A Soft Global Precedence Constraint

David Lesaint
BT, UK
david.lesaint@bt.com

Deepak Mehta
4C, UCC, Ireland
d.mehta@4c.ucc.ie

Barry O'Sullivan
4C, UCC, Ireland
b.osullivan@4c.ucc.ie

Luis Quesada
4C, UCC, Ireland
l.quesada@4c.ucc.ie

Nic Wilson
4C, UCC, Ireland
n.wilson@4c.ucc.ie

Abstract

Hard and soft precedence constraints play a key role in many application domains. In telecommunications, one application is the configuration of call-control feature subscriptions where the task is to sequence a set of user-selected features subject to a set of hard (catalogue) precedence constraints and a set of soft (user-selected) precedence constraints. When no such consistent sequence exists, the task is to find an optimal relaxation by discarding some features or user precedences. For this purpose, we present the global constraint SOFTPREC. Enforcing Generalized Arc Consistency (GAC) on SOFTPREC is NP-complete. Therefore, we approximate GAC based on domain pruning rules that follow from the semantics of SOFTPREC; this pruning is polynomial. Empirical results demonstrate that the search effort required by SOFTPREC is up to one order of magnitude less than the previously known best CP approach for the feature subscription problem. SOFTPREC is also applicable to other problem domains including minimum cutset problems for which initial experiments confirm the interest.

1 Introduction

Precedence constraints play a key role in planning, scheduling and sequencing problems. In internet telephony for instance, service delivery architectures support the sequential activation of call control features. Users can then personalise their call logic by selecting and sequencing features from a catalogue, e.g., announcement followed by call-divert on incoming calls. However, hard constraints apply to avoid undesirable feature interactions, e.g., call-divert and call-waiting are mutually exclusive. This feature subscription configuration problem, which generalises the minimum cutset problem, can be formulated as a constraint optimisation problem [Lesaint *et al.*, 2008b]. Informally, a *feature subscription* is defined by a set of features, a set of user specified precedence constraints, a function that maps features and user specified precedence constraints to weights, and a set of precedence and exclusion constraints from the catalogue. The task is to find a sequence of features that is consistent with all the constraints. If such a sequence does not exist then the task is to find an optimal relaxation of the feature subscription that is closest to the initial requirements of the user. This problem is NP-hard [Lesaint *et al.*, 2008b].

We propose the soft global precedence constraint SOFTPREC that holds if and only if there is a strict partial order on the selected features subject to hard (catalogue) precedence constraints and soft (user-selected) precedence constraints, and the value of the subscription is within the provided bounds. Note that a sequence can be derived from a strict partial order in polynomial time. Enforcing generalised arc consistency (GAC) [Rossi *et al.*, 2006](Chap. 3) on SOFTPREC is NP-complete. Therefore, we approximate it by pruning the domains of the variables based on the rules that follow from the definition of SOFTPREC; this pruning is polynomial. The pruning rules are presented declaratively which allows us to separate concerns. We also present five upper bounds for pruning the bounds of the value of the subscription. These bounds are computed based on the incompatibilities that are inferred between pairs of features, and the dependencies between user precedences and their corresponding features. The tightness of these bounds depends on the degree of inference made on these incompatibilities and dependencies.

We use these bounds along with the other pruning rules of SOFTPREC within branch and bound search to find an optimal relaxation of a feature subscription. We also compare our approach with the alternative constraint programming (CP) approaches presented in [Lesaint *et al.*, 2008b]. Empirical results demonstrate that the filtering achieved by using SOFTPREC reduces the search effort to such an extent that it is up to one order of magnitude faster than the previously known best CP approach for feature subscription problems. We also experiment with minimum cutset problems and compare our approach with the one presented in [Barták and Čepek, 2008]. When compared with the latter approach, SOFTPREC reduces the search effort up to two orders of magnitude.

2 Feature Subscription

In this section we provide some background relevant with the feature subscription problem considered in this paper.

Let f_i and f_j be features, we write a precedence constraint of f_i before f_j as $f_i \prec f_j$ or as $\langle f_i, f_j \rangle$. We write $f_i \prec\succ f_j$ to mean $f_i \prec f_j$ and $f_j \prec f_i$, which corresponds to f_i and f_j being incompatible (mutually exclusive). A **catalogue** is a pair $\langle \mathcal{F}, \mathcal{H} \rangle$, where $\mathcal{F}$ is a set of features and $\mathcal{H}$ is a set of precedence constraints on $\mathcal{F}$. A **feature subscription** $\mathcal{S}$ of a catalogue $\langle \mathcal{F}, \mathcal{H} \rangle$ is a tuple $\langle F, H, P, w \rangle$, where $F \subseteq \mathcal{F}$, H is the projection of $\mathcal{H}$ on F, i.e., $\mathcal{H}\downarrow_F = \{f_i \prec f_j \in \mathcal{H} : \{f_i, f_j\} \subseteq F\}$, P is a set of (user defined) precedence constraints on F, w is a function that maps features and user precedence constraints to weights. The value of $\mathcal{S}$ is defined by $\text{Value}(\mathcal{S}) = \sum_{f \in F} w(f) + \sum_{p \in P} w(p)$.

