear constraints with the other CP variables. These constraints ensure that, at each node of the search tree, the 0/1 ILP variables properly reflect the state of the objective function and the linear constraints that will be passed to CPLEX for the computation of a lower bound.

5.1 BACP

Figures 7 and 8 show the results on the three real-life instances used in [Castro et al. (2001)] to find the optimal solution and prove optimality, respectively. In the instances, we have 8, 10, and 12 periods, and 46, 42, and 66 courses, respectively. We adopt the same branching heuristic for $CP1_{BACP}$ as in [Castro et al. (2001)], which groups the variables by periods and assigns the value 1 first. Note that this branching heuristic achieves the best results in [Castro et al. (2001)]. As for $CP1_{BACP} + CP2_{BACP}$, we use the *smallest-domain* branching strategy on CUR_{1d}, choosing values in lexicographical order. The model $ILP_{BACP} + CP2_{BACP}$ uses the same labelling strategy as model $CP1_{BACP} + CP2_{BACP}$ for the 8 and 10 period instances, and uses the same labelling strategy as the model $CP1_{BACP}$ for the 12 periods instance.

Proving optimality was difficult for both CP models ($CP1_{BACP}$ and $CP1_{BACP} + CP2_{BACP}$). We observe that $ILP_{BACP} + CP2_{BACP}$ finds an optimal solution and proves optimality quicker than ILP_{BACP}, demonstrating that a hybrid model can be more effective despite the increased number of variables and additional

Outputs:	$Cost\ in\ 0..maxint$ $Size_{1d}[1..j]\ in\ sizes$ $Size_{2d}[1..j, sizes]\ in\ 0..1$	$Colour_{2d}[1..j, colours]\ in\ 0..1$ $Order_{1d}[1..j]\ in\ 1..j$ $Order_{2d}[1..j, 1..j]\ in\ 0..1$
Minimize:	$Cost$	
Constraints:	% Objective function on 1d matrix and 2d matrix $Cost = \sum_{s \in 1..j} Size_{1d}[s]$ $Cost = \sum_{s \in 1..j} \sum_{z \in sizes} z * Size_{2d}[s, z]$ % Implied constraint: tightens the lower bound of Cost $Cost \geq \sum_{o \in 1..j} weight(o)$ % row sum to enforce that every order is assigned exactly 1 slab $\forall o \in 1..j\ .\ \sum_{s \in 1..j} Order_{2d}[o, s] = 1$ % row sum to enforce that every slab is assigned exactly 1 size $\forall s \in 1..j\ .\ \sum_{z \in sizes} Size_{2d}[s, z] = 1$ % capacity constraints $\forall o \in 1..j\ .\ \sum_{s \in 1..j} Order_{2d}[o, s] * weight(o) \leq sum_{z \in sizes} z * Size_{2d}[s, z]$ % colour constraints $\forall s \in 1..j\ .\ \sum_{c \in colours} Colour_{2d}[s, c] \leq p$ % Colour Channelling constraints $\forall o \in 1..j\ .\ Colour_{2d}[Order_{1d}[o], colour(o)] = 1$ $\forall o \in 1..j, s \in 1..j\ .\ Order_{2d}[o, s] \leq Colour_{2d}[s, colour(o)]$ % Order Channelling constraints $\forall o \in 1..j\ .\ Order_{2d}[o, Order_{1d}[o]] = 1$ % Order Channelling constraints $\forall s \in 1..j\ .\ Size_{2d}[s, Size_{1d}[s]] = 1$ % Symmtery-breaking constraints using 1d matrix $\forall s_1, s_2 \in 1..j\ .\ s_1 < s_2 \rightarrow Size_{1d}[s_1] \leq Size_{1d}[s_2]$ $\forall o_1, o_2 \in 1..j\ .\ o_1 < o_2 \wedge weight(o_1) = weight(o_2) \wedge colour(o_1) = colour(o_2)$ $\rightarrow Order_{1d}[o_1] \leq Order_{1d}[o_2]$	
Advantages:	ease of statement of the symmetry-breaking constraints ease of statement of the capacity constraints	
Disadvantages:	redundant variables extra channelling constraints	

Fig. 6. SMSDP: the $(ILP_{SMSDP} + CP2_{SMSDP})$ hybrid model.

channelling constraints. Note that the proof of optimality is immediate for the model $ILP_{BACP} + CP2_{BACP}$ and the model ILP_{BACP}. In this hybrid integration, the CP model is essential in reducing the search space while the ILP model, via relaxation, is essential for bounding and guiding the search. The second best model is the ILP model ILP_{BACP}, which out-performed all the pure CP models. Finally, the integrated CP model $CP1_{BACP} + CP2_{BACP}$ has less failures and run-time than the CP model $CP1_{BACP}$ on the first instance. This is due to the increase in the amount of pruning, which led to a reduction in the search space, compensating for the increased number of variables and constraints.

