Estimer le nombre de solutions des contraintes de cardinalité grâce à leur décomposition `range` et `roots`

Giovanni Lo Bianco[1]*Xavier Lorca[2] Charlotte Truchet[3]

[1] IMT Atlantique, Nantes
[2] IMT Mines Albi-Carmaux, Albi
[3] Université de Nantes
giovanni.lo-bianco@imt-atlantique.fr

Résumé

En programmation par contraintes, le choix d'une heuristique de recherche plutôt qu'une autre dépend souvent du problème. Cependant il existe des heuristiques génériques utilisant plutôt des indicateurs sur la structure combinatoire du problème. Les heuristiques "Counting-Based", introduites par Pesant et al., font des choix basés sur une estimation du nombre de solutions restantes dans tel ou tel sous-arbre de l'arbre de recherche. Un inconvénient de ces heuristiques est qu'elles nécessitent des algorithmes de dénombrement spécifiques à chaque contrainte. Cette étude s'intéresse aux contraintes de cardinalité, dont alldifferent, atmost, nvalue, etc... Nous proposons une méthode de comptage de solutions pour les contraintes range et roots, introduites par Bessiere et al. Grâce à la décomposition des contraintes de cardinalité en contraintes range et roots, nous dérivons une méthode systématique de dénombrement de solutions pour la plupart de ces contraintes.

Abstract

This paper proposes a systematic approach for solution counting on cardinality constraints. A main limitation of the classical counting-based search approaches is due to the fact that counting solutions is at least as hard as the constraints' development itself. A typical example is the process of counting solutions for the `alldifferent` and `gcc` constraints in the context of counting-based search [6]. This paper shows that it is possible to obtain a systematic and efficient method by counting solutions on the global constraint decomposition. For this purpose, we use the `range` and `roots` constraint decomposition [3] to count solutions for a large family of cardinality constraints such as `alldifferent`, `disjoint`, `nvalue`, `atmost`, `atleast`, etc.

1 Introduction

L'un des sujets de recherche actifs en Programmation par contraintes est la conception d'heuristiques indépendantes des problèmes traités, mais tout de même adaptées à leur structure combinatoire. Dans [6], les auteurs présentent les heuristiques appelées counting-based search. Ces heuristiques sélectionnent une paire variable/valeur en se basant sur le nombre de solutions restant après l'instanciation. Par exemple, l'heuristique `maxSD`,(pour maximal solutions density), explore d'abord la branche de l'arbre de recherche qui promet le plus de solutions. En pratique, elle évalue, pour chaque contrainte du problème individuellement, la densité de solutions pour chaque paire variable/valeur impliquée dans cette contrainte. De telles heuristiques requièrent des algorithmes de dénombrement de solutions dédiés pour chaque contrainte. Ces algorithmes de dénombrement sont donnés pour `alldifferent`, `gcc`, `knapsack` et `regular` dans [6]. L'une des limites de ces heuristiques est que la conception d'un algorithme de comptage efficace pour une contrainte spécifique est aussi difficile que le développement de la contrainte elle-même.

Dans cet article, nous portons nos efforts sur plusieurs contraintes de cardinalité : `alldifferent`, `disjoint`, `nvalue`, `atleast_nvalue`, `atmost_nvalue`, `atleast`, `atmost`. Ces contraintes limitent le nombre d'occurrences de certaines valeurs ou le nombre de va-

*Papier doctorant : Giovanni Lo Bianco[1] est auteur principal.

leurs différentes dans une solution. Elles peuvent être modélisées grâce à des graphes bipartis et [6] montre, pour `alldifferent` et `gcc` que le problème consistant à dénombrer les solutions sur ces contraintes peut être ramené à un problème de dénombrement de couplages dans ces graphes. La résolution de tels problèmes est très difficile : ils appartiennent souvent à la classe de complexité #P-complet. C'est pourquoi les counting-based search ne reposent pas sur un comptage exact, mais sur des estimations du nombre réel de solutions. Dans [6], les auteurs donnent une estimation du nombre de solutions sur `alldifferent` et `gcc` avec une borne supérieure. Dans cet article, nous proposons une approche probabiliste pour calculer de telles estimations.

Dans [3], les auteurs présentent deux nouvelles contraintes globales `range` et `roots`, qui spécifient un grand nombre de contraintes de cardinalité. En d'autres termes, pour presque toutes les contraintes de cardinalité, il existe un modèle équivalent utilisant uniquement les contraintes plus primitives `range` et `roots` (ainsi que certaines contraintes arithmétiques). Ce modèle équivalent est appelé la décomposition de la contrainte initiale. Dans [3], les auteurs développent un algorithme de propagation pour `range` et pour `roots` et montrent que l'étape de propagation peut être plus efficace en utilisant la décomposition plutôt que l'algorithme de propagation de la contrainte initial.

Dans cet article, nous utilisons la décomposition en contraintes `range` et `roots` pour compter les solutions. Plus précisément, nous développons une approche probabiliste pour estimer le nombre de solutions sur les contraintes `range` et `roots` et nous développons une méthode systématique pour estimer le nombre de solutions sur de nombreuses contraintes de cardinalité. Par rapport à [6], nous obtenons une estimation au lieu d'une borne supérieure et nous proposons une méthode qui peut être généralisée à un grand nombre de contraintes de cardinalité sans redévelopper un modèle mathématique dédié.

Plan : L'article est organisé de la manière suivante. La section 2 sert d'introduction aux contraintes `range` et `roots` et donne quelques éléments pour comprendre le modèle de graphe biparti associé. Dans la section 3, nous détaillons comment compter exactement le nombre de solutions sur `range` et `roots` puis nous utilisons un modèle probabiliste sur ces contraintes pour estimer le nombre réel de solutions. Dans la section 4, nous donnons la décomposition en contraintes `range` et `roots` et un estimateur du nombre de solutions pour plusieurs contraintes de cardinalité.

