Filtrage basé sur des propriétés de graphe

Nicolas Beldiceanu, Matts Carlsson, Sophie Demassey, Thierry Petit

Filtrage basé sur des propriétés de graphe

Nicolas Beldiceanu, Mats Carlsson, Sophie Demassey et Thierry Petit

École des Mines de Nantes, LINA FRE CNRS 2729, FR-44307 Nantes, France.
SICS, P.O. Box 1263, SE-164 29 Kista, Sweden.
{Nicolas.Beldiceanu,Sophie.Demassey,Thierry.Petit}@emn.fr, Mats.Carlsson@sics.se

Résumé

Cet article présente un schéma de filtrage générique, basé sur la description de contraintes globales sous la forme de propriétés de graphes. Cette description est définie par un réseau de contraintes binaires et une liste de propriétés de graphe élémentaires : chaque solution de la contrainte globale correspond à un sous-graphe du réseau initial, dans lequel ne sont retenues que les contraintes binaires satisfaites. Ce sous-graphe doit vérifier les propriétés de graphe qui définissent la contrainte. Le filtrage consiste en l'identification des arcs du réseau qui appartiennent ou non aux sous-graphes solution. L'objectif est de construire, à côté du catalogue de contraintes, une liste de règles de filtrage systématiques. Ces règles sont basées sur un ensemble limité de propriétés. Elles s'appliquent à toutes les contraintes du catalogue décrites à l'aide de ces propriétés. Nous illustrons ce principe sur les propriétés de graphe les plus usuelles, et nous expérimentons notre technique sur la contrainte `group`.

1 Introduction

Le catalogue de contraintes globales [3] fournit la description de centaines de contraintes en termes de propriétés de graphe : les solutions d'une contrainte globale sont identifiées aux sous-graphes d'un unique digraphe initial, qui satisfont un ensemble de propriétés de graphes définissant la contrainte. Les propriétés de graphes utilisent un ensemble de paramètres, tels le nombre de sommets ou le nombre de composantes connexes (cc). Les paramètres les plus communs ont été étudiés dans [6], où il est montré comment on estime, à partir du digraphe (*i.e.*, graphe orienté) final, les bornes inférieures et supérieures dans les sous-graphes solution possibles. Ces bornes fournissent les conditions nécessaires de faisabilité de presque toutes les contraintes globales. Cet article fait un pas en avant supplémentaire, en introduisant des règles de filtrage systématiques. L'idée intuitive vient du fait que le digraphe initial décrivant une contrainte globale représente un réseau de contraintes définies sur des paires de variables. À chaque instanciation complète des variables correspond un sous-graphe final, obtenu en supprimant du digraphe initial tous les arcs (*i.e.,* les contraintes binaires) qui ne sont pas satisfaites. Sachant que les solutions d'une contrainte globale correspondent aux sous-graphes finaux satisfaisant un ensemble de propriétés de graphe, le filtrage consiste en l'identification et la suppression d'arcs et de de sommets du digraphe initial n'appartenant pas à ces sous-graphes, ou, à l'inverse, il consiste à forcer certains arcs ou sommets à appartenir à tout digraphe solution. Une permière façon d'effectuer cette identification pourrait être d'employer une technique de "shaving" [10], *i.e*, fixer l'état d'un arc ou d'un sommet, et tester si cet état conduit à une contradiction. Malheureusement cela serait trop coûteux en pratique. Aussi, nous présentons dans cet article une technique plus réaliste. Les règles de filtrage proposées par la suite s'appliquent dès lors q'un paramètre est fixé à la valeur d'une de ses bornes. Enfin, les contraintes globales peuvent être partitionnées en fonction de la classe à laquelle appartient leur digraphe final. Prendre en compte une classe de graphe donnée permet une meilleure estimation des bornes des paramètres, et un filtrage plus efficace. L'article est organisé de la sorte : la section 2 rappelle la description des contraintes sous la forme de propriétés de graphes et introduit une reformulation correspondant à cette représentation. La section 3 établit une série de notations utiles à la formalisation du filtrage systématique basé sur les graphes. La section 4 présente les règles relatives aux bornes de certains paramètres de graphes. La section 5 montre comment le filtrage basé sur les graphes rejoint les algorithmes de filtrage ad-hoc existants pour certaines contraintes globales. La section 6 illustre comment raffiner le filtrage en fonction d'une classe de graphe particulière et fournit des résultats expérimentaux sur la contrainte `group`, qui appartient à la classe de graphe `path_with_loops` .

2 Description des contraintes globales basée sur les graphes

2.1 Description sous la forme de graphes

Soit $C(V_1, \ldots, V_p, x_1, \ldots, x_n)$ une contrainte globale impliquant des variables domaine[1] $V_1, \ldots, V_p$, et des variables domaine ou ensemblistes[2] $x_1, \ldots, x_n$. Quand celle-ci existe, une description de C sous la forme de graphes est donnée par un (ou plusieurs) réseau(x) $G_{\mathcal{R}} = (X, E_{\mathcal{R}})$ de contraintes binaires sur $X = \{x_1, \ldots, x_n\}$, et par un ensemble $\mathcal{GP}_{\mathcal{R}} = \{\mathcal{P}_l \ op_l \ V_l \mid l = 1, \ldots, m\}$ de propriétés de graphe, et, optionnellement, par une classe de graphe $c_{\mathcal{R}}$, de sorte que :

- Les contraintes binaires définissant le digraphe $G_{\mathcal{R}} = (X, E_{\mathcal{R}})$ partagent la même sémantique (typiquement, il peut s'agir d'une égalité, ou bien d'une inégalité, ou d'une contrainte supérieur ou inférieur). On note $x_j \mathcal{R} x_k$ la *contrainte d'arc* définie sur la paire ordonnée de variables $(x_j, x_k) \in E_{\mathcal{R}}$ (ou la contrainte unaire correspondante si $j = k$).
- $\mathcal{P}_l \ op_l \ V_l$ exprime la propriété de graphe comparant la valeur d'un paramètre $\mathcal{P}_l$ à la valeur de la variable V_l. L'opérateur de comparaison op_l peut être $\geq, \leq, =$, ou $\neq$. Parmi les paramètres $\mathcal{P}_l$ les plus usuels, $\mathbf{NARC}$ désigne le nombre d'arcs d'un graphe, $\mathbf{NVERTEX}$ le nombre de sommets, $\mathbf{NCC}$ le nombre de cc, $\mathbf{MIN\_NCC}$ et $\mathbf{MAX\_NCC}$ le nombre de sommets de la plus petite et plus grande cc respectivement.
- $c_{\mathcal{R}}$ correspond aux classes de graphes récurrentes qui sont impliquées pour différentes contraintes globales. Par exemple, on considère les graphes des classes suivantes : `acyclic`, `bipartite`, `path_with_loops`.

$G_{\mathcal{R}}$ est appellé le *digraphe initial*. Quand toutes les variables x ont été instanciées, le sous-graphe de $G_{\mathcal{R}}$ obtenu en supprimant toutes les contraintes sur des couples de variables $x_j \mathcal{R} x_k$ non satistaites, ainsi que les sommets devenant isolés, est appelé un *digraphe final* (associé à l'instanciation). Il est noté $G_f = (X_f, E_f)$. La relation entre C et sa description sous la forme de graphe est la suivante.

Définition 1 *Une instanciation complète des variables $V_1, \ldots, V_p, x_1, \ldots, x_n$ est une solution de C ssi le digraphe final associé à l'instanciation de $x_1, \ldots, x_n$, satisfait l'ensemble des propriétés de graphe $\mathcal{P}_l \ op_l \ V_l$ dans $\mathcal{GP}_{\mathcal{R}}$ et appartient à la classe de graphes $c_{\mathcal{R}}$.*

[1] Une *variable domaine* D est une variable prenant ses valeurs dans un ensemble fini d'entiers $dom(D)$. $min(D)$ et $max(D)$ désignent respectivement les valeurs *minimales* et *maximales* dans $dom(D)$.

