RAIRO Operations Research
RAIRO Oper. Res. **40** (2006) 327–353
DOI: 10.1051/ro:2007001

BOUNDS OF GRAPH PARAMETERS FOR GLOBAL CONSTRAINTS

Nicolas Beldiceanu[1], Thierry Petit[1]
and Guillaume Rochart[2]

Abstract. This article presents a basic scheme for deriving systematically a filtering algorithm from the graph properties based representation of global constraints. This scheme is based on the bounds of the graph parameters used in the description of a global constraint. The article provides bounds for the most common used graph parameters.

Keywords. Global constraint, graph constraint, filtering, bound.

Mathematics Subject Classification. 68R01.

1. Introduction

One of the main objectives of constraint programming is to provide generic tools for solving combinatorial problems, notably AC algorithms [9,14,18], general purpose combinators [15], and automata for characterizing the solutions of a constraint [4,20,30]. To deal with real-world problems, various global constraints with ad-hoc filtering algorithms, usually based on graph theory [16,17,23–26,28], have been introduced. In [8], Bessière and Van Hentenryck proposed several definitions of the notion of globality. They introduced the notion of "*semantic globality* (expressiveness), *operational globality* (quality of filtering), and *algorithmic globality* (computational efficiency of the filtering)". Beldiceanu presented in [2] a systematic description of these global constraints in terms of graph properties: among the 234 constraints of the catalog of global constraints [5], about 200 constraints are described as a conjunction of graph properties where each graph property has

Received November 16, 2006. Accepted November 20, 2006.

[1] École des Mines de Nantes, LINA FRE CNRS 2729, 44307 Nantes, France; Nicolas.Beldiceanu@emn.fr; Thierry.Petit@emn.fr

[2] Bouygues e-lab, 78061 St Quentin en Yvelines, France; Grochart@bouygues.com

the form P *op* V, where P is a graph parameter, *op* is a comparison operator in $\{\leq, \geq, =, \neq\}$, and V a domain variable[1].

Example 1. Consider the `nvalue(`$N, \{x_1, ..., x_m\}$`)` constraint [7], where $N, x_1, ..., x_m$ are domain variables. The `nvalue` constraint holds if and only if the number of distinct values assigned to the variables in $\mathcal{X} = \{x_1, ..., x_m\}$ is equal to N. It can be seen as enforcing the following graph property: The number of strongly connected components of the *intersection graph* $G(\mathcal{X}, E)$, where $E = \{x_i \in \mathcal{X}, x_j \in \mathcal{X} : x_i = x_j\}$, is equal to N.

In the context of the catalog of global constraints, Dávid Hanák made a preliminary exploitation of this description for designing filtering algorithms, for a particular graph property [13]. In this article we present a systematic approach which aims at providing generic filtering algorithms for the most used graph properties [3]: given a specification of a global constraint C in terms of graph properties, we can derive a filtering algorithm for C. Section 2 recalls the representation of global constraints as graph properties. It presents the most used graph parameters on which this article focuses. Section 3 introduces how to filter by reasoning on the bounds of the graph parameters involved in the graph-based description of a global constraint. Section 4 presents how to compute the lower and upper bounds for the graph parameters introduced in Section 2. For those bounds which are not sharp, we focus on the graph classes effectively used in the catalog of global constraints [5] in order to get sharpness for the practical cases. Section 5 recapitulates the different bounds and their respective sharpness and complexity class. Finally, Section 6 illustrates how reasoning on the bounds of graph parameters is relevant in the context of the CP(Graph) framework [10] as well as in the context of over-constrained problems [11].

2. Background

This section summarizes the representation of global constraints as graph properties introduced in [2], and illustrates this framework with the `nvalue` [7] constraint.

A global constraint C is represented as an initial digraph $G_i = (\mathcal{X}_i, E_i)$: to each vertex in $\mathcal{X}_i$ corresponds a variable involved in C, while to each arc e in E_i corresponds a binary constraint involving the variables at both extremities of e. To generate G_i from the parameters of C, the set of arcs generators described in [2] is used. When all variables of C are fixed, we remove from G_i all binary constraints which do not hold as well as isolated vertices, *i.e.*, vertices which are not extremity of any arc. This final digraph is denoted by G_f. Then C is defined by a conjunction of graph properties which should be satisfied by G_f. Each graph property has the form P *op* V; P is a graph parameter, V is a domain variable

[1] A *domain variable* is a variable that ranges over a finite set of integers; $dom(V)$, $\min(V)$ and $\max(V)$ respectively denote the set of possible values for variable V, the minimum possible value for V and the maximum possible value for V.

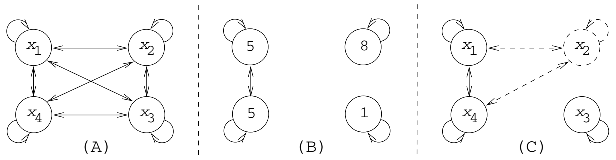

FIGURE 1. (A) Initial digraph G_i associated with the nvalue(N, $\{x_1, x_2, x_3, x_4\}$) constraint. (B) Final digraph G_f of the ground solution nvalue($3, \{5, 8, 1, 5\}$). (C) Intermediate digraph when $dom(x_1) = \{5\}$, $dom(x_3) = \{1\}$, $dom(x_4) = \{5\}$ and $dom(x_2) = \{5, 8\}$.

and *op* is one of the comparison operator $\geq, \leq, =, \neq$. Within the global constraint catalog [5], most commonly used graph parameters are:

- **NARC** and **NVERTEX**. They respectively denote the number of arcs and vertices. They are used by respectively 95 and 17 global constraints.
- **NCC** and **NSCC**. They respectively denote the number of connected and strongly connected components. They are used by respectively 19 and 13 global constraints.
- **NSINK** and **NSOURCE**. They respectively denote the number of vertices that do not have any successor and the number of vertices that do not have any predecessor. **NSINK** and **NSOURCE** are used by respectively 16 and 15 global constraints. By reversing each arc, the computation of **NSINK** and **NSOURCE** is the same. Therefore, the rest of this article considers only **NSINK**.

Example 2. Consider the nvalue(N,$\mathcal{X}$) constraint presented in the introduction. Parts (A) and (B) of Figure 1 respectively show the initial digraph G_i generated for the nvalue constraint with $\mathcal{X} = \{x_1, x_2, x_3, x_4\}$ and the digraph G_f associated with the ground solution nvalue($3, \{5, 8, 1, 5\}$). Each vertex of G_i depicts its corresponding variable. All arcs corresponding to equality constraints that are not satisfied are removed to obtain G_f from G_i. Each vertex of G_f depicts the value assigned to its corresponding variable. The nvalue constraint is defined by the graph property **NSCC** $= N$. The nvalue($3, \{5, 8, 1, 5\}$) constraint holds since G_f contains three strongly connected components, which can be interpreted as the fact that N is equal to the number of distinct values taken by the variables x_1, x_2, x_3 and x_4. Part (C) of Figure 1 will be referenced in Example 3.

3. Filtering from graph properties

Given a graph property P *op* V occurring in the description of a global constraint, this section first shows how to reduce the domain of V in order to enforce P *op* V. Finally, it discusses the case where several graph properties are used to define a global constraint.

3.1. Handling one single graph property

We first introduce the notion of intermediate digraph derived from the initial digraph G_i, where vertices and arcs can have different status as detailed below. The purpose of this intermediate digraph is to reflect the knowledge we currently have about the vertices and the arcs of G_i that may or may not belong to the final digraph G_f. The lower and upper bounds of a graph property will be computed from this intermediate digraph. This knowledge comes from two sources:

- Because of the current domain of its variables, a binary constraint associated with an arc of G_i does not hold (or is entailed).
- Because of an external reason (*e.g.*, if a global constraint is defined by two graph properties on the same initial digraph G_i then an arc may be forced to belong to G_f by one of the two properties), a given arc or vertex of G_i is forced to belong to G_f (or is forced not to belong to G_f).

When a global constraint C is posted, the intermediate digraph corresponds to the initial digraph G_i, while, when all variables of C are fixed, the intermediate digraph is equal to the final digraph G_f. At any stage all the possible final digraphs are induced digraphs of the intermediate digraph.

Notation 1. *Let $G_i = (X_i, E_i)$ be the initial digraph of a global constraint C, and $G_f = (X_f, E_f)$ its final digraph. At a given step corresponding to a partial assignment of values to the variables of C, we classify a vertex $v_j \in X_i$ and an arc $e_k \in E_i$ as follows:*

- *v_j is a T-vertex (i.e., a* true*-vertex) if and only if $v_j \in X_f$; v_j is a F-vertex (i.e., a* false*-vertex) if and only if $v_j \notin X_f$; otherwise v_j is a U-vertex (i.e., an* undetermined*-vertex). X_T, X_F and X_U respectively denote the sets of T-vertices, of F-vertices and of U-vertices.*
- *e_k is a T-arc (i.e., a* true*-arc) if and only if $e_k \in E_f$; e_k is a F-arc (i.e., a* false*-arc) if and only if $e_k \notin E_f$; otherwise e_k is an U-arc (i.e., an* undetermined*-arc). E_T, E_F and E_U respectively denote the sets of T-arcs, of F-arcs and of U-arcs.*

For two distinct elements Q and R in $\{T, U, F\}$, let X_{QR} denote the vertex subset $X_Q \dot\cup X_R$, and E_{QR} denote the arc subset $E_Q \dot\cup E_R$.

The following definition of the intermediate digraph takes into account the fact that the final digraph G_f will not contain any isolated vertex (*i.e.*, a vertex not involved in any arc) [5, page 45].

Definition 1. The *intermediate digraph* is the digraph defined from the initial digraph G_i, from the sets of vertices X_T, X_F, X_U and from the sets of arcs E_T, E_F, E_U by applying the next rules while they induce some modifications:

- Remove all F-arcs from G_i.
- Any F-vertex which is not the extremity of at least one T-arc is removed from G_i[2]. When a vertex is removed, we remove also all its ingoing and outgoing arcs that are turned into F-arcs.

[2] A F-vertex that is linked to a T-arc is not removed so that Property 1 catches a contradiction.

- Any U-vertex, which is not the extremity of at least one arc, is turned into a F-vertex.
- Any U-vertex, which is an extremity of a T-arc, is turned into a T-vertex.
- If a T-vertex with no loop has a unique neighbor, which is a U-vertex, then this U-vertex is turned into a T-vertex.
- If a T-vertex is the extremity of exactly one U-arc e and not the extremity of any T-arc, then e is turned into a T-arc.

