THÈSE

présentée à

L'UNIVERSITÉ DE LA MÉDITERRANÉE AIX-MARSEILLE II

École doctorale de Mathématiques et Informatique

par

Bruno GILLETA

pour obtenir le grade de

DOCTEUR ÈS SCIENCES

spécialité :

INFORMATIQUE

Placement des pentaminos par résolution de contraintes

14 novembre 2000

composition provisoire du jury :

M. Frédéric BENHAMOU	Rapporteur
Mme. Noëlle BLANC-GUERNALEC	Examinateur
M. Alain COLMERAUER	Directeur de thèse
M. Michel RUEHER	Rapporteur
M. Michel VAN CANEGHEM	Examinateur

Rapporteurs et membre du Jury

Frédéric Benhamou : Rapporteur
Professeur à l'Université de Nantes
Institut de Recherche en Informatique de Nantes
2 rue de la Houssinière
BP 92208
44322 NANTES

Noëlle Blanc-Guernalec : Examinateur
Maître de conférences à l'Université de la Méditerranée
Département de Mathématiques
Faculté des Sciences de Luminy
163, avenue de Luminy
13288 MARSEILLE Cedex 9

Alain Colmerauer : Directeur de thèse
Professeur à l'Université de la Méditerranée
Département d'Informatique
Faculté des Sciences de Luminy
163, avenue de Luminy
13288 MARSEILLE Cedex 9

Michel Rueher : Rapporteur
Professeur à l'Université de Nice-Sophia Antipolis
Université de Nice-Sophia Antipolis
Laboratoire I3S
ESSI (École Supérieure en Sciences Informatiques)
930 route des Colles
BP 145
06903 SOPHIA ANTIPOLIS

Michel Van Caneghem : Examinateur
Professeur à l'Université de la Méditerranée
Département d'Informatique
Faculté des Sciences de Luminy
163, avenue de Luminy
13288 MARSEILLE Cedex 9

Résumé

L'objet de cette thèse est le placement de 12 pentaminos pour former un parallélépipède rectangle d'un volume de 60 cubes unités. Un pentamino est un assemblage de 5 cubes unités posés sur un plan et connectés par au moins une face commune. Les 12 pentaminos correspondent au 12 formes possibles, ressemblant aux lettres F, I, L, P, N, T, U, V, W, X, Y, Z. Il existe des placements pour former des parallélépipèdes de dimensions $1\times3\times20$, $1\times4\times15$, $1\times5\times12$, $1\times6\times10$, $2\times3\times10$, $2\times5\times6$ et $3\times4\times5$. La thèse s'organise en quatre parties et une annexe.

(1) On expose tout d'abord la méthode générale pour exprimer de tels placements par des contraintes sur les entiers et comment résoudre ces contraintes à l'aide de résolutions locales et de propagation d'intervalles, exprimées sous forme de transformations de sous-contraintes.

(2) Puis on introduit une contrainte de « distance » qui se montre trop inefficace pour calculer tous les placements.

(3) On introduit alors d'autres contraintes plus élémentaires, comme la multiplication entière par une constante, la somme multiple d'entiers, la contrainte « être des entiers tous différents », une contrainte de translation et une contrainte de placement. On montre comment résoudre localement chacune de ces contraintes.

(4) Après avoir défini une stratégie de transformation de nos sous-contraintes, et une méthode pour éliminer les symétries, nous pouvons effectivement calculer tous les placements possibles de pentaminos et comparer nos temps à celui d'un pur algorithme énumératif. Nos temps restent malheureusement bien en dessous.

(5) Enfin on trouve en annexe la description de l'algorithme énumératif pur et surtout, sous formes de dessins, tous les placements possibles de pentaminos dans un parallélépipède rectangle d'un volume de 60 cubes unités.

Table des matières

Introduction **1**

1 Méthode générale **3**
1.1 Notations et définitions . . . 3
1.1.1 Intervalles et blocs . . . 3
1.1.2 Formule normalisée . . . 3
1.1.3 Translations et rotations . . . 4
1.1.4 Pentaminos . . . 4
1.2 Énoncé du problème . . . 5
1.3 L'algorithme général de résolution . . . 6
1.3.1 Règles de transformation . . . 6
1.3.2 Correction de l'algorithme . . . 6

2 Utilisation de la contrainte de distance **8**
2.1 Une contrainte de distances . . . 8
2.1.1 Simplification de la contrainte de distances . . . 10
2.1.2 Réduction de la contrainte de distance . . . 11
2.1.3 « Être des points distincts » . . . 11
2.2 Élimination des symétries . . . 12
2.3 Nouvelle formulation de la contrainte . . . 14
2.4 Résultats . . . 14

3 Utilisation d'autres contraintes plus élémentaires **15**
3.1 Simplification de la contrainte d'être des vecteurs distincts . . . 15
3.2 Réduction des contraintes élémentaires . . . 16
3.2.1 Multiplication entière par une constante . . . 16
3.2.2 Somme multiple d'entiers . . . 19
3.3 Réduction de la contrainte d'être des entiers distincts . . . 21
3.3.1 Algorithme . . . 22
3.3.2 Exemple . . . 25
3.4 Réduction de la contrainte de placement . . . 29
3.4.1 Contrainte de translation . . . 29
3.4.2 Contrainte de placement . . . 33

4 Résultats expérimentaux **35**
4.1 Stratégie de résolution 35
4.2 Premières solutions 36
4.3 Élimination des symétries en deux dimensions 37
4.4 Élimination des symétries en trois dimensions 39
4.4.1 Sans symétries pour les boîtes d'épaisseur 2 40
4.4.2 Quelques symétries en moins pour le problème $3\times4\times5$ 40
4.5 Illustration de quelques solutions 42

Conclusion **47**

Références **48**

Annexes **49**

A L'algorithme énumératif sans contraintes **49**

B Catalogue de solutions **53**

Introduction

Un *polyomino* (ou n-omino) désigne une figure simple composée de n carrés 1×1 connectés le long de leurs cotés. Le terme polyomino a été créé par Solomon W. Golomb en 1953 dans une présentation qu'il en fit au *Harvard Mathematics Club*, l'article [Gol54] présente différentes façons de paver un échiquier avec des polyominos de deux (les classiques dominos), trois, quatre et cinq carrés : les *pentaminos*. Mais le premier problème connu portant sur les pentaminos fut publié en 1907 dans les *Canterbury Puzzles* par Henry Ernest Dudeny [Dud07].

Dans [Gar58] Martin Gardner donne une épaisseur aux pentaminos, ce sont maintenant des pièces connexes formés de 5 cubes unitaires assemblés à plat. Il en existe 12 différents et leurs formes ressemblent aux 12 lettres F, I, L, P, N, T, U, V, W, X, Y et Z.

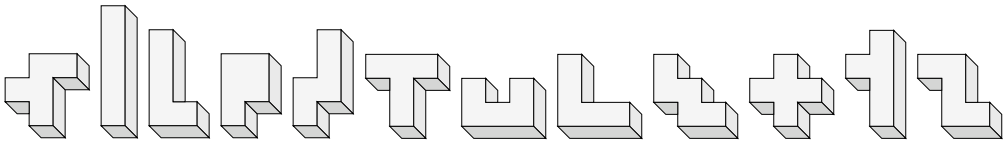


FIG. 1 – Les 12 pentaminos

Nous nous intéressons à placer de toutes les façons possibles ces 12 pentaminos dans un parallélépipède rectangle d'un volume de 60 cubes unités. Pour ceci, nous introduisons des contraintes aux nombres entiers dont les solutions sont précisément les placements recherchés. Nous résolvons ces contraintes par une méthode de réduction d'intervalles classique qui consiste à isoler l'ensemble des solutions dans plusieurs produits cartésiens d'intervalles de plus en plus réduits.

Organisation de ce document

Ce document comprend quatre chapitres et deux annexes :

Dans le premier chapitre, nous présentons le problème des pentaminos et l'énonçons sous la forme d'une contrainte composée de deux contraintes globales : une contrainte de « points distincts » qui impose à un ensemble de vecteurs de $\mathbf{Z}^3$ de prendre des valeurs distinctes deux à deux et une contrainte de « placement » qui impose à un ensemble de vecteurs de $\mathbf{Z}^3$ à être isométriques à une forme donnée. Nous présentons aussi notre algorithme général pour résoudre de telles contraintes.

Dans le second chapitre, nous décomposons ces deux contraintes globales à l'aide d'une

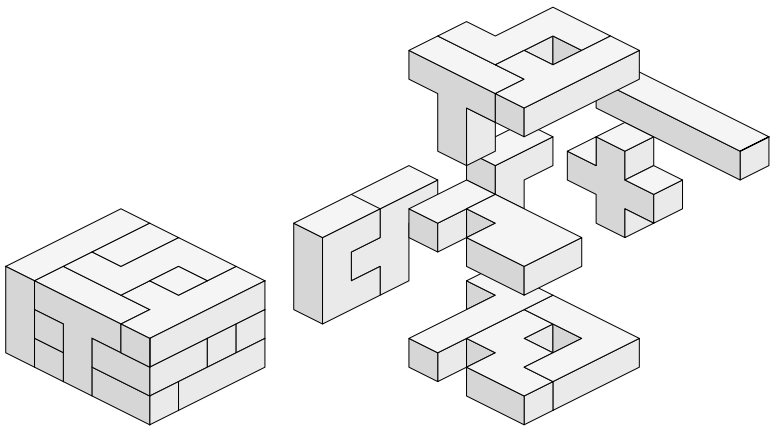


FIG. 2 – Exemple de placement des pentaminos dans une boîte de dimensions $3 \times 4 \times 5$

contrainte de « distances » qui fixe toutes les distances entre plusieurs vecteurs de $\mathbf{Z}^3$. Cette méthode, bien que fournissant des solutions à notre problème se révèle inefficace pour rechercher toutes les solutions.

Dans le troisième chapitre, nous décomposons cette fois-ci la contrainte d'être des vecteurs distincts au moyen d'une contrainte d'« entiers distincts » et des contraintes de « produit par une constante » et de « sommes multiples ». La contrainte de placement est maintenant traitée sans décomposition. Nous donnons également les méthodes de réduction de ces nouvelles contraintes. Cette nouvelle formulation permet de calculer l'intégralité des solutions aux différents problèmes de placement des pentaminos,

Le chapitre quatre décrit les résultats obtenus par cette dernière approche et les compare avec ceux d'un algorithme classique dit « de force brute ». Ce dernier reste malheureusement de loin la meilleure méthode pour résoudre le problème des pentaminos.

La première annexe décrit l'algorithme classique mis en concurrence avec notre méthode de programmation par contraintes.

La deuxième annexe donne sous forme de dessins tous les placements de pentaminos dans une boîte de dimensions $1\times3\times20$, $1\times4\times15$, $1\times5\times12$, $1\times6\times10$, $2\times3\times10$, $2\times5\times6$ et $3\times4\times5$.

Chapitre 1

Méthode générale

1.1 Notations et définitions

1.1.1 Intervalles et blocs

On désigne par $\mathbf{Z}$ l'ensemble des entiers. Si a et b sont des entiers, on appelle *intervalle* l'ensemble éventuellement vide des entiers x tels que $a \leq x$ et $x \leq b$ et on le note $[a, b]$ On note $\underline{I}$ (resp. $\overline{I}$) le plus petit (resp. le plus grand) élément de l'intervalle I quand il existe. Soit z un réel, on note $\lceil z \rceil$ (resp. $\lfloor z \rfloor$) le plus petit (resp. le plus grand) entier supérieur (resp. inférieur) ou égal à z. On appelle *bloc* tout produit cartésien fini $I_1 \times \cdots \times I_n$ d'intervalles I_i.

Si r est un sous-ensemble de $\mathbf{Z}^n$, la i-ème *projection* de r, notée $\text{proj}_i(r)$, est l'ensemble des entiers e tels qu'il existe $(x_1, \ldots, x_n) \in r$ avec $x_i = e$. L'*approximation* d'un ensemble de r, notée $\text{apx}(r)$ le plus petit produit cartésien d'intervalles d'entiers contenant r.

De l'ensemble agréables des propriétés [BO93] de l'application $r \mapsto \text{apx}(r)$ nous en retiendrons deux qui nous seront utiles dans différents calculs de pavés de la forme $\text{apx}(r \cap I_1 \times \cdots \times I_n)$:

$$\text{apx}(r_1 \cup \cdots \cup r_n) = \text{apx}(\text{apx}(r_1) \cup \cdots \cup \text{apx}(r_n)) \tag{1.1}$$

et

$$\text{apx}(r) = \text{apx}(\text{proj}_1(r)) \times \cdots \times \text{apx}(\text{proj}_m(r)) \,, \tag{1.2}$$

où r et les r_i sont des sous-ensembles de $\mathbf{Z}^n$ et $\text{proj}_i(r)$ est la *ième projection* de r, c'est-à-dire l'ensemble des éléments b de $\mathbf{Z}$ tels qu'il existe un n-uplet $(a_1, \ldots, a_n)$ de r avec $b = a_i$.

1.1.2 Formule normalisée

Une formule f est dite *normalisée* lorsque elle est écrite sous la forme suivante :

$$\begin{array}{l} \exists\, x_m \, \exists\, x_{m+1} \ldots \exists\, x_n \\ \left(\begin{array}{ll} & x_1 \in I_1 \wedge \cdots \wedge x_n \in I_n \\ \wedge & p_1 \wedge \cdots \wedge p_k \\ \wedge & q_1 \wedge \cdots \wedge q_l \\ \wedge & \mathit{vrai} \end{array} \right) \end{array}$$

où les x_i sont des variables ; les I_i sont des intervalles d'entiers ; les p_i sont des contraintes de la forme $(y_1, \ldots, y_m) \in r$, r étant un ensemble de m-uplets d'entiers et les y_i des variables distinctes prises dans l'ensemble des x_i ; et les q_i sont des formules normalisées sans occurrence des x_i sous la portée d'un quantificateur.

1.1.3 Translations et rotations

Par *matrice de rotation* M nous entendons une matrice dont les éléments sont -1, 0 ou 1, qui est de la forme

$$M = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{32} & \theta_{33} \end{bmatrix} \tag{1.3}$$

avec au plus un élément non nul par ligne et par colonne, et dont le déterminant vaut 1.

Appelons *point* tout élément de $\mathbf{Z}^3$ et *configuration* tout ensemble fini de points. Soient

$$\begin{aligned} f &= \{(a_1, b_1, c_1), \ldots, (a_n, b_n, c_n)\}, \\ f' &= \{(a'_1, b'_1, c'_1), \ldots, (a'_n, b'_n, c'_n)\} \end{aligned}$$

deux configurations de n points. On dit que f' est une *translatée* de f si il existe des éléments c, d, e de $\mathbf{Z}$ tels que, pour chaque i pris dans $[1, n]$, on ait

$$\begin{aligned} a'_i &= a_i + c, \\ b'_i &= b_i + d, \\ c'_i &= c_i + e. \end{aligned}$$

On dit que f' est une *pivotée* de f s'il existe une matrice de rotation M de la forme (1.3) telle que, pour chaque i pris dans $[1, n]$, on ait

$$\begin{aligned} a'_i &= \theta_{11} a_i + \theta_{12} b_i + \theta_{13} c_i, \\ b'_i &= \theta_{21} a_i + \theta_{22} b_i + \theta_{23} c_i, \\ c'_i &= \theta_{31} a_i + \theta_{32} b_i + \theta_{33} c_i. \end{aligned}$$

On note *translatés*(f) l'ensemble des translatées de la configuration f et *pivotés*(f) l'ensemble des pivotées de f. On peut alors définir la classe de la configuration f comme l'ensemble *classe*(f) des configurations qui sont égales à f à une translation et une rotation près. On a donc

$$\textit{classe}(f) = \bigcup_{g \in \textit{pivotés}(f)} \textit{translatés}(g) \tag{1.4}$$

1.1.4 Pentaminos

Un *pentamino* est une configuration $\{(a_1, b_1, c_1), \ldots, (a_5, b_5, c_5)\}$ de 5 points qui satisfait aux conditions de planarité et de connexité suivantes :

- les a_i sont tous égaux ou les b_i sont tous égaux ou les c_i sont tous égaux,
- il existe une suite $i_1, \ldots, i_k$ d'entiers pris dans 1..5 qui fait intervenir chaque élément de 1..5 et qui est telle que $(a_{i_{j-1}} - a_{i_j})^2 + (b_{i_{j-1}} - b_{i_j})^2 + (c_{i_{j-1}} - c_{i_j})^2 = 1$, pour chaque j pris dans 2..k.

On se donne les 12 pentaminos de bases $\pi_i, \ldots, \pi_{12}$ suivants :

$$\begin{array}{rcl}
\pi_1 & = & \{(1,2,0),(2,2,0),(3,2,0),(3,1,0),(2,3,0)\},\\
\pi_2 & = & \{(1,1,0),(2,1,0),(3,1,0),(4,1,0),(5,1,0)\},\\
\pi_3 & = & \{(1,1,0),(1,2,0),(2,2,0),(3,2,0),(4,2,0)\},\\
\pi_4 & = & \{(2,1,0),(1,1,0),(1,2,0),(2,2,0),(3,2,0)\},\\
\pi_5 & = & \{(1,1,0),(2,1,0),(2,2,0),(3,2,0),(4,2,0)\},\\
\pi_6 & = & \{(1,1,0),(1,2,0),(1,3,0),(2,2,0),(3,2,0)\},\\
\pi_7 & = & \{(1,2,0),(1,1,0),(2,1,0),(3,1,0),(3,2,0)\},\\
\pi_8 & = & \{(1,1,0),(2,1,0),(3,1,0),(3,2,0),(3,3,0)\},\\
\pi_9 & = & \{(1,1,0),(2,1,0),(2,2,0),(3,2,0),(3,3,0)\},\\
\pi_{10} & = & \{(1,2,0),(2,2,0),(2,3,0),(2,1,0),(2,3,0)\},\\
\pi_{11} & = & \{(1,1,0),(2,1,0),(3,1,0),(4,1,0),(3,2,0)\},\\
\pi_{12} & = & \{(1,1,0),(2,1,0),(2,2,0),(2,3,0),(3,3,0)\}.
\end{array} \tag{1.5}$$

L'ensemble de sous-ensembles $\{\mathit{classe}(\pi_1), \ldots, \mathit{classe}(\pi_{12})\}$ est alors une partition de l'ensemble des pentaminos en 12 classes. En accord avec les formes physiques qui leurs sont associées les ensembles $\mathit{classe}(\pi_1), \ldots, \mathit{classe}(\pi_{12})$ sont souvent notées F, I, L, P, N, T, U, V, W, X, Y, Z.

1.2 Énoncé du problème

Étant donnés trois entiers positifs a, b et c tels que $abc = 60$, nous voulons partitionner l'ensemble $[0, a-1] \times [0, b-1] \times [0, c-1]$ de *points* en 12 pentaminos $f_1, \ldots, f_n$ appartenant respectivement aux ensembles $\mathit{classe}(\pi_1), \ldots, \mathit{classe}(\pi_{12})$. En posant, pour chaque $i \in 1..12$,

$$f_i = \{(x_{5i-4}, y_{5i-4}, z_{5i-4}), \ldots, (x_{5i-0}, y_{5i-0}, z_{5i-0})\}$$

avec $((x_{5i-4}, y_{5i-4}, z_{5i-4}), \ldots, (x_{5i-0}, y_{5i-0}, z_{5i-0}))$ lexicographiquement croissant, le calcul d'une partition revient au calcul d'une solution en $x_1, y_1, z_1, \ldots, x_{60}, y_{60}, z_{60}$ de la contrainte

$$\begin{pmatrix}
& x_1 \in [0, a-1] \wedge \ldots \wedge x_{60} \in [0, a-1] \\
\wedge & y_1 \in [0, b-1] \wedge \ldots \wedge y_{60} \in [0, b-1] \\
\wedge & z_1 \in [0, c-1] \wedge \ldots \wedge z_{60} \in [0, c-1] \\
\wedge & (x_1, y_1, z_1, \ldots, x_{60}, y_{60}, z_{60}) \in \mathit{PointsDistincts} \\
\wedge & (x_1, y_1, z_1, \ldots, x_5, y_5, z_5) \in \mathit{Classe}(\pi_1) \\
\wedge & \quad \vdots \\
\wedge & (x_{56}, y_{56}, z_{56}, \ldots, x_{60}, y_{60}, z_{60}) \in \mathit{Classe}(\pi_{12})
\end{pmatrix}$$

où

- *PointsDistincts* est l'ensemble des $3n$-uplets $(a_1, b_1, c_1, \ldots, a_n, b_n, c_n)$ d'entiers qui sont tels que, pour tout i, j pris dans $1..n$, le point (a_i, b_i, c_i) soit distinct du point (a_j, b_j, c_j), si i est distinct de j,
- *Classe*(f), avec f une configuration de n points, est l'ensemble des $3n$-uplets $(a_1, b_1, c_1, \ldots, a_n, b_n, c_n)$ d'entiers qui sont tels que

  $\{(a_1, b_1, c_1), \ldots, (a_n, b_n, c_n)\} \in \mathit{classe}(f)$ et
  $((a_1, b_1, c_1), \ldots, (a_n, b_n, c_n))$ est lexicographiquement croissant,

– les π_i sont les pentaminos de base définis en (1.5).

1.3 L'algorithme général de résolution

Soit f une formule normalisée. L'algorithme de résolution utilisé consiste à appliquer à la formule $f \vee \mathit{faux}$ une suite de transformations qui à chaque étape produisent une formule $f_1 \vee \ldots \vee f_n \vee \mathit{faux}$ équivalente, où les f_i sont des formules normalisées. Après un nombre fini de transformations, nous obtenons une formule équivalente à f de la forme :

$$\begin{cases} \mathit{faux}, \text{ si } f \text{ est équivalente à } \mathit{faux} \\ \\ \mathit{vrai} \vee \cdots \vee \mathit{vrai} \vee \mathit{faux}, \text{ si } f \text{ est équivalente à } \mathit{vrai} \\ \\ \begin{pmatrix} & x_1 \in \{a_1\} \\ \wedge & \cdots \\ \wedge & x_n \in \{a_n\} \\ \wedge & \mathit{vrai} \end{pmatrix} \vee \cdots \vee \begin{pmatrix} & x_1 \in \{z_1\} \\ \wedge & \cdots \\ \wedge & x_n \in \{z_n\} \\ \wedge & \mathit{vrai} \end{pmatrix} \vee \mathit{faux}, \text{ dans les autres cas.} \end{cases}$$

où $\{(a_1, \ldots, a_n), \ldots, (z_1, \ldots, z_n)\}$ est l'ensemble des n-uplets de valeurs pour lesquelles la formule obtenue en substituant dans f ces valeurs aux x_i est équivalente à *vrai*.

Les opérateurs $\wedge$ et $\vee$ sont associatifs et commutatifs. Pour des raisons de commodité, nous omettons d'écrire certaines parenthèses dans nos formules.

1.3.1 Règles de transformation

Les six transformations de la table 1.1 sont appliquées sur une formule $f_1 \vee \ldots \vee f_n \vee \mathit{faux}$ où les f_i sont des formules normalisées ; quand une de ses sous-formules est sous la forme du membre gauche d'une de ces règles, on la remplace par le membre droit qui lui est équivalent. La formule $f'_1 \vee \ldots \vee f'_m \vee \mathit{faux}$ ainsi obtenue est équivalente à $f_1 \vee \ldots \vee f_n \vee \mathit{faux}$ et les f'_i sont des formules normalisées.

Dans ces six transformations, p et q sont des conjonctions de formules, r est un sous-ensemble de $\mathbf{Z}^n$, les entiers $a_1, \ldots, a_n$ sont tels que $(a_1, \ldots, a_n) \in r$, les intervalles $I_1, \ldots, I_n$ sont non vides et $I'_1, \ldots, I'_n$ sont tels que $I'_1 \times \cdots \times I'_n = \text{apx}(I_1 \times \cdots \times I_n \cap r)$ et il qu'il existe un i tel que $I'_i \neq I_i$, les intervalles I, J, K sont non vides et tels que $I = J \cup K$ et $J \cap K = \emptyset$ et q est sans occurrence de x_i.

1.3.2 Correction de l'algorithme

On associe à toute formule de la forme $f_1 \vee \ldots \vee f_n \vee \mathit{faux}$ où les f_i sont des formules normalisées le couple d'entiers positifs (n_1, n_2) où :

– n_1 est la somme sur i des produit des carrés de la taille des intervalles apparaissant dans chaque formule normalisée f_i,

– n_2 est le nombre d'occurrences du $\wedge$ dans la formule $f_1 \vee \ldots \vee f_n \vee \mathit{faux}$

Considérons le couple d'entiers positifs (n'_1, n'_2) associé à la formule $f'_1 \vee \ldots \vee f'_m \vee \mathit{faux}$ obtenue par application de la (i)-ème transformation

– $n'_1 < n_1$ pour $i \in \{5, 6\}$,

$$(1)\qquad q \wedge \overset{\exists x_1 \ldots \exists x_n}{\begin{pmatrix} & p \\ \wedge & \textit{vrai} \end{pmatrix}} \rightarrow \overset{\exists x_1 \ldots \exists x_n}{\begin{pmatrix} & p \\ \wedge & q \end{pmatrix}}$$

$$(2)\qquad \begin{pmatrix} & (x_1, \ldots, x_n) \in r \\ \wedge & x_1 \in \{a_1\} \\ \wedge & \cdots \\ \wedge & x_n \in \{a_n\} \end{pmatrix} \rightarrow \begin{pmatrix} & x_1 \in \{a_1\} \\ \wedge & \cdots \\ \wedge & x_n \in \{a_n\} \end{pmatrix}$$

$$(3)\qquad \textit{faux} \vee \overset{\exists x_1 \ldots \exists x_n}{\begin{pmatrix} & x \in \emptyset \\ \wedge & p \end{pmatrix}} \rightarrow \textit{faux}$$

$$(4)\qquad \overset{\exists x_1 \ldots \exists x_n}{\begin{pmatrix} & x_1 \in I_1 \\ \wedge & p \end{pmatrix}} \rightarrow \overset{\exists x_2 \ldots \exists x_n}{\begin{pmatrix} p \end{pmatrix}}$$

$$(5)\qquad \begin{pmatrix} & (x_1, \ldots, x_n) \in r \\ \wedge & x_1 \in I_1 \\ \wedge & \cdots \\ \wedge & x_n \in I_n \end{pmatrix} \rightarrow \begin{pmatrix} & (x_1, \ldots, x_n) \in r \\ \wedge & x_1 \in I'_1 \\ \wedge & \cdots \\ \wedge & x_n \in I'_n \end{pmatrix}$$

$$(6)\qquad \textit{faux} \vee \overset{\exists x_1 \cdots x_n}{\begin{pmatrix} & x \in I \\ \wedge & p \end{pmatrix}} \rightarrow \textit{faux} \vee \overset{\exists x_1 \ldots x_n}{\begin{pmatrix} & x \in J \\ \wedge & p \end{pmatrix}} \vee \overset{\exists x_1 \ldots x_n}{\begin{pmatrix} & x \in K \\ \wedge & p \end{pmatrix}}$$

TAB. 1.1 – Les règles de réécriture

– $n'_1 \leq n_1$ et $n'_2 < n_2$ pour $i \in \{1, 2, 3, 4\}$.

Toute suite de transformations que l'on applique sur une formule $f \vee \textit{faux}$ où f est normalisée produit donc une suite de couples (n_1, n_2) strictement décroissante lexicographiquement ; n_1 et n_2 étant positifs, cette suite converge et le nombre de transformation que l'on peut appliquer itérativement sur $f \vee \textit{faux}$ est donc fini.

Après avoir épuisé toutes les transformations possibles, notre formule sera sous la forme $f_1 \vee \ldots \vee f_n \vee \textit{faux}$, n pouvant être nul, où les f_i sont des formules normalisées sans contrainte de la forme $x \in I$ avec I de taille supérieure à 1 (on pourrait appliquer la règle 6) ou nulle (on pourrait appliquer la règle 3), sans contrainte de la forme $(x_1, \ldots, x_n) \in r$ (la valeur de chaque variable étant fixée, on pourrait appliquer soit la règle 2, soit la règle 5), sans quantificateur en tête (la valeur de chaque variable quantifiée étant fixée, on pourrait appliquer la règle 4) ni dans le corps de la formule (on pourrait appliquer la règle 1).

Les f_i sont donc toutes de la forme $(x_1 \in \{a_1\} \wedge \ldots \wedge x_m \in \{a_m\} \wedge \textit{vrai})$ et ont toutes le même nombre de variables (qui peut être nul) qui est le nombre de variables libre de la formule initiale.

Chapitre 2

Expression du problème avec une contrainte de distance

Une des première modélisation du problème a été de spécifier la forme des pentaminos à l'aide d'une contrainte de distance, la donnée des distances entre les couples de points d'un pentamino étant suffisante pour définir sa forme.

