Quel codage CNF pour les contraintes de concordance de motifs d'Eternity II ?

Olivier Bailleux[1]

[1] Université Bourgogne Europe, LIB

olivier.bailleux@ube.fr

Résumé

Les puzzles de type Edge-matching — dont Eternity II (irrésolu depuis 2007) est un exemple emblématique — sont l'archétype du problème combinatoire difficile à résoudre. Ce problème a été abordé avec le parti pris de se limiter à des modélisations simples, implémentables ex nihilo en moins de 100 d'heures d'ingénierie, en utilisant la technologie SAT. Quatre encodages CNF ont été essayés. Le plus efficace s'est avéré être celui qui produit les plus petites formules, bien que ce soit le seul qui ne permette pas à la propagation unitaire de restaurer la cohérence de domaine sur les contraintes de concordance de motifs.

Mots-clés

Concordance de motifs, encodage CNF, SAT, CSP

Abstract

Edge-matching puzzles — of which Eternity II (unsolved since 2007) is an iconic example — are the archetype of the notoriously difficult combinatorial problem. This problem was tackled with the aim of restricting the models to simple ones, implementable from scratch in less than 100 engineering hours, using SAT technology. Four CNF encodings were attempted. The most effective turned out to be the one that produced the smallest formulas, although it is the only one that does not allow unit propagation to restore domain consistency on the pattern-matching constraints.

Keywords

Edge Matching Problems, CNF encoding, SAT, CSP

1 Introduction

Le problème "Edge Matching Puzzle", popularisé par la commercialisation du jeu Eternity II en 2007, constitue un terrain idéal pour challenger les techniques les plus efficaces de résolution de problèmes combinatoires. Les instances basées sur des grilles de dimensions supérieures à 10×10 sont de véritables challenges pour les solveurs SAT les plus performants, sans compter que le puzzle original de 2007, de dimension 16 par 16, est encore irrésolu aujourd'hui.

Quatre encodages CNF ont été essayés. Ils se différencient par la modélisation des contraintes de concordance de motifs, la manière de les encoder en clauses propositionnelles, et les capacités déductives de la propagation unitaire appliquée aux formules produites. Il s'avère que le codage le plus performant est celui qui propage le moins, mais qui produit les plus petites formules, ce qui confirme qu'il est hasardeux de choisir à priori un encodage CNF de contraintes énumératives sur le seul critère de sa capacité à permettre à la propagation unitaire de restaurer la cohérence de domaine, que nous appellerons *DC-compliance* dans la suite de l'article. Une telle observation a déjà été faite, notamment dans [16] et [15], mais entre des représentations respectivement linéaires et logarithmiques des valeurs des variables, alors que les encodages comparés dans cet article sont tous basés sur des représentations linéaires.

Le puzzle Eternity II est constitué d'une grille carrée de 16x16 et 256 pièces carrées ayant des motifs sur chacun de leurs 4 côtés. Un motif spécifique, de couleur grise, indique que le côté concerné doit être situé en bordure de la grille. Quatre pièces de coin possèdent chacune deux cotés gris. Cinquante-six pièces de bords possèdent chacune un côté gris. Les 196 autres pièces, dites centrales, ne possèdent aucun coté gris. Le but du jeu est de choisir, pour chaque pièce, une position et une orientation telles que chaque bord gris se situe en bordure de la grille (on dira que les contraintes de bords sont respectées) et que chaque paire de pièces adjacentes aient les mêmes motifs sur leurs côtés contigus (on dira que les contraintes de concordance de motifs sont respectées). Deux millions de dollars étaient promis à la première personne trouvant une solution avant le 31 décembre 2008. Cette date limite a été repoussée au 31 décembre 2009, puis 2010, mais aucune solution n'a encore été exhibée au moment où je rédige cet article, en février 2025.

Le puzzle Eternity II est une instance d'un problème NP-complet [7], qui a inspiré plusieurs travaux dans le milieu de la recherche en informatique et au-delà, visant essentiellement à maximiser le nombre de concordances de motifs de l'instance originale et des variantes dont les plus connues, de dimensions 10×10, 12×12, 14×14, 16×16 ont été publiées à l'occasion du "Meta 2010 Eternity II contest" [9]. Ces instances sont challengées comme des problèmes d'optimisation, notamment dans [13, 14, 12].

