Disjoint, Partition and Intersection Constraints for Set and Multiset Variables [⋆]

Christian Bessiere[1], Emmanuel Hebrard[2], Brahim Hnich[2], and Toby Walsh[2]

[1] LIRMM, Montpelier, France. bessiere@lirmm.fr
[2] Cork Constraint Computation Center, University College Cork, Ireland. {e.hebrard, b.hnich, tw}@4c.ucc.ie

Abstract. We have started a systematic study of global constraints on set and multiset variables. We consider here disjoint, partition, and intersection constraints. These global constraints fall into one of three classes. In the first class, we show that we can decompose the constraint without hindering bound consistency. No new algorithms therefore need be developed for such constraints. In the second class, we show that decomposition hinders bound consistency but we can present efficient polynomial algorithms for enforcing bound consistency. Many of these algorithms exploit a dual viewpoint, and call upon existing global constraints for finite-domain variables like the global cardinality constraint. In the third class, we show that enforcing bound consistency is NP-hard. We have little choice therefore but to enforce a lesser level of local consistency when the size of such constraints grows.

1 Introduction

Global (or non-binary) constraints are one of the factors central to the success of constraint programming [8, 9, 1]. Global constraints permit the user to model a problem easily (by compactly specifying common patterns that occur in many problems) and solve it efficiently (by calling fast and effective constraint propagation algorithms). Many problems naturally involve sets and multisets. For example, the social golfers problem (prob010 at CSPLib.org) partitions a set of golfers into foursomes. Set or multiset variables have therefore been incorporated into most of the major constraint solvers (see, for example, [5, 7] for sets, [6] for multisets). There are few papers in the literature describing specialized propagators for global constraints on set and multiset variables. One exception is a recent report which describes a propagator for a global disjoint constraint on set variables with a fixed cardinality [10]. The aim of this paper is to study other such global constraints on set and multiset variables. Using the techniques proposed in [2], we have proved that some of these global constraints are NP-hard to propagate. For example, both the `atmost1-incommon` and `distinct` constraints on sets of fixed cardinality proposed in [11] are NP-hard to propagate. We prove that others are polynomial but not decomposable without hindering propagation. We therefore give efficient algorithms for enforcing bound consistency on such constraints.

[⋆] The last three authors are supported by Science Foundation Ireland and an ILOG software grant.

2 Taxonomy of global constraints

We have started to construct a taxonomy of global constraints over set and multiset variables. This consists of constraints composed from the following (more primitive) constraints.

Cardinality constraints: Many problems involve constraints on the cardinality of a set or multiset. For example, each shift must contain at least five nurses.

Intersection constraints: Many problems involve constraints on the intersection between any pair of sets or multisets. For example, shifts must have at least one person in common.

Partition constraints: Many problems involve partitioning a set or multiset. For example, orders must be partitioned to slabs in the steel mill slab design problem.

Ordering constraints: Many problems involve sets or multisets which are indistinguishable. For example, if each group in the social golfers problem is represented by a set then, as groups are symmetric, these sets can be permuted. We can break this symmetry by ordering the set variables.

Counting constraints: We often wish to model situations when there are constraints on the number of resources (values) used in a solution.

Weight and colour constraints: Many problems involve sets in which there is a weight or colour associated with each element of a set and there are constraints on the weights or colours in each set.

In this paper, we start a systematic study of the constraints within this taxonomy. We focus here on the intersection and partition constraints. Tables 1 and 2 summarize some of the results presented in this paper. All results apply to set or multiset variables unless otherwise indicated. Note that a constraint is bound consistency (BC) when the upper bound of each set or multiset variable is the union of all the consistent values, and the lower bound is the intersection. An alternative definition of BC for set and multiset variables is that the characteristic function (a vector of 0/1 variables) for each set variable, or the occurrence representation (a vector of integer variables representing the number of occurrences of the different values) for each multiset variable is bound consistent [12]. ¿From now on, “decomposable” means that there exists a decomposition into constraints on which BC is polynomial, and this decomposition does not hinder bound consistency. We will also use generalized arc consistency (GAC) on integer variables. A constraint is GAC iff every value for every variable can be extended to a solution of the constraint.

3 Disjoint constraints

`Disjoint`$(X_1, \ldots, X_n)$ iff $X_i \cap X_j = \{\}$ for any $i \neq j$.
`NEDisjoint`$(X_1, \ldots, X_n)$ iff $|X_i| > 0$ and $X_i \cap X_j = \{\}$ for any $i \neq j$.
`FCDisjoint`$(X_1, \ldots, X_n, k_1, \ldots, k_n)$ iff $|X_i| = k_i$ and $X_i \cap X_j = \{\}$ for any $i \neq j$.

The disjoint constraint on set or multiset variables is decomposable into binary empty intersection constraints without hindering bound consistency [12]. The non-empty and fixed cardinality disjoint constraints are not decomposable so we present

	$\forall i < j\ \ldots$			
$\forall k\ \ldots$	$\lvert X_i \cap X_j \rvert = 0$	$\lvert X_i \cap X_j \rvert \leq k$	$\lvert X_i \cap X_j \rvert \geq k$	$\lvert X_i \cap X_j \rvert = k$
-	`Disjoint` polynomial *decomposable*	`Intersect`$_\leq$ polynomial *decomposable*	`Intersect`$_\geq$ polynomial *decomposable*	`Intersect`$_=$ NP-hard *not decomposable*
$\lvert X_k \rvert > 0$	`NEDisjoint` polynomial *not decomposable*	`NEIntersect`$_\leq$ polynomial *decomposable*	`NEIntersect`$_\geq$ polynomial *decomposable*	`FCIntersect`$_=$ NP-hard *not decomposable*
$\lvert X_k \rvert = m_k$	`FCDisjoint` poly on sets, NP-hard on multisets *not decomposable*	`FCIntersect`$_\leq$ NP-hard *not decomposable*	`FCIntersect`$_\geq$ NP-hard *not decomposable*	`NEIntersect`$_=$ NP-hard *not decomposable*

Table 1. Intersection $\times$ Cardinality

	$\bigcup_i X_i = X\ \wedge\ \forall i < j\ \ldots$			
$\forall k\ \ldots$	$\lvert X_i \cap X_j \rvert = 0$	$\lvert X_i \cap X_j \rvert \leq k$	$\lvert X_i \cap X_j \rvert \geq k$	$\lvert S_i \cap S_j \rvert = k$
-	`Partition`: polynomial *decomposable*	?	?	?
$\lvert X_k \rvert > 0$	`NEPartition`: polynomial *not decomposable*	?	?	?
$\lvert X_k \rvert = m_k$	`FCPartition` polynomial on sets, NP-hard on multisets *not decomposable*	?	?	?

Table 2. Partition + Intersection $\times$ Cardinality

algorithms for enforcing on them or we prove intractability of enforcing bound consistency.