2.1 Une contrainte de distances

Étant donné $n(n-1)/2$ entiers positifs $d_{1,2}, d_{1,3}, \ldots, d_{1,n}, d_{2,3}, \ldots, d_{n-1,n}$, on définit par *Distance*$(d_{1,2}, d_{1,3}, \ldots, d_{1,n}, d_{2,3}, \ldots, d_{n-1,n})$ l'ensemble des $3n$-uplets d'entiers $(x_1, y_1, z_1, \ldots, x_n, y_n, z_n)$ qui sont tels que :

$$\left(\begin{array}{lrcl} & d_{1,2} & = & (x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2 \\ \wedge & d_{1,3} & = & (x_1-x_3)^2+(y_1-y_3)^2+(z_1-z_3)^2 \\ \wedge & & \vdots & \\ \wedge & d_{n-1,n} & = & (x_{n-1}-x_n)^2+(y_{n-1}-y_n)^2+(z_{n-1}-z_n)^2 \end{array}\right)$$

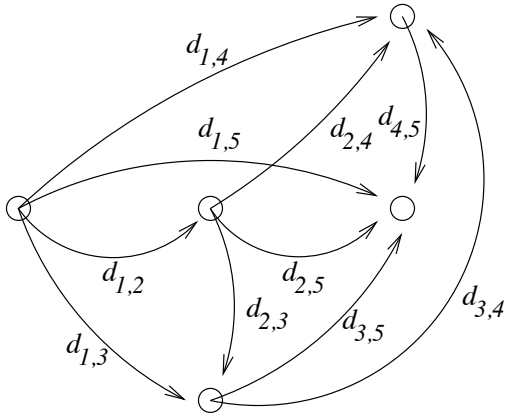


FIG. 2.1 – Calcul des $d_{i,j}$ pour le pentamino F

Nommons $(a_{5i-4}, b_{5i-4}, c_{5i-4})$, ..., $(a_{5i-0}, b_{5i-0}, c_{5i-0})$ les coordonnées des 5 points du pentamino de base π_i, prises dans l'ordre lexicographique, pour tout i pris entre 1 et 12. Posons pour tout j pris entre 1 et 4 et pour tout k inférieur à j pris entre 0 et 3 :

$$d_{5i-j,5i-k} = (a_{5i-j} - a_{5i-k})^2 + (b_{5i-j} - b_{5i-k})^2 + (c_{5i-j} - c_{5i-k})^2$$

En posant

$$\mathcal{P}_i = \begin{bmatrix} d_{5i-4,5i-3} & d_{5i-4,5i-2} & d_{5i-4,5i-1} & d_{5i-4,5i} \\ & d_{5i-3,5i-2} & d_{5i-3,5i-1} & d_{5i-3,5i} \\ & & d_{5i-2,5i-1} & d_{5i-2,5i} \\ & & & d_{5i-1,5i} \end{bmatrix}$$

nous obtenons à partir des coordonnées de nos pentaminos de base les 12 matrices triangulaires supérieures définissant la forme des pentaminos :

$$\mathcal{P}_1 = \begin{bmatrix} 1 & 2 & 5 & 4 \\ & 1 & 2 & 1 \\ & & 5 & 2 \\ & & & 1 \end{bmatrix} \quad \mathcal{P}_2 = \begin{bmatrix} 1 & 4 & 9 & 16 \\ & 1 & 4 & 9 \\ & & 1 & 4 \\ & & & 1 \end{bmatrix} \quad \mathcal{P}_3 = \begin{bmatrix} 1 & 2 & 5 & 10 \\ & 1 & 4 & 9 \\ & & 1 & 4 \\ & & & 1 \end{bmatrix}$$

$$\mathcal{P}_4 = \begin{bmatrix} 1 & 1 & 2 & 5 \\ & 2 & 1 & 4 \\ & & 1 & 2 \\ & & & 1 \end{bmatrix} \quad \mathcal{P}_5 = \begin{bmatrix} 1 & 2 & 5 & 10 \\ & 1 & 2 & 5 \\ & & 1 & 4 \\ & & & 1 \end{bmatrix} \quad \mathcal{P}_6 = \begin{bmatrix} 1 & 4 & 2 & 5 \\ & 1 & 1 & 4 \\ & & 2 & 5 \\ & & & 1 \end{bmatrix}$$

$$\mathcal{P}_7 = \begin{bmatrix} 1 & 1 & 4 & 5 \\ & 2 & 5 & 4 \\ & & 1 & 2 \\ & & & 1 \end{bmatrix} \quad \mathcal{P}_8 = \begin{bmatrix} 1 & 4 & 5 & 8 \\ & 1 & 2 & 5 \\ & & 1 & 4 \\ & & & 1 \end{bmatrix} \quad \mathcal{P}_9 = \begin{bmatrix} 1 & 2 & 5 & 8 \\ & 1 & 2 & 5 \\ & & 1 & 2 \\ & & & 1 \end{bmatrix}$$

$$\mathcal{P}_{10} = \begin{bmatrix} 2 & 1 & 2 & 4 \\ & 1 & 4 & 2 \\ & & 1 & 1 \\ & & & 2 \end{bmatrix} \quad \mathcal{P}_{11} = \begin{bmatrix} 1 & 4 & 5 & 9 \\ & 1 & 2 & 4 \\ & & 1 & 1 \\ & & & 2 \end{bmatrix} \quad \mathcal{P}_{12} = \begin{bmatrix} 1 & 2 & 5 & 8 \\ & 1 & 4 & 5 \\ & & 1 & 2 \\ & & & 1 \end{bmatrix}$$

Soit $f = \{(x_1, y_1, z_1), \ldots, (x_5, y_5, z_5)\}$ une configuration de 5 points. Nous avons maintenant une condition suffisante pour que f appartienne à *classe* (π_i) :

$$(x_1, y_1, z_1, \ldots, x_5, y_5, z_5) \in \mathit{Distance}(d_{5i-4,5i-3}, d_{5i-4,5i-2}, , \ldots, d_{5i-1,5i})$$
$$\Downarrow$$
$$f \in \mathit{classe}\ (\pi_i)$$

Soit $f = \{(x_1, y_1, z_1), \ldots, (x_5, y_5, z_5)\}$ une configuration de 5 points appartiennant à *classe* (π_i). Il existe alors une permutation σ : 1..5 $\mapsto$ 1..5 telle que

$$(x_{\sigma(1)}, y_{\sigma(1)}, z_{\sigma(1)}, \ldots, x_{\sigma(5)}, y_{\sigma(5)}, z_{\sigma(5)}) \in \mathit{Distance}(d_{5i-4,5i-3}, d_{5i-4,5i-2}, , \ldots, d_{5i-1,5i})$$

Il s'ensuit que pour toute configuration $f = \{(x_1, y_1, z_1), \ldots, (x_5, y_5, z_5)\}$

$$\bigvee_{\sigma\in\Sigma} \Big((x_{\sigma(1)}, y_{\sigma(1)}, z_{\sigma(1)}, \ldots, x_{\sigma(5)}, y_{\sigma(5)}, z_{\sigma(5)}) \in \mathit{Distance}(d_{5i-4,5i-3}, d_{5i-4,5i-2}, , \ldots, d_{5i-1,5i})\Big)$$
$$\Updownarrow$$
$$f \in \mathit{classe}\ (\pi_i)$$

ce qui entraîne

$$\bigvee_{\sigma\in\Sigma} \Big((x_{\sigma(1)}, y_{\sigma(1)}, z_{\sigma(1)}, \ldots, x_{\sigma(5)}, y_{\sigma(5)}, z_{\sigma(5)}) \in \mathit{Distance}(d_{5i-4,5i-3}, d_{5i-4,5i-2}, , \ldots, d_{5i-1,5i})\Big)$$
$$\text{et } ((x_1, y_1, z_1), \ldots, (x_5, y_5, z_5)) \text{ est lexicographiquement croissant}$$
$$\Updownarrow$$
$$(x_1, y_1, z_1, \ldots, x_5, y_5, z_5) \in \mathit{Classe}(\pi_i)$$

avec Σ l'ensemble des permutations $1..5 \mapsto 1..5$

La contrainte de distance est donc suffisante pour exprimer qu'une configuration appartient à *classe*(π_i) mais elle n'entraîne pas l'appartenance à *Classe*(π_i). Nous alons dans ce chapitre substituer aux contraintes

$$(x_{5i-4}, y_{5i-4}, z_{5i-4}, \ldots, x_{5i}, y_{5i}, z_{5i}) \in \mathit{Classe}(\pi_i)$$

de notre problème les nouvelles contraintes

$$(x_{5i-4}, y_{5i-4}, z_{5i-4}, \ldots, x_{5i}, y_{5i}, z_{5i}) \in \mathit{Distance}(d_{5i-4,5i-3}, d_{5i-4,5i-2}, \ldots, d_{5i-1,5i})\,.$$

Cette substitution nous pose un problème : elle génère des solutions composées de configurations identiques, bien que différentes. Ces solution ne différent que par l'ordre de placement des cubes d'un même pentamino. Il nous faut alors éliminer ces solutions redondantes introduites par les symétries internes de certains pentaminos.

2.1.1 Simplification de la contrainte de distances

Nous n'avons pas trouvé d'algorithme efficace permettant de réduire parfaitement la contrainte de distances. Pour pouvoir la traiter, nous avons décomposé cette contrainte de distances en plusieurs petites contraintes de distance ne faisant intervenir que des couples de points :

$$(x_1, y_1, z_1, \ldots x_n, y_n, z_n) \in \mathit{Distance}(d_{1,2}, d_{1,3}, \ldots, d_{n-1,n})$$
$$\Downarrow$$
$$\begin{pmatrix} & (x_1, y_1, z_1, x_2, y_2, z_2) \in \text{distance}(d_{1,2}) \\ \wedge & (x_1, y_1, z_1, x_3, y_3, z_3) \in \text{distance}(d_{1,3}) \\ \wedge & \vdots \\ \wedge & (x_{n-1}, y_{n-1}, z_{n-1}, x_n, y_n, z_n) \in \text{distance}(d_{n-1,n}) \end{pmatrix}$$

où pour tout entier d, distance(d) est l'ensemble des $2n$-uplets d'entiers $(x_1, \ldots, x_n, y_1, \ldots, y_n)$ qui sont tels que

$$(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2 \;=\; d$$

2.1.2 Réduction de la contrainte de distance

Soit $2n$ intervalles d'entiers $I_1, \ldots, I_n, J_1, \ldots, J_n$, on cherche à calculer les plus petits (au sens de l'inclusion) intervalles d'entiers $I'_1, \ldots, I'_n, J'_1, \ldots, J'_n$ qui sont tels que

$$I_1 \times \cdots \times J_n \cap \text{distance}(d) \;=\; I'_1 \times \cdots \times J'_n \cap \text{distance}(d)$$

Pour tout i compris entre 1 et n, on calcule tout d'abord les intervalles Δ_i égaux aux $I_i - J_i$:

$$\Delta_i = I_i - J_i = I_i + (-J_i) = \left[\underline{I_i} - \overline{J_i}, \overline{I_i} - \underline{J_i}\right]$$

suivi du carré Δ_i^2 de ces intervalles :

$$\Delta_i^2 = \begin{cases} \left[\underline{\Delta_i}^2, \overline{\Delta_i}^2\right] & \text{si } \underline{\Delta_i} \geq 0 \\ \left[\overline{\Delta_i}^2, \underline{\Delta_i}^2\right] & \text{si } \overline{\Delta_i} \leq 0 \\ \left[0, \max\left(\underline{\Delta_i}^2, \overline{\Delta_i}^2\right)\right] & \text{sinon} \end{cases}$$

ce qui nous permet de rechercher les n-uplets d'entiers $(a_1, \ldots, a_n)$ qui sont tels que $(a_1^2, \ldots, a_n^2) \in \Delta_1^2 \times \cdots \times \Delta_n^2$ et $a_1^2 + \cdots + a_n^2 = d$. Soit $\Delta'^2_1 \times \cdots \times \Delta'^2_n$ l'enveloppe entière de l'ensemble de tous les n-uplets $(a_1^2, \ldots, a_n^2)$. Les bornes inférieures et supérieures des Δ'^2_1 étant des carrés d'entiers, on calcule alors maintenant les $\Delta'_i = \sqrt{\Delta'^2_i}$ de la façon suivante :

$$\Delta'_i = \begin{cases} \left[\sqrt{\underline{\Delta'^2_i}}, \sqrt{\overline{\Delta'^2_i}}\right] & \text{si } \underline{\Delta_i} \geq 0 \\ \left[\sqrt{-\overline{\Delta'^2_i}}, -\sqrt{\underline{\Delta'^2_i}}\right] & \text{si } \overline{\Delta_i} \leq 0 \\ \text{apx}\left(\Delta_i \cap \left(\left[-\sqrt{\overline{\Delta'^2_i}}, -\sqrt{\underline{\Delta'^2_i}}\right] \cup \left[\sqrt{\underline{\Delta'^2_i}}, \sqrt{\overline{\Delta'^2_i}}\right]\right)\right) & \text{sinon} \end{cases}$$

Enfin, sachant que $\Delta_i = I_i - J_i$, on calcule pour tout i entre 1 et n

$$I'_i = I_i \cap (\Delta'_i - J_i)$$
$$J'_i = J_i \cap (I_i - \Delta'_i)$$

2.1.3 « Être des points distincts »

Deux points (x_i, y_i, z_i) et (x_j, y_j, z_j) ne sont pas confondus si et seulement si la distance entre ces deux points est non nulle. Pour exprimer cette contrainte, nous utilisons une nouvelle contrainte de distance, mais cette fois-ci, on exprimera juste que la distance est positive et non plus égale à une constante :

$$(x_i, y_i, z_i) \neq (x_j, y_j, z_j)$$
$$\Downarrow$$
$$(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 > 0$$

Pour résoudre cette contrainte de distances positives, nous utilisons les mêmes calculs que précédemment, mais en recherchant cette fois-ci les n-uplets d'entiers $(a_1, \ldots, a_n)$ qui sont tels que $(a_1^2, \ldots, a_n^2) \in \Delta_1^2 \times \cdots \times \Delta_n^2$ et $a_1^2 + \cdots + a_n^2 > 0$.

Pour que les 60 points de notre problème soient distincts deux à deux, nous posons alors une contrainte de ce type pour tous les couples de points :

$$(x_1, y_1, z_1, \ldots, x_{60}, y_{60}, z_{60}) \in \mathit{PointsDistincts}$$
$$\Downarrow$$
$$\begin{pmatrix} \wedge & (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 > 0 \\ \wedge & (x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2 > 0 \\ \wedge & \vdots \\ \wedge & (x_{59} - x_{60})^2 + (y_{59} - y_{60})^2 + (z_{59} - z_{60})^2 > 0 \end{pmatrix}$$

2.2 Élimination des symétries

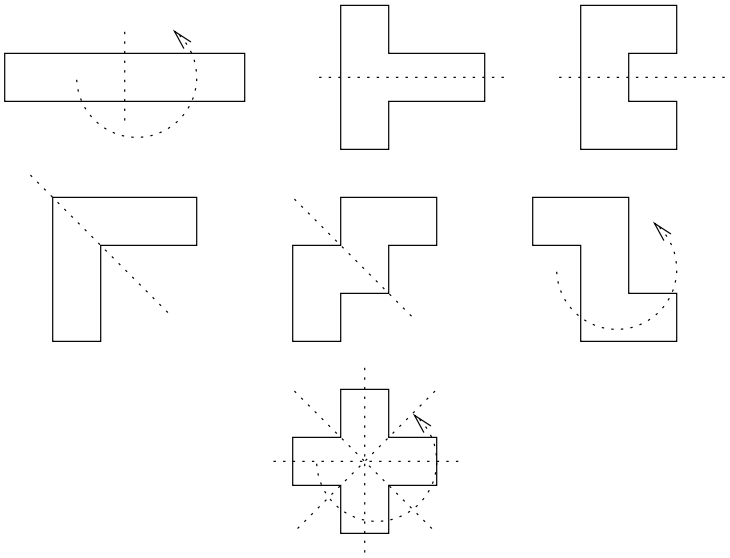


FIG. 2.2 – Symétries internes des pièces I, T, U, V, W, Z et X

Les pentaminos I, T, U, V, W, X et Z présentent des symétries internes (Figure 2.2) ; lors de la recherche des solutions à notre problème, les mêmes placements de pentaminos apparaîtront plusieurs fois, comprenant chacun les mêmes orientations de pentaminos, mais certaines orientations se distinguant par une permutation du placement des cubes la composant. Pour éviter ce problème, il nous faut rajouter des contraintes qui vont forcer les cubes de ces pentaminos à ne pas prendre d'orientations symétriques d'autres.

Les symétries internes des pièces I, T, U et Z sont celles qui s'éliminent le plus facilement : il suffit de contraindre les points du I, de la barre supérieure du T, de la base du U et

de l'axe du Z à être alignés par x, y ou bien z (suivant l'orientation de la pièce) croissants :

Pentamino	*Contrainte*
I	$x_1 \leq x_5 \wedge y_1 \leq y_5 \wedge z_1 \leq z_5$
T	$x_{26} \leq x_{28} \wedge y_{26} \leq y_{28} \wedge z_{26} \leq z_{28}$
U	$x_{31} \leq x_{36} \wedge y_{31} \leq y_{36} \wedge z_{31} \leq z_{36}$
Z	$x_{57} \leq x_{59} \wedge y_{57} \leq y_{59} \wedge z_{57} \leq z_{59}$

Le V et le W présentent par contre une symétrie par rapport à un axe diagonal, pour éliminer les placements de points du pentamino symétriques, on impose au vecteur perpendiculaire au plan de placement du pentamino à ne prendre qu'un sens sur deux, en imposant par exemple à ses coordonnées d'être positives, à l'aide d'un simple produit vectoriel :

Pentamino	*Contrainte*
V	$(y_{40} - y_{38}) \times (z_{36} - z_{38}) - (y_{36} - y_{38}) \times (z_{40} - z_{38}) \geq 0$ $\wedge \quad (x_{36} - x_{38}) \times (z_{40} - z_{38}) - (x_{40} - x_{38}) \times (z_{36} - z_{38}) \geq 0$ $\wedge \quad (x_{40} - x_{38}) \times (y_{36} - y_{38}) - (x_{36} - x_{38}) \times (y_{40} - y_{38}) \geq 0$
W	$(y_{45} - y_{43}) \times (z_{41} - z_{43}) - (y_{41} - y_{43}) \times (z_{45} - z_{43}) \geq 0$ $\wedge \quad (x_{41} - x_{43}) \times (z_{45} - z_{43}) - (x_{45} - x_{43}) \times (z_{41} - z_{43}) \geq 0$ $\wedge \quad (x_{45} - x_{43}) \times (y_{41} - y_{43}) - (x_{41} - x_{43}) \times (y_{45} - y_{43}) \geq 0$

Le pentamino X combine quand à lui toutes les précédentes symétries internes. Étant symétrique par rapport à son axe horizontal et à son axe vertical, on contraint les points de ces deux barres à être alignés par x, y ou bien z croissants, comme pour les pièces I, T, U et Z ; étant symétrique par rapport aux deux diagonales passant par son centre, on contraint le vecteur perpendiculaire à son plan de placement à avoir toutes ses coordonnées positives, comme pour les pentaminos V et W :

Pentamino	*Contrainte*
X	$x_{46} \leq x_{50} \wedge y_{46} \leq y_{50} \wedge z_{46} \leq z_{50}$ $\wedge \quad x_{47} \leq x_{49} \wedge y_{47} \leq y_{49} \wedge z_{47} \leq z_{49}$ $\wedge \quad (y_{48} - y_{47}) \times (z_{46} - z_{47}) - (y_{46} - y_{47}) \times (z_{48} - z_{47}) \geq 0$ $\wedge \quad (x_{46} - x_{47}) \times (z_{48} - z_{47}) - (x_{48} - x_{47}) \times (z_{46} - z_{47}) \geq 0$ $\wedge \quad (x_{48} - x_{47}) \times (y_{46} - y_{47}) - (x_{46} - x_{47}) \times (y_{48} - y_{47}) \geq 0$

Toutes les symétries internes des pièces sont ainsi éliminées.

2.3 Nouvelle formulation de la contrainte

Après application de ces décompositions et ajout de ces nouvelles contraintes, notre contrainte se trouve sous la forme suivante :

$$
\begin{pmatrix}
 & (x_1, y_1, z_1, x_2, y_2, z_2) \in \text{distance}(d_{1,2}) \\
\wedge & (x_1, y_1, z_1, x_3, y_3, z_3) \in \text{distance}(d_{1,3}) \\
\wedge & \quad \vdots \\
\wedge & (x_4, y_4, z_4, x_5, y_5, z_5) \in \text{distance}(d_{4,5}) \\
\wedge & \quad \vdots \\
\wedge & (x_{56}, y_{56}, z_{56}, x_{57}, y_{57}, z_{57}) \in \text{distance}(d_{56,57}) \\
\wedge & (x_{56}, y_{56}, z_{56}, x_{58}, y_{58}, z_{58}) \in \text{distance}(d_{56,58}) \\
\wedge & \quad \vdots \\
\wedge & (x_{59}, y_{59}, z_{59}, x_{60}, y_{60}, z_{60}) \in \text{distance}(d_{59,60}) \\
\wedge & (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 > 0 \\
\wedge & (x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2 > 0 \\
\wedge & \quad \vdots \\
\wedge & (x_{59} - x_{60})^2 + (y_{59} - y_{60})^2 + (z_{59} - z_{60})^2 > 0 \\
\wedge & x_1 \leq x_5 \wedge y_1 \leq y_5 \wedge z_1 \leq z_5 \\
\wedge & x_{26} \leq x_{28} \wedge y_{26} \leq y_{28} \wedge z_{26} \leq z_{28} \\
\wedge & x_{31} \leq x_{36} \wedge y_{31} \leq y_{36} \wedge z_{31} \leq z_{36} \\
\wedge & x_{57} \leq x_{59} \wedge y_{57} \leq y_{59} \wedge z_{57} \leq z_{59} \\
\wedge & (y_{40} - y_{38}) \times (z_{36} - z_{38}) - (y_{36} - y_{38}) \times (z_{40} - z_{38}) \geq 0 \\
\wedge & (x_{36} - x_{38}) \times (z_{40} - z_{38}) - (x_{40} - x_{38}) \times (z_{36} - z_{38}) \geq 0 \\
\wedge & (x_{40} - x_{38}) \times (y_{36} - y_{38}) - (x_{36} - x_{38}) \times (y_{40} - y_{38}) \geq 0 \\
\wedge & (y_{45} - y_{43}) \times (z_{41} - z_{43}) - (y_{41} - y_{43}) \times (z_{45} - z_{43}) \geq 0 \\
\wedge & (x_{41} - x_{43}) \times (z_{45} - z_{43}) - (x_{45} - x_{43}) \times (z_{41} - z_{43}) \geq 0 \\
\wedge & (x_{45} - x_{43}) \times (y_{41} - y_{43}) - (x_{41} - x_{43}) \times (y_{45} - y_{43}) \geq 0 \\
\wedge & x_{46} \leq x_{50} \wedge y_{46} \leq y_{50} \wedge z_{46} \leq z_{50} \\
\wedge & x_{47} \leq x_{49} \wedge y_{47} \leq y_{49} \wedge z_{47} \leq z_{49} \\
\wedge & (y_{48} - y_{47}) \times (z_{46} - z_{47}) - (y_{46} - y_{47}) \times (z_{48} - z_{47}) \geq 0 \\
\wedge & (x_{46} - x_{47}) \times (z_{48} - z_{47}) - (x_{48} - x_{47}) \times (z_{46} - z_{47}) \geq 0 \\
\wedge & (x_{48} - x_{47}) \times (y_{46} - y_{47}) - (x_{46} - x_{47}) \times (y_{48} - y_{47}) \geq 0
\end{pmatrix}
$$

2.4 Résultats

La résolution du problème des pentaminos modélisé avec cette contrainte ne fonctionne pas. Lors de l'exécution de l'algorithme de résolution, nous n'obtenons dans les meilleurs cas qu'une solution par heure de calcul ; ce qui est bien trop lent.

Notre problème a été décomposé en un ensemble de contraintes trop locales, ne mettant en relation que des couples de points. Dans le chapitre suivant, nous étudions les contraintes de « placement » et de « points distincts » de façon globale, en posant la première sur l'ensemble de cinq points de chaque pentamino et la suivante sur l'ensemble des points du problème, en espérant bénéficier de réductions plus efficaces.

Chapitre 3

Expression du problème avec des contraintes plus élémentaires

Nous allons dans ce chapitre décomposer notre problème en contraintes plus élémentaires, et étudier la façon de réduire ces contraintes. Cette nouvelle contrainte décrivant le problème des pentaminos sera ensuite fournie en entrée à l'algorithme de résolution présenté dans le premier chapitre. Cet algorithme transformera notre contrainte en une nouvelle contrainte plus simple composée d'une disjonction de plusieurs contraintes simples représentant chacune une solution de notre problème.

3.1 Simplification de la contrainte d'être des vecteurs distincts

Ne disposant pas de moyen efficace de calcul de la contrainte « être n vecteurs distincts » et remarquant que les triplets $(x, y, z) \in [a-1]\times[b-1]\times[c-1]$, nous utilisons l'application $(x, y, z) \mapsto x + ay + abz$ pour ré-exprimer cette contrainte au moyen de la contrainte « être n entiers distincts ». La contrainte

$$(x_1, y_1, z_1, \ldots, x_{60}, y_{60}, z_{60}) \in \mathit{PointsDistincts}$$

deviens alors

$$\exists\, m_1 \ldots \exists\, m_{60} \begin{pmatrix} & m_1 \in [0, abc-1] \wedge \ldots \wedge m_{60} \in [0, abc-1] \\ \wedge & (m_1, x_1, y_1, z_1) \in \mathit{SommeP}(1, a, ab) \\ \wedge & \vdots \\ \wedge & (m_{60}, x_{60}, y_{60}, z_{60}) \in \mathit{SommeP}(1, a, ab) \\ \wedge & (m_1, \ldots, m_{60}) \in \mathit{EntiersDistincts} \end{pmatrix}$$

où *EntiersDistincts* est l'ensemble des n-uplets d'entiers distincts deux à deux et $\mathit{SommeP}(a_1, \ldots, a_n)$ l'ensemble des n-uplets d'entiers $(x, y_1, \ldots, y_n)$ tels que $x = a_1y_1 + \cdots + a_ny_n$.

Calculer efficacement une contrainte de « somme pondérée » étant quelque chose d'aussi difficile que de résoudre un problème de sac à dos (qui en est un sous-problème) ; nous décomposons cette contrainte en contraintes élémentaires de « somme » et de « produit par une constante » en introduisant à nouveau des variables quantifiées qui n'apparaissent pas

dans les solutions :

$$(m,x,y,z) \in \mathit{SommeP}(1,a,ab)$$
$$\Updownarrow$$
$$\exists\, s\, \exists\, t \left(\begin{array}{ll} & s \in [0, a(b-1)] \\ \wedge & t \in [0, ab(c-1)] \\ \wedge & (s,y) \in \mathit{Produit}(a) \\ \wedge & (t,z) \in \mathit{Produit}(ab) \\ \wedge & (m,x,s,t) \in \mathit{Somme} \end{array}\right)$$

où *Produit*(a) est l'ensemble des couples (x,y) tels que $x = ay$ et *Somme* est l'ensemble des n-uplets $(x, y_1, \ldots, y_n)$ tels que $x = y_1 + \cdots + y_n$.

Après application de ces transformations, notre ensemble de contraintes se trouve sous la forme

$$\left(\begin{array}{ll} & x_1 \in [0, a-1] \wedge \ldots \wedge x_{60} \in [0, a-1] \\ \wedge & y_1 \in [0, b-1] \wedge \ldots \wedge y_{60} \in [0, b-1] \\ \wedge & z_1 \in [0, c-1] \wedge \ldots \wedge z_{60} \in [0, c-1] \\ \wedge & (x_1, y_1, z_1, \ldots, x_5, y_5, z_5) \in \mathit{Classe}(\pi_1) \\ \wedge & \quad \vdots \\ \wedge & (x_{56}, y_{56}, z_{56}, \ldots, x_{60}, y_{60}, z_{60}) \in \mathit{Classe}(\pi_{12}) \\ \wedge & \exists\, m_1 \ldots \exists\, m_{60} \\ & \left(\begin{array}{ll} & m_1 \in [0, abc-1] \wedge \ldots \wedge m_{60} \in [0, abc-1] \\ \wedge & (m_1, \ldots, m_{60}) \in \mathit{EntiersDistincts} \\ & \exists\, s_1\, \exists\, t_1 \qquad\qquad\qquad\qquad \exists\, s_{60}\, \exists\, t_{60} \\ \wedge & \left(\begin{array}{ll} & s_1 \in [0, a(b-1)] \\ \wedge & t_1 \in [0, ab(c-1)] \\ \wedge & (s_1, y_1) \in \mathit{Produit}(a) \\ \wedge & (t_1, z_1) \in \mathit{Produit}(ab) \\ \wedge & (m_1, x_1, s_1, t_1) \in \mathit{Somme} \end{array}\right) \wedge \cdots \wedge \left(\begin{array}{ll} & s_{60} \in [0, a(b-1)] \\ \wedge & t_{60} \in [0, ab(c-1)] \\ \wedge & (s_{60}, y_{60}) \in \mathit{Produit}(a) \\ \wedge & (t_{60}, z_{60}) \in \mathit{Produit}(ab) \\ \wedge & (m_{60}, x_{60}, s_{60}, t_{60}) \in \mathit{Somme} \end{array}\right) \end{array}\right) \end{array}\right)$$

3.2 Réduction des contraintes élémentaires

3.2.1 Multiplication entière par une constante

Étant donné un entier a, on rappelle que *Produit*(a) est l'ensemble des couples d'entiers (x,y) qui sont tels que

$$(x,y) \in \mathit{Produit}(a) \iff x = ay$$

Proposition : Si X, Y, X', Y' sont des intervalles tels que $X' \times Y' = \text{apx}(X \times Y \cap \mathit{Produit}(a))$, avec X et Y non vides, alors

$$\begin{pmatrix} X' \\ Y' \end{pmatrix} = \begin{cases} \begin{pmatrix} \emptyset \\ \emptyset \end{pmatrix}, \text{ si } \overline{X} < \min\left(a\underline{Y}, a\overline{Y}\right) \text{ ou } \max\left(a\underline{Y}, a\overline{Y}\right) < \underline{X} \\ \begin{pmatrix} \{0\} \\ Y \end{pmatrix}, \text{ si } \overline{X} \geq \min\left(a\underline{Y}, a\overline{Y}\right) \text{ et } \max\left(a\underline{Y}, a\overline{Y}\right) \geq \underline{X} \text{ et } a = 0 \\ \begin{pmatrix} \left[\max\left(a\left\lceil \underline{X}/a \right\rceil, \min\left(a\underline{Y}, a\overline{Y}\right)\right), \min\left(a\left\lfloor \overline{X}/a \right\rfloor, \max\left(a\overline{Y}, a\underline{Y}\right)\right)\right] \\ \left[\max\left(\min\left(\left\lceil \underline{X}/a \right\rceil, \left\lceil \overline{X}/a \right\rceil\right), \underline{Y}\right), \min\left(\max\left(\left\lfloor \overline{X}/a \right\rfloor, \lfloor \underline{X}/a \rfloor\right), \overline{Y}\right)\right] \end{pmatrix}, \text{ sinon} \end{cases}$$

Preuve Soit x et y deux entiers. Remarquons tout d'abord que la propriété $(x, y) \in \mathit{Produit}(a) \cap X \times Y$ est équivalente à

$$x = ay \wedge \underline{X} \leq x \leq \overline{X} \wedge \underline{Y} \leq y \leq \overline{Y}$$

Trois cas se présentent alors

1. Si $\overline{X} < \min\left(a\underline{Y}, a\overline{Y}\right)$ ou $\max\left(a\underline{Y}, a\overline{Y}\right) < \underline{X}$ cette propriété est alors équivalente à

   $$\begin{array}{rl} & x = ay \wedge \underline{X} \leq x \leq \overline{X} \wedge \underline{Y} \leq y \leq \overline{Y} \\ \wedge & \left(\overline{X} < \min\left(a\underline{Y}, a\overline{Y}\right) \vee \max\left(a\underline{Y}, a\overline{Y}\right) < \underline{X}\right) \end{array}$$

   elle entraîne

   $$\begin{array}{rl} & x = ay \wedge \left(x \leq \overline{X} < \min\left(a\underline{Y}, a\overline{Y}\right) \leq ay \vee ay \leq \max\left(a\underline{Y}, a\overline{Y}\right) < \underline{X} \leq x\right) \\ \Rightarrow & x = ay \wedge (x < ay \vee ay < x) \\ \Rightarrow & x = ay \wedge x \neq ay \\ \Rightarrow & \textit{faux} \end{array}$$

   et donc dans ce cas

   $$\text{apx}\left(\mathit{Produit}(a) \cap X \times Y\right) = \emptyset$$

2. Si $\overline{X} \geq \min\left(a\underline{Y}, a\overline{Y}\right)$ et $\max\left(a\underline{Y}, a\overline{Y}\right) \geq \underline{X}$ et $a = 0$, la propriété est maintenant équivalente à

   $$x = 0 \wedge \underline{Y} \leq y \leq \overline{Y}$$

   d'où

   $$\text{apx}\left(\mathit{Produit}(a) \cap X \times Y\right) = \{O\} \times Y$$

3. Supposons que $\overline{X} \geq \min\left(a\underline{Y}, a\overline{Y}\right)$ et $\max\left(a\underline{Y}, a\overline{Y}\right) \geq \underline{X}$ et $a \neq 0$, la propriété est alors équivalente à

   $$\begin{array}{rl} & x = ay \wedge a \neq 0 \wedge \underline{X} \leq x \leq \overline{X} \wedge \underline{Y} \leq y \leq \overline{Y} \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \end{array}$$

Supposons cette propriété vérifiée, d'un côté la propriété $x \in \text{proj}_1\left(\textit{Produit}(a) \cap X \times Y\right)$ est équivalente à

$$\exists y \begin{pmatrix} & x = ay \wedge a \neq 0 \wedge \underline{X} \leq x \leq \overline{X} \wedge \underline{Y} \leq y \leq \overline{Y} \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} & a\underline{X}/a \leq x \leq a\overline{X}/a \wedge \left(a\underline{Y} \leq x \leq a\overline{Y} \vee a\overline{Y} \leq x \leq a\underline{Y}\right) \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \wedge \left(a|x\right) \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} & a\left\lceil \underline{X}/a \right\rceil \leq x \leq a\left\lfloor \overline{X}/a \right\rfloor \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq x \leq \max\left(a\underline{Y}, a\overline{Y}\right) \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \wedge \left(a|x\right) \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} & \max\left(a\left\lceil \underline{X}/a \right\rceil, \min\left(a\underline{Y}, a\overline{Y}\right)\right) \leq x \leq \min\left(a\left\lfloor \overline{X}/a \right\rfloor, \max\left(a\underline{Y}, a\overline{Y}\right)\right) \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \wedge \left(a|x\right) \end{pmatrix}$$

Les valeurs $a\left\lceil \underline{X}/a \right\rceil, a\underline{Y}, a\overline{Y} a\left\lfloor \overline{X}/a \right\rfloor, a\underline{Y}$ et $a\overline{Y}$ étant entières, on en déduit que

$$\begin{array}{rl} & \text{apx}\left(\text{proj}_1\left(\textit{Produit}(a) \cap X \times Y\right)\right) \\ = & \left[\max\left(a\left\lceil \underline{X}/a \right\rceil, \min\left(a\underline{Y}, a\overline{Y}\right)\right), \min\left(a\left\lfloor \overline{X}/a \right\rfloor, \max\left(a\underline{Y}, a\overline{Y}\right)\right)\right] \end{array}$$

D'un autre côté la propriété $y \in \text{proj}_2\left(\textit{Produit}(a) \cap X \times Y\right)$ est équivalente à

$$\exists x \begin{pmatrix} & x = ay \wedge a \neq 0 \wedge \underline{X} \leq x \leq \overline{X} \wedge \underline{Y} \leq y \leq \overline{Y} \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} & \left(\underline{X}/a \leq ay/a \leq \overline{X}/a \vee \overline{X}/a \leq ay/a \leq \underline{X}/a\right) \wedge \underline{Y} \leq y \leq \overline{Y} \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} & \left(\left\lceil \underline{X}/a \right\rceil \leq y \leq \left\lfloor \overline{X}/a \right\rfloor \vee \left\lceil \overline{X}/a \right\rceil \leq y \leq \left\lfloor \underline{X}/a \right\rfloor\right) \wedge \underline{Y} \leq y \leq \overline{Y} \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} & \min\left(\left\lceil \underline{X}/a \right\rceil, \left\lceil \overline{X}/a \right\rceil\right) \leq y \leq \max\left(\left\lfloor \underline{X}/a \right\rfloor, \left\lfloor \overline{X}/a \right\rfloor\right) \wedge \underline{Y} \leq y \leq \overline{Y} \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} & \max\left(\min\left(\left\lceil \underline{X}/a \right\rceil, \left\lceil \overline{X}/a \right\rceil\right), \underline{Y}\right) \leq y \leq \min\left(\max\left(\left\lfloor \underline{X}/a \right\rfloor, \left\lfloor \overline{X}/a \right\rfloor\right), \overline{Y}\right) \\ \wedge & \underline{X} \leq \max\left(a\underline{Y}, a\overline{Y}\right) \wedge \min\left(a\underline{Y}, a\overline{Y}\right) \leq \overline{X} \end{pmatrix}$$

Les valeurs $\left\lceil \underline{X}/a \right\rceil, \left\lceil \overline{X}/a \right\rceil, \underline{Y}, \left\lfloor \underline{X}/a \right\rfloor, \left\lfloor \overline{X}/a \right\rfloor$ et $\overline{Y}$ étant entières, on en déduit que

$$\begin{array}{rl} & \text{apx}\left(\text{proj}_2\left(\textit{Produit}(a) \cap X \times Y\right)\right) \\ = & \left[\max\left(\min\left(\left\lceil \underline{X}/a \right\rceil, \left\lceil \overline{X}/a \right\rceil\right), \underline{Y}\right), \min\left(\max\left(\left\lfloor \underline{X}/a \right\rfloor, \left\lfloor \overline{X}/a \right\rfloor\right), \overline{Y}\right)\right] \end{array}$$

L'utilisation de la propriété (1.2) termine la démonstration.

