Proceedings CPAIOR'02

Modelling a Balanced Academic Curriculum Problem

Brahim Hnich
Uppsala University
Department of Information Science
Box 513 SE-75120, Uppsala, Sweden
email: `Brahim.Hnich@dis.uu.se`

Zeynep Kızıltan
Uppsala University
Department of Information Science
Box 513 SE-75120, Uppsala, Sweden
email: `Zeynep.Kiziltan@dis.uu.se`

Toby Walsh
Cork Constraint Computation Center
University College Cork
Cork, Ireland
email: `tw@4c.ucc.ie`

Abstract

In this paper, we study a balanced academic curriculum problem. We show that this problem can be modelled in different ways, and argue why each model is useful. We also propose integrating the models so as to benefit from the complimentary strengths of each model. Experimental results show that the integration significantly increases the domain pruning, and even decreases the run-time on many instances. General lessons are learnt 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 (CP) models can be best at finding optimal or near-optimal solutions, integer linear programming (ILP) models may be better for proving optimality. Hybrid CP and ILP models or a two phase approach may therefore be advantageous. Third, CP and ILP tools should provide primitives for channelling between models. Finally, we can often profitably combine different problem representations, as well as different solution methods.

1 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 different formulation of the constraints. This may make a model be better or worse than any of the other models. It may also be the case that different models have complimentary strengths. In such a case, alternate 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.* [3] and Smith [7], and a similar idea was previously suggested by Geelen [5]. By integrating different models, the domain pruning carried out in each model may significantly be improved, 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 *channeling constraints* that link the variables of the participating models. Also, there are some benefits of combining different solution techniques such as CP and ILP [1, 6]. The CP model may be essential in reducing the search space while the ILP model with its relaxation may be essential for bounding and guiding the search.

In this paper, we study a balanced academic curriculum problem (BACP) proposed in [2] 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. We show that this problem 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 complimentary strengths of each model. Experimental results show that the integration significantly increases the domain pruning, and even decreases the run-time on many instances.

The rest of this paper is organised as follows. In Section 2, we explain the problem. In Section 3, we study an Integer Linear Programming (ILP) model of the problem, and then in Section 4 we consider two Constraint Programming (CP) models. In Section 5, we show why and how some models are integrated. Then, in Section 6, we present the performance of the models on three real-life instances of the problem. An alternative model is examined in Section 7. Finally, in Section 8, we summarise and conclude our work.

2 Problem Description

The BACP proposed in [2] 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.

$courses = \{1, \ldots, m\} : set(int)$
$periods = \{1, \ldots, n\} : set(int)$
$a : int$ % minimum academic load allowed per period
$b : int$ % maximum academic load allowed per period
$c : int$ % minimum amount of courses allowed per period
$d : int$ % maximum amount of courses allowed per period
$credit : courses \longrightarrow int$
$prereq : set(courses \times courses)$

Figure 1: Inputs to the BACP.

- *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.

3 An ILP Model

The inputs to the BACP (shown in Figure 1) 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 [2] and shown in Figure 2 is as follows. The assignment of periods to courses is represented by a 2d 0/1 matrix of decision variables ($CURRICULUM$). The meaning of $CURRICULUM[i, j] = 1$ 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 minimizes C. The first constraint enforces

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

Figure 2: An ILP model of the BACP.

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.

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 2 as ILP when an ILP solver is used to solve the model.

4 Two CP Models

Outputs:	C *in* $0..maxint$ $LOAD[periods]$ *in* $0..maxint$ $CURRICULUM[courses]$ *in periods*
Minimize:	C
Constraints:	% inefficient way of computing the academic load of each period $\forall j \in periods\,.\; LOAD[j] = \sum_{i \in courses: CURRICULUM[i]=j} credit(i)$ % C is the maximum academic load $C = max_{j \in periods} LOAD[j]$ % set of inequalities to enforce the prerequisite constraints $\forall \langle i, j \rangle \in prereq\,.\; CURRICULUM[i] < CURRICULUM[j]$ % set of inequalities restricting the academic load of each period $\forall j \in periods\,.\; a \leq LOAD[j] \leq b$ % global constraints to restrict % the number of courses for each period $\forall j \in periods\,.\; atleast(j, CURRICULUM, c)$ $\forall j \in periods\,.\; atmost(j, CURRICULUM, d)$
Advantages:	ease of statement of the prerequisite constraints use of global constraints (better propagation)
Disadvantages:	inefficient statement of the academic load constraint

Figure 3: The CP_2 model of the BACP.

