Université de Montpellier - Master 2
Module **Contraintes**

Feuille TD 2 - 23/10/2023

Exercice 1

Soit la contrainte globale `ValueCount`$([X_1, \ldots, X_n], [Y_1, \ldots, Y_q])$ qui est satisfaite si et seulement si pour chaque valeur $v \in \{1, \ldots, q\}$ on a $Y_v = |\{i \in \{1, \ldots, n\} \mid X_i = v\}|$. Par exemple, $([2, 1, 5, 1, 2, 3, 1], [3, 2, 1])$ satisfait la contrainte `ValueCount` alors que $([1, 1, 1, 2, 3, 2], [2, 3, 1])$ ne la satisfait pas.

Question 1. Soit une contrainte globale `AllDifferent`$(X_1, \ldots, X_n)$ telle que l'union des domaines forme un intervalle $[1, \ldots, m]$. Montrez que l'on peut exprimer cette contrainte à l'aide d'une contrainte `ValueCount` portant sur un nombre de variables polynomial en $n + m$.

Correction. La contrainte `AllDifferent`$(X_1, \ldots, X_n)$ peut s'exprimer de manière équivalente par la condition "toute valeur $v \in \cup_{i \leq n} D(X_i)$ apparaît au plus une fois dans le vecteur $(X_1, \ldots, X_n)$". En utilisant la contrainte `ValueCount` et le fait que l'union des domaines forme un intervalle $[1, \ldots, m]$, on peut donc reformuler `AllDifferent`$(X_1, \ldots, X_n)$ par

$$\texttt{ValueCount}([X_1, \ldots, X_n], [Y_1, \ldots, Y_m])$$

avec pour tout $i \leq m$, $D(Y_i) = \{0, 1\}$.

Question 2. Justifiez brièvement que si la contrainte globale `AllDifferent` admet une AC-décomposition lorsque l'union des domaines des variables forme un intervalle $[1, \ldots, m]$, alors `AllDifferent` admet une AC-decomposition dans le cas général.

Correction. Soit `AllDifferent`$(X_1, \ldots, X_n)$ une contrainte (sans hypothèse sur les domaines), m le nombre de valeurs distinctes qui apparaissent dans les domaines de $X_1, \ldots, X_n$ et ϕ une bijection quelconque de $\cup_{i \leq n} D(X_i)$ dans $\{1, \ldots, m\}$. On a alors

$$\texttt{AllDifferent}(X_1, \ldots, X_n) \iff \texttt{AllDifferent}(\phi(X_1), \ldots, \phi(X_n))$$

où $\phi(X_i)$ désigne la variable obtenue à partir de X_i en remplaçant chaque $v \in D(X_i)$ par $\phi(v)$. Toute AC-décomposition de `AllDifferent`$(\phi(X_1), \ldots, \phi(X_n))$ (dont l'union des domaines forme par construction un intervalle) peut donc être transformée en AC-décomposition de `AllDifferent`$(X_1, \ldots, X_n)$ en remplaçant chaque valeur $v \in \{1, \ldots, m\}$ par $\phi^{-1}(v)$.

Question 3. Calculer l'arc cohérence sur la contrainte globale `ValueCount` est NP-difficile. Il n'existe donc pas de AC-décomposition pour `ValueCount` si P $\neq$ NP. S'il l'on suppose au contraire que P = NP, peut-on espérer que `ValueCount` admette une AC-décomposition ? Justifiez votre réponse.

Correction. En combinant les réponses aux questions précédentes :

`ValueCount` admet une AC-décomposition
  ⇒ `AllDifferent` admet une AC-décomposition lorsque l'union des domaines forme un intervalle (Q1)
  ⇒ `AllDifferent` admet une AC-décomposition dans le cas général (Q2)

ce qui est faux d'après le cours (indépendamment des classes P et NP). `ValueCount` n'admet donc pas d'AC-décomposition, même si P = NP.

Question 4. On a vu en cours que calculer l'arc cohérence sur la contrainte globale $\texttt{NValue}(X_1, \ldots, X_n, N)$ (qui est satisfaite si et seulement si $N = |\{X_1, \ldots, X_n\}|$) est NP-difficile. Supposons à nouveau que P = NP. Peut-on espérer que `NValue` admette une AC-décomposition ? Justifiez votre réponse.