3.2.2 Somme multiple d'entiers

On rappelle que *Somme* est l'ensemble des $(n+1)$-uplets d'entiers $(x, y_1, \ldots, y_n)$ qui sont tels que

$$(x, y_1, \ldots, y_n) \in \mathit{Somme} \Leftrightarrow x = y_1 + \cdots + y_n$$

Proposition : Si $X, Y_1, \ldots, Y_n, X', Y'_1, \ldots, Y'_n$ sont des intervalles tels que $X' \times Y'_1 \times \cdots \times Y'_n = \text{apx}\,(X \times Y_1 \times \cdots \times Y_n \cap \mathit{Somme})$ avec $X, Y_1, \ldots, Y_n$ non vides alors

$$\begin{pmatrix} X' \\ Y'_i \end{pmatrix} = \begin{cases} \begin{pmatrix} \emptyset \\ \emptyset \end{pmatrix}, \text{ si } \overline{X} < \underline{Y_1} + \cdots + \underline{Y_n} \text{ ou } \overline{Y_1} + \cdots + \overline{Y_n} < \underline{X} \\ \begin{pmatrix} \left[\max(\underline{X}, \underline{Y_1} + \cdots + \underline{Y_n}), \min\left(\overline{X}, \overline{Y_1} + \cdots + \overline{Y_n}\right)\right] \\ \left[\max\left(\underline{Y_i}, \underline{X} - \sum_{j=1, j\neq i}^{n} \underline{X_j}\right), \min\left(\overline{Y_i}, \overline{X} - \sum_{j=1, j\neq i}^{n} \overline{Y_j}\right)\right] \end{pmatrix}, \text{ sinon.} \end{cases}$$

Preuve Soit $x, y_1, \ldots, y_n$ des entiers. Remarquons tout d'abord que la propriété $(x, y_1, \ldots, y_n) \in \mathit{Somme} \cap X \times Y_1 \times \cdots \times Y_n$ est équivalente à

$$x = y_1 + \cdots + y_n \wedge \underline{X} \leq x \leq \overline{X} \wedge \underline{Y_1} \leq y \leq \overline{Y_1} \wedge \ldots \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n}$$

Deux cas se présentent alors

1. Si $\overline{X} < \underline{Y_1} + \cdots + \underline{Y_n}$ ou $\overline{Y_1} + \cdots + \overline{Y_n} < \underline{X}$ alors cette propriété est équivalente à

$$\begin{array}{ll} & x = y_1 + \cdots + y_n \\ \wedge & \underline{X} \leq x \leq \overline{X} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \ldots \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \\ \wedge & (\overline{X} < \underline{Y_1} + \cdots + \underline{Y_n} \vee \overline{Y_1} + \cdots + \overline{Y_n} < \underline{X}) \end{array}$$

   elle entraîne dans ce cas

$$\begin{array}{ll} & x = y_1 + \cdots + y_n \\ \wedge & \begin{pmatrix} x \leq \overline{X} < \underline{Y_1} + \cdots + \underline{Y_n} \leq y_1 + \cdots + y_n \\ \vee \quad y_1 + \cdots + y_n \leq \overline{Y_1} + \cdots + \overline{Y_n} < \underline{X} \leq x \end{pmatrix} \\ \\ \Rightarrow & x = y_1 + \cdots + y_n \wedge (x < y_1 + \cdots + y_n \vee y_1 + \cdots + y_n < x) \\ \Rightarrow & x = y_1 + \cdots + y_n \wedge x \neq y_1 + \cdots + y_n \\ \Rightarrow & \mathit{faux} \end{array}$$

   nous avons donc dans ce cas

$$\text{apx}\,(\mathit{Somme} \cap X \times Y_1 \times \cdots \times Y_n) = \emptyset$$

2. Si $\overline{X} \geq \underline{Y_1} + \cdots + \underline{Y_n}$ et $\overline{Y_1} + \cdots + \overline{Y_n} < \underline{X}$ la propriété est maintenant équivalente à

$$\begin{array}{ll} & x = y_1 + \cdots + y_n \\ \wedge & \underline{X} \leq x \leq \overline{X} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \ldots \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array}$$

Supposons cette propriété vérifiée, d'un côté la propriété $x \in \text{proj}_1\,(\mathit{Somme} \cap X \times Y_1 \times \cdots \times Y_n)$ est équivalente à

$$\begin{array}{l} \exists y_1 \ldots \exists y_n \\ \left(\begin{array}{ll} & x = y_1 + \cdots + y_n \\ \wedge & \underline{X} \leq x \leq \overline{X} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \ldots \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \end{array}$$

$$\begin{array}{rl} & \exists y_1 \ldots \exists y_n \\ \Leftrightarrow & \left(\begin{array}{ll} & x = y_1 + \cdots + y_n \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq y_1 + \cdots + y_n \leq \overline{Y_1} + \cdots + \overline{Y_n} \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \\ \Leftrightarrow & \left(\begin{array}{ll} & \underline{Y_1} + \cdots + \underline{Y_n} \leq x \leq \overline{Y_1} + \cdots + \overline{Y_n} \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \\ \Leftrightarrow & \left(\begin{array}{ll} & \max(\underline{X}, \underline{Y_1} + \cdots + \underline{Y_n}) \leq x \leq \min(\overline{X}, \overline{Y_1} + \cdots + \overline{Y_n}) \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \end{array}$$

On en déduit que

$$\begin{array}{rl} & \text{apx}\,(\text{proj}_1\,(\mathit{Somme} \cap X \times Y_1 \times \cdots \times Y_n)) \\ = & \left[\max(\underline{X}, \underline{Y_1} + \cdots + \underline{Y_n})\,, \min(\overline{X}, \overline{Y_1} + \cdots + \overline{Y_n})\right] \end{array}$$

D'un autre côté, pour tout $i \in 1..n$, la propriété $y \in \text{proj}_{i+1}\,(\mathit{Somme} \cap X \times Y_1 \times \cdots \times Y_n)$ est équivalente à

$$\begin{array}{l} \exists x \exists y_1 \ldots \exists y_{i-1} \exists y_{i+1} \ldots \exists y_n \\ \left(\begin{array}{ll} & x = y_1 + \cdots + y_n \\ \wedge & \underline{X} \leq x \leq \overline{X} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \ldots \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \end{array}$$

$$\begin{array}{rl} & \exists x \exists y_1 \ldots \exists y_{i-1} \exists y_{i+1} \ldots \exists y_n \\ \Leftrightarrow & \left(\begin{array}{ll} & x = y_1 + \cdots + y_n \\ \wedge & \underline{Y_i} \leq y_i \leq \overline{Y_i} \wedge \underline{X} - \Sigma_{j=1..n}^{j \neq i} \underline{Y_j} \leq x - \Sigma_{j=1..n}^{j \neq i} y_j \leq \overline{X} - \Sigma_{j=1..n}^{j \neq i} \overline{Y_j} \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \\ \Leftrightarrow & \left(\begin{array}{ll} & \underline{Y_i} \leq y_i \leq \overline{Y_i} \wedge \underline{X} - \Sigma_{j=1..n}^{j \neq i} \underline{Y_j} \leq y_i \leq \overline{X} - \Sigma_{j=1..n}^{j \neq i} \overline{Y_j} \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \\ \Leftrightarrow & \left(\begin{array}{ll} & \max\Big(\underline{Y_i}, \underline{X} - \Sigma_{j=1..n}^{j \neq i} \underline{Y_j}\Big) \leq y_i \leq \min\Big(\overline{Y_i}, \overline{X} - \Sigma_{j=1..n}^{j \neq i} \overline{Y_j}\Big) \\ \wedge & \underline{Y_1} + \cdots + \underline{Y_n} \leq \overline{X} \wedge \underline{X} \leq \overline{Y_1} + \cdots + \overline{Y_n} \end{array} \right) \end{array}$$

On en déduit que

$$\begin{array}{rl} & \text{apx}\left(\text{proj}_{i+1}\,(\mathit{Somme} \cap X \times Y_1 \times \cdots \times Y_n)\right) \\ = & \left[\max\Big(\underline{Y_i}, \underline{X} - \Sigma_{j=1..n}^{j \neq i} \underline{Y_j}\Big)\,, \min\Big(\overline{Y_i}, \overline{X} - \Sigma_{j=1..n}^{j \neq i} \overline{Y_j}\Big)\right] \end{array}$$

L'utilisation de la propriété (1.2) termine la démonstration.

3.3 Réduction de la contrainte d'être des entiers distincts

On rappelle qu'*EntiersDistincts* est l'ensemble des n-uplets d'entiers distincts deux à deux :

$$(x_1, \ldots, x_n) \in \textit{EntiersDistincts}$$
$$\Updownarrow$$
$$x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge \ldots \wedge x_{n-2} \neq x_n \wedge x_{n-1} \neq x_n$$

Soit n intervalles d'entiers non vides $X_1, \ldots, X_n$, pour trouver les intervalles d'entiers $X'_1, \ldots, X'_n$ qui sont tels que $X'_1 \times \cdots \times X'_n$ soit le plus petit produit cartésien d'intervalles tel que :

$$X_1 \times \cdots \times X_n \cap \textit{EntiersDistincts} \;=\; X'_1 \times \cdots \times X'_n \cap \textit{EntiersDistincts}$$

nous avons utilisé tout d'abord l'algorithme en $\mathcal{O}(n^2)$ présenté dans [Lec96] dans le cas général.

Pour notre problème de placement des pentaminos où le nombre de variables contraintes à prendre des valeurs distinctes est égal au nombre de valeurs qu'elles peuvent prendre, nous avons finalement repris un algorithme réduisant en $\mathcal{O}(n \log(n))$ opérations la contrainte de « tri », décrit dans [BGC97]. Pour celà, nous imposons aux n premiers entiers d'être une permutation, et par conséquent un tri, des x_i :

$$\begin{pmatrix} & x_1 \in X_1 \cap [1,n] \wedge \cdots \wedge x_n \in X_n \cap [1,n] \\ \wedge & (x_1, \ldots, x_n) \in \textit{EntiersDistincts} \end{pmatrix}$$
$$\Updownarrow$$
$$\begin{pmatrix} & x_1 \in X_1 \wedge \cdots \wedge x_n \in X_n \\ \wedge & (x_1, \ldots, x_n) \in \textit{Perm}(1, \ldots, n) \end{pmatrix}$$
$$\Updownarrow$$
$$\begin{pmatrix} & x_1 \in X_1 \wedge \cdots \wedge x_n \in X_n \\ \wedge & (x_1, \ldots, x_n, 1, \ldots, n) \in \textit{Tri} \end{pmatrix}$$

Soit $x_1, \ldots, x_n$ des entiers, $(a_1, \ldots, a_n)$ une suite d'entiers strictement croissante, on définit la relation $\textit{Perm}(a_1, \ldots, a_n)$ de la façon suivante

$$(x_1, \ldots, x_n) \in \textit{Perm}(a_1, \ldots, a_n)$$

s'il existe une permutation $\sigma : 1..n \mapsto 1..n$ telle que

$$x_1 = a_{\sigma(1)}, \ldots, x_n = a_{\sigma(n)}$$

Si $X_1, \ldots, X_n, X'_1, \ldots, X'_n$ sont des intervalles d'entiers tels que $X'_1 \times \cdots \times X'_n = \text{apx}\,(X_1 \times \cdots \times X_n \cap \textit{Perm}(a_1, \ldots, a_n))$ avec $X_1, \ldots, X_n$ non vides, alors les X'_i se calculent à partir des X_i avec l'algorithme suivant, qui est une version simplifiée de l'algorithme de « tri » décrit dans [BGC97].

3.3.1 Algorithme

L'algorithme utilisé ici se compose de quatre étapes qui en premier lieu décomposent la contrainte de permutation en une conjonction équivalente de contraintes et réduit les intervalles en modifiant leurs bornes inférieures. Puis ces quatre étapes sont cette fois-ci appliquées indépendamment sur chacune de ces nouvelles contraintes de façon duale : ces contraintes sont à nouveau décomposées et les intervalles sont cette fois-ci réduits par modification de leurs bornes supérieures. L'algorithme étant essentiellement une simplification de celui donné en [BGC97], nous n'avons pas jugé nécessaire de démontrer sa correction.

Principe de dualité

Généralisons nos notations. Soit ρ une relation d'ordre total sur $\mathbf{Z}$, soit I un intervalle d'entiers, soit d un élément de $\mathbf{Z}^n$ et e, e' des entiers. Définissons la relation globale $\mathit{Perm}^{\rho}(d)$, les entiers $\underline{I}_{\rho}$ et $\overline{I}^{\rho}$ et l'intervalle d'entiers $[e,e']^{\rho}$ de la même façon que $\mathit{Perm}(d)$, $\underline{I}$, $\overline{I}$ et $[e,e']$, mais en remplaçant $\leq$ par ρ.

Désignons par $\rho^{\top}$ la relation définie par $e\rho^{\top}e'$ si et seulement si $e'\rho\, e$. L'ensemble ordonné $(\mathbf{Z},\rho^{\top})$ sera dit *dual* de $(\mathbf{Z},\rho)$. On constate que $(\rho^{\top})^{\top}=\rho$ et que :

Propriété : Soit I un intervalle d'entiers et $(d_1,\ldots,d_n)$ un élément de $\mathbf{Z}^n$. On a les propriétés suivantes :

1. I est un intervalle de $(\mathbf{Z},\rho^{\top})$,
2. soit les entiers $\underline{I}_{\rho}$ et $\overline{I}^{\rho^{\top}}$ existent et sont égaux, soit ils n'existent pas,
3. soit les entiers $\overline{I}^{\rho}$ et $\underline{I}_{\rho^{\top}}$ existent et sont égaux, soit ils n'existent pas,
4. $(d_1,\ldots,d_n)\in \mathit{Perm}^{\rho}(a_1,\ldots,a_n)$ si et seulement si $(d_1,\ldots,d_n)\in \mathit{Perm}^{\rho^{\top}}(a_n,\ldots,a_1)$

Description de l'algorithme

Étant donné n intervalles d'entiers $X_1,\ldots,X_n$, on s'intéresse à calculer apx $(X_1\times\cdots\times X_n \cap \mathit{Perm}(a_1,\ldots,a_n))$, cet algorithme consiste à calculer une suite $q_1,\ldots,q_8$ de huit contraintes composées. Chacune de ces contraintes fait intervenir le même ensemble $y_1,\ldots,y_n$ de n variables, uniquement la relation $\mathit{Perm}(a_1,\ldots,a_n)$ ou la relation $\mathit{Perm}^{\leq^{\top}}(a_1,\ldots,a_n)$, et est définie par :

$$q_1 \quad = \quad (y_1,\ldots,y_2)\in \mathit{Perm}^{\rho}(a_1,\ldots,a_n)\wedge y_1\in X_1\wedge\cdots\wedge y_n\in X_n$$

$$q_{i+1} \quad = \quad \begin{cases} q_i, \text{ s'il existe } j\in 1..n \text{ tel que } \mathit{dom}\,(y_i,q_i)=\emptyset \\ \mathcal{T}_i\,(p_{i1})\wedge\cdots\wedge\mathcal{T}_i\,(p_{ik_i}), \text{ avec } q_i=p_{i1}\wedge\cdots\wedge p_{ik_i}, \text{ sinon} \end{cases}$$

où i est pris dans $1..8$, où chaque p_{ij} est une contrainte de base et où $\mathcal{T}_i\,(p_{ij})$ est la contrainte composée définie ci-dessous. On a alors

$$\text{apx}\,(X_1\times\cdots\times X_n\cap \mathit{Perm}(a_1,\ldots,a_n)) = \mathit{dom}\,(y_1,q_8)\times\cdots\times\mathit{dom}\,(y_n,q_8)$$

Définition : Pour toute contrainte de base p de la forme

$$p = \left[\begin{array}{l} (x_1,\ldots,x_n) \in \mathit{Perm}^{\rho}(a_1,\ldots,a_n) \\ \wedge\, x_1 \in X_1 \\ \wedge\, \cdots \\ \wedge\, x_n \in X_n \end{array}\right]$$

où ρ est $\leq$ ou $\leq^{\top}$, la contrainte composée $\mathcal{T}_i(p)$, avec i pris dans 1..8, est définie comme suit au paragraphe (i).

(1) Tri des X_i par bornes supérieures croissantes

$$\mathcal{T}_1(p) = \left[\begin{array}{l} \left(x_{\nu(1)},\ldots,x_{\nu(n)}\right) \in \mathit{Perm}^{\rho}(a_1,\ldots,a_n) \\ \wedge\, x_1 \in X_1 \\ \wedge\, \cdots \\ \wedge\, x_n \in X_n \end{array}\right]$$

où ν est une bijection de 1..n dans 1..n telle que $(\overline{X_{\nu(1)}}^{\rho},\ldots,\overline{X_{\nu(n)}}^{\rho})$ est croissant pour ρ.

(2) Superposition des a_i aux X_i

$$\mathcal{T}_2(p) = \begin{cases} \left[\begin{array}{l} (x_1,\ldots,x_n) \in \mathit{Perm}^{\rho}(a_1,\ldots,a_n) \\ \wedge\, x_1 \in \emptyset \\ \wedge\, \cdots \\ \wedge\, x_n \in \emptyset \end{array}\right], \text{si } \varphi \text{ n'existe pas} \\ \left[\begin{array}{l} \left(x_{\varphi^{-1}(1)},\ldots,x_{\varphi^{-1}(n)}\right) \in \mathit{Perm}^{\rho}(a_1,\ldots,a_n) \\ \wedge\, x_1 \in X_1 \\ \wedge\, \cdots \\ \wedge\, x_n \in X_n \end{array}\right], \text{si } \varphi \text{ existe} \end{cases}$$

où φ est la bijection de 1..n dans 1..n définie par

$$\begin{array}{l} \varphi(1) = \min(\mathit{rel}(X,a,1)) \\ \varphi(i) = \min(\mathit{rel}(X,a,i) - \varphi(1..(i-1)))\,, \text{ si } i \in 1..n \end{array}$$

Avec

$$\mathit{rel}(a,X,i) = \min\{j \in 1..n \mid a_j \in X_i\}$$

(3) Décomposition de la contrainte

$$\mathcal{T}_3(p) = \begin{array}{ll} & \left[\begin{array}{l} \left(x_{h(1,1)},\ldots,x_{h(1,n_1)}\right) \in \mathit{Perm}^{\rho}\left(a_{h(1,1)},\ldots,a_{h(1,n_1)}\right) \\ \wedge\, x_{h(1,1)} \in X_{h(1,1)} \\ \wedge\, \cdots \\ \wedge\, x_{h(1,n_1)} \in X_{h(1,n_1)} \end{array}\right] \\ \wedge & \cdots \\ \wedge & \left[\begin{array}{l} \left(x_{h(k,1)},\ldots,x_{h(k,n_k)}\right) \in \mathit{Perm}^{\rho}\left(a_{h(k,1)},\ldots,a_{h(k,n_k)}\right) \\ \wedge\, x_{h(k,1)} \in X_{h(k,1)} \\ \wedge\, \cdots \\ \wedge\, x_{h(k,n_k)} \in X_{h(k,n_k)} \end{array}\right] \end{array}$$

où k, les n_i et les $h_{i,j}$, pour $i \in 1..k$ et $j \in 1..(2n_i)$ sont calculés comme suit. Nous trions le n-uplet $(1,\ldots,n)$ dans l'ordre croissant de $\leq_{X,a}$; la relation $\leq_{X,a}$ entre deux éléments de $1..n$ étant définie par

$$i \leq_{X,a} j \iff \begin{cases} \text{soit } i \leq j & \text{et il existe } k_1,\ldots,k_h \text{ avec } i = k_1, h \geq 1, k_h = j, \\ & k_i < k_{i+1} \text{ et } a_{k_{i+1}} \in X_{k_i}, \text{ pour tout } i \in 1..(h-1), \\ \text{soit } i > j & \text{et on n'a pas } j \leq_{X,a} i. \end{cases}$$

On met le n-uplet ainsi obtenu sous la forme $w_1 \cdot - - - - \cdot w_k$, où les n_i-uplets w_i sont croissants pour $<$ et de longueurs n_i maximales. Pour chaque $i \in 1..k$, le n_i-uplet $(h(i,1),\ldots,h(i,n_i))$ est défini par

$$(h(i,1),\ldots,(i,n_i)) = w_i.$$

Pour réaliser effectivement ce tri, nous avons utilisé un algorithme de complexité linéaire. Il consiste à appliquer tant que possible une des transformations décrites ci-dessous, sur le triplet $(\varepsilon,(1,\ldots,n),\varepsilon)$. Après $2n$ transformations, nous obtenons un triplet de la forme $(\varepsilon,\varepsilon,w)$, où w est le n-uplet $w_1 \cdot - - - - \cdot w_k$ attendu. Ces deux transformations sont :

1. $(u,\ (l)\cdot v,\ w) \longrightarrow (u\cdot(l),\ v,\ w)$ si $u = \varepsilon$ ou si $u \neq \varepsilon$ et $a_l \in X_{\grave{u}}$
2. $(u\cdot(l),\ v,\ w) \longrightarrow (u,\ v,\ (l)\cdot w)$ si $v = \varepsilon$ ou si $v \neq \varepsilon$ et $a_{\acute{v}} \notin X_l$

où l est un entier, u et v des suites d'entiers, $\grave{u}$ désigne le dernier élément de u et $\acute{v}$ le premier élément de v, quand u et v sont non vides.

(4) Écrêtement inférieur des X_i

$$\mathcal{T}_4(p) = \left[\begin{array}{l} (x_1,\ldots,x_n) \in \mathit{Perm}^{\rho^\top}(a_n,\ldots,a_1) \\ \wedge\, x_1 \in \left[a_{f(1)}, \overline{X_1}^{\rho}\right]^{\rho} \\ \wedge\, \cdots \\ \wedge\, x_n \in \left[a_{f(n)}, \overline{X_n}^{\rho}\right]^{\rho} \end{array}\right]$$

où $f(i) = \min\left\{j \in 1..n \mid \underline{X_i}_{\rho}\rho\, a_j\right\}$, pour $i \in 1..n$.

(5) à (8) La suite $\mathcal{T}_5(p)$, $\mathcal{T}_6(p)$, $\mathcal{T}_7(p)$, $\mathcal{T}_8(p)$ est égale à la suite $\mathcal{T}_1(p)$, $\mathcal{T}_2(p)$, $\mathcal{T}_3(p)$, $\mathcal{T}_4(p)$.

Complexité

Cet algorithme se résume à la suite des quatre opérations suivantes :
- un tri,
- une superposition,
- une décomposition
- et un écrêtement.

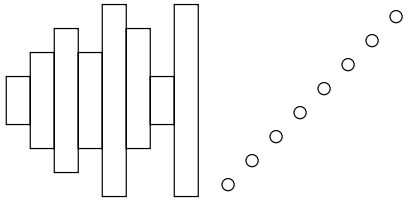
Les algorithmes que nous utilisons pour chacune de ces opérations sont de complexité respectives $\mathcal{O}(n\log(n))$, $\mathcal{O}(n\log(n))$, $\mathcal{O}(n)$ et $\mathcal{O}(n)$. La complexité de notre algorithme est donc en $\mathcal{O}(n\log(n))$.

