

# Improving the Global Constraint SoftPrec

David Lesaint<sup>1</sup> and Deepak Mehta and Barry O’Sullivan and Luis Quesada and Nic Wilson<sup>2</sup>

**Abstract.** A soft global constraint SOFTPREC has been proposed recently for solving optimisation problems involving precedence relations. In this paper we present new pruning rules for this global constraint. We introduce a pruning rule that improves propagation from the objective variable to the decision variables, which is believed to be harder to achieve. We further introduce a pruning rule based on linear programming, and thereby make SOFTPREC a hybrid of constraint programming and linear programming. We present results demonstrating the efficiency of the pruning rules.

## 1 INTRODUCTION

Precedence constraints play a key role in many application domains e.g., in scheduling activities [Bv08], and in configuring telecom features [LMO<sup>+</sup>09]. Recently a soft global constraint, SOFTPREC, has been proposed for a feature subscription problem that arises in telecommunications [LMO<sup>+</sup>09]. A feature subscription,  $\langle F, H, P, w \rangle$ , is defined by a set of call-control features  $F$ ; a set of (hard) precedence constraints  $H$  from a given catalogue of features; a set of user-specified (soft) precedence constraints  $P$ ; and a function  $w$  that maps features and user-specified precedence constraints to weights. The value of the subscription is the sum of the weights of the features and user precedences. The task is to find a subset of features and user precedences that satisfy hard precedence constraints and maximise the value of the resulting subscription.

Given a subscription, the global constraint SOFTPREC holds if and only if there is a strict partial order on the features subject to hard and soft precedence constraints, and the value of the subscription is within the provided bounds. Achieving generalised arc consistency (GAC) on this global constraint is NP-complete and therefore it must be approximated. In [LMO<sup>+</sup>09], the pruning is mainly achieved by enforcing transitivity on the mandatory precedences when the connecting features are included, since features are optional. This enables us to learn incompatible pairs of undecided features, which helps in pruning undecided features, user precedences and bounds.

In this paper, we propose a pruning rule that infers incompatible pairs of features based on both the cost of including features and the lower bound of the value of the subscription. We compute tighter bounds from a given set of incompatible pairs of undecided features by using linear programming, making the system a hybrid of constraint programming and operations research. We present some empirical results obtained by solving instances of the minimum cycle cutset problem and the feature subscription problem. The results suggest that the proposed pruning rules can improve the efficiency of the global constraint in terms of time and search nodes.

## 2 THE GLOBAL CONSTRAINT SOFTPREC

In this section first we recall the definition of SOFTPREC from [LMO<sup>+</sup>09]. Second we present a novel way of learning incompatible pairs of features. Finally, a linear programming approach for computing tighter bounds is presented.

Let  $\langle F, H, P, w \rangle$  be a feature subscription. Let  $bf$  be a vector of Boolean variables associated with  $F$ . We say that  $i \in F$  is included if  $bf(i) = 1$ , and  $i$  is excluded if  $bf(i) = 0$ . We abuse the notation by using  $bf(i)$  to mean  $bf(i) = 1$ , and  $\neg bf(i)$  to mean  $bf(i) = 0$ . A similar convention is adopted for the other Boolean variables. Let  $bp$  be an  $|F| \times |F|$  matrix of Boolean variables. Here  $bp$  is intended to represent a strict partial order compatible with the hard constraints and restricted to  $bf$ , which implies  $bp(i, j) \Rightarrow bf(i) \wedge bf(j)$ .

**Definition 1** (SOFTPREC). *Let  $S = \langle F, H, P, w \rangle$  be a feature subscription,  $bf$  be a vector of Boolean variables,  $bp$  be a matrix of Boolean variables and  $v$  be an integer variable, SOFTPREC( $S, bf, bp, v$ ) holds if and only if*

1.  $bp$  is a strict partial order restricted to  $bf$ ,
2.  $\forall (i \prec j) \in H : bf(i) \wedge bf(j) \Rightarrow bp(i, j)$ ,
3.  $v = \sum_{i \in F} bf(i) \times w(i) + \sum_{(i \prec j) \in P} bp(i, j) \times w(i \prec j)$ .

Assume that  $S = \langle \{1, 2\}, \{1 \prec 2, 2 \prec 1\}, \emptyset, w \rangle$ , where  $w$  maps each feature to 1. Assume also that  $v = 2$  and  $bf$  and  $bp$  are totally undetermined. This instantiation of the arguments of SOFTPREC is inconsistent since the subscription should have two features. Achieving GAC on SOFTPREC is NP-complete [LMO<sup>+</sup>09].