A subscription $\mathcal{S} = \langle F, H, P, w \rangle$ is defined to be **consistent** if and only if the directed graph $\langle F, H \cup P \rangle$ is acyclic. The time complexity for checking the consistency of a subscription is $\mathcal{O}(|F| + |H \cup P|)$ [Lesaint *et al.*, 2008b]. If a subscription is consistent then there exists at least one sequence of features that respects all the constraints. A **relaxation** of $\langle F, H, P, w \rangle$ is a subscription $\langle F', H', P', w' \rangle$ such that $F' \subseteq F$, $H' = \mathcal{H}\downarrow_{F'}$, $P' \subseteq P\downarrow_{F'}$, w' is w restricted to F' and P'. Let $R_{\mathcal{S}}$ be the set of all consistent relaxations of a feature subscription $\mathcal{S}$. We say that $\mathcal{S}_i \in R_{\mathcal{S}}$ is an **optimal relaxation** of $\mathcal{S}$ if it has maximum value among all relaxations, i.e., if and only if there does not exist $\mathcal{S}_j \in R_{\mathcal{S}}$ such that $\text{Value}(\mathcal{S}_j) > \text{Value}(\mathcal{S}_i)$. Finding an optimal relaxation of a subscription is NP-hard [Lesaint *et al.*, 2008b].

A simple constraint optimisation problem (COP) formulation for finding an optimal relaxation of $\langle F, H, P, w \rangle$ is as follows. Each feature $f_i \in F$ is associated with a *Boolean variable* $bf(i)$ and an *integer variable* $pf(i)$. The domain of each integer variable $pf(i)$ is $\{1, \ldots, |F|\}$. Each user precedence constraint $(f_i \prec f_j) \in P$ is associated with a *Boolean variable* $bp(i,j)$. A variable v is associated with the objective function. A catalogue precedence constraint, $(f_i \prec f_j) \in \mathcal{H}$, can be expressed as $bf(i) \wedge bf(j) \Rightarrow (pf(i) < pf(j))$. A user precedence constraint $(f_i \prec f_j) \in P$ can be expressed as $bp(i,j) \Rightarrow (bf(i) \wedge bf(j) \wedge (pf(i) < pf(j)))$. The objective is to maximise the value of v, where $v = \sum_{f_i \in F} bf(i) \times w(f_i) + \sum_{\langle i,j \rangle \in P} bp(i,j) \times w(\langle i,j \rangle)$. Note that this model encodes the precedence relation using absolute position variables. [Lesaint *et al.*, 2008b] present results of using the following consistency techniques: arc consistency (AC), singleton arc consistency (SAC), and restricted singleton arc consistency (RSAC) on the Boolean variables of the model. All these consistency techniques enforce bounds consistency on the objective variable v.

3 The SoftPrec Global Constraint

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 f_i is included if $bf(i) = 1$, and f_i is excluded if $bf(i) = 0$. We abuse 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 a matrix of Boolean variables. Here *bp* is intended to represent a strict partial order on the included features F' which is compatible with the catalogue constraints restricted to F'.

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, the global constraint* SoftPrec(S, bf, bp, v) *holds if and only if*

1. *bp is a strict partial order restricted to bf, i.e.,*

$$\begin{array}{ll} \forall i,j \in F : bp(i,j) \Rightarrow bf(i) \wedge bf(j) & \text{(restricted)}, \\ \forall i,j \in F : bp(i,j) \Rightarrow \neg bp(j,i) & \text{(asymmetric)}, \\ \forall i,j,k \in F : bp(i,j) \wedge bp(j,k) \Rightarrow bp(i,k) & \text{(transitive)}, \end{array}$$

2. *bp is compatible with H restricted to bf, i.e.,*

$$\forall (f_i \prec f_j) \in H : bf(i) \wedge bf(j) \Rightarrow bp(i,j),$$

3. $v = \sum_{i \in F} bf(i) \times w(i) + \sum_{\langle i,j \rangle \in P} bp(i,j) \times w(\langle i,j \rangle)$.

A solution of SoftPrec is a consistent relaxation of the subscription $\langle F, H, P, w \rangle$. Notice that the minimum cutset problem [Garey and Johnson, 1979] can be expressed in terms of SoftPrec by associating vertices with features and arcs with catalogue precedence constraints. Therefore, achieving generalised arc consistency on SoftPrec is NP-hard and hence NP-complete.