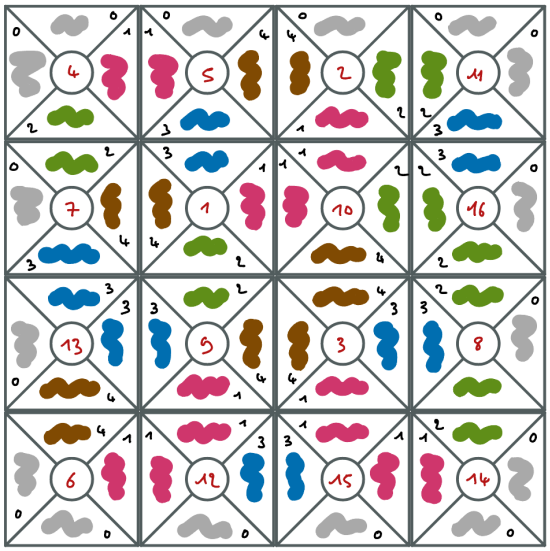

FIGURE 1 – Une instance résolue de Edge-Matching problem.

2 Résolution avec un solveur SAT

Les quatre encodages CNF ayant été essayés pour la résolution du problème à l'aide d'un solveur SAT reposent sur quatre modélisations de plus haut niveau sous forme de réseaux de contraintes à domaines finis. Dans cette section, ces quatre modèles sont d'abord présentés, puis leur traduction en clauses propositionnelles sont ensuite explicitées. Enfin, les capacités de déduction de la propagation unitaire appliquée à ces encodages sont comparées.

On notera n le nombre de lignes et de colonnes de la grille, dont chaque case sera identifiée par ses coordonnées (i, j) avec $i, j \in \{0, \ldots, n-1\}$. On notera m le nombre de motifs possibles pour les côtés des pièces, sans comptabiliser le motif de bordure de grille. Les motifs seront identifiés par les entiers 0 à m, 0 désignant le motif de bordure. Les pièces sont numérotées de 0 à $n^2 - 1$.

2.1 Modélisation à l'aide de contraintes

Les noms des quatre modèles présentés indiquent la nature des contraintes de concordance qui imposent que les motifs des côtés en contact de deux pièces situées dans des cases contiguës soient identiques. FULL exprime que chaque situation d'adjacence est représentée par une seule contrainte énumérative, HALF exprime que chaque contrainte de concordance est décomposée en deux demi-contraintes connectées par des variables représentant le motif des deux bords en contact. Le chiffre qui suit indique l'arité des contraintes énumératives.

2.1.1 Modèle FULL-4

À chaque case de la grille de coordonnées (i, j) sont associées deux variables :

— Une variable $P_{i,j}$ de domaine $\{0, \ldots, n^2 - 1\}$, dont la valeur indique la pièce présente dans la case.

— Une variable $O_{i,j}$ de domaine $\{0, \ldots, 3\}$, dont la valeur indique l'orientation de la pièce présente dans la case.

Les contraintes sont les suivantes :

1. Chaque pièce occupe exactement une case : alldiff$(P_{i,j})$.
2. À chaque paire (i, j), (k, l) de cases contiguës, verticalement ou horizontalement, on associe une contrainte de concordance reliant les variables $P_{i,j}$, $O_{i,j}$, $P_{k,l}$ et $O_{k,l}$. Cette contrainte est définie en extension. Elle exprime les combinaisons (pièce, orientation) qui assignent le même motif aux côtés en contact.

2.1.2 Modèle FULL-2

Ce modèle est utilisé dans [1]. On reprend le principe de FULL-4, mais en fusionnant les variables indiquant la pièce située dans chaque case et son orientation, de manière à rendre binaires les contraintes de concordance. À chaque case (i, j) est donc associée une seule variable $Q_{i,j}$ de domaine $0 \ldots 4n^2 - 1$ dont chaque valeur représente une pièce et une des orientations possibles de cette pièce.