[2] Une *variable ensembliste* S est une variable qui sera instanciée à un ensemble fini de valeurs entières. Son domaine est défini par sa *borne inférieure* $\underline{S}$, et sa *borne supérieure* $\overline{S}$, et contient tous les ensembles dans lesquels $\underline{S}$ est inclus et qui sont inclus dans $\overline{S}$.

Exemple 1 *Soit la contrainte* `proper_forest`(`NTREE`, `VER`) *introduite dans [5]. Elle implique une variable domaine* `NTREE` *et un digraphe G décrit par une collection de n sommets* `VER` *: chaque sommet est étiqueté par un entier compris entre 1 et n et il est représenté par une variable ensembliste dont les bornes inférieures et supérieures sont les ensembles d'étiquettes de, respectivement, ses* voisins obligatoires *et ses* voisins obligatoires ou potentiels *dans G. Cette contrainte partitionne les sommets de G en un ensemble d'arbres propres ayant chacun au moins deux sommets. La partie (A) de la figure 1 illustre un tel digraphe G ; les flêches pleines sont les arcs obligatoires et les flêches en pointillés sont les arcs possibles. La partie (B) de la figure montre une solution possible sur ce digraphe avec trois arbres propres.*

Dans la description de `proper_forest` *basée sur les graphes, le digraphe initial correspond exactement à G et n'a pas de boucle. N'importe quel digraphe final G_f contient tous les arcs obligatoires de G et appartient à la classe de graphe* `symmetric`[3]*. En outre, G_f doit satisfaire les propriétés de graphe suivantes :* $\mathbf{NVERTEX} = n$ *(comme il s'agit d'un problème de partitionnement, G_f contient tous les sommets de G),* $\mathbf{NARC} = 2 \cdot (n - \mathtt{NTREE})$ *(2 car G_f est symétrique, et $n - $* `NTREE` *car on a* `NTREE` *graphes connexes acycliques) et* $\mathbf{NCC} =$ `NTREE`*.*

2.2 Reformulation basée sur les graphes

D'après la définition 1, n'importe quelle contrainte globale $C(V_1, \ldots, V_p, x_1, \ldots, x_n)$ possédant une description basée sur les graphes peut être reformulée.

Proposition 1 *Soit un ensemble de variables additionnelles à chaque réseau de contraintes $G_{\mathcal{R}} = (X, E_{\mathcal{R}})$ défini par : À chaque sommet x_j et à chaque arc e_{jk} de $G_{\mathcal{R}}$ correspondent des variables* 0-1 *notées respectivement $vertex_j$ et arc_{jk}. Vertex et Arc désignent ces ensembles de variables. Enfin, G_f désigne le sous-graphe de $G_{\mathcal{R}}$, dont les sommets et les arcs correspondent aux variables $vertex_j$ et arc_{jk} instanciées à 1. La contrainte C est satisfaite ssi les contraintes suivantes sont vérifiées :*
− Pour chaque arc e_{jk} de $G_{\mathcal{R}}$:

$$arc_{jk} = 1 \ \Leftrightarrow \ x_j \mathcal{R} x_k \tag{1}$$

− Pour chaque sommet x_j de $G_{\mathcal{R}}$:

$$vertex_j = \min(1, \sum_{\{k \mid e_{jk} \in E_{\mathcal{R}}\}} arc_{jk} + \sum_{\{k \mid e_{kj} \in E_{\mathcal{R}}\}} arc_{kj}) \tag{2}$$

− Pour chaque propriété de graphe $\mathcal{P}_l \ op_l \ V_l$ dans $\mathcal{GP}_{\mathcal{R}}$:

$$ctr_{\mathcal{P}_l}(Vertex, Arc, P_l) \tag{3}$$

Cette contrainte est satisfaite quand P_l est égale à la valeur du paramètre correspondant $\mathcal{P}_l$ dans G_f.

[3] Un digraphe est *symétrique* ssi, s'il existe un arc de u à v, alors il existe aussi un arc de v à u.

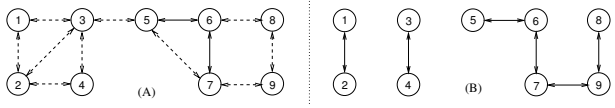

FIG. 1 – (A) A digraphe et (B) une solution avec trois arbres propres pour la contrainte `proper_forest`

– Pour chaque propriété de graphe, $\mathcal{P}_l$ op_l V_l dans $\mathcal{GP}_\mathcal{R}$:

$$P_l \; op \; V_l \tag{4}$$

– La contrainte de classe de graphe *suivante est satisfaite si G_f appartient à la classe de graphe $c_\mathcal{R}$:*

$$ctr_{c_\mathcal{R}}(Vertex, Arc) \tag{5}$$

Exemple 2 *Soit, à nouveau, la contrainte* `proper_forest`. *Comme son digraphe final est symétrique et ne contient pas de sommets isolés, la contrainte de classe de graphe* (5) *correspond à la conjonction de contraintes suivante : $arc_{jk} = arc_{kj}$ pour chaque arc e_{jk} et $vertex_j = \min(1, 2 \cdot \sum_{\{k \mid e_{jk} \in E_\mathcal{R}\}} arc_{jk})$ pour chaque sommet x_j de $G_\mathcal{R}$.*

Le filtrage des domaines des variables V et x en fonction de C peut être effectué en imposant alternativement, et pour chaque réseau de contraintes $G_\mathcal{R}$, les cinq ensembles de contraintes de cette reformulation. Propager les contraintes (1), (2), (4) et (5) est plutôt trivial car ces contraintes sont des contraintes arithmétiques élémentaires. Le filtrage générique basé sur les graphes consiste alors principalement à maintenir la consistance entre les variables arc et $vertex$ et les bornes des variables paramètres P_l, en respectant les contraintes (3). L'article [6] propose, pour les paramètres les plus usuels, l'estimation des valeurs minimales ($\underline{P_l}$) et maximales ($\overline{P_l}$) dans tout graphe final G_f, étant donné l'état courant des sommets et arcs de $G_\mathcal{R}$. La section 4 montre comment à l'inverse, l'état de certains arcs et sommets peut être déterminé en fonction d'un paramètre quand celui-ci est instancié à une de ses valeurs extrêmes (i.e., $dom(P_l) = \{\underline{P_l}\}$ or $dom(P_l) = \{\overline{P_l}\}$). En effet, l'approche consiste en l'identification des digraphes finaux possibles dans $G_\mathcal{R}$ qui minimisent ou maximisent un paramètre donné. N'importe quel digraphe final contient (resp. ne contient pas) les arcs et les sommets correspondant aux variables arc et $vertex$ fixées à 1 (resp. 0). Puisqu'un graphe final n'a pas de sommets isolés, nous supposons que les *contraintes de normalisation* (2) sont prpagées avant d'estimer chaque paramètre. La section 6 montre comment raffiner l'estimation quand les digraphes finaux appartiennent à une classe de graphes donnée, en propageant à nouveau des contraintes de type (5).

3 Notations pour un filtrage systématique

Comme pour la description de contraintes globales sous la forme de graphes, nous souhaitons fournir un catalogue de règles de filtrage génériques relatives aux bornes des paramètres. Afin de formaliser cela, nous introduisons un certain nombre de notations. Soit $G_\mathcal{R}$ un digraphe initial associé à la description d'une contrainte globale sous la forme de graphes. Les domaines des variables arc et $vertex$ de la reformulation correspondent à un partitionnement unique des ensembles d'arcs et de sommets de $G_\mathcal{R}$.