3.3.2 Exemple

Nous illustrons cet algorithme par l'exemple suivant : on se donne les intervalles $X_1 = [3,4]$, $X_2 = [2,5]$, $X_3 = [1,6]$, $X_4 = [2,5]$, $X_5 \in [0,7]$, $X_6 = [2,6]$, $X_7 = [3,4]$ $X_8 = [0,7]$ et les entiers $a_1 = 0$, $a_2 = 1$, $a_3 = 2$, $a_4 = 3$, $a_5 = 4$, $a_6 = 5$, $a_7 = 6$, $a_8 = 7$; on souhaite donc réduire la contrainte

$$p = \left[\begin{array}{l} (x_1, \ldots, x_8) \in \mathit{Perm}^{\leq}(0, \ldots, 7) \\ \wedge\, x_1 \in [3,4] \ \wedge\ x_2 \in [2,5] \\ \wedge\, x_3 \in [1,6] \ \wedge\ x_4 \in [2,5] \\ \wedge\, x_5 \in [0,7] \ \wedge\ x_6 \in [2,6] \\ \wedge\, x_7 \in [3,4] \ \wedge\ x_8 \in [0,7] \end{array} \right]$$

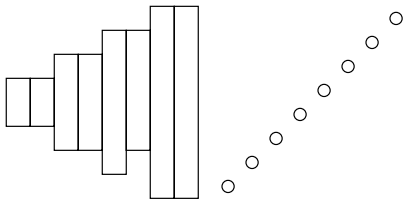
dans cet exemple, les modifications intervenues sur la contrainte seront mises en évidence par un fond grisé et chaque étape de l'algorithme sera illustrée par un dessin figurant les intervalles de valeurs des x_i et un point pour les valeurs de chaque a_i, comme ci-dessous pour nos valeurs initiales :



Étapes (1) à (4)

Le tri des X_i par bornes supérieures croissantes nous donne une première permutation $\nu = \{1 \mapsto 1, 2 \mapsto 7, 3 \mapsto 2, 4 \mapsto 4, 5 \mapsto 3, 6 \mapsto 6, 7 \mapsto 5, 8 \mapsto 8\}$ et la nouvelle contrainte $\mathcal{T}_1(p)$:

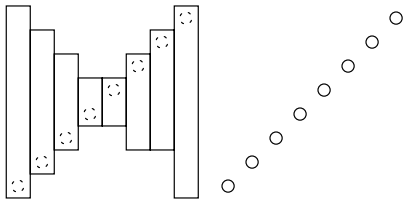
$$\mathcal{T}_1(p) = \left[\begin{array}{l} (x_1, x_7, x_2, x_4, x_3, x_6, x_5, x_8) \in \mathit{Perm}^{\leq}(0, \ldots, 7) \\ \wedge\, x_1 \in [3,4] \ \wedge\ x_2 \in [2,5] \\ \wedge\, x_3 \in [1,6] \ \wedge\ x_4 \in [2,5] \\ \wedge\, x_5 \in [0,7] \ \wedge\ x_6 \in [2,6] \\ \wedge\, x_7 \in [3,4] \ \wedge\ x_8 \in [0,7] \end{array} \right]$$



La bijection ν nous permet de calculer la superposition des a_i aux X_i qui est la permutation $\varphi = \{1 \mapsto 4, 2 \mapsto 3, 3 \mapsto 2, 4 \mapsto 6, 5 \mapsto 1, 6 \mapsto 7, 7 \mapsto 5, 8 \mapsto 8\}$. φ existe, l'ensemble

$X_1 \times \cdots \times X_n \cap \mathit{Perm}^{\leq}(a_1, \ldots, a_n)$ est donc non vide et

$$\mathcal{T}_2\left(\mathcal{T}_1\left(p\right)\right) = \left[\begin{array}{l} (x_5, x_3, x_2, x_1, x_7, x_4, x_6, x_8) \in \mathit{Perm}^{\leq}(0, \ldots, 7) \\ \wedge\ x_1 \in [3,4] \ \wedge\ x_2 \in [2,5] \\ \wedge\ x_3 \in [1,6] \ \wedge\ x_4 \in [2,5] \\ \wedge\ x_5 \in [0,7] \ \wedge\ x_6 \in [2,6] \\ \wedge\ x_7 \in [3,4] \ \wedge\ x_8 \in [0,7] \end{array}\right]$$



Trions le 8-uplet $(1, \ldots, 8)$ dans l'ordre croissant de $\leq_{X,a}$, en utilisant l'algorithme décrit précédement. Nous avons :

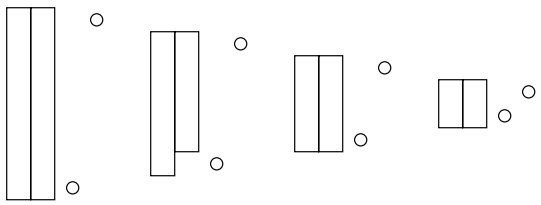
$$\begin{array}{rcl} (X_1, \ldots, X_8) & = & ([0,7],[1,6],[2,5],[3,4],[3,4],[2,5],[2,6],[0,7]) \\ (a_1, \ldots, a_8) & = & (0,1,2,3,4,5,6,7) \end{array}$$

ce tri s'effectue par les 16 tranformations du triplet $(\varepsilon,\ (1,2,3,4,5,6,7,8),\ \varepsilon)$:

$$\begin{array}{lcll} (\varepsilon,\ (1)\cdot(2,3,4,5,6,7,8),\ \varepsilon) & \longrightarrow & (\varepsilon\cdot(1),\ (2,3,4,5,6,7,8),\ \varepsilon) & \\ ((1),\ (2)\cdot(3,4,5,6,7,8),\ \varepsilon) & \longrightarrow & ((1)\cdot(2),\ (3,4,5,6,7,8),\ \varepsilon) & \text{car } a_2 \in X_1 \\ ((1,2),\ (3)\cdot(4,5,6,7,8),\ \varepsilon) & \longrightarrow & ((1,2)\cdot(3),\ (4,5,6,7,8),\ \varepsilon) & \text{car } a_3 \in X_2 \\ ((1,2,3),\ (4)\cdot(5,6,7,8),\ \varepsilon) & \longrightarrow & ((1,2,3)\cdot(4),\ (5,6,7,8),\ \varepsilon) & \text{car } a_4 \in X_3 \\ ((1,2,3,4),\ (5)\cdot(6,7,8),\ \varepsilon) & \longrightarrow & ((1,2,3,4)\cdot(5),\ (6,7,8),\ \varepsilon) & \text{car } a_5 \in X_4 \\ ((1,2,3,4)\cdot(5),\ (6,7,8),\ \varepsilon) & \longrightarrow & ((1,2,3,4),\ (6,7,8),\ (5)\cdot\varepsilon) & \text{car } a_6 \notin X_5 \\ ((1,2,3)\cdot(4),\ (6,7,8),\ (5)) & \longrightarrow & ((1,2,3),\ (6,7,8),\ (4)\cdot(5)) & \text{car } a_6 \notin X_4 \\ ((1,2,3),\ (6)\cdot(7,8),\ (4,5)) & \longrightarrow & ((1,2,3)\cdot(6),\ (7,8),\ (4,5)) & \text{car } a_6 \in X_3 \\ ((1,2,3)\cdot(6),\ (7,8),\ (4,5)) & \longrightarrow & ((1,2,3),\ (7,8),\ (6)\cdot(4,5)) & \text{car } a_7 \notin X_6 \\ ((1,2)\cdot(3),\ (7,8),\ (6,4,5)) & \longrightarrow & ((1,2),\ (7,8),\ (3)\cdot(6,4,5)) & \text{car } a_7 \notin X_3 \\ ((1,2),\ (7)\cdot(8),\ (3,6,4,5)) & \longrightarrow & ((1,2)\cdot(7),\ (8),\ (3,6,4,5)) & \text{car } a_7 \in X_2 \\ ((1,2)\cdot(7),\ (8),\ (3,6,4,5)) & \longrightarrow & ((1,2),\ (8),\ (7)\cdot(3,6,4,5)) & \text{car } a_8 \notin X_7 \\ ((1)\cdot(2),\ (8),\ (7,3,6,4,5)) & \longrightarrow & ((1),\ (8),\ (2)\cdot(7,3,6,4,5)) & \text{car } a_8 \notin X_2 \\ ((1),\ (8)\cdot\varepsilon,\ (2,7,3,6,4,5)) & \longrightarrow & ((1)\cdot(8),\ \varepsilon,\ (2,7,3,6,4,5)) & \text{car } a_8 \in X_1 \\ ((1)\cdot(8),\varepsilon,\ (2,7,3,6,4,5)) & \longrightarrow & ((1),\ \varepsilon,\ (8)\cdot(2,7,3,6,4,5)) & \\ (\varepsilon\cdot(1),\varepsilon,\ (8,2,7,3,6,4,5)) & \longrightarrow & (\varepsilon,\ \varepsilon,\ (1)\cdot(8,2,7,3,6,4,5)) & \end{array}$$

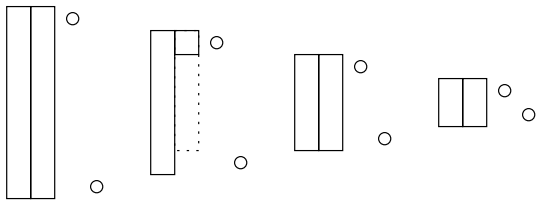
et nous donne $(1,8)\cdot(2,7)\cdot(3,6)\cdot(4,5)$, la contrainte précédente se décompose alors en la conjonction des quatre contraintes suivantes :

$$\mathcal{T}_3\left(\mathcal{T}_2\left(\mathcal{T}_1\left(p\right)\right)\right) = q_1 \wedge q_2 \wedge q_3 \wedge q_4 = \left[\begin{array}{l} (x_5, x_8) \in \mathit{Perm}^{\leq}(0,7) \\ \wedge\ x_5 \in [0,7] \ \wedge\ x_8 \in [0,7] \end{array}\right] \wedge \left[\begin{array}{l} (x_3, x_6) \in \mathit{Perm}^{\leq}(1,6) \\ \wedge\ x_3 \in [1,6] \ \wedge\ x_6 \in [2,6] \end{array}\right]$$
$$\wedge \left[\begin{array}{l} (x_2, x_4) \in \mathit{Perm}^{\leq}(2,5) \\ \wedge\ x_2 \in [2,5] \ \wedge\ x_4 \in [2,5] \end{array}\right] \wedge \left[\begin{array}{l} (x_1, x_7) \in \mathit{Perm}^{\leq}(3,4) \\ \wedge\ x_1 \in [3,4] \ \wedge\ x_7 \in [3,4] \end{array}\right]$$



Nous pouvons maintenant pour chaque q_i appliquer l'écrêtement inférieur des X_i :

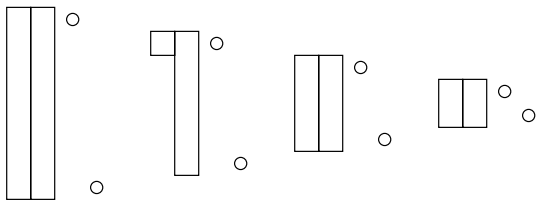
$$\mathcal{T}_4(q_1)\wedge\mathcal{T}_4(q_2)\wedge\mathcal{T}_4(q_3)\wedge\mathcal{T}_4(q_4) = \wedge \begin{bmatrix} (x_5,x_8)\in \mathit{Perm}^{\leq^\top}(7,0) \\ \wedge\, x_5\in[0,7]\ \wedge\ x_8\in[0,7] \end{bmatrix} \wedge \begin{bmatrix} (x_3,x_6)\in \mathit{Perm}^{\leq^\top}(6,1) \\ \wedge\, x_3\in[1,6]\ \wedge\ x_6=6 \end{bmatrix} \wedge \begin{bmatrix} (x_2,x_4)\in \mathit{Perm}^{\leq^\top}(5,2) \\ \wedge\, x_2\in[2,5]\ \wedge\ x_4\in[2,5] \end{bmatrix} \wedge \begin{bmatrix} (x_1,x_7)\in \mathit{Perm}^{\leq^\top}(4,3) \\ \wedge\, x_1\in[3,4]\ \wedge\ x_7\in[3,4] \end{bmatrix}$$



Étapes (5) à (8)

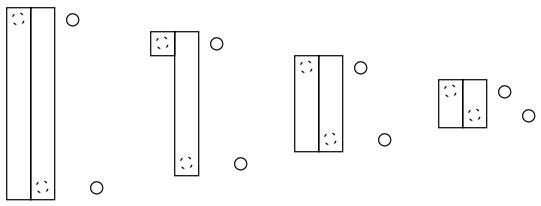
Nous appliquons maintenant à chaque contrainte simple la version duale de l'algorithme précédent. Tout d'abord le tri des X_i par bornes inférieures décroissantes :

$$\begin{array}{ll} & \mathcal{T}_5(\mathcal{T}_4(q_1)) \\ \wedge & \mathcal{T}_5(\mathcal{T}_4(q_2)) \\ \wedge & \mathcal{T}_5(\mathcal{T}_4(q_3)) \\ \wedge & \mathcal{T}_5(\mathcal{T}_4(q_4)) \end{array} = \wedge \begin{bmatrix} (x_5,x_8)\in \mathit{Perm}^{\leq^\top}(7,0) \\ \wedge\, x_5\in[0,7]\ \wedge\ x_8\in[0,7] \end{bmatrix} \wedge \begin{bmatrix} (x_6,x_3)\in \mathit{Perm}^{\leq^\top}(6,1) \\ \wedge\, x_3\in[1,6]\ \wedge\ x_6=6 \end{bmatrix} \wedge \begin{bmatrix} (x_2,x_4)\in \mathit{Perm}^{\leq^\top}(5,2) \\ \wedge\, x_2\in[2,5]\ \wedge\ x_4\in[2,5] \end{bmatrix} \wedge \begin{bmatrix} (x_1,x_7)\in \mathit{Perm}^{\leq^\top}(4,3) \\ \wedge\, x_1\in[3,4]\ \wedge\ x_7\in[3,4] \end{bmatrix}$$



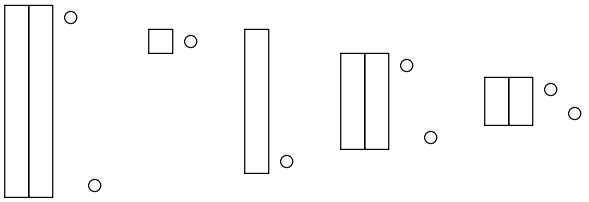
La superposition des a_i aux X_i :

$$\begin{array}{ll} & \mathcal{T}_6(\mathcal{T}_5(\mathcal{T}_4(q_1))) \\ \wedge & \mathcal{T}_6(\mathcal{T}_5(\mathcal{T}_4(q_2))) \\ \wedge & \mathcal{T}_6(\mathcal{T}_5(\mathcal{T}_4(q_3))) \\ \wedge & \mathcal{T}_6(\mathcal{T}_5(\mathcal{T}_4(q_4))) \end{array} = \wedge \begin{bmatrix} (x_5,x_8)\in \mathit{Perm}^{\leq^\top}(7,0) \\ \wedge\, x_5\in[0,7]\ \wedge\ x_8\in[0,7] \end{bmatrix} \wedge \begin{bmatrix} (x_6,x_3)\in \mathit{Perm}^{\leq^\top}(6,1) \\ \wedge\, x_3\in[1,6]\ \wedge\ x_6=6 \end{bmatrix} \wedge \begin{bmatrix} (x_2,x_4)\in \mathit{Perm}^{\leq^\top}(5,2) \\ \wedge\, x_2\in[2,5]\ \wedge\ x_4\in[2,5] \end{bmatrix} \wedge \begin{bmatrix} (x_1,x_7)\in \mathit{Perm}^{\leq^\top}(4,3) \\ \wedge\, x_1\in[3,4]\ \wedge\ x_7\in[3,4] \end{bmatrix}$$



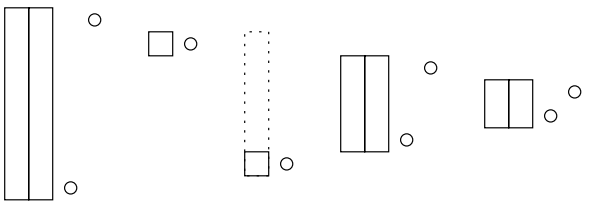
La décomposition de chaque contrainte :

$$\begin{array}{l} \mathcal{T}_7\left(\mathcal{T}_6\left(\mathcal{T}_5\left(\mathcal{T}_4\left(q_1\right)\right)\right)\right) \\ \wedge\ \mathcal{T}_7\left(\mathcal{T}_6\left(\mathcal{T}_5\left(\mathcal{T}_4\left(q_2\right)\right)\right)\right) \\ \wedge\ \mathcal{T}_7\left(\mathcal{T}_6\left(\mathcal{T}_5\left(\mathcal{T}_4\left(q_3\right)\right)\right)\right) \\ \wedge\ \mathcal{T}_7\left(\mathcal{T}_6\left(\mathcal{T}_5\left(\mathcal{T}_4\left(q_4\right)\right)\right)\right) \end{array} = r_1 \wedge r_2 \wedge r_3 \wedge r_4 \wedge r_5 = \begin{array}{ll} & \left[\begin{array}{l}(x_5, x_8) \in \mathit{Perm}^{\leq^\top}(7,0) \\ \wedge\, x_5 \in [0,7]\ \wedge\ x_8 \in [0,7]\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_3) \in \mathit{Perm}^{\leq^\top}(1) \\ \wedge\, x_3 \in [1,6]\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_6) \in \mathit{Perm}^{\leq^\top}(6) \\ \wedge\, x_6 = 6\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_2, x_4) \in \mathit{Perm}^{\leq^\top}(5,2) \\ \wedge\, x_2 \in [2,5]\ \wedge\ x_4 \in [2,5]\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_1, x_7) \in \mathit{Perm}^{\leq^\top}(4,3) \\ \wedge\, x_1 \in [3,4]\ \wedge\ x_7 \in [3,4]\end{array}\right] \end{array}$$



Et enfin l'écrêtement supérieur des X_i :

$$\mathcal{T}_8\left(r_1\right) \wedge \mathcal{T}_8\left(r_2\right) \wedge \mathcal{T}_8\left(r_3\right) \wedge \mathcal{T}_8\left(r_4\right) \wedge \mathcal{T}_8\left(r_5\right) = \begin{array}{ll} & \left[\begin{array}{l}(x_5, x_8) \in \mathit{Perm}^{\leq}(0,7) \\ \wedge\, x_5 \in [0,7]\ \wedge\ x_8 \in [0,7]\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_3) \in \mathit{Perm}^{\leq}(1) \\ \wedge\ x_3 = 1\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_6) \in \mathit{Perm}^{\leq}(6) \\ \wedge\, x_6 = 6\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_2, x_4) \in \mathit{Perm}^{\leq}(2,5) \\ \wedge\, x_2 \in [2,5]\ \wedge\ x_4 \in [2,5]\end{array}\right] \\ \wedge & \left[\begin{array}{l}(x_1, x_7) \in \mathit{Perm}^{\leq}(3,4) \\ \wedge\, x_1 \in [3,4]\ \wedge\ x_7 \in [3,4]\end{array}\right] \end{array}$$



Nous obtenons finalement :

$$\text{apx}\,(X_1\times\cdots\times X_n\cap \textit{Perm}(a_1,\ldots,a_n)) = [3,4]\times[2,5]\times\{1\}\times[2,5]\times[0,7]\times\{6\}\times[3,4]\times[0,7]$$

Nous aurions obtenu exactement le même résultat avec une contrainte d'« être des entiers distincts ». L'intérêt d'utiliser une contrainte de permutation est ici d'avoir calculé la décomposition de notre contrainte en une conjonction de cinq petites contraintes ; une contrainte d'entiers distincts ne ce serait pas décomposée dans cet exemple, et se décomposerait beaucoup moins dans tous les autres cas.

3.4 Réduction de la contrainte de placement

Pour réduire la contrainte *Classe*(f), nous introduisons tout d'abord la contrainte *Translaté*(f) et donnons la façon de la réduire.

3.4.1 Contrainte de translation

Si f est une configuration de n points, alors *Translaté*(f) est l'ensemble des $3n$-uplets d'entiers tels que

$$\begin{array}{c}(x_1,y_1,z_1,\ldots,x_n,y_n,z_n)\in \textit{Translaté}(f)\\ \Updownarrow\\ \{(x_1,y_1,z_1),\ldots,(x_n,y_n,z_n)\}\in \textit{translatés}\,(f) \text{ et}\\ ((x_1,y_1,z_1),\ldots,(x_n,y_n,z_n)) \text{ est lexicographiquement croissant}\end{array}$$

Proposition : Soit $X_1,Y_1,Z_1,\ldots,X_n,Y_n,Z_n,X'_1,Y'_1,Z'_1,\ldots,X'_n,Y'_n,Z'_n$ des intervalles tels que $X'_1\times Y'_1\times Z'_1\times\cdots\times X'_n\times Y'_n\times Z'_n = \text{apx}(\textit{Translaté}\,(a_1,b_1,c_1,\ldots,a_n,b_n,c_n)\cap X_1\times Y_1\times Z_1\times\cdots\times X_n\times Y_n\times Z_n)$. Si $((a_1,b_1,c_1),\ldots,(a_n,b_n,c_n))$ est un n-uplet de points de f croissants dans l'ordre lexicographique alors, pour tout $i\in\{1,\ldots,n\}$

$$\begin{pmatrix}X'_i\\Y'_i\\Z'_i\end{pmatrix} = \begin{cases}\begin{pmatrix}\emptyset\\\emptyset\\\emptyset\end{pmatrix}, \text{ si } & \begin{array}{r}\max_{j=1..n}\left(\underline{X_j}-a_j\right) > \min_{j=1..n}\left(\overline{X_j}-a_j\right)\\ \text{ou } \max_{j=1..n}\left(\underline{Y_j}-b_j\right) > \min_{j=1..n}\left(\overline{Y_j}-b_j\right)\\ \text{ou } \max_{j=1..n}\left(\underline{Z_j}-c_j\right) > \min_{j=1..n}\left(\overline{Z_j}-c_j\right)\end{array}\\ \begin{pmatrix}\left[a_i+\max_{j=1..n}\left(\underline{X_j}-a_j\right), a_i+\min_{j=1..n}\left(\overline{X_j}-a_j\right)\right]\\ \left[b_i+\max_{j=1..n}\left(\underline{Y_j}-b_j\right), b_i+\min_{j=1..n}\left(\overline{Y_j}-b_j\right)\right]\\ \left[c_i+\max_{j=1..n}\left(\underline{Z_j}-c_j\right), c_i+\min_{j=1..n}\left(\overline{Z_j}-c_j\right)\right]\end{pmatrix}, \text{ autrement}\end{cases}$$

Preuve Soit $x_1,y_1,z_1,\ldots,x_n,y_n,z_n$ des entiers. Remarquons tout d'abord que la propriété $(x_1,y_1,z_1,\ldots,x_n,y_n,z_n)\in \textit{Translaté}\,(a_1,b_1,c_1,\ldots,a_n,b_n,c_n)\cap X_1\times Y_1\times Z_1\times\cdots\times X_n\times Y_n\times Z_n$

est équivalente à

$$\exists u \exists v \exists w \begin{pmatrix} & x_1 = a_1 + u \wedge y_1 = b_1 + v \wedge z_1 = c_1 + w \\ \wedge & \cdots \\ \wedge & x_n = a_n + u \wedge y_n = b_n + v \wedge z_n = c_1 + w \\ \wedge & \underline{X_1} \leq x_1 \leq \overline{X_1} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \underline{Z_1} \leq z_1 \leq \overline{Z_1} \\ \wedge & \cdots \\ \wedge & \underline{X_n} \leq x_n \leq \overline{X_n} \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \wedge \underline{Z_n} \leq z_n \leq \overline{Z_n} \end{pmatrix}$$

Deux cas se présentent alors :

1. Si $\max_{i=1..n}(\underline{X_i} - a_1) > \min_{i=1..n}(\overline{X_i} - a_1)$ ou $\max_{i=1..n}(\underline{Y_i} - b_1) > \min_{i=1..n}(\overline{Y_i} - b_1)$ ou $\max_{i=1..n}(\underline{Z_i} - c_1) > \min_{i=1..n}(\overline{Z_i} - c_1)$ alors cette propriété est équivalente à

$$\exists u \exists v \exists w \begin{pmatrix} & x_1 = a_1 + u \wedge y_1 = b_1 + v \wedge z_1 = c_1 + w \\ \wedge & \cdots \\ \wedge & x_n = a_n + u \wedge y_n = b_n + v \wedge z_n = c_1 + w \\ \wedge & \underline{X_1} \leq x_1 \leq \overline{X_1} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \underline{Z_1} \leq z_1 \leq \overline{Z_1} \\ \wedge & \cdots \\ \wedge & \underline{X_n} \leq x_n \leq \overline{X_n} \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \wedge \underline{Z_n} \leq z_n \leq \overline{Z_n} \\ & \begin{pmatrix} & \max_{i=1..n}(\underline{X_i} - a_i) > \min_{i=1..n}(\overline{X_i} - a_i) \\ \wedge & \vee \quad \max_{i=1..n}(\underline{Y_i} - b_i) > \min_{i=1..n}(\overline{Y_i} - b_i) \\ & \vee \quad \max_{i=1..n}(\underline{Z_i} - c_i) > \min_{i=1..n}(\overline{Z_i} - c_i) \end{pmatrix} \end{pmatrix}$$

   elle entraîne dans ce cas

$$\exists u \exists v \exists w \begin{pmatrix} & \underline{X_1} \leq x_1 \leq \overline{X_1} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \underline{Z_1} \leq z_1 \leq \overline{Z_1} \\ \wedge & \cdots \\ \wedge & \underline{X_n} \leq x_n \leq \overline{X_n} \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \wedge \underline{Z_n} \leq z_n \leq \overline{Z_n} \\ & \begin{pmatrix} & \max_{i=1..n}(\underline{X_i} - x_i + u) > \min_{i=1..n}(\overline{X_i} - x_i + u) \\ \wedge & \vee \quad \max_{i=1..n}(\underline{Y_i} - y_i + v) > \min_{i=1..n}(\overline{Y_i} - y_i + v) \\ & \vee \quad \max_{i=1..n}(\underline{Z_i} - z_i + w) > \min_{i=1..n}(\overline{Z_i} - z_i + w) \end{pmatrix} \end{pmatrix}$$

$$
\begin{array}{ll}
\Rightarrow & \exists u \exists v \exists w \\
& \left(\begin{array}{ll}
 & \underline{X_1} \leq x_1 \leq \overline{X_1} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \underline{Z_1} \leq z_1 \leq \overline{Z_1} \\
\wedge & \cdots \\
\wedge & \underline{X_n} \leq x_n \leq \overline{X_n} \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \wedge \underline{Z_n} \leq z_n \leq \overline{Z_n} \\
 & \left(\begin{array}{ll}
 & \max_{i=1..n} (\underline{X_i} - x_i + u) > \min_{i=1..n} (\overline{X_i} - x_i + u) \\
\vee & \max_{i=1..n} (\underline{Y_i} - y_i + v) > \min_{i=1..n} (\overline{Y_i} - y_i + v) \\
\vee & \max_{i=1..n} (\underline{Z_i} - z_i + w) > \min_{i=1..n} (\overline{Z_i} - z_i + w)
\end{array} \right)
\end{array} \right) \\
\Rightarrow & \exists u \exists v \exists w \\
& \left(\begin{array}{ll}
 & \underline{X_1} \leq x_1 \leq \overline{X_1} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \underline{Z_1} \leq z_1 \leq \overline{Z_1} \\
\wedge & \cdots \\
\wedge & \underline{X_n} \leq x_n \leq \overline{X_n} \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \wedge \underline{Z_n} \leq z_n \leq \overline{Z_n} \\
 & \left(\begin{array}{ll}
 & \max_{i=1..n} (\underline{X_i} - x_i) > \min_{i=1..n} (\overline{X_i} - x_i) \\
\vee & \max_{i=1..n} (\underline{Y_i} - y_i) > \min_{i=1..n} (\overline{Y_i} - y_i) \\
\vee & \max_{i=1..n} (\underline{Z_i} - z_i) > \min_{i=1..n} (\overline{Z_i} - z_i)
\end{array} \right)
\end{array} \right) \\
\Rightarrow & \exists u \exists v \exists w \\
& \left(\begin{array}{ll}
 & 0 \geq \max_{i=1..n} (\underline{X_i} - x_i) > \min_{i=1..n} (\overline{X_i} - x_i) \geq 0 \\
\vee & 0 \geq \max_{i=1..n} (\underline{Y_i} - y_i) > \min_{i=1..n} (\overline{Y_i} - y_i) \geq 0 \\
\vee & 0 \geq \max_{i=1..n} (\underline{Z_i} - z_i) > \min_{i=1..n} (\overline{Z_i} - z_i) \geq 0
\end{array} \right) \\
\Rightarrow & \exists u \exists v \exists w \quad 0 > 0 \\
\Rightarrow & faux
\end{array}
$$

d'où dans ce cas

$$\text{apx}\,(\textit{Translaté}\,(a_1, b_1, c_1, \ldots, a_n, b_n, c_n) \cap X_1 \times Y_1 \times Z_1 \times \cdots \times X_n \times Y_n \times Z_n) = \emptyset$$

2. Si maintenant $\max_{i=1..n}(\underline{X_i} - a_1) \leq \min_{i=1..n}(\overline{X_i} - a_1)$ ou $\max_{i=1..n}(\underline{Y_i} - b_1) \leq \min_{i=1..n}(\overline{Y_i} -$

b_1) ou $\max_{i=1..n}(\underline{Z_i} - c_1) \leq \min_{i=1..n}(\overline{Z_i} - c_1)$, la propriété est alors équivalente à

$$
\begin{array}{l}
\exists u \exists v \exists w \\
\left(\begin{array}{ll}
 & x_1 = a_1 + u \wedge y_1 = b_1 + v \wedge z_1 = c_1 + w \\
\wedge & \cdots \\
\wedge & x_n = a_n + u \wedge y_n = b_n + v \wedge z_n = c_1 + w \\
\wedge & \underline{X_1} \leq x_1 \leq \overline{X_1} \wedge \underline{Y_1} \leq y_1 \leq \overline{Y_1} \wedge \underline{Z_1} \leq z_1 \leq \overline{Z_1} \\
\wedge & \cdots \\
\wedge & \underline{X_n} \leq x_n \leq \overline{X_n} \wedge \underline{Y_n} \leq y_n \leq \overline{Y_n} \wedge \underline{Z_n} \leq z_n \leq \overline{Z_n} \\
\wedge & \left(\begin{array}{ll}
 & \max_{i=1..n}(\underline{X_i} - a_1) \leq \min_{i=1..n}(\overline{X_i} - a_1) \\
\vee & \max_{i=1..n}(\underline{Y_i} - b_1) \leq \min_{i=1..n}(\overline{Y_i} - b_1) \\
\vee & \max_{i=1..n}(\underline{Z_i} - c_1) \leq \min_{i=1..n}(\overline{Z_i} - c_1)
\end{array} \right)
\end{array} \right)
\end{array}
$$

$$
\begin{array}{ll}
\Leftrightarrow & \exists u \exists v \exists w \\
 & \left(\begin{array}{ll}
 & x_1 = a_1 + u \wedge y_1 = b_1 + v \wedge z_1 = c_1 + w \\
\wedge & \cdots \\
\wedge & x_n = a_n + u \wedge y_n = b_n + v \wedge z_n = c_1 + w \\
\wedge & \underline{X_1} \leq a_1 + u \leq \overline{X_1} \wedge \underline{Y_1} \leq b_1 + v \leq \overline{Y_1} \wedge \underline{Z_1} \leq c_1 + w \leq \overline{Z_1} \\
\wedge & \cdots \\
\wedge & \underline{X_n} \leq a_n + u \leq \overline{X_n} \wedge \underline{Y_n} \leq b_n + v \leq \overline{Y_n} \wedge \underline{Z_n} \leq c_n + w \leq \overline{Z_n} \\
\wedge & \left(\begin{array}{ll}
 & \max_{i=1..n}(\underline{X_i} - a_i) \leq \min_{i=1..n}(\overline{X_i} - a_i) \\
\vee & \max_{i=1..n}(\underline{Y_i} - b_i) \leq \min_{i=1..n}(\overline{Y_i} - b_i) \\
\vee & \max_{i=1..n}(\underline{Z_i} - c_i) \leq \min_{i=1..n}(\overline{Z_i} - c_i)
\end{array} \right)
\end{array} \right)
\end{array}
$$

$$
\begin{array}{ll}
\Leftrightarrow & \exists u \exists v \exists w \\
 & \left(\begin{array}{ll}
 & x_1 = a_1 + u \wedge y_1 = b_1 + v \wedge z_1 = c_1 + w \\
\wedge & \cdots \\
\wedge & x_n = a_n + u \wedge y_n = b_n + v \wedge z_n = c_1 + w \\
\wedge & \underline{X_1} - a_1 \leq u \leq \overline{X_1} - a_1 \wedge \underline{Y_1} - b_1 \leq v \leq \overline{Y_1} - b_1 \wedge \underline{Z_1} - c_1 \leq w \leq \overline{Z_1} - c_1 \\
\wedge & \cdots \\
\wedge & \underline{X_n} - a_n \leq u \leq \overline{X_n} - a_n \wedge \underline{Y_n} - b_n \leq v \leq \overline{Y_n} - b_n \wedge \underline{Z_n} - c_n \leq w \leq \overline{Z_n} - c_n \\
\wedge & \left(\begin{array}{ll}
 & \max_{i=1..n}(\underline{X_i} - a_i) \leq \min_{i=1..n}(\overline{X_i} - a_i) \\
\vee & \max_{i=1..n}(\underline{Y_i} - b_i) \leq \min_{i=1..n}(\overline{Y_i} - b_i) \\
\vee & \max_{i=1..n}(\underline{Z_i} - c_i) \leq \min_{i=1..n}(\overline{Z_i} - c_i)
\end{array} \right)
\end{array} \right)
\end{array}
$$

$$
\begin{array}{ll}
\Leftrightarrow & \exists u \exists v \exists w \\
& \left(\begin{array}{ll}
 & x_1 = a_1 + u \wedge y_1 = b_1 + v \wedge z_1 = c_1 + w \\
\wedge & \cdots \\
\wedge & x_n = a_n + u \wedge y_n = b_n + v \wedge z_n = c_1 + w \\
\wedge & \max_{i=1..n} \left(\underline{X_i} - a_i\right) \leq u \leq \min_{i=1..n} \left(\overline{X_i} - a_i\right) \\
\wedge & \max_{i=1..n} \left(\underline{Y_i} - b_i\right) \leq v \leq \min_{i=1..n} \left(\overline{Y_i} - b_i\right) \\
\wedge & \max_{i=1..n} \left(\underline{Z_i} - c_i\right) \leq w \leq \min_{i=1..n} \left(\overline{Z_i} - c_i\right) \\
\wedge & \left(\begin{array}{ll}
 & \max_{i=1..n} \left(\underline{X_i} - a_i\right) \leq \min_{i=1..n} (\overline{X_i} - a_i) \\
\vee & \max_{i=1..n} \left(\underline{Y_i} - b_i\right) \leq \min_{i=1..n} (\overline{Y_i} - b_i) \\
\vee & \max_{i=1..n} \left(\underline{Z_i} - c_i\right) \leq \min_{i=1..n} (\overline{Z_i} - c_i)
\end{array} \right)
\end{array} \right) \\
\Leftrightarrow & \left(\begin{array}{ll}
 & a_1 + \max_{i=1..n} \left(\underline{X_i} - a_i\right) \leq x_1 \leq a_1 + \min_{i=1..n} (\overline{X_i} - a_i) \\
\wedge & b_1 + \max_{i=1..n} \left(\underline{Y_i} - b_i\right) \leq y_1 \leq b_1 + \min_{i=1..n} (\overline{Y_i} - b_i) \\
\wedge & c_1 + \max_{i=1..n} \left(\underline{Z_i} - c_i\right) \leq z_1 \leq c_1 + \min_{i=1..n} (\overline{Z_i} - c_i) \\
\wedge & \cdots \\
\wedge & a_n + \max_{i=1..n} \left(\underline{X_i} - a_i\right) \leq x_n \leq a_n + \min_{i=1..n} (\overline{X_i} - a_i) \\
\wedge & b_n + \max_{i=1..n} \left(\underline{Y_i} - b_i\right) \leq y_n \leq b_n + \min_{i=1..n} (\overline{Y_i} - b_i) \\
\wedge & c_n + \max_{i=1..n} \left(\underline{Z_i} - c_i\right) \leq z_n \leq c_n + \min_{i=1..n} (\overline{Z_i} - c_i)
\end{array} \right)
\end{array}
$$

