Faculté des Sciences
Université de Montpellier
Lirmm

Rapport de stage
Ouvertures et finales d'Eternity II

Auteurs :
Fati Chen

Référent :
Eric Bourreau

20 août 2016

1 Introduction

Les puzzles et casses-têtes nous ont toujours passionnés, pour faire passer le temps ou pour se mettre des défis. Eternity II est un de ces jeux où le principe peux être compris par tous, mais pourtant sa résolution est extrêmement complexe. Ce genre de paradigme est à l'heure actuelle l'un des problèmes mathématiques qui régit notre monde, car la plupart des systèmes informatiques et méthodes de chiffrement reposent sur ce genre de problème (simple à faire mais pourtant trouver la solution ne l'est pas).

Eternity II n'est résoluble à l'heure actuelle qu'en testant toutes les combinaisons (bruteforce). Ce qui nous fait poser une question importante, comment, avec l'augmentation exponentielle des données et des nouvelles technologies, sommes nous réduit à utiliser une méthode aussi simple. Par extension, est-il plus efficace d'accumuler des données afin de le résoudre plutôt qu'essayer d'accélérer la résolution basique.

Dans un premier temps, nous verrons les origines du jeu, la difficulté à laquelle nous sommes confrontés et l'état de l'art des méthodes de résolutions.

Ensuite, nous présenterons la problématique, ce qui à déjà tout au long de l'année et l'approche initiale du problème.

Pour conclure, les résultats et réflexions qui peuvent en être tirés.

Par ailleurs, ce compte rendu comporte un manuel d'utilisation et un manuel technique fourni, car les application développées, ou tout du moins leur logique est destinée à être réutilisés ou améliorés.

2 Eternity II

2.1 Les origines

— [réel]

Eternity II est le fier successeur de Eternity.

La première version sortie en 1999, était composée de 159 pièces de différentes formes, cependant ces formes peuvent être décomposés en formes de trianges equilatéraux (ou leur moitié) qui devaient être placés sur un plateau octogonal.