Notation 1 *Un sommet x_j ou un arc e_{jk} de $G_\mathcal{R}$ est soit* true *(T), soit* false *(F), ou* undetermined *(U) si la variable correspondante $vertex_j$ ou arc_{jk} est fixée à 1, fixée à 0, ou n'a pas encore une valeur déterminée (domaine $\{0, 1\}$). Cette terminologie induit un partitionnement de l'ensemble des sommets de $G_\mathcal{R}$ en $X_T \dot\cup X_F \dot\cup X_U$ et au partitionnement de l'ensemble des arcs de $G_\mathcal{R}$ en $E_T \dot\cup E_F \dot\cup E_U$. Pour deux éléments distincts Q et R dans $\{T, U, F\}$, soit X_{QR} le sous-ensemble de sommets $X_Q \dot\cup X_R$, et E_{QR} le sous ensemble d'arcs $E_Q \dot\cup E_R$.*

Une fois les contraintes de normalisation propagées, le sous-graphe (X_T, E_T) est bien défini, inclus dans tout digraphe final potentiel. (X_{TU}, E_{TU}) est aussi un sous-graphe de $G_\mathcal{R}$, que l'on appelle le *digraphe intermédiaire*. Tout digraphe final est dérivé de ce digraphe intermédiaire en changeant l'état des U-arc et U-sommets vers T ou F[4]. L'objectif est d'identifier les digraphes finaux dans lesquels le paramètre P atteint sa plus petite valeur $\underline{P}$ ou sa plus grande valeur $\overline{P}$. Une borne est dite *réalisable* si pour tout digraphe intermédiaire il existe au moins un digraphe final où le paramètre prend cette valeur. Afin d'estimer ces bornes, on traite différents digraphes dérivés du digraphe intermédiaire.

Notation 2 *Pour tous les mots Q, R et S formés à partir de l'alphabet $\{T, U, F\}$, X_Q et X_S sont des sous-ensembles de sommets et E_R est un sous-ensemble d'arcs du digraphe initial, et :*

- *$X_{Q,R}$ (resp. $X_{Q,\neg R}$) désigne l'ensemble de sommets de X_Q qui sont extrémité d'au moins un arc (resp. d'aucun arc) dans E_R.*
- *$X_{Q,R,S}$ (resp. $X_{Q,R,\neg S}$) désigne l'ensemble des sommets de $X_{Q,R}$ qui sont liés à au moins un sommet (resp. à aucun sommet) dans X_S par un arc de E_R.*

[4] Dans le contexte de CP(Graph) [8], ces deux digraphes correspondent respectivement à la borne inférieure et à la borne supérieure d'une variable de graphe. On remarquera que, en conséquence, notre approche peut facilement être adaptée afin de fournir un filtrage générique pour CP(Graph).

- $X_{Q,\neg R,S}$ *(resp.* $X_{Q,\neg R,\neg S}$*) désigne l'ensemble des sommets de* $X_{Q,\neg R}$ *qui sont liés à au moins un sommet (resp. à aucun sommet) de* X_S *par un arc de* E_{TU}*.*
- $E_{R,Q}$ *est l'ensemble des arcs de* E_R *qui sont incidents à au moins un sommet de* X_Q*.*
- $E_{R,Q,S}$ *est l'ensemble des arcs de* E_R *qui sont incidents à un sommet de* X_Q *et un sommet de* X_S*.*

Notation 3 *Etant donné un digraphe* G *et les sous-ensembles* $\mathcal{X}$ *de sommets et* $\mathcal{E}$ *d'arcs :*

- $\overrightarrow{G}(\mathcal{X},\mathcal{E})$ *désigne le sous-graphe induits de* G *contenant tous les sommets de* $\mathcal{X}$ *et tous les arcs de* $\mathcal{E}$ *ayant leurs deux extrémités dans* $\mathcal{X}$*.*
- $\overleftrightarrow{G}(\mathcal{X},\mathcal{E})$ *désigne le graphe non orienté correspondant : à une arête* (u,v) *correspond au moins un arc (ou boucle)* (u,v) *ou* (v,u) *dans* $\overrightarrow{G}(\mathcal{X},\mathcal{E})$*.*
- $cc(G)$ *désigne l'ensemble des cc de* G *et* $cc_{[cond]}(G)$ *désigne le sous-ensemble de cc satisfaisant une condition* $cond$*.*

Notation 4 $\overleftrightarrow{G}_{rem}$ *désigne le sous-graphe induit (non orienté) de* $\overleftrightarrow{G}(X_{TU},E_U)$ *obtenu en supprimant tous les sommets présents dans* $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T,E_T))$ *et ensuite en supprimant du graphe non orienté résultant tous les sommets devenant isolés.*

Enfin nous rappelons quelques notions de théorie des graphes.

Définition 2 *Un* couplage *d'un graphe non orienté* G *est un sous ensemble d'arêtes, sans boucle, tel qu'aucune paire d'arête n'ait de sommet commun. Un* couplage maximal *est un couplage de cardinalité maximale.* $\mu(G)$ *est la cardinalité d'un couplage maximal de* G*. Si la présence de boucles est autorisée dans un couplage alors on appelle ce couplage un* l-couplage *et la cardinlité maximum d'un* l*-couplage de* G *est notée* $\mu_l(G)$*.*

Étant donné un graphe biparti $G((Y,Z),E)$*, un* hitting set *de* $G((Y,Z),E)$ *est un sous-ensemble* Z' *de* Z *tel que pour n'importe quel sommet* $ey \in Y$ *il existe un arête de* E *joignant* y *à un sommet de* Z'*.* $h(G)$ *est la cardinalité d'un minimum hitting set de* G*.*

4 Filtrage basé sur les bornes

Cette section illustre, à travers les exemples de **NVERTEX** et **NCC**, comment réaliser un filtrage pour une contrainte exprimée par des paramètres de graphe $ctr_{\mathcal{P}_l}(Vertex, Arc, P_l)$ (3). Le table 1 rappelle préalablement les formules génrériques utiles pour estimer les bornes de ces deux paramètres en fonction de l'instanciation courante de $Vertex$ et Arc. Ces résultats ont été publiés dans [6]. On remarquera que toutes ces bornes sont réalisables. Nous présentons ensuite un filtrage en sens inverse, lorsque $dom(P) = \{\underline{P}\}$ ou $dom(P) = \{\overline{P}\}$. Les règles suivantes permettent de déterminer l'état des U-sommets et U-arcs du digraphe intermédiaire quand n'importe quel digraphe doit contenir exactement le nombre maximal ou le nombre minimal de sommets ou de cc. Avant d'établir les théorèmes, une intuition de la formule correspondante est fournie.

Paramètre	Borne
$\underline{\mathbf{NVERTEX}}$	$\lvert X_T\rvert + h(\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T}))$
$\overline{\mathbf{NVERTEX}}$	$\lvert X_{TU}\rvert$
$\underline{\mathbf{NCC}}$	$\lvert cc_{[\lvert X_T\rvert\geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))\rvert$
$\overline{\mathbf{NCC}}$	$\lvert cc_{[\lvert E_T\rvert\geq 1]}(\overrightarrow{G}(X_T, E_T))\rvert + \mu_l(\overleftrightarrow{G}_{rem})$

TAB. 1 – Bornes des différents paramètres.

4.1 Filtrage en fonction de $\underline{\mathbf{NVERTEX}}$

$\underline{\mathbf{NVERTEX}}$ est égal au nombre courant de T-sommets, $|X_T|$, plus le nombre minimal de U-sommets qui devront être transformés en T-sommets afin d'éviter les T-sommets isolés dans un graphe final. Cette estimation est basée sur le calcul de la cardinalité minimale d'un hitting set.