3.1 FCDisjoint

It is polynomial to enforce BC on the `FCDisjoint` constraint on set variables, but NP-hard on multisets. When $X_1, \ldots, X_n$ are set variables, we can achieve BC on a `FCDisjoint`$(X_1, \ldots, X_n, k_1, \ldots, k_n)$ constraint as follows:

Algorithm `BCFCDisjointsets`

```
1. For all $v \in \bigcup ub(X_i)$, introduce an integer variable $Y_v$ with $dom(Y_v) = \{\}$
2. Initialize the domain of each $Y_v$ as follows:
   (a) $dom(Y_v) \leftarrow \{X_i | v \in lb(X_i)\}$
   (b) if $|dom(Y_v)| > 1$ then **fail**
   (c) if $|dom(Y_v)| = 0$ then $dom(Y_v) \leftarrow \{X_i | v \in ub(X_i)\} \cup \{X_{n+1}\}$
3. Maintain GAC on gcc$(Y, X, B)$ where $Y$ is the array of $Y_i$'s, $X$ is an array of $X_i$'s (in-
   cluding the dummy set variable $X_{n+1}$), and $B$ is the array of the corresponding bounds of
   the $X_i$'s where for all $i \leq n$ we have $B[i] = k_i..k_i$ and $B[n+1] = 0..\infty$
4. Maintain the following channelling constraints, for all $i \leq n$ and for all $v$:
   (a) $X_i \in dom(Y_v) \leftrightarrow v \in ub(X_i)$
   (b) $dom(Y_v) = \{X_i\} \leftrightarrow v \in lb(X_i)$
```

Remark. `gcc`(Y, X, B) is the global cardinality constraint that imposes that in any assignment S of the variables Y, the value X_i from X appears a number of times in the range $B[i]$.

We first prove the following lemma.

Lemma 1. *Define the one-to-one mapping between assignments S of the dual variables Y and assignments S' of the original set variables X_i by: $v \in S'[X_i]$ iff $S[Y_v] = X_i$. Then S is consistent with* `gcc` *in step (3) of* `BCFCDisjointsets` *iff S' is consistent for* `FCDisjoint`*.*

Proof. ($\Rightarrow$) We prove that S' is:

Disjoint: Each dual variable Y_v has a unique value, say X_i. Therefore in S' a value v cannot appear in more than one of the variables $X_1 \dots X_n$. In the case where $Y_v = X_{n+1}$, v does not belong to any set variable assignment.

Fixed Cardinality: `gcc` ensures that the "values" X_i are used by exactly k_i dual variables Y_{v_j}. Hence, $|S'[X_i]| = k_i$.

($\Leftarrow$) We prove that S is:

Consistent with `gcc`: By construction of Y, if $|S'[X_i]| = k_i$ for each $i \in 1..n$, each "value" X_i will appear exactly k_i times in S, thus satisfying the `gcc`. (The dummy value X_{n+1} has no restriction on its number of occurrences in Y.)

Consistent with Y domains: By construction. □

Theorem 1. *`BCFCDisjointsets` is a sound and complete algorithm for enforcing bound consistency on `FCDisjoint` with set variables, that runs in $O(nd^2)$ time.*

Proof. *Soundness*. A value v is pruned from $ub(X_i)$ in step (4) of `BCFCDisjoint-sets` either because X_i was not put in $dom(Y_v)$ in step (2) or because the `gcc` has removed X_i from $dom(Y_v)$ in step (3). Lemma 1 tells us that both cases imply that v cannot belong to X_i in a satisfying tuple for `FCDisjoint`. A value v is added to $lb(X_i)$ in step (4) if $dom(Y_v) = \{X_i\}$ after applying GAC on the `gcc`. From Lemma 1 we deduce that any satisfying tuple for `FCDisjoint` necessarily contains v in X_i. We must also show that the algorithm does not fail if `FCDisjoint` can be made bounds consistent. `BCFCDisjointsets` can fail in only two different ways. First, it fails in step (2) if a value belongs to two different lower bounds. Clearly, `FCDisjoint` cannot then be made bounds consistent. Second, it fails in step (3) if the `gcc` cannot be made GAC. In this case, we know by Lemma 1 that `FCDisjoint` cannot then be made bounds consistent.

Completeness. Let $v \in ub(X_i)$ after step (4). Then, $X_i \in dom(Y_v)$ after step (3). The `gcc` being GAC, there exists an assignment S satisfying `gcc`, with $S[Y_v] = X_i$. Lemma 1 guarantees there exists an assignment S' with $\{v\} \subseteq S'[X_i]$. In addition, let $v \notin lb(X_i)$ after step (4). Then, there exists $X_{j \neq i} \in dom(Y_v)$ after step (3). Thus, there is an assignment S satisfying `gcc` with $S[Y_v] = X_j$. Lemma 1 tells us that there is a satisfying assignment S' of `FCDisjoint` with v not in $S'[X_i]$.

Complexity. Step (1) is in $O(d)$, and step (2) in $O(nd)$. Step (3) has the complexity of the `gcc`, namely $O(nd^2)$ since we have d variables with domains of size at most $n+1$. Step (4) is in $O(nd)$. Thus, `BCFCDisjointsets` is in $O(nd^2)$.

□

Theorem 2. *Enforcing bound consistency on `FCDisjoint` with multiset variables is NP-hard.*

Proof. We transform 3SAT into the problem of the existence of a satisfying assignment for `FCDisjoint`. Let $F = \{c_1, \dots, c_m\}$ be a 3CNF on the Boolean variables $x_1, \dots, x_n$. We build the constraint `FCDisjoint`$(X_1, \dots, X_{3n+m}, k_1, \dots, k_{3n+m})$ as follows. Each time a Boolean variable x_i appears positively (resp. negatively) in a clause c_j, we create a value v_i^j (resp. w_i^j). For each Boolean variable x_i, we create two values p_i and n_i. Then, we build the $3n + m$ multiset variables as follows.