Les contraintes sont les suivantes :

1. Chaque pièce occupe exactement une case.
2. À chaque paire (i, j), (k, l) de cases contiguës, verticalement ou horizontalement, on associe une contrainte de concordance reliant les variables $Q_{i,j}$ et $Q_{k,l}$. Cette contrainte est définie en extension. Elle exprime les combinaisons (pièce, orientation) qui assignent le même motif aux côtés en contact.

2.1.3 Modèle HALF-3

Pour chaque case (i, j), on reprend les variables $P_{i,j}$, $O_{i,j}$ et la contrainte alldiff$(P_{i,j})$ du modèle FULL-4 et on introduit des variables supplémentaires à domaines $0 \ldots m$ représentées par les symboles $U_{i,j}$ (pour up), $R_{i,j}$ (pour right), $D_{i,j}$ (pour down) et $L_{i,j}$ (pour left). Les symboles associés à deux bords en contact représentent une même variable. Par exemple, $R_{0,0}$ et $L_{0,1}$ représentent une même variable.

Pour chaque case (i, j), 4 contraintes d'arité 3 définissent le motif de chaque côté :

1. Une contrainte reliant $P_{i,j}$, $O_{i,j}$ et $U_{i,j}$.
2. Une contrainte reliant $P_{i,j}$, $O_{i,j}$ et $R_{i,j}$.
3. Une contrainte reliant $P_{i,j}$, $O_{i,j}$ et $D_{i,j}$.
4. Une contrainte reliant $P_{i,j}$, $O_{i,j}$ et $L_{i,j}$.

Ces contraintes sont définies en extension.

Des contraintes supplémentaires restreignent les valeurs et orientations des pièces situées dans les cases de bords et de coins, ainsi que les valeurs des pièces situées dans les cases centrales.

Ce modèle est très proche de celui utilisé dans [6], dans lequel des variables distinctes sont utilisées pour représenter les motifs des 4 bords de chaque pièce, avec ajout de contraintes d'égalités entre bords contigus.

2.1.4 Modèle HALF-2

On reprend le principe HALF-3, mais en fusionnant les variables représentant la pièce située dans une case et son orientation selon le principe du modèle FULL-2. La figure 2 donne une vue synthétique des contraintes représentant les règles de concordance de motifs pour chacun des 4 modèles.

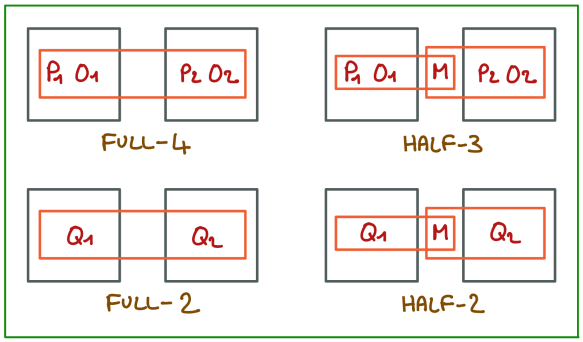

FIGURE 2 – Vue synthétique des contraintes exprimant les règles de concordance des motifs entre pièces adjacentes dans les 4 modèles proposés.

2.2 Puissance déductive de la propagation

2.2.1 Propagation de FULL-4 versus HALF-3

Avant de passer aux encodages CNF ayant été utilisés pour ces quatre modèles, il est intéressant de comparer les capacités de déduction de la restauration de la cohérence de domaines sur les contraintes utilisées, aussi appelée propagation.

Le premier point important est que la décomposition de chaque contrainte d'adjacence en deux demi-contraintes, avec introduction de variables auxiliaires, n'affecte pas les déductions réalisables par propagation. En effet, les mêmes informations peuvent être déduites par propagation des contraintes de concordance de FULL-4 et HALF-3.

Rappelons que chaque contrainte de concordance de motifs entre cases contiguës de FULL-4 est décomposée dans HALF-3 en deux contraintes reliées par une variable qui représente le motif commun aux deux côtés en contact des pièces concernées.

Soient deux cases contiguës C_1 et C_2. Soient q la contrainte de concordance sur P_1, O_1, P_2, O_2 associée à la frontière commune entre ces deux cases. Soient q_1 sur P_1, O_1, M et q_2 sur P_2, O_2, M, les contraintes représentant la décomposition de q.