Correction. Non car on peut exprimer `AllDifferent` avec une contrainte `NValue` :

$$\texttt{AllDifferent}(X_1, \ldots, X_n) \iff \texttt{NValue}(X_1, \ldots, X_n, N)$$

où la variable N a pour domaine $\{n\}$. `AllDifferent` n'est pas AC-décomposable, donc `NValue` ne l'est pas non plus, même si P = NP.

Question 5. Soit la contrainte globale $\texttt{MultisetEqual}([X_1, \ldots, X_n], [Y_1, \ldots, Y_n])$ qui est satisfaite si et seulement si les multisets $\{\{X_i \mid i \in \{1, \ldots, n\}\}\}$ et $\{\{Y_i \mid i \in \{1, \ldots, n\}\}\}$ sont égaux. (Un multiset diffère d'un ensemble par le fait qu'il tient compte du nombre d'occurences de chaque élément.) Par exemple $([1,1,1,2,3],[2,1,3,1,1])$ satisfait la contrainte `MultisetEqual` alors que $([1,1,1,2,3],[2,2,3,1,1])$ ne la satisfait pas. Contrairement à `ValueCount` et `NValue`, on sait que calculer l'arc cohérence est polynomial sur `MultisetEqual`. `MultisetEqual` admet-elle une AC-décomposition ? Justifiez votre réponse.

Correction. Si l'on considère n variables $X_1, \ldots, X_n$ telles que l'union des domaines forme un intervalle $\{1, \ldots, m\}$ avec $k = m - n \geq 0$, alors on a

$$\texttt{AllDifferent}(X_1, \ldots, X_n) \iff \texttt{MultisetEqual}([X_1, \ldots, X_n, Z_1, \ldots, Z_k], [Y_1, \ldots, Y_m])$$

où le domaine de chaque Z_i est $\{1, \ldots, m\}$ et le domaine de chaque Y_i est $\{i\}$. Par la réponse à la question 2, `AllDifferent` n'est pas AC-décomposable même lorsque l'union des domaines forme un intervalle, donc `MultisetEqual` ne l'est pas non plus.

Exercice 2

Soit le réseau de contraintes $N = (X, D, C)$, où $X = \{X_1, \ldots, X_5\}$, $D(X_1) = D(X_2) = \{0, 2, 4, 6\}$, $D(X_3) = \{2, 5, 6, 7\}$, $D(X_4) = \{0, 2, 6, 8, 12\}$, $D(X_5) = \{0, 1, 6\}$, $c_1 \equiv X_1 + X_2 + X_3 = X_4$, $c_2 \equiv |X_1 - X_3| = X_5$, $c_3 \equiv |X_1 - X_2| = 4$ et $C = \{c_1, c_2, c_3\}$.

Question 1. Appliquez BC sur N.

Correction. On examine les contraintes une par une, en supprimant les bornes des domaines qui n'ont pas de support BC. Attention, un support BC peut affecter certaines variables à des valeurs qui ne sont pas dans leur domaine ; il faut juste que la valeur se trouve entre les bornes !

Notation pour cet exercice : "$c_i : (X_j, v)$" signifie "on supprime la valeur $v \in D(X_j)$ car elle n'a pas de support BC dans la contrainte c_i".

$$c_1 : (X_4, 0)$$

Toutes les bornes restantes ont un support BC pour chaque contrainte. La fermeture BC de D est donc :

$$\begin{aligned}
&D(X_1) = D(X_2) = \{0, 2, 4, 6\}\\
&D(X_3) = \{2, 5, 6, 7\}\\
&D(X_4) = \{2, 6, 8, 12\}\\
&D(X_5) = \{0, 1, 6\}
\end{aligned}$$

Question 2. Appliquez AC sur N.