Théorème 1 *Si* $dom(\mathbf{NVERTEX}) = \{\underline{\mathbf{NVERTEX}}\}$ *alors*

1. *Tous les* U*-sommets de* $X_{U,\neg T,\neg T}$ *peuvent être transfromés en* F*-sommets.*
2. *Tous les* U*-sommets de* $X_{U,\neg T,T}$ *qui n'appartiennent à aucun hitting set minimal de* $\overleftrightarrow{G}$ $((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T})$ *peuvent être transformés en* F*-sommets (notamment les sommets reliés à aucun sommet de* $X_{T,\neg T,\neg T}$*).*
3. *Tous les* U*-sommets de* $X_{U,\neg T,T}$ *qui appartiennent à tous les hitting set minimaux de* $\overleftrightarrow{G}$ $((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T})$ *peuvent être transformés en* T*-sommets.*
4. *Pour toutes les arêtes* $e = (u,v)$ *telles que* $u \in X_{T,\neg T,\neg T}$ *et* $v \in X_{U,\neg T,T}$*, si tous les minimum hitting sets sont tels que* v *est le seul sommet qui puisse être associé à* u*, et si un unique arc correspond à* e *dans le digraphe orienté* $\overrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T})$*, alors cet arc peut être transformé en* T*-arc.*

Exemple 3 *La partie (A) de la figure 2 illustre le théorème 1 avec l'hypothèse que le digraphe final ne doive pas contenir plus de 7 sommets*[5]*. La partie (B) illustre le graphe biparti correspondant* $\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T})$*, utilisé pour calculer la cardinalité minimale d'un hitting set. Les lignes en gras correspondent à un hitting set minimal. La partie (A) illustre la propagation obtenue en appliquant les différents items du théorème 1 :*

[5]Comme dans la figure 1, les flêches pleines/cercles pleins sont des T-arcs/sommets et les flêches en pointillés/cercles en pointillés sont des U-arcs/sommets.

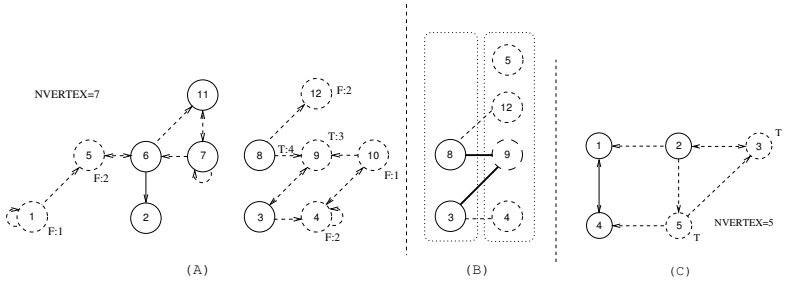

FIG. 2 – Filtrage en fonction de $\mathbf{NVERTEX}$: illustration des théorèmes 1 (A, B) et 2 (C).

- *Les U-sommets 1 et 10 peuvent être transformés en F-sommets d'après l'item (1.).*
- *Sachant qu'ils n'appartiennent à aucun hitting set minimal, les U-sommets 4, 5 et 12 peuvent être transformés en F-sommets d'après l'item (2.).*

Le théorème 1 impose de calculer la cardinalité minimale d'un hitting set, ce qui est exponentiel. Nous proposons donc une forme allégée de ce théorème.

Corollaire 1 *Si $dom(\mathbf{NVERTEX}) = \{|X_T|\}$ alors tous les U-sommets peuvent être transformés en F-sommets.*

4.2 Filtrage en fonction de $\overline{\mathbf{NVERTEX}}$

$\overline{\mathbf{NVERTEX}}$ correspond aux digraphes finaux dérivés de $\overleftrightarrow{G}(X_{TU}, E_{TU})$ en transformant tous les U-sommets en T.

Théorème 2 *Si $dom(\mathbf{NVERTEX}) = \{\overline{\mathbf{NVERTEX}}\}$ alors tous les U-sommets de $\overleftrightarrow{G}(X_{TU}, E_{TU})$ peuvent être transformés en T-sommets.*

Exemple 4 *La partie (C) de la figure 2 illustre le théorème 2 avec l'hypothèse que le digraphe final puisse contenir au moins 5 sommets. Le théorème 2 transforme tous les U-sommets en T-sommets.*

4.3 Filtrage en fonction de $\underline{\mathrm{NCC}}$

Le nombre minimal de cc dans tout graphe final issu du graphe intermédiaire est égal au nombre de cc du graphe intermédiaire qui contiennent au moins un T-sommet.

Théorème 3 *Si $dom(\mathbf{NCC}) = \{\underline{\mathbf{NCC}}\}$ alors*

1. *N'importe quel U-arc ou U-sommet d'une cc dans $|cc_{[|X_T|=0]}(\overrightarrow{G}(X_{TU}, E_{TU}))|$ peut être transformé en F-arc ou F-sommet.*
2. *Tout U-sommet qui est un point d'articulation de $\overleftrightarrow{G}(X_{TU}, E_{TU})$ et tel que sa suppression déconnecte deux T-sommets[6] peut être transformé en T-sommet.*
3. *Pour chaque arête e de $\overleftrightarrow{G}(X_{TU}, E_{TU})$ qui est un isthme dont la suppression déconnecterait deux T-sommets, si un unique U-arc de $\overrightarrow{G}(X_{TU}, E_{TU})$ correspond à e alors cet U-arc peut être transformé en T-arc.*

[6]Les deux T-sommets n'appartiennent pas à la même cc.

Exemple 5 *La partie (A) de la figure 3 illustre le théorème 3 en posant l'hypothèse que le digraphe final ne doit pas contenir plus qu'une cc. La partie (B) représente le graphe non orienté $\overleftrightarrow{G}(X_{TU}, E_{TU})$, tel que les sommets gris correspondent à des points d'articulation et les les lignes en gras correspondent aux isthme. Sachant que $\overrightarrow{G}(X_{TU}, E_{TU})$ contient une unique cc contenant des T-sommets, la pré-condition du théorème 3 est satisfaite et on obtient le filtrage suivant :*

- *Puisque la cc de $\overrightarrow{G}(X_{TU}, E_{TU})$ correspondant aux U-sommets 4 et 9 n'appartient pas à $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$, d'après l'item (1.) les U-sommets 4 et 9 et les U-arcs $(4,4)$ et $(9,4)$ sont respectivement transformés en F-sommets et F-arcs.*
- *D'après l'item (2.), les deux U-sommets 7 et 8, qui sont des points d'articulation de $\overleftrightarrow{G}(X_{TU}, E_{TU})$ appartenant à une chaîne élémentaire entre deux T-sommets (3 et 6), peuvent être transformés en T-sommets.*
- *D'après l'item (3.), parmi les 3 isthme de $\overleftrightarrow{G}(X_{TU}, E_{TU})$ appartenant à une chaîne élémentaire entre deux T-sommets (3 et 6), $(3,8)$ et $(7,6)$ peuvent être transformés en T-arcs puisque leur inverses respectifs $(8,3)$ et $(6,7)$ n'appartiennent pas à $\overrightarrow{G}(X_{TU}, E_{TU})$.*

4.4 Filtrage en fonction de $\overline{\mathrm{NCC}}$

$\overline{\mathbf{NCC}}$ est égal au nombre courant de ccde $\overrightarrow{G}(X_T, E_T)$ contenant au moins un T-arc (c'est à dire, $|cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))|$) plus la cardinalité d'un couplage minimal $\mu_l(\overleftrightarrow{G}_{rem})$, qui est le nombre maximal possible de nouvelles cc pouvant être présentes dans un digraphe final issu de $\overrightarrow{G}(X_{TU}, E_{TU})$, ajouté à $|cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))|$. Tous les U-arcs (et U-sommets) qui réduiraient le nombre de cc si elles appartenaient au digraphe final peuvent être transformées en F-arcs (et F-sommets).

Théorème 4 *Si $dom(\mathbf{NCC}) = \{\overline{\mathbf{NCC}}\}$*

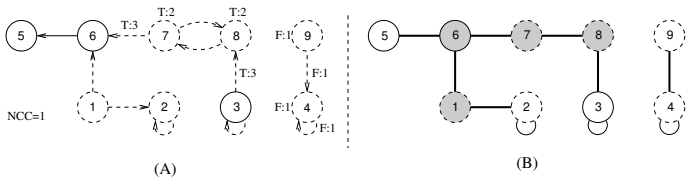

FIG. 3 – Filtrage en fonction de **NCC** : illustration du théorème 3.