When a vertex or an arc is removed from G_i, or when the status of a vertex or of an arc is changed by one of the previous rule, the sets of vertices X_T, X_F, X_U and the sets of arcs E_T, E_F, E_U are updated to reflect this change.

Example 3. Consider again the `nvalue(`N`,`$\mathcal{X}$`)` constraint presented in the introduction, and assume that not all variables of $\mathcal{X} = \{x_1, x_2, x_3, x_4\}$ are fixed: $dom(x_1) = \{5\}$, $dom(x_2) = \{5, 8\}$, $dom(x_3) = \{1\}$, $dom(x_4) = \{5\}$. Furthermore, without loss of generality, assume that, for an equality constraint ec associated with an arc of the initial digraph G_i of `nvalue`, entailment is only detected when the two variables occurring in ec are fixed[3]. This leads to partition the arcs of G_i in the following three sets $E_T = \{(x_1, x_1),(x_1, x_4),(x_3, x_3),(x_4, x_1),(x_4, x_4)\}$, $E_U = \{(x_1, x_2),(x_2, x_1),(x_2, x_2),(x_2, x_4),(x_4, x_2)\}$ and $E_F = \{(x_1, x_3),(x_2, x_3),(x_3, x_1),(x_3, x_2), (x_3, x_4),(x_4, x_3)\}$. The status of the vertices is initially set *undetermined* (*i.e.*, $X_U = \{x_1, x_2, x_3, x_4\}$) and we apply the rules of Definition 1 in order to obtain the intermediate digraph depicted by Part (C) of Figure 1. A solid line depicts a T-vertex or a T-arc, while a dashed line indicates a U-vertex or an U-arc. The same style will be used in all other figures of this article in order to depict T-vertices, T-arcs, U-vertices and U-arcs.

Property 1. *From the definition of G_f, a global constraint C has no solution if its intermediate digraph contains a F-vertex or if it contains a T-vertex that is not the extremity of any arc.*

Given a graph property $P\ op\ V$ associated with a global constraint C, the intermediate digraph will be used for evaluating a lower bound $\underline{P}$ and an upper bound $\overline{P}$ of the graph parameter P. Section 4 provides graph invariants for evaluating $\underline{P}$ and $\overline{P}$ for different graph parameters. It assumes that all U-vertices or arcs of the intermediate digraph can be freely turned into T-vertices or T-arcs (resp. F-vertices or F-arcs). According to the comparison operator op, Table 1 gives the four possible cases for reducing the domain of variable V according to $\underline{P}$ and $\overline{P}$.

3.2. Handling several graph properties

Quite often, it happens that one wants to enforce the final digraph G_f of a global constraint C to verify more than one graph property. In this context, these graph properties involve several graph parameters that cannot vary independently. So, in order to get stronger necessary conditions for the feasibility of C we can search for graph invariants that link the different graph parameters. These graph

[3] As a consequence, since x_2 is not yet fixed, the arc (x_2, x_2) is an U-arc.

Table 1. The four possible cases for reducing the domain of a variable V according to $\underline{P}$ and $\overline{P}$.

$P \leq V$	$\min(V) \geq \max(\underline{P}, \min(V))$
$P \geq V$	$\max(V) \leq \min(\overline{P}, \max(V))$
$P = V$	$\min(V) \geq \max(\underline{P}, \min(V)) \wedge \max(V) \leq \min(\overline{P}, \max(V))$
$P \neq V$	$\underline{P} = \overline{P} \Rightarrow \underline{P} \notin \mathrm{dom}(V)$

invariants are typically inequalities between two arithmetic expressions mentioning several graph parameters[4]. In this context each graph invariant will be used in order to adjust the minimum and the maximum value of its graph parameters. An initial lower and upper bound of each graph parameter will be computed by using the inequalities of Section 4. Finally we make a last observation: quite often, it happens that the final digraph G_f associated with a global constraint has a regular structure that comes from its initial digraph or from a property of the arc constraint. In this context we can have tighter graph invariants which hold for specific graph classes.

Example 4. Consider again the `nvalue` constraint. Bessière *et al.* [7] give a necessary condition based on a result by Turán [27]. For the final digraph G_f of the `nvalue` constraint, which is symmetric, reflexive and transitive[5], this necessary condition[6] links the *number of strongly connected components*, the *number of vertices* and the *number of arcs* of G_f: $\mathbf{NSCC} \geq \left\lceil \frac{\mathbf{NVERTEX}^2}{\mathbf{NARC}} \right\rceil$. This allows us to evaluate the minimum number of distinct values according to the number of variables of the `nvalue` constraint (*i.e.*, **NVERTEX**, which is fixed to the number of variables of the `nvalue` constraint since each vertex of the initial digraph belongs to the final digraph) and to the maximum number of arcs of its intermediate digraph (*i.e.*, $\overline{\mathbf{NARC}}$).

4. Bounds of graph parameters

This section is devoted to the evaluation of lower and upper bounds of the graph parameters introduced in Section 2. For this purpose, we will deal with graphs derived from the intermediate digraph with different sets of arcs and vertices that are described by the following notations.

[4] See Chapter 3 of [5] for a collection of about 200 graph invariants.

[5] G_f consists of one or several cliques.

[6] We recast the original condition given in [7] to the context of the intermediate digraph of the `nvalue` constraint.

Notation 2. *For any words Q, R and S on the alphabet $\{T, U, F\}$, X_Q and X_S are vertex subsets and E_R is an arc subset of the initial digraph G_i (as stated in the last part of notation 1), and:*

- *$X_{Q,R}$ (resp. $X_{Q,\neg R}$) denotes the set of vertices in X_Q which are incident with at least one arc (resp. with no arc) in E_R:* $X_{Q,R} = \{x \in X_Q \mid \exists e \in E_R \wedge x \in e\}$ *and* $X_Q = X_{Q,R} \dot\cup X_{Q,\neg R}$.
- *$X_{Q,R,S}$ (resp. $X_{Q,R,\neg S}$) denotes the set of vertices in $X_{Q,R}$ that are neighbors to at least one vertex (resp. to no vertex) in X_S by an arc in E_R:* $X_{Q,R,S} = \{x \in X_{Q,R} \mid \exists y \in X_S \wedge ((x, y) \in E_R \vee (y, x) \in E_R)\}$ *and* $X_{Q,R} = X_{Q,R,S} \dot\cup X_{Q,R,\neg S}$.
- *$X_{Q,\neg R,S}$ (resp. $X_{Q,\neg R,\neg S}$) denotes the set of vertices in $X_{Q,\neg R}$ that are neighbors to at least one vertex (resp. to no vertex) in X_S by an arc in E_{TU}:* $X_{Q,\neg R,S} = \{x \in X_{Q,\neg R} \mid \exists y \in X_S \wedge ((x, y) \in E_{TU} \vee (y, x) \in E_{TU})\}$ *and* $X_{Q,\neg R} = X_{Q,\neg R,S} \dot\cup X_{Q,\neg R,\neg S}$.
- *$E_{R,Q}$ is the set of arcs in E_R which are incident to at least one vertex in X_Q:* $E_{R,Q} = \{e \in E_R \mid \exists x \in X_Q \wedge x \in e\}$.
- *$E_{R,Q,S}$ is the set of arcs in E_R which are incident to one vertex in X_Q and to one vertex in X_S:* $E_{R,Q,S} = \{e \in E_R \mid \exists x \in X_Q \wedge \exists y \in X_S \wedge (e = (x, y) \vee e = (y, x))\}$.

Based on the previous notations, we define two kinds of graphs, where $\mathcal{X}$ is a set of vertices and $\mathcal{E}$ a set of arcs:

- $\overrightarrow{G}(\mathcal{X}, \mathcal{E})$ denotes the digraph defined by the vertex set $\mathcal{X}$ and the subset of arcs of $\mathcal{E}$ having their two extremities in $\mathcal{X}$.
- $\overleftrightarrow{G}(\mathcal{X}, \mathcal{E})$ denotes the undirected graph derived from $\overrightarrow{G}(\mathcal{X}, \mathcal{E})$ by keeping all vertices of $\mathcal{X}$ and by considering one edge (u, v) (where u and v are not necessarily distinct) when at least one arc in $\{(u, v), (v, u)\}$ belongs to $\overrightarrow{G}(\mathcal{X}, \mathcal{E})$.

Within the last notation $\overleftrightarrow{G}(\mathcal{X}, \mathcal{E})$, note that $\mathcal{E}$ still refers to the set of arcs of the corresponding digraph $\overrightarrow{G}(\mathcal{X}, \mathcal{E})$ (and not to the derived sets of edges). This notation is used either when any two arcs (u, v) and (v, u) in $\mathcal{E}$ have the same status, or when we do not care about the status of the arcs. Note also that $\overrightarrow{G}(X_{TU}, E_{TU})$ is the intermediate digraph introduced by Definition 1.

As we will see later, computing lower and upper bounds of graph parameters can essentially be seen as computing some graph parameters on the graphs previously introduced. For this reason we define the following notations for a digraph G.

Notation 3. *$cc(G)$, $scc(G)$, $sink(G)$ and $source(G)$ respectively denote the set of connected components, strongly connected components, sinks and sources of G. Similarly, in order to restrict to specific induced subgraphs where a given condition cond is required on each subgraph we use the following notation: $cc_{[cond]}(G)$, $scc_{[cond]}(G)$, $sink_{[cond]}(G)$ and $source_{[cond]}(G)$.*

Example 5. For instance, the set of connected components of $\overrightarrow{G}(X_{TU}, E_{TU})$ containing at least one T-arc is denoted by $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$.

Some bounds will be expressed in terms of graph parameters that correspond to non-polynomial problems. However in such a case we will provide bounds that are sharp. Note that many of the digraphs, which express a global constraint, belong to specific graph classes for which a non-polynomial problem becomes polynomial. Even when the computation is polynomial, we can get better worst-case complexity by exploiting the structure of the intermediate digraph. For instance, in the context of the `group` constraint [3], we have to estimate the minimum number of connected components of the final digraph G_f. For this purpose, Property 10 in Section 4 requires to compute the number of connected components on the intermediate digraph. This digraph is a subgraph of an elementary path augmented with loops [5], p. 516. Therefore the complexity for computing the number of connected components is linear in the number of vertices.