2 Résultats préliminaires : Introduction à `range` et `roots`

Dans toute cette étude, nous utiliserons les notations suivantes. Soit $X = \{x_1, \ldots, x_n\}$, l'ensemble des variables. Pour chaque variable $x_i \in X$, on note D_i, son domaine, $Y = \bigcup_{i=1}^{n} D_i = \{y_1, \ldots, y_m\}$, l'union des domaines et $\mathcal{D} = D_1 \times \ldots \times D_n$, le produit cartésien des domaines. On note $d_i = |D_i|$, la taille du domaine de x_i. Soit C, une contrainte portant sur l'ensemble X, on note $\mathcal{S}_{C(X)}$, l'ensemble des tuples autorisés par C sur X et on note $\#C(X)$, le nombre de tuples autorisés.

Les contraintes de cardinalités portent sur le nombre d'occurrences dans la solution d'une valeur spécifique, ou sur le nombre de valeurs ou variables répondant à certaines critères. Parmi elles, nous pouvons lister `alldifferent`, `disjoint`, `nvalue`, `atleast_nvalue`, `atmost_nvalue`, `atleast`, `atmost`. Nous reviendrons sur ces contraintes et les définirons convenablement une à une dans la section 4. Ces contraintes peuvent, la plupart du temps, être modélisées à l'aide de graphes bipartis, dans lesquelles nous recherchons certaines structures, comme, par exemple, des couplages.

Définition 1 (Graphe des domaines)**.** *Soit $G_{X,Y} = G(X \cup Y, E)$, le graphe sur les noeuds $X \cup Y$, et les arêtes $E = \{(x_i, y_j) \mid y_j \in D_i\}$. $G_{X,Y}$ est un graphe biparti représentant le domaine des variables. Il y a une arête entre x_i et y_j ssi $y_j \in D_i$.*

Exemple 1. *Soit $X = \{x_1, x_2, x_3, x_4, x_5\}$ avec $D_1 = \{1, 2, 4\}$, $D_2 = \{2, 3\}$, $D_3 = \{1, 2, 3, 5\}$, $D_4 = \{4, 5\}$ et $D_5 = \{2, 4, 5\}$. Nous obtenons le graphe des domaines $G_{X,Y}$ tel que sur la Figure 1a.*

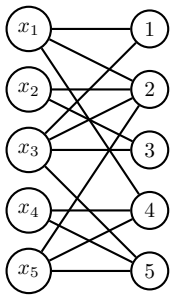
Nous définissons aussi le sous-graphe des domaines induit par les sous-ensembles $X' \subseteq X$ and $Y' \subseteq Y$, comme le graphe des domaines restreints aux sous ensembles de noeuds considérés

Définition 2 (Sous-graphe des domaines)**.** *Soit $G_{X',Y'} = G(X' \cup Y', E)$, le graphe des domaines de X' avec $E = \{(x_i, y_j) \mid y_j \in D'_i = D_i \cap Y'\}$. $G_{X',Y'}$ est un graphe biparti représentant les sous-domaines induits par Y' de chaque variable. Il y a une arête entre x_i et y_j ssi $y_j \in D'_i$*

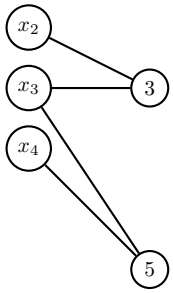
On note $d_i(Y') = |D'_i|$, la taille du sous-domaine de x_i restreint aux valeurs de Y'

L'exemple 2 illustre un sous-graphe des domaines possible au graphe des domaines présenté dans l'exemple 1

Exemple 2. *Soit $X' = \{x_2, x_3, x_4\} \subseteq X$ et $Y' = \{3, 5\} \subseteq Y$. Le sous-graphe des domaines induit par X' et Y' est représenté dans la Figure 1b*



(a) Value Graph $G_{X,Y}$



(b) Sub-Value Graph $G_{X',Y'}$

FIGURE 1 – Value graph and Sub-Value Graph of Example 1 and Example 2.

Les contraintes `range` et `roots` [3] sont deux contraintes auxiliaires qui servent à modéliser la plupart des contraintes de cardinalité. Dans cette étude, nous nous servons de la décomposition en contraintes `range` et `roots` pour dénombrer les solutions des contraintes de cardinalité. La contrainte `range` encapsule la notion d' "image" d'une fonction et la contrainte `roots` encapsule la notion de "domaine". Nous utiliserons ici, une formalisation différente pour ces contraintes de la manière dont elles sont présentées dans [3].

Définition 3 (`range`). *Soit $X' \subseteq X$ et $Y' \subseteq Y$. La contrainte* `range`(X, X', Y') *est satisfaite si les valeurs affectées aux variables de X' couvrent* ***exactement*** *Y' et pas plus. Formellement :*

$$\mathcal{S}_{\texttt{range}(X,X',Y')} = \{(v_1, \ldots, v_n) \in \mathcal{D} \mid \{v_i | x_i \in X'\} = Y'\} \quad (1)$$

Définition 4 (`roots`). *Soit $X' \subseteq X$ et $Y' \subseteq Y$. La contrainte* `roots`(X, X', Y') *est satisfaite si les variables qui sont instanciées aux valeurs de Y' couvrent* ***exactement*** *X' et pas plus. Formellement :*

$$\mathcal{S}_{\texttt{roots}(X,X',Y')} = \{(v_1, \ldots, v_n) \in \mathcal{D} \mid \{x_i | v_i \in Y'\} = X'\} \quad (2)$$

Dans l'exemple 3, nous illustrons comment ces contraintes fonctionnent et pourquoi ces contraintes ne sont pas l'inverse l'une de l'autre.