If $(f_i \prec f_j) \in H$ and $(f_j \prec f_k) \in H$ then $f_i \prec f_k$ can be inferred only if f_j is included, as features are optional. In order to do this kind of inference we introduce another matrix ψ of auxiliary Boolean variables.

Proposition 1. *Let $\psi(i,j) \equiv \neg bf(i) \vee \neg bf(j) \vee bp(i,j)$.* SoftPrec$(S, bf, bp, v)$ *holds if and only if the following holds:*

1. $bp(i,j) \Rightarrow bf(i) \wedge bf(j)$;
2. $bf(i) \wedge bf(j) \wedge \psi(i,j) \Rightarrow \neg\psi(j,i)$;
3. $bf(j) \wedge \psi(i,j) \wedge \psi(j,k) \Rightarrow \psi(i,k)$;
4. $(f_i \prec f_j) \in H \Rightarrow \psi(i,j)$;
5. $v = \sum_{i \in F} bf(i) \times w(i) + \sum_{\langle i,j \rangle \in P} bp(i,j) \times w(\langle i,j \rangle)$.

4 Pruning Rules

An algorithm for pruning the domains is presented in terms of rules that follow from the semantics of SoftPrec. Each pruning rule has the following form: $\frac{\textit{Precondition}}{\textit{Postcondition}}$. Here *Precondition* defines what should be true for the rule to be triggered and *Postcondition* defines what should be true after executing the pruning rule. The algorithm iterates until no precondition is met.

4.1 Pruning Rules Related to Transitive and Asymmetric Properties

From Proposition 1.1 and the definition of ψ:

$$\frac{bf(i) \wedge bf(j) \wedge \psi(i,j)}{bp(i,j)} \qquad \frac{bf(i) \wedge \neg bp(i,j) \wedge \psi(i,j)}{\neg bf(j)}$$

$$\frac{bp(i,j)}{bf(i) \wedge bf(j) \wedge \psi(i,j)} \qquad \frac{\neg bf(i) \vee \neg bf(j) \vee \neg\psi(i,j)}{\neg bp(i,j)}$$

From Proposition 1.2:

$$\frac{bf(i) \wedge bf(j) \wedge \psi(i,j)}{\neg\psi(j,i)} \qquad \frac{bf(i) \wedge \psi(i,j) \wedge \psi(j,i)}{\neg bf(j)}$$

From Proposition 1.3:

$$\frac{bf(j) \wedge \psi(i,j) \wedge \psi(j,k)}{\psi(i,k)}$$

From Proposition 1.1, 1.2, and the definition of ψ:

$$\frac{\psi(j,i)}{\neg bp(i,j)}$$

From Proposition 1.4, consistency with the catalogue precedence constraints is ensured by pruning the domain of $\psi(i,j)$ based on the implication $(f_i \prec f_j) \in H \Rightarrow \psi(i,j)$.

4.2 Pruning Rules Related to Bounds

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 user precedences. Let b_v be the *backward value* of the subscription, which is defined to be the sum of the weights of included features and user precedences. Let b_c be the *backward cost*, which is defined to be the sum of the weights of the excluded features and user precedences.

We say that f_i and f_j are mutually exclusive when we cannot include both of them, i.e., $\psi(i,j) \wedge \psi(j,i)$. Let M be the set of mutually exclusive pairs of undecided (unassigned) features. Let $c_f^+(i)$ be the *connected cost of including* f_i, which is defined to be the sum of the weights of the undecided f_j and user precedences involving f_j such that $\{f_i, f_j\} \in M$. Let $c_f^-(i)$ be the *connected cost of excluding* f_i, which is defined to be the sum of the weight of f_i and the weights of the undecided user precedences that involve f_i.

Let $lb_f^+(i)$ and $lb_f^-(i)$ be the lower bounds on v when f_i is included and excluded, respectively. Let $ub_f^+(i)$ and $ub_f^-(i)$ be the upper bounds on v when f_i is included and excluded, respectively:

$$\begin{array}{lclclcl} lb_f^+(i) & = & b_v + w(i), & \quad & ub_f^+(i) & = & m_v - (b_c + c_f^+(i)), \\ lb_f^-(i) & = & b_v, & & ub_f^-(i) & = & m_v - (b_c + c_f^-(i)). \end{array}$$

We maintain a lower bound, v^-, on v and an upper bound, v^+, on v. If a bound obtained when excluding (including) f_i is not within the bounds of v then f_i must be included (excluded).

$$\frac{(lb_f^-(i) > v^+) \vee (ub_f^-(i) < v^-)}{bf(i)} \qquad \frac{(lb_f^+(i) > v^+) \vee (ub_f^+(i) < v^-)}{\neg bf(i)} \tag{1}$$