Ceci termine la démonstration : pour tout i pris entre 1 et n, on a alors

$$
\begin{array}{ll}
 & \text{apx}\Big(\text{proj}_{3i-2}(\textit{Translaté}\,(a_1, b_1, c_1, \ldots, a_n, b_n, c_n) \cap X_1 \times Y_1 \times Z_1 \times \cdots X_n \times Y_n \times Z_n)\Big) \\
= & \left[a_i + \max_{j=1..n} \left(\underline{X_j} - a_j\right), a_i + \min_{j=1..n} \left(\overline{X_j} - a_j\right)\right]
\end{array}
$$

$$
\begin{array}{ll}
 & \text{apx}\Big(\text{proj}_{3i-1}(\textit{Translaté}\,(a_1, b_1, c_1, \ldots, a_n, b_n, c_n) \cap X_1 \times Y_1 \times Z_1 \times \cdots X_n \times Y_n \times Z_n)\Big) \\
= & \left[b_i + \max_{j=1..n} \left(\underline{Y_j} - b_j\right), b_i + \min_{j=1..n} \left(\overline{Y_j} - b_j\right)\right]
\end{array}
$$

$$
\begin{array}{ll}
 & \text{apx}(\text{proj}_{3i}(\textit{Translaté}\,(a_1, b_1, c_1, \ldots, a_n, b_n, c_n) \cap X_1 \times Y_1 \times Z_1 \times \cdots X_n \times Y_n \times Z_n)) \\
= & \left[c_i + \max_{j=1..n} \left(\underline{Z_j} - c_j\right), c_i + \min_{j=1..n} \left(\overline{Z_j} - c_j\right)\right]
\end{array}
$$

3.4.2 Contrainte de placement

Nous pouvons maintenant traiter la contrainte *Classe*(f). Soit f une configuration de n points, on rappelle que *Classe*(f) est l'ensemble des $3n$-uplets d'entiers tels que

$$
\begin{array}{c}
(x_1, y_1, z_1, \ldots, x_n, y_n, z_n) \in \textit{Classe}(f) \\
\Updownarrow \\
\{(x_1, y_1, z_1), \ldots, (x_n, y_n, z_n)\} \in \textit{classe}\,(f) \text{ et} \\
((x_1, y_1, z_1), \ldots, (x_n, y_n, z_n)) \text{ est lexicographiquement croissant}
\end{array}
$$

Proposition : Soit $X_1, Y_1, Z_1, \ldots, X_n, Y_n, Z_n, X'_1, Y'_1, Z'_1, \ldots, X'_n, Y'_n, Z'_n$ des intervalles tels que $X_1, Y_1, Z_1, \ldots, X_n, Y_n, Z_n$, sont non vides et $X'_1 \times Y'_1 \times Z'_1 \times \ldots \times X'_n \times Y'_n \times Z'_n = \text{apx}\,(\textit{Classe}(f) \cap X_1 \times Y_1 \times Z_1 \times \ldots \times X_n \times Y_n \times Z_n)$,
soit $\{g_1, \ldots, g_{24}\} = \textit{pivotés}\,(f)$,
soit $r = \textit{Classe}(f) \cap X_1 \times Y_1 \times Z_1 \times \ldots \times X_n \times Y_n \times Z_n$,

soit $r_j = \textit{Translaté}(g_j) \cap X_1 \times Y_1 \times Z_1 \times \ldots \times X_n \times Y_n \times Z_n$,
soit $X_{1j} \times Y_{1j} \times Z_{1j} \times \cdots \times X_{nj} \times Y_{nj} \times Z_{nj} = \text{apx}(r_j)$,
soit J l'ensemble des $j \in \{1, \ldots, 24\}$ tels que $\text{apx}(r_j) \neq \emptyset$.

Alors pour tout $i \in 1..n$

$$\begin{pmatrix} X'_i \\ Y'_i \\ Z'_i \end{pmatrix} = \begin{cases} \begin{pmatrix} \emptyset \\ \emptyset \\ \emptyset \end{pmatrix}, \text{ si } J = \emptyset \\ \begin{pmatrix} \left[\min_{j \in J} \underline{X_{ij}},\ \max_{j \in J} \overline{X_{ij}}\right] \\ \left[\min_{j \in J} \underline{Y_{ij}},\ \max_{j \in J} \overline{Y_{ij}}\right] \\ \left[\min_{j \in J} \underline{Z_{ij}},\ \max_{j \in J} \overline{Z_{ij}}\right] \end{pmatrix}, \text{ sinon} \end{cases}$$

Ce résultat découle de la définition de *Translaté*(f), de l'égalité (1.1) et de l'égalité (1.4).

Chapitre 4

Résultats expérimentaux

4.1 Stratégie de résolution

Nous avons vu dans le premier chapitre que l'algorithme de résolution termine toujours si la formule entrée est bien sous forme normalisée. Toutefois, pour des raisons d'efficacité, il est préférable de donner un ordre de priorité à chacune des transformations. L'ordre choisi est celui de l'énoncé de ces règles.

Pour notre problème de placement des pentaminos, les règles 5 et 6 ont également été traitées spécialement, de la façon suivante :

- pour la règle 6, nous avons coupé en priorité l'intervalle I contenant l'inconnue m_i dont la borne inférieure est la plus petite, cet intervalle est coupé en deux intervalles $\{\underline{I}\}$ et $\left[\underline{I}+1, \overline{I}\right]$; essayant ainsi de placer en priorité les pentaminos en comblant les vides du pavé de plus faible abscisse, puis de plus faible ordonnée, puis de plus faible hauteur. Une fois que tous les intervalles des m_i sont réduits à un entier, toutes les autres valeurs du problème sont déterminées,
- la règle 5 de réduction des domaines a été appliquée en priorité sur les contraintes les plus simples (*i.e.* celles dont l'algorithme de calcul est de plus faible complexité par rapport au nombre de variables),

$$(5)' \quad \left(\begin{array}{ll} & (x_1,\ldots,x_n) \in \mathbf{r} \\ \wedge & x_1 \in I_1 \\ \wedge & \cdots \\ \wedge & x_n \in I_n \end{array}\right) \rightarrow \left(\begin{array}{ll} & \left(x_{d_{1,1}},\ldots,x_{d_{1,n_1}}\right) \in \mathbf{r}_1 \\ \wedge & \cdots \\ \wedge & \left(x_{d_{m,1}},\ldots,x_{d_{m,n_m}}\right) \in \mathbf{r}_m \\ \wedge & x_1 \in I'_1 \\ \wedge & \cdots \\ \wedge & x_n \in I'_n \end{array}\right)$$

TAB. 4.1 – La règle de réduction et de décomposition

- pour prendre en compte la décomposition dynamique de certaines contraintes, nous remplaçons la règle (5) par la règle (5′) de la table 4.1. Dans cette transformation, $n_1,\ldots,n_m$ sont des entiers strictement positifs tels que $n_1+\cdots+n_m=n$, la suite d'en-

tiers $(d_{1,1}, \ldots, d_{1,n_1}, \ldots, d_{m,1}, \ldots, d_{m,n_m})$ est une permutation de $(1, \ldots, n)$ et $r_1, \ldots, r_m$ sont des sous-ensembles de $\mathbf{Z}^n$ vérifiant l'égalité $((I_{d_{1,1}} \times \cdots \times I_{d_{1,n_1}}) \times \cdots \times (I_{d_{m,1}} \times \cdots \times I_{d_{m,n_m}})) \cap s = (I_{d_{1,1}} \times \cdots \times I_{d_{1,n_1}} \cap r_1) \times \cdots \times (I_{d_{m,1}} \times \cdots \times I_{d_{m,n_m}} \cap r_m)$ avec s l'ensemble des n-uplets $(x_{d_{1,1}}, \ldots, x_{d_{1,n_1}}, \ldots, x_{d_{m,1}}, \ldots, x_{d_{m,n_m}})$ tels que $(x_1, \ldots, x_n) \in r$.

La stratégie de choix de l'intervalle à couper et de la méthode de coupure est primordiale. Sélectionner les variables dans leur ordre d'apparition dans la formule et les couper simplement en leur milieu ne permettrait pas de trouver la moindre solution à notre problème.

	Avec décomposition	*Sans décomposition*
Avec priorité	5,5 secondes 12 700 appels 57 % du temps	15 secondes 12 419 appels 86 % du temps
Sans priorité	38 secondes 44 998 appels 76 % du temps	47 secondes 39 250 appels 95 % du temps

TAB. 4.2 – Efficacité de la stratégie de résolution

La table 4.2 donne le temps pris par l'algorithme de résolution de contraintes, le nombre de fois qu'est appelé l'algorithme de réduction de la contrainte d'être des entiers distincts et le pourcentage de temps pris par cette réduction par rapport au reste du programme pour trouver une première solution au problème 6×10. Elle illustre les effets de l'application ou non des deux dernières stratégies : donner la priorité aux contraintes les plus simples nous permet d'accélérer notre programme d'un facteur variant de 4 à 6, décomposer dynamiquement les contraintes permet de doubler voir tripler la vitesse d'exécution.

4.2 Premières solutions

Pour avoir une première idée de la complexité des problèmes sur les pentaminos, nous avons tout d'abord programmé un algorithme énumératif classique (décrit en appendice). Pour chaque combinaison de a, b et c telle que $a \times b \times c = 60$, la table 4.3 donne le nombre total de solutions trouvées, le temps pris par l'algorithme classique et le nombre de nœuds de son arbre de recherche. Nous considérons ici les solutions symétriques comme distinctes.

Nous calculons ensuite le temps et le nombre de coupures (nombre d'application de la règle de réécriture numéro 6) de notre algorithme de résolution appliqué à la contrainte décrivant le problème des pentaminos pour trouver les 8 solutions du problème 3×20, les 96 solutions du cas $2 \times 3 \times 10$ et les 100 premières solutions des autres cas. Ces chiffres sont mis en comparaison avec le temps et le nombre de nœuds de l'arbre de recherche de l'algorithme énumératif dans la table 4.4.

Nous voyons immédiatement que l'algorithme énumératif est de loin la méthode la plus efficace pour résoudre le problème des pentaminos. La dernière ligne de ce tableau nous laisse par contre espérer un espace de recherche plus réduit pour le problème $3 \times 4 \times 5$ qui est le cas le plus combinatoire.

dimensions du volume	*nombre de solutions*	*algorithme énumératif*	
		temps	*nœuds*
2×30	0	0 s	100
3×20	4×2	0 s	71 191
4×15	4×368	3 s	1 789 678
5×12	$4\times1\,010$	11 s	8 387 260
6×10	$4\times2\,339$	35 s	25 848 916
$2\times2\times15$	0	0 s	3 474
$2\times3\times10$	8×12	5 s	3 543 583
$2\times5\times6$	8×264	6 m 36 s	257 639 965
$3\times4\times5$	$8\times3\,940$	2 h 11 m	9 279 204 103

TAB. 4.3 – Mesures de temps de l'algorithme énumératif

dimensions du volume	*algorithme énumératif*		*notre algorithme*	
	temps	*nœuds*	*temps*	*coupures*
3×20	0 s	71 191	25 s	72 036
4×15	0 s	134 982	37 s	121 293
5×12	0 s	247 082	1 m	217 212
6×10	0 s	324 969	1 m 36 s	351 510
$2\times3\times10$	3 s	3 543 583	16 m 42 s	2 914 667
$2\times5\times6$	12 s	7 775 773	28 m 45 s	4 187 511
$3\times4\times5$	3 m 43 s	19 600 732	50 m 53 s	7 588 096

TAB. 4.4 – Comparaison entre l'algorithme énumératif et notre algorithme, pour les 100 premières solutions

Ces deux algorithmes ont été écrits en C, compilés avec gcc v2.95 et exécutés sur un Pentium III cadencé à 350 MHz avec 512 Ko de cache, sous le système d'exploitation Linux.

Ces deux algorithmes énumèrent bien trop de solutions (quatre fois trop en deux dimensions et huit fois trop en trois dimensions). Même en divisant par huit nos temps de calcul, l'algorithme de résolutions de contraintes sur les entiers sera encore dix fois plus lent que le classique ; mais il reste intéressant d'étudier l'élimination des solutions symétriques dans les deux cas et de voir les résultats de cette énumération sur nos temps de calcul.

4.3 Élimination des symétries en deux dimensions

Dans les problèmes de placements dans le plan chaque pentamino a 8 (les pentaminos F, L, P, N et Y), 4 (pour le T, le U, le V, le W et le Z), 2 (le I) ou bien 1 (le X) orientations possibles en fonction de ses symétries internes. Considérons l'ensemble S_1 des solutions où un des pentaminos π_i a une certaine orientations α et l'ensemble S_2 des solutions où l'orientation de ce même pentamino est obtenue par symétrie d'α par rapport à l'une des médianes du rectangle à paver.

Toute solution de S_2 est obtenue par combinaison de symétries par rapport aux médianes

du plan à paver d'une solution de S_1 (et vice-versa) (Si de plus le rectangle à paver était un carré, la remarque précédente s'appliquerait aussi pour les symétries de pentaminos par rapport à la diagonale du carré, ceci n'arrive jamais dans le cas des pentaminos, 60 n'étant pas un carré).

Pour éviter de parcourir un espace de recherche trop grand, nous choisissons un pentamino π_i ayant 8 orientations possibles. Parmi ses orientations $\alpha_1, \ldots, \alpha_8$, on choisis un sous-ensemble d'orientations qui ne sont pas symétriques l'une par rapport à l'autre, de taille maximale ; (il est de taille 2). On contraindra ce pentamino à ne pas se placer suivant les 6 orientations non retenues.

Nous évitons ainsi les solutions symétriques : si nous avions deux solutions symétriques, elles le seraient par symétrie par rapport à l'une des médianes du carré ; le pentamino π_8 aurait dans une des solution une orientation symétrique par rapport à cette même médiane à son orientation dans l'autre solution, ce qui est impossible car on a interdit ses orientations symétriques.

Pour nos calculs, nous avons choisi d'interdire les placements symétriques du pentamino F (voir la figure 4.1).

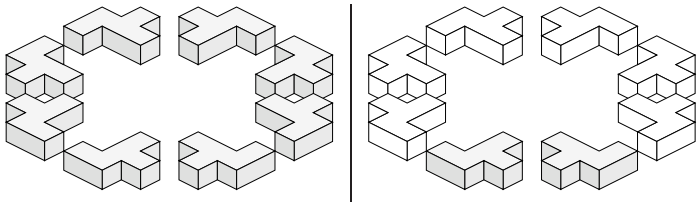


FIG. 4.1 – Choix de 2 orientations du F parmi les 8 possibles

Résultats

Nous obtenons les résultats présentés dans la table 4.5. On remarque que les espaces de recherches de ces deux programmes sont relativement similaires. L'élimination des symétries a divisé les temps de calculs et les espaces de recherche par trois, pour un nombre d'orientations d'une pièce divisé par quatre.

dimensions du volume	*nombre de solutions*	*algorithme énumératif*		*notre algorithme*	
		temps	*nœuds*	*temps*	*coupures*
2×30	0	0 s	100	0 s	0
3×20	2	0 s	38 792	12 s	32 809
4×15	368	1 s	708 508	3 m 56 s	771 425
5×12	1 010	4 s	3 285 507	17 m 03 s	3 532 267
6×10	2 339	12 s	9 918 168	46 m 27 s	9 752 037

TAB. 4.5 – Temps en deux dimensions, après élimination des symétries

4.4 Élimination des symétries en trois dimensions

Dans les problèmes de placements dans l'espace chaque pentamino a 24 (pour F, L, P, N et Y), 12 (T, U, V, W et Z), ou bien 3 (les pentaminos I et X) orientations possibles en fonctions de ses symétries internes. Considérons l'ensemble S_1 des solutions où un des pentaminos π_i a une certaine orientations α et l'ensemble S_2 des solutions où l'orientation de ce même pentamino est obtenue par symétrie d'α par rapport à l'un des trois plans médians (i.e. d'équation $x = (a+1)/2$, $y = (b+1)/2$ ou bien $z = (c+1)/2$) du bloc à paver.

Toute solution de S_2 est obtenue par une combinaisons de symétries par rapport à l'un de ces plans d'une solution de S_1 (et vice-versa) (Si le bloc à paver avait deux dimensions égales, la remarque précédente s'appliquerait aussi pour les symétries de pentaminos par rapport à des plans diagonaux ; mais dans le cas des pentaminos, on voit instantanément que le seul problème dans ce cas est le $2\times2\times15$ qui n'admet pas de solution, on ne se soucie donc pas non plus de cette remarque).

Pour éviter de parcourir un espace de recherche trop grand, on choisis un pentamino π_i ayant 24 orientations possibles. Parmi ses orientations $\alpha_1, \ldots, \alpha_{24}$, on choisis un sous-ensemble d'orientations qui ne sont pas symétriques l'une par rapport à l'autre, de taille maximale, cet ensemble est de taille 6. On contraint ce pentamino à ne pas se placer suivant les 18 orientations non retenues.

Si deux solutions du problème sont symétriques après avoir éliminé les 18 orientations, alors elles le sont par réflection par rapport au plan médian parallèle au plan contenant le pentamino choisi.

Pour nos expérimentations, c'est encore le pentamino F que nous avons sélectionné (voir la figure 4.2).

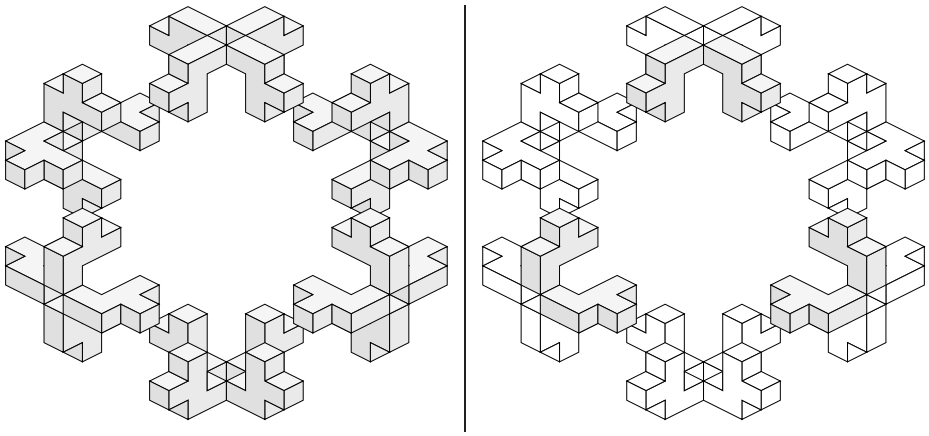


FIG. 4.2 – Choix de 6 orientations du F parmi les 24 possibles

Résultats

La table 4.6 nous donne les premiers résultats de cette élimination des symétries. Comme précédemment, nous sommes de deux à trois fois plus rapide, et l'espace de recherche des solutions a été diminué d'un même ordre de grandeur. L'ensemble des solutions du

problème $3 \times 4 \times 5$ est maintenant trouvé en moins de neuf jours en programmation par contraintes.

dimensions du volume	*nombre de solutions*	*algorithme énumératif*		*notre algorithme*	
		temps	*nœuds*	*temps*	*coupures*
$2\times2\times15$	0	0 s	3 474	0 s	0
$2\times3\times10$	2×12	3 s	1 860 942	9 m 42 s	1 559 135
$2\times5\times6$	2×264	2 m 45 s	114 424 641	25 h 9 m	102 516 710
$3\times4\times5$	$2\times3\,940$	50 m 16 s	2 039 115 519	216 h 30 m	1 672 290 710

TAB. 4.6 – Temps en trois dimensions, après une première élimination des symétries

4.4.1 Sans symétries pour les boîtes d'épaisseur 2

Pour les problèmes où $a = 2$ on peut facilement éliminer le reste des solutions symétriques en choisissant un pentamino π_j ne pouvant être contenu entièrement que dans un plan horizontal. L'ensemble des solutions S est alors constitué de l'ensemble S_1 des solutions où ce pentamino est contenu dans le plan inférieur et de l'ensemble S_2 de celles où il est contenu dans le plan supérieur. Ces deux ensembles sont disjoints (π_j est pour chaque solution contenu dans un seul de ces deux plans) et chaque élément de l'un est obtenu par symétrie d'un élément de l'autre (cette symétrie étant la réflection par rapport au plan horizontal $x = 1/2$).

Choisissons comme pentamino π_j le pentamino π_i précédent.

Si deux solutions de S_1 obtenues sont symétriques alors la position de π_i dans l'une ne peut être obtenue par symétrie de la position de π_i dans l'autre car on a supprimé les 18 orientations symétriques par rapport aux plans verticaux de π_i ainsi que la symétrie par rapport au plan horizontal. Deux solutions ne peuvent donc être symétriques.

En forçant le pentamino π_i à être dans le plan inférieur avec la contrainte $x_1 = \cdots = x_5 = 0$ on élimine donc toutes les solutions symétriques.

Ici encore, nous avons choisi le pentamino F pour nos mesures de temps.

Résultats

Nous obtenons enfin les temps et nombres de solutions dans la table 4.7. Ces résultats sont plus intéressants, on remarque que l'élimination des symétries est coûteuse en temps pour l'algorithme classique, bien que l'on ait restreint le domaine de valeurs possible de certaines variables, l'espace de recherche a quand à lui très peu diminué (de 10 à 15 % suivant les cas).

L'algorithme de résolution par contraintes est par contre accéléré, mais d'un ordre de grandeur bien inférieur à celui de la précédente élimination.

4.4.2 Quelques symétries en moins pour le problème $3\times4\times5$

Pour le problème $3\times4\times5$, on ne peut éliminer aussi facilement que précédemment les solutions symétriques introduites par le passage en trois dimensions. Nous découpons mainte-

dimensions du volume	*nombre de solutions*	*algorithme énumératif*		*notre algorithme*	
		temps	*nœuds*	*temps*	*coupures*
$2\times3\times10$	12	3 s	1 511 443	7 m 43 s	1 408 022
$2\times5\times6$	264	3 m 36 s	97 192 503	9 h 48 m	90 565 828

TAB. 4.7 – Temps pour les boîtes d'épaisseur 2, après élimination des symétries

nant le problème en 5 sous-problèmes, toujours suivant le placement du pentamino F (voir la figure 4.3) :

1. dans le plan horizontal inférieur ($x = 0$),
2. dans le plan horizontal médian ($x = 1$),
3. dans un des plans de la moitié gauche de la boîte ($y = 0$ ou $y = 1$),
4. dans un des plans de la partie avant de la boîte ($z = 0$ ou $z = 1$),
5. dans le plan vertical médian ($z = 2$).

On remarque que dans les cas 2 et 5, on obtient encore le double des solutions désirées, en raison de la symétrie par rapport aux plans médians.

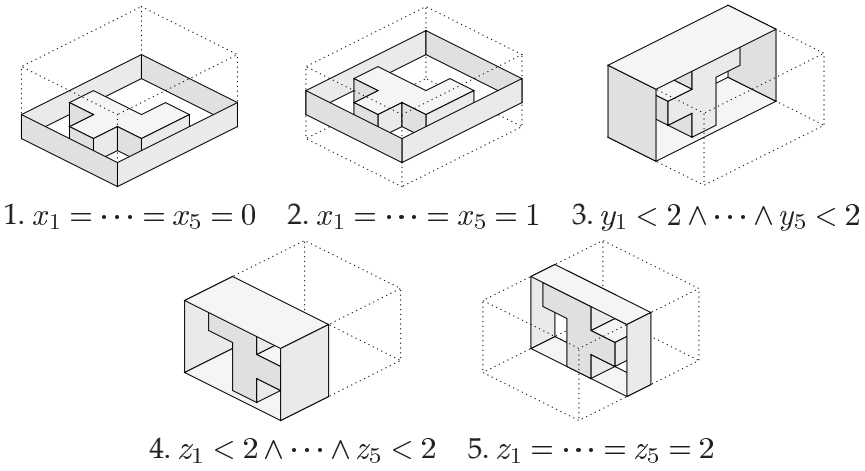


FIG. 4.3 – Décomposition du problème $3\times4\times5$ en 5 sous-problèmes

Résultats

Les mesures des résultats obtenus se trouvent dans la table 4.8. La décomposition en sous-problèmes a été plus gênante qu'autre chose pour l'algorithme énumératif, bien que l'espace de recherche ait été diminué de plus de 45 %, les temps de calcul de chaque sous-problème sont très proches des précédents et éliminer les solutions après l'énumération précédente aurait été bien plus efficace.

contrainte ajoutée	*nombre de solutions*	*algorithme énumératif*		*notre algorithme*	
		temps	*nœuds*	*temps*	*coupures*
$x_1 = \cdots = x_5 = 0$	1 207	42 m 54 s	1 373 875 178	117 h 36 m	1 065 181 264
$x_1 = \cdots = x_5 = 1$	2×322	44 m 59 s	1 338 513 351	117 h 33 m	1 070 316 969
$y_1 < 2 \wedge \cdots \wedge y_5 < 2$	1 185	53 m 44 s	1 388 531 635	87 h 51 m	796 027 174
$z_1 < 2 \wedge \cdots \wedge z_5 < 2$	1 150	56 m 42 s	1 380 943 824	12 h 47 m	121 259 983
$z_1 = \cdots = z_5 = 2$	2×76	47 m 46 s	1 225 618 582	5 h 43 m	55 311 242

TAB. 4.8 – Temps pour le problème $3 \times 4 \times 5$ décomposé

4.5 Illustration de quelques solutions

Nous venons de constater que notre algorithme de résolution de contraintes est bien moins efficace qu'une méthode de programmation classique. Il est toutefois utilisable pour obtenir un catalogue de solutions de l'ensemble des problèmes de placement des pentaminos (donné en annexe) et pour illustrer dans les pages qui suivent quelques solutions intéressantes.

Problèmes plans

Dans un numéro du *Problemist Fairy Chess Supplement* ([Han35]) Frans Hansson propose le problème 3×20 et en décrit les deux seules solutions (figure 4.4).

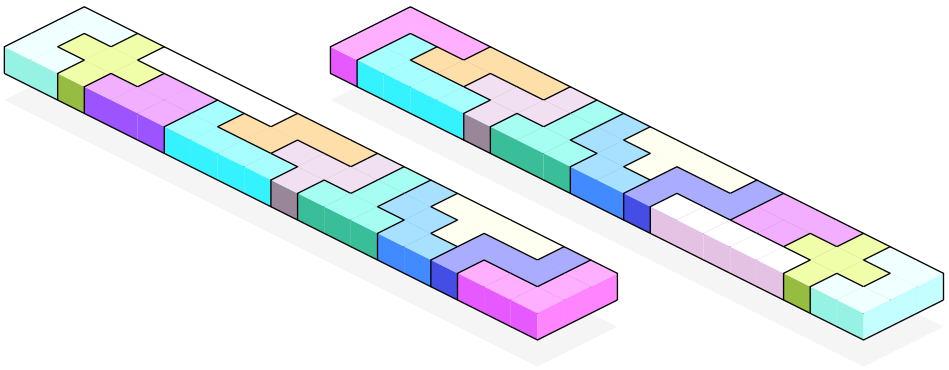


FIG. 4.4 – Les 2 solutions du problème 3×20

Dans le *Fairy Chess Review* [Ste54], Walter S. Stead propose et donne plusieurs solutions aux problèmes 3×20, 4×15, 5×12 et 6×10 (figures 4.5, 4.7 et 4.9), mais la totalité les 2 339 solutions du problème 6×10 ont été trouvées par C. B. Haselgrove et Jenifer Haselgrove par un programme décrit en [HH60], la découverte de l'ensemble des solutions des problèmes 4×15 et 5×12 leur est également attribuée.

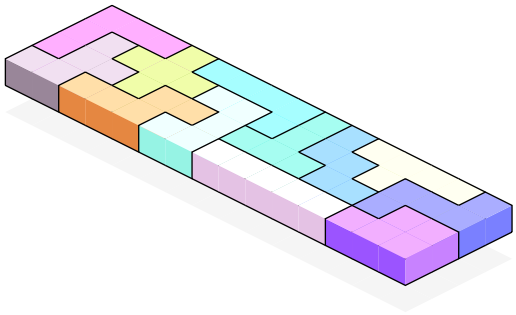


FIG. 4.5 – Une des 368 solutions du problème 4×15

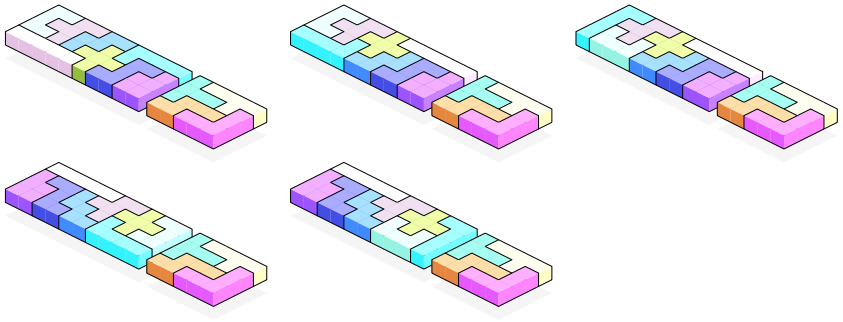


FIG. 4.6 – Cinq solutions du problème 4×15 composées d'un bloc 4×5 accolé à différents blocs 4×10, les quatre orientations et les deux positions (à gauche ou bien à droite du grand bloc) possibles de ce petit bloc nous donnent le total des 40 solutions composées possibles de ce problème

Problème d'épaisseur 2

C. J. Bouwkamp est le premier à trouver à l'aide de trois ordinateurs toutes les solutions des problèmes $2\times3\times10$, $2\times5\times6$ et $3\times4\times5$. Dans [Bou69], il décrit les 12 solutions du $2\times3\times10$ (figure 4.11). C. J. Bouwkamp énumère également les solutions du problème $2\times5\times6$ (figure 4.12) dans [Bou78] ; parmi celles-ci, on remarque huit solutions composées de deux blocs 5×6 (figure 4.13).