Soit une valeur $P_1 = p$ n'ayant pas de support pour q. Soit d l'ensemble des motifs appartenant à des supports de $P_1 = p$ pour q_1. Le fait que $P_1 = p$ n'ait pas de support implique qu'aucune des assignations autorisées de P_2 et O_2 ne comporte un motif de d sur le côté de C_2 en contact avec C_1. En conséquence, la propagation de q_2 retire ces motifs du domaine de M, ce qui permet à la propagation de q_1 de retirer p du domaine de P_1. Symétriquement, toute valeur $P_2 = v$ n'ayant pas de support pour q est filtrée par la propagation de q_1 et q_2.

Un raisonnement similaire, appliqué à une valeur $O_1 = r$ (respectivement $O_2 = r$) n'ayant pas de support pour q, montre que les propagations de q_1 et q_2 retirent r du domaine de O_1 (respectivement O_2).

2.2.2 Impact de la fusion des variables dans FULL-2 et HALF-2

Les deux variables de HALF-3 représentant respectivement la pièce située dans une case et son orientation sont fusionnées en une seule dans HALF-2, dans le but de représenter les règles de concordance par des contraintes binaires. Cette fusion permet de réaliser toutes les propagations supportées par HALF-3, *et plus encore*, car certaines déductions pouvant être inférées par restauration de la cohérence de domaine sur les demi contraintes de concordance binaires ne sont *pas* représentables par des sous-domaines des variables utilisées par les demi contraintes de concordance d'arité 3 du modèle HALF-3.

Plus précisément, certaines assignations partielles des variables fusionnées de HALF-2 permettent de représenter des situations qui ne sont pas représentables par des assignations partielles de variables de HALF-3. Par exemple, pour un puzzle de 4×4, l'assignation $Q_{2,2} \in \{19, 33\}$ de HALF-2, qui exprime que la case $(2, 2)$ peut être occupée uniquement par la pièce 3 en orientation 1 ou par la pièce 1 en orientation 2, ne peut être représentée par aucune assignation partielle de HALF-3. L'assignation partielle de HALF-3 la plus restrictive couvrant ces deux possibilités est $P_{2,2} \in \{1, 3\}$, $O_{2,2} \in \{1, 2\}$, qui donne moins d'information, puisqu'elle couvre quatre configurations (pièce 1 en orientations 1 et 2 et pièce 3 en orientations 1 et 2) au lieu de deux.

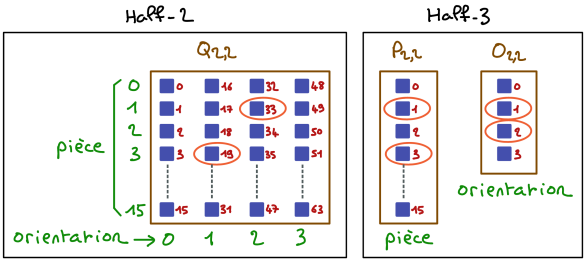

FIGURE 3 – Une assignation partielle de la variable de fusion $Q_{2,2}$ du modèle HALF-2 et l'assignation partielle la plus restrictive des variables correspondantes du modèle HALF-3 qui couvre les mêmes configurations (pièce, orientation).

Le même raisonnement s'applique à la fusion des variables représentant les pièces et orientations permettant de passer du modèle FULL-4 au modèle FULL-2.

En résumé, le modèle HALF-2 permet de réaliser plus de propagations que le modèle HALF-3 parce qu'il exploite des représentations plus expressives des combinaison (pièce , orientation). Pour la même raison, le modèle FULL-2 permet de réaliser plus de propagations que le modèle FULL-4. Les modèles HALF-3 et FULL-4 permettent de réaliser les mêmes propagations sur les variables repré-

sentant les pièces et leurs orientation, mais HALF-3 permet de réaliser en plus des propagations prenant en compte les variables de motifs, qui n'existent pas dans FULL-4. La même remarque s'applique à HALF-2 et FULL-2. Par ordre décroissant de puissance de propagation, on a donc HALF-2, puis HALF-3 et FULL-2, et enfin FULL-4. HALF-3 et FULL-2 sont difficiles à comparer car HALF-3 tire parti des variables de motifs et peut permettre à un solveur de faire des hypothèses sur ces variables, alors que FULL-2 n'a pas cet avantage, mais permet des représentations plus fines des combinaisons (pièce, orientation).