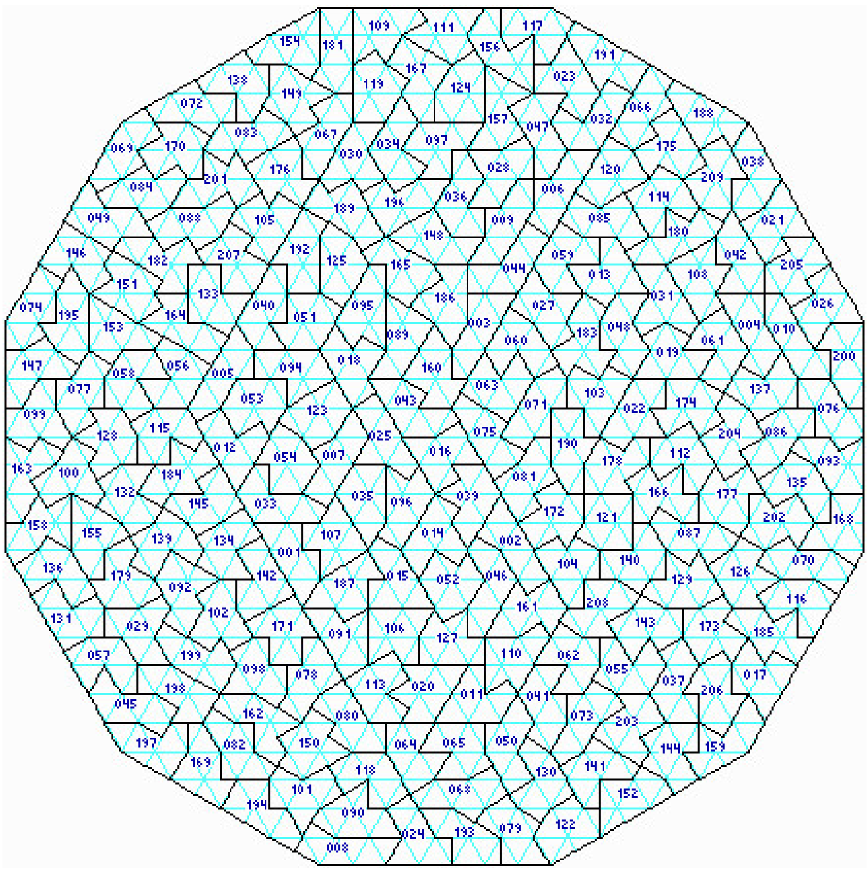

FIGURE 1 – Eternity I

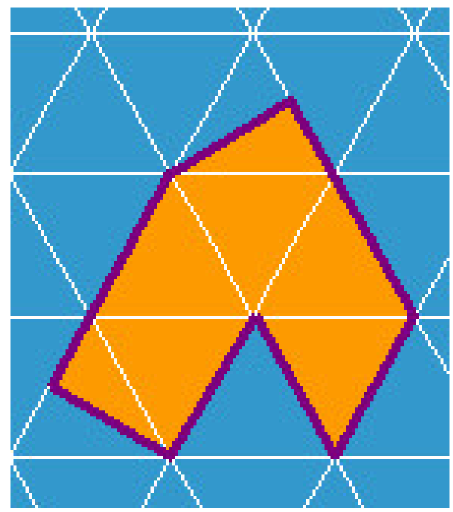

FIGURE 2 – Forme d'une pièce d'EternityI

Son point faible se trouvaient dans la disposition de ces pièces sur le plateau : il était possible de précalculer des régions, puis de les comparer entre eux afin d'en dégager une solution. De cette facon, le puzzle fut résolu en à peine un an (contrairement aux 3 ans prévus par le créateur), par deux mathématiciens, qui ont ainsi empoché la récompense s'élevant à 1000000£.

Après cet « echec », Christopher Monckton, le créateur d'Eternity, décide en 2008 de sortir une deuxième version, bien plus complexe avec à la clé 2000000$pour celui qui arriverait à la résoudre au bout de deux ans.

FIGURE 3 – La boite et les pièces d'Eternity II

C'est un puzzle de 16 par 16 qui sort sous le nom d'Eternity II. Ce puzzle est composé de 256 pièces carrés, qui ont chacune 4 faces colorés (ou un demi motif).

Ces pièces peuvent être classés en trois catégories suivant le nombre de faces grises qu'elles possèdent :

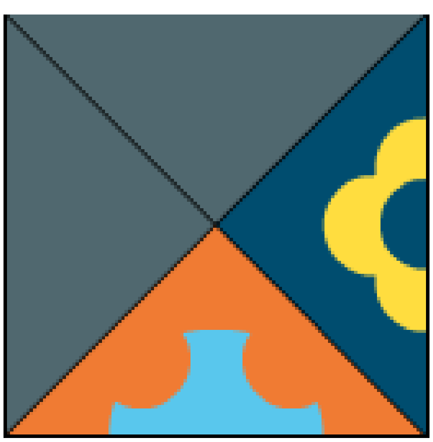

FIGURE 4 – **pièce de coin :** 2 faces grises

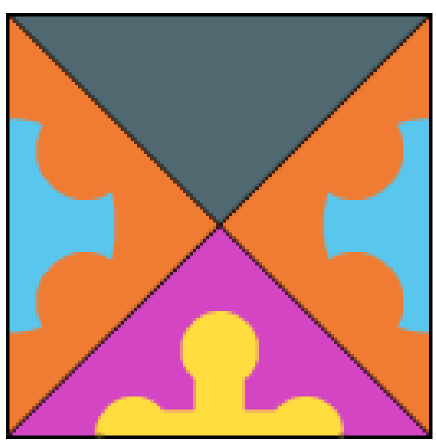

FIGURE 5 – **pièce de bord :** 1 faces grise

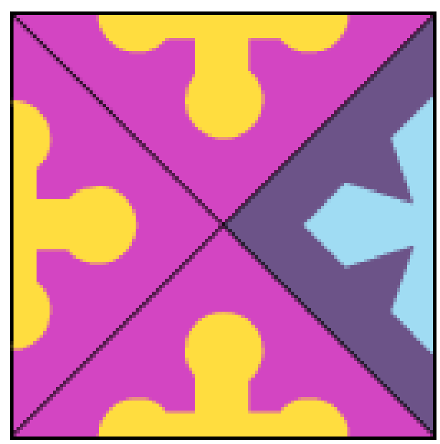

FIGURE 6 – **pièce d'intérieur :** toutes les faces de couleur

Les pièces ne possèdent pas de formes comme dans un puzzle classique. Afin de les faire correspondre l'une avec l'autre, il est nécessaire que les faces adjacentes de chaque pièce voisine soient de la même couleur, dés lors les pièces « matchent » 7.