The upper (lower) bound of v cannot be greater (less) than the maximum (minimum) of the upper (lower) bound of the value resulting from the inclusion and exclusion of f_i:

$$\frac{v^+ > \max(ub_f^+(i), ub_f^-(i))}{v^+ := \max(ub_f^+(i), ub_f^-(i))} \qquad \frac{v^- < \min(lb_f^+(i), lb_f^-(i))}{v^- := \min(lb_f^+(i), lb_f^-(i))} \tag{2}$$

Let $v_p^+(\langle i,j \rangle)$ be the *connected value of including a user precedence* $\langle i,j \rangle \in P$, which is defined to be the sum of the weights of the user precedence and the features involved in it (if the latter are undecided). Let $c_p^+(\langle i,j \rangle)$ be the *connected cost of including* $\langle i,j \rangle \in P$, which is defined to be the sum of the weights of the undecided f_k and user precedences involving f_k such that either $\{f_i, f_k\} \in M$ or $\{f_j, f_k\} \in M$.

Let $lb_p^+(\rho)$ and $lb_p^-(\rho)$ be the lower bounds on v and let $ub_p^+(\rho)$ and $ub_p^-(\rho)$ be the upper bounds on v when a user precedence ρ is included and excluded, respectively.

$$\begin{array}{lclclcl} lb_p^+(\rho) & = & b_v + v_p^+(\rho), & \quad & ub_p^+(\rho) & = & m_v - (b_c + c_p^+(\rho)), \\ lb_p^-(\rho) & = & b_v, & & ub_p^-(\rho) & = & m_v - (b_c + w(\rho)). \end{array}$$

The pruning rules associated with these bounds are obtained by replacing $lb_f^+(i)$, $lb_f^-(i)$, $ub_f^+(i)$, $ub_f^-(i)$ and $bf(i)$ with $lb_p^+(\rho)$, $lb_p^-(\rho)$, $ub_p^+(\rho)$, $ub_p^-(\rho)$ and $bp(\rho)$, respectively, in Equations (1) and (2).

If n is the sum of the number of features and user precedences then the worst-case time complexity of the pruning rules presented in this section is $\mathcal{O}(n^3)$ [Lesaint *et al.*, 2008a].

5 Tighter Upper Bounds using Forward Costs

In this section, we introduce the notion of *forward cost* and use it to compute tighter upper bounds on v when a feature (and a user precedence) is included/excluded.

5.1 Forward Costs

The set of pairs of incompatible features, M, can be associated with an undirected graph where the vertices represent features involved in the incompatibilities and edges represent incompatibilities. Let $C_1, \ldots, C_k$ be the components (maximal connected subgraphs) of the graph M. The features involved in any intersecting pairs of incompatible features are always in the same component, e.g. if $\{f_i, f_j\} \in M$ and $\{f_j, f_k\} \in M$ then f_i, f_j and f_k are in the same component. If $f_i \in F$ is not involved in any pair of incompatible features of M, then f_i is not in any component.

Let us assume that $F = \{a, b, c, d, g, h, j\}$, $P = \{p, q, r\}$. Here p, q and r are user precedences defined on features a and b, c and g, and g and j, respectively. Let us also assume that $M = \{\{a,b\}, \{b,c\}, \{c,d\}, \{g,h\}\}$. There are two components of the graph associated with M, which are $C1 = \{a,b,c,d\}$ and $C2 = \{g,h\}$. This is pictorially depicted in Figure 1. An ellipse represents a feature, and the integer number its weight. A dashed line denotes a user precedence between the corresponding features and the integer number its weight. A solid line represents incompatibility between two features based on the set M. The features involved in the components of M are encapsulated in the boxes. Notice that feature j is not involved in any pair of incompatible features of M, therefore, it is not in any component of M.

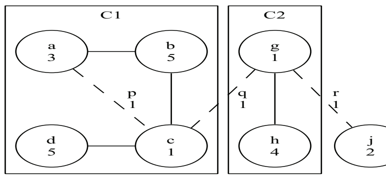

Figure 1: A graph depicting pairs of incompatible features.

Given the components of M, the forward cost is the sum of the lower-bounds of the costs incurred from the components. We present different forward costs and explain them with an example as illustrated in Figure 1. As the forward cost depends on M, we use the notation $fc_l(M)$ to denote the l^{th} forward cost.

The simplest way of computing the forward cost is by computing the minimum weight of features of each component and summing up those weights:

$$fc_1(M) = \sum_{i=1}^{k} \min\{w(f) | f \in C_i\}.$$

This is based on the fact that at least one feature will be removed from each component. For Figure 1, the forward cost based on fc_1 is $w(c) + w(g)$, which is equal to 2. Observe that $w(c)$ and $w(g)$ are the minimum weights of features of *C1* and *C2*, respectively.