2.3 Encodage des contraintes en CNF

2.3.1 Contraintes d'unicité

Chaque variable initiale V de domaine $a \dots b$ est traduite par n variables propositionnelles qui seront notées $V = a, \dots, V = b$. Un ensemble de clauses exprime l'unicité de la valeur de V en utilisant l'encodage proposé dans [11], de taille linéaire, avec un nombre linéaire de variables auxiliaires. Une telle contrainte peut être encodée sans variables auxiliaires avec un nombre quadratique de clauses, mais des essais réalisés avec les instances de problèmes utilisés pour comparer les différents encodages ont montré que cette option est moins efficace.

La contrainte alldiff des modèles FULL-4 et HALF-3 est spécifiquement une contrainte de permutation. Elle est décomposée en deux contraintes d'unicité : une qui exprime que chaque case contient exactement une pièce, et une qui exprime que chaque pièce occupe exactement une case. La première contrainte est déjà nécessaire pour encoder l'unicité de la valeur assignée à chaque variable. La deuxième est encodée de la même manière, avec l'approche présentée dans [11]. Une telle décomposition ne permet pas à la propagation unitaire du solveur SAT de restaurer la cohérence de domaine du alldiff, mais aucun encodage ne le permet avec un nombre polynomial de clauses [4].

Dans les modèles FULL-2 et HALF-2, la nécessité que chaque pièce $p \in 0 \,..\, n^2-1$ occupe exactement une case est encodée par une contrainte d'unicité entre les $4n^2$ variables propositionnelles $Q_{i,j} = rn^2+p$ pour $i, j \in \{0, \dots, n-1\}$ et $r \in \{0, \dots, 3\}$, qui exprime qu'une case et une seule contient la pièce p dans une et une seule orientation r.

2.3.2 Contraintes énumératives

Les contraintes de concordance de motifs entre pièces adjacentes se présentent sous forme énumérative dans les quatre modèles proposés, mais des approches différentes ont été adoptées pour les encoder en CNF.

Les contraintes d'adjacence du modèle FULL-4 ont été encodées avec la méthode introduite dans [2], qui est une généralisation du « support encoding » introduit dans [8] dans le cadre restreint des contraintes binaires. Cette approche encode en CNF chaque propagation possible de la contrainte source en introduisant une variable auxiliaire pour chaque support. Le nombre de clauses produit est en $\Theta(km)$, avec k l'arité de la contrainte source et m son nombre de solutions.

Les contraintes d'adjacence binaires des modèles FULL-2 et HALF-2 ont été encodées avec la technique « support encoding » introduite dans [8], qui permet à la propagation unitaire de restaurer la cohérence de domaine de la contrainte source sans ajout de variables auxiliaires, avec un nombre de clauses proportionnel au nombre de solutions de la contrainte source.

Dans le cas du modèle HALF-3, dans lequel les règles d'adjacence sont décomposées en deux contraintes ternaires partageant une même variable auxiliaire, le choix d'un encodage plus concis a été fait : pour chaque case (i, j), chaque pièce p et chaque orientation r, et chaque variable C associée à un côté de la case, la clause

$$\neg(P_{i,j} = p) \vee \neg(O_{i,j} = r) \vee (C = m)$$

est produite, où m est le motif placé sur le côté désigné par C lorsque la pièce p a l'orientation r.

En complément, des clauses sont ajoutées pour encoder les propriétés suivantes (qui sont prises en compte implicitement dans les autres modèles) :

1. Une case de coin ne peut contenir qu'une pièce de coin, dans une seule orientation possible.
2. Une case de bord ne peut contenir qu'une pièce de bord, avec une seule orientation possible.
3. Une case de centre ne peut contenir qu'une pièce de centre.