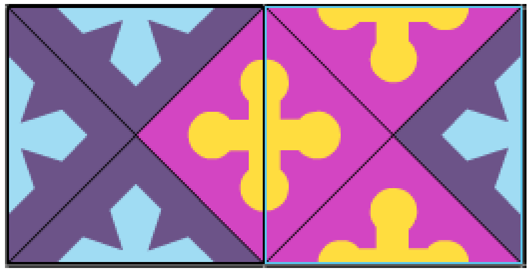

FIGURE 7 – Deux pièces correctement placés (matchés)

Par conséquent, une pièce peut quasiment être placée n'importe où sur le plateau car son placement dépend des couleurs des pièces d'à côté, de plus, les pièces n'ont pas d'orientation prédéterminés (elles peuvent être rotationnée).

Pour résumer, la plupart des pièces peuvent être posés n'importe où sur le plateau à différentes rotation car la position dépends entièrement des pièces adjacentes posés auparavant.

Enfin, comme leur nom l'indiquent, les pièces de coins sont les seules à pouvoir être posés dans les coins du plateau, c'est aussi valable pour les pièces de bord qui ne peuvent être placés que sur les bords du plateau, ces deux types de pièces ne peuvent pas être ailleurs car leurs faces grises doit « matcher » avec les bords du plateau.

— [résumé]

Eternity II est un jeu sorti en 2008 qui repose sur un principe assez simple, c'est un puzzle de 16 par 16 qu'il faut réassembler. Il est composé de 256 pièces carrés, qui ont, sur chaque arête une couleur donnée. [images tt ca tt ca]. Afin de pouvoir assembler le puzzle, il suffit placer les pièces de façon à ce que les faces adjacentes soient de même couleur. Comme un puzzle classique, il y a des pièces de coin et de bord. Ceux-ci sont reconnaissables car ils possèdent une ou deux arêtes grises. Par contre, la où ca devient complexe, c'est qu'une pièce n'as pas une place prédéterminée (comme dans un puzzle), c'est à dire qu'elle peux se situer n'importe où sur le plateau.

2.2 Le défi

— [réel]

Malgré le fait que la récompense à expiré le 31 décembre de l'année 2010, le problème et l'enthousiasme qu'a engendré Eternity II ne c'est pas calmé pour autant (enfin si…un peu). Car loin d'être juste un jeu avec une importante cagnotte il recel en son cœur des secrets d'une certaine valeur.

En effet, jusqu'à maintenant, personne n'a réussi à résoudre ce puzzle, même pas effleuré la solution, malgré l'aide de supercalculateurs et de nombreux spécialistes, que ce soient des mathématiciens ou des informaticiens.

Pourquoi ? Car derrière ce jeu anodin se cache l'un des plus grand problème du monde actuel : les problèmes NP-difficiles. Ceux-ci sont fait de telle sorte que même ne connaissant leur structure ou fonctionnement, il est pratiquement impossible d'en déduire un algorithme (moyen de résoudre) afin de trouver la solution. Ce type de problème est communément appliqué dans le chiffrement. Car le meilleur moyen de cacher une aiguille (solution) est de la cacher dans gros paquet d'aiguilles, plus le tas est gros, plus on met de temps à la [l'aiguille] trouver.

Exemple 2.1. Le nombre de combinaisons possibles pour Eternity II s'élève à 10^{545}, c'est à dire environ 10^{450} fois le nombre d'atomes dans l'univers connu (estimé à au plus 10^{80}) ! ! ! Ca fait un gros tas d'aiguilles ! !

— [résumé]

A ce jour, personne n'a réussi à résoudre ce puzzle (même grâce à l'aide de supercalculateurs) malgré les différentes stratégies mise en place. Pourquoi ? Car derrière ce jeu anodin se cache l'un des plus grand problème du monde actuel : les problèmes NP-difficiles. Ceux-ci sont fait de tel sorte que même en connaissant leur structure ou fonctionnement, il est pratiquement impossible d'en déduire un algorithme de résolution. L'une des solutions les plus fiables à ce jour est de tester tout les cas possible (qui est évidemment très important).