Many scheduling, assignment, routing and other problems can be efficiently and effectively solved by constraint programs on matrices of decision variables (so called "matrix models" [4]). The ILP model in Figure 2 can be used directly as such a constraint model. We refer to the model in Figure 2 as CP_1 when a CP solver is used to solve the model. The model CP_1 has already been proposed in [2].

The assignment of periods to courses can also be modelled using a 1d matrix indexed by courses and ranging over periods. The model shown in Figure 3 uses this data representation. We refer to this model as CP_2. The restriction on the number of periods to be assigned a course is captured by the 1d matrix. Unlike CP_1, computing the academic load requires the introduction of extra Boolean variables and reification constraints; such constraints are typically delayed, resulting in an inefficient model. However, in this model, the prerequisite constraints are easily stated by enforcing an ordering on the courses that have a prerequisite relationship. Furthermore, the global constraints *atleast* and *atmost* can be used to enforce the restrictions on the amount of courses allowed per period. Finally, the rest of the constraints and the objective function are the same in models CP_1 and CP_2.

Model	Main characteristics	Solver
ILP	2d 0/1 matrix	ILP
CP_1	2d 0/1 matrix	CP
CP_2	1d matrix	CP
$ILP + CP_2$	2d 0/1 matrix 1d matrix only prerequisite constraints using 1d matrix all other constraints using 2d 0/1 matrix channelling constraints	ILP+CP
$CP_1 + CP_2$	2d 0/1 matrix 1d matrix only computation of academic load using 2d 0/1 matrix all other constraints using 1d matrix channeling constraints	CP

Figure 4: Summary of the models.

5 Integrating the Models: $ILP + CP_2$ and $CP_1 + CP_2$

The models ILP and CP_1 that are based on a 2d 0/1 matrix of decision variables, and the model CP_2 that is based on a 1d 0/1 matrix of decision variables have complementary strengths. The models ILP and CP_1 differ only in the solution methods used. Each method has its own ability to reason with the constraints. Moreover, the academic load constraint for each period is easily stated in the models ILP and CP_1 by a weighted column sum on the matrix. In CP_2, however, this constraint is poorly stated by reifications that do not propagate well. The situation is the other way around if we consider the prerequisite constraints. In CP_2, this constraint is easily stated by ordering the courses that have prerequisite relationship, while in the models ILP and CP_1 we need to impose partial row sum constraints. Furthermore, in CP_2, the restriction on the number of courses for each period can be achieved by global constraints that maintain generalised arc-consistency. In order to benefit from the effectiveness of each model, we propose integrating the models ILP and CP_2 ($ILP + CP_2$), and the models CP_1 and CP_2 ($CP_1 + CP_2$) by channelling the variables of the participating models. The disadvantages of these integrations are the increased number of variables, and additional channeling constraints to be processed. A summary of the proposed models is shown in Figure 4.

In $ILP + CP_2$, we specify the prerequisite constraints on the 1d matrix, and all the other constraints on the 2d matrix. This allows ILP to benefit from CP_2's power in the statement of the prerequisite constraints, and CP_2 benefit from ILP'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. In $CP_1 + CP_2$, we pose the academic load constraints on the 2d matrix, and all the other constraints on the 1d matrix. This allows CP_1 to benefit from CP_2's global constraints and prerequisite constraints, and CP_2 benefit from CP_1's effectiveness of the academic load constraints. The constraints of the integrated models are shown in Figure 5.