Cet encodage ne permet pas à la propagation unitaire de restaurer complètement la cohérence de domaine des contraintes de concordance de motifs. Il ne permet même pas à la propagation unitaire de détecter certaines incohérences. Par exemple, supposons que le domaine de la variable $P_{1,1}$ contient uniquement des pièces dont les côtés ne sont ni rouge, ni vert, que le domaine de la variable $P_{1,2}$ contient uniquement des motifs dont les côtés sont rouges ou verts, et que le domaine de la variable $R_{1,1}$, aussi nommée $L_{1,2}$, qui représente le motif de l'arête commune des cases $(1, 1)$ et $(1, 2)$, contienne toutes ses valeurs initiales. Une restauration de la cohérence de domaines des contraintes énumératives sous-jacentes aurait pour effet de propager $R_{1,1} \neq$ rouge et $R_{1,1} \neq$ vert, puis de retirer toutes les valeurs du domaine de $P_{1,2}$. Mais les clauses de l'encodage HALF-3 ne permettent pas ces propagations.

2.4 Comparaison synthétique des quatre encodages CNF

Voici un bref résumé des points faibles et des points forts des 4 encodages CNF.

Les encodages FULL-2 et HALF-2 sont ceux qui permettent à la propagation unitaire de réaliser le plus de déductions, d'une part grâce à une représentation des combinaisons (pièces, orientations) qui maximise la quantité d'information représentable par les assignations partielles, et d'autre part en réalisant toutes les inférences relevant de la restauration de la cohérence de domaine. Le modèle HALF-2 a l'avantage de produire des formules de plus petite taille.

Le modèle FULL-4 utilise une représentation moins expressive que FULL-2 et HALF-2. Son encodage CNF permet à

la propagation unitaire de restaurer complètement la cohérence de domaine des contraintes de concordance de motifs, mais au prix de l'ajout de variables auxiliaires qui ne sont pas nécessaires dans les représentations CNF de FULL-2 et HALF-2. De surcroît, c'est celui des quatre encodages implémentés qui produit les plus grosses formules. Il ne présente à priori aucun avantage par rapport à FULL-2-CNF et HALF-2-CNF.
La représentation adoptée pour encoder HALF-3 en CNF est celle qui propage le moins. C'est la conséquence du choix qui a été fait de privilégier la compacité des formules produites au détriment des déductions réalisables par propagation unitaire. "Sur le papier", le seul intérêt de l'encodage CNF choisi pour le modèle HALF-3 est donc de produire de plus petites formules que tous les autres encodages, mais sans toutefois changer l'ordre de grandeur de leurs tailles. Selon des critères basés sur les capacités de propagation, on pourrait considérer un tel choix comme une erreur de conception. Et pourtant, les résultats présentés dans la section suivante montrent que c'est l'encodage le plus efficace !

3 Résultats

Les différentes approches de résolution ont été appliquées à des lots de 100 instances de puzzle de tailles 5×5 à 10×10 générées aléatoirement, avec une limite de temps de 1000 secondes pour chaque instance, sur un ordinateur MacBook M1 Pro. Un vérificateur a été utilisé pour valider toutes les solutions trouvées.
La production d'un puzzle aléatoire de $n \times n$ avec m motifs différents se fait en trois étapes :

1. Production d'une solution d'un puzzle torique (donc sans bordure, le bord bas d'une pièce située sur la dernière ligne est en contact avec le bord haut de la pièce située sur la première ligne dans la même colonne, le bord droit d'une pièce située sur la dernière colonne est en contact avec le bord gauche de la pièce de la même ligne située sur la première colonne) avec équilibrage des nombres d'occurrences de motifs identiques, au sens où chaque motif apparaît $2n^2/m$ fois ou $1 + 2n^2/m$ fois sur les $2n^2$ frontières entre pièces contiguës.
2. Assignation du motif 0 à chacun des $4n$ bords de la grille.
3. Mélange des pièces ainsi obtenues.