1. *Tous les U-arc de $\overrightarrow{G}(X_T, E_{TU})$ joignant deux T-sommets qui appartiennent à deux cc distinctes de $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$ peuvent être transformés en F-arcs.*
2. *Pour toute arête de $\overleftrightarrow{G}_{rem}$ n'appartenant pas à un l-couplage de $\overleftrightarrow{G}_{rem}$, le ou les U-arc(s) correspondants peuvent être transformés en F-arcs.*
3. *Tout U-arc $e = (u, v)$ tel que u appartienne à une cc de $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$ et v soit saturé dans tout l-couplage maximal de $\overleftrightarrow{G}_{rem}$ peut être transformé en F-arc.*
4. *Tout U-sommet de $\overleftrightarrow{G}_{rem}$ appartenant à tous les l-couplages maximaux de $\overleftrightarrow{G}_{rem}$ peut être transformé en T-sommet.*
5. *Pour toutes les arêtes e appartenant à tous les l-couplages maximaux de $\overleftrightarrow{G}_{rem}$, si un unique U-arc de $\overrightarrow{G}(X_{TU}, E_U)$ correspond à e alors cet arc peut être transformé en T-arc.*
6. *Tout U-sommet de $\overleftrightarrow{G}_{rem}$ qui n'appartient à aucun l-couplage maximal de $\overleftrightarrow{G}_{rem}$ peut être transformé en F-sommet*

Exemple 6 *La partie (A) de la figure 4 illustre le théorème 4 en posant l'hypothèse que le digraphe final doit contenir au moins 6 cc. $cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))$ consiste en les deux ccsuivantes, respectivement constituées des ensembles de sommets $\{2, 3\}$ et $\{4, 5, 6\}$. La partie (B) illustre les graphe non orienté $\overleftrightarrow{G}_{rem}$, tel que les lignes en gras correspondent à un l-couplage de cardinalité 4, et les sommets gris sont les sommets qui sont saturés dans tout l-couplage maximal. Sachant que la précondition $\mathbf{NCC} = |cc_{[|E_T|\geq 1]}(\overrightarrow{G}(X_T, E_T))| + \mu_l(\overleftrightarrow{G}_{rem}) = 6$ du théorème 4 est satisfaite, les items (1.), (2.) et (3.) transforment respectivement les U-arcs de $\{(4, 3)\}$, de $\{(9, 7), (9, 10), (10, 9)\}$ et de $\{(4, 8), (7, 5)\}$ en F-arcs. L'item (4.) transforme les U-sommets $\{8, 13\}$ en T-sommets. Enfin, l'tem (5.) transforme les U-arcs $\{(7, 8), (9, 9)\}$ en T-arcs.*

4.5 Complexité

La table 2 fournit les résultats de complexité pour les conditions de faisabilité (consistance de la contrainte) ainsi que pour chacun des items des théorèmes introduits précédamment. La plupart de ces items correspondent directement à un problème de la théorie des graphes connu, qui est mentionné dans la troisième colonne de la table. La complexité établie pour chaque item d'un théorème suppose que la condition de faisabilité a été préalablement calculée et testée : par exemple, en supposant que la cardinalité d'un couplage maximal ait déjà été calculée, identifie les sommets qui sont saturés dans tous les couplage maximaux est linéaire dans le nombre d'arêtes du graphe [5].

Théorèmes	Complexité	Problèmes de graphe
Théorème 1		
• Faisabilité	NP	*cardinalité d'un hitting set maximal* [9]
• Item 1	$O(m)$	*itérartion sur les arcs*
• Item 2,4	NP	*identification de sommets n'appartenant à aucun hitting set minimal*
• Item 3	NP	*identification de sommets appartenant à tous les hitting set minimaux*
Corollaire 1		
• Faisabilité	$O(n)$	*itérartion sur les sommets*
• Item 1	$O(n)$	*itérartion sur les sommets*
Théorème 2		
• Faisabilité	$O(n)$	*itérartion sur les sommets*
• Item 1	$O(n)$	*itérartion sur les sommets*
Théorème 3		
• Faisabilité	$O(n)$	*itérartion sur les sommets*
• Item 1	$O(m)$	*itérartion sur les arcs*
• Item 2,3	$O(m)$	*parcours en profondeur*
Théorème 4		
• Faisabilité	$O(m\sqrt{n})$	*couplage maximal* [11]
• Item 1	$O(m)$	*calcul des cc*
• Items 2,5,6	$O(m \cdot n)$	*identification des arêtes n'appartenant pas à un couplage maximal* [13]
• Items 3,4	$O(m)$	*identification des sommets saturés dans tout couplage maximal* [5]

TAB. 2 – Complexité de chaque théorème.

5 Lien avec les algorithmes ad-hoc

On pourait s'interroger sur le fait que la technique de filtrage basé sur les graphes exposée dans la section précédente soit plutôt théorique, et par conséquent difficile à appliquer en pratique. Dans cette section, nous montrons comment reconstruire rationnellement des algorithmes ad-hoc qui avaient été proposés pour des contraintes globales spécifiques. Nous considérons la contrainte `proper_forest` préalablement introduite dans l'exemple 1 et pour laquelle un algorithme de filtrage spécialisé a été récemment proposé [5]. Après avoir rappelé les différentes étapes de cet algorithme, nous "déconstruisons" cet algorithme et réinterprêtons ses différentes étapes en terme de filtrage générique basé sur les graphes. L'algorithme spécialisé de la contrainte `proper_forest` est constitué des étapes suivantes :

1. Renormaliser $\overrightarrow{G}(X_{TU}, E_{TU})$ en fonction du fait que le digraphe doit être symétrique.
2. Tester la faisabilité de la contrainte `proper_forest` :
   (a) Le digraphe intermédiaire n'a pas de sommet isolé.
   (b) Il n'y a pas de cycle formé de T-arcs.

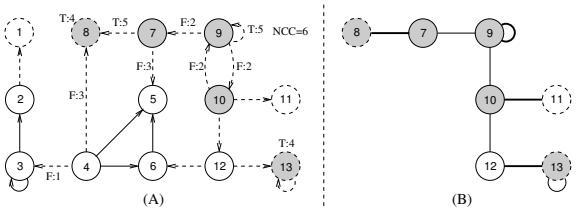

FIG. 4 – Filtrage en fonction de $\overline{\mathbf{NCC}}$: illustration du théorème 4.

(c) NTREE a au moins une valeur dans [MINTREE, MAXTREE] où MINTREE est le nombre de cc du digraphe intermédiaire et MAXTREE est le nombre de cc de $\overrightarrow{G}(X_T, E_T)$ plus la cardinalité d'un couplage maximal dans le sous-graphe induit par les sommets qui ne sont pas liés à des T-sommets.

3. Chaque U-arc qui pourrait créer un cycle de T-sommets, est modifié en F-arc.
4. Les valeurs minimales et maximales de NTREE sont respectivement ajustées à MINTREE et MAXTREE.
5. Quand NTREE est fixée à MINTREE tous les U-arcs correspondant à des isthmes $\overrightarrow{G}(X_{TU}, E_{TU})$ sont modifiés en T-arcs.
6. Quand NTREE est fixé à MAXTREE chaque U-arc (u, v) satisfaisant une des conditions suivantes est modifié en F-arc :
   (a) u et v appartiennent tous les deux à deux cc distinctes de $\overrightarrow{G}(X_T, E_T)$ contenant plus d'un sommet.
   (b) (u, v) n'appartient à aucun couplage maximal dans le sous-graphe induit par les sommets qui ne sont pas liés à des T-sommets.
   (c) u est l'extrémité d'un T-arc et v est saturé dans chaque couplage maximal du sous-graphe induit par les sommets qui ne sont pas liés à des T-sommets.
7. Chaque U-arc dont une extrémité est une source ou un puits est modifié en T-arc.