Example 6. We illustrate some sets of vertices and arcs previously introduced and some graphs on the intermediate digraph depicted by Part (C) of Figure 1:

- $X_T = \{x_1, x_3, x_4\}$ denotes the set of T-vertices.
- $X_U = \{x_2\}$ denotes the set of U-vertices.
- $E_T = \{(x_1, x_1), (x_1, x_4), (x_3, x_3), (x_4, x_1), (x_4, x_4)\}$ denotes the set of T-arcs.
- $E_U = \{(x_1, x_2), (x_2, x_1), (x_2, x_2), (x_2, x_4), (x_4, x_2\}$ denotes the set of U-arcs.
- $X_{U,T} = \emptyset$ denotes the set of U-vertices that are the extremity of at least one T-arc. Observe that, from Definition 1, this set is always empty.
- $X_{U,\neg T} = \{x_2\}$ denotes the set of U-vertices that are not the extremity of any T-arc. Observe that, from Definition 1, this set is always equal to X_U.
- $E_{TU} = \{(x_1, x_1), (x_1, x_2), (x_1, x_4), (x_2, x_1), (x_2, x_2), (x_2, x_4), (x_3, x_3), (x_4, x_1), (x_4, x_2), (x_4, x_4)\}$ denotes the set of arcs that are U-arcs or T-arcs.
- $E_{U,T} = \{(x_1, x_2), (x_2, x_1), (x_2, x_4), (x_4, x_2)\}$ denotes the set of U-arcs such that at least one of their extremities is a T-vertex.
- $\overrightarrow{G}(X_T, E_T) = \overrightarrow{G}(\{x_1, x_3, x_4\}, \{(x_1, x_1), (x_1, x_4), (x_3, x_3), (x_4, x_1), (x_4, x_4)\})$ denotes the digraph defined by the set of vertices X_T and the arcs of E_T that have their two extremities in X_T.

We now go through the different graph parameters and provide for each of them a lower and an upper bound. For each graph parameter, we give typical examples of global constraints for which the graph-based description presented in [5] uses that parameter.

4.1. Bounds for the number of arcs

Within [5], $\underline{\mathbf{NARC}}$ and $\overline{\mathbf{NARC}}$ are respectively referenced in the description of about 50 constraints (see, *e.g.*, the `disjoint` (p. 436), the `element` (p. 460) and the `inverse` (p. 568) constraints) and of 80 constraints (see, *e.g.*, the `change` (p. 284) and the `element_sparse` (p. 486) constraints).

4.1.1. *Estimating* $\underline{\mathbf{NARC}}$

Before presenting the idea behind the lower bound on the number of arcs we first need to introduce the notions of *matching* and of *edge cover.*

Definition 2. Given an undirected graph G, a *matching* of G is a set of edges of G, excluding loops, such that no two edges have a vertex in common. A *maximum matching* is a matching of maximum cardinality. $\mu(G)$ depicts the *cardinality of a maximum matching* of G.

Definition 3. An *edge cover* of an undirected graph G is a subset $\mathcal{S}$ of edges of G such that, for each vertex v of G, there is at least one edge e of $\mathcal{S}$ such that v is an extremity of e. A *minimum edge cover* is an edge cover of minimum cardinality.

The minimum number of arcs in the final digraph G_f will be at least equal to the current number of T-arcs $|E_T|$. Moreover, in $\overrightarrow{G}(X_T, E_T)$ some isolated T-vertices may remain. Since they will belong to G_f, each of them will be the extremity of at least one arc in G_f. Therefore the lower bound is equal to $|E_T|$ plus the minimum number of U-arcs of the intermediate digraph $\overrightarrow{G}(X_{TU}, E_{TU})$ required to cover all T-vertices that are not extremities of an arc in E_T. This second quantity is computed polynomially on an undirected graph derived from $\overleftrightarrow{G}(X_{TU}, E_U)$[7]. The principle is first to identify the number of isolated T-vertices in $\overrightarrow{G}(X_T, E_T)$ which can be joined one another by turning U-arcs into T-arcs, since this allows saving one arc in the estimation, for some of them. For all other isolated T-vertices in $\overrightarrow{G}(X_T, E_T)$ an extra arc will be required.

Property 2. $\underline{\mathbf{NARC}} \geq |E_T| + |X_{T,\neg T}| - \mu(\overleftrightarrow{G}(X_{T,\neg T}, E_U))$.

Proof. By definition $\forall e \in E_T : e \in E_f$ (*i.e.*, T-arcs belong to G_f). Thus $\underline{\mathbf{NARC}} \geq |E_T|$. Moreover in $\overleftrightarrow{G}(X_T, E_T)$ some vertices may have a null degree. Since they will necessarily belong to G_f (since G_f does not contain isolated vertices), additional arcs will be required for connecting them. We consider the undirected graph $\overleftrightarrow{G}(X_{T,\neg T}, E_U)$, build with T-vertices of null degree in $\overleftrightarrow{G}(X_T, E_T)$ as well as the U-arcs between these vertices. $X_{T,\neg T}$ can be partitioned into the two sets $X_{T,\neg T,\neg T}$ and $X_{T,\neg T,T}$. Vertices of the first set cannot be connected one another, therefore one new arc will be necessary for each of them: $\underline{\mathbf{NARC}} \geq |E_T| + |X_{T,\neg T,\neg T}|$. Vertices of the second set might be connected each other. We will need at least a number of arcs equal to the cardinality of a minimum edge cover in $\overleftrightarrow{G}(X_{T,\neg T,T}, E_U)$. From [19] we know that this quantity is $|X_{T,\neg T,T}| - \mu(\overleftrightarrow{G}(X_{T,\neg T,T}, E_U))$. Since we consider a partition of vertices and since by definition all edges in $\overleftrightarrow{G}(X_{T,\neg T,T}, E_U)$ involve only vertices of $X_{T,\neg T,T}$, from $\underline{\mathbf{NARC}} \geq |E_T| + |X_{T,\neg T,\neg T}|$ we have $\underline{\mathbf{NARC}} \geq |E_T| + |X_{T,\neg T,\neg T}| + |X_{T,\neg T,T}| - \mu(\overleftrightarrow{G}(X_{T,\neg T,T}, E_U))$. Observe that $|X_{T,\neg T,\neg T}| + |X_{T,\neg T,T}| = |X_{T,\neg T}|$. Moreover by definition $\overleftrightarrow{G}(X_{T,\neg T,\neg T}, E_U)$ has

[7] Since in $\overrightarrow{G}(X_{TU}, E_U)$ all arcs have the same status (*i.e.*, U-arcs), we can safely consider the undirected graph $\overleftrightarrow{G}(X_{TU}, E_U)$.

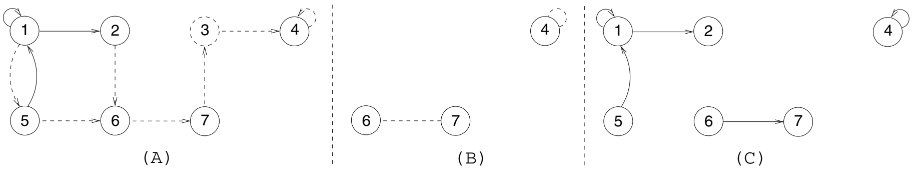

Figure 2. (A) Intermediate digraph and (B) undirected graph used to compute the cardinality of a maximum matching for estimating $\underline{\mathbf{NARC}}$. (C) Example of a final digraph achieving the lower bound.

no edges (All vertices of $\overleftrightarrow{G}(X_{T,\neg T,\neg T}, E_U)$ belong to $X_{T,\neg T,\neg T}$, which is contradictory with the fact of having an arc joining two of them in $\overrightarrow{G}(X_{T,\neg T,\neg T}, E_U)$.). Therefore $\mu(\overleftrightarrow{G}(X_{T,\neg T,T}, E_U)) = \mu(\overleftrightarrow{G}(X_{T,\neg T}, E_U))$. The property holds. □

Example 7. Figure 2 illustrates Property 2. It shows how to compute $\underline{\mathbf{NARC}}$ according to the intermediate digraph $\overrightarrow{G}(X_{TU}, E_{TU})$ given by Part (A). Part (B) shows the corresponding undirected graph $\overleftrightarrow{G}(X_{T,\neg T}, E_U)$ used for computing the cardinality of a maximum matching. We have $E_T = \{(1,1),(1,2),(5,1)\}$, $E_U = \{(1,5),(2,6),(3,4),(4,4),(5,6),(6,7),(7,3)\}$, $X_{T,\neg T} = \{4,6,7\}$, $|E_T| = 3$, $|X_{T,\neg T}| = 3$, $\mu(\overleftrightarrow{G}(X_{T,\neg T}, E_U)) = 1$, thus $\underline{\mathbf{NARC}} \geq 3 + 3 - 1 = 5$. Part (C) provides a final digraph where this lower bound of 5 arcs is reached.

Property 3. *The lower bound provided by Property 2 is sharp.*

Proof. We can build a final digraph by adding to the current digraph $\overrightarrow{G}(X_T, E_T)$:

- For each $x \in X_{T,\neg T,\neg T}$ any U-arc in $\overleftrightarrow{G}(X_T, E_U)$ such that x is an extremity.
- For each edge of a minimum edge cover of $\overleftrightarrow{G}(X_{T,\neg T,T}, E_U)$ one arc in $\overrightarrow{G}(X_{T,\neg T,T}, E_U)$ having the same extremities than that edge.

By definition all T-vertices will be covered, and the total number of arcs will be equal to $|E_T| + |X_{T,\neg T}| - \mu(\overleftrightarrow{G}(X_{T,\neg T}, E_U))$. □

4.1.2. *Estimating* $\overline{\mathbf{NARC}}$

Property 4. $\overline{\mathbf{NARC}} \leq |E_{TU}|$.

Proof. By definition of $\overrightarrow{G}(X_{TU}, E_{TU})$, any arc of the initial digraph that does not belong to E_{TU} will also not belong to the final digraph G_f. Thus $\overline{\mathbf{NARC}} \leq |E_{TU}|$. □

Property 5. *The upper bound provided by Property 4 is sharp.*

Proof. If all arcs of $\overrightarrow{G}(X_{TU}, E_{TU})$ belong to G_f then $\mathbf{NARC} = |E_{TU}|$. □

4.2. Bounds for the Number of Vertices

Within [5], **NVERTEX** is only used in graph properties of the form **NVERTEX** = *variable*, where *variable* is a domain variable, which means that we need to both evaluate $\underline{\mathbf{NVERTEX}}$ as well as $\overline{\mathbf{NVERTEX}}$. **NVERTEX** is referenced in about 15 constraints such as, for instance, `cutset` (p. 382) or `group` (p. 516).