1. $\forall i \in 1..n$, /* X_i will take the p_i's iff $x_i = 1$ */
   (a) k_i = number of occurrences of x_i in a clause
   (b) $\{\} \subseteq X_i \subseteq \{v_i^j \mid x_i \in c_j\} \cup \{k_i$ occurrences of $p_i\}$
2. $\forall i \in n+1..2n$, /* X_i will take the n_i's iff $x_i = 0$ */
   (a) k_i = number of occurrences of $\neg x_i$ in a clause
   (b) $\{\} \subseteq X_i \subseteq \{w_i^j \mid x_i \in c_j\} \cup \{k_i$ occurrences of $n_i\}$
3. $\forall i \in 2n+1..3n$, /* X_i forces X_{i-n} and X_{i-2n} to be consistent */
   (a) k_i = 1
   (b) $\{\} \subseteq X_i \subseteq \{n_i, p_i\}$
4. $\forall j \in 1..m$, /* X_{3n+j} represents the clause c_j */
   (a) k_{3n+j} = 1
   (b) $\{\} \subseteq X_{3n+j} \subseteq \{v_{i_1}^j, w_{i_2}^j, v_{i_3}^j\}$ if $c_j = x_{i_1} \vee \neg x_{i_2} \vee x_{i_3}$

Let M be a model of F. We build the assignment S on the X_i's such that $\forall i \in 1..n$, if $M[x_i] = 1$ $S[X_i] = \{p_i, \ldots, p_i\}$, $S[X_{i+n}] = \{w_i^j \in ub(X_{i+n})\}$, $S[X_{i+2n}] = \{n_i\}$, else $S[X_i] = \{v_i^j \in ub(X_i)\}$, $S[X_{i+n}] = \{n_i, \ldots, n_i\}$, $S[X_{i+2n}] = \{p_i\}$,

By construction, the cardinalities k_i are satisfied and the disjointness are satisfied on $X_1 \ldots, X_{3n}$. In addition, the construction ensures that if a Boolean variable x_i is true in M (resp. false in M) none of the v_i^j (resp. w_i^j) are used and all the w_i^j (resp. v_i^j) are already taken by $X_1 \ldots, X_{3n}$. Thus, $\forall j \in 1..m$, $S[X_{3n+j}]$ is assigned one of the values v_i^j or w_i^j representing a true literal x_i or $\neg x_i$ in M. And M being a 3SAT model, we are sure that there exists such values not already taken by $X_1 \ldots, X_{3n}$. Therefore, S satisfies `FCDisjoint`.

Consider now an assignment S of the X_i's consistent with `FCDisjoint`. Build the interpretation M such that $M[x_i] = 1$ iff $S[X_{i+2n}] = \{n_i\}$. Thanks to the disjointness and cardinalities among $X_1 \ldots, X_{3n}$, we guarantee that if $S[X_{i+2n}] = \{n_i\}$ all the w_i^j are already taken by X_{i+n}, and if $S[X_{i+2n}] = \{p_i\}$ all the v_i^j are already taken by X_i, so that they cannot belong to any X_{3n+j}. But S satisfying `FCDisjoint`, we know that for each $j \in 1..m$, X_{3n+j} is assigned a value consistent with $X_1 \ldots, X_{3n}$. Therefore, M is a model of F.

As a result, deciding the existence of a satisfying assignment for `FCDisjoint` with multiset variables is NP-complete. Then, deciding whether GAC finds a wipe out on the occurence representation is coNP-complete. In addition, on the transformation we use, if GAC detects a wipe then BC does[3] (because of the way p_i and n_i values are set). So, deciding whether BC detects a wipe out is coNP-complete, and enforcing bound consistency on `FCDisjoint` with multiset variables is NP-hard. □

3.2 NEDisjoint

The constraint `NEDisjoint`$(X_1, \ldots, X_n)$ on set variables can be seen as a particular case of constraint `FCDisjoint` in which the cardinality of the variables X_i can vary

[3] GAC on the occurence representation of multisets is in general not equivalent to BC (whilst on sets it is). If $ub(X_1) = ub(X_2) = \{1, 1, 2, 2\}$, and $k_1 = k_2 = 2$, GAC on the occurence representation of `FCDisjoint` removes the possibility for X_1 to have 1 occurence of 1. BC does not remove anything since the bounds 0 and 2 for $occ(1, X_1)$ are consistent.

from 1 to ∞ instead of being fixed to k_i. Since the way the algorithm `BC-FCDisjoint` is written permits to express such an interval of values for the cardinality of the set variables X_i, the algorithm `BC-NEDisjoint-sets` is a very simple modification of it. In step (3) of `BCFCDisjointsets` it is indeed sufficient to assign $B[i]$ to $1..\infty$ instead of $k_i..k_i$, for $1 \leq i \leq n$.

When `NEDisjoint` involves multiset variables, BC remains polynomial. In fact, it is sufficient to transform the multisets in sets and to use `BC-NEDisjoint-sets` on the obtained sets. Once BC achieved on these sets, we just have to restore the initial number of occurrences, noted init-occ, for each remaining value. The cardinality of the multisets are not bounded above, so that if one value has support, any number of occurrences of the same value have support also.

Algorithm `BC-NEDisjoint-msets`

```
1. for each $i \in 1..n, v$ occurring in $ub(X_i)$ do
       init-occ$_{ub}(X_i, v) \leftarrow$ occ$(v, ub(X_i))$; occ$(v, ub(X_i)) \leftarrow 1$
       init-occ$_{lb}(X_i, v) \leftarrow$ occ$(v, lb(X_i))$; occ$(v, lb(X_i)) \leftarrow min(1,$ init-occ$_{lb}(X_i, v))$
2. BC-NEDisjoint-sets$(X_1, \ldots, X_n)$
3. for each $i \in 1..n, v \in ub(X_i)$ do
       occ$(v, ub(X_i)) \leftarrow$ init-occ$_{ub}(X_i, v)$
       if $v \in lb(X_i)$ then occ$(v, lb(X_i)) \leftarrow max(1,$ init-occ$_{lb}(X_i, v))$
```

4 Partition constraints

`Partition`$(X_1, \ldots, X_n, X)$ iff $X_i \cap X_j = \{\}$ for any $i \neq j$ and $\bigcup_i X_i = X$.
`NEPartition`$(X_1, \ldots, X_n, X)$ iff $|X_i| > 0$, $X_i \cap X_j = \{\}$ for any $i \neq j$ and $\bigcup_i X_i = X$.
`FCPartition`$(X_1, \ldots, X_n, X, k_1, \ldots, k_n)$ iff $|X_i| = k_i$, $X_i \cap X_j = \{\}$ for any $i \neq j$ and $\bigcup_i X_i = X$.

The partition constraint is decomposable into binary empty intersection constraints and ternary union constraints involving n additional variables without hindering bound consistency [12]. On the other hand, the non-empty and fixed cardinality partition constraints are not decomposable. We therefore present algorithms for enforcing BC on them or we prove intractability of enforcing BC.

4.1 FCPartition

It is polynomial to enforce BC on the `FCPartition` constraint on set variables, but NP-hard on multisets. On set variables, enforcing BC on `FCPartition` is very similar to enforcing BC on `FCDisjoint`. Indeed, if the set X being partitioned is fixed, then we can simply decompose a fixed cardinality partition constraint into a fixed cardinality disjoint, union and cardinality constraints without hindering bound consistency. If X is not fixed, we need to do slightly more reasoning to ensure that the X_i's are a partition of X. We present here the additional lines necessary to deal with this.

Line numbers with a prime represent lines modified wrt `BCFCDisjointsets`. The others are additional lines.