Exemple 3. *Reprenons le graphe des domaines de l'exemple 1a.*

- *Le tuple* $(2, 2, 3, 4, 5)$ *est autorisé par la contrainte* `range`$(X, \{x_1, x_2, x_3\}, \{2, 3\})$.
- *Le tuple* $(2, 2, 5, 4, 5)$ *est autorisé par la contrainte* `range`$(X, \{x_1, x_2, x_3\}, \{2, 3\})$ *mais pas par* `range`$(X, \{x_1, x_2, x_3\}, \{2, 5\})$.
- *Le tuple* $(2, 2, 3, 4, 5)$ *est autorisé par la contrainte* `roots`$(X, \{x_1, x_2, x_3\}, \{2, 3\})$.
- *Le tuple* $(2, 2, 3, 4, 2)$ *n'est pas autorisé par la contrainte* `roots`$(X, \{x_1, x_2, x_3\}, \{2, 3\})$, *mais l'est par* `roots`$(X, \{x_1, x_2, x_3, x_5\}, \{2, 3\})$.
- *Le tuple* $(2, 2, 2, 4, 5)$ *est autorisé par la contrainte* `range`$(X, \{x_1, x_2, x_3\}, \{2\})$ *mais pas par* `range`$(X, \{x_1, x_2, x_3\}, \{2, 3\})$, *puisque* 3 *n'est ni assigné à x_1, x_2 ou x_3. En revanche, le tuple* $(2, 2, 2, 4, 5)$ *est autorisé par* `roots`$(X, \{x_1, x_2, x_3\}, \{2, 3\})$. *En effet,* 3 *n'est assigné à aucune variable, donc X' n'est défini que par les variables instanciées à* 2.

La raison pour laquelle `range` et `roots` ne sont pas l'inverse l'une de l'autre est que toutes les variables doivent être instanciées à une valeur mais une valeur n'est pas nécessairement affectée à une variable.

3 Dénombrement de solutions sur les contraintes `range` et `roots`

Le dénombrement de solutions sur les contraintes de cardinalité nécessite des algorithmes dédiés à chaque contrainte. Dans cette section, nous nous intéressons au dénombrement de solutions sur les contraintes `range` et `roots`. L'idée est alors d'utiliser la décomposition des contraintes de cardinalité en ces contraintes plus primitives et de réutiliser la méthode de comptage sur `range` et `roots` pour dénombrer les solutions sur les contraintes de cardinalité.

3.1 Dénombrement exact sur `range` and `roots`

Dans cette sous-section, nous sommes intéressés par le dénombrement exact des tuples autorisés par les contraintes `range` et `roots`.

Proposition 1. *Soit $X' \subseteq X$ et $Y' \subseteq Y$. On note $\overline{X'}$, le complémentaire de X' dans X, tel que $\overline{X'} \cup X' = X$ et $\overline{X'} \cap X' = \emptyset$. Le nombre de tuples autorisés par* `range`(X, X', Y') *est*

$$\#\texttt{range}(X, X', Y') = \#\texttt{range}(X', X', Y') \cdot \prod_{x_i \in \overline{X'}} d_i \quad (3)$$

Démonstration. D'un côté, nous devons considérer toutes les instanciations possibles des variables de $\overline{X'}$, qui ne sont pas contraintes : $\prod_{x_i \in \overline{X'}} d_i$

Et de l'autre, nous devons énumérer toutes les instanciations possibles pour les variables de X' qui sont contraintes : $\#\texttt{range}(X', X', Y')$ □

Proposition 1 réduit le problème du comptage d'instanciations autorisées pour les variables de X au comptage d'instanciations autorisées pour les variables contraintes de X' seulement. Nous cherchons maintenant à dénombrer les tuples autorisés lorsque que toutes les variables de X sont contraintes.

Proposition 2.

$$\#\texttt{range}(X, X, Y) = \prod_{x_i \in X} d_i - \sum_{Y' \subsetneq Y} \#\texttt{range}(X, X, Y') \tag{4}$$

Démonstration. Dans $G_{X,Y}$, nous devons compter toutes les affectations possibles des variables de X de sorte que toutes les valeurs de Y soient couvertes. Pour ce faire, nous comptons d'abord le nombre d'affectations possibles des variables de X dans $G_{X,Y}$ (sans tenir compte de la contrainte `range`) :

$$\prod_{x_i \in X} d_i$$

Et ensuite, nous retirons, une à une, les affectations de X qui ne couvrent pas entièrement Y , c'est-à-dire, pour chaque sous-ensemble $Y' \subsetneq Y$, les solutions de $\texttt{range}(X, X, Y')$:

$$\sum_{Y' \subsetneq Y} \#\texttt{range}(X, X, Y')$$

En effet, pour deux sous-ensembles $Y'_1 \neq Y'_2 \subsetneq Y$, l'ensemble des tuples autorisés $\mathcal{S}_{\texttt{range}(X,X,Y'_1)}$ et $\mathcal{S}_{\texttt{range}(X,X,Y'_2)}$ sont nécessairement disjoints : il existe une valeur $y_j \in Y$ tel que $y_j \in Y'_1$ et $y_j \notin Y'_2$ (ou bien $y_j \in Y'_2$ et $y_j \notin Y'_1$), alors y_j doit être affectée à une variable de X pour satisfaire $\texttt{range}(X, X, Y'_1)$ mais aucune des variables de X doit être instanciée à y_j pour satisfaire $\texttt{range}(X, X, Y'_2)$ (ou inversement). Une solution de $\texttt{range}(X, X, Y'_1)$ ne peut être solution de $\texttt{range}(X, X, Y'_2)$ et inversement.