4.2.1. *Estimating* $\underline{\mathbf{NVERTEX}}$

Before presenting the idea of the lower bound on the number of vertices we first need to introduce the notions of *bipartite graph* and of *hitting set*.

Definition 4. An undirected graph $G(X, E)$ is *bipartite* if there is a partition (Y, Z) of X such that $\forall(u, v) \in E$, either $u \in Y$ and $v \in Z$, or $u \in Z$ and $v \in Y$. Such a graph will be denoted by $G((Y, Z), E)$.

Definition 5. Given a bipartite graph $G((Y, Z), E)$, a *hitting set* [12] of $G((Y, Z), E)$ is a subset Z' of Z such that for any vertex $y \in Y$ there exists an edge in E connecting y to a vertex in Z'. $h(G)$ denotes the *cardinality of a minimum hitting set* of G[8].

When estimating $\underline{\mathbf{NVERTEX}}$, we know that all T-vertices will belong to G_f. Since G_f does not contain isolated vertices and since $\overrightarrow{G}(X_T, E_T)$ may contain isolated vertices, some additional U-vertices may have to be turned to T-vertices, in order to connect those vertices. All isolated vertices in $\overrightarrow{G}(X_T, E_T)$ that are the extremity of an U-arc such that the other extremity is a T-vertex may be connected to G_f by that arc. Therefore we ignore these vertices. This case includes the case of U-loops where the extremity is a T-vertex. From the remaining isolated vertices, the number of required additional vertices is equal to the cardinality of a minimum hitting set in an undirected graph derived from $\overleftrightarrow{G}(X_{TU}, E_U)$.

Property 6. $\underline{\mathbf{NVERTEX}} \geq |X_T| + h(\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T}))$.

Proof. In the previous formula, $X_{T,\neg T,\neg T}$ represents the set of T-vertices that are not the extremity of any T-arcs and that have not a T-vertex as neighbor, while $X_{U,\neg T,T}$ is the set of U-vertices that have at least one T-vertex as neighbor. Finally $E_{U,T}$ is the set of U-arcs such that at least one of their extremities is a T-vertex.

All T-vertices in X_T will belong to G_f. Starting from the point that G_f cannot contain isolated vertices, some vertices in X_T having a null degree in $\overleftrightarrow{G}(X_T, E_T)$ may require to turn some U-vertices into T-vertices. By definition this is not the case for vertices in $X_{T,T}$. This is also not the case for vertices which are not covered by a T-arc but which are the extremity of an arc in E_U where the other extremity is a T-vertex (*i.e.*, the set $X_{T,\neg T,T}$). $(X_{T,T}, X_{T,\neg T,T}, X_{T,\neg T,\neg T})$ is a partition of X_T: let us consider now the case of vertices in $X_{T,\neg T,\neg T}$. They are not directly

[8] From [12], computing the exact value of the cardinality of a minimum hitting set is NP-Hard in the general case.

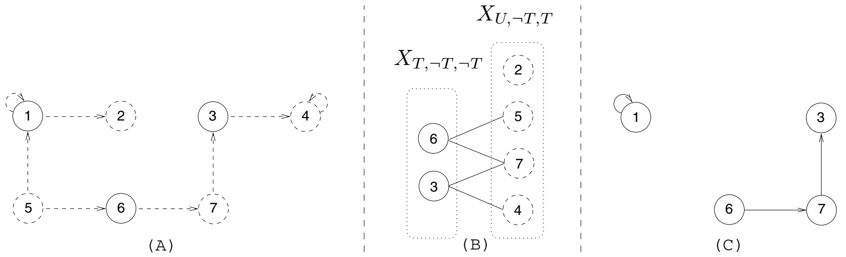

Figure 3. (A) Intermediate digraph and (B) undirected graph used to compute the cardinality of a minimum hitting set for estimating $\underline{\textbf{NVERTEX}}$. (C) Example of a final digraph achieving the lower bound.

connected one another neither by a T-arc nor by an U-arc. Therefore if this set is not empty then some U-vertices will be forced to be turned to T-vertices. Those U-vertices must be an extremity of at least one edge containing a T-vertex, so we consider $X_{U,\neg T,T}$, and the arcs joining them to T-vertices. By definition the minimum number of vertices in $X_{U,\neg T,T}$ required to cover all vertices in $X_{T,\neg T,\neg T}$ is $h(\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T}))$. The property holds. □

Example 8. Figure 3 illustrates Property 6. It shows how to compute the lower bound of **NVERTEX** according to the intermediate digraph $\overrightarrow{G}(X_{TU}, E_{TU})$ depicted by Part (A) of Figure 3. Part (B) illustrates the corresponding bipartite graph $\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T})$ used for computing the cardinality of the minimum hitting set. We have $X_T = \{1,3,6\}$, $E_{U,T} = \{(1,1),(1,2),(3,4),(5,1),(5,6),(6,7),(7,3)\}$, $X_{T,\neg T,\neg T} = \{3,6\}$, $X_{U,\neg T,T} = \{2,4,5,7\}$, $|X_T| = 3$ and $h(\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T})) = 1$ (*i.e.*, 7 allows to cover both 6 and 3), thus $\underline{\textbf{NVERTEX}} \geq 3+1 = 4$. Part (C) provides a final digraph where this lower bound of 4 vertices is reached.

Property 7. *The lower bound provided by Property 6 is sharp.*

Proof. Following the schema of the proof, it is possible to build a final digraph from $\overleftrightarrow{G}(X_T, E_T)$ by turning to T-vertices $h(\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T}))$ U-vertices. □

4.2.2. *Estimating* $\overline{\textbf{NVERTEX}}$

Property 8. $\overline{\textbf{NVERTEX}} \leq |X_{TU}|$.

Proof. By definition of $\overrightarrow{G}(X_{TU}, E_{TU})$, any vertex of the initial digraph that does not belong to X_{TU} will also not belong to the final digraph G_f. Thus $\overline{\textbf{NVERTEX}} \leq |X_{TU}|$. □

Property 9. *The upper bound provided by Property 8 is sharp.*

Proof. This upper bound is reached when all vertices of $\overrightarrow{G}(X_{TU}, E_{TU})$ belong to the final digraph G_f. □

4.3. Bounds for the number of connected components

Within [5], **NCC** is most of the time used in graph properties of the form **NCC** = *variable*, where *variable* is a domain variable, which means that we need to both evaluate $\underline{\mathbf{NCC}}$ as well as $\overline{\mathbf{NCC}}$. **NCC** is referenced in about 18 constraints such as, for instance, `cycle` (p. 386), `group` (p. 516) or `tree` (p. 902).

4.3.1. *Estimating* $\underline{\mathbf{NCC}}$

Property 10. $\underline{\mathbf{NCC}} \geq |cc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))|$.

Proof. First observe that from Property 1, $\overrightarrow{G}(X_{TU}, E_{TU})$ cannot have a connected component consisting of one single isolated T-vertex. Second, note that all arcs and vertices of a connected component of $\overrightarrow{G}(X_{TU}, E_{TU})$, where there is no T-vertices, may finally not belong to G_f. We can ignore them. To obtain in G_f less connected components than the ones of $\overrightarrow{G}(X_{TU}, E_{TU})$ containing T-vertices, it is necessary to have at least one new arc that joins two of them. This is not possible because $\overrightarrow{G}(X_{TU}, E_{TU})$ contains all the arcs that will potentially belong to G_f. The property holds. □

Property 11. *The lower bound provided by Property 10 is sharp.*

Proof. Build a final digraph with all arcs and vertices from E_{TU} that belong to a connected component of the set $cc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$. All the other U-arcs from E_{TU} are removed. The number of connected components will be equal to $|cc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))|$. □

4.3.2. *Estimating* $\overline{\mathbf{NCC}}$

T-vertices and T-arcs of $\overrightarrow{G}(X_{TU}, E_{TU})$ will necessarily belong to G_f. If we except T-vertices with no predecessor and no successor in $\overrightarrow{G}(X_T, E_T)$ we are able to count in $\overrightarrow{G}(X_{TU}, E_{TU})$ a certain number of connected components with only T-vertices and T-arcs. By definition, augmenting any of these connected components by new arcs cannot increase their number. Therefore, the intuitive idea for computing an upper bound of **NCC** is first to count these connected components, to remove them from $\overrightarrow{G}(X_{TU}, E_{TU})$, to remove also all the arcs connected to them, and then to estimate the maximum number of connected components that may exist in the remaining digraph.

We have to take into account that some T-vertices with no predecessors and no successors in $\overrightarrow{G}(X_T, E_T)$ may exist. These ones will belong to G_f. If they are finally connected to one of the connected components of $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$ (*i.e.*, the set of connected components of $\overrightarrow{G}(X_T, E_T)$ containing at least one T-arc), then they will not increase the number of connected components. Therefore, we can ignore T-vertices with no predecessors and no successors *exclusively* connected to the connected components previously counted[9] in $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$. Similarly we have to ignore U-vertices which are also exclusively connected to the connected components in $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$.

Definition 6. $\overrightarrow{G}_{rem} = (X_{rem}, E_{rem})$ is the digraph obtained from $\overrightarrow{G}(X_{TU}, E_{TU})$ by first removing all vertices present in the set of connected components $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T,\ E_T))$ and then removing all vertices becoming isolated in the remaining digraph.

Property 12. *From Definition 6 we have:*

- *No vertex is common to X_{rem} and to a connected component of $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T,\ E_T))$.*
- *All T-arcs belong to connected components in $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$.*
- *Any vertex, which is neither present in X_{rem} nor in $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$, cannot be an extremity of an arc involving a vertex of X_{rem}.*

Proof. By construction of $\overrightarrow{G}_{rem}$. □

From Property 12 we can safely add $|cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))|$ to an estimation of the maximum possible number of connected components in any final digraph derived from $\overrightarrow{G}_{rem}$. This will give us the estimation $\overline{\mathbf{NCC}}$ we are looking for. Let us now go into details.

Notation 4. $\overline{\mathbf{NCC}}_{rem}$ *denotes the maximum possible number of connected components in any final digraph derived from* $\overrightarrow{G}_{rem}$.

Notation 5. $\overleftrightarrow{G}_{rem} = (X_{rem}, E_{rem})$ *is the undirected graph obtained from the undirected graph* $\overleftrightarrow{G}(X_{TU}, E_U)$ *by removing all vertices present in* $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T,$ $E_T))$ *and then all vertices becoming isolated in the remaining undirected graph.*

Definition 7. Given an undirected graph G, which eventually contains loops, a *l-matching* of G is a set of edges such that no two edges have a vertex in common. A *maximum l-matching* is a l-matching of maximum cardinality[10]. $\mu_l(G)$ denotes the *cardinality of a maximum l-matching* of G.