Algorithm `BC-FCPartitionsets`

```
1'. For all $v \in ub(X)$, introduce an integer variable $Y_v$ with $dom(Y_v) = \{\}$
2. Initialize the domain of each $Y_v$ as follows:
    ...
    (c') if $|dom(Y_v)| = 0$ then $dom(Y_v) \leftarrow \{X_i | v \in ub(X_i)\}$
    (d) if $v \notin lb(X)$ then $dom(Y_v) \leftarrow dom(Y_v) \cup \{X_{n+1}\}$
    (e) if $|dom(Y_v)| = 0$ then fail
...
4. Maintain the following channelling constraints, for all $i \leq n$ and for all $v$:
    ...
    (c) $X_{n+1} \notin dom(Y_v) \leftrightarrow v \in lb(X)$
    (d) $ub(X) \leftarrow \bigcup ub(X_i)$
```

Lemma 2. *Define the one-to-one mapping between assignments S of the dual variables Y and assignments S' of the original set variables X_i and X by: $S'[X] = \bigcup S'[X_i]$ and $v \in S'[X_i]$ iff $S[Y_v] = X_i$. Then S is consistent with* `gcc` *in step (3) of* `BC-FCPartitionsets` *iff S' is consistent for* `FCPartition`*.*

Proof. ($\Rightarrow$) We prove that S' is:

Disjoint and Fixed Cardinality: See Lemma 1.

Partition: Lines (2.c'-d) guarantee that a value $v \in lb(X)$ cannot be assigned the dummy variable in S. Hence, S' necessarily has an X_i with $v \in S'[X_i]$. Because of line (1'), none of the Y_v represent a value $v \notin ub(X)$. Hence, for all i, $S'[X_i] \subseteq ub(X)$, then $S'[X] \subseteq ub(X)$.

($\Leftarrow$) We prove that S is:

Consistent with `gcc`*:* See Lemma 1.

Consistent with Y: If S' is a satisfying assignment for `FCPartition`, $S'[X_i] \subseteq S'[X], \forall i$. Since $S'[X] \subseteq ub(X)$, we know that any value v appearing in S' has a corresponding variable Y_v. And by construction (lines 2.a, 2.c, 2.d, we know that S is consistent with Y domains. □

Theorem 3. `BC-FCPartitionsets` *is a sound and complete algorithm for enforcing bound consistency on* `FCPartition` *with set variables that runs in $O(nd^2)$ time.*

Proof. Soundness. A value v is pruned from $ub(X_i)$ in step (4) of `BC-FCPartitionsets` for one of the reasons that already held in `FCDisjoint` or because Y_v has not been created in line (1'). Lemma 2 tells us that all cases imply that v cannot belong to X_i in a satisfying tuple for `FCPartition`. Soundness of $lb(X_i)$ comes from Lemma 2 as it came from Lemma 1 on `FCDisjoint`. We must also show that the algorithm does not fail if `FCPartition` can be made bounds consistent. `BC-FCPartitionsets` can fail in line (2.e) if a value v that belongs to $lb(X)$ cannot belong to any X_i. Clearly, `FCPartition` cannot then be made bounds consistent. The other cases of failure are the same as for `FCDisjoint`. A value v is pruned from $ub(X)$ in step (4.d) because none of the X_i contains v in its upper bound. This means that this value cannot belong to any satisfying assignment S' (Lemma 2). A value v is added to $lb(X)$ in line (4.c) if no assignment S satisfying the `gcc` verifies $S[Y_v] = X_{n+1}$. This means that v is assigned to a variable X_i in all assignments satisfying `FCPartition`.

Completeness. Let $v \in ub(X)$ after step (4). Then, there exists X_i with $v \in ub(X_i)$, and so $X_i \in dom(Y_v)$ after step (3). `gcc` being GAC, there exists an assignment S satisfying `gcc`, with $S[Y_v] = X_i$. Lemma 2 guarantees there exists an assignment S' with $\{v\} \subseteq S'[X]$. Thus, v is in $ub(X)$. In addition, let $v \notin lb(X)$ after step (4). Then, $X_{n+1} \in dom(Y_v)$ after step (3). Thus, there is an assignment S satisfying `gcc` with $S[Y_v] = X_{n+1}$. Lemma 2 tells us that there is a satisfying assignment S' of `FCPartition` with v not in $S'[X]$.

Complexity. See proof of `BCFCDisjointsets`. □

Theorem 4. *Enforcing BC on `FCPartition`$(X_1, \ldots, X_n, X, k_1, \ldots, k_n)$ with multiset variables is NP-hard.*

Proof. We know that deciding the existence of a satisfying assignment is NP-complete for `FCDisjoint`$(X_1, \ldots, X_n, k_1, \ldots, k_n)$ with multiset variables. If we build a multiset variable X with $lb(X) = \emptyset$ and $ub(X) = \bigcup_i ub(X_i)$, then `FCPartition`$(X_1, \ldots, X_n, X, k_1, \ldots, k_n)$ has a satisfying assignment if and only if `FCDisjoint`$(X_1, \ldots, X_n, k_1, \ldots, k_n)$ has one. Thus, enforcing bound consistency on `FCPartition` is NP-hard. □

4.2 NEPartition

The constraint `NEDisjoint`$(X_1, \ldots, X_n)$ on set variables was a particular case of `FCDisjoint` in which the cardinality of the variables X_i can vary from 1 to ∞ instead of being fixed to k_i. This is exactly the same for `NEPartition` on set variables, which is a particular case of `FCPartition`. Replacing "$B[i] \leftarrow k_i..k_i$" by "$B[i] \leftarrow 1..\infty$" in `BC-FCPartitionsets`, we obtain `BC-NEPartition-sets`.

When `NEPartition` involves multiset variables, BC remains polynomial. As for `BC-NEDisjoint-msets`, the trick is to transform multisets in sets and to use `BC-NEPartition-sets` on the obtained sets. We just need to be careful with the compatibility of the occurrences of values in X_i variables and the X being partitioned. Once BC is achieved on these sets, we have to restore the initial number of occurrences and check again compatibility with X.