Aucune solution n'est comptée deux fois dans $\sum_{Y' \subset Y} \#\texttt{range}(X, X, Y')$. On a alors :

$$\#\texttt{range}(X, X, Y) = \prod_{x_i \in X} d_i - \sum_{Y' \subsetneq Y} \#\texttt{range}(X, X, Y')$$

□

Remarque 1. *Proposition 2 peut être utilisée dans Proposition 1 :*

$$\#\texttt{range}(X, X', Y') =$$
$$\prod_{x_i \in \overline{X'}} d_i \cdot \left(\prod_{x_i \in X'} d_i(Y') - \sum_{Y'' \subsetneq Y'} \#\texttt{range}(X', X', Y'') \right)$$

Cette formule est récursive et n'est pas utile en pratique. Dans la sous-section suivante, nous verrons qu'il s'agit d'un résultat préliminaire pour le calcul d'une estimation du nombre de solutions pour la contraintes `range`. Nous nous intéressons maintenant à la contrainte `roots`.

Proposition 3. *Soit $X' \subseteq X$ et $Y' \subseteq Y$. On note $\overline{X'}$, le complémentaire de X' dans X et $\overline{Y'}$ le complémentaire de Y' dans Y. Le nombre de tuples autorisés par $\texttt{roots}(X, X', Y')$ est*

$$\#\texttt{roots}(X, X', Y') = \prod_{x_i \in X'} d_i(Y') \cdot \prod_{x_i \in \overline{X'}} d_i(\overline{Y'}) \tag{5}$$

Pour rappel, $d_i(Y') = |D_i \cap Y'|$ et $d_i(\overline{Y'}) = |D_i \cap \overline{Y'}|$

Démonstration. Pour satisfaire la contrainte $\texttt{roots}(X, X', Y')$, toutes les variables de X' doivent prendre des valeurs dans Y' et aucune valeur de Y' doit être affectée aux variables de $\overline{X'}$, c'est-à-dire toutes les variables de $\overline{X'}$ doivent être affectées à des valeurs de $\overline{Y'}$:

- $\prod_{x_i \in X'} d_i(Y')$ donne le nombre d'affectations possibles pour X'
- $\prod_{x_i \in \overline{X'}} d_i(\overline{Y'})$ donne le nombre d'affectations possibles pour $\overline{X'}$

□

3.2 Modèle probabiliste appliqué à `range` et `roots`

Dans cette sous-section, nous présentons un modèle probabiliste pour les contraintes de cardinalité en se basant sur les travaux de Erdős et Renyi [4]. L'idée est de probabiliser le domaine des variables. Ensuite, nous utiliserons ce modèle pour obtenir une estimation du nombre de solutions sur `range` et `roots`.

3.2.1 Le modèle Erdős-Renyi appliqué aux CSP

Erdős et Renyi [4] ont mené des recherches sur les graphes aléatoires et ont étudié l'existence et le nombre de couplages parfaits dans ces structures aléatoires. Appliquée directement à notre problème, l'idée est de probabiliser le domaine de chaque variable de sorte que : pour tout $x_i \in X$ et pour tout $y_j \in Y$, l'évènement

$\{y_j \in D_i\}$ se produit avec une probabilité $p \in [0,1]$ et tous ces évènements sont **indépendants** :

$$\mathbb{P}(\{y_j \in D_i\}) = p \in [0,1] \tag{6}$$

Dans la suite, $\mathbb{E}(.)$ désigne l'espérance sous les hypothèses du modèle Erdős-Renyi

3.2.2 Le modèle Erdős-Renyi appliqué à la contrainte `range`

Nous étudions maintenant l'espérance du nombre de solutions d'une contrainte `range`. Dans le cas où chaque variable de X et chaque valeur de Y sont contraintes, l'espérance de $\#\texttt{range}(X,X,Y)$ est une fonction de n, m et p (pour rappel, $|X| = n$ et $|Y| = m$). Proposition 4 précise la forme de cette fonction.

Proposition 4. *Dans le cas où toutes les variables de X et toutes les valeurs de Y sont contraintes, il existe $a_{n,m} \in \mathbb{N}$ tel que :*

$$\mathbb{E}(\#\texttt{range}(X,X,Y)) = a_{n,m} \cdot p^n \tag{7}$$

Démonstration. Pour montrer ce résultat, nous raisonnons simplement par récurrence sur $|Y| = m$. Soit $|X| = n \in \mathbb{N}$,

Initialisation. Soit $Y = \{y\}$ un singleton. Dans ce cas particulier, une instance $\texttt{range}(X,X,Y)$ a un seul tuple autorisé si y est dans tous les domaines D_i, et n'a aucun tuple autorisé sinon. Donc,

$$\begin{aligned}
&\mathbb{E}(\#\texttt{range}(X,X,\{y\})) \\
&= 0 * \mathbb{P}(\{\texttt{range}(X,X,\{y\}) \text{ n'a pas de solution}\}) \\
&\quad + 1 * \mathbb{P}(\{\texttt{range}(X,X,\{y\}) \text{ a une seule solution}\}) \\
&= \mathbb{P}(\{\texttt{range}(X,X,\{y\}) \text{ a une seule solution}\}) \\
&= \mathbb{P}(\{\forall x_i \in X, y \in D_i\}) \\
&= \prod_{i=1}^{n} \mathbb{P}(\{y \in D_i\}), \text{par hypothèse d'indépendance} \\
&= p^n
\end{aligned}$$

On a alors $a_{n,1} = 1$.