### 2.1 Incompatible Pairs of Features

As  $bp$  is a strict partial order restricted to  $bf$ , transitivity on  $bp$  can be enforced only on included features. However, if  $(i \prec j) \in H$  and  $(j \prec k) \in H$  then  $i \prec k$  can be inferred as soon as  $j$  is included regardless of the inclusion of  $i$  and  $k$ . In order to do this kind of inference another matrix  $\psi$  of auxiliary Boolean variables is introduced. Roughly speaking, if  $\psi(i, j) = 1$  then it means if features  $i$  and  $j$  are included then  $i$  must precede  $j$ . If  $\psi(i, j) \wedge \psi(j, i)$  is true then it implies that features  $i$  and  $j$  are incompatible and they cannot be included together. Let  $I(i, j)$  be the set of undecided features (i.e., features that are neither included nor excluded) that are incompatible with either  $i$  or  $j$ . Let  $c_f^+(i, j)$  be the cost of including both features  $i$  and  $j$ , which is defined to be the sum of all the weights of features  $k$  of  $I(i, j)$ , and the sum of all the weights of soft precedences that involve any  $k$  of  $I(i, j)$ .

Let  $m_v$  be the *maximum value* of the subscription, which is defined to be the sum of the weights of all the features and soft precedences. Let  $v^-$  be a lower bound on  $v$ , which is always greater than or equal to the sum of the included features and soft precedences. Therefore,  $m_v - v^-$  is an upper bound on the cost, which we call maximum

<sup>1</sup> British Telecommunications plc, UK, david.lesaint@bt.com

<sup>2</sup> Cork Constraint Computation Centre, University College Cork, Ireland, {d.mehta|b.osullivan|l.quesada|n.wilson}@4c.ucc.ie

**Table 1.** Average results over 10 feature subscription problem instances.

|                                       | $\langle 50, 250, \{<, >\} \rangle$ |               |          |               | $\langle 50, 500, \{<, >, <>\} \rangle$ |       |             |           | $\langle 50, 750, \{<, >\} \rangle$ |       |              |            |
|---------------------------------------|-------------------------------------|---------------|----------|---------------|-----------------------------------------|-------|-------------|-----------|-------------------------------------|-------|--------------|------------|
| $\langle F, P, w \rangle$             | SP                                  | SP+LP         | SP+IB    | SP+LP+IB      | SP                                      | SP+LP | SP+IB       | SP+LP+IB  | SP                                  | SP+LP | SP+IB        | SP+LP+IB   |
| Results in terms of search nodes      |                                     |               |          |               |                                         |       |             |           |                                     |       |              |            |
| $\langle 40, 40, 4 \rangle$           | 2,708                               | <b>1,134</b>  | 1,993    | 1,165         | 188                                     | 87    | <b>57</b>   | 91        | 993                                 | 377   | 359          | <b>356</b> |
| $\langle 45, 90, 4 \rangle$           | 103,064                             | <b>38,227</b> | 65,719   | 39,596        | 620                                     | 315   | <b>191</b>  | 381       | 2,902                               | 1,224 | <b>1,035</b> | 1,294      |
| $\langle 50, 4, 4 \rangle$            | 9,133                               | 1,937         | 5,001    | <b>1,910</b>  | 954                                     | 124   | 176         | <b>94</b> | 5,569                               | 1,022 | 1,251        | <b>730</b> |
| Results in terms of time (in seconds) |                                     |               |          |               |                                         |       |             |           |                                     |       |              |            |
| $\langle 40, 40, 4 \rangle$           | <b>10.03</b>                        | 11.76         | 15.47    | 11.48         | 0.81                                    | 1.42  | <b>0.51</b> | 1.63      | 4.74                                | 6.78  | <b>3.49</b>  | 7.5        |
| $\langle 45, 90, 5 \rangle$           | 792.12                              | 600.66        | 1,133.23 | <b>517.41</b> | 4.78                                    | 7.10  | <b>3.43</b> | 7.51      | 23.49                               | 30.20 | <b>19.24</b> | 31.23      |
| $\langle 50, 4, 4 \rangle$            | 36.82                               | <b>25.82</b>  | 30.24    | 28.00         | 2.80                                    | 2.03  | <b>1.42</b> | 2.69      | 23.03                               | 26.30 | <b>13.91</b> | 32.06      |

allowed cost. Let  $b_c$  be the *backward cost*, which is defined to be the sum of the weights of the excluded features and soft precedences. As  $b_c$  is the cost that is already incurred, if  $(m_v - v^-) - b_c$  is less than the cost of including both features  $i$  and  $j$ , then they are incompatible, and based on which the following pruning rule is introduced:

$$\frac{(m_v - v^- - b_c) < c_f^+(i, j)}{\psi(i, j) \wedge \psi(j, i)}. \quad (1)$$

## 2.2 LP based Pruning

[LMO<sup>+</sup>09] presented several ways of computing *forward cost*, which is a lower bound on the cost that is incurred from a given set of incompatible pairs of undecided features. Here, we present another way of computing forward cost using linear programming (LP).