Correction. Avec les mêmes notations que dans le TD1 :

$$\begin{aligned}
&c_1 : (X_3, 5), (X_3, 7), (X_4, 0)\\
&c_2 : (X_5, 1), (X_1, 4)\\
&c_3 : (X_2, 0)\\
&c_1 : (X_4, 2)
\end{aligned}$$

Toutes les valeurs restantes ont un support pour chaque contrainte. La fermeture AC de D est donc :

$$\begin{aligned}
&D(X_1) = \{0, 2, 6\}\\
&D(X_2) = \{2, 4, 6\}\\
&D(X_3) = \{2, 6\}\\
&D(X_4) = \{6, 8, 12\}\\
&D(X_5) = \{0, 6\}
\end{aligned}$$

Question 3. Appliquez SAC sur N.

Correction. On part des domaines AC de la question 2 car toute valeur supprimée par AC l'est aussi par SAC. On fixe $D(X_1) = \{0\}$ et on applique AC :

[Sous l'hypothèse $D(X_1) = \{0\}$] $c_2 : (X_3, 2), (X_5, 0)$
[Sous l'hypothèse $D(X_1) = \{0\}$] $c_3 : (X_2, 2), (X_2, 6)$
[Sous l'hypothèse $D(X_1) = \{0\}$] c_1 : domaines vides pour X_1, X_2, X_3, X_4

On a au moins un domaine vide, donc SAC supprime 0 du domaine de X_1. On recommence en fixant cette fois-ci $D(X_1) = \{2\}$:

[Sous l'hypothèse $D(X_1) = \{2\}$] $c_3 : (X_2, 2), (X_2, 4)$
[Sous l'hypothèse $D(X_1) = \{2\}$] $c_2 : (X_3, 6), (X_5, 6)$
[Sous l'hypothèse $D(X_1) = \{2\}$] c_1 : domaines vides pour X_1, X_2, X_3, X_4

SAC supprime donc également 2 du domaine de X_1, qui devient alors le singleton $\{6\}$. Par AC on a finalement

$c_3 : (X_2, 4), (X_2, 6)$
$c_2 : (X_3, 2), (X_5, 6)$
c_1 : domaines vides pour X_1, X_2, X_3, X_4

et après SAC tous les domaines sont donc vides : $\forall i \leq 5, D(X_i) = \emptyset$.

Soit N^* le réseau de contraintes obtenu à partir de N en fixant les domaines à $D(X_1) = D(X_2) = \{0,\dots,6\}$, $D(X_3) = \{2,\dots,7\}$, $D(X_4) = \{0,\dots,12\}$, $D(X_5) = \{0,\dots,6\}$.

Question 4. Appliquez AC et BC sur N^*. Comparez AC et BC sur les réseaux de contraintes dont les domaines sont des intervalles de $\mathbb{Z}$.

Correction. La fermeture BC de N^* est

$$\begin{aligned} D(X_1) &= \{0,\dots,6\} \\ D(X_2) &= \{0,\dots,6\} \\ D(X_3) &= \{2,\dots,7\} \\ D(X_4) &= \{2,\dots,12\} \\ D(X_5) &= \{0,\dots,6\} \end{aligned}$$

et la fermeture AC de N^* est presque identique, à l'exception près que la valeur 3 est enlevée des domaines de X_1 et X_2 à cause de la contrainte c_3. AC reste donc une propriété strictement plus forte que BC, même lorsque les domaines sont des intervalles de $\mathbb{Z}$.

Exercice 3

Pour $q \geq k \geq 1$, on considère la contrainte globale `(k,q)-ConsecutiveOnes`$([X_1,\dots,X_n])$ qui porte sur n variables booléennes $X_1,\dots,X_n$. Cette contrainte est satisfaite si et seulement si chaque séquence maximale (non vide) de 1 consécutifs dans le vecteur $(X_1,\dots,X_n)$ est de longueur au moins k et au plus q. Par exemple, $([0,1,1,0,1,1,1])$ satisfait `(2,3)-ConsecutiveOnes` mais $([0,1,0,0,1,1,1])$ ne la satisfait pas.

Question 1. Montrez que pour tout $q \geq k \geq 1$, la contrainte globale `(k,q)-ConsecutiveOnes` admet une AC-décomposition.