Le problème $3\times4\times5$

L'idée de placer les pentaminos dans un bloc de taille $3\times4\times5$ (figure 4.14) et une première solution sont données dans [Nix48]. C'est de nouveau C. J. Bouwkamp qui décrit toutes les solutions de ce problème, dans les 310 pages de [Bou67].

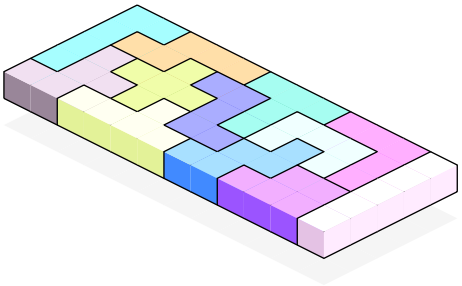


FIG. 4.7 – Une des 1 010 solution du problème 5×12

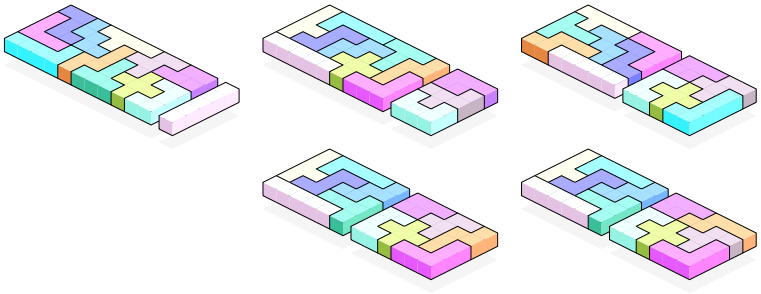


FIG. 4.8 – 224 solutions du problème 5×12 sont composées d'un bloc 5×11 auquel est accolé le pentamino I, 8 sont obtenues en accolant un bloc 5×3 à un bloc 5×9, 16 autres sont obtenues en juxtaposant des blocs 5×5 et 5×7 et enfin il existe 16 solutions composées de deux blocs 5×6, ces derniers permettant de trouver 8 solutions au problème $2\times5\times6$. À l'exception du I accolé à un bloc 5×11, cette figure permet de retrouver toutes ces solutions composées par déplacement et rotation des blocs

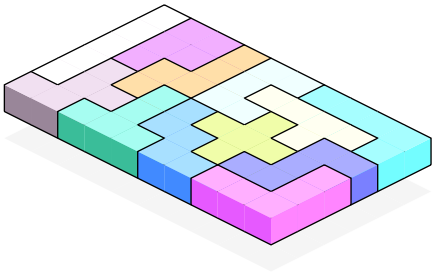


FIG. 4.9 – Une des 2 339 solutions du problème 6×10

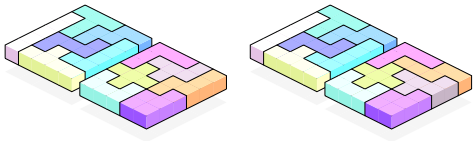


FIG. 4.10 – Seules 16 solutions au problème 6×10 sont composées. Elles sont la juxtaposition de deux bloc 6×5, les mêmes que pour le problème 5×12

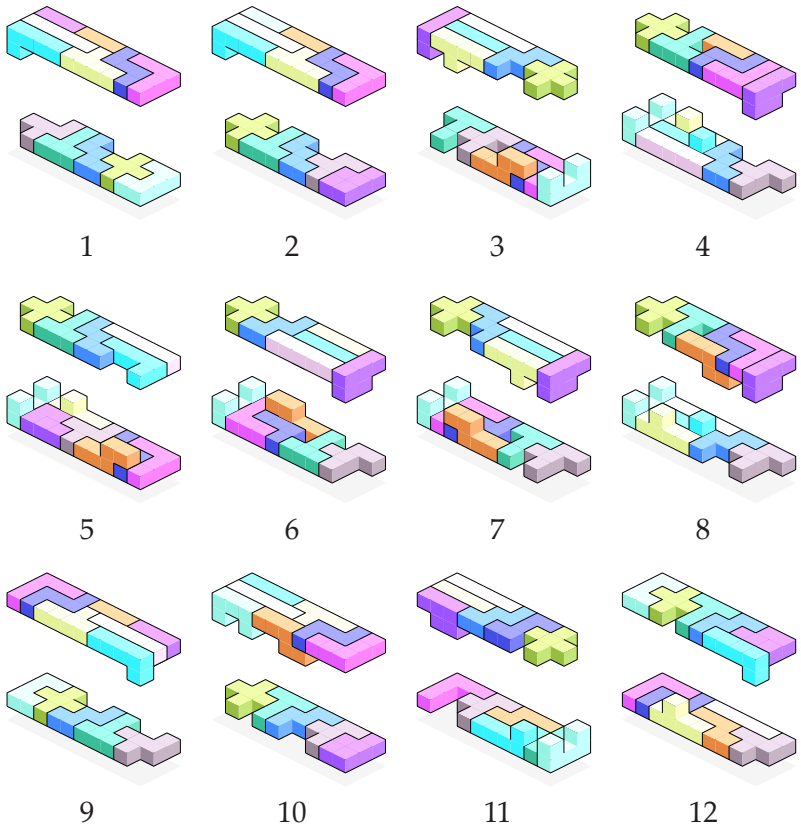


FIG. 4.11 – Les 12 solutions du problème $2\times3\times10$

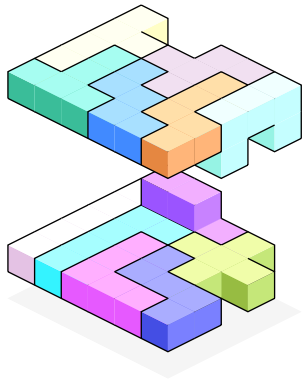


FIG. 4.12 – Une des 264 solutions du problème $2\times5\times6$

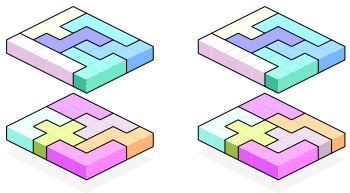


FIG. 4.13 – Deux solutions du problème $2 \times 5 \times 6$ composées de deux blocs distincts. Par rotations du bloc supérieur, on obtient un total de 8 solutions distinctes

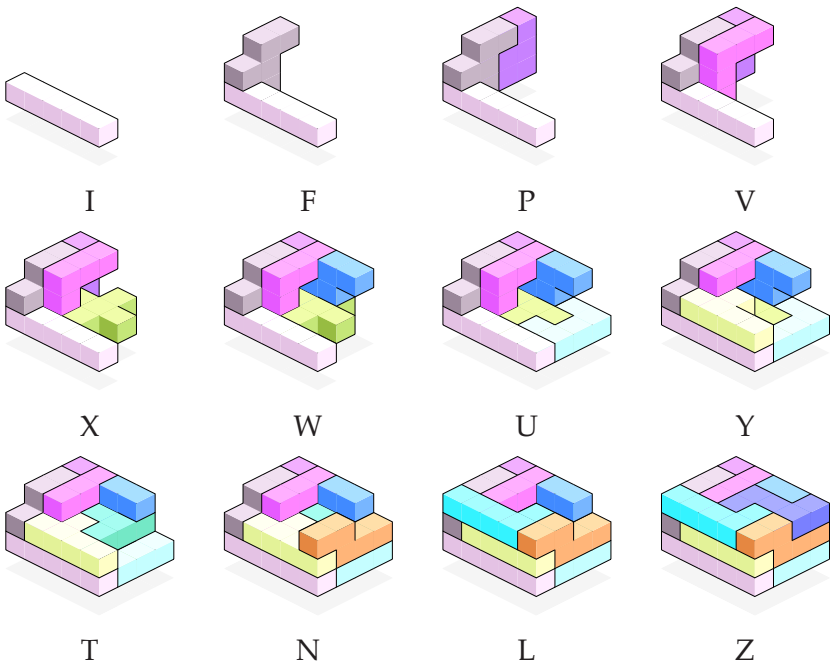


FIG. 4.14 – Construction pas à pas d'une des 3 940 solutions du problème $3 \times 4 \times 5$

Conclusion

Dans ce travail, nous avons cherché à mettre en évidence des contraintes utiles à la résolution de problèmes de placement d'objets dans l'espace. Pour cela, nous avons défini une méthodologie générale de résolution de contraintes par réduction d'intervalles d'entiers, à laquelle nous avons soumis le problème de placement des pentaminos sous diverses formes.

Deux contraintes globales se sont dégagées de cette étude : une contrainte de placement et une contrainte de points distincts. Nous avons donné la méthode de réduction de la contrainte de placement et avons utilisé une contrainte de permutation pour simplifier la contrainte d'être des points distincts.

L'utilisation de ces deux contraintes nous permet de résoudre les problèmes de pentaminos et d'éviter facilement de retrouver les solutions obtenues par symétries. Mais bien que suffisante pour calculer l'ensemble des solutions aux différents problèmes sur les pentaminos, cette formulation s'avère bien moins efficace qu'un algorithme classique par énumération exhaustive pour résoudre notre problème.

Références

[BGC97] N. Bleuzen-Guernalec and A. Colmerauer. Optimal narrowing of a block of sortings in optimal time. In G. Smolka, editor, *Lecture Notes in Computer Science, Principle and Practice of Constraint Programming, CP'97*, Austria, 1997. Springer.

[BO93] F. Benhamou and W. J. Older. Applying interval arithmetic to real, integer and boolean constraints. *Logic Programming : The ALP Newsletter*, 6(2) :13–14, May 1993. (Extended Abstract).

[Bou67] C. J. Bouwkamp. Catalogue of solutions of the rectangular $3\times4\times5$ solid pentomino problem. Technical report, Dept. of Math., Technische Hogeschool, Eindhoven, 1967. (reprinted 1981).

[Bou69] C. J. Bouwkamp. Packing a rectangular box with the twelve solid pentominoes. *Journal of Combinatorial Theory*, 7 :278–280, 1969.

[Bou78] C. J. Bouwkamp. Catalogue of solutions of the rectangular $2\times5\times6$ solid pentomino problem. In *Proc. Koninklitjke Nederlandse Akad. van Wetenschappen A81 :2*, pages 177–186, Eindhoven, 1978.

[Dud07] H. E. Dudeney. Problem 74 : The broken chessboard. In *The Canterbury Puzzles and Other Curious Problems*, pages 119–121, 220–221. Dover Publications, New York, 1907. (reprinted 1958).

[Gar58] M. Gardner. Solid polyominoes. *Scientific American*, 1958.

[Gol54] S. W. Golomb. Checker boards and polyominoes. *Amer. Math. Monthly*, 61 :675–682, 1954.

[Han35] F. Hansson. Problem 1844. *Problemist Fairy Chess Supplement*, 2(12, 13) :128, 135, 1935.

[HH60] C. B. Haselgrove and J. Haselgrove. A computer program for pentominoes. *Eureka*, (23) :16–18, 1960.

[Lec96] M. Leconte. A bounds-based reduction scheme for constraints of difference. In *Constraint-96, Second International Workshop on Constraint Based Reasoning*, Key West, Florida, 1996.

[Nix48] D. Nixon. Problem 7560. *Fairy Chess Review*, 6(16, 17) :12, 131, 1948.

[Ste54] W. S. Stead. Dissection. *Fairy Chess Review*, 9(1) :2–4, 1954.

Annexe A

L'algorithme énumératif sans contraintes

La méthode la plus efficace de placement des pentaminos s'est révélée être un algorithme énumératif classique. Le principe de cet algorithme est de remplir récursivement le volume en plaçant à chaque itération un des douze pentaminos dans une de ses 3, 12 ou 24 orientations possibles.

Comme pour le choix de la coupure dans l'algorithme de résolution par contraintes, cette orientation du pentamino est placée dans l'espace de telle façon qu'un de ses cubes occupe la case vide de plus faible abscisse, puis de plus faible ordonnée, puis de plus faible hauteur du bloc (voir la figure A.1).

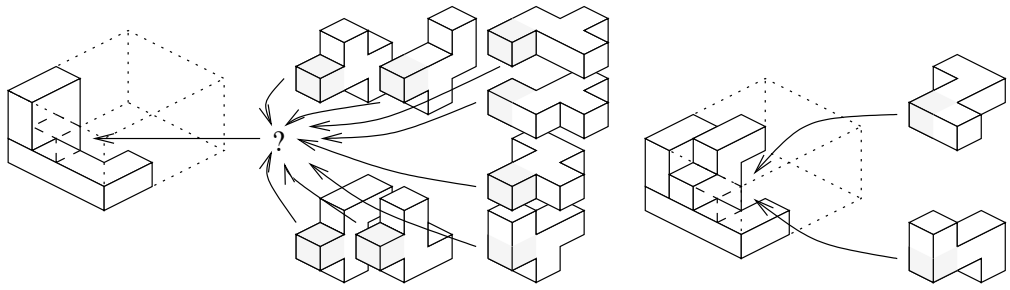


FIG. A.1 – Choix et placement d'une des huit orientationS plaçables du pentamino F puis des deux du Z à partir du point libre de plus petites coordonnées

Comme pour la contrainte décrivant le problème des pentaminos, le volume à paver a été linéarisé, cette fois-ci par l'application

$$(x, y, z) \mapsto x + (a+1)\, y + 2b\,(a+1)\, z$$

Le volume est ainsi « découpé » en plusieurs rectangles suivant sa hauteur, chaque rectangle est ensuite séparé en lignes horizontales de cubes ; ces lignes sont ensuite mises bout à bout en intercalant un cube entre chaque ligne d'un même rectangle et $b(a+1)$ cubes entre deux lignes d'un rectangle différent. L'intercalement de cubes supplémentaires servant à éviter aux pentaminos de « sortir » du volume par une de ses faces et d'y « entrer » par la face opposée.

Ce volume linéarisé est alors stocké dans une liste de variables et constantes $l_1 \cdot l_2 \cdots l_{2bc(a+1)}$ où les l_i sont :

- des variables libres quand il existe un triplet $(x, y, z) \in [0, a-1] \times [0, b-1] \times [0, c-1]$ tel que $i = x + (a+1)\, y + 2b\,(a+1)\, z$,
- égaux à une constante, fixée arbitrairement, sinon.

La figure A.2 donne un aperçu de cette linéarisation pour une boîte de dimensions $3 \times 4 \times 5$.

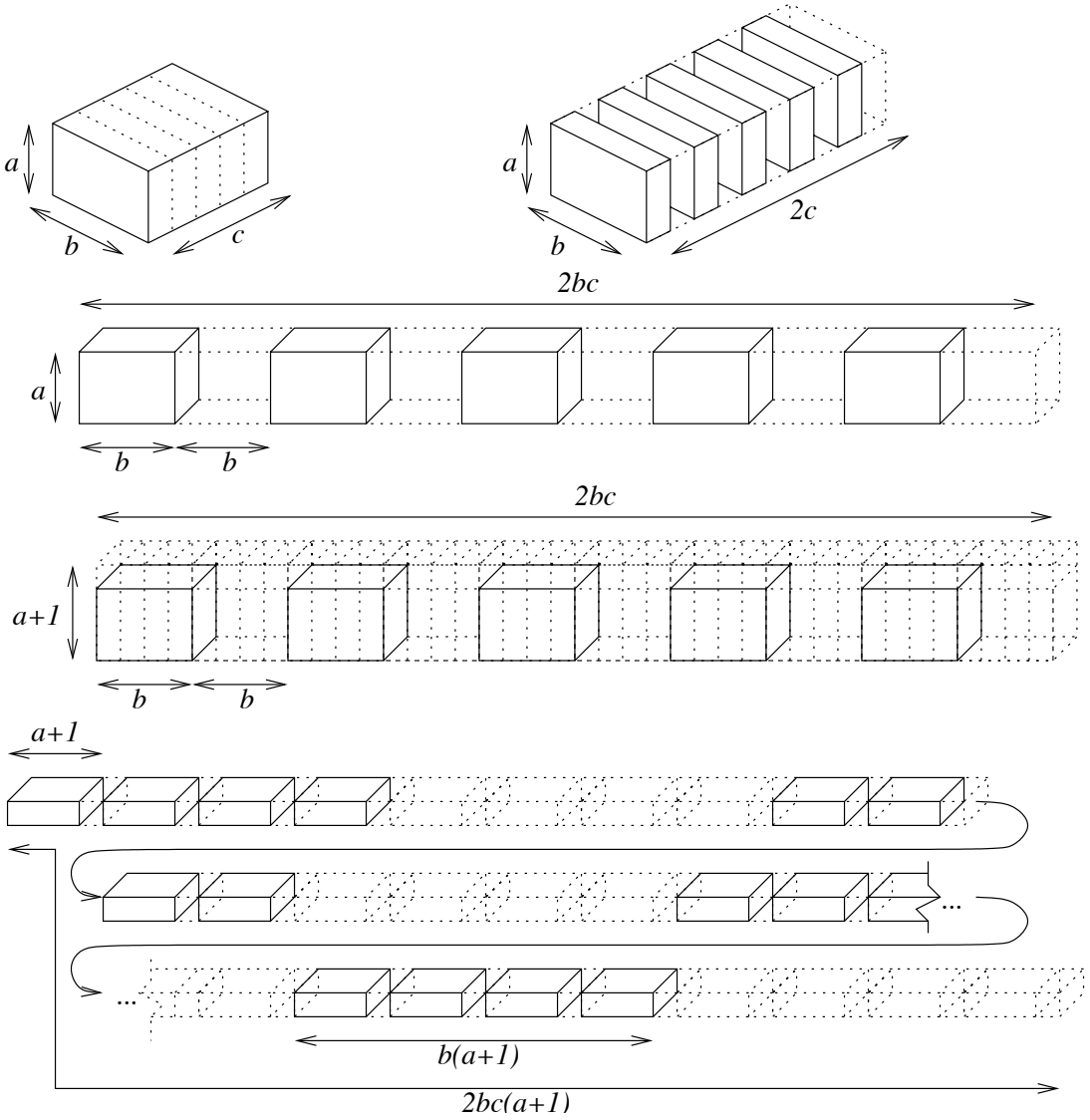


FIG. A.2 – Linéarisation du volume $3 \times 4 \times 5$

Les pentaminos de base $\pi_1, \ldots, \pi_{12}$ sont eux aussi linéarisés de la façon suivante : on calcule les 24 orientations dans l'espace de chaque pentamino π_i, puis chacune de ces orientations est linéarisée avec l'application précédente, mais à l'inverse du volume, les cases occupées par un cube du pentamino seront des constantes (égales à la lettre représentant le pentamino) et celles non occupées seront des variables libres. De chacune de ces listes, on

retire les premiers éléments qui sont des variables libres ; puis, pour chaque pentamino, on retire les listes d'orientations qui sont en plusieurs exemplaires, c'est à dire les orientations du pentaminos obtenues par des symétries internes.

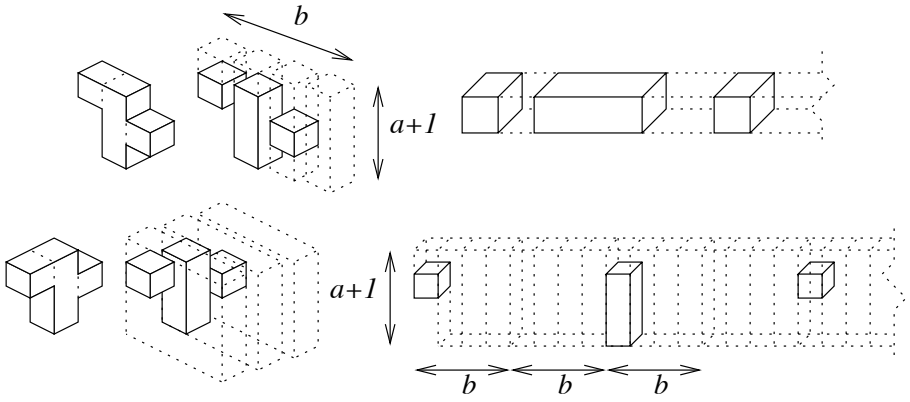


FIG. A.3 – Linéarisation de deux des orientations du pentamino F

Un pentamino est donc stocké comme la liste de ses orientations, chaque orientation étant la liste des cases qu'il occupe (les constantes) et des cases qu'il laisse libres (les variables libres). La figure A.3 donne une idée de cette linéarisation pour le pentamino F.

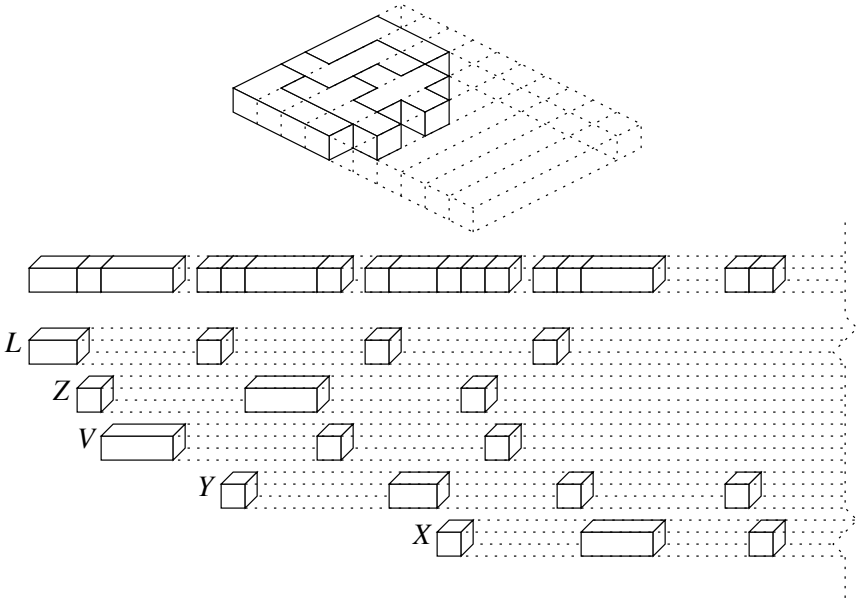


FIG. A.4 – Placement des pentaminos L, Z, V, Y et X dans une boîte 6×10

Pour résoudre notre problème, il va falloir mettre le volume et les orientations des pentaminos en relations. Le placement des pentaminos sera alors l'unification d'un choix d'orientations de ces douze pentaminos avec des sous-listes de la liste décrivant le volume. L'ensemble des solutions est donc l'ensemble de toutes ces unifications possibles (voir la figure

A.4).

Le cœur de l'algorithme énumératif peut se décrire par le programme Prolog suivant :

```
remplis(Volume, []).
remplis([Case | Volume], LesPentaminos):-
    nonvar(Case), !,
    remplis(Volume, LesPentaminos).
remplis(Volume, LesPentaminos):-
    element(UnPentamino, LesPentominos, PentaminosRestants),
    element(Orientation, UnPentomino),
    Orientation = Volume,
    remplis(Volume, PentaminosRestants).
```

où le prédicat `element(X, L1, L2)` signifie que `L2` est la liste `L1` privée de l'élément `X` et le prédicat `element(X, L)` signifie que `X` est un élément de la liste `L`.

Voici un exemple d'utilisation de ce prédicat dans le cas où le volume où l'on veut placer les pentaminos est un plateau de taille 6×10 :

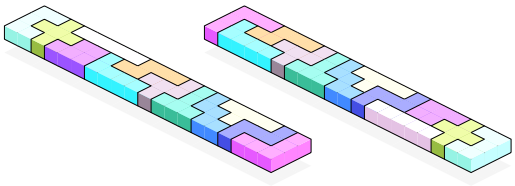
```
?- remplis( [ _,_,_,_,_,_,_,_,_,_,o,
              _,_,_,_,_,_,_,_,_,_,o,
              _,_,_,_,_,_,_,_,_,_,o,
              _,_,_,_,_,_,_,_,_,_,o,
              _,_,_,_,_,_,_,_,_,_,o,
              _,_,_,_,_,_,_,_,_,_,o,
              o,o,o,o,o,o,o,o,o,o,o ],
   [F, I, L, P, N, T, U, V, W, X, Y, Z]).
```

les variables `F`, `I`, `L`, `P`, `N`, `T`, `U`, `V`, `W`, `X`, `Y` et `Z` ayant été préalablement déclarées comme la liste des orientations possibles de chacun des douze pentaminos. Par exemple, le T a été défini ainsi :

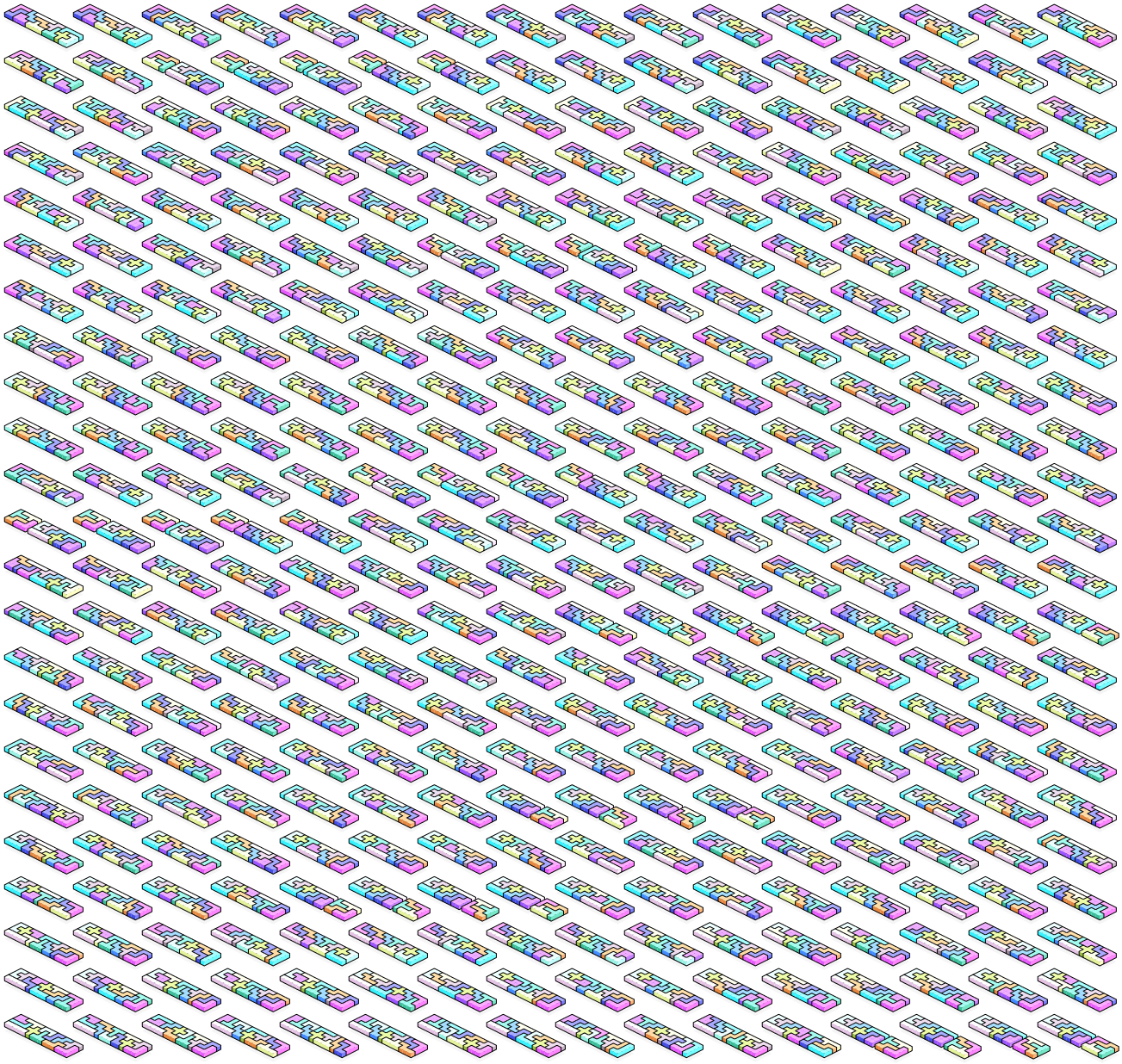
```
T = [[ t,_,_,_,_,_,_,_,_,_,_,
       t,t,t,_,_,_,_,_,_,_,_,
       t                     |_ ],
     [     t,_,_,_,_,_,_,_,_,_,
       t,t,t,_,_,_,_,_,_,_,_,
       _,_,t                 |_ ],
     [ t,t,t,_,_,_,_,_,_,_,_,
       _,t,_,_,_,_,_,_,_,_,_,
       _,t                   |_ ],
     [   t,_,_,_,_,_,_,_,_,_,_,
       _,t,_,_,_,_,_,_,_,_,_,
       t,t,t                 |_ ]].
```