The CP solver used in the experiments is Ilog Solver and the ILP solver is CPLEX, both called via OPL. In [Castro et al. (2001)], the authors used OZ to solve the model $CP1_{BACP}$, and *lp-solve*[4] to solve the model ILP_{BACP}. Their experiments are concerned with finding the optimal solution but not with proving optimality. They showed that with varying the default labelling heuristic of the model $CP1_{BACP}$, the three instances were solved very quickly, but lp-solve could only solve the first instance. However, our experiments using OPL showed the opposite, as seen in Figure 7. We conjecture that the difference might be connected to the superiority of CPLEX over lp_solve. Note that we used the same labelling strategy for the model $CP1_{BACP}$ as in [Castro et al. (2001)].

5.2 SMSDP

We tested a number of variations of the ILP model. The experiments are split into two categories. In the first, we use the decision variables in $Size_{2d}$ as our branching variables. In the second, we use the variables in $Order_{2d}$. We ruled out the case where we use the variables in $Colour_{2d}$ because this leads to comparatively very poor results. Within each category, we tested four ILP models:

[4] An ILP solver: available free at ftp://ftp.ics.ele.tue.nl/pub/lp_solve

Model	Results	8 periods	10 periods	12 periods
ILP_{BACP}	runtime	1.27	3.17	13.77
	failures	N/A	N/A	N/A
$CP1_{BACP}$	runtime	148.44	-	-
	failures	5828091	-	-
$CP1_{BACP} + CP2_{BACP}$	runtime	30.59	-	-
	failures	304963	-	-
$ILP_{BACP} + CP2_{BACP}$	runtime	0.38	3.06	1.20
	failures	235	6068	11452

Fig. 7. BACP: finding an optimal solution.

Model	Results	8 periods	10 periods	12 periods
ILP_{BACP}	runtime	1.27	3.17	13.77
	failures	N/A	N/A	N/A
$CP1_{BACP}$	runtime	-	-	-
	failures	-	-	-
$CP1_{BACP} + CP2_{BACP}$	runtime	-	-	-
	failures	-	-	-
$ILP_{BACP} + CP2_{BACP}$	runtime	0.38	3.06	1.20
	failures	235	6068	11452

Fig. 8. BACP: Finding an optimal solution and proving optimality.

1. Basic: ILP_{SMSDP}, shown in Figure 2.
2. Basic+O: ILP_{SMSDP} augmented with the symmetry-breaking constraints for the identical orders symmetry.
3. Basic+$S_\downarrow$ and Basic+$S_\uparrow$: ILP_{SMSDP} augmented with the symmetry-breaking constraints for the slabs symmetry, where $S_\downarrow$ denotes the decreasing slab symmetry-breaking constraints and $S_\uparrow$ for the increasing version.
4. Basic+O+$S_\downarrow$ and Basic+O+$S_\uparrow$: ILP_{SMSDP} augmented with the symmetry-breaking constraints for both the slabs and identical orders symmetry.

Figures 9 and 10 present results generated from (small) subsets of industrial data, implemented in OPL [Van Hentenryck (1999)] and solved using the CPLEX solver. From the results we observe the following:

- All the models were sensitive to the instance distributions. They might perform well on some instances, but poorly on others.
- The branching strategy that uses $Size_{2d}$ solves 7 out of the 8 instances faster than the branching strategy that uses $Order_{2d}$.
- Using both symmetry-breaking constraints at the same time give the worst results on all instances.
- Among the models that use the branching strategy on $Size_{2d}$, the best is either the basic model, or the basic model augmented with either the order symmetry-breaking constraints or the (increasing) slab symmetry-breaking constraints.