Algorithm `BC-NEPartition-msets`

```
1. if $\bigcup_i lb(X_i) \not\subseteq ub(X)$ or $lb(X) \not\subseteq \bigcup_i ub(X_i)$ then failure
2. for each $i \in 1..n, v$ occurring in $ub(X_i)$ do
     2.1. if $\texttt{occ}(v, ub(X_i)) < \texttt{occ}(v, lb(X))$ then $\texttt{occ}(v, ub(X_i)) \leftarrow 0$
     2.2. if $\texttt{occ}(v, ub(X_i)) > \texttt{occ}(v, ub(X))$ then $\texttt{occ}(v, ub(X_i)) \leftarrow \texttt{occ}(v, ub(X))$
     2.3. $\text{init-occ}_{ub}(X_i, v) \leftarrow \texttt{occ}(v, ub(X_i))$; $\texttt{occ}(v, ub(X_i)) \leftarrow 1$
     2.4. $\text{init-occ}_{lb}(X_i, v) \leftarrow \texttt{occ}(v, lb(X_i))$; $\texttt{occ}(v, lb(X_i)) \leftarrow min(1, \text{init-occ}_{lb}(X_i, v))$
3. store $lb(X)$; $lb(X) \leftarrow \texttt{set-of}(lb(X))$; $ub(X) \leftarrow \texttt{set-of}(ub(X))$
4. BC-NEPartition-sets$(X_1, \ldots, X_n, X)$
5. restore $lb(X)$
6. for each $i \in 1..n, v \in ub(X_i)$ do
     6.1. $\texttt{occ}(v, ub(X_i)) \leftarrow \text{init-occ}_{ub}(X_i, v)$
     6.2. if $v \in lb(X_i)$ then $\texttt{occ}(v, lb(X_i)) \leftarrow max(1, \text{init-occ}_{lb}(X_i, v), \texttt{occ}(v, lb(X)))$
7. $lb(X) \leftarrow lb(X) \cup \bigcup_i lb(X_i)$; $ub(X) \leftarrow \bigcup_i ub(X_i)$
```

Theorem 5. *`BC-NEPartition-msets` is a sound and complete algorithm for enforcing bound consistency on `NEPartition` with multiset variables, that runs in $O(nd^2)$ time.*

Proof. (Sketch.) As for `NEDisjoint` on multiset variables, enforcing bound consistency on `NEPartition` after having transformed the multisets in sets (i.e., we keep only one occurrence of each value in the lower and upper bounds), the removal of a value v from an upper bound by `BC-NEPartition-sets` is a sufficient condition for the removal of all occurrences of v in the original multiset upper bound. The addition v to a lower bound is a sufficient condition for the addition of some occurrences of v in the lower bound (the right number depends on the number of occurrences of v in $lb(X)$ and in the lower bound of the X_i holding v. It is then sufficient to ensure consistency between the number of occurrences in the X_i and X (lines 1, 2.1, 2.2, and 7), to transform multisets in sets (lines 2.3, 2.4, and 3), to call `BC-NEPartition-sets` (line 4), and to restore appropriate numbers of occurrences (lines 5 and 6). Line 1 guarantees that $ub(X)$ can cover all the X_i's lower bounds and that $lb(X)$ can be covered by the X_i's upper bounds. A value v can be assigned in X_i if and only if it can cover the occurrences of v in $lb(X)$ (line 2.1), and it cannot occur more than in $ub(X)$ (line 2.2). Finally, a value v occurs in $lb(X)$ at least as many times as it occurs in some $lb(X_i)$, and occurs in $ub(X)$ exactly as many times as in the $ub(X_i)$ having its greatest number of occurrences (line 7). The complexity is dominated by line 4, with the call to `BC-NEPartition-sets`which is $O(nd^2)$.

5 Intersection constraints

The `Disjoint` constraint restricts the pair-wise intersection of any two set or multiset variables to the empty set. We new consider the cases where the cardinality of the pair-wise intersection is either bounded or equal to a given constant or integer variable:

`Intersect`$_{\leq}(X_1, \ldots, X_n, K)$ iff $|X_i \cap X_j| \leq K$ for any $i \neq j$.

`Intersect`$_{\geq}(X_1, \ldots, X_n, K)$ iff $|X_i \cap X_j| \geq K$ for any $i \neq j$.

`Intersect`$_{=}(X_1, \ldots, X_n, K)$ iff $|X_i \cap X_j| = K$ for any $i \neq j$.

As usual, we can also add non-emptiness and fixed cardinality constraints to the set or multiset variables. For example, `FCIntersect`$_{\leq}(X_1, \ldots, X_n, K, C)$ iff $|X_i \cap X_j| \leq K$ for any $i \neq j$ and $|X_i| = C$ for all i. If $K = 0$, `Intersect`$_{\leq}$ and `Intersect`$_{=}$ are equivalent to `Disjoint`.

5.1 At most intersection constraints

We show that `Intersect`$_{\leq}$ and `NEIntersect`$_{\leq}$ can be decomposed without hindering bound consistency, but that it is NP-hard to enforce BC on `FCIntersect`$_{\leq}$.

Theorem 6. *$BC($`Intersect`$_{\leq}(X_1, \ldots, X_n, K)$ is equivalent to $BC(|X_i \cap X_j| \leq K)$ for all $i < j$.*

Proof. Suppose $BC(|X_i \cap X_j| \leq K)$ for all $i < j$. We will show that $BC(\forall i < j. |X_i \cap X_j| \leq K)$. Consider the occurrence representation of the set or multiset variables. Let

X_{il} be the number of occurrences of the value l in X_i. Consider the upper or lower bound of X_{il}. We will construct a support for the upper or lower bound of X_{il} that simultaneously satisfies $|X_i \cap X_j| \leq K$ for all $i < j$. First, we assign K with its upper bound. Then we pick any j with $i \neq j$. As $BC(|X_i \cap X_j| \leq K)$, there is some assignment for X_{jn} and X_{im} ($l \neq m$) within their bounds that satisfies $|X_i \cap X_j| \leq K$. We now extend these assignments to get a complete assignment for every other set or multiset variable as follows. Every other X_{pq} ($p \neq i$ and $p \neq j$) is assigned its lower bound. This can only help satisfy $|X_i \cap X_j| \leq K$ for all $i < j$. This assignment is therefore support for X_{il}. We can also construct support for the upper of lower bound of K in a similar way. Maximality of the bounds consistent domains is easy. Consider any value for X_{il} smaller than the lower bound or larger than the upper bound. As this cannot be extended to satisfy $|X_i \cap X_j| \leq K$ for some j, it clearly cannot be extended to satisfy $|X_i \cap X_j| \leq K$ for all $i < j$. A similar argument holds for any value for K smaller than the lower bound or larger than the upper bound. Hence, $BC(\forall i < j.|X_i \cap X_j| \leq K)$. □

Given a set of set or multiset variables, the non-empty intersection constraint `NEIntersect`$_{\leq}(X_1, \ldots, X_n, K)$ ensures that $|X_i \cap X_j| \leq K$ for $i \neq j$ and $|X_i| > 0$ for all i. If $K = 0$, this is the `NEDisjoint` constraint which is not decomposable. If $K > 0$, the constraint is decomposable.