Exemple 2.2. Le nombre de combinaisons pour Eternity II s'élève à 10^{545}, c'est à dire environ 10^{450} fois le nombre d'atomes dans l'univers connu (estimé à au plus 10^{80}) ! ! !

2.3 Les II lois d'Eternity II

— [réel]

Pour rendre Eternity II complexe et combinatoire, il est nécessaire de respecter les deux lois d'Eternity II :

Loi 1. Chaque pièce est unique

L'unicité des pièces est indispensable pour complexifier le problème, car sinon, on peux considérer qu'une pièce peux être placée à plusieurs endroits, en fonction du nombre de « clones » qu'elle possède. Ce qui réduit grandement l'espace de recherche [de la solution].

Loi 2. La quantité de couleurs et de pièces est finement calculée

En effet, si l'on augmente le nombre de couleurs, on obtient des couplage uniques : une pièce ne peux être couplée qu'avec une autre pièce (ou dans le meilleur des cas limite le couplage des pièces). A l'inverse, si il n'y a pas assez de couleurs, on obtient des doublons, les pièces ne sont plus uniques, ce qui va à l'encontre de la première loi.

Par ailleurs, certaines couleurs sont exclusives aux pièces de coin et de bord, car ceux-ci étant liés en eux mais seulement sur le périmètre extérieur du plateau il est nécessaire d'ajouter des couleurs supplémentaires tenant compte qu'ils n'ont que 2 ou 3 faces disponibles (le reste étant des faces grises).

— [résumé]

Pour rendre ce problème combinatoire, il est nécessaire de respecter plusieurs conditions.

Chaque pièce est unique l'unicité des pièces est importante, car si une pièce est en double, cela veux dire que la pièce peux être placée à deux endroits différents (ce qui simplifie le problème)

Le ratio de nombre de couleurs par nombre de pièces est calculé [joindre graphique tt ca tt ca] : il faut qu'il y ait assez de couleur pour que chaque pièce soit unique, mais pas assez pour que l'on puisse déterminer les pièces adjacentes

Exemple 2.3. Supposons qu'il y ait trop de couleurs. Cela veux dire qu'une pièce à peu de voisins (car chaque pièce est unique, par conséquent les couleurs sont distribués uniformément à travers les pièces), si la pièce à très peu de voisins, je peux déduire des groupement de pièces assez facilement.

Donc je simplifie mon problème.

2.4 Etat de l'art

Un grand nombre de méthodes ont été mis en place afin de résoudre ce problème.

Il serait trop long de présenter et décrire les différents méthodes mise en place car il requièrent une certaine connaissance dans les domaines auxquels ils sont appliqués. Malgré tout, Les différentes approches seront notés ici à titre informatif.

Pour commencer, il existe plusieurs solveurs graphiques afin de pouvoir résoudre le puzzle manuellement ou assisté par l'ordinateur. Certains d'entre eux permettent même l'import export de la progression actuelle.

- E2_manual : `https://sourceforge.net/projects/e2manual/`(anglais)
- E2Lab : `http://eternityii.free.fr/`(anglais)

Ensuite, il existe un très grand nombre de solveurs bruteforce plus au moins rapides. Certains d'entre eux peuvent agréger plusieurs machines afin d'augmenter la puissance de calcul via le réseau.

Ce problème a été abordé de façon très varié au niveau théorique et applicatif.

Par exemple, Eternity à été adopté sous forme de graphe [5] ou de sous forme de contraintes [1].

On note aussi un procédé intéressent de résolution grâce à une approche nommée **SAT** (satisfaisabilité booléenne) s'appuyant sur la logique propositionnelle [2] [3].

— [résumé]

Nombreux sont ceux qui ont essayé de résoudre le problème... plusieurs moyens ont été mis en place, grâce au solveur SAT (solveur de satisfiabilité booléenne dont le fonctionnement est assez complexe pour ne pas être abordé ici), par une approche graphe ou encore par bruteforce.

Il est estimé que actuellement, l'approche la plus performante est la résolution par bruteforce, car c'est elle qui est la plus rapide.

3 Problématique

Le but du stage est donc de déterminer si, malgré les dires, on pourrait trouver une méthode de résolution plus efficace que la bruteforce, qui reposerait sur une grande quantité d'information pré-calculées, cette méthode sera appelée smartforce par la suite.