Let λ_i be the lower bound on the number of features that should be excluded from C_i in order to break all the incompatibilities. The value of λ_i for each C_i is implicitly 1 for fc_1. However, a tighter forward cost can be computed by using a stronger λ_i and summing the λ_i smallest costs of excluding features of each component. There exist several ways of doing this. However, for this paper, we define λ_i to be the maximum value of k such that the sum of the degrees of the first k vertices is no more than the number of edges, where the vertices are listed in decreasing order of degree. Based on this, a forward cost can be computed as follows:

$$fc_2(M) = \sum_{i=1}^{k} \sum_{j=1}^{\lambda_i} \min{}^j \{w(f) | f \in C_i\}.$$

Here $\min^j$ is the function that gives the j^{th} smallest weight. For Figure 1, the forward cost based on fc_2 is $w(c) + w(a) + w(g)$, which is 5. Unlike fc_1, $\lambda_1 = 2$ for $C1$ and, therefore, the two smallest weights of features are selected from $C1$ which are $w(a)$ and $w(c)$.

Neither fc_1 nor fc_2 take advantage of the fact that the exclusion of a feature may also exclude a user precedence. For this purpose, we present fc_3 which can be seen as an extension of fc_1. Instead of considering only the minimum weight of the features of each component, fc_3 considers the minimum sum of the weight of a feature and the weights of user precedences that involve the feature.

$$fc_3(M) = \sum_{i=1}^{k} \min \left\{ w(f) + \frac{1}{2} \sum_{p \in inter_i(f)} w(p) + \sum_{p \notin inter_i(f)} w(p) \,\middle|\, f \in C_i \right\}.$$

Here $inter_i(f)$ is the set of user precedences that involve feature f of C_i and feature g of $C_j \neq C_i$. The forward cost based on fc_3 is 5 for Figure 1. It is the sum of the cost of excluding c from *C1*, which is $w(a) + w(p) + w(q)/2$, and the cost of excluding g from *C2*, which is $w(g) + w(q)/2 + w(r)$. Although c and g are in different components, their exclusion costs include the weight of the user precedence q, since excluding c or g would exclude q. In order to ensure that the weights of user precedences involving features of different components are considered only once, the weights are halved. This allows us to compute the cost of each component independently.

The cost of each component in fc_3 is based on the minimum cost incurred by excluding only one feature. However, fc_4 may consider multiple features along with their user precedences as shown below:

$$fc_4(M) = \sum_{i=1}^{k} \sum_{j=1}^{\lambda_i} \min{}^j \left\{ w(f) + \frac{1}{2} \sum_{p \notin rest_i(f)} w(p) + \sum_{p \in rest_i(f)} w(p) \,\middle|\, f \in C_i \right\}.$$

Here, $rest_i(f)$ is the set of user precedences that involve features f and g such that f is in C_i and g is not in any C_j. The cost of the component $C1$ based on fc_4 is the sum of the costs of excluding features a and c which are $w(a) + w(p)/2$ and $w(c) + w(p)/2 + w(q)/2$ respectively. The cost of the component $C2$ based on fc_4 is the cost of excluding feature g which is $w(g) + w(q)/2 + w(r)$. Hence, the forward cost is 8. Notice that the weights of the user precedences that involve features of the same component are not divided by 2 in fc_3. The reason is that the cost of each component is based on the cost of excluding only one feature which is not true for fc_4. Therefore, fc_4 also halves the weights of those user precedences that involve features of the same component.

If the value obtained after subtracting the backward cost and the forward cost from the maximum value of the subscription is less than the upper bound of v, then the upper bound can be updated accordingly.

5.2 Tighter Upper Bounds

This section describes the computation of tighter upper bounds on v due to the inclusion/exclusion of a feature based on the notion of forward cost. Remember that these upper bounds also use the connected costs. Therefore, it may not be possible to consider all the pairs of incompatible features of M for the forward cost computation. Let us consider an example. Let $M = \{\{f_1, f_2\}, \{f_2, f_3\}, \{f_3, f_4\}, \{f_5, f_6\}\}$. The connected cost of including f_2 is $w(f_1) + w(f_3)$. Therefore, all the incompatibilities that involve either f_1 or f_3 cannot be considered for the forward cost computation. So the forward cost of including f_2 would be based on only one pair of incompatible features, which is $\{f_5, f_6\}$. Similarly, the forward cost of excluding f_2 would be based on $\{f_3, f_4\}$ and $\{f_5, f_6\}$ pairs of incompatible features.