Correction. On va construire une AC-décomposition de `(k,q)-ConsecutiveOnes`$(X_1,\dots,X_n)$ en s'inspirant de celle de `AtLeast-p-v` vue en cours. On ajoute n variables $C_1,\dots,C_n$ de domaine $\{0,\dots,q\}$, dont le but va être de compter le nombre de 1 consécutifs dans les séquences de $X_1,\dots,X_n$. On fixe $X_1 = C_1$, et pour chaque $i \in \{2,\dots,n\}$ on impose une contrainte :

$$c(C_{i-1}, C_i, X_i) \equiv ((X_i = 1) \wedge (C_i = C_{i-1} + 1)) \vee ((X_i = 0) \wedge (C_i = 0) \wedge (C_{i-1} \in \{0, k, \dots, q\}))$$

Le sens de cette contrainte est assez intuitif. Si $X_i = 1$, alors le compteur pour la séquence de 1 courante est incrémenté. Si $X_i = 0$, le compteur est fixé à zéro et on s'assure que C_{i-1} (qui contient alors le nombre de 1 consécutifs dans la séquence maximale de 1 qui se termine en X_{i-1}) est soit 0 soit dans l'intervalle $[k,\dots,q]$. Il reste alors à vérifier la séquence qui se termine en X_n, et pour cela on fixe $D(C_n) = \{0, k, \dots, q\}$.

La conjonction de ces contraintes est équivalente à `(k,q)-ConsecutiveOnes`$(X_1,\dots,X_n)$, le réseau est de taille polynomiale, l'arité est bornée par 3 et le graphe d'incidence est Berge-acyclique : c'est donc une AC-décomposition.

La contrainte globale `(k,q)-ConsecutiveValues`$([X_1,\dots,X_n])$ porte sur n variables entières et est satisfaite si et seulement si chaque séquence maximale (non vide) de valeurs identiques consécutives est de longueur au moins k et au plus q.

Question 2. La contrainte globale `(k,q)-ConsecutiveValues` admet-elle une AC-décomposition ? Justifiez votre réponse.

Correction. Oui. On construit une AC-décomposition de `(k,q)-ConsecutiveValues`$(X_1, \ldots, X_n)$ dans le même esprit que pour la question précédente. Soit S l'union des domaines de $X_1, \ldots, X_n$. On ajoute n variables $C_1, \ldots, C_n$ de domaine $S \times \{1, \ldots, q\}$, qui vont garder en mémoire la dernière valeur observée et la longueur de la séquence courante. On remarquera que chaque valeur du domaine de C_1 est une paire (v, l). Pour chaque $i \in \{2, \ldots, n\}$ on impose une contrainte $c(C_{i-1}, C_i, X_i)$ définie comme suit :

$$((v_1, l_1), (v_2, l_2), v) \in c \iff \begin{cases} (v_1 = v_2 = v) \wedge (l_2 = l_1 + 1), \text{ ou} \\ (v_1 \neq v_2 = v) \wedge (l_2 = 1) \wedge (l_1 \in \{k, \ldots, q\}) \end{cases}$$

Tout d'abord, on observe que $X_i = v$ implique que C_i doit être affecté à une paire ayant v pour premier élément (reflété par le fait que $v_2 = v$ dans tous les tuples de c). Les tuples de $c(C_{i-1}, C_i, X_i)$ sont divisés en deux groupes. Dans le premier, X_i est égal à X_{i-1} et le compteur l est incrémenté. Dans le second, X_i est différent de X_{i-1} et le compteur est remis à 1 ; cela arrive à la fin de chaque séquence (sauf la dernière) et donc on vérifie que sa longueur est bien dans l'intervalle $\{k, \ldots, q\}$.

Il reste à ajouter des contraintes pour initialiser C_1 et vérifier la longueur de la dernière séquence. On impose donc $C_1 = (X_1, 1)$ et $D(C_n) = S \times \{k, \ldots, q\}$. Pour les mêmes raisons que la question précédente, ce réseau de contraintes est une AC-décomposition de `(k,q)-ConsecutiveValues`$(X_1, \ldots, X_n)$.