Observe that a l-matching can contain loops.

Property 13. $\overline{\mathbf{NCC}}_{rem} = \mu_l(\overleftrightarrow{G}_{rem})$.

[9] We do not loose the sharp aspect of the evaluation because if such a vertex is connected by only one arc to a connected component then this does not decrease the number of connected components. It is possible to build a final digraph where all such vertices satisfy this property (that is, connect them only by one arc to a connected component of $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$).

[10] Existing polynomial algorithms for computing maximum matchings can be adapted to this case: each loop can be unfolded by creating a new vertex and then those algorithms apply.

Proof. For evaluating the maximum possible cardinality of a set of connected components in any final digraph derived from $\overrightarrow{G}_{rem}$, we evaluate the maximum number ns of subgraphs containing at least one edge among all sets of subgraphs built from $\overleftrightarrow{G}_{rem}$ as follows (we have to consider subgraphs with at least one edge because in the final digraph G_f any vertex will be the extremity of at least one arc). Let S be such a set of subgraphs. To each subgraph in S corresponds a subset of vertices and a subset of edges of $\overleftrightarrow{G}_{rem}$ forming a connected component, such that no two subgraphs in S have a vertex in common.

Among the sets S where each subgraph is formed either by a single edge or by an isolated U-vertex, ns is by construction equal to the cardinality of a maximum l-matching. Therefore $\overline{\mathbf{NCC}}_{rem} \geq \mu_l(\overleftrightarrow{G}_{rem})$.

Let us now prove by induction on the number of edges of $\overleftrightarrow{G}_{rem}$ that $\overline{\mathbf{NCC}}_{rem} \leq \mu_l(\overleftrightarrow{G}_{rem})$. For a graph $\overleftrightarrow{G}_{rem}$ reduced to one single edge this is true. Assume that this is true for a graph of m edges. We show that this still holds for a graph $\overleftrightarrow{G}_{rem}$ with $m+1$ edges. Let S be a set of subgraphs of $\overleftrightarrow{G}_{rem}$ that maximizes the number of subgraphs containing at least on edge. Two possible cases may occur for S:

- Any subgraph in S contains at most one edge. In this case it forms a maximum l-matching and $ns = \mu_l(\overleftrightarrow{G}_{rem})$.
- There is a subgraph $s \in S$ containing $k > 1$ edges.
  - Either there exists one edge in s whose removal creates two connected components, each of them involving at least one edge. This contradicts the fact that S maximizes the number of subgraphs containing at least one edge.
  - Or such an edge does not exists. In this case removing any edge of s has no impact on ns and by the induction hypothesis, ns is equal to the cardinality of a maximum l-matching on the graph $\overleftrightarrow{G}'_{rem}$ obtained from $\overleftrightarrow{G}_{rem}$ by removing that edge.

  By definition of a maximum l-matching, any maximum l-matching of $\overleftrightarrow{G}_{rem}$ have a cardinality greater than or equal to $\mu_l(\overleftrightarrow{G}'_{rem})$ when $\mu_l(\overleftrightarrow{G}'_{rem})$ is obtained from $\overleftrightarrow{G}_{rem}$ by removing one edge. $\overline{\mathbf{NCC}}_{rem} \leq \mu_l(\overleftrightarrow{G}_{rem})$.

The property holds. □

Property 14. $\overline{\mathbf{NCC}} \leq |cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))| + \mu_l(\overleftrightarrow{G}_{rem})$.

Proof. From Property 12, any arc has at least either a vertex in $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$, or a vertex in $\overrightarrow{G}_{rem}$. □

Example 9. Figure 4 illustrates Property 14. It shows how to compute the upper bound of $\mathbf{NCC}$ according to the intermediate digraph $\overrightarrow{G}(X_{TU}, E_{TU})$ depicted by Part (A) of Figure 4. In Part (A), C_1 and C_2 corresponds to the connected components of $\overrightarrow{G}(X_T, E_T)$ that have at least one T-arc (*i.e.*, $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$).

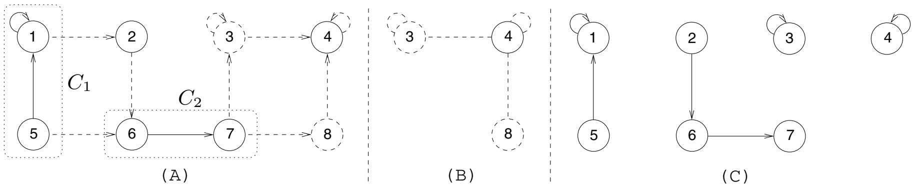

FIGURE 4. (A) Intermediate digraph and (B) undirected graph used to compute $\overline{\mathbf{NCC}}_{rem}$ for estimating $\overline{\mathbf{NCC}}$. (C) Example of a final digraph achieving the upper bound.

Part (B) illustrates the corresponding undirected graph $\overleftrightarrow{G}_{rem}$ on which we compute a maximum l-matching. We have $|cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))| = 2$, $\overline{\mathbf{NCC}}_{rem} = \mu_l(\overleftrightarrow{G}_{rem}) = 2$, thus $\overline{\mathbf{NCC}} \leq 2 + 2 = 4$. Part (C) provides a final digraph where this upper bound of 4 connected components is reached.

Property 15. *The upper bound provided by Property 14 is sharp.*

Proof. From Property 13, $\overline{\mathbf{NCC}}_{rem}$ is a sharp upper bound of the number of connected components of $\overleftrightarrow{G}_{rem}$. From Property 13, $\overline{\mathbf{NCC}}_{rem} = \mu_l(\overleftrightarrow{G}_{rem})$. Within $\overrightarrow{G}(X_{TU}, E_{TU})$, the upper bound $\overline{\mathbf{NCC}}_{rem}$ is related to all U-arcs such that their vertices are either U-vertices, or T-vertices with neither predecessor nor successor in $\overrightarrow{G}(X_T, E_T)$, that is, T-vertices that do not belong to a connected component of $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$.

The exact number of connected components of $\overrightarrow{G}(X_T, E_T)$ containing at least one T-arc is $|cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))|$. Within $\overrightarrow{G}(X_{TU}, E_{TU})$, from Property 12, all T-arcs of $\overrightarrow{G}(X_{TU}, E_{TU})$ belong to $cc_{[|E_T|\geq 1]}$ $(\overrightarrow{G}(X_T, E_T))$.

Finally, consider U-arcs such that at least one of their two vertices is a T-vertex belonging to $cc_{[|E_T|\geq 1]}$ $(\overrightarrow{G}(X_T, E_T))$. By construction turning any of such an U-arc to a T-arc will augment a previously counted connected component of $cc_{[|E_T|\geq 1]}$ $(\overrightarrow{G}(X_T, E_T))$, but cannot create a new one. Therefore, $|cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))| + \mu_l(\overleftrightarrow{G}_{rem})$ is a sharp bound. □

4.4. Bounds for the number of strongly connected components

Within [5], **NSCC** is most of the time used in graph properties of the form **NSCC** = *variable*, where *variable* is a domain variable. This means that we need to both evaluate $\underline{\mathbf{NSCC}}$ as well as $\overline{\mathbf{NSCC}}$. **NSCC** is referenced in about 12 constraints such as, for instance, `not_all_equal` (p. 688), `nvalue` (p. 698) or `soft_alldifferent_var` (p. 814).

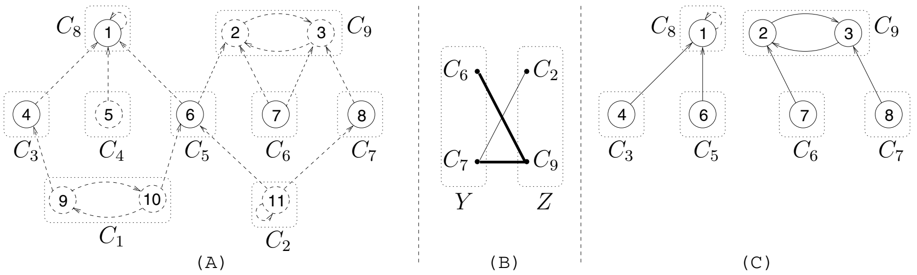

FIGURE 5. (A) Intermediate digraph and (B) corresponding bipartite graph used to compute the cardinality of a minimum hitting set. (C) Example of a final digraph achieving the lower bound.

4.4.1. *Estimating* $\underline{\mathbf{NSCC}}$

Property 16. $\underline{\mathbf{NSCC}} \geq |scc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| + h(G_{\underline{\mathbf{NSCC}}}((Y,Z),E))$, *where* $G_{\underline{\mathbf{NSCC}}}((Y,Z),E)$ *is the bipartite graph defined as follows:*

- *To each strongly connected component of* $scc_{[|X_T|=1\wedge|E_{TU}|=0]}(\overrightarrow{G}(X_{TU}, E_{TU}))$*, which has not as successor or predecessor a vertex belonging to a strongly connected component of* $scc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$*, corresponds a vertex in* Y*.*
- *To each strongly connected component of* $\overrightarrow{G}(X_{TU}, E_{TU})$*, which has an arc ingoing to or outgoing from a strongly connected component associated with an element of* Y*, corresponds a vertex of* Z*.*
- *Given a vertex* $y \in Y$ *and a vertex* $z \in Z$*,* $(y,z) \in E$ *if and only if there is an arc between a vertex of the strongly connected component associated with* Y *and a vertex of the strongly connected component associated with* Z *in the digraph* $\overrightarrow{G}(X_{TU}, E_{TU})$*.*

From Definition 5, $h(G_{\underline{\mathbf{NSCC}}}((Y,Z),E))$ *is the minimum number of elements of* Z *required to connect all elements of* Y *(i.e., the cardinality of a minimum hitting set).*

Proof. First observe that two T-vertices that belong to two distinct strongly connected components of $\overrightarrow{G}(X_{TU}, E_{TU})$ also belong to distinct strongly connected components of the final digraph G_f. Therefore $\underline{\mathbf{NSCC}} \geq |scc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))|$. Since in the final digraph any strongly connected component contains at least one arc, the T-vertices corresponding to X require to be linked to some additional strongly connected components. Then, by definition of a minimum hitting set, the minimum number of such additional strongly connected components is $h(G_{\underline{\mathbf{NSCC}}}((Y,Z),E))$. □

Example 10. Figure 5 illustrates Property 16 on the intermediate digraph $\overrightarrow{G}(X_{TU}, E_{TU})$ depicted by Part (A). All the strongly connected components C_1,

$C_2, \ldots, C_9$ of this intermediate digraph are enclosed in a dotted rectangle. Part (B) shows the corresponding bipartite graph $G_{\underline{\mathbf{NSCC}}}((Y,Z),E)$ and outlines one of its minimum hitting set with thick lines. Since 5 strongly connected components of $\overrightarrow{G}(X_{TU}, E_{TU})$ contain at least one T-vertex (*i.e.*, C_3, C_5, C_6, C_7, C_8) and since the cardinality of a minimum hitting set on the bipartite graph $G_{\underline{\mathbf{NSCC}}}((Y,Z),E)$ is equal to 1 (*i.e.*, we need one extra strongly connected component in order to connect the vertices of C_6 and C_7) we have $\underline{\mathbf{NSCC}} \geq 5+1=6$. Part (C) provides a final digraph where this lower bound of 6 strongly connected components is reached.