Annexe B

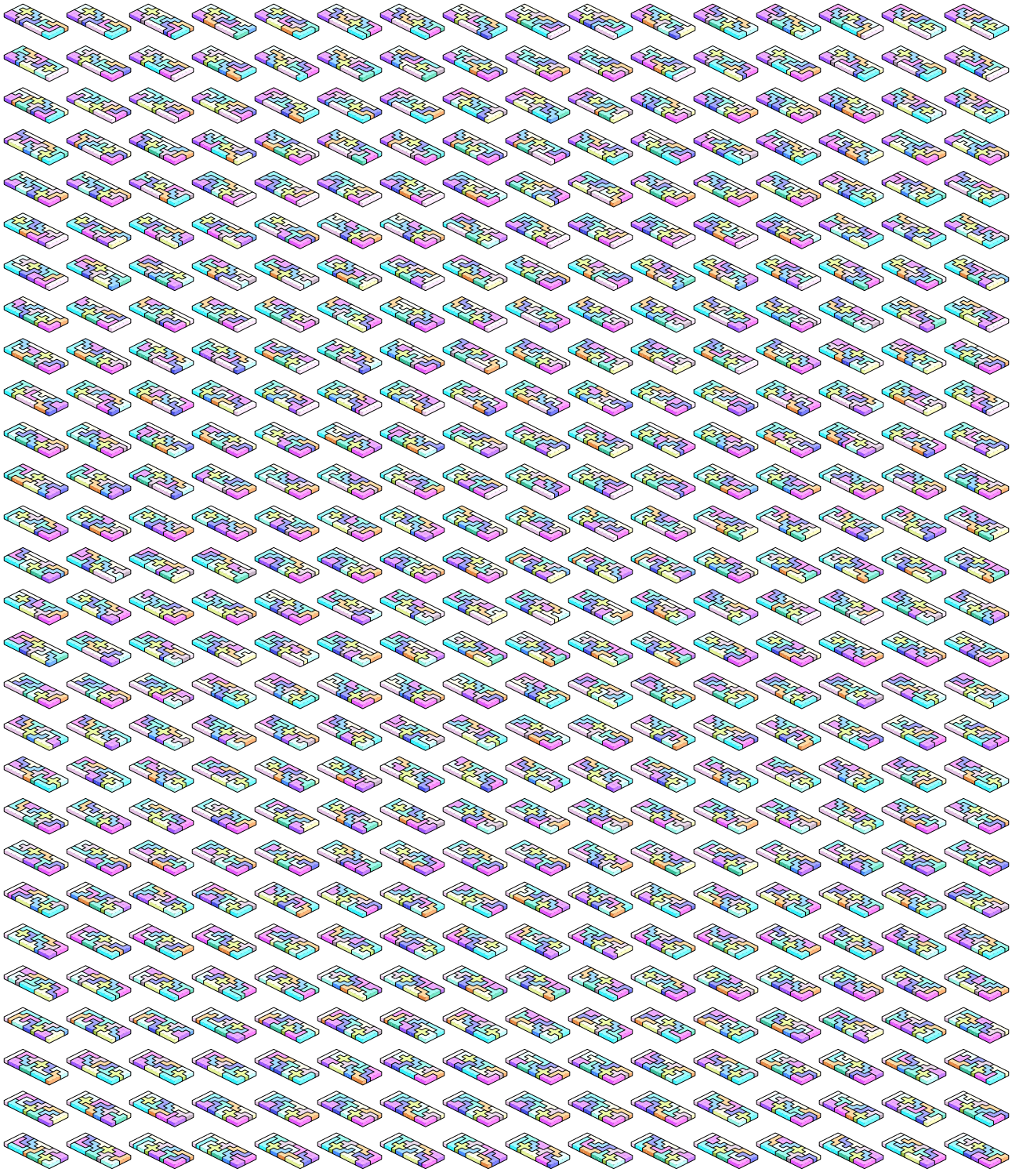
Catalogue de solutions



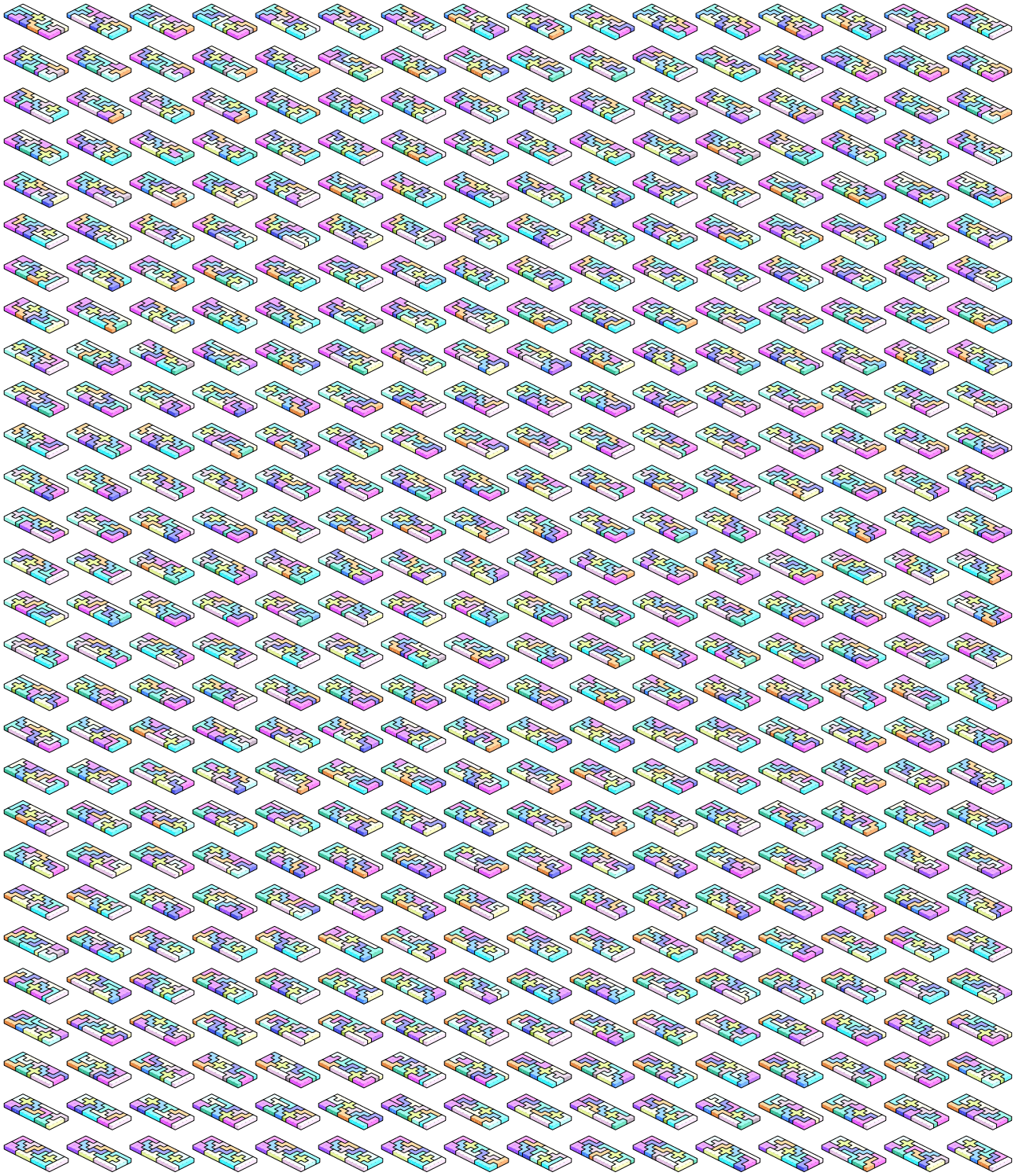
Solutions du problème 3×20



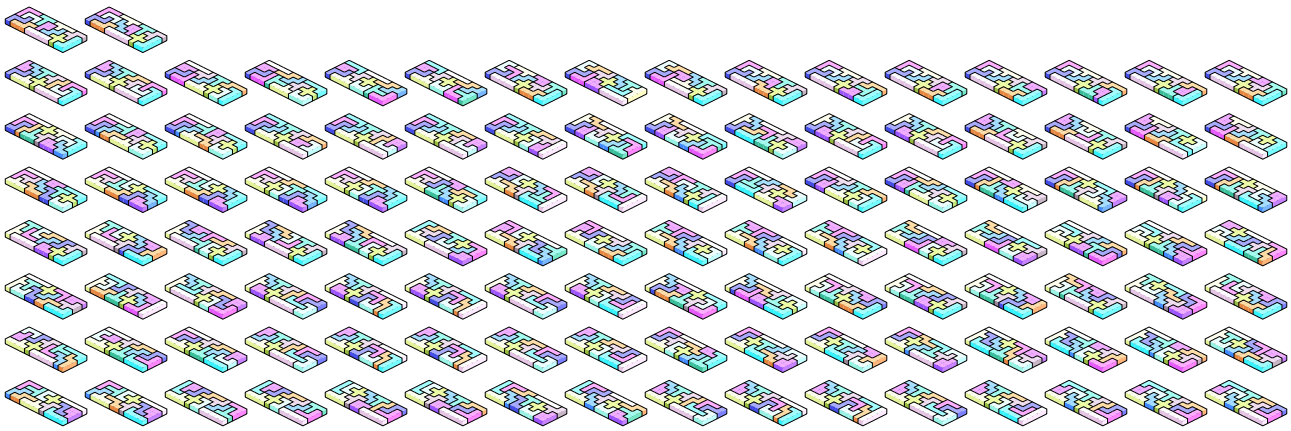
Solutions du problème 4×15



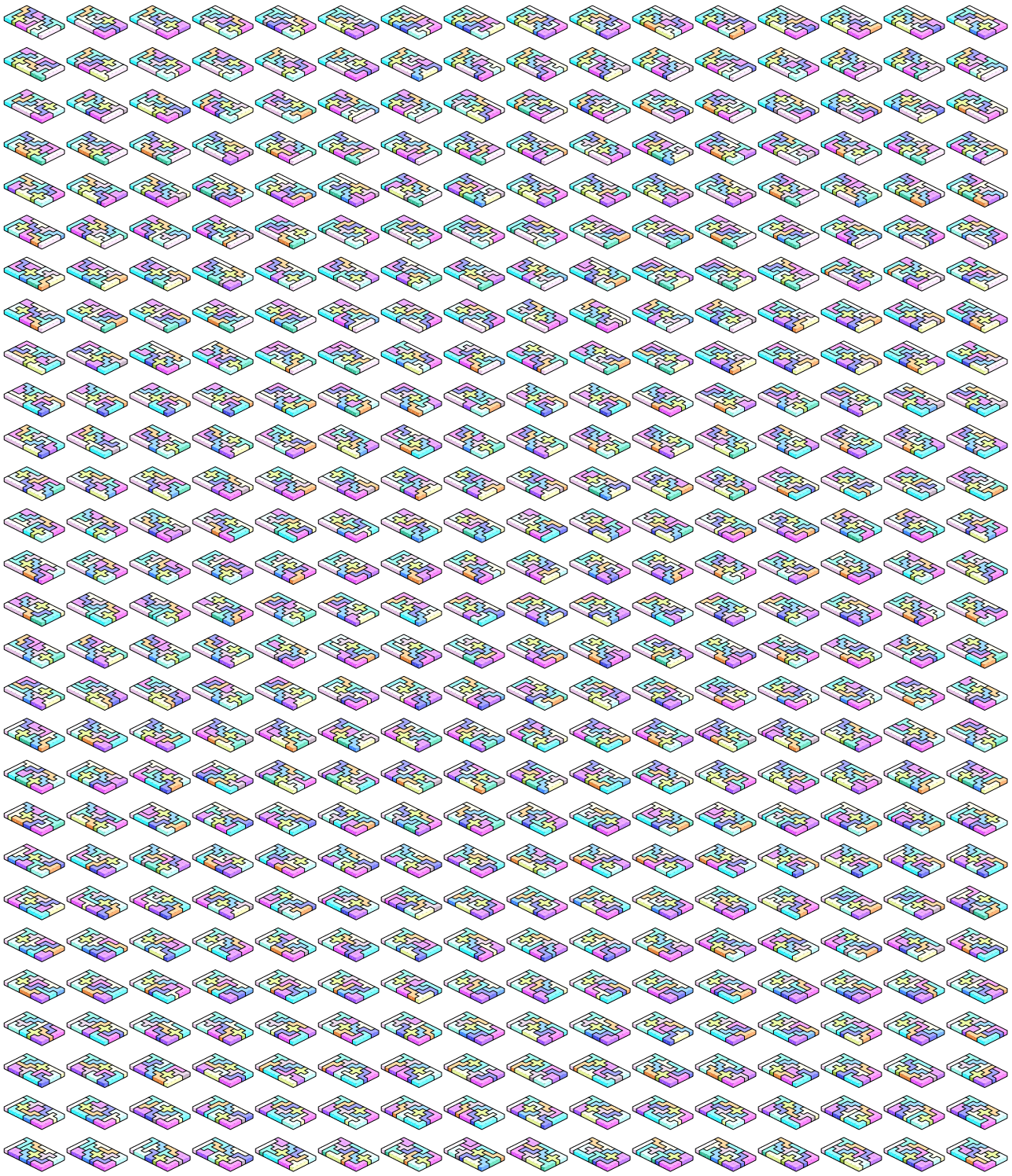
Solutions du problème 5×12 *(1/3)*



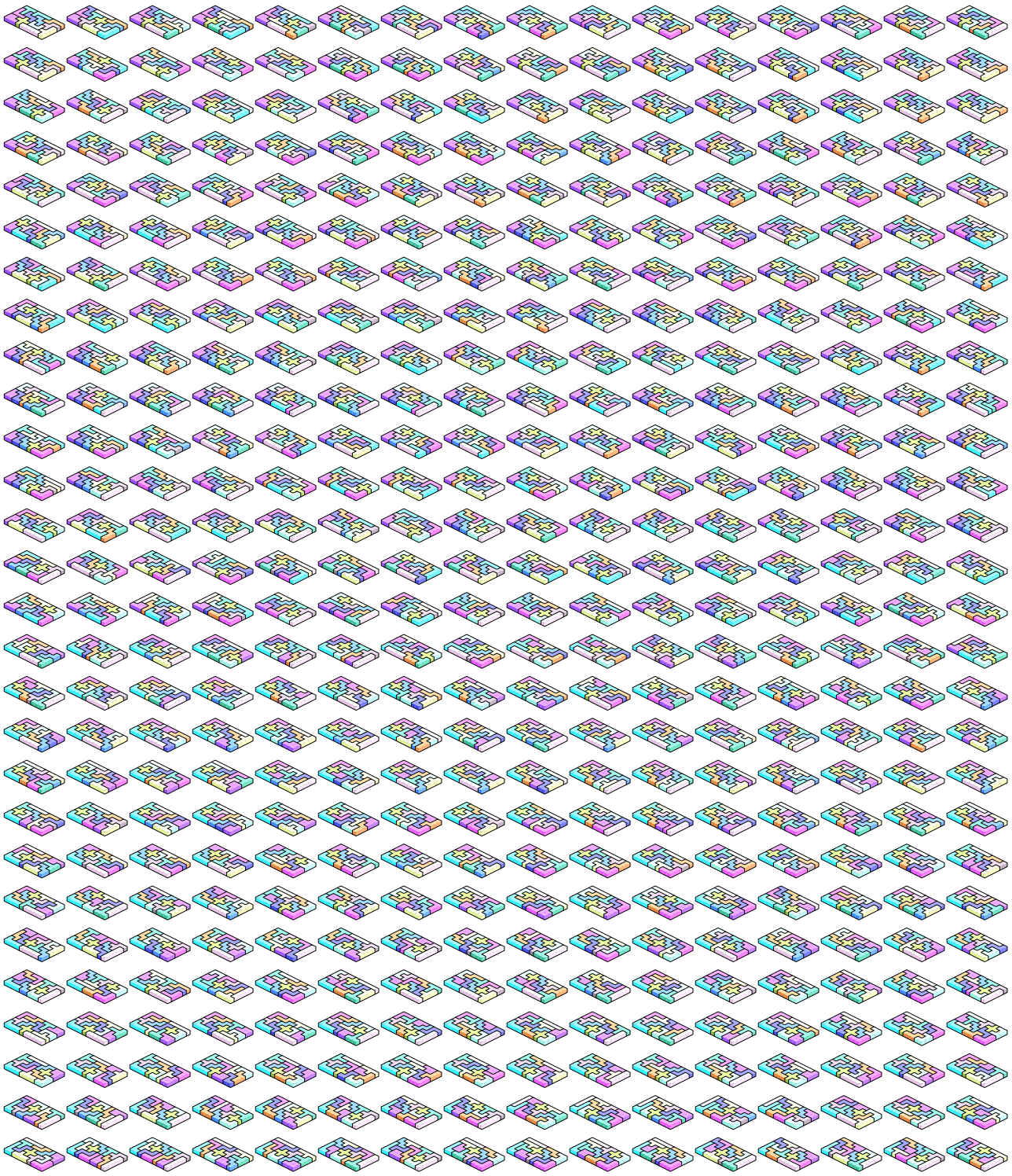
Solutions du problème 5×12 *(2/3)*



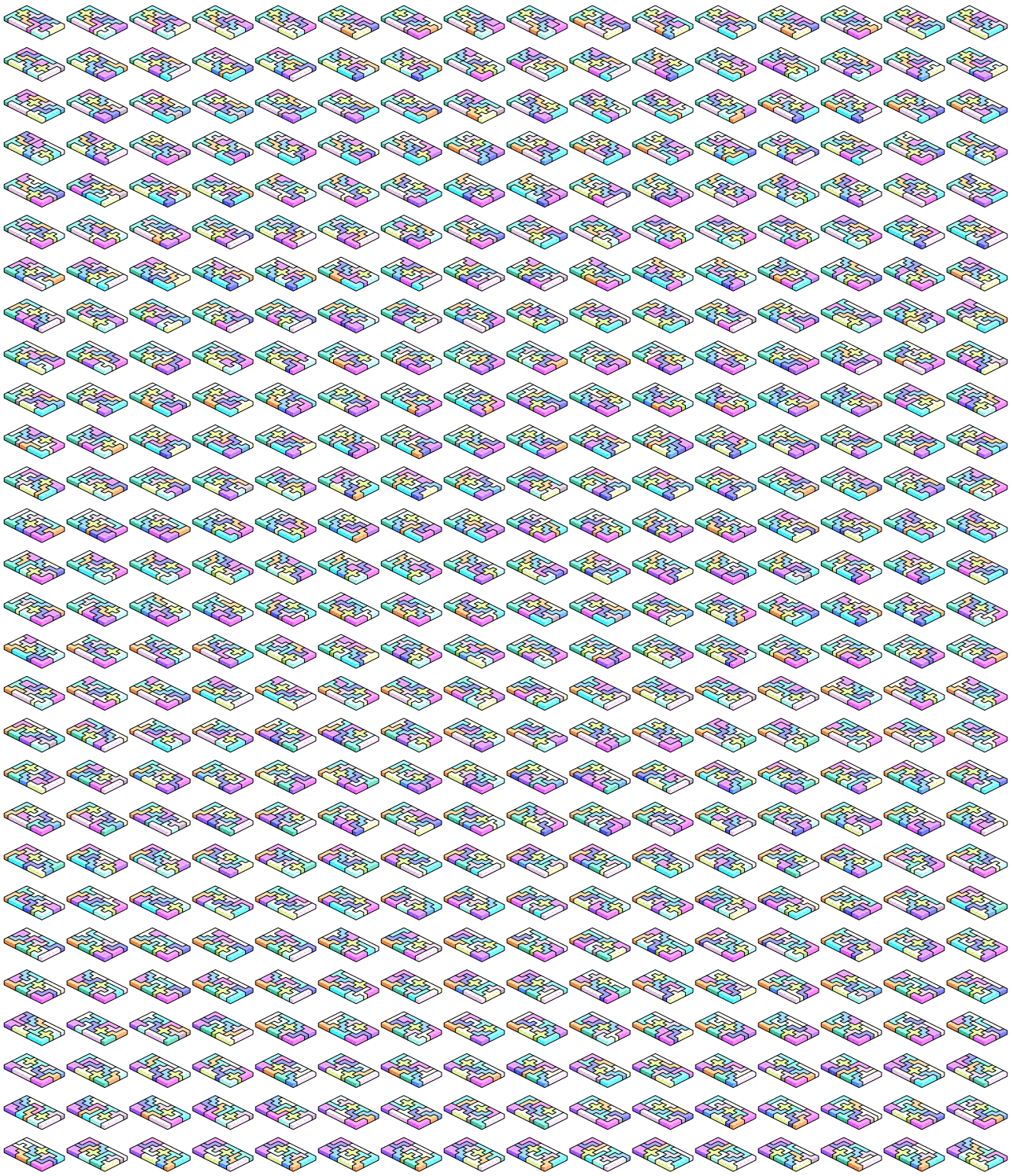
Solutions du problème 5×12 *(3/3)*



Solutions du problème 6×10 *(1/6)*



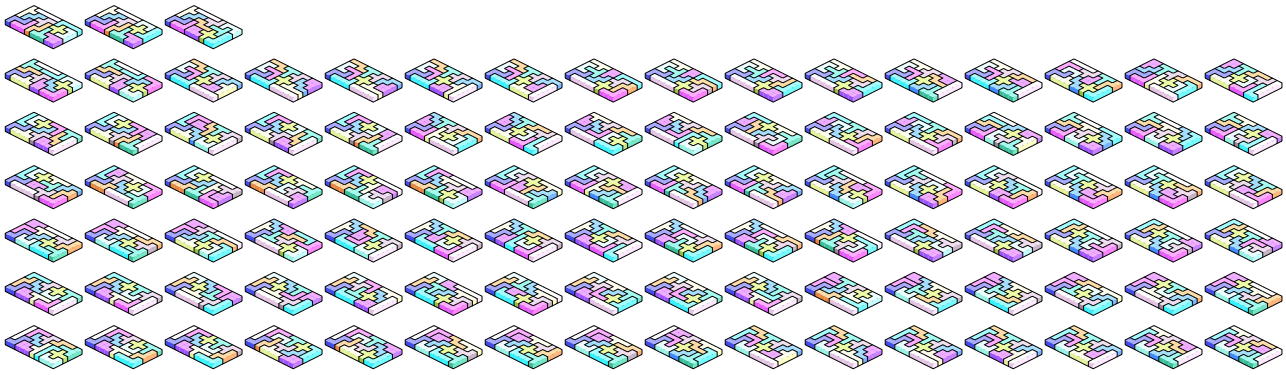
Solutions du problème 6×10 *(2/6)*



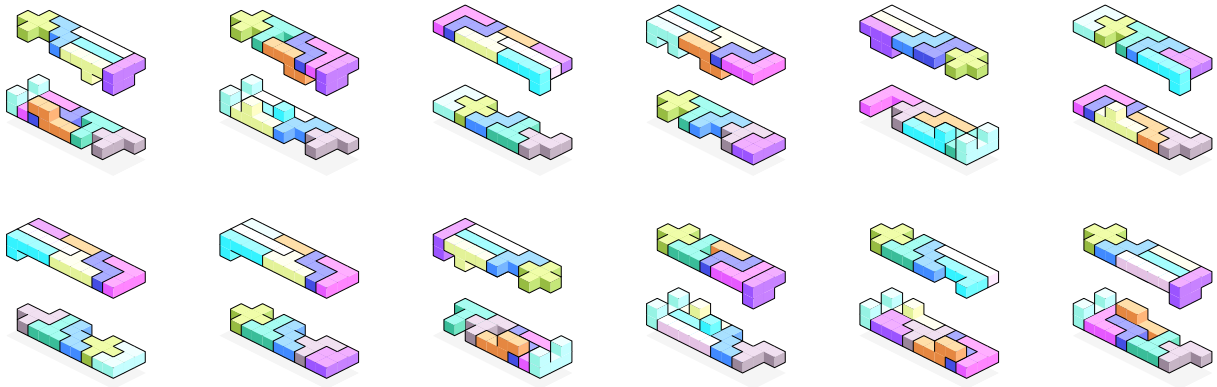
Solutions du problème 6×10 *(3/6)*

Solutions du problème 6×10 *(4/6)*

Solutions du problème 6×10 *(5/6)*



Solutions du problème 6×10 *(6/6)*



Solutions du problème $2 \times 3 \times 10$

Solutions du problème $2\times5\times6$ *(1/2)*

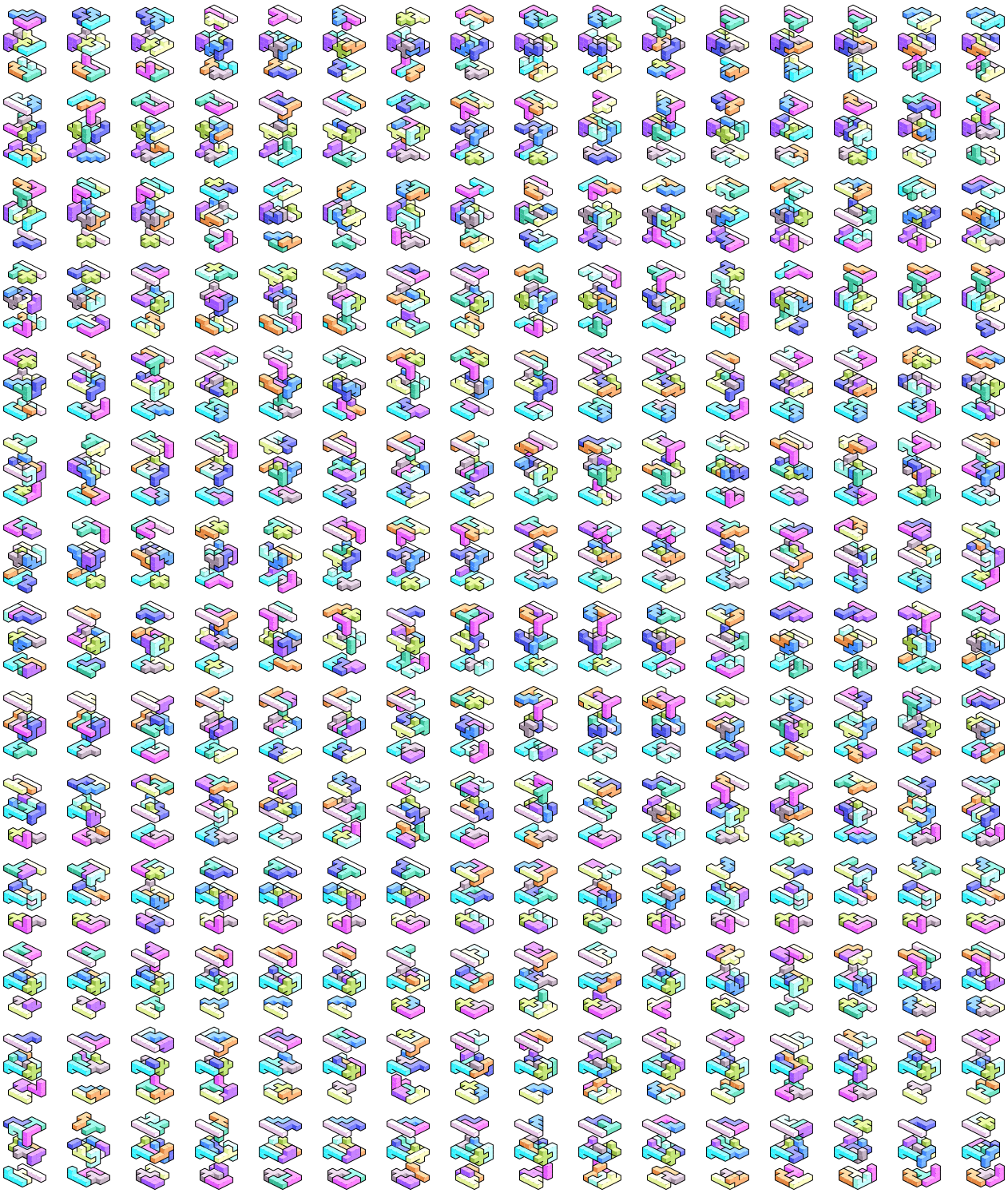
Solutions du problème $2 \times 5 \times 6$ *(2/2)*

Solutions du problème $3\times4\times5$ *(1/20)*

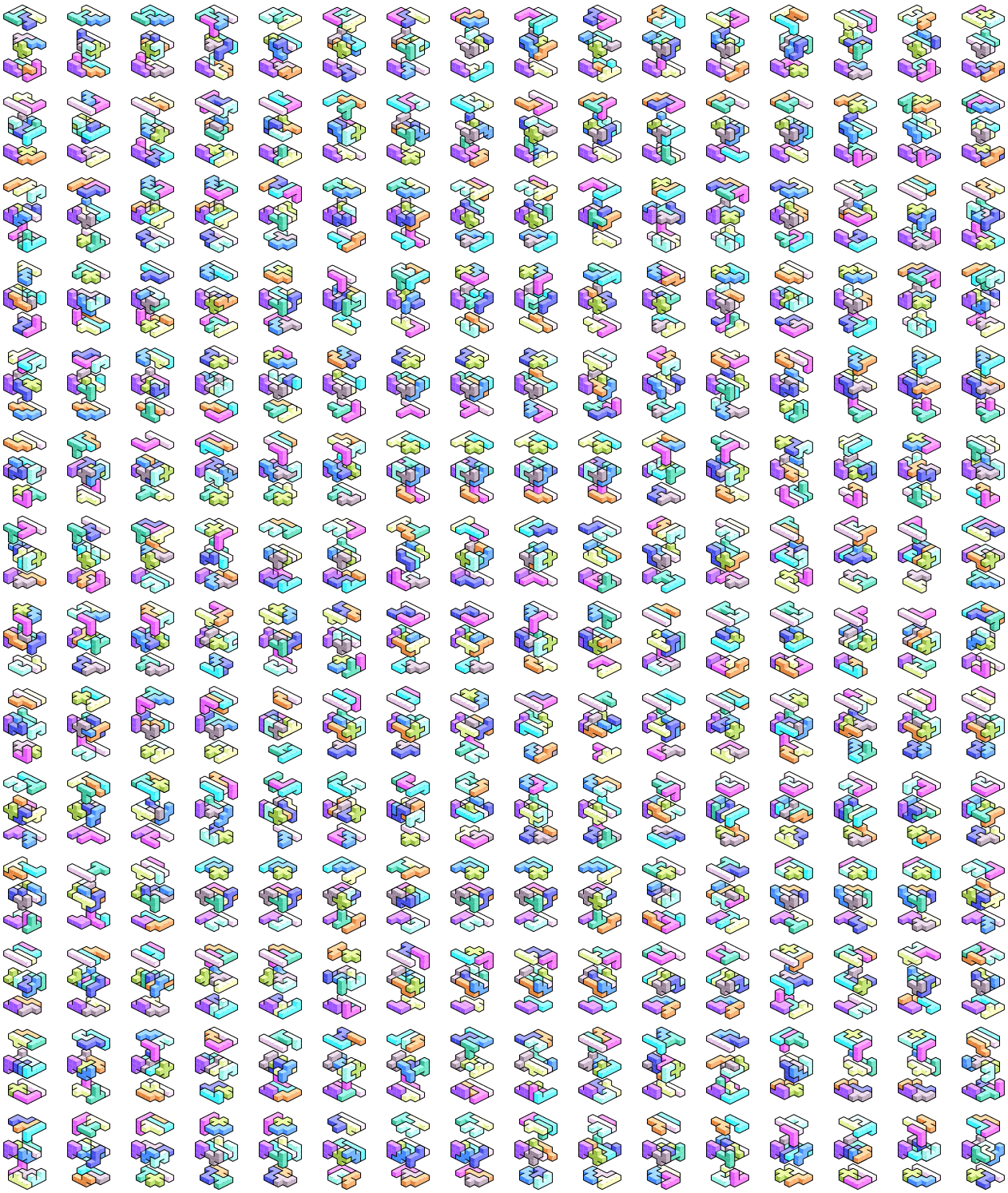
Solutions du problème $3\times4\times5$ *(2/20)*