Outputs:	$C \ in \ 0..maxint$ $LOAD[periods] \ in \ 0..maxint$ $CURRICULUM1[courses] \ in \ periods$ $CURRICULUM2[courses, periods] \ in \ 0..1$
Minimize:	C
Constraints:	% using the 2d matrix to compute the academic load %of each period $\forall j \in periods\,.$ $LOAD[j] = \sum_{i \in courses} CURRICULUM2[i,j] * credit(i)$ % C is the maximum academic load $C = max_{j \in periods} LOAD[j]$ % using the 1d matrix to state the prerequisite constraints $\forall \langle i,j \rangle \in prereq\,.\ CURRICULUM1[i] < CURRICULUM1[j]$ % set of inequalities restricting the academic load of each period $\forall j \in periods\,.\ a \leq LOAD[j] \leq b$ % using the 2d matrix for restricting the number of % courses of each period in $ILP + CP_2$ $\forall j \in periods\,.\ c \leq \sum_{i \in courses} CURRICULUM[i,j] \leq d$ % using the 1d matrix to state global constraints %restricting the number of courses of each period in $CP_1 + CP_2$ $\forall j \in periods\,.\ atleast(j, CURRICULUM1, c)$ $\forall j \in periods\,.\ atmost(j, CURRICULUM1, d)$ % channelling constraints % between the 1d matrix and 2d 0/1 matrix $\forall i \in courses, j \in periods\,.$ $CURRICULUM1[i] = j \leftrightarrow CURRICULUM2[i,j] = 1$
Advantages:	ease of statement of all problem constraints use of global constraints (better propagation)
Disadvantages:	redundant variables extra channelling constraints

Figure 5: The $ILP/CP_1 + CP_2$ integrated models.

6 Experimental Results

In order to evaluate the performances of the proposed models, we carried out some experiments by implementing the models in OPL [8]. Figure 6 and Figure 7 show the results on the three real-life instances used in [2] 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 CP_1 as in [2], which groups the variables by periods and assigns the value 1 first. Note that this branching heuristic achieves the best results in [2]. As for CP_2, we use the *fail first* branching strategy, i.e., branching on the variable with the smallest domain, and choosing values

Model	Results	8 periods	10 periods	12 periods
ILP	runtime	3.45	4.23	131.30
	failures	N/A	N/A	N/A
CP_1	runtime	58.52	-	-
	failures	499336	-	-
CP_2	runtime	45.10	-	-
	failures	56766	-	-
$ILP + CP_2$	runtime	0.81	8.44	3.05
	failures	183	4445	525
$CP_1 + CP_2$	runtime	0.29	0.59	1.09
	failures	651	1736	1539

Figure 6: Finding an optimal solution.

Model	Results	8 periods	10 periods	12 periods
ILP	runtime	3.45	4.23	131.30
	failures	N/A	N/A	N/A
CP_1	runtime	-	-	-
	failures	-	-	-
CP_2	runtime	-	-	-
	failures	-	-	-
$ILP + CP_2$	runtime	0.81	8.44	3.05
	failures	183	4445	525
$CP_1 + CP_2$	runtime	-	-	-
	failures	-	-	-

Figure 7: Proving optimality.

in lexicographical order.

The quickest model that found the optimal solution is $CP_1 + CP_2$, which shows that integrating the models CP_1 and CP_2 resulted in a better model despite the increased number of variables and additional channelling constraints. This is due to the increase in the amount of pruning, which led to a reduction in the search space that compensated the increase of variables and constraints. The second best model is $ILP + CP_2$, which is better than the model CP_2 on all instances, and better than the model ILP on two instances. This shows that a hybrid solution method achieved better results. In this integration, 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. However, the model $CP_1 + CP_2$ is still better than $ILP + CP_2$ in finding the optimal solution because it performs more inference and thus eliminating more search space.

Proving optimality was tough for all CP models (CP_1, CP_2, and $CP_1 + CP_2$). We observe that $ILP + CP_2$ proves optimality quicker than ILP on two instances, and is the quickest on these instances. This is because it benefits from the CP model in reducing the search space, and from the relaxation of the ILP model, which bounds and guides the search.

The CP solver used in the experiments is the one used by OPL, while the ILP solver is the CPLEX one (used by OPL again). In [2], the authors used OZ to solve the model CP_1, and *lp-solve*[1] to solve the model ILP. 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 CP_1, 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 5. Note that we used the same labelling strategy for the model CP_1 as in [2].