Let $M_f^+(i)$ be the subset of the pairs of features of M based upon which the forward cost of including f_i is computed. Formally, $M_f^+(i) = M - \{\{j, k\} \in M : \{i, j\} \in M\}$. Let $M_f^-(i)$ be the subset of pairs of features of M based upon which the forward cost of excluding f_i is computed. Formally, $M_f^-(i) = M - \{\{i, j\} \in M\}$. Based on the forward cost with respect to the sets $M_f^+(i)$ and $M_f^-(i)$, the tighter upper bounds can be defined as follows:

$$\begin{aligned} ub_f^+(i) &= m_v - (b_c + c_f^+(i) + fc(M_f^+(i))) \\ ub_f^-(i) &= m_v - (b_c + c_f^-(i) + fc(M_f^-(i))). \end{aligned} \tag{3}$$

The computation of the upper bounds, as presented in Equation (3), could be expensive as its worst-case time complexity is $\mathcal{O}(n^4)$, where n is $|F| + |P|$. A less expensive approach would be to compute $fc(M)$ and store the cost of each component of M. An under-estimate of the forward cost of including/excluding f_i $(fc(M_f^+(i))/fc(M_f^-(i)))$ can be computed by subtracting the cost of the component associated with f_i from the sum of the cost of all the components of M, i.e., $fc(M)$. By sacrificing some incompatible pairs of features of the component associated with f_i, the worst-case time complexity would then be $\mathcal{O}(n^3)$. A similar approach can be used for tightening the upper bounds on v when a user precedence is included/excluded.

6 Comparison of Pruning

We use the notation Φ_{ac} and Φ_{sac} to denote the pruning achieved by enforcing AC and SAC respectively on the Boolean variables of the model, as presented in Section 2. Both Φ_{ac} and Φ_{sac} enforce bounds consistency on the objective variable v. We use the notation Φ_{sp} to denote the pruning achieved by using the rules of SoftPrec. In this section we compare Φ_{ac}, Φ_{sp}, and Φ_{sac}. For simplicity, we ignore user precedences. Nevertheless, the results can be extended for user precedences.

Proposition 2. *Φ_{sp} and Φ_{sac} are incomparable.*

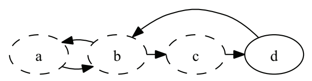

Figure 2: $\Phi_{sac} > \Phi_{sp}$

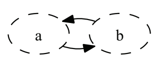

Figure 3: $\Phi_{sp} > \Phi_{sac}$

Proof. We show that Φ_{sp} and Φ_{sac} are incomparable by showing cases where each prunes more than the other one. In Figures 2 and 3, a dashed (solid) ellipse represents an undecided (included) feature, and an edge represents a hard precedence constraint. All features have weight equal to 1.

In Figure 2, if $v^- = 3$ and $v^+ = 4$, then Φ_{sac} determines the undecided variables. Φ_c includes feature a, since excluding it results in the inclusion of b and c, which causes a failure. As a is included, b is excluded, which causes the inclusion of c. Φ_{sp} is not able to include a, since the bounds on v when excluding a are within the bounds of v.

In Figure 3, if $v^- = 0$ and $v^+ = 2$ then Φ_{sp} updates v^+ to 1. This is not done by Φ_{sac} since the inclusion/exclusion of each feature does not make the subproblem arc-inconsistent. □

Proposition 3. *Φ_{sp} is tighter than Φ_{ac}.*

Proof. To prove Φ_{sp} is tighter than Φ_{ac}, we show that if Φ_{ac} includes/excludes a feature or prunes the bounds of v then Φ_{sp} also does that, and the converse does not always hold.

Only the sum constraint of Φ_{ac} can include/exclude a feature. The lower bounds on v when including/excluding f_i in Φ_{ac} are $b_v + w(i)$ and b_v respectively, which are the same for Φ_{sp}. The upper bounds of including/excluding f_i in Φ_{ac} are $m_v - b_c$ and $m_v - b_c - w(i)$ respectively, which are greater than or equal to the bounds of Φ_{sp} (see Equation (1)). Therefore, each feature included/excluded by Φ_{ac} is also included/excluded by Φ_{sp}, and each value removed from v by Φ_{ac} is also removed by Φ_{sp}.

In order to show that the inclusion/exclusion of a feature by Φ_{sp} does not imply the same by Φ_{ac}, we consider a trivial example. Assume that f_1, f_2 and f_3 are in F and have weight equal to 1, $f_1 \prec\succ f_2$, $f_1 \prec\succ f_3$ are catalogue constraints, and v^- is 2 and v^+ is 3. The upper bound on v when including f_1 in Φ_{sp} would be 1, since f_1 is mutually exclusive with f_2 and f_3. Therefore Φ_{sp} would exclude f_1. Based on the sum constraint, the upper bound on v when including f_1 in Φ_{ac} would be 3. Therefore, Φ_{ac} would not exclude f_1. Thus, Φ_{sp} is tighter than Φ_{ac}. □

7 Experimental Results

In this section, we empirically evaluate the performance of the pruning rules and bounds computation of SOFTPREC.