Theorem 7. *If $K > 0$ then $BC($`NEIntersect`$_{\leq}(X_1, \ldots, X_n, K)$ is equivalent to $BC(|X_i \cap X_j| \leq K)$ for all $i < j$ and $BC(|X_i| > 0)$ for all i.*

Proof. Suppose $BC(|X_i \cap X_j| \leq K)$ for all $i < j$ and $BC(|X_i| > 0)$ for all i. Then $|lb(X_i) \cap lb(X_j)| \leq max(K)$ for all $i < j$. And if $|ub(X_i)| = 1$ for any i then $lb(X_i) = ub(X_i)$. Consider some variable X_i and any value $a \in ub(X_i) - lb(X_i)$. We will find support in the global constraint for X_i to take the value $\{a\} \cup lb(X_i)$. Consider any other variable X_j. If $|lb(X_j)| = 0$ then we pick any value $b \in ub(X_j)$ and set X_j to $\{b\}$. This will ensure we satisfy the non-emptiness constraint on X_j. As $k > 0$ and $|X_j| = 1$, we will satisfy the intersection constraint between X_j and any other variable. If $|lb(X_j)| > 0$ then we set X_j to $lb(X_j)$. This again satisfy the non-emptiness constraint on X_j. Since $|lb(X_i) \cap lb(X_j)| \leq max(K)$ for all $i < j$, we will also satisfy the intersection constraints. Support can be found in a similar way for X_i to take the value $lb(X_i)$ if this is non-empty. Finally, $min(K)$ has support since $BC(|X_i \cap X_j| \leq K)$ for all $i < j$. Hence `NEIntersect`$_{\leq}(X_1, \ldots, X_n, K)$ is BC. □

Enforcing BC on `FCIntersect`$_{\leq}$ is intractable.

Theorem 8. *Enforcing BC on `FCIntersect`$_{\leq}(X_1, \ldots, X_n, k, c)$ for $c > k > 0$ is NP-hard.*

Proof. Immediate from Theorem 5 in [2]. □

Sadler and Gervet introduce the `atmost1-incommon` and `distinct` constraints for set variables with a fixed cardinality [11]. The `atmost1-incommon` constraint is `FCIntersect`$_{\leq}(X_1, \ldots, X_n, 1, c)$. Similarly, the `distinct` constraint on sets of fixed cardinality is is `FCIntersect`$_{\leq}(X_1, \ldots, X_n, c - 1, c)$. The reduction used in Theorem 5 in [2] works with all these parameters. Hence, all are NP-hard to propagate.

5.2 At least intersection constraints

Similar to the at most intersection constraint, $\texttt{Intersect}_{\geq}$ and $\texttt{NEIntersect}_{\geq}$ can be decomposed without hindering bound consistency, but that it is NP-hard to enforce BC on $\texttt{FCIntersect}_{\geq}$.

Theorem 9. *$BC(\forall i < j. |X_i \cap X_j| \geq K)$ is equivalent to $BC(|X_i \cap X_j| \geq K)$ for all $i < j$.*

Proof. The proof is analogous to that of Theorem 6 except we extend a partial assignment to a complete assignment that is interval support by assigning each of the additional X_{pq} with the upper bound and (where appropriate) K with its lower bound. □

Two sets cannot have an intersection unless they are non-empty. Hence this result also shows that $BC(\texttt{NEIntersect}_{\geq}(X_1, \ldots, X_n, K)$ for $K > 0$ is equivalent to BC on the decomposition. By comparison, enforcing BC on $\texttt{FCIntersect}_{\geq}$ is intractable.

Theorem 10. *Enforcing BC on $\texttt{FCIntersect}_{\geq}(X_1, \ldots, X_n, k, c)$ for $c > k > 0$ is NP-hard.*

Proof. We let $k = 1$. We can reduce the $k = 1$ case to the $k > 1$ case by adding $k - 1$ additional common values to each set variable. The proof again uses a reduction of a 3SAT problem in n variables. The same reduction is used for set or multiset variables. We let $c = n$ and introduce a set variable, S with domain $\{\} \subseteq S \subseteq \{1, \neg 1, \ldots, n, \neg n\}$. This will be set of literals assigned true in a satisfying assignment. For each clause, φ we introduce a set variable, X_φ with fixed domain. Suppose $\varphi = x_i \vee \neg x_j \vee x_k$, then X_φ has the domain $\{i, \neg j, k, d_1, \ldots, d_{n-3}\}$. To satisfy the intersection and cardinality constraint, S must take at least one of the literals which satisfy φ. Finally, we introduce n set variables, X_i to ensure that only one of i and $\neg i$ is in S. Each X_i has domain $\{d_1, \ldots, d_{n-1}\} \subseteq X_i \subseteq \{d_1, \ldots, d_{n-1} i, \neg i\}$. The constructed set variables then have a solution which satisfies the intersection and cardinality constraints iff the original 3SAT problem is satisfiable. Hence testing a value for support is NP-complete, and enforcing bound consistency is NP-hard. □

5.3 Equal intersection constraints

Unlike the at most or at least intersection constraints, enforcing BC on $\texttt{Intersect}_{=}$ is intractable even without cardinality constraints on the individual set or multiset variables.

Theorem 11. *Enforcing BC on $\texttt{Intersect}_{=}(X_1, \ldots, X_n, k)$ is NP-hard for $k > 0$.*

Proof. Immediate from Theorem 6 in [2]. □

The same reduction can also be used with the constraint that each set or multiset has a non-empty or fixed cardinality. Thus, enforcing BC on $\texttt{FCIntersect}_{=}(X_1, \ldots, X_n, K)$ or $\texttt{NEIntersect}_{=}(X_1, \ldots, X_n, K, C)$ is intractable.

Lemma 3. *Enforcing BC on $\texttt{FCIntersect}_{=}(X_1, \ldots, X_n, k)$ is NP-hard for $k > 0$.*

Lemma 4. *Enforcing BC on $\texttt{NEIntersect}_{=}(X_1, \ldots, X_n, K, C)$ is NP-hard for $k > 0$.*

6 Ordering constraints

Multiset ordering constraints are useful for breaking symmetry on set or multiset variables. We say $X <_{\text{mset}} Y$ iff X is empty and Y is not, or the largest value in X is smaller than the largest value in Y, or the largest values are the same and, if we elimante the one occurrence of the largest value from both X and Y, the result is multiset ordered. We can enforce BC on such a constraint with the lex ordering algorithm [4] applied to the occurrence representation. Similarly, if we have a chain of ordering constraints on set or multiset variables, $X_0 \leq_{\text{mset}} \ldots \leq_{\text{mset}} X_n$ we can enforce BC on this chain by using the lex-chain algorithm of [3].

We may wish to combine ordering constraints with other constraints on set or multiset variables. For example, suppose we want to find a set of sets that have at most one element in common with each other [11]. We can enforce BC on $\texttt{FCIntersect}_{\leq}(X_i, \ldots, X_n, 1)$ in polynomial time. However, numbering the set variables from 1 to n introduces an unnecessary symmetry. We can break this symmetry by ordering $X_1 <_{\text{mset}} \ldots <_{\text{mset}} X_n$. We might therefore decide to post an ordered intersection constraint. Unfortunately, combining an ordering constraint on set or multiset variables with other constraints often makes constraint propagation intractable.