En considérant la reformulation basée sur les graphes de la proposition 1 sur la propriété de graphe NTREE $= \mathbf{NCC}$ on peut retrouver presque toutes les étapes de l'algorithme précédent, à l'exception des étapes 2(b) et 3, qui sont issues de l'invariant $\mathbf{NARC} = 2 \cdot (n - \mathbf{NCC})$ liant les deux paramètres de graphes $\mathbf{NARC}$ et $\mathbf{NCC}$) :

– L'item (1.) correspond la propagation des contraintes de classes de graphes (5) en fonction de la classe de graphe c of C (dans le contexte de proper_forest, le digraphe final doit être symmétrique).
– L'item (2.a) correspond à la propagation des contraintes de normalisation (2).
– L'item (2.c) correspond au test des contraintes de propriété de graphe $\mathbf{NCC} =$ NTREE (4). MINTREE et MAXTREE correspondent respectivement aux bornes inférieures et supérieures générales de $\mathbf{NCC}$ données dans le table 1 et déduites de la contrainte (3).
– L'item (4.) correspond à la propagation induite par la contrainte de propriété de graphe (4).
– Les items (5.) et (6.) correspondent à la propagation de la contrainte (3) sur le paramètre $\mathbf{NCC}$: l'item (5.) est la spécialisation du théorème 3, plus précisément son troisième item (les deux premiers items du théorème 3 ne sont pas significatifs car $\overrightarrow{G}(X_{TU}, E_{TU})$ ne contient pas de U-sommet). L'item (6.) correspond au théorème 4.
– Enfin, l'item (7.) correspond à la propagation obtenue par la contrainte de classe de graphe (5), qui évite la création de sommets isolés dans le graphe symétrique (cf. exemple 2).

6 Spécialisation du filtrage

Il arrive souvent que le digraphe final d'une contrainte globale ait une structure régulière héritée de son digraphe initial ou bien issue de certaines propriétés de la contrainte d'arc $\mathcal{R}$. Cela se traduit par des contraintes élémentaires supplémentaires dans la reformulation des contraintes globales basée sur les graphes, les *contraintes de classe de graphes* (5). Propager ces contraintes avant d'évaluer les bornes des paramètres dans le digraphe intermédiaire permet de raffiner les formules de la table 1 et le filtrage basé sur les bornes (section 4), autant en terme de réalisabilité des bornes qu'en terme de complexité algorithmique. Cette section illustre ce principe sur la classe de graphes path_with_loops pour les quatre paramètres $\mathbf{NVERTEX}$, $\mathbf{NCC}$, $\mathbf{MIN\_NCC}$, et $\mathbf{MAX\_NCC}$. Le filtrage est alors expérimenté sur l'exemple de la contrainte group qui appartient à la classe de graphes path_with_loops , et dans la description de laquelle ces quatre paramètres interviennent.

6.1 La classe de graphes path_with_loops

La classe path_with_loops groupe ensemble les contraintes globales qui ont la description suivante :

– Le digraphe initial utilise exclusivement les générateurs d'arcs $PATH$ et $LOOP$. Cela fournit une séquence de sommets $X = \{x_1, \ldots, x_n\}$ avec un arc $(x_j, x_{j+1}) \in E_{\mathcal{R}}$, $j \in \{1, \ldots, n-1\}$, pour chaque paire de sommets consécutifs, et une boucle $(x_j, x_j) \in E_{\mathcal{R}}$, $j \in \{1, \ldots, n\}$, sur chaque sommet (cf. la partie (A) de la figure 5).

– Dans tout digraphe final, chaque sommet restant a une boucle et deux sommets directement consécutifs doivent être liés par un arc (cf. la partie (B) de la figure 5). Ces conditions correspondent aux contraintes de classe de graphes suivantes dans la reformulation de la proposition 1 :

$$vertex_j = arc_{j,j} \quad (6)$$
$$\min(vertex_j, vertex_{j+1}) = arc_{j,j+1} \quad (7)$$

Parmi les contraintes globales appartenant à la classe de graphe `path_with_loops` le catalogue mentionne par exemple `group` [7] et `stretch` [12].

Exemple 7 *Soit la contrainte* `group`(`NGROUP`, `MIN_SIZE`, `MAX_SIZE`, `MIN_DIST`, `MAX_DIST`, `NVAL`, `VARIABLES`, `VALUES`) *telle que les six premiers paramètres sont des variables, tandis que* `VARIABLES` *est une séquence de* n *variables et* `VALUES` *un ensemble finit d'entiers.*

Soit $x_i, x_{i+1}, \ldots, x_j$ $(1 \leq i \leq j \leq n)$ *des variables consécutives dans la séquence* `VARIABLES` *telles que toutes les conditions suivantes soient simultannément vérifiées : (1) toutes les variables* $x_i, \ldots, x_j$ *prennent leur valeur dans l'ensemble de valeurs* `VALUES`*, (2) soit* $i = 1$*, ou* x_{i-1} *ne prennent pas de valeur dans* `VALUES`*, (3) soit* $j = n$*, ou* x_{j+1} *ne prennent pas de valeur dans* `VALUES`*. Nous appelons un tel ensemble de variables un* group*. La contrainte* `group` *est remplie si toutes les conditions suivantes sont vérifiées :*

– *il y a exactement* `NGROUP` *groupes de variables,*
– `MIN_SIZE` *et* `MAX_SIZE` *représentent le nombre de variables du plus petit et du plus grand groupe.*
– `MIN_DIST` *et* `MAX_DIST` *représentent le nombre de variables minimal et maximal entre deux groupes consécutifs ou entre un bord et un groupe.*
– `NVAL` *est le nombre de variables prenant leur valeur dans l'ensemble* `VALUES`*.*

Par exemple, `group`$(2, 2, 4, 1, 2, 6, \langle 0, 0, 1, 3, 0, 2, 2, 2, 3\rangle, \{1, 2, 3\})$ *est satisfaite car la séquence* $\langle 0, 0, 1, 3, 0, 2, 2, 2, 3\rangle$ *contient* 2 *groupes* $\langle 1, 3\rangle$ *et* $\langle 2, 2, 2, 3\rangle$ *de valeur non nulles de taille* 2 *et* 4*,* 2 *groupes* $\langle 0, 0\rangle$ *et* $\langle 0\rangle$ *de zéros, et* 6 *valeurs non nulles. La description basée sur les graphes de la contrainte* `group` *utilise deux graphes de contraintes qui mentionnent respectivement les propriétés de graphes* $\mathbf{NCC}$ = `NGROUP`*,* $\mathbf{MIN\_NCC}$ = `MIN_SIZE`*,* $\mathbf{MAX\_NCC}$ = `MAX_SIZE`*,* $\mathbf{NVERTEX}$ = `NVAL` *et* $\mathbf{MIN\_NCC}$ = `MIN_DIST`*,* $\mathbf{MAX\_NCC}$ = `MAX_DIST`*. La figure 5 illustre le digraphe initial ainsi que les digraphes finaux associés aux deux contraintes de graphes de l'exemple donné pour la contrainte* `group`*.*