We also experimented with several variations derived from the CP models $CP1_{SMSDP}$ and $CP1_{SMSDP} + CP2_{SMSDP}$, such as using the increasing slab symmetry-breaking constraints, and different branching strategies. For clarity, we show the best results achieved. The results for $CP1_{SMSDP}$ are presented in Figure 11. The labelling strategy used is to try to assign each order in turn from the largest to the smallest slab (the decreasing slab symmetry-breaking constraints are used). The results for $CP1_{SMSDP} + CP2_{SMSDP}$ are presented in Figure 12. The labelling strategy used here is to branch on the variables in $Order_{1d}$ with the smallest domain heuristic. Values are assigned in increasing order, which is equivalent to trying the largest slab first, since the decreasing slab symmetry-breaking constraints are again used.

It is immediately clear that the integrated model $CP1_{SMSDP} + CP2_{SMSDP}$ outperforms the CP model $CP1_{SMSDP}$. As per BACP, this demonstrates that a naive use of an ILP model with a CP solver typically leads to poor performance. Contrary to the results obtained with the ILP models, symmetry-breaking leads to a marked improvement in performance. Finally, except for the first two instances, both CP models are not competitive with the ILP models. Nevertheless, the integrated CP model quickly finds near-optimal solutions.

Orders	Optimal	Basic	Basic+O	Basic+$S_\downarrow$	Basic+$S_\uparrow$	Basic+O+$S_\downarrow$	Basic+O+$S_\uparrow$
12	77	88.82	15.11	63.10	8.83	21.50	19.48
16	99	25.34	87.48	39.01	10.68	>120	97.90
18	110	1.98	>100	36.46	>100	49	>100
19	115	131.87	2.6	12.4	270.38	>350	87.69
20	122	5.91	8.44	97.58	>350	>100	>350
21	135	11.96	206	>216	>300	>216	>300
25	166	815.58	65.24	>1000	>1000	>1000	>1000
30	195	2991	1180	>3000	>3000	>3000	>3000

Fig. 9. SMSDP: Runtimes in seconds. Branching is on $Size_{2d}$.

Orders	Optimal	Basic	Basic+O	Basic+$S_\downarrow$	Basic+$S_\uparrow$	Basic+O+$S_\downarrow$	Basic+O+$S_\uparrow$
12	77	104.45	18.32	66.26	9.90	24.84	20.29
16	99	29.84	105.29	39.98	11.62	267.57	102.12
18	110	2.36	>350	37.89	>350	58.58	3250
19	115	153.73	3.14	12.60	285	1388.27	99.42
20	122	6.89	9.85	75.26	>250	262	>250
21	135	14.03	241.74	>300	>300	>300	>300
25	166	945	74.11	>1000	>1000	>1000	>1000
30	195	>3500	1098.58	>3500	>3500	>3500	>3500

Fig. 10. SMSDP: Runtimes in seconds. Branching is on $Order_{2d}$.

Our final set of experiments tested multiple variations of hybrid ILP/CP models by varying the labelling strategy and the constraint formulation of the slab symmetry-breaking constraints of model $ILP_{SMSDP} + CP2_{SMSDP}$. Again for clarity, we present in Figure 13 the best results achieved where a CP solver (ILOG) and an ILP solver (CPLEX) are used to solve the model. The branching strategy is the smallest-domain heuristic used on the variables of $Size_{1d}$ of the model $ILP_{SMSDP} + CP2_{SMSDP}$ presented in Figure 6.

The results show that the hybrid model outperforms all other models in terms of run-time and failures. In this hybrid integration, we observe a similar behavior to the BACP, where the CP model is essential in reducing the search space while the ILP model with its relaxation is essential for bounding and guiding the search.

5.3 Robustness

Our results have provided evidence for the superiority, in terms of search efficiency, of a hybrid CP/ILP model over a pure CP or pure ILP approach for solving both the BACP and SMSDP. The results also show that, on these problems, the CP/ILP hybrid offers a more *robust* model. We use the term robust in a different sense to [Ginsberg et al. (1998)], who define the degree of robustness of a *solution* according to the size of the repair necessary given a number of modifications to the variable assignments. In contrast, and following the informal usage in [Rodosek et al. (1998)], we define a *robust model* for a class of problems reliant on instance data to be a model whose performance does not vary greatly with the instance data distribution. From Figures 7, 8 and 13 it is clear that the CP/ILP hybrid is the most robust model considered here. Given the experimental evidence herein, and supporting evidence in [Rodosek et al. (1998)], we conjecture that such hybrids will provide robust models in general. However, we stress that it is not sufficient simply to combine any two ILP and CP models. The best hybrid models were carefully designed to combine the complementary strengths of the two approaches.