Récurrence. On suppose que la proposition est vraie pour tout Y, tel que $1 \leq |Y| = k \leq m-1, \exists a_{n,k} \in \mathbb{N}$,

$$\mathbb{E}(\#\texttt{range}(X,X,Y)) = a_{n,k} \cdot p^n$$

On veut prouver, sous ces hypothèses, pour un ensemble Y avec $|Y| = m$, qu'il existe $a_{n,m}$, tel que $\mathbb{E}(\#\texttt{range}(X,X,Y)) = a_{n,m} \cdot p^n$

On a :

$$\begin{aligned}
&\mathbb{E}(\#\texttt{range}(X,X,Y)) \\
&= \mathbb{E}\left(\prod_{x_i \in X} d_i\right) - \sum_{Y' \subsetneq Y} \mathbb{E}(\#\texttt{range}(X,X,Y')) \\
&= \mathbb{E}\left(\prod_{x_i \in X} d_i\right) - \sum_{k=1}^{m-1} \binom{m}{k} a_{n,k} \cdot p^n \\
&= \prod_{x_i \in X} \mathbb{E}(d_i) - \sum_{k=1}^{m-1} \binom{m}{k} a_{n,k} \cdot p^n \\
&= (mp)^n - \sum_{k=1}^{m-1} \binom{m}{k} a_{n,k} \cdot p^n \\
&= \left(m^n - \sum_{k=1}^{m-1} \binom{m}{k} a_{n,k}\right) \cdot p^n
\end{aligned}$$

La première égalité vient de Proposition 2 et de la linéarité de l'opérateur $\mathbb{E}(.)$. La seconde vient de l'hypothèse de récurrence. Le troisième vient de l'hypothèse d'indépendance. Et le quatrième est dû au fait que $\forall x_i \in X, \mathbb{E}(d_i) = \sum_{j=1}^{m} \mathbb{P}(\{y_j \in D_i\}) = mp$.

Nous avons identifié le $a_{n,m}$:

$$a_{n,m} = m^n - \sum_{k=1}^{m-1} \binom{m}{k} a_{n,k} \tag{8}$$

□

En remarquant que $\binom{m}{m} = 1$, on peut réécrire 8 comme suit :

$$m^n = \sum_{k=1}^{m} \binom{m}{k} a_{n,k} \tag{9}$$

Aussi, $\forall n \in \mathbb{N}^+, a_{n,1} = 1$. Ces coefficients sont référencés comme les "triangles of numbers" dans OEIS [1]. Les coefficients $a_{n,m}$ sont en fait le nombre de surjections possibles d'un ensemble de taille n dans un autre ensemble de taille m [2]. Il existe une formule non récursive pour calculer ces coefficients. Le résultat suivant est admis ici. L'intuition de la preuve est qu'il s'agit d'une application du principe inclusion-exclusion (voir section 1.9. The Twelvefold Way de [8]).

Proposition 5. *Pour $0 < m \leq n$,*

$$a_{n,m} = \sum_{k=0}^{m} (-1)^k \binom{m}{k} (m-k)^n \tag{10}$$

1. https://oeis.org/A019538
2. $a_{n,m}$ est en fait égale à $m! \cdot S_2(n,m)$, où $S_2(n,m)$ est le nombre de Stirling de deuxième espèce. Plus d'informations dans la section 1.9 de [8]

Proposition 6 est une propriété des "triangles of numbers" and sera utilisée pour faire des simplifications lors de prochains calculs.

Proposition 6.

$$a_{n,n} = n! \tag{11}$$

Démonstration. $a_{n,n}$ est le nombre de surjections possibles d'un ensemble de cardinalité n dans un ensemble de cardinalité n, soit le nombre de bijections dans ce cas particulier. □

Nous pouvons maintenant étendre Proposition 4 au cas où la contrainte `range` concerne uniquement des sous-ensembles $X' \subseteq X$ et $Y' \subseteq Y$:

Proposition 7. *Soit* $X' \subseteq X$ *et* $Y' \subseteq Y$. *On note* $|X'| = n'$ *et* $|Y'| = m'$.

$$\mathbb{E}\left(\#\texttt{range}(X, X', Y')\right) = a_{n',m'} \cdot m^{n-n'} \cdot p^n \tag{12}$$

Démonstration. On déduit des propositions 1 et 4 et par hypothèse d'indépendance :

$$\begin{aligned}
&\mathbb{E}\left(\#\texttt{range}(X, X', Y')\right) \\
&= \mathbb{E}\left(\#\texttt{range}(X', X', Y')\right) \cdot \mathbb{E}\left(\prod_{x_i \in \overline{X'}} d_i\right) \\
&= a_{n',m'} \cdot p^{n'} \cdot \prod_{x_i \in \overline{X'}} \mathbb{E}\left(d_i\right) \\
&= a_{n',m'} \cdot p^{n'} \cdot (mp)^{n-n'} \\
&= a_{n',m'} \cdot m^{n-n'} \cdot p^n
\end{aligned}$$

□

3.2.3 Le modèle Erdős-Renyi appliqué à la contrainte `roots`

On étudie maintenant l'espérance du nombre de solutions sur une contrainte `roots`.

Proposition 8. *Soit* $X' \subseteq X$ *et* $Y' \subseteq Y$. *On note* $|X'| = n'$ *et* $|Y'| = m'$.

$$\mathbb{E}\left(\#\texttt{roots}(X, X', Y')\right) = m'^{n'} \cdot (m-m')^{n-n'} \cdot p^n \tag{13}$$

Démonstration. D'après Proposition 3 et par hypothèse d'indépendance :

$$\begin{aligned}
&\mathbb{E}\left(\#\texttt{roots}(X, X', Y')\right) \\
&= \mathbb{E}\left(\prod_{x_i \in X'} d_i(Y')\right) \cdot \mathbb{E}\left(\prod_{x_i \in \overline{X'}} d_i(\overline{Y'})\right) \\
&= \prod_{x_i \in X'} \mathbb{E}\left(d_i(Y')\right) \cdot \prod_{x_i \in \overline{X'}} \mathbb{E}\left(d_i(\overline{Y'})\right) \\
&= (m'p)^{n'} \cdot ((m-m')p)^{n-n'} \\
&= m'^{n'} \cdot (m-m')^{n-n'} \cdot p^n
\end{aligned}$$

□

Remarque 2. *Le paramètre p correspond à la densité d'arêtes dans le graphe des domaines. Pour utiliser l'estimateur en pratique, il nous faut donc évaluer p en divisant la somme de la taille des domaines par le nombre total d'arêtes possible :* $n \cdot m$.