Theorem 12. *Enforcing BC($\forall i < j.|X_i \cap X_j| \leq k$) is polynomial but enforcing BC($\forall i < j.|X_i \cap X_j| \leq k$ and $X_1 <_{\text{mset}} \ldots <_{\text{mset}} X_n$) for $k > 0$ is NP-hard.*

Proof. The proof is similar to the NP-hardness of enforcing BC on an $\texttt{FCIntersect}_{\leq}$. We let $k = 1$ as we can reduce the $k = 1$ case to the $k > 1$ case by adding $k - 1$ additional common values to each set variable.

The proof uses a reduction from 3SAT. For each of the m clauses, φ_i (here i is a numerical index), we introduce two set variables, X_i and Y_i. Suppose $\varphi_i = x \vee \neg y \vee z$, then X_i has the domain $\{i\}$ whilst Y_i has the domain $\{i\} \subseteq Y_i \subseteq \{i, x_i, \neg y_i, z_i\}$. We will assume that the index i is larger than the literals $x_i, \neg y_i, z_i$ in our multiset ordering, and that we have any ordering on these literals. By ordering $X_i <_{\text{mset}} Y_i$, we must have $|Y_i| > 1$. We interpret the literals assigned to Y_i as those that must be $true$. We also insist that $X_1 <_{\text{mset}} Y_1 <_{\text{mset}} \ldots <_{\text{mset}} X_m <_{\text{mset}} Y_m$. The set variables are therefore totally ordered.

We will use an additional (at most quadratic) number of set variables ordered after Y_m. These ensure that two clauses do not use contradictory satisfying assignments. We describe here how we construct the kth to $k + 3$th such set variables. Consider any other clause, φ_j which contains a contradictory literal to φ_i. For example, suppose φ_j contains $\neg x$. We have four set variables: $X_{ijx} <_{\text{mset}} Y_{ijx} <_{\text{mset}} Z_{ijx} <_{\text{mset}} W_{ijx}$ with domains $X_{ijx} = \{m + k, i\}$, $\{m + k, i\} \subseteq Y_{ijx} \subseteq \{m + k, i, x_i, \neg x_j\}$, $Z_{ijx} = \{m+k+1, \neg x_j\}$ and $\{m+k+1, \neg x_j\} \subseteq W_{ijx} \subseteq \{m+k+1, \neg x_j, i, j\}$. Suppose we satisfy φ_i by assigning x to $true$. That is, $\{i, x_i\} \subseteq Y_i$. Then $Y_{ijx} = \{m + k, i, \neg x_j\}$ and $Z_{ijx} = \{m+k+1, \neg x_j, j\}$. Hence, Y_j cannot contain $\{j, \neg x_j\}$ or the intersection constraint between Y_j and Z_{ijx} will be violated. That is, φ_j cannot be satisfied by $\neg x$ being assigned $true$. Some other literal in φ_j will have to satisfy the clause.

The constructed set variables thus have a solution which satisfies the ordered intersection constraint iff the original 3SAT problem is satisfiable. Hence enforcing bound consistency is NP-hard. □

This proof leaves open the case of $k = 0$ (i.e. ordered disjoint).

Theorem 13. *Enforcing BC($\forall i < j.|X_i \cap X_j| \geq k$) is polynomial but enforcing BC($\forall i < j.|X_i \cap X_j| \geq k$ and $X_1 <_{\mathrm{mset}} \ldots <_{\mathrm{mset}} X_n$) for $k > 0$ is NP-hard.*

Proof. The proof is somewhat different to the NP-hardness of enforcing BC on `FCIntersect`$_{\geq}$. We let $k = 1$. As before, we can reduce the $k = 1$ case to the $k > 1$ case by adding $k - 1$ additional common values to each set variable.

The proof uses a reduction from 3SAT. We introduce $2n$ set variables, X_i and Y_i to represent an assignment to the n Boolean variables which satisfies the clauses. If x is the ith of the n Boolean variables, then $\{d, i\} \subseteq X_i \subseteq \{d, i, x, \neg x\}$ and $Y_i = \{d, i, x, \neg x\}$. We will assume that the value d is larger than any index i, that the index i is larger than any of the literals, and that x and $\neg x$ are the $2i$th and $2i + 1$ literals in the ordering. By ordering $X_i <_{\mathrm{mset}} Y_i$, we cannot have both x and $\neg x$ in X_i. $x \in X_i$ is interpreted as x being assigned $true$, whilst $\neg x \in X_i$ is interpreted as x being assigned $false$. We also insist that $X_1 <_{\mathrm{mset}} Y_1 <_{\mathrm{mset}} \ldots <_{\mathrm{mset}} X_n <_{\mathrm{mset}} Y_n$. The set variables are therefore totally ordered.

For each of the clauses, φ_m (m is a numerical index), we introduce two set variables, U_m and V_m. Suppose $\varphi_m = x \vee \neg y \vee z$ and x, y, z are respectively the ith, jth and kth variables in the ordering. Then $\{e, n + m, 1, \ldots, n, x, \neg y, z\} - \{i, j, k\} \subseteq U_m \subseteq \{e, n+m, 1, \ldots, n, x, \neg y, z\}$, and $V_m = \{e, n+m, 1, \ldots, n, x, \neg y, z\}$. We will assume that the value e is larger than d. We also insist that $Y_n <_{\mathrm{mset}} U_1 <_{\mathrm{mset}} V_1 <_{\mathrm{mset}} \ldots <_{\mathrm{mset}} U_m <_{\mathrm{mset}} V_m$. The set variables are therefore totally ordered. To satisfy the ordering constraint $U_m <_{\mathrm{mset}} V_m$, at least one of i, j and k cannot be in U_m. Suppose $j \notin U_m$, then to satisfy the intersection constraint between U_m and Y_j, $\neg y$ must be in Y_j. A similar argument holds when $i \notin U_m$ and when $k \notin U_m$. Hence, at least one of x, $\neg y$, z must be in X_i, X_j or X_k respectively. That is, the clause φ_m must be satisfied.

The constructed set variables thus have a solution which satisfies the ordered intersection constraint iff the original 3SAT problem is satisfiable. Hence enforcing bound consistency is NP-hard. □

7 Experimental results

To illustrate the benefits of these global constraints, we ran some experiments using ILOG's Solver toolkit using a popular benchmark involving set variables. The social golfers problem $\langle p, m, n, t\rangle$ is to schedule m groups of n golfers over p weeks, such that no golfer plays in the same group as any other golfer twice. When the total number of golfers t is $n*m$, then each week is a partitioning of the set of golfers into groups. To model this problem, we introduce a set variable G_i^j of fixed cardinality n representing the ith group in the jth week. Each week is a partition of the set of golfers. Between any two groups, their intersection must contain at most one golfer. We also consider a generalization of the problem in which there is an excess of golfers and some golfers rest each week. We then replace the fixed cardinality partitioning constraint with a fixed cardinality disjoint constraint.