Property 17. *The lower bound provided by Property 16 is sharp.*

Proof. For proving that this lower bound is sharp, we show how to construct in four steps a solution that reaches $|scc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| + h(G_{\underline{\mathbf{NSCC}}}((Y,Z),E))$ strongly connected components:

(1) All U-vertices and U-arcs of strongly connected components of $scc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ are respectively turned into T-vertices and T-arcs.

(2) For all T-vertices connected to such strongly connected components, which are not involved in a T-arc, an U-arc is turned to a T-arc to connect them.

(3) Consider a minimum hitting set of $G_{\underline{\mathbf{NSCC}}}((Y,Z),E)$: we turn all U-vertices and U-arcs of the strongly connected components of the subset Z' of Z corresponding to that minimum hitting set to T-vertices and T-arcs. For each T-vertex in Y we turn an U-arc connected to a strongly connected component of Z' (by definition of $G_{\underline{\mathbf{NSCC}}}((Y,Z),E)$ such an arc always exists).

(4) Finally, all remaining U-vertices and U-arcs are respectively turned into F-vertices and F-arcs. This leads to a final digraph containing $|scc_{[|X_T|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| + h(G_{\underline{\mathbf{NSCC}}}((Y,Z),E))$ strongly connected components.

□

4.4.2. *Estimating* $\overline{\mathbf{NSCC}}$

Property 18. $\overline{\mathbf{NSCC}} \leq |scc(\overrightarrow{G}(X_{TU}, E_T))|$.

Proof. Since the digraph $\overrightarrow{G}(X_{TU}, E_T)$ does not contain any U-arcs, all its U-vertices are isolated. Thus, $|scc(\overrightarrow{G}(X_{TU}, E_T))| = |scc(\overrightarrow{G}(X_T, E_T))| + |X_U|$. Two distinct vertices of $\overrightarrow{G}(X_T, E_T)$ that belong to the same strongly connected component of $\overrightarrow{G}(X_T, E_T)$ will also belong the same strongly connected component of the final digraph G_f. Therefore we can have $|scc(\overrightarrow{G}(X_T, E_T))|$ strongly connected components in a final digraph. Moreover, we count one extra strongly connected component for each U-vertex since we can connect each of them without creating any new circuit. This leads to the upper bound. □

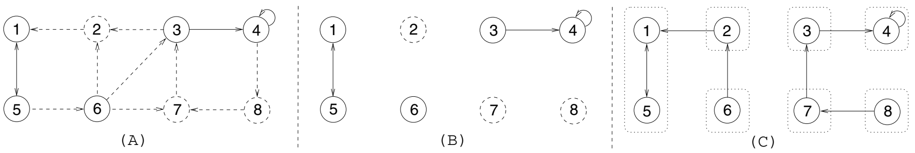

FIGURE 6. (A) Intermediate digraph and (B) graph used for estimating $\overline{\textbf{NSCC}}$. (C) Example of a final digraph achieving the upper bound.

Example 11. Parts (A) and (B) of Figure 6 illustrate Property 18. It shows how to compute the upper bound of **NSCC** according to the intermediate digraph $\overrightarrow{G}(X_{TU}, E_{TU})$ depicted by Part (A). Part (B) illustrates the corresponding digraph $\overrightarrow{G}(X_{TU}, E_T)$, which has 7 strongly connected components, thus $\overline{\textbf{NSCC}} \leq 7$. Part (C) provides a final digraph where this upper bound of 7 strongly connected components is reached.

Property 19. *The upper bound provided by Property 18 is sharp.*

Proof. We show how to construct a solution that reaches $|scc(\overrightarrow{G}(X_{TU}, E_T))|$ strongly connected components with no isolated vertices. Consider the digraph $\overrightarrow{G}(X_{TU}, E_{TU})$, where we contract each strongly connected component consisting of only T-vertices into one single vertex. On this new digraph we construct a spanning forest $\mathcal{F}$. Now, if we remove from $\overrightarrow{G}(X_{TU}, E_{TU})$ all U-arcs that do not belong to $\mathcal{F}$, we get a solution with the required number of strongly connected components with no isolated vertices. □

4.5. BOUNDS FOR THE NUMBER OF SINKS

Within [5], **NSINK** is most of the time used in graph properties of the form **NSINK** = *variable*, where *variable* is a domain variable, which means that we need to both evaluate $\underline{\textbf{NSINK}}$ as well as $\overline{\textbf{NSINK}}$. **NSINK** is referenced in about 15 constraints such as, for instance, `common` (p. 332), `same` (p. 754) or `sort` (p. 842).

4.5.1. *Estimating* $\underline{\textbf{NSINK}}$

In order to evaluate $\underline{\textbf{NSINK}}$ we first need to introduce the notion of *reduced digraph*.

Definition 8. Given a digraph G, the *reduced digraph* G_r is derived from G in the following way:

- To each strongly connected component of G corresponds a vertex of G_r.
- To each arc of G that connects different strongly connected components corresponds an arc in G_r (multiple arcs between the same pair of vertices are merged).

Property 20. **<u>NSINK</u>** $\geq |sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| + h(G'_r((Y,Z),E))$, *where* $G'_r((Y,Z),E)$ *is the bipartite graph stem from the reduced digraph of* $\overrightarrow{G}(X_{TU}, E_{TU})$ *in the following way:*

- Y *denotes the set of strongly connected components* $scc_{[|X_T|=1 \wedge |E_{TU}|=0]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ *such that:*
  - *For all* $y \in Y$, y *is not a sink in the reduced digraph.*
  - *All descendants in the reduced digraph of a vertex* $y \in Y$ *correspond to strongly connected components of* $\overrightarrow{G}(X_{TU}, E_{TU})$ *reduced to one single* U*-vertex with no arc, that is, belonging to* $scc_{[|X_T|=0 \wedge |X_U|=1 \wedge |E_{TU}|=0]}(\overrightarrow{G}(X_{TU}, E_{TU}))$.
- Z *denotes the set of strongly connected components in* $scc_{[|X_T|=0 \wedge |X_U|=1 \wedge |E_{TU}|=0]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ *that are sinks in the reduced digraph.*
- *An edge* $e = (y,z)$, $y \in Y$, $z \in Z$ *belongs to* E *if and only if there is a path from* y *to* z *in the reduced digraph.*

Proof. The vertices in $sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ will also be sinks of the final digraph G_f. Moreover all T-vertices such that there is a path to a vertex in $sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ or to a circuit of $\overrightarrow{G}(X_{TU}, E_{TU})$ can be non-sink vertices in G_f by setting to T-arcs and T-vertices all the U-arcs and U-vertices of the corresponding path.

Let us now consider the *simplified digraph* $\overrightarrow{G'}(X_{TU}, E_{TU})$ stem from $\overrightarrow{G}(X_{TU}, E_{TU})$ from which we remove all vertices v verifying one of the following conditions:

- v is in $sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ or v is in $scc_{[|E_{TU}|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ (*i.e.*, v belongs to a circuit of $\overrightarrow{G}(X_{TU}, E_{TU})$).
- There is a non-empty path from v to a T-vertex or to a circuit of $\overrightarrow{G}(X_{TU}, E_{TU})$.

By construction the minimum number of sinks over all final digraphs associated with $\overrightarrow{G}(X_{TU}, E_{TU})$ is equal to $|sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))|$ plus the minimum number of sinks over all final digraphs associated with $\overrightarrow{G'}(X_{TU}, E_{TU})$. Observe that all sources of $\overrightarrow{G'}(X_{TU}, E_{TU})$ are T-vertices corresponding to the set Y of $G'_r((Y,Z),E)$. In addition all sinks of $\overrightarrow{G'}(X_{TU}, E_{TU})$ are U-vertices corresponding to the set Z of $G'_r((Y,Z),E)$. Let v be such a sink. By construction, any U-vertex, which is an ascendant of v in $\overrightarrow{G'}(X_{TU}, E_{TU})$ has a set of ascendant T-vertices included in the set of ascendant T-vertices of v. Therefore the minimum number of sinks over all final digraphs associated with $\overrightarrow{G'}(X_{TU}, E_{TU})$ is equal to the minimum hitting set of $G'_r((Y,Z),E)$. The property holds. □

Example 12. Figure 7 illustrates Property 20 on the intermediate digraph $\overrightarrow{G}(X_{TU}, E_{TU})$ depicted by Part (A). Parts (B) and (C) respectively give the corresponding simplified digraph $\overrightarrow{G'}(X_{TU}, E_{TU})$ as well as the corresponding bipartite graph $G'_r((Y,Z),E)$, where $Y = \{6,7\}$, $Z = \{14,15\}$

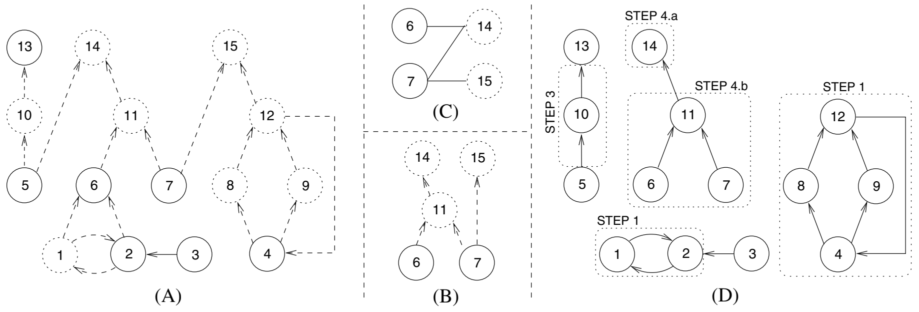

FIGURE 7. (A) Intermediate digraph, (B) corresponding simplified digraph and (C) corresponding bipartite graph used to compute the cardinality of a minimum hitting set. (D) Steps for constructing a final digraph achieving the lower bound.

and $E = \{(6,14),(7,14),(7,15)\}$. Since $\overrightarrow{G}(X_{TU}, E_{TU})$ contains a single T-vertex that is a sink, $sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU})) = \{\{13\}\}$ and thus $|sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| = 1$. The cardinality of the minimum hitting set, $h(G'_r((Y,Z),E))$, is equal to 1. As a consequence $\underline{\mathbf{NSINK}} \geq |sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| + h(G'_r((Y,Z),E)) = 2$. Part (D) provides a final digraph where this lower bound of 2 sinks is reached. The four steps for constructing such a lower bound are given in the proof of the next property.