Les résultats obtenus avec les quatre encodages CNF et la version 4.0.1 du solveur kissat [5] (en utilisant les paramètres par défaut) sont présentés dans les tableaux 4, 5, 6, 7 du plus performant au moins performant. Les temps moyens de résolution ont été calculés uniquement sur les instances résolues.
Pour les quatre encodages, la dispersion des temps de résolution entre puzzles de même taille est très élevée. Par exemple, dans le lot de puzzles de 8×8, avec l'encodage le plus performant, on constate un facteur 100 entre les temps de résolution minimaux et maximaux. Les 6 puzzles de 9×9 résolus le plus rapidement (moins de 63 secondes) sont traités plus rapidement que les 6 puzzles de 8×8 dont la résolution prend le plus de temps (plus de 63 secondes).
La mauvaise performance du FULL-4 montre que la DC-compliance ne garantit pas l'efficacité d'un encodage CNF de contraintes énumératives, en particulier lorsqu'il introduit des variables impliquées.
L'efficacité relative de HALF-3 par rapport à HALF-2 est surprenante, car le second permet toutes les propagations possibles de chaque contrainte de concordance de motifs, alors que le premier ne permet même pas à la propagation unitaire de détecter les incohérences sur ces contraintes, qui sont par ailleurs décomposées de la même manière dans les modèles CSP sous-jacents. En outre, les formules produites par HALF-2 ne sont que deux fois plus grosses que celles produites par HALF-3.
Les performances relativement proches de FULL-2 et HALF-2 n'appellent pas de commentaire.

3.1 Conclusion et perspectives

La résolution du problème de recherche de type « Edge matching puzzle », connu sous le nom commercial « Eternity II », a été abordée avec quatre encodages CNF et un solveur SAT « Up-to-date ».
Les modèles utilisés sont directement inspirés de la spécification native du problème, reposant sur des contraintes de concordance de motifs sur les paires de cases contiguës. Bien que la littérature propose des modèles plus sophistiqués, impliquant des contraintes sur un nombre plus élevé de cases [10, 3], elles n'ont pas été exploitées ici. En effet, l'objectif de ce travail n'était pas de relever le défi particulier posé par ce problème, mais plutôt de le traiter comme un exemple type de problème combinatoire difficile se présentant sous la forme d'un réseau relativement homogène de contraintes de même nature et de même taille.
Le codage CNF qui s'est avéré le plus efficace est celui qui produit les plus petites formules, mais au prix d'une réduction drastique des capacités de déduction de la résolution unitaire, puisque les propagations des contraintes de concordance de motifs ne peuvent se faire que lorsque la pièce située dans une case et son orientation sont complètement déterminées.
Ce constat nous rappelle que la résolution des problèmes NP-complets relève encore aujourd'hui d'une démarche essentiellement expérimentale, et que nous sommes loin de comprendre parfaitement les critères d'efficacité des solveurs CDCL. Le cas du codage HALF-3 mérite une étude plus approfondie. Pourquoi est-il plus efficace qu'un codage produisant des formules de taille seulement deux fois plus grande tout en permettant plus de propagations ? Est-ce que les propagations supplémentaires sont inutiles pour la recherche d'une solution ? (Le seraient-elles pour une preuve d'incohérence ?) Ou bien les clauses apprises par le solveur permettent-elles de réaliser certaines propagations qui ne sont pas initialement implémentées dans l'encodage ?
Il semble donc utile d'étudier plus en détail le comportement des solveurs CDCL sur des formules CNF résultant

d'un encodage de contraintes, et en particulier l'hypothèse que ces solveurs puissent compenser des déficits de propagation en produisant des clauses permettant de réaliser des propagations qui ne sont pas initialement supportées.

taille	# variables	# clauses	poids	% résolu	temps moyen
5×5	2.6k	15k	0.2 Mo	100	0.03
6×6	5.2k	31k	0.5 Mo	100	0.18
7×7	9.1k	57k	1 Mo	100	1.39
8×8	15k	97k	1.7 Mo	100	20.2
9×9	24k	154k	2.7 Mo	49	380
10×10	35k	233k	4.5 Mo	0	—

FIGURE 4 – Résultats de l'encodage **HALF-3**

taille	# variables	# clauses	poids	% résolu	temps moyen
5×5	7.5k	23k	0.3 Mo	100	0.00
6×6	15k	48k	0.8 Mo	100	0.09
7×7	29k	90k	1.8 Mo	100	1.04
8×8	49k	155k	3.8 Mo	100	27.8
9×9	79k	250k	7.6 Mo	30	425
10×10	120k	383k	14 Mo	0	—