4 Généralisation aux contraintes de cardinalité

Cette section présente une méthode systématique pour dénombrer les solutions pour la plupart des contraintes de cardinalité grâce à leur décomposition `range` et `roots`. Chaque sous-section rappelle d'abord les définitions des contraintes puis leur décomposition.

4.1 `alldifferent` [7]

Définition 5. *La contrainte* $\texttt{alldifferent}(X)$ *est satisfaite ssi chaque variable* $x_i \in X$ *est instanciée à une valeur de son domaine* D_i *et chaque valeur de* $y_j \in Y$ *est affectée au plus une fois. Formellement :*

$$\begin{aligned}
&\mathcal{S}_{\texttt{alldifferent}(X)} = \\
&\{(v_1, \ldots, v_n) \in \mathcal{D} \mid \forall i, j \in \{1, ..., n\}, i \neq j \Leftrightarrow v_i \neq v_j\}
\end{aligned}$$

La décomposition de `alldifferent` en contrainte `range` est la suivante [3] :

$$\texttt{alldifferent}(X) \Leftrightarrow \texttt{range}(X, X, Y') \;\&\; |Y'| = n$$

À partir de cette décomposition, on en déduit une formule pour estimer le nombre de solutions sur `alldifferent`, d'après le modèle Erdős-Renyi.

Proposition 9.

$$\mathbb{E}\left(\#\texttt{alldifferent}(X)\right) = \frac{m!}{(m-n)!} \cdot p^n \tag{14}$$

Démonstration. D'après la décomposition de `alldifferent`

$$\#\texttt{alldifferent}(X) = \sum_{Y' \subseteq Y,\, |Y'|=n|} \#\texttt{range}(X, X, Y')$$

Alors,

$$\begin{aligned}
&\mathbb{E}\left(\#\texttt{alldifferent}(X)\right)\\
&= \sum_{Y' \subseteq Y,\, |Y'|=n} \mathbb{E}\left(\#\texttt{range}(X, X, Y')\right)\\
&= \binom{m}{n} \cdot a_{n,n} \cdot p^n\\
&= \frac{m!}{(m-n)!} \cdot p^n
\end{aligned}$$

□

Remarque 3. *Compter le nombre de solutions sur la contrainte* `alldifferent` *équivaut à compter le nombre de couplages couvrant X dans le graphe des domaines [9]. L'application du modèle Erdős-Renyi sur ce problème et le calcul de l'espérance du nombre de couplages parfaits ont été étudiés dans [4] dans le cas de graphes bipartis équilibrés. La proposition 9 est en quelque sorte une extension de ce résultat pour les graphes bipartis non-équilibrés.*

4.2 `nvalue` [5]

Définition 6. *Soit $N \in \mathbb{N}$. La contrainte* `nvalue`(X, N) *est satisfaite si il y a exactement N valeurs de Y assignées aux variables de X. Formellement :*

$$\begin{aligned}
&\mathcal{S}_{\texttt{nvalue}(X,N)} =\\
&\{(v_1, \ldots, v_n) \in \mathcal{D} |\ N = |\{y_j \in Y | \exists x_i \in X, v_i = y_j\}|\}
\end{aligned}$$

La décomposition de `nvalue` en contrainte `range` est la suivante [3] :

$$\texttt{nvalue}(X, N) \Leftrightarrow \texttt{range}(X, X, Y') \ \&\ |Y'| = N$$

À partir de cette décomposition, on en déduit une formule pour estimer le nombre de solutions sur `nvalue`, d'après le modèle Erdős-Renyi.

Proposition 10.

$$\mathbb{E}\left(\#\texttt{nvalue}(X, N)\right) = \binom{m}{N} \cdot a_{n,N} \cdot p^n \qquad (15)$$

Démonstration. La preuve est la même que pour Proposition 9 □

4.3 `atmost_nvalue` et `atleast_nvalue`

À partir de Proposition 10, on peut déduire l'espérance du nombre de solutions pour `atmost_nvalue`et `atleast_nvalue`.

Définition 7 (`atmost_nvalue` [2])**.** *Soit $N \in \mathbb{N}$. La contrainte* `atmost_nvalue`(X, N) *est satisfaite si au plus N valeurs de Y sont affectées aux variables de X.*

Définition 8 (`atleast_nvalue` [2])**.** *Soit $N \in \mathbb{N}$. La contrainte* `atleast_nvalue`(X, N) *est satisfaite si au moins N valeurs de Y sont affectées aux variables de X.*

On peut décomposer `atleast_nvalue` and `atmost_nvalue` directement avec la contrainte `nvalue` :

$$\texttt{atmost\_nvalue}(X, N) \Leftrightarrow \texttt{nvalue}(X, k) \ \&\ k \leq N$$

$$\texttt{atleast\_nvalue}(X, N) \Leftrightarrow \texttt{nvalue}(X, k) \ \&\ k \geq N$$

Proposition 11.

$$\mathbb{E}\left(\#\texttt{atmost\_nvalue}(X, N)\right) = \sum_{k=1}^{N} \binom{m}{k} a_{n,k} \cdot p^n \qquad (16)$$

$$\mathbb{E}\left(\#\texttt{atleast\_nvalue}(X, N)\right) = \sum_{k=N}^{n} \binom{m}{k} a_{n,k} \cdot p^n \qquad (17)$$

4.4 `disjoint` [1]

Définition 9. *Soit $X_1 \subset X$ et $X_2 \subset X$, tels que $X_1 \cap X_2 = \emptyset$,* `disjoint`(X, X_1, X_2) *assure qu'aucune des variables de X_1 n'est affectée à la même valeur que l'une des variables de X_2. Formellement :*

$$\begin{aligned}
&\mathcal{S}_{\texttt{disjoint}(X,X_1,X_2)} =\\
&\{(v_1, \ldots, v_n) \in \mathcal{D} |\ \{v_i | x_i \in X_1\} \cap \{v_i | x_i \in X_2\} = \emptyset\}
\end{aligned}$$