6 Related Work

It has been show in the literature that certain classes of problems are best solved by careful integration of CP and Operations Research (OR) techniques. One popular method is to combine Lagrangian relaxations and CP [Benoist (2001),Sellman et al. (2001)]. The idea is to relax constraints in the objective function. In [Benoist (2001)], the authors study hybrid algorithms combining Lagrange relaxations and constraint programming on a Travelling Tournament Problem (TTP) combining round-robin assignment and travel optimization. Round-robin tournament problems are best solved using CP with global constraints while travelling salesman

Orders	Optimal	Runtime (sec)	Failures
12	77	— (Best found: 79 in 1.01)	— (Best found: 79 in 18148)
16	99	— (Best found: 112 in 89.14)	— (Best found: 122 in 392933)
18	110	— (Best found: 121 in 17.99)	— (Best found: 121 in 109016)
19	115	— (Best found: 121 in 6.70)	— (Best found: 121 in 71778)
20	122	— (Best found: 152 in 6.50)	— (Best found: 152 in 71030)
21	135	—	—
25	166	—	—
30	195	—	—

Fig. 11. SMSDP: Runtimes/Failures for $CP1_{SMSDP}$: Branching is on $Order_{2d}$, order by order assigning the value 1 first. A dash means no results are returned after 1 hour.

Orders	Optimal	Runtime (sec)	Failures
12	77	0.28	925
16	99	6.51	49055
18	110	144.9	1046744
19	115	302.97	2128822
20	122	2014.04	13430974
21	135	21.77	143250
25	166	1215.83	6206154
30	195	— (Best found: 197 in 12.81)	— (Best found: 197 in 39142)

Fig. 12. SMSDP: Runtimes/Failures for $CP1_{SMSDP}$+$CP2_{SMSDP}$: Branching is on $Order_{1d}$ (smallest-domain heuristic). A dash means no results are returned after 1 hour.

problems are best solved using IP techniques. The hybrid algorithm is shown to be very effective in solving the TTP problem. In [Sellman et al. (2001)], the authors introduce an algorithm where linear optimization techniques can strengthen their propagation abilities via Lagrangian relaxation. The algorithm is tested on a set of problems originating from multimedia applications. The results show the superiority of the combined method to pure CP and OR approaches. Another interesting work is the framework of Mixed Logical Linear Programming proposed by Hooker *et al* [Hooker et al. (1999)]. In [Hooker et al. (1999)], the authors propose a declarative modelling framework in which the structure of the constraints indicates how CP and LP can interact to solve the problem. This modelling framework has not only modelling advantages but can often permit more rapid solution than traditional Mixed Integer Linear Programming solvers. The authors of [El Sakkout et al. (2000)] consider a scheduling problem where the optimization requirement may be captured using a linear optimization function over linear constraints. However, the disjunctive nature of the resource constraints impairs traditional mathematical programming approaches. Therefore the authors decompose the problem, and solve the subproblems by CP and ILP techniques. It is shown to be the most effective way of solving the problem.

7 Conclusion

We have proposed a variety of different models of the Balanced Academic Curriculum Problem and the Steel Mill Slab Design Problem. Some of the models were created with an integer linear programming solver in mind, others with a constraint programming solver. We also considered hybrid CP/ILP models which employ a hybrid solution method combining an ILP solver with a CP solver, propagating information from one part of the model to the other using channelling constraints. Each model was evaluated experimentally on real instances (subsets of real instances in the case of SMSDP). Our results show that, for both problems, the hybrid CP/ILP models are the most successful, both in solving individual problems and as the instance data varies (*robustness*). A good hybrid model uses the strengths of one approach (CP or ILP) to negate the weaknesses of the other, providing a much more efficient overall model. This is counter to the perceived wisdom that a good model should minimise the number of variables and constraints.