6.2 Filtrage pour la classe de graphes `path_with_loops`

La propriété `path_with_loops` met bien en évidence l'intérêt de spécialiser les formules des bornes de paramètres et les règles de filtrage. Tout d'abord, dans ce contexte, la structure de chemin des digraphes rend naturellement polynomiaux les différents algorithmes. L'état des sommets et des arcs peut être déterminé et fixé pendant le filtrage en temps linéaire, uniquement en parcourant le chemin du sommet x_1 au sommet x_n. En outre, des bornes qui n'étaient pas réalisables dans le cas général le deviennent dans ce contexte à cause des contraintes de classe de graphes. On obtient ainsi à l'aide de ces bornes un filtrage plus complet. Considérons par exemple la borne $\overline{\mathbf{NVERTEX}} = |X_{TU}|$. Dans le cas général, il existe un digraphe final avec un nombre de sommets égal à $|X_{TU}|$ car tous les U-sommets du digraphe intermédiaire peuvent être transformés en T-sommets. Puisque les contraintes (7) interdisent que deux U-sommets liés par un F-arc soient simultannément modifiés en T-sommets, la borne $\overline{\mathbf{NVERTEX}}$ peut être raffinée dans le contexte de `path_with_loops` à $|X_{TU}| - \sum_{i \in cc(\overrightarrow{G}(X_U, E_F))} \lfloor \frac{|vertex(i)|}{2} \rfloor$. Cela signifie que dans tous les sous-graphes (chemins) consitués de U-vertices et F-arcs, uniquement un sommet sur deux peut appartenir à un digraphe final maximisant $\mathbf{NVERTEX}$. Enfin, les contraintes de classe de graphes permettent de simplifier certaines bornes qui étaient déjà réalisables. Par exemple, la borne générale $\underline{\mathbf{NVERTEX}} = |X_T| + h(\overleftrightarrow{G}((X_{T,\neg T,\neg T}, X_{U,\neg T,T}), E_{U,T}))$ reste (naturellement) correcte et réalisable pour la classe de graphe `path_with_loops` . Mais ici, $X_{T,\neg T,\neg T}$ est vide. En effet, les contraintes (6) imposent que chaque T-sommet soit une extrémité d'au moins un T-arc (sa boucle). Aussi, la formule se simplifie en $\underline{\mathbf{NVERTEX}} = |X_T|$, en supprimant le terme relatif au calcul d'un hitting set minimal. De tels arguments restent valables pour les trois autres paramètres impliqués dans la description de la contrainte `group`. La table 3[7] résume les bornes de ces paramètres dans la classe `path_with_loops` . On note que toutes ces bornes peuvent être évaluées en $O(n)$ en irérant sur les n sommets du digraphe initial. De plus, toutes les bornes sont réalisables lorsqu'on prend en compte les contraintes de classe de graphes. Comme dans le cas général, elles sont dérivées en considérant les digraphes finaux qui minimisent ou maximisent le paramètre correspondant. À nouveau, identifier les U-arcs et les U-sommets appartenant ou pas à ces digraphes produit des règles de filtrage qui s'appliquent au digraphe intermédiaire quand une borne est atteinte. Par exemple, chaque digraphe final G_f ayant exactement $\overline{\mathbf{NVERTEX}}$ sommets peut être définie d'après le digraphe intermédiaire de la façon suivante : d'une part tout U-sommet de $X_{U,\neg F}$ appartient à G_f ; d'autre part si une cc $\mathcal{C}$ de $\overrightarrow{G}(X_U, E_F)$ possède un nombre de sommets impair $x_{p+1}, x_{p+2}, \ldots, x_{p+l_\mathcal{C}}$, alors seuls les sommets $x_{p+2\cdot i-1}$ $(i \in [1, \lceil \frac{l_\mathcal{C}}{2} \rceil])$ dans $\mathcal{C}$ appartiennent à G_f. Filtrer lorsque la condition $dom(\mathbf{NVERTEX}) = \{\overline{\mathbf{NVERTEX}}\}$ est satisfaite consiste à modifier les U-sommets pré-cités du di-

[7] Par convention, dans ces formules, une valeur maximale sur un ensemble vide est zéro. Dans le formule relative à $\overline{\mathbf{MIN\_NCC}}$, $\epsilon = 1$ s'il existe deux cc adjacentes (*i.e.,* liées par un F-arc) de taille minimale dans $cc_{[|X_T| \geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))$, et $\epsilon = 0$ dans le cas contraire.

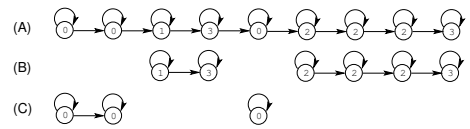

FIG. 5 – (A) digraphe initial et (B,C) digraphes finaux de la contrainte group.

graphe intermédiaire en T-sommets et tous les autres sommets d'une cc de $\mathcal{C}$ en F-sommets.

Exemple 8 *La figure 6 illustre le filtrage en posant l'hypothèse que le digraphe final doive contenir* 8 *sommets. Les deux cc de* $\overrightarrow{G}(X_U, E_F)$ *correspondent respectivement les ensembles de sommets* $\{3,4,5,6,7\}$ *et* $\{9,10,11,12\}$*. La borne* $\overline{\mathbf{NVERTEX}}$ *et alors égale à* $12 - \lfloor \frac{5}{2} \rfloor - \lfloor \frac{4}{2} \rfloor = 8$*. D'une part, puisque la première cc contient un nombre de sommets impair, les sommets* $3,5,7$ *sont transformés en* T*-sommets et les sommets* $4,6$ *sont transformés en* F*-sommets. D'autre part, sachant que la seconde cc contient un nombre de sommets pairs, rien ne peut être déduit concernant leur état. Enfin, tous les sommets de* $X_{U,\neg F} = \{1,2,8\}$ *sont transormés en* T*-sommets.*

Nous avons dérivé des règles de filtrage similaires correspondant aux bornes de la table 3. Les détails de ces règles sont fournies dans [1]. On remarque qu'elles peuvent être implémentées en $O(n)$ en itérant une fois sur les n variables du digraphe initial.

Paramètre	Borne
$\underline{\mathbf{NVERTEX}}$	$\lvert X_T \rvert$
$\overline{\mathbf{NVERTEX}}$	$\lvert X_{TU} \rvert - \sum_{i \in cc(\overrightarrow{G}(X_U, E_F))} \lfloor \frac{\lvert vertex(i) \rvert}{2} \rfloor$
$\underline{\mathbf{NCC}}$	$\lvert cc_{[\lvert X_T \rvert \geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU})) \rvert$
$\overline{\mathbf{NCC}}$	$\lvert cc(\overrightarrow{G}(X_T, E_T)) \rvert +$ $\sum_{i \in cc_{[\lvert X_{U,U,T} \rvert = 0]}(\overrightarrow{G}(X_U, E_{UF}))} \lceil \frac{\lvert vertex(i) \rvert}{2} \rceil +$ $\sum_{i \in cc_{[\lvert X_{U,U,T} \rvert = 1]}(\overrightarrow{G}(X_U, E_{UF}))} \lfloor \frac{\lvert vertex(i) \rvert}{2} \rfloor +$ $\sum_{i \in cc_{[\lvert X_{U,U,T} \rvert = 2]}(\overrightarrow{G}(X_U, E_{UF}))} (\lceil \frac{\lvert vertex(i) \rvert}{2} \rceil - 1)$
$\underline{\mathbf{MIN\_NCC}}$ si $\lvert X_T \rvert = 0$	0
si $\lvert X_T \rvert \geq 1$ $\wedge \lvert X_{U,U,\neg T} \rvert \geq 1$	1
si $\lvert X_T \rvert \geq 1$ $\wedge \lvert X_{U,U,\neg T} \rvert = 0$	$\min_{i \in cc(\overrightarrow{G}(X_T, E_T))} \lvert vertex(i) \rvert$
$\overline{\mathbf{MIN\_NCC}}$ si $\lvert X_T \rvert \geq 1$	$\min_{i \in cc_{[\lvert X_T \rvert \geq 1]}(\overrightarrow{G}(X_{TU}, E_{TU}))} \lvert vertex(i) \rvert - \epsilon$
si $\lvert X_T \rvert = 0$	$\max_{i \in cc(\overrightarrow{G}(X_{TU}, E_{TU}))} \lvert vertex(i) \rvert$
$\underline{\mathbf{MAX\_NCC}}$	$\max_{i \in cc(\overrightarrow{G}(X_T, E_T))} \lvert vertex(i) \rvert$
$\overline{\mathbf{MAX\_NCC}}$	$\max_{i \in cc(\overrightarrow{G}(X_{TU}, E_{TU}))} \lvert vertex(i) \rvert$

TAB. 3 – Bornes des différents paramètres de graphes dans le contexte de la classe de graphes path_with_loops .