La contrainte globale `DifferentConsecutiveLengths`$([X_1, \ldots, X_n])$ porte sur n variables entières et est satisfaite si et seulement si pour toute valeur v, il n'existe pas deux séquences maximales (non vides) de v consécutifs qui ont la même longueur.

Question 3. La contrainte globale `DifferentConsecutiveLengths` admet-elle une AC-décomposition ? Justifiez votre réponse.

Correction. Non car on peut exprimer la contrainte `AllDifferent`$(X_1, \ldots, X_n)$ à partir de cette contrainte :

$$\texttt{AllDifferent}(X_1, \ldots, X_n) \iff \texttt{DifferentConsecutiveLengths}(X_1, Z_1, X_2, Z_2, \ldots, Z_{n-1}, X_n)$$

où le domaine de chaque variable Z_i est composé d'une valeur quelconque qui n'apparaît dans aucun autre domaine. `AllDifferent` n'est pas AC-décomposable, donc `DifferentConsecutiveLengths` ne l'est pas non plus.

Question 4. Les réponses aux questions 2 et 3 vous permettent-elles de déterminer si l'arc cohérence sur ces deux contraintes est calculable en temps polynomial ?

Correction. Par la question 2, on peut déduire que l'AC sur `(k,q)-ConsecutiveValues` est calculable en temps polynomial. Pour `DifferentConsecutiveLengths`, la réponse à la question 3 ne permet pas de conclure car il existe des contraintes pour lesquelles on peut calculer l'AC en temps polynomial mais qui ne sont pas AC-décomposables (e.g. `AllDifferent`).

Exercice 4

Dans un réseau de contraintes $N = (X, D, C)$, on dit qu'une valeur $v \in D(X_i)$ est *singleton viable* si le réseau de contraintes obtenu à partir de N' en fixant $D(X_i) = \{v\}$ a une fermeture arc cohérente non vide. Un réseau est *singleton arc cohérent* (SAC) si toutes les valeurs sont singleton viables ; "appliquer SAC" désigne le processus d'enlever des valeurs qui ne sont pas singleton viables jusqu'à ce que toutes le soient.

Soit le réseau de contraintes $N^* = (X, D, C)$, où $X = \{X_1, \ldots, X_4\}$, $D(X_1) = D(X_2) = \{1, 2, 7, 8\}$, $D(X_3) = D(X_4) = \{1, 2, 3, 4\}$, $c_1 \equiv |X_1 - X_2| = |X_3 - X_4|$, $c_2 \equiv X_3 = 2X_2$, $c_3 \equiv X_1 + X_4 = 5$ et $C = \{c_1, c_2, c_3\}$.

Question 1. Appliquez BC sur N^*.

Correction.

$$\begin{aligned} &c_2 : (X_3, 1), (X_2, 7), (X_2, 8) \\ &c_3 : (X_1, 7), (X_1, 8), (X_4, 1), (X_4, 2) \end{aligned}$$

Toutes les bornes restantes ont un support BC pour chaque contrainte. La fermeture BC de D est donc :

$$\begin{aligned} &D(X_1) = D(X_2) = \{1, 2\} \\ &D(X_3) = \{2, 3, 4\} \\ &D(X_4) = \{3, 4\} \end{aligned}$$

Question 2. Appliquez SAC sur N^*.

Correction. On part des domaines BC de la question 1. La valeur $3 \in D(X_3)$ n'est pas viable pour la contrainte c_2, donc elle n'est pas singleton viable non plus. On fixe $D(X_1) = \{1\}$ et on applique AC :

$$\begin{aligned} &[\text{Sous l'hypothèse } D(X_1) = \{1\}]\ c_3 : (X_4, 3) \\ &[\text{Sous l'hypothèse } D(X_1) = \{1\}]\ c_1 : (X_3, 2), (X_2, 2) \\ &[\text{Sous l'hypothèse } D(X_1) = \{1\}]\ c_2 : (X_2, 1), (X_3, 4) \end{aligned}$$

La dernière étape dérive des domaines vides pour X_3 et X_2, donc SAC supprime la valeur 1 du domaine de X_1. Par AC on a ensuite :

$$\begin{aligned} &c_3 : (X_4, 4) \\ &c_1 : (X_2, 2), (X_2, 2) \\ &c_2 : (X_3, 4) \end{aligned}$$

Les domaines sont alors

$$\begin{aligned} &D(X_1) = \{2\} \\ &D(X_2) = \{1\} \\ &D(X_3) = \{2\} \\ &D(X_4) = \{3\} \end{aligned}$$

qui est singleton arc cohérent.