What general lessons can we learn from this modelling exercise? First, when constraints are difficult to specify in a particular model, we should consider channelling into a second model in which these constraints are easier to specify and reason about. Second, whilst constraint programming models can be best at finding optimal or near-optimal solutions, integer linear programs may be better for proving optimality. Hybrid CP and

Orders	Optimal	Runtime (sec)	Failures
12	77	0.52	328
16	99	0.53	154
18	110	0.67	177
19	115	1.91	492
20	122	2.97	765
21	135	4.84	1052
25	166	7.6	1351
30	195	15.85	1615

Fig. 13. SMSDP: Run-times/Failures for $ILP_{SMSDP}+CP2_{SMSDP}$: Branching is on $Size_{1d}$ (smallest-domain heuristic).

ILP models or a two phase approach may therefore be advantageous. Third, CP and ILP tools should provide primitives for channelling between models. In addition to being able to specify such constraints compactly, such primitives can permit efficient constraint propagation between models. Finally, we can often profitably combine different problem representations, as well as different solution methods. Each view of the problem and solution method can exploit different aspects of the problem. Careful integration of different models can result in better models despite the increase in the number of variables and constraints.

References

[Castro et al. (2001)] C. Castro and S. Manzano. (2001). Variable and value ordering when solving balanced academic curriculum problem. In: *Proc. of the ERCIM WG on constraints.*

[Cheng et al. (1999)] B.M.W. Cheng, K.M.F. Choi, J.H.M. Lee, and J.C.K. Wu. (1999).Increasing constraint propagation by redundant modelling: An experience report. *Constraints*, 4:167–192.

[Flener et al. (2001)] P. Flener, A. Frisch, B. Hnich, Z. Kızıltan, I. Miguel, and T. Walsh. (2001). Matrix Modelling. In: *Proc. of the CP-01 Workshop on Modelling and Problem Formulation.* International Conference on the Principles and Practice of Constraint Programming.

[Frisch et al. (2001a)] A.M. Frisch, I. Miguel, and T. Walsh. (2001). Modelling a Steel Mill Slab Design Problem. In: *proc. of the IJCAI-01 Workshop on Modelling and Solving Problems with Constraints.*

[Frisch et al. (2001b)] A.M. Frisch, I. Miguel, and T. Walsh. (2001). Symmetry and Implied Constraints in the Steel Mill Slab Design Problem. In: *proc. of the CP-01 Workshop on Modelling and Problem Formulation.*

[Ginsberg et al. (1998)] M.L. Ginsberg, A. J. Parkes, and A. Roy. (1998). Supermodels and Robustness. In: *proc. of AAAI/IAAI-98*, pages 334–339.

[Geleen (1992)] P.A. Geelen. (1992). Dual viewpoint heuristics for binary constraint satisfaction problems. In *Proc. of ECAI'92*, pp. 31–35.

[El Sakkout et al. (2000)] H. El Sakkout and M. Wallace. (2000). Probe Backtrack Search for Minimal Perturbation in Dynamic Scheduling . In *Constraints*, 5(4):359-388.

[Hooker et al. (1999)] J.N. Hooker, G. Ottosson, E.S. Thorsteinsson, and H-J. Kim. (1999). On Integrating Constraint Propagation and Linear Programming for Combinatorial Optimization. In *AAAI/IAAI*, pp. 136-141.

[Régin (1996)] J-C. Régin. (1996). Generalized arc consistency for global cardinality constraints. In *Proc. of the Eighth National Conference on Aritficial Intelligence*, pp. 25–32.

[Rodosek et al. (1998)] R. Rodosek and M. Wallace. (1998). A Generic Model and Hybrid Algorithm for Hoist Scheduling Problems. In *Proc. of CP-98*, pp. 385–399.

[Sellman et al. (2001)] M. Sellmann and T. Fahle. (2001). CP-based Lagrangian Relaxation for a Multimedia Application. In *Proc. CP-AI-OR'01.*

[Smith (2001)] B.M. Smith. (2001). Dual models in constraint programming. Research Report 2001.02, University of Leeds (UK), School of Computing.

[Benoist (2001)] T. Benoist, F. Laburthe, and B. Rottembourg. (2001). Lagrange Relaxation and Constraint Programming Collaborative Schemes for Travelling Tournament Problems. In *Proc. of CP-AI-OR'01.*

[Van Hentenryck (1999)] P. Van Hentenryck. (1999). *The* OPL *Optimization Programming Language.* The MIT Press.