Une décomposition de `disjoint` avec deux contraintes `range` est donnée par l'équivalence suivante [3] :

$$\begin{aligned}
\texttt{disjoint}(X, X_1, X_2) \Leftrightarrow\ &\texttt{range}(X, X_1, Y_1) \ \&\ \texttt{range}(X, X_2, Y_2)\\
&\&\ Y_1 \subset Y \ \&\ Y_2 \subset Y \ \&\ Y_1 \cap Y_2 = \emptyset
\end{aligned}$$

On en déduit une formule pour estimer le nombre de solutions sur `disjoint`, sous les hypothèses du modèle Erdős-Renyi.

Proposition 12. *Soit $X_1 \in X$ et $X_2 \in X$ avec $|X_1| = n_1$ and $|X_2| = n_2$ et $X_1 \cap X_2 = \emptyset$*

$$\mathbb{E}\left(\#\texttt{disjoint}(X, X_1, X_2)\right) = p^n \cdot m^{n-n_1-n_2} \cdot \sum_{k=1}^{min(n_1,m)} \sum_{l=1}^{min(n_2,m-k)} \binom{m}{k}\binom{m-k}{l} a_{n_1,k} a_{n_2,l}$$

Démonstration. D'après la décomposition de `disjoint`, on a :

$$\#\texttt{disjoint}(X, X_1, X_2) = \left(\prod_{x_i \in X \setminus \{X_1 \cup X_2\}} d_i\right) \cdot \left(\sum_{Y_1 \cap Y_2 = \emptyset} \#\texttt{range}(X_1, X_1, Y_1) \cdot \#\texttt{range}(X_2, X_2, Y_2)\right)$$

Le premier facteur du produit est le nombre total de combinaisons pour les variables non contraintes. Une fois que ces variables sont instanciées, on doit choisir deux sous-ensembles Y_1 et Y_2 tels que $Y_1 \cap Y_2 = \emptyset$ et calculer le nombre de solutions de $\texttt{range}(X_1, X_1, Y_1)$ et $\texttt{range}(X_2, X_2, Y_2)$. Pour deux paires différentes de sous-ensembles (Y_1^A, Y_2^A) et (Y_1^B, Y_2^B), l'ensemble des solutions de $\{\texttt{range}(X_1, X_1, Y_1^A) \& \texttt{range}(X_2, X_2, Y_2^A)\}$ et $\{\texttt{range}(X_1, X_1, Y_1^B) \& \texttt{range}(X_2, X_2, Y_2^B)\}$ sont disjoints. Nous ne comptons alors pas plusieurs fois une même solution.

On a aussi :

$$\mathbb{E}\left(\prod_{x_i \in X \setminus \{X_1 \cup X_2\}} d_i\right) = \prod_{x_i \in X \setminus \{X_1 \cup X_2\}} \mathbb{E}\left(d_i\right) = (mp)^{n-n_1-n_2}$$

Et,

$$\mathbb{E}\left(\sum_{Y_1 \cap Y_2 = \emptyset} \#\texttt{range}(X_1, X_1, Y_1) \cdot \#\texttt{range}(X_2, X_2, Y_2)\right) = \sum_{Y_1 \cap Y_2 = \emptyset} a_{n_1,|Y_1|} p^{n_1} \cdot a_{n_2,|Y_2|} p^{n_2},$$

par la Proposition 4 et hypothèse d'indépendance

Chaque paire (Y_1, Y_2) avec une taille fixée à $|Y_1| = k$ et $|Y_2| = l$ aboutisse à une même formule d'estimation, alors nous choisissons d'abord un sous-ensemble de valeur Y_1 de taille $k \in \{1, \ldots, min(n_1, m)\}$ et ensuite un sous-ensemble de valeur Y_2 de taille $l \in \{1, \ldots, min(n_2, m-k)\}$:

$$\sum_{Y_1 \cap Y_2 = \emptyset} a_{n_1,|Y_1|} \cdot a_{n_2,|Y_2|} = \sum_{k=1}^{min(n_1,m)} \sum_{l=1}^{min(n_2,m-k)} \binom{m}{k}\binom{m-k}{l} a_{n_1,k} a_{n_2,l}$$

En multipliant tous ces facteurs, nous déduisons la Proposition 12 □

4.5 `atmost` et `atleast`

Définition 10 (`atmost`). *Soit $y \in Y$ et $N \in \mathbb{N}$, la contrainte* `atmost`(X, y, N) *est satisfaite si au plus N variables sont instanciées à la valeur y*

$$\mathcal{S}_{\texttt{atmost}(X,y,N)} = \{(d_1, \ldots, d_n) | N \geq |\{x_i | d_i = y\}|\}$$

Définition 11 (`atleast`). *Soit $y \in Y$ et $N \in \mathbb{N}$, la contrainte* `atleast`(X, y, N) *est satisfaite si au moins N variables sont instanciées à la valeur y.*

$$\mathcal{S}_{\texttt{atleast}(X,y,N)} = \{(d_1, \ldots, d_n) | N \leq |\{x_i | d_i = y\}|\}$$

Les décompositions des contraintes `atmost` and `atleast` en contraintes `range` et `rootss`ont données par les équivalences suivantes [3] :

$$\texttt{atmost}(X, y, N) \Leftrightarrow \texttt{roots}(X, X', \{y\}) \;\&\; |X'| \leq N$$

$$\texttt{atleast}(X, y, N) \Leftrightarrow \texttt{roots}(X, X', \{y\}) \;\&\; |X'| \geq N$$