FIGURE 5 – Résultats de l'encodage **FULL-2**

taille	# variables	# clauses	poids	% résolu	temps moyen
5×5	8k	28k	0.4 Mo	100	0.00
6×6	17k	58k	0.9 Mo	100	0.03
7×7	30k	107k	1.7 Mo	100	0,54
8×8	51k	180k	3.1 Mo	100	28.6
9×9	82k	286k	5 Mo	25	530
10×10	124k	433k	7.8 Mo	0	—

FIGURE 6 – Résultats de l'encodage **HALF-2**

taille	# variables	# clauses	poids	% résolu	temps moyen
5×5	21k	64k	0.7 Mo	100	0.34
6×6	80k	230k	3.5 Mo	100	2.68
7×7	246k	736k	12 Mo	100	47.4
8×8	672k	2010k	33 Mo	37	497
9×9	1470k	4420k	75 Mo	0	—

FIGURE 7 – Résultats de l'encodage **FULL-4**

Références

[1] Carlos Ansótegui, Ramón Béjar, César Fernàndez, and Carles Mateu. How hard is a commercial puzzle : the eternity ii challenge. In *Proceedings of the 11th International Conference of the Catalan Association for Artificial Intelligence*, volume 184, 01 2008.

[2] Fahiem Bacchus. Gac via unit propagation. In Christian Bessière, editor, *Principles and Practice of Constraint Programming – CP 2007*, pages 133–147, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[3] Thierry Benoist. Fast global filtering for eternity ii. *Constraint Programming Letters*, 3 :36–49, 01 2008.

[4] Christian Bessiere, George Katsirelos, Nina Narodytska, and Toby Walsh. Circuit complexity and decompositions of global constraints. *IJCAI International Joint Conference on Artificial Intelligence*, 05 2009.

[5] Armin Biere, Tobias Faller, Katalin Fazekas, Mathias Fleury, Nils Froleyks, and Florian Pollitt. CaDiCaL, Gimsatul, IsaSAT and Kissat entering the SAT Competition 2024. In Marijn Heule, Markus Iser, Matti Järvisalo, and Martin Suda, editors, *Proc. of SAT Competition 2024 – Solver, Benchmark and Proof Checker Descriptions*, volume B-2024-1 of *Department of Computer Science Report Series B*, pages 8–10. University of Helsinki, 2024.

[6] Joffrey Cuvillier and Rémi Szymkowiak. Résolution du jeu eternity 2 avec les technologies sat, 2010.

[7] Erik Demaine and Martin Demaine. Jigsaw puzzles, edge matching, and polyomino packing : Connections and complexity. *Graphs and Combinatorics*, 23 :195–208, 06 2007.

[8] Ian Gent. Arc consistency in sat. In *Proceedings of the 15th Eureopean Conference on Artificial Intelligence, ECAI'2002*, pages 121–125, 01 2002.

[9] International conference on metaheuristics and nature inspired computing (meta), 2010.

[10] Fabio Salassa, Wim Vancroonenburg, Tony Wauters, Federico Della Croce, and Greet Vanden Berghe. Milp and max-clique based heuristics for the eternity ii puzzle, 09 2017.

[11] Carsten Sinz. Towards an optimal cnf encoding of boolean cardinality constraints. In Peter van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005*, pages 827–831, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[12] Wim Vancroonenburg, Tony Wauters, and Greet Vanden Berghe. A two phase hyper-heuristic approach for solving the eternity ii puzzle. In *Conference : 3rd International Conference on Metaheuristics and Nature Inspired Computing (META 2010)*, 10 2010.

[13] Wei-Sin Wang and Tsung-Che Chiang. Solving eternity-ii puzzles with a tabu search algorithm. In *Proceedings of International Conference on Metaheuristics and Nature Inspired Computing, 2010*, 10 2010.

[14] Tony Wauters, Wim Vancroonenburg, and Greet Vanden Berghe. A guide-and-observe hyper-heuristic approach to the eternity ii puzzle. *Journal of Mathematical Modelling and Algorithms*, 11 :217–233, 09 2012.

[15] Neng-Fa Zhou. Yet another comparison of sat encodings for the at-most-k constraint, 2020.

[16] Neng-Fa Zhou and Håkan Kjellerstrand. Optimizing sat encodings for arithmetic constraints. In J. Christopher Beck, editor, *Principles and Practice of Constraint Programming*, pages 671–686, Cham, 2017. Springer International Publishing.