Question 3. Combien comporte de variables le plus petit réseau de contraintes binaires normalisé (c-à-d tel que deux contraintes ne peuvent pas avoir la même portée) qui est SAC mais n'a pas de solution ? Justifiez.

Correction. Le plus petit réseau de contraintes binaires normalisé qui est SAC mais n'a pas de solution a 4 variables. En effet, pour trois variables ou moins toute valeur singleton viable s'étend à une solution. En revanche, il est facile de vérifier que la clique de différences sur quatre variables

$$\begin{cases} X_1 \neq X_2 \\ X_1 \neq X_3 \\ X_1 \neq X_4 \\ X_2 \neq X_3 \\ X_2 \neq X_4 \\ X_3 \neq X_4 \end{cases}$$

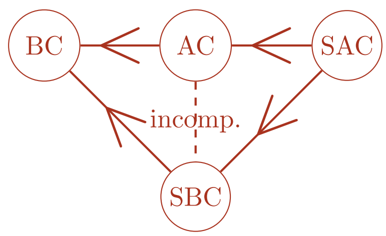
avec un domaine $\{1, 2, 3\}$ pour toutes les variables est SAC mais n'a pas de solution.

Question 4. On peut définir la propriété SBC de façon analogue à SAC, en remplaçant AC par BC dans la définition. Comparez les propriétés AC, SAC, BC et SBC.

Correction. Par le cours, SAC est strictement plus fort que AC, qui est strictement plus fort que BC. Maintenant, comparons AC et SBC. AC n'est pas plus fort que SBC car le réseau $(X_1 = X_2)$, $(X_2 \neq X_1)$ avec des domaines $\{0, 1\}$ est AC mais pas SBC. De même, SBC n'est pas plus fort que AC car le réseau composé d'une unique contrainte $2 \in \{X_1, X_2, X_3\}$ avec des domaines $\{1, 3\}$ est SBC mais pas AC. SBC et AC sont donc incomparables.

Comme SBC est au moins aussi fort que BC et incomparable avec AC (qui est strictement plus fort que BC), SBC est strictement plus fort que BC. De même, SAC est au moins aussi fort que SBC et SBC est incomparable avec AC (qui est strictement plus faible que SAC), donc SAC est strictement plus fort que SBC.

Visuellement, on a donc :



Un réseau de contraintes binaires $N = (X, D, C)$ est k-quasi-arborescent s'il est normalisé et qu'il existe k variables dont la suppression (ainsi que la suppression des contraintes incidentes) produit un réseau arborescent. On rappelle qu'un réseau arborescent AC admet toujours une solution.

Question 5. Un réseau 1-quasi-arborescent qui est SAC admet-il toujours une solution ? Justifiez.

Correction. Oui. En effet, soit N un réseau 1-quasi-arborescent et SAC, X_1 une variable de N dont la suppression produit un réseau arborescent et $v \in D(X_1)$ une valeur singleton viable dans son domaine. Par définition, le réseau N' obtenu en fixant $D(X_1) = v$ est arc cohérent. De plus, les contraintes comprenant X_1 dans leur portée peuvent être supprimées sans conséquences car elles sont satisfaites par toute affectation des variables à des valeurs de leur domaine. Le réseau ainsi obtenu est arborescent et arc-cohérent, donc il a une solution (voir TD 1) qui est également une solution de N.

Question 6. Soit $k \geq 1$. Peut-on résoudre en temps polynomial les réseaux de contraintes k-quasi-arborescents ? Justifiez.

Correction. Oui. En effet, soit N un réseau k-quasi-arborescent. On commence par énumérer les sous-ensembles de variables de taille k jusqu'à en trouver un dont la suppression produit un réseau arborescent. On note S ce sous-ensemble de variables.