7 Future work

In our future work, we intend to look at models for the BACP problem based on set variables. So far, we have been viewing the BACP as finding a mapping from the set of courses into the set of periods. However, one can also view the BACP as a set partitioning problem, where the set of courses ought to be partitioned into n subsets, one for each period. With this view on the BACP, the model shown in Figure 8 declares n set variables that are subsets of the set of courses. To ensure that every course is given in exactly 1 period, the n set variables must all be pairwise disjoint and their union must equal the set of courses. The availability of global constraints on set variables, such as weighted sum and the cardinality constraint, in most constraint programming languages supporting set variables (CONJUNTO, OZ, ILOG SOLVER), makes it possible to compute the load of each period, as well as stating the restrictions of the amount of courses per period. However, with set variables, an inefficient formulation of the prerequisite constraints is unavoidable. Therefore, we will also consider channelling the model based on set variables into the model CP_2 in which the prerequisite constraints are easier to specify and reason about.

8 Conclusion

In this paper different models of the BACP are proposed and evaluated experimentally on three real-life instances. The models ILP and CP_1 have already been proposed in [2]. We propose another CP model CP_2 as well as carefully integrated models $ILP+CP_2$ and CP_1+CP_2. The integrated model $ILP+CP_2$ combines two different representations of the BACP in such a way that the cons of the model ILP is overcome with the pros of the model CP_2, and vice versa. Additionally, the integrated model employs a hybrid solution method, which combines an ILP solver with a CP solver, and thus the information is propagated from one model to the other through the usage of channelling constraints. The integrated model CP_1+CP_2 also exploits the complimentary strengths of the participating models. For this integrated model, a CP solver is used and domain pruning on the set of variables in CP_1 is propagated to the variables in the CP_2 model with the help of the channeling constraints (and vice versa), allowing more pruning to take place. The integrated model CP_1+CP_2 is the quickest model to find an optimal solution, while the model $ILP+CP_2$ is the quickest (for two instances) to prove optimality. Hence,

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

Outputs:	$C\ in\ 0..maxint$ $LOAD[periods]\ in\ 0..maxint$ $P_1 : set(courses)$ ... $P_n : set(courses)$
Minimize:	C
Constraints:	% enforcing that every course is assigned only one period $\forall i \neq j \in 1..n \cdot P_i \cap P_j = \emptyset$ $P_1 \cup \ldots \cup P_n = courses$ % weighted sum over set variable %that computes the academic load of each period $\forall j \in 1..n\,.\ \sum_{i \in P_j} credit(i) = LOAD[j]$ % C is the maximum academic load $C = max_{j \in periods} LOAD[j]$ % inefficient expression of the prerequisite constraints $\forall \langle i,j \rangle \in prereq\,.\, \forall k \in 1..n-1\,.\ \ i \in P_k \rightarrow (j \notin P_1 \wedge \ldots \wedge j \notin P_k)$ % set of inequalities restricting the academic load of each period $\forall j \in periods\,.\, a \leq LOAD[j] \leq b$ % cardinality of set variables used % to restrict the amount of courses of each period $\forall j \in 1..n\,.\, c \leq \lvert P_j \rvert \leq d$
Advantages:	use of global constraints on set variables
Disadvantages:	inefficient statement of the prerequisite constraints

Figure 8: A model of the BACP based on set variables

the integration results in better models despite the increase of 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 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.

Acknowledgments

We acknowledge Pierre Flener for his feedback on an earlier draft of this paper. We would like also to thank C. Castro and S. Manzano for providing us with the real-life instances they used in their experiments.

References

[1] A. Bockmayr and T. Kasper. Combining and linking LP and CP: Branch-and-Infer: A unifying framework for integer and finite domain constraint programming. *INFORMS J. Computing*, 10:3, 287 – 300, 1998.

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

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

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

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

[6] R. Rodosek, M.G. Wallace, and M.T. Hajian. A new approach to integrating mixed integer programming with constraint logic programming. *Annals of Operational Research*, Vol 86, 63–87, 1999.

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

[8] P. Van Hentenryck. *The* OPL *Optimization Programming Language*. The MIT Press, 1999.