Let  $M = (V_M, E_M)$  be an undirected graph associated with a set of incompatible pairs of undecided features, where: a vertex represents a feature that is incompatible with at least one other feature; and an edge between two vertices encodes incompatibility between the features. For each  $i \in V_M$  a weight  $\delta(i)$  is defined as follows:

$$\delta(i) = \left( w(i) + \sum_{p \in \{i \sim j, j \sim i\}} \left( \sum_{j \notin V_M} w(p) + \frac{1}{2} \sum_{j \in V_M} w(p) \right) \right).$$

$\delta(i)$  is a lower bound on the cost of excluding  $i$ . It is the sum of the weight of the feature  $i$ , the sum of the undecided soft precedences involving  $j$  such that  $j$  is not incompatible with any other feature, and half of the sum of soft precedences involving  $j$  such that  $j$  is incompatible with at least one other feature. The latter is divided by 2 to avoid over-estimating the cost of a soft precedence when the two features involved in the soft precedence are excluded.

We remark that the value of an optimal weighted vertex cover of  $M$  is a lower bound on the cost based on the incompatible pairs of features. As computing an optimal weighted vertex cover is NP-hard, we compute a lower bound by associating a real variable  $r(i)$  with each  $i \in V_M$  and by using the following linear program:

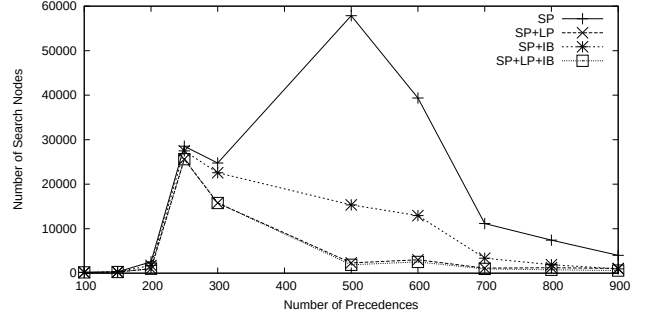
$$\begin{aligned} & \text{minimize} && \sum_{i \in V_M} r(i) \delta(i) \\ & \text{subject to} && r(i) + r(j) \geq 1 \quad \forall (i, j) \in E_M \\ & && r(i) \leq 1 \quad \forall i \in V_M \\ & && r(i) \geq 0 \quad \forall i \in V_M \end{aligned} \quad (2)$$

Here  $r(i) = 1$  means that feature  $i$  is excluded. Let  $fc_{lp}(M)$  be the forward cost obtained by solving (2). This LP-based forward cost can be used to compute tighter upper bounds on  $v$  when including/excluding a feature, which may help in achieving a higher level of pruning when rules (1) and (2) of [LMO<sup>+</sup>09] are applied.

## 3 EXPERIMENTAL RESULTS

We tested with instances of the feature subscription problem<sup>3</sup> and the minimum cycle cutset problem as used in [Bv08]. The results for the

SOFTPREC version as used in [LMO<sup>+</sup>09] is denoted by SP, which is shown to perform better than the approach suggested in [Bv08]. When incompatibilities are learned using bounds using Rule (1), it is denoted by IB. When LP based pruning is added to SOFTPREC it is denoted by LP. Note that SOFTPREC is implemented in the CHOCO constraint solver and its LP part is solved using CPLEX. Some of these results are presented in Table 1 and Figure 1. These results clearly demonstrate that the two enhanced versions of SOFTPREC outperform the original SOFTPREC when it comes to pruning. The overhead associated with IB is such that it does not always reduce the time. Despite a significant overhead associated with the creation of an LP problem at each node of the search tree, LP reduces the time when solving hard instances, e.g.,  $\langle 45, 90, 4 \rangle$  in Table 1.



**Figure 1.** Minimum cycle cutset problem with 50 variables.

## 4 CONCLUSION

We have presented two different ways of increasing pruning for the SOFTPREC global constraint and demonstrated their effectiveness on a variety of instances of the feature subscription problem and the minimum cycle cutset problem. In future, we would like to integrate the LP-based pruning more seamlessly with SOFTPREC to reduce the overhead caused by the creation of CPLEX objects during search.

**Acknowledgements.** This work has been supported by the Science Foundation Ireland under Grant No. 05/IN/I886 and 08/PI/I1912, and Embark Post Doctoral Fellowships No. CT1080049908 and No. CT1080049909.

## REFERENCES

- [Bv08] Roman Barták and Ondřej Čepek. Incremental filtering algorithms for precedence and dependency constraints. *International Journal on Artificial Intelligence Tools*, 2008.
- [LMO<sup>+</sup>09] David Lesaint, Deepak Mehta, Barry O’Sullivan, Luis Quesada, and Nic Wilson. A Soft Global Precedence Constraint. In *IJCAI-09*, Pasadena, CA, USA, 2009.

<sup>3</sup> <http://4c.ucc.ie/~lquesada/FeatureSubscription/page/instances.htm>