Proposition 13.

$$\mathbb{E}\left(\#\texttt{atmost}(X, y, N)\right) = \sum_{k=1}^{N} \binom{n}{k} (m-1)^{n-k} \cdot p^n \quad (18)$$

$$\mathbb{E}\left(\#\texttt{atleast}(X, y, N)\right) = \sum_{k=N}^{n} \binom{n}{k} (m-1)^{n-k} \cdot p^n \quad (19)$$

Démonstration. Nous faisons la preuve seulement pour la contrainte `atmost`. On a :

$$\#\texttt{atmost}(X, y, N) = \sum_{X' \subseteq X, |X'| \leq N} \#\texttt{roots}(X, X', \{y\})$$

En effet, pour deux sous-ensembles différents $X'_1 \neq X'_2 \subseteq X$, l'ensemble des solutions de $\texttt{roots}(X, X'_1, \{y\})$ et $\texttt{roots}(X, X'_2, \{y\})$ ont une intersection vide. Aucune solution n'est comptée plusieurs fois. Et :

$$\begin{aligned}
&\mathbb{E}\left(\#\texttt{atmost}(X, y, N)\right) \\
&= \sum_{X' \subseteq X, |X'| \leq N} \mathbb{E}\left(\#\texttt{roots}(X, X', \{y\})\right) \\
&= \sum_{X' \subseteq X, |X'| \leq N} (m-1)^{n-|X'|} \cdot p^n, \text{ par la Proposition 8} \\
&= \sum_{k=1}^{N} \binom{n}{k} (m-1)^{n-k} \cdot p^n
\end{aligned}$$

□

Remarque 4. *Nous remarquons que beaucoup de ces formules nécessitent le calcul de coefficients binomiaux, de factorielles et des "triangles of numbers" $a_{n,m}$. Lors de l'utilisation d'heuristiques counting-based search, ces estimateurs doivent être calculés plusieurs fois au cours de la recherche. Nous proposons de pré-calculer tous ces coefficients au début de la résolution pour diminuer le temps de calcul. Le calcul des estimateurs pour* `alldifferent` *et* `nvalue` *est donc en temps constant, le calcul des estimateurs pour* `atmost_nvalue`*,* `atleast_nvalue`*,* `atmost`*,* `atleast` *est linéaire et le calcul de l'estimateur pour* `disjoint` *a une complexité quadratique.*

5 Conclusion

Dans cet article, nous avons proposé une méthode pour estimer le nombre de solutions des contraintes `range` et `roots` de manière probabiliste avec le modèle Erdős-Renyi. Nous avons montré que nous pouvons estimer le nombre de solutions pour de nombreuses contraintes de cardinalité en utilisant leur décomposition `range` et `roots`. Nous avons expliqué notre méthode pour les contraintes `alldifferent`, `disjoint`, `nvalue`, `atleast_nvalue`, `atmost_nvalue`, `atleast`, `atmost`.

Dans la poursuite de ce travail, nous pensons étendre cette méthode de dénombrement probabiliste à d'autres contraintes de cardinalité. Nous souhaitons également expérimenter l'utilisation de ces estimateurs au sein d'heuristiques counting-based search et de les comparer à d'autres stratégies, telles que `dom/wdeg`, `abs` ou bien `ibs`.

Les contraintes pour lesquelles la décomposition est une conjonction de plusieurs contraintes `range` et/ou `roots` non indépendantes (`gcc` par exemple) posent encore problème. En d'autres termes, si au moins deux contraintes `range` et/ou `roots` concernent des ensembles de variables qui ne sont pas disjoints, notre méthode doit être réadaptée.

Références

[1] Nicolas Beldiceanu : Global constraints as graph properties on a structured network of elementary constraints of the same type. *In* Rina Dechter, éditeur : *Principles and Practice of Constraint Programming - CP 2000, 6th International Conference, Singapore, September 18-21, 2000, Proceedings*, volume 1894 de *Lecture Notes in Computer Science*, pages 52–66. Springer, 2000.

[2] Christian Bessière, Emmanuel Hebrard, Brahim Hnich, Zeynep Kiziltan et Toby Walsh : Filtering algorithms for the nvalue constraint. *In* Roman Barták et Michela Milano, éditeurs : *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Second International Conference, CPAIOR 2005, Prague, Czech Republic, May 30 - June 1, 2005, Proceedings*, volume 3524 de *Lecture Notes in Computer Science*, pages 79–93. Springer, 2005.

[3] Christian Bessiere, Emmanuel Hebrard, Brahim Hnich, Zeynep Kiziltan et Toby Walsh : Range and roots : Two common patterns for specifying and propagating counting and occurrence constraints. *Artif. Intell.*, 173(11):1054–1078, 2009.

[4] P. Erdos et A. Renyi : On random matrices. *Publication of the Mathematical Institute of the Hungarian Academy of Science*, 1963.

[5] François Pachet et Pierre Roy : Automatic generation of music programs. *In* Joxan Jaffar, éditeur : *Principles and Practice of Constraint Programming - CP'99, 5th International Conference, Alexandria, Virginia, USA, October 11-14, 1999, Proceedings*, volume 1713 de *Lecture Notes in Computer Science*, pages 331–345. Springer, 1999.

[6] Gilles Pesant, Claude-Guy Quimper et Alessandro Zanarini : Counting-based search : Branching heuristics for constraint satisfaction problems. *J. Artif. Intell. Res.*, 43:173–210, 2012.

[7] Jean-Charles Régin : A filtering algorithm for constraints of difference in csps. pages 362–367, 1994.

[8] Richard P. Stanley : *Enumerative Combinatorics : Volume 1.* Cambridge University Press, New York, NY, USA, 2nd édition, 2011.

[9] Willem Jan van Hoeve : The alldifferent constraint : A survey. *CoRR*, cs.PL/0105015, 2001.