Ces informations pré-calculées serviront différentes causes, mais deux objectifs principaux peuvent en être explicités :

- les ouvertures
- les finales

Afin de comprendre le principe des ouvertures et des finales, il est important de pouvoir se représenter un arbre de possibilités où chaque branche de l'arbre est une combinaison spécifique.

Exemple 3.1. Prenons pour exemple l'arbre des possibilités d'un lancé de monnaie (en supposant que la monnaie ne tombe pas sur la tranche [4]). A chaque étape, deux choix s'offrent à nous :

- la pièce tombe sur pile
- la pièce tombe sur face

Supposons que l'on lance la pièce trois fois. On à donc un arbre ressemblant à ça :

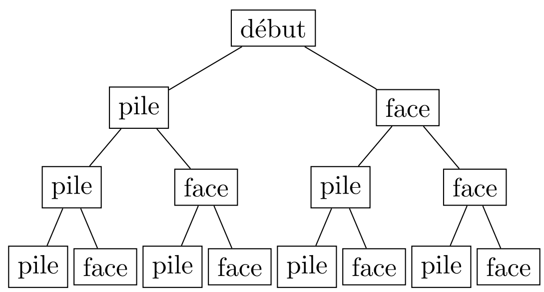

FIGURE 8 – Arbre de possibilités d'un lancer de monnaie

Grâce à cet arbre, à chaque fois qu'une pièce sera lancée jusqu'à trois fois d'affilés, la combinaison (ou chemin) figurera dans l'arbre.

Chaque nœud de l'arbre possède un sous-arbre (si il est pris comme racine), ce sous-arbre peut être vide.

Note. Pour l'arbre de résolution d'EternityII, c'est à peu près la même chose mais l'arbre est bien plus grand. Les nœuds de celui-ci ne peuvent pas être prédits (on ne connait que les nœuds suivants du chemin emprunté).

Cet arbre sera représenté par la suite sous cette forme :

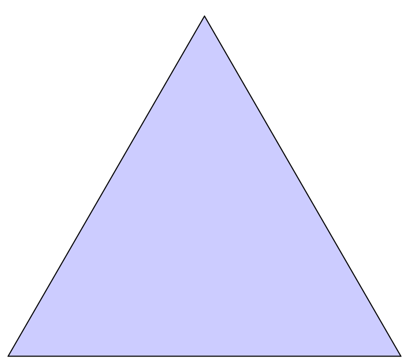

FIGURE 9 – Représentation simplifiée d'un arbre de possibilité

3.1 Ouvertures

L'idée est de pré-calculer toutes les débuts jusqu'à un certaine profondeur de l'arbre pour ensuite créer des sous-arbres pouvant être parcourus parallèlement ou pondérer les sous-arbres générés pour les classer (par taille, ...).

Exemple 3.2. Dans le cas du lancé de monnaie, je sais que le premier lancer, me donne soit pile soit face. En connaissant cela, je peux séparer l'arbre en deux sous-arbres. Cela m'évite de recalculer la première profondeur

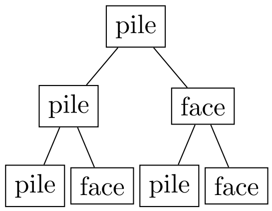

FIGURE 10 – Sous-arbre de pile

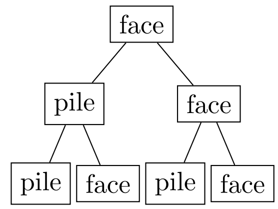

FIGURE 11 – Sous-arbre de face

Remarque. L'utilité des ouvertures dans un lancé de pièce (de monnaie) est très discutable, mais elle prends son sens lorsque

- le calcul de chaque nœud est gourmand
- la quantité de nœuds est importante

L'arbre des possibilités dans le cas des ouvertures est représenté comme ceci :

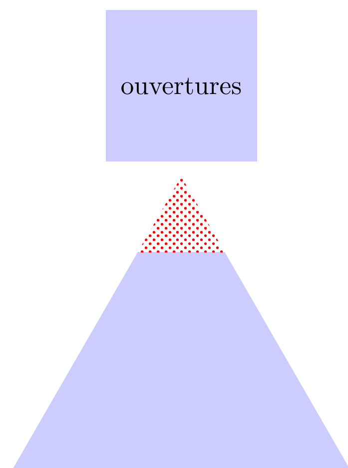

FIGURE 12 – Représentation simplifiée des ouvertures

Le triangle des possibilités est tronqué car le début a déjà été pré-calculé, c'est comme si l'on avait collé tout les sous-arbre entre eux. Les ouvertures étant stockés sous forme de données, elle sont représentés par une carré.