To measure the effectiveness of our global constraints we ran some experiments with a time limit of 10 minutes. We arbitrarily chose five settings for m and n. For each, we present the results for all numbers p of weeks such that at least one strategy needs at least one fail, and at least one strategy can solve the problem within the time limit. We looked at problems with up to 4 golfers resting each week. We solved each problem using five different variable ordering strategies either with the new propagator or with its decomposition.

- **static golfer:** Pick each golfer in turn, and assign him to the first possible group of every week.
- **static week:** Pick each golfer in turn, and assign him to one group in the first incomplete week.
- **min domain:** Pick a pair (golfer, week) such that the total number of groups in which the golfer can participate in during the given week is minimum. Then assign this golfer to one group.
- **default (group):** ILOG Solver's default strategy for set variables ordered by groups.
- **default (week):** ILOG Solver's default strategy for set variables ordered by weeks.

Number of Fails

problem	model	static golfer	static week	min domain	group (set)	week (set)
$\langle 6, 8, 4, 36\rangle$	global constraint	**10**	-	52	183	-
	decomposition	-	-	190	183	-
$\langle 3, 6, 6, 37\rangle$	global constraint	-	548	**0**	27	22232
	decomposition	-	-	**0**	27	22232
$\langle 3, 6, 6, 38\rangle$	global constraint	-	67	**0**	4	3446
	decomposition	-	-	**0**	4	3446
$\langle 3, 6, 6, 39\rangle$	global constraint	-	1261	**0**	7	171574
	decomposition	-	-	**0**	7	171574
$\langle 3, 6, 6, 40\rangle$	global constraint	12	48	**0**	**0**	8767
	decomposition	-	-	**0**	**0**	8767
$\langle 3, 5, 5, 26\rangle$	global constraint	-	44	**0**	2	813
	decomposition	-	177880	**0**	2	813
$\langle 3, 5, 5, 27\rangle$	global constraint	967161	5	**0**	1	62
	decomposition	-	1106	**0**	1	62
$\langle 3, 5, 5, 28\rangle$	global constraint	9	32	**0**	19	661
	decomposition	58218	22860	**0**	19	661
$\langle 3, 5, 5, 29\rangle$	global constraint	6	2	**0**	**0**	18
	decomposition	37208	209	**0**	**0**	18
$\langle 3, 9, 9, 83\rangle$	global constraint	-	-	**0**	453	-
	decomposition	-	-	-	453	-
$\langle 3, 9, 9, 84\rangle$	global constraint	-	-	**0**	5	-
	decomposition	-	-	-	5	-
$\langle 3, 9, 9, 85\rangle$	global constraint	-	-	**0**	30	-
	decomposition	-	-	1442064	30	-
$\langle 10, 9, 3, 30\rangle$	global constraint	464	264	-	16055	**15**
	decomposition	-	-	-	16055	**15**
$\langle 10, 9, 3, 31\rangle$	global constraint	37	1	**0**	2	113
	decomposition	-	51223	**0**	2	113

We observe that, in terms of fails, either our global constraint outperforms the decomposition or they are the same. The runtimes follow a similar behaviour. With the two default heuristics (corresponding to the last two columns in the table), we notice no difference between our global constraint and the decomposition. These heuristics are not, however, always the best. The min domain heuristic is often superior, but sometimes needs the pruning provided by the global constraint to prevent it performing poorly (see, for example, the $\langle 3, 9, 9, 85\rangle$ problem).

8 Conclusions

We have begun a systematic study of global constraints on set and multiset variables. We have studied here a wide range of disjoint, partition, and intersection constraints. The disjoint constraint on set or multiset variables is decomposable (and hence polynomial). On the other hand, the non-empty and fixed cardinality disjoint constraints are not decomposable without hindering bound consistency. We therefore present polynomial algorithms for enforcing bound consistency on the non-empty disjoint constraints for set or multiset variables, for enforcing BC on the fixed cardinality disjoint constraint for set variables, and prove that enforcing BC on the fixed cardinality disjoint constraint on multiset variables is NP-hard. We give very similar results for the partition, non-empty and fixed cardinality partition constraints. We also identify those non-empty intersection constraints which are decomposable, those which are not decomposable but polynomial, and those that are NP-hard. Many of the propagation algorithms we propose here exploit a dual viewpoint, and call upon existing global constraints for finite-domain variables like the global cardinality constraint. We are currently extending this study to counting constraints on set and multiset variables. Propagation algorithms for such constraints also appear to exploit dual viewpoints extensively.

References

1. N. Beldiceanu. Global constraints as graph properties on a structured network of elementary constraints of the same type. In *Proc. CP'00*, pages 52–66. Springer, 2000.
2. C. Bessiere, E. Hebrard, B. Hnich, and T. Walsh. The complexity of global constraints. In *Proc. AAAI'04*, 2004.
3. M. Carlsson and N. Beldiceanu. Arc-consistency for a chain of lexicographic ordering constraints. Technical report T2002-18, Swedish Institute of Computer Science, 2002. ftp://ftp.sics.se/pub/SICS-reports/Reports/SICS-T–2002-18–SE.ps.Z.
4. A. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, and T. Walsh. Global constraints for lexicographic orderings. In *Proc. CP'02*. Springer, 2002.
5. C. Gervet. Conjunto: constraint logic programming with finite set domains. *Proc. of the 1994 Int. Symp. on Logic Programming*, pages 339–358. MIT Press, 1994.
6. Ilog. User's manual. ILOG JConfigurator 2.1. 2003.
7. T. Müller and M. Müller. Finite set constraints in Oz. *13th Logic Programming Workshop*, pages 104–115, Technische Universität München, 1997.
8. J-C. Régin. A filtering algorithm for constraints of difference in CSPs. In *Proc. AAAI'94*, pages 362–367. AAAI, 1994.
9. J-C. Régin. Generalized arc consistency for global cardinality constraints. In *Proc. AAAI'96*, pages 209–215. AAAI Press/The MIT Press, 1996.
10. A. Sadler and C. Gervet. Global Filtering for the Disjointness Constraint on Fixed Cardinality Sets. Technical report IC-PARC-04-02, Imperial College London, March 2004.
11. A. Sadler and C. Gervet. Global reasoning on sets. In *Proc. of Workshop on Modelling and Problem Formulation (FORMUL'01)*, 2001. Held alongside CP-01.
12. T. Walsh. Consistency and propagation with multiset constraints: A formal viewpoint. In *Proc. CP'03*. Springer, 2003.