Property 21. *The lower bound provided by Property 20 is sharp.*

Proof. For proving that this lower bound is sharp, we show how to construct a solution that reaches $|sink_{[|X_T|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| + h(G'_r((Y,Z),E))$ sinks:

(1) All U-vertices and U-arcs of a strongly connected component of $scc_{[|E_{TU}|\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$ are respectively turned into T-vertices and T-arcs.
(2) For each still isolated T-vertex u such that there is a non-empty path from u to a strongly connected component containing at least one arc, turn all U-vertices and U-arcs of that path into T-vertices and T-arc.
(3) For each still isolated T-vertex u such that there is a non-empty path from u to a T-vertex that is a sink, turn all U-vertices and U-arcs of that path into T-vertices and T-arcs.
(4) Given a minimum hitting set of $G'_r((Y,Z),E)$:
    (a) Turn all U-vertices of $\overrightarrow{G}(X_{TU}, E_{TU})$ corresponding in $G'_r((Y,Z),E)$ to the subset Z' of Z belonging to the minimum hitting set into T-vertices.
    (b) For each still isolated T-vertex u such that there is a non-empty path from u to a vertex of Z', turn all U-vertices and U-arcs of that path into T-vertices and T-arc.

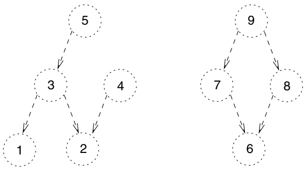

FIGURE 8. An intermediate digraph for which the upper bound of **NSINK** is not sharp.

Observe that only Step 4 implies the creation of new T-vertices that are sinks. Their number is equal to $h(G'_r((Y,Z),E))$. □

4.5.2. *Estimating* $\overline{\textbf{NSINK}}$

Notation 6. *Within $\overrightarrow{G}(X_{TU},E_{TU})$, let XP denote the set of U-vertices that are not sources and such that at least one successor is a sink having one single predecessor.*

Property 22. $\overline{\textbf{NSINK}} \leq |sink(\overrightarrow{G}(X_T,E_T))| + |X_U| - |source_{[|X_U|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))| - |XP|$.

Proof. Let S be the set of vertices that cannot for sure be sinks of the final digraph G_f. S corresponds to those T-vertices which have an outgoing T-arc. The quantity $|sink(\overrightarrow{G}(X_T,E_T))|+|X_U|$ is equal to the difference between the number of vertices of the initial digraph G_i and the number of elements of S. We can remove from this quantity the number of U-vertices that are sources in the intermediate digraph. Moreover a pending vertex in $\overrightarrow{G}(X_{TU},E_{TU})$ and its predecessor cannot be both sinks in a final digraph. Therefore we can also remove $|XP|$. □

Property 23. *The upper bound provided by Property 22 is not sharp.*

Proof. Consider the intermediate digraph depicted by Figure 8. Since we do not have any T-vertex, $|sink(\overrightarrow{G}(X_T,E_T))| = 0$. Since the number of sources is equal to 3 and since $XP = \{3\}$ we have $|sink(\overrightarrow{G}(X_T,E_T))| + |X_U| - |source_{[|X_U|=1]}(\overrightarrow{G}(X_{TU},E_{TU}))| - |XP| = 0 + 9 - 3 - 1 = 5$. But we can only get a maximum of 4 sinks since the leftmost and rightmost connected components respectively can generate at most 2 and 2 (and not 3 as suggested by the upper bound) sinks. □

Within the catalog of global constraints [5], all constraints mentioning **NSINK** are represented by a bipartite graph where the set of vertices is partitioned into the set of sources and the set of sinks. In this context the upper bound provided by Property 22 is sharp. This stems from the fact that $|X_U|-|source_{[|X_U|=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))|$ represents the set of U-vertices with no outgoing arc in the bipartite graph. Since they are not isolated they can all be turned to T-vertices.

Table 2. Bounds of the different graph parameters. Columns S and C respectively indicate the sharpness and the complexity of each bound (P and N respectively denote "polynomial" and "NP-Complete").

Graph Parameter	Bound	S	C
$\underline{\mathbf{NARC}}$	$\vert E_T\vert + \vert X_{T,\neg T}\vert - \mu(\overleftrightarrow{G}(X_{T,\neg T}, E_U))$	Yes	P
$\overline{\mathbf{NARC}}$	$\vert E_{TU}\vert$	Yes	P
$\underline{\mathbf{NVERTEX}}$	$\vert X_T\vert + h(\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T}))$	Yes	N
$\overline{\mathbf{NVERTEX}}$	$\vert X_{TU}\vert$	Yes	P
$\underline{\mathbf{NCC}}$	$\vert cc_{[\vert X_T\vert\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))\vert$	Yes	P
$\overline{\mathbf{NCC}}$	$\vert cc_{[\vert E_T\vert\geq 1]}(\overrightarrow{G}(X_T, E_T))\vert + \mu_l(\overleftrightarrow{G}_{rem})$	Yes	P
$\underline{\mathbf{NSCC}}$	$\vert scc_{[\vert X_T\vert\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))\vert + h(G_{\underline{\mathbf{NSCC}}}((Y,Z),E))$	Yes	N
$\overline{\mathbf{NSCC}}$	$\vert scc(\overrightarrow{G}(X_{TU}, E_T))\vert$	Yes	P
$\underline{\mathbf{NSINK}}$	$\vert sink_{[\vert X_T\vert=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))\vert + h(G'_r((Y,Z),E))$	Yes	N
$\overline{\mathbf{NSINK}}$	$\vert sink(\overrightarrow{G}(X_T, E_T))\vert + \vert X_U\vert - \vert source_{[\vert X_U\vert=1]}(\overrightarrow{G}(X_{TU}, E_{TU}))\vert - \vert XP\vert$	No	P

5. Summary

This section recapitulates the different bounds and provides the related information with respect to complexity and sharpness (see Tab. 2).

The problem of a sharp bound on the maximum number of sinks remains open. Regarding complexity, the three non-polynomial bounds are the lower bounds of **NVERTEX**, **NSCC** and **NSINK**. They all involve the minimum hitting set problem. For **NVERTEX** and **NSCC** the part of the formula corresponding to the minimum hitting set comes from the fact that no isolated vertex can remain in the final digraph G_f.

6. Applications to other contexts

This section shows how to use the bounds of graph parameters in the context of the CP(Graph) framework [10] and in the context of over-constrained problems.

6.1. CP(Graph)

This section shows how the results of this paper can be exploited within the CP(Graph) framework. The CP(Graph) framework introduces graph variables as well as constraints on these variables for representing some constraints related to graphs. The main similarities and differences of both approaches respectively are:

- Both the graph-based representation of global constraints introduced in [1] and the CP(Graph) framework represent a global constraint as the search of a subgraph of an initial digraph, so that this subgraph verifies some graph properties. Similar graph properties are relevant in both approaches.

Table 3. Bounds of the different graph parameters in the context of the CP(Graph) framework. Columns S and C respectively indicate the sharpness and the complexity of each bound (P and N respectively denote "polynomial" and "NP-Complete").

Graph Parameter	Bound	S	C
$\underline{\mathbf{NARC}}$	$\lvert E_T\rvert$	Yes	P
$\overline{\mathbf{NARC}}$	$\lvert E_{TU}\rvert$	Yes	P
$\underline{\mathbf{NVERTEX}}$	$\lvert X_T\rvert$	Yes	P
$\overline{\mathbf{NVERTEX}}$	$\lvert X_{TU}\rvert$	Yes	P
$\underline{\mathbf{NCC}}$	$\lvert cc_{[\lvert X_T\rvert\geq 1]}(\overrightarrow{G}(X_{TU},E_{TU}))\rvert$	Yes	P
$\overline{\mathbf{NCC}}$	$\lvert cc(\overrightarrow{G}(X_T,E_T))\rvert + \lvert X_U\rvert$	Yes	P
$\underline{\mathbf{NSCC}}$	$\lvert scc_{[\lvert X_T\rvert\geq 1]}(\overrightarrow{G}(X_{TU},E_{TU}))\rvert$	Yes	P
$\overline{\mathbf{NSCC}}$	$\lvert scc(\overrightarrow{G}(X_{TU},E_T))\rvert$	Yes	P
$\underline{\mathbf{NSINK}}$	$\lvert sink_{[\lvert X_T\rvert=1]}(\overrightarrow{G}(X_{TU},E_{TU}))\rvert + h(G'_r((Y,Z),E))$	Yes	N
$\overline{\mathbf{NSINK}}$	$\lvert sink(\overrightarrow{G}(X_T,E_T))\rvert + \lvert X_U\rvert$	Yes	P

- The CP(Graph) framework introduces graph variables, while the graph properties representation stays with domain and set variables, which allows us to model existing global constraints. The graph-based representation forbids isolated vertices in the final digraph, which is not the case for the CP(Graph) framework.

From a filtering perspective, both the CP(Graph) framework and the graph-based representation have to find out which arcs and vertices will effectively belong to the final digraph. Observe that the graph-based representation has also to propagate all binary constraints associated with the arcs of the final digraph, which is not the case for the CP(Graph) framework. The lower and upper bounds of a graph variable can be respectively reinterpreted as the graphs $\overrightarrow{G}(X_T, E_T)$ and $\overrightarrow{G}(X_{TU}, E_{TU})$. As a consequence, if we take into account the fact that isolated vertices may belong to the final digraph, all bounds of Section 4 can be easily adapted to the context of CP-graph. This leads to new bounds summarized in Table 3.