Solutions du problème $3\times4\times5$ *(3/20)*

Solutions du problème $3\times4\times5$ *(4/20)*

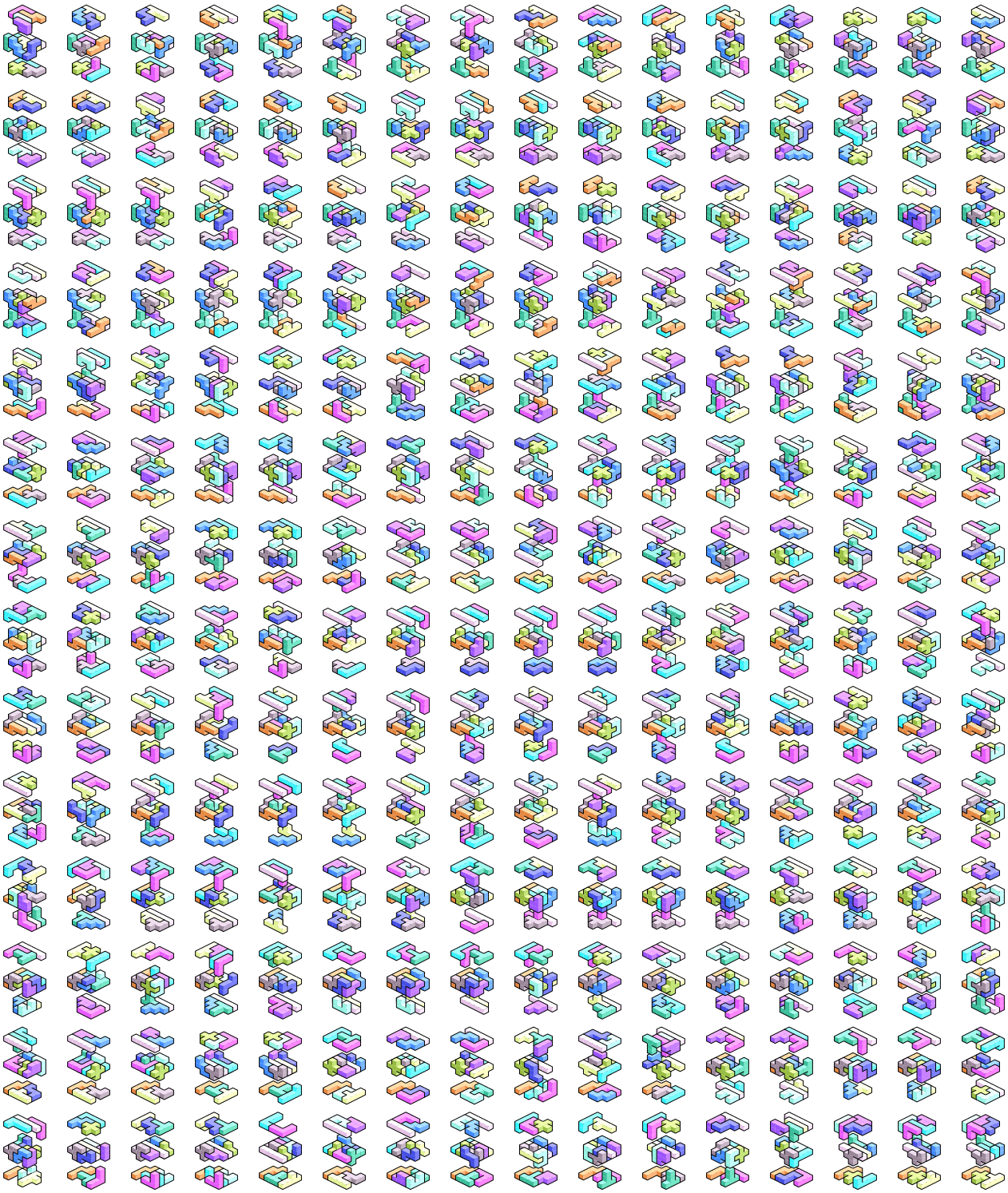


Solutions du problème $3\times4\times5$ *(5/20)*

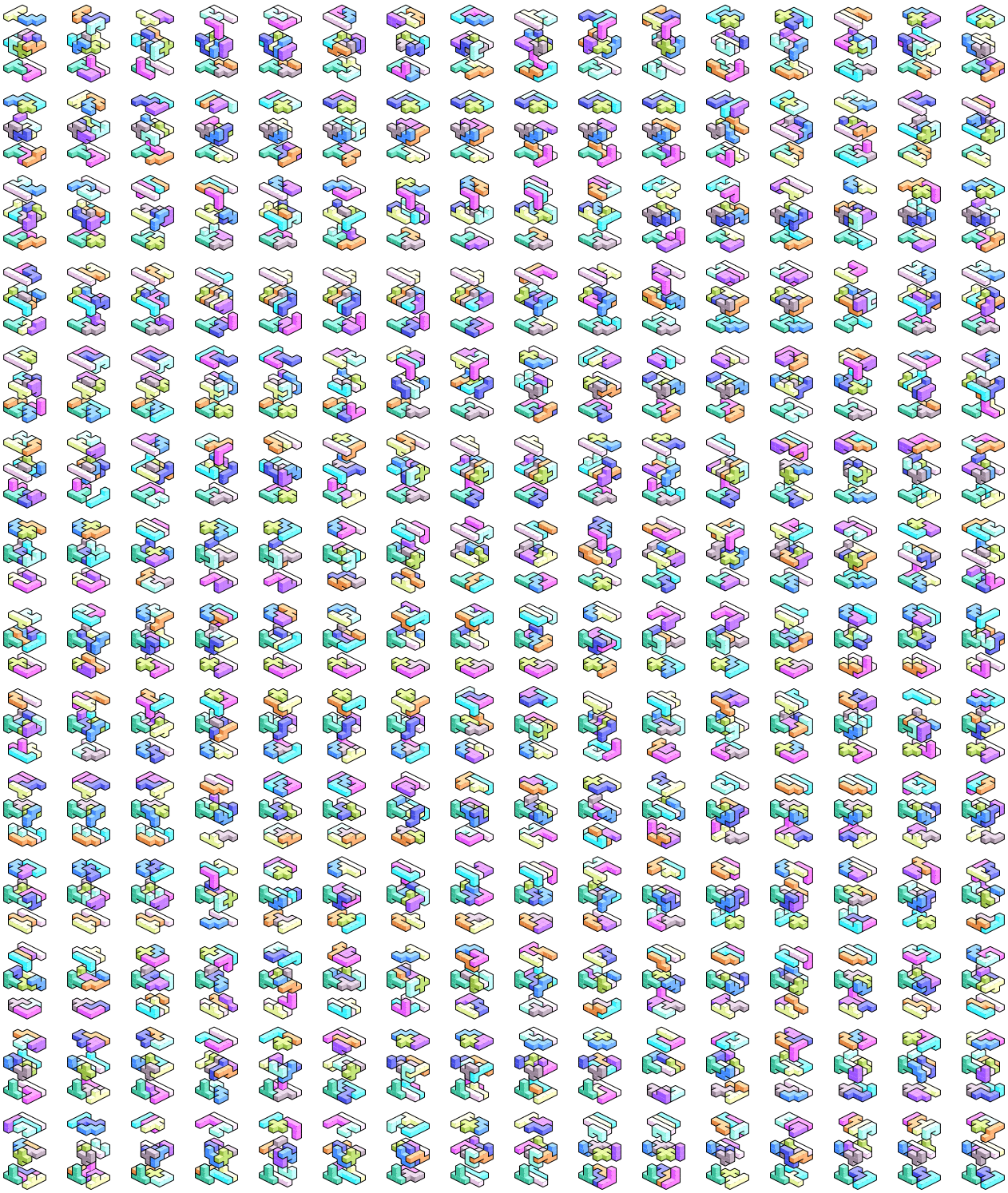


Solutions du problème $3\times4\times5$ *(6/20)*

Solutions du problème $3\times4\times5$ *(7/20)*

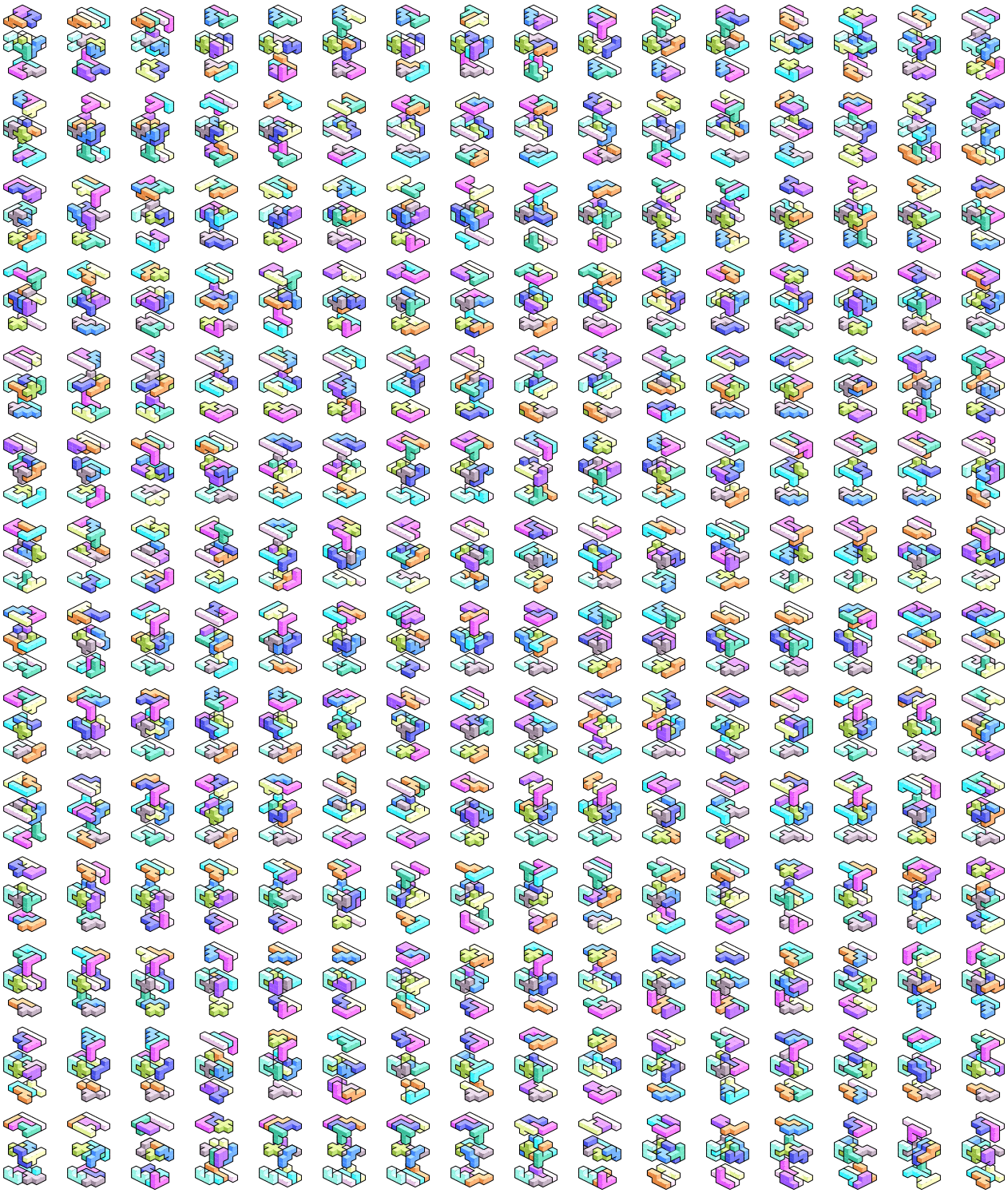


Solutions du problème $3\times4\times5$ *(8/20)*



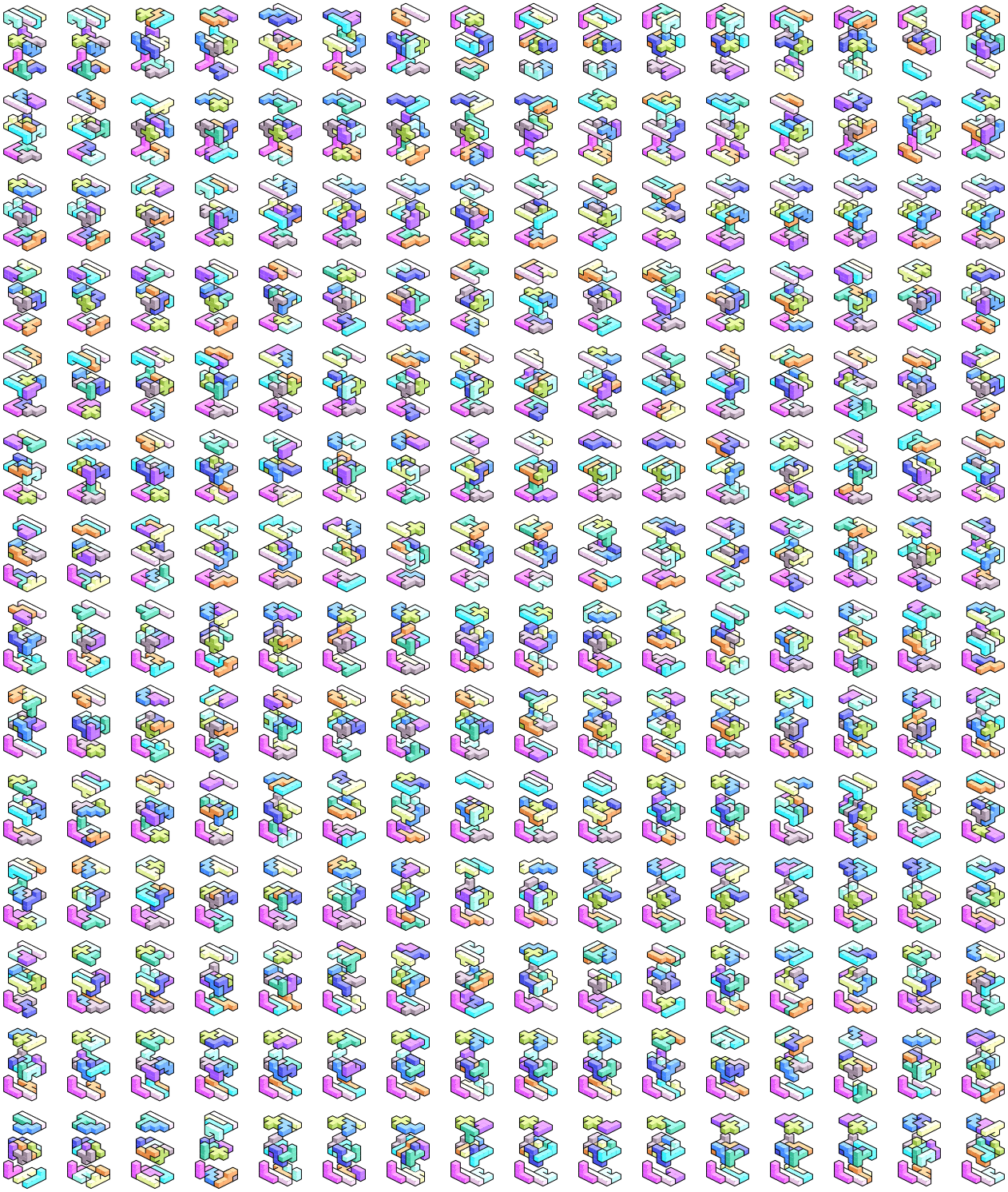
Solutions du problème $3\times4\times5$ *(9/20)*

Solutions du problème $3\times4\times5$ *(10/20)*



Solutions du problème $3\times4\times5$ *(11/20)*

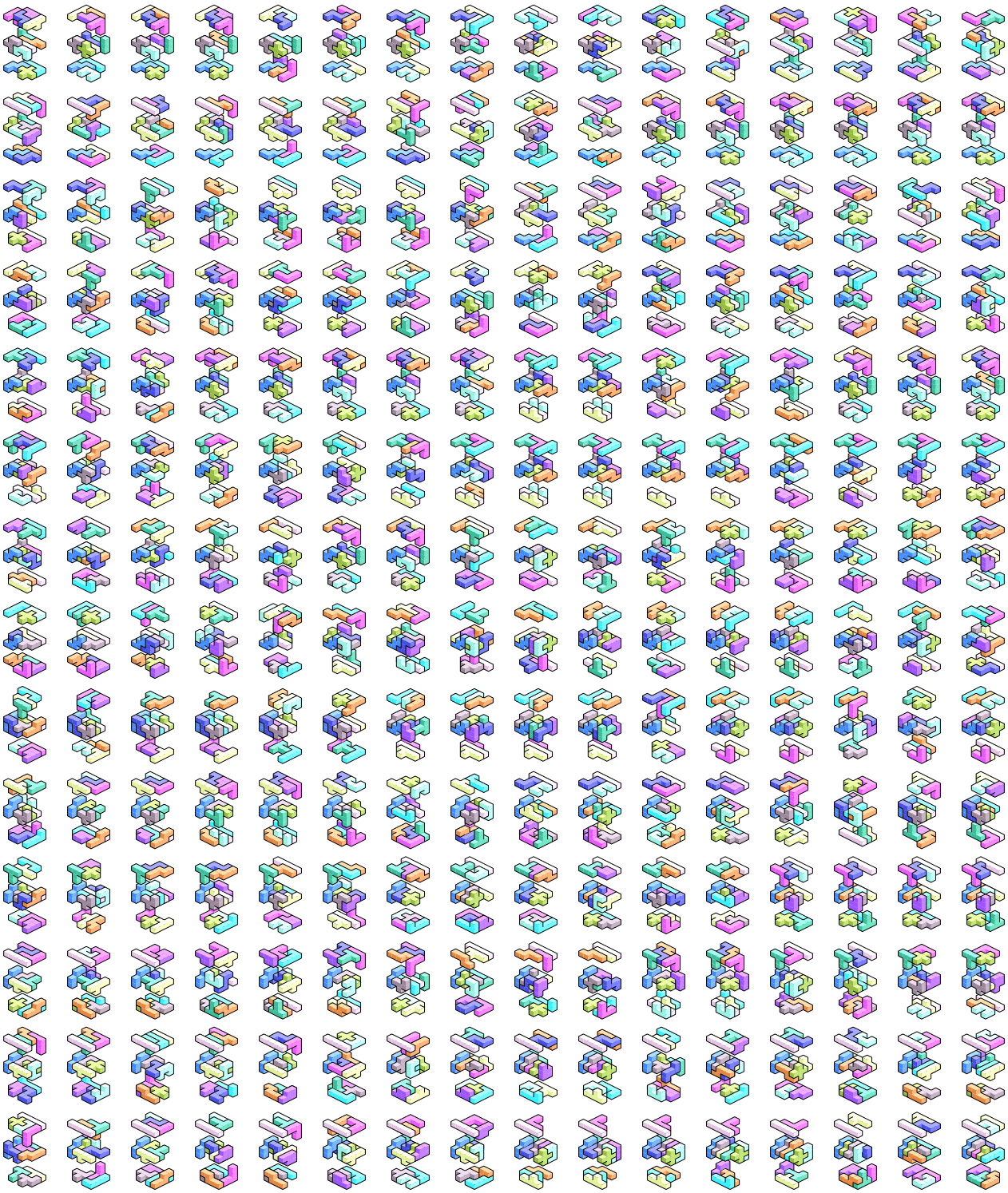
Solutions du problème $3\times4\times5$ *(12/20)*



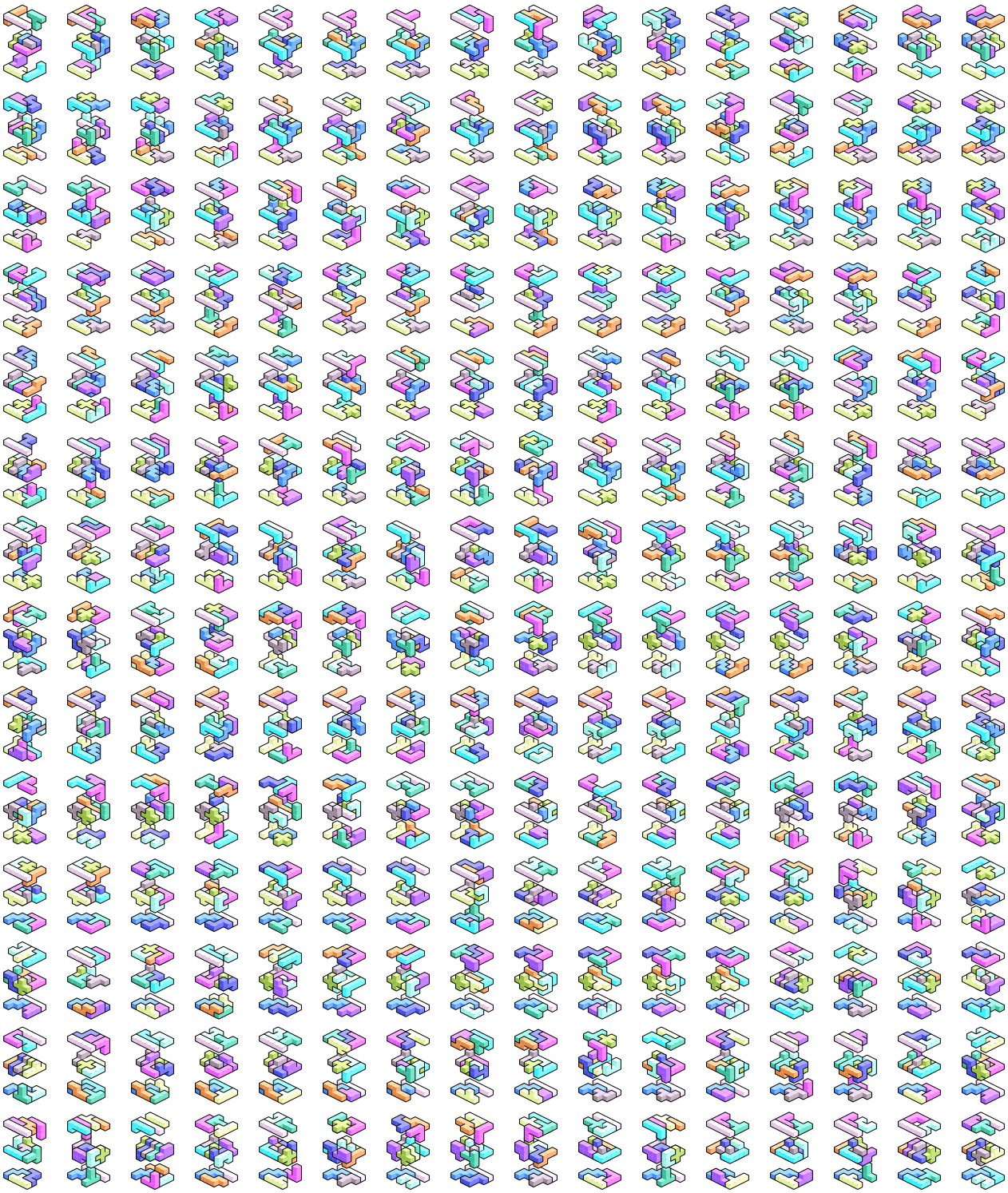
Solutions du problème $3\times4\times5$ *(13/20)*

Solutions du problème $3\times4\times5$ *(14/20)*

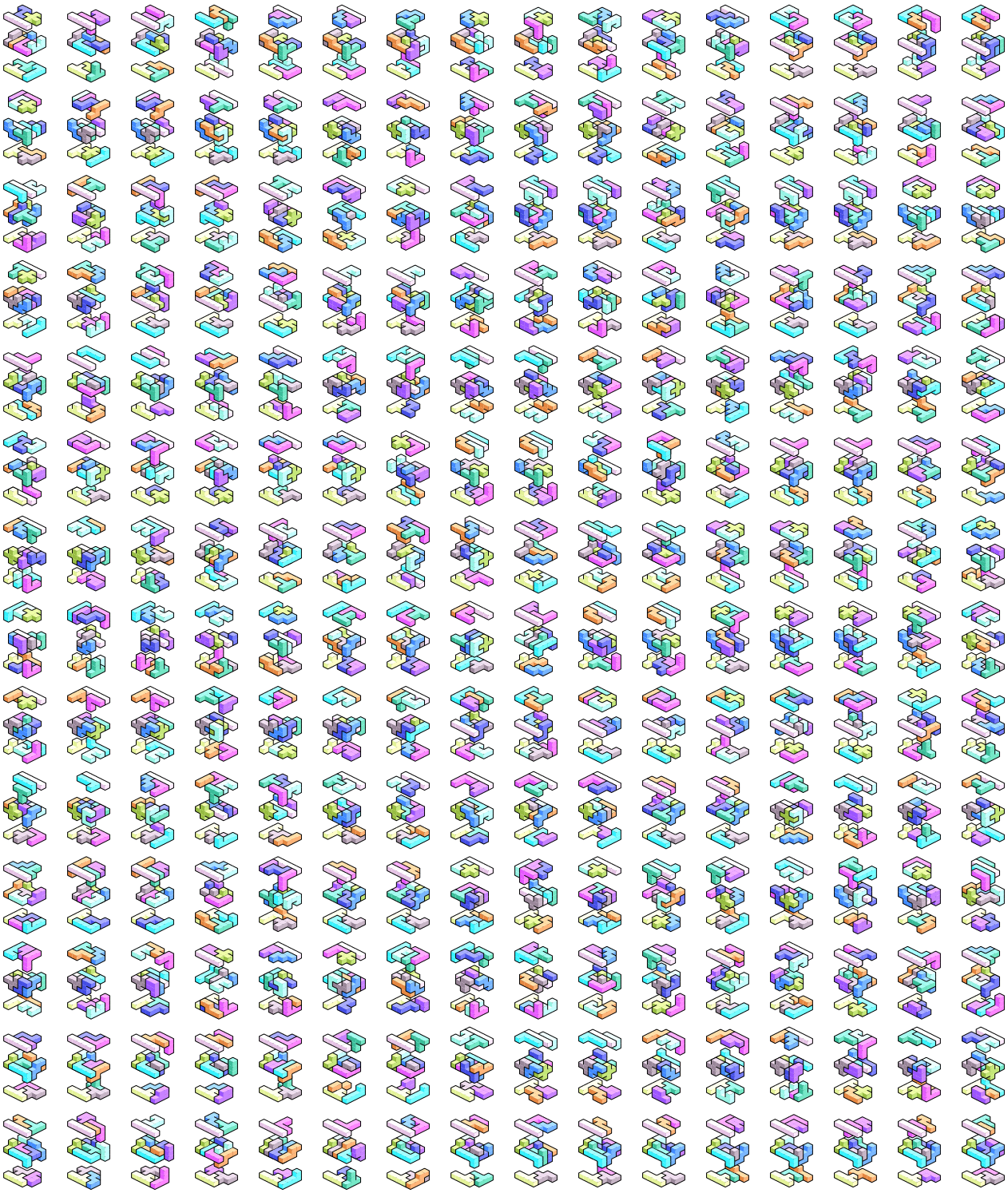
Solutions du problème $3\times4\times5$ *(15/20)*



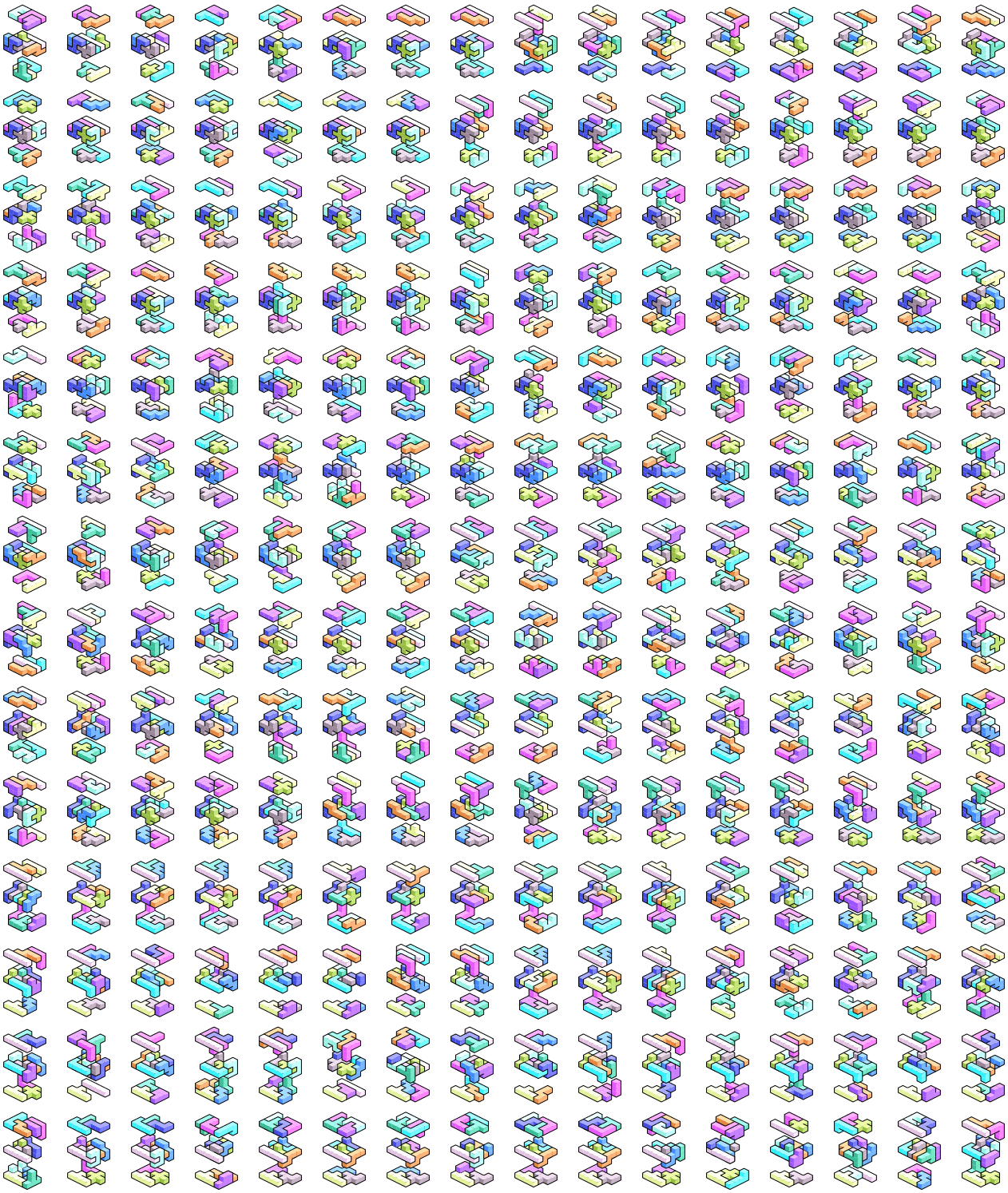
Solutions du problème $3\times4\times5$ *(16/20)*



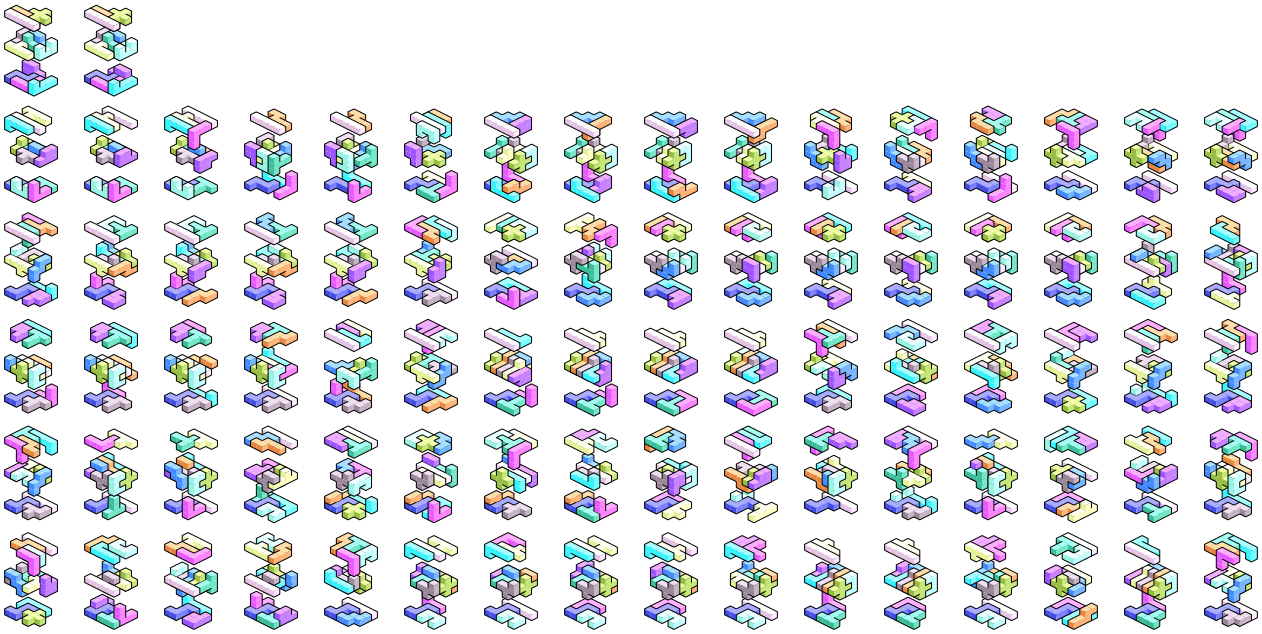
Solutions du problème $3\times4\times5$ *(17/20)*



Solutions du problème $3\times4\times5$ *(18/20)*



Solutions du problème $3\times4\times5$ *(19/20)*



Solutions du problème $3\times4\times5$ *(20/20)*