Maintenant, on énumère toutes les affectations possibles de S. Pour chaque affectation ϕ, on fixe chaque variable $X \in S$ à la valeur $\phi(X)$ et on applique l'arc cohérence. Puisque le réseau résiduel (après affectation de S) est arborescent, si l'arc cohérence ne vide pas les domaines pour au moins une affectation ϕ alors on peut déduire que N admet une solution. Réciproquement, si l'arc cohérence dérive au moins un domaine vide pour toute affectation ϕ alors aucune affectation de S ne peut s'étendre à une solution de N, et donc N n'admet pas de solution.

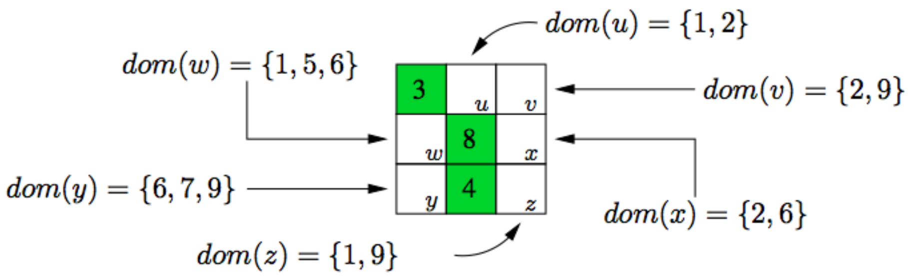
Il y a $O(n^k)$ sous-ensembles de variables S à énumérer, où n est le nombre de variables. Pour chacun, on peut vérifier en temps linéaire si le réseau résiduel (après suppression) est arborescent et il y a $O(d^k)$

affectations à énumérer, où d est la taille maximum des domaines. L'algorithme ci-dessus est donc polynomial pour un k fixé.

Exercice 5

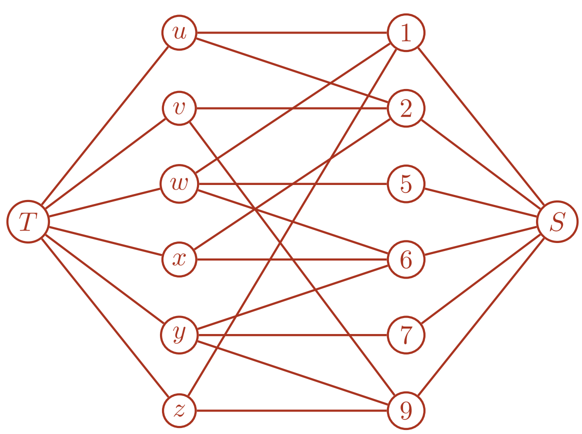
Soit le réseau de contraintes $N = (X, D, C)$, où $X = \{u, v, w, x, y, z\}$, $C = \{$`AllDifferent`$(u, v, w, x, y, z)\}$ et les domaines sont définis comme ci-dessous.

Question 1. Donner l'état des domaines après fermeture GAC pour les domaines suivants :

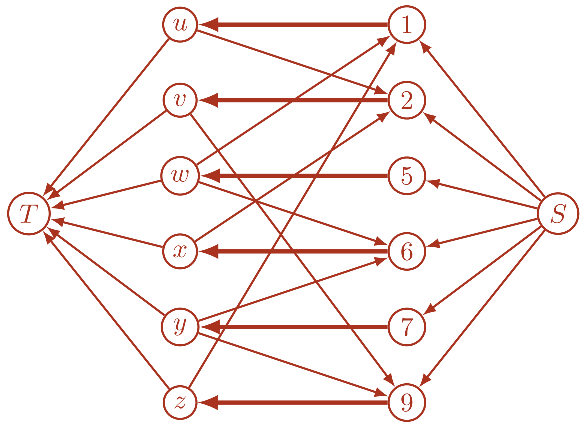


Correction. Les questions de cet exercice peuvent être traitées de deux façons différentes : soit en revenant à la définition de l'arc cohérence et en cherchant des supports pour chaque valeur, soit en appliquant l'algorithme d'arc cohérence spécialisé pour la contrainte `AllDifferent` entrevu en cours. La deuxième approche est plus instructive (et vous devriez être déjà familiers avec la première), donc c'est celle que nous verrons dans cette correction.