6.3 Performances

Afin d'évaluer l'efficacité du filtrage basé sur les graphes, nous avons réalisé deux séries d'expérimentations, en gérant des instances aléatoires de la contrainte group. Nous avons choisi VARIABLES comme une séquence de m variables définies sur un domaine $[0,1]$, VALUES comme étant l'ensemble singleton $\{1\}$, et CTR étant $=$. Une instance de la contrainte est alors générée en fixant le domaine initial de chaque variable à un intervalle généré aléatoirement. En outre, avec 10% de probabilité, chaque variable de VARIABLES a été fixée aléatoirement. Les expérimentations comparent l'effet du filtrage basé sur les graphes avec l'approche consistant à construire un automate pour chaque paramètre de graphe et en reformulant ensuite cet automate en une conjonction de contraintes, décrite dans [2]. Dans la suite nous appelons cette dernière approche le filtrage basé sur des automates. Pour les deux méthodes, des invariants de graphes, qui fournissent des contraintes supplémentaires améliorant la résolution [4], ont été utilisés. Dans la première expérimentation, nous avons calculé le nombre de choix effectués pendant la recherche de toutes les solutions pour $m = 10$. Dans la seconde expérimentation nous avons calculé le nombre de choix effectués pendant la recherche pour obtenir la première solution, pour $m = 20$. Nous avons choisi de compter le nombre de points de choix plutôt que de mesurer le temps d'exécution, car une comparaison de temps équitable aurait nécessité une implémentation plus polie du filtrage basé sur les graphes que celle que nous avons utilisée. On notera cependant, afin d'affirmer la validité de cet argument, que toutes les règles de filtrage ont une complexité $O(m)$. Dans la figure 7, chaque point représente une instance aléatoire, son abscisse (resp. son ordonnée) correspond au filtrage basé sur les automates (resp. au filtrage basé sur les propriétés de graphes). Les instances satisfiables et insatisfiables sont distinguées sur la figure. Nous observons que la plupart du temps, mais parfois non, le filtrage basé sur les graphes domine celui basé sur les automates. On aurait pu s'attendre à une domination car la technique basée sur les graphes raisonne sur les variables d'arc ainsi que sur les variables de sommets. Notre méthode basée sur les graphes est actuellement limitée par le fait que le filtrage est déclenché uniquement lorsque un paramètre atteint une de ses bornes. Nous observons qu'il n'y a pas de différence significative entre les patterns des instances faisables et infaisables.

7 Conclusion

Dans cet article, nous avons fourni un premier schéma de filtrage générique issu des bornes inférieures et supérieures des paramètres de graphes usuels, utilisés dans la reformulation basée sur les graphes des contraintes glo-

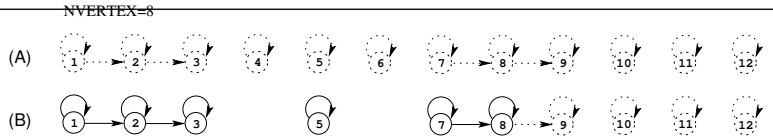

FIG. 6 – Digraphe initial (A) et digraphes finaux (B) en filtrant d'après $dom(\mathbf{NVERTEX}) = 8$.

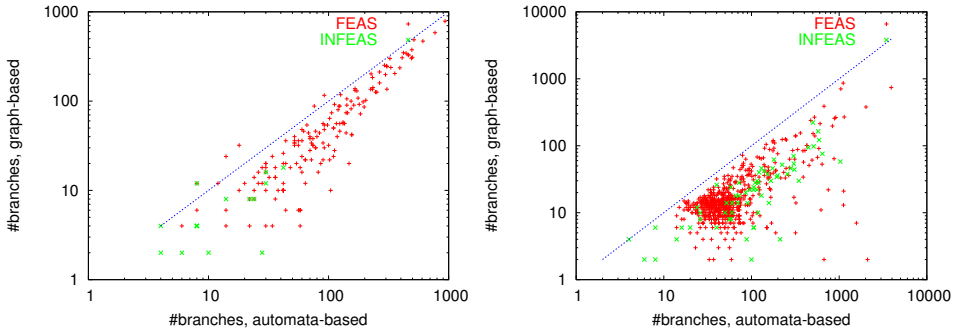

FIG. 7 – Diagrammes de points d'instances aléatoires. Gauche : comparaison du nombre de choix pour trouver toutes les solutions. Droite : comparaison du nombre de choix pour trouver la première solution.

bales. De plus, l'article montre comment on peut reconstruire la plupart des étapes d'un algorithme de filtrage spécialisé existant, uniquement à partir de la description basée sur les graphes. Nos expérimentations sur l'exemple de la classe de graphes `path_with_loops`mettent en relief une amélioration à apporter à l'approche : filtrer avant qu'un paramètre atteigne une de ses bornes.

Références

[1] N. Beldiceanu, M. Carlsson, S. Demassey, and T. Petit. Graph properties based filtering. Technical report, Swedish Institute of Computer Science, 2006. Preprint available at ftp ://ftp.sics.se/pub/isl/papers/filtering06.pdf.

[2] N. Beldiceanu, M. Carlsson, and T. Petit. Deriving Filtering Algorithms from Constraint Checkers. In M. Wallace, editor, *Principles and Practice of Constraint Programming (CP'2004)*, volume 3258 of *LNCS*, pages 107–122. Springer-Verlag, 2004.

[3] N. Beldiceanu, M. Carlsson, and J.-X. Rampon. Global constraint catalog. Technical Report T2005-06, Swedish Institute of Computer Science, 2005.

[4] N. Beldiceanu, M. Carlsson, J.-X. Rampon, and C. Truchet. Graph Invariants as Necessary Conditions for Global Constraints. In P. van Beek, editor, *Principles and Practice of Constraint Programming (CP'2005)*, volume 3709 of *LNCS*, pages 92–106. Springer-Verlag, 2005. Preprint available as SICS Tech Report T2005-07.

[5] N. Beldiceanu, I. Katriel, and X. Lorca. Undirected Forest Constraints. In *CP-AI-OR'06*, volume 3990 of *LNCS*, pages 29–43, 2006. Available at http ://www.daimi.au.dk/ irit/pub/forest.ps.

[6] N. Beldiceanu, T. Petit, and G. Rochart. Bounds of Graph Characteristics. In P. van Beek, editor, *CP'2005*, volume 3709 of *LNCS*, pages 742–746, 2005.

[7] COSYTEC. *CHIP Reference Manual*, release 5.1 edition, 1997.

[8] G. Dooms, Y. Deville, and P. Dupont. CP(Graph) : Introducing a Graph Computation Domain in Constraint. In P. van Beek, editor, *Principles and Practice of Constraint Programming (CP'2005)*, volume 3709 of *LNCS*, pages 211–225. Springer-Verlag, 2005.

[9] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W.H.Freeman and co., San Francisco, 1979.

[10] P. Martin and D. B. Shmoys. A New Approach to Computing Optimal Schedules for the Job-Shop Scheduling Problem. In *IPCO'96*, volume 1084 of *LNCS*, pages 389–403, 1996.

[11] S. Micali and V. V. Vazirani. An $\mathcal{O}(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs. In *FOCS 1980*, pages 17–27, New York, 1980. IEEE.

[12] G. Pesant. A Filtering Algorithm for the *stretch* Constraint. In *CP'2001*, volume 2239 of *LNCS*, pages 183–195, 2001.

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