All the search algorithms were implemented using Choco[1] and were equipped with a static version of the *dom/deg* variable ordering heuristic, where *dom* is the domain size and *deg* is the original degree of a variable. All the experiments were performed on a PC Pentium 4 (CPU 1.8 GHz and 768MB of RAM) processor. The performances of all the approaches are measured in terms of nodes and runtime in seconds.

[1]http://choco.sourceforge.net/

Table 1: Mean results for feature subscription $\langle 45, 45, 4\rangle$ of catalogue $\langle 50, 250, \{\prec, \succ\}\rangle$.

	SP_0	SP_1	SP_2	SP_3	SP_4
time (ms)	105,240	65,539	60,689	52,735	45,720
nodes	32,880	18,111	15,272	14,052	10,143

7.1 Feature Subscription Problem

We experimented with a variety of *random catalogues* and many classes of *random feature subscriptions* [Lesaint *et al.*, 2008b]. A random catalogue is defined by a tuple $\langle n_c, m_c, T_c\rangle$. Here, n_c is the number of features, m_c is the number of catalogue precedence constraints and $T_c \subseteq \{\prec, \succ, \prec\succ\}$ is a set of types of constraints. A random feature subscription is defined by a tuple $\langle n_u, m_u, w\rangle$. Here, n_u is the number of features, m_u is the number of user precedence constraints and w is an integer greater than 0. Each feature and each user precedence constraint is associated with an integer weight which is between 1 and w inclusive.

We generated catalogues of the following forms: $\langle 50, 250, \{\prec, \succ\}\rangle$, $\langle 50, 500, \{\prec, \succ, \prec\succ\}\rangle$ and $\langle 50, 750, \{\prec, \succ\}\rangle$. For each random catalogue, we generated $\langle 5, 5, 4\rangle$, $\langle 10, 10, 4\rangle, \ldots, \langle 45, 45, 4\rangle$ classes of random feature subscriptions. For each class, 10 instances were generated and their mean results are reported in this paper.

We first investigated the impact of using different upper bounds on v. The notation *SP*$_0$ is used to denote the computation of the upper bound without any forward cost and *SP*$_i$, $1 \leq i \leq 4$, to denote the computation of the upper bound with fc_i as presented in Section 5. Due to the lack of space, the results of only the hardest problems that we experimented with are shown in Table 1. Notice that without forward cost SOFTPREC visits almost 50% more nodes. The computation of a tight forward cost, as in *SP*$_4$, pays off by saving time and nodes. In the remainder of this section, the results obtained by using *SP*$_4$ are denoted by SP.

We also compared the results of SP with the results obtained by maintaining AC, SAC and RSAC on the COP model as presented in Section 2. The results are shown in Table 2. Note that RSAC is known to be the most efficient CP approach for feature subscription problems [Lesaint *et al.*, 2008b]. The results suggest that AC is inferior to the other approaches by several orders of magnitude. Results also suggest that maintaining RSAC on the Boolean variables reduces the number of nodes and time significantly. Note that RSAC usually visits more nodes than those visited by SAC, but the difference between them is not that significant. SP outperforms all the approaches in terms of nodes and time. It spends at least 5 to 11 times less time than RSAC on hard instances.

7.2 Minimum Cutset Problem

We also demonstrate the efficiency of SOFTPREC on the well known minimum cutset problem [Garey and Johnson, 1979]. The minimum cutset problem consists of only hard precedence constraints and the task is to find the largest set of vertices such that the sub-graph induced by these vertices does not contain a cycle. We use the same instances as those used in [Barták and Čepek, 2008] and compare with their approach as well as with RSAC. We remark that when there

Table 2: Mean Results for Feature Subscription Problems.

	$\langle 50, 750, \{\prec, \succ\}\rangle$				$\langle 50, 500, \{\prec, \succ, \prec\succ\}\rangle$				$\langle 50, 250, \{\prec, \succ\}\rangle$			
subscription	AC	SAC	RSAC	SP	AC	SAC	RSAC	SP	AC	SAC	RSAC	SP
Results in terms of search nodes												
$\langle 30, 30, 4\rangle$	100,644	655	787	183	17,160	143	167	50	104,731	230	262	158
$\langle 35, 35, 4\rangle$	508,318	2,008	2,366	396	97,321	378	451	111	722,328	1,224	1,465	744
$\langle 40, 40, 5\rangle$	2,209,450	5,947	6,843	993	242,609	781	922	188	6,155,598	9,398	11,123	2,707
$\langle 45, 45, 4\rangle$	8,055,304	16,047	18,469	2,425	740,587	1,607	1,925	428	66,905,754	31,404	37,277	10,143
Results in terms of time (in seconds)												
$\langle 30, 30, 4\rangle$	24.27	3.92	3.52	0.50	2.49	0.54	0.48	0.15	20.34	1.16	1.04	0.33
$\langle 35, 35, 4\rangle$	146.72	15.01	13.68	1.52	16.18	1.82	1.69	0.44	183.49	9.13	8.70	2.16
$\langle 40, 40, 5\rangle$	779.77	50.62	47.34	4.71	47.73	4.43	4.09	0.83	1,836.39	81.77	76.59	9.46
$\langle 45, 45, 4\rangle$	3,468.08	164.83	149.36	13.86	164.89	11.03	10.19	2.26	23,009.72	345.10	320.01	45.72