On commence par tracer le graphe des valeurs de la contrainte :



Il faut ensuite calculer un flot maximum de S vers T. Cela revient à calculer un couplage maximum dans le graphe biparti ayant les variables à gauche et les valeurs à droite. Ici, le couplage qui associe chaque variable à la valeur qui lui fait directement face est parfait. Ce couplage est de taille $|X|$, donc la contrainte admet au moins une affectation cohérente (donnée par le couplage). On oriente les arêtes utilisées par le flot de droite à gauche, et toutes les autres arêtes dans l'autre sens (le couplage utilisé est dessiné en surbrillance) :

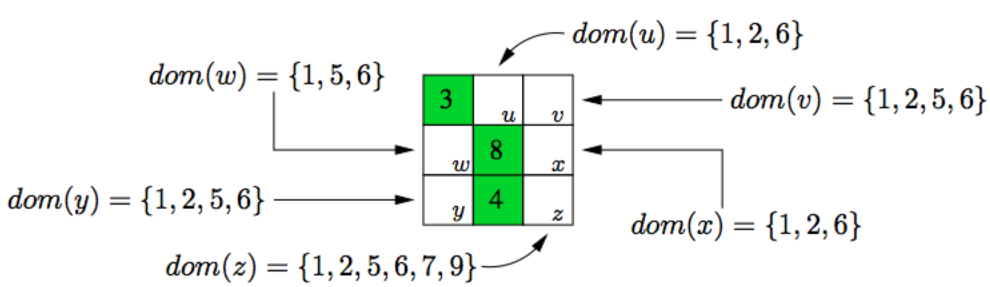


On identifie maintenant les composantes fortement connexes du graphe. Ici, on en a sept : $\{u, v, z, 1, 2, 9\}$, $\{y\}$, $\{7\}$, $\{5\}$, $\{w\}$, $\{6\}$ et $\{x\}$. Finalement, on supprime des domaines toutes les valeurs correspondant à des arêtes connectant des composantes connexes distinctes, à l'exception des arêtes du couplage. Les domaines résiduels sont alors :

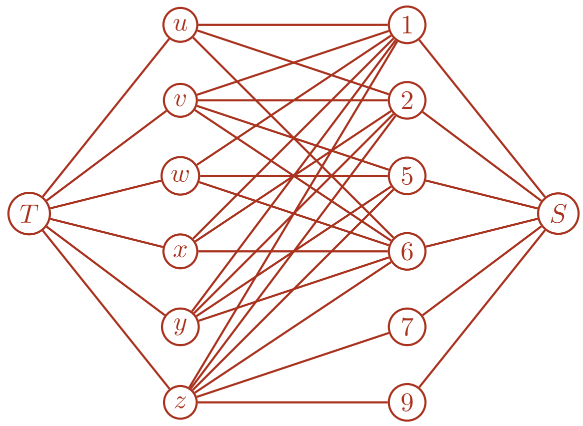
$$
\begin{aligned}
D(u) &= \{1, 2\} \\
D(v) &= \{2, 9\} \\
D(w) &= \{5\} \\
D(x) &= \{6\} \\
D(y) &= \{7\} \\
D(z) &= \{1, 9\}
\end{aligned}
$$

ce qui est la fermeture AC de N.

Question 2. Donner l'état des domaines après fermeture GAC pour les domaines suivants :



Correction. On trace le graphe des valeurs :



Cette fois-ci, il n'existe pas de couplage de taille $|X|$ car $|X| = |D|$ et les valeurs 7 et 9 ont z comme unique variable voisine. Comme les couplages de taille $|X|$ du graphe des valeurs sont en correspondance avec les affectations de X qui satisfont la contrainte `AllDiff`, cette contrainte n'a aucun support pour aucune valeur ; la fermeture AC de N est donc un ensemble de domaines vides.