All previous bounds are sharp. Regarding complexity, all the bounds are polynomial but $\underline{\mathbf{NSINK}}$.

6.2. Over-constrained problems

In over-constrained problems all the constraints cannot be simultaneously satisfied. It is required to relax some of them to obtain a solution. Relaxed constraints are called soft constraints. A soft constraint is derived from its original constraint by adding a cost. This cost expresses a distance to the satisfaction of the original constraint. Within the context of soft global constraints, Beldiceanu and Petit presented in [6] a generic definition of costs based on graph properties. They showed that, given a ground solution, computing such a cost is polynomial. Unfortunately, the paper does not deal with filtering aspects. In order to do so, we propose to exploit the properties that were presented in Section 4. Let us first consider those soft constraints derived from global constraints that can be represented by one single graph property.

Definition 9 (Ground solutions: violation cost of a graph property [6])**.** Consider the graph property $P\ op\ V$ representing a global constraint C. Let p denotes the effective value of the graph parameter P on the final digraph G_f associated with the instantiated global constraint C. Depending on the value of op (*i.e.*, $=, \neq, \geq, \leq$), the violation cost of $P\ op\ I$ is:

- $cost(p, =, V) = |p - V|$,
- $cost(p, \neq, V) = 1 - \min(1, |p - V|)$,
- $cost(p, \geq, V) = \max(0, V - p)$,
- $cost(p, \leq, V) = \max(0, p - V)$.

In constraint programming the cost of a soft constraint can be directly represented by a domain variable of the problem[11]. In our context this variable is $cost(P, op, V)$. The soft constraint is consistent if and only if the intermediate digraph can be extended to a final digraph where the effective value p of P satisfies the expression of Definition 9 that corresponds to op. It depends on the current domains of variables V and on $cost(P, op, V)$. Typically, branch and bound algorithms for solving over-constrained problems progressively reduce the upper bound of the domain of $cost(P, op, V)$.

Furthermore, propagation according to $\underline{P}$ and $\overline{P}$ can be performed on variables V and $cost(P,\ op,\ V)$. A constraint involving P, V and $cost(P, op, V)$ can be directly derived from Definition 9: instead of considering a single value p, we then consider a range of allowed values for P, which is defined by $\underline{P}$ and $\overline{P}$. Such a constraint can be propagated in a classical way, providing us with a filtering technique. In this context, all the propositions of Section 4 are relevant.

Definition 10 (Graph properties based violation cost [6])**.** Consider a global constraint defined by a conjunction of graph properties. The violation cost of such a global constraint is the sum of the violation costs of its graph properties.

From Definition 10, the filtering technique can be generalized to constraints defined by several graph properties, by propagating a constraint derived from several expressions instead of being derived from a single one.

7. Conclusion

This article provides lower and upper bounds for the set of most common graph parameters used in the graph-based representation of global constraints. It is the first step to mechanically obtain a filtering algorithm for those global constraints that can be specified by a conjunction of graph properties. It also highlights the use of bounds of graph parameters for the CP(Graph) framework as well as for

[11] Both for the sake of efficiency [1, 22, 28, 29] and expressivity [21].

constraint relaxation. In order to get a stronger filtering, a natural continuation of this work would be to identify and drop elements of the initial digraph according to the bounds of graph parameters. Finally to get efficient filtering algorithms one should also specialized our bounds to specific graph classes that arise in practice in the catalog [5].

Acknowledgements. The authors thank Sophie Demassey for clarifying notation 2.

References

[1] P. Baptiste, C. Le Pape and L. Peridy, Global Constraints for Partial CSPs: A Case-Study of Resource and Due Date Constraints, in *Principles and Practice of Constraint Programming (CP'98)*, edited by M. Maher and J.-F. Puget, Springer-Verlag, *Lect. Notes Comput. Sci.* **1520** (1998) 87–101.

[2] N. Beldiceanu, Global Constraints as Graph Properties on a Structured Network of Elementary Constraints of the Same Type, in *Principles and Practice of Constraint Programming (CP'2000)*, edited by R. Dechter, Springer-Verlag, *Lect. Notes Comput. Sci.* **1894** (2000) 52–66. Preprint available as SICS Tech Report T2000-01.

[3] N. Beldiceanu, *Global Constraints as Graph Properties on Structured Network of Elementary Constraints of the Same Type.* Technical Report T2000-01, Swedish Institute of Computer Science (2000).

[4] N. Beldiceanu, M. Carlsson and T. Petit, Deriving Filtering Algorithms from Constraint Checkers, in *Principles and Practice of Constraint Programming (CP'2004)*, edited by M. Wallace, Springer-Verlag, *Lect. Notes Comput. Sci.* **3258** (2004) 107–122. Preprint available as SICS Tech Report T2004-08.

[5] N. Beldiceanu, M. Carlsson and J.-X. Rampon, *Global Constraint Catalog.* Technical Report T2005-08, Swedish Institute of Computer Science (2005).

[6] N. Beldiceanu and T. Petit, Cost Evaluation of Soft Global Constraints, in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimisation Problems (CP-AI-OR 2004)*, edited by J.-C. Régin and M. Rueher, Springer-Verlag, *Lect. Notes Comput. Sci.* **3011** (2004) 80–95.

[7] C. Bessière, E. Hebrard, B. Hnich, Z. Kızıltan and T. Walsh, Filtering Algorithms for the *nvalue* Constraint, in *International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CP-AI-OR'05), Prague, Czech Republic*, edited by R. Barták and M. Milano, Springer-Verlag, *Lect. Notes Comput. Sci.* **3524** (2005) 79–93

[8] C. Bessière and P. Van Hentenryck, To Be or not to Be . . . a Global Constraint, in *Principles and Practice of Constraint Programming (CP'2003)*, edited by F. Rossi, Springer-Verlag, *Lect. Notes Comput. Sci.* **2833** (2003) 789–794.

[9] C. Bessière and J.-C. Régin, Refining the Basic Constraint Propagation Algorithm, in *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*, edited by B. Nebel, Morgan Kaufmann (2001) 309–315.

[10] G. Dooms, Y. Deville and P. Dupont, CP(Graph): Introducing a Graph Computation Domain in Constraint Programming, in *Principles and Practice of Constraint Programming (CP'2005)*, edited by P. van Beek, Springer-Verlag, *Lect. Notes Comput. Sci.* **3709** (2005) 211–225.

[11] E.C. Freuder and R.J. Wallace, Partial constraint satisfaction. *Artificial Intelligence* **58** (1992) 21–70.

[12] M.R. Garey and D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness.* W.H. Freeman and Company (1979).

[13] D. Hanák, Implementing Global Constraints as Structured Graphs of Elementary Constraints. *Scientific Journal Acta Cybernetica* **16** (2003) 241–258.

[14] P. Van Hentenryck, Y. Deville and C.M. Teng, A Generic Arc Consistency Algorithm and its Specializations. *Artificial Intelligence* **57** (1992) 291–321.

[15] P. Van Hentenryck, V. Saraswat and Y. Deville, Design, implementation, and evaluation of the constraint language cc(FD). *J. Logic Programming* **37** (1998) 139–164.

[16] I. Katriel and S. Thiel, Fast Bound Consistency for the *global cardinality* Constraint, in *Principles and Practice of Constraint Programming (CP'2003)*, edited by F. Rossi, Springer-Verlag, *Lect. Notes Comput. Sci.* **2833** (2003) 437–451.

[17] K. Mehlhorn and S. Thiel, Faster Algorithms for Bound-Consistency of the *sortedness* and the *alldifferent* Constraint, in *Principles and Practice of Constraint Programming (CP'2000)*, edited by R. Dechter, Springer-Verlag, *Lect. Notes Comput. Sci.* **1894** (2000) 306–319.

[18] U. Montanari, Networks of constraints: Fundamental properties and applications to picture processing. *Information Science* **7** (1974) 95–132.

[19] R.Z. Norman and M.O. Rabin, An algorithm for minimum cover of a graph. *American Math. Soc.* **10** (1959) 315–319.

[20] G. Pesant, A Regular Language Membership Constraint for Finite Sequences of Variables, in *Principles and Practice of Constraint Programming (CP'2004)* edited by M. Wallace, Springer-Verlag, *Lect. Notes Comput. Sci.* **3258** (2004) 482–495.

[21] T. Petit, J-C. Régin and C. Bessière, Meta constraints on violations for over constrained problems, in *12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2000), 13-15 November 2000, Vancouver, BC, Canada*, IEEE Computer Society (2000) 358–365.

[22] T. Petit, J-C. Régin and C. Bessière, Specific filtering algorithms for over constrained problems, in *Principles and Practice of Constraint Programming (CP'2001)*, edited by T. Walsh, Springer-Verlag, *Lect. Notes Comput. Sci.* **2239** (2001) 451–463.

[23] C.-G. Quimper, A. López-Ortiz, P. van Beek and A. Golynski, Improved Algorithms for the *global cardinality* Constraint, in *Principles and Practice of Constraint Programming (CP'2004)*, edited by M. Wallace, Springer-Verlag, *Lect. Notes Comput. Sci.* **3258** (2004) 542–556.

[24] J.-C. Régin, A Filtering Algorithm for Constraints of Difference in CSP, in *12th National Conference on Artificial Intelligence (AAAI-94)* (1994) 362–367.

[25] J.-C. Régin, Generalized Arc Consistency for *global cardinality* Constraint, in *14th National Conference on Artificial Intelligence (AAAI-96)* (1996) 209–215.

[26] J.-C. Régin, The Symmetric *alldiff* Constraint, in *16th Int. Joint Conf. on Artificial Intelligence (IJCAI-99)* (1999) 420–425.

[27] P. Turán, On an Extremal Problem in Graph Theory. *Mat. Fiz. Lapok* **48** (1941) 436–452, in Hungarian.

[28] W.-J. van Hoeve, A Hyper-Arc Consistency Algorithm for the *soft alldifferent* Constraint, in *Principles and Practice of Constraint Programming (CP'2004)*, edited by M. Wallace, Springer-Verlag, *Lect. Notes Comput. Sci.* **3258** (2004) 679–689.

[29] W.-J. van Hoeve, G. Pesant and L.-M. Rousseau, On global warming: Flow-based soft global constraints, in *Journal of Heuristics* **12** (2006) 347–373.

[30] N.R. Vempaty, Solving Constraint Satisfaction Problems using Finite State Automata, in *National Conference on Artificial Intelligence (AAAI-92)*, AAAI Press (1992) 453–458.