are no user precedences the approach in [Barták and Čepek, 2008] is similar to SP_0. Figure 4 shows the results in terms of nodes using a logarithmic scale. Results in terms of time coincide with nodes, but they are not shown due to lack of space. SOFTPREC outperforms both the approaches significantly in terms of nodes and time. When there are 200 precedence constraints, SOFTPREC visits two orders of magnitude fewer nodes when compared with the approach in [Barták and Čepek, 2008], and three orders of magnitude fewer nodes when compared with those of RSAC.

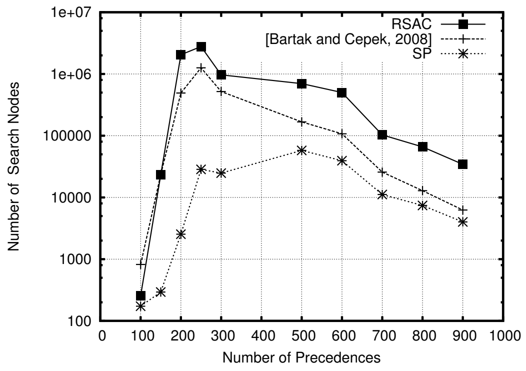

Figure 4: Results for minimum cutset problem with 50 variables.

8 Related Work

[Codish *et al.*, 2008] propose a SAT approach to the feature subscription problem by modeling it as a partial order constraint problem. Although the results reported in [Codish *et al.*, 2008] appear to be better than the results in [Lesaint *et al.*, 2008b], the results of the former were obtained on easier instances. A global constraint for cutset problems is proposed in [Fages and Lal, 2006]. The filtering rules of this global constraint are based on graph contraction operations. We still need to compare SOFTPREC with the cutset constraint.

9 Conclusions and Future Works

We proposed the soft global precedence constraint SOFTPREC. Since achieving GAC on SOFTPREC is NP-hard, we presented a set of rules that follow from the semantics of SOFTPREC. The pruning achieved by these rules approximates GAC on SOFTPREC. We introduced the notion of forward cost, and based on it, we proposed various upper bounds computation techniques. We also showed that the pruning achieved by the rules of SOFTPREC is tighter than the pruning achieved by AC on the discussed COP model. The former is incomparable with SAC on the COP model. Empirical results obtained using feature subscription and min-cutset problems demonstrate that SOFTPREC significantly outperforms the other approaches.

In our future work we would like to further investigate tighter bounds. It would also be interesting to investigate the performance of SOFTPREC on other benchmarks.

Acknowledgements

This material is based upon works supported by the Science Foundation Ireland under Grant No. 05/IN/I886, and Embark Post Doctoral Fellowships No. CT1080049908 and No. CT1080049909.

References

[Barták and Čepek, 2008] Roman Barták and Ondřej Čepek. Incremental filtering algorithms for precedence and dependency constraints. *IJAIT*, 17(1):205–221, 2008.

[Codish *et al.*, 2008] M. Codish, V. Lagoon, and P. J. Stuckey. Telecommunications feature subscription as a partial order constraint problem. In *ICLP*, pages 749–753, 2008.

[Fages and Lal, 2006] François Fages and Akash Lal. A constraint programming approach to cutset problems. *Comput. Oper. Res.*, 33(10):2852–2865, 2006.

[Garey and Johnson, 1979] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[Lesaint *et al.*, 2008a] D. Lesaint, D. Mehta, B. O'Sullivan, L. Quesada, and N. Wilson. Consistency techniques for finding an optimal relaxation of a feature subscription. In *ICTAI*, pages 283–290, 2008.

[Lesaint *et al.*, 2008b] D. Lesaint, D. Mehta, B. O'Sullivan, L. Quesada, and N. Wilson. Solving a Telecommunications Feature Subscription Configuration Problem. In *CP*, pages 67–81, 2008.

[Rossi *et al.*, 2006] Francesca Rossi, Peter van Beek, and Toby Walsh. *Handbook of Constraint Programming*. Elsevier Science Inc., 2006.