3.2 Finales

Les finales permettent de prédire si le chemin emprunté dans l'arbre mène à une solution. Par conséquent, toutes les finales possibles sont pré-calculés et stockés sous forme de données.

Grâce à cela, on peux connaître si le chemin choisi (ou combinaison actuelle) est possible sans avoir à finir le chemin en entier.

Les finales sont représentés comme ceci :

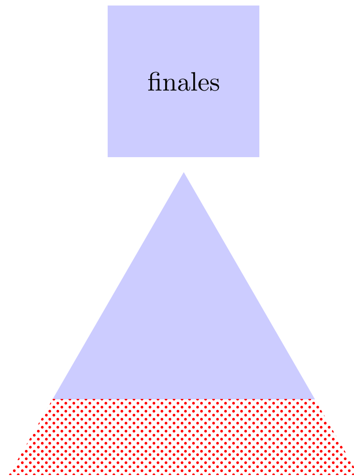

Figure 13 – Représentation simplifiée des ouvertures

3.3 Objectif

Grâce à l'aide de ces deux approches. on peux donc diminuer drastiquement la quantité de calcul nécessaire en augmentant la quantité de données stockés.

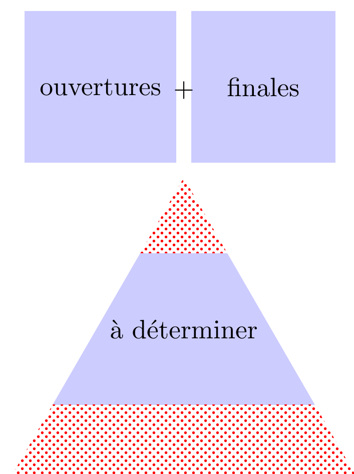

Figure 14 – Représentation simplifiée de l'objectif

3.4 Difficultés préliminaires

Afin de résoudre le problème, la principale difficulté c'est que le jeu de base est trop complexe et ne permet pas de déterminer si l'approche actuelle est adapté. Par conséquent, on utilisera des instances d'Eternity II, qui sont des plateaux de plus petite taille (4x4, 5x5 ...) qui ont les mêmes propriétés que le jeu de base.

4 Approche

Dans cette partie, nous expliquerons quels sont les différents outils et stratégies mis en place pour permettre la résolution du problème. Dans un premier temps nous avons eu besoin de mettre en place une valeur étalon, qui nous permet de savoir si les differentes méthodes de smartforce sont bonnes ou pas. Cette valeur étalon est un programme de bruteforce qui nous fournit le nombre total de noeuds, le nombre de solution, le nombre de noeuds à la premiere solution et les tous les timers correspondant.

4.1 Bruteforce

Afin de pouvoir partir sur de bonnes bases, plusieurs différents méthodes de parcours (quel chemin prendre pour résoudre mon plateau) ont été utilisé, afin de voir quel parcours est le plus performant pour la résolution brute. En sachant que le nombre de noeuds/sec est la même, l'unité de mesure est le nombre de noeuds.

Les differents types de parcours sont :

rowscan on pose les pieces en lignes horizontales sur le plateau

diagonal on pose les pieces en diagonal

spiral in on dispose les pièces en spirale en partant de l'exterieur [image]

spiral out idem que spiral in mais en partant de l'interieur vers l'exterieur

Ces différents types de parcours ont été testés sur plusieurs instances de taille variable.

4.2 Smartforce

Une fois la valeur étalon fixée, il est maintenant facile de mettre en place un autre approche du problème qui à pour principe de cumuler une grande quantité de donnée pour faire face au nombre exponentiel de possibilités.

Les différents types de données (nommés modèles) sont comme différents points de vues du problème. Ils sont plus ou moins utiles, mais la force réside dans leur union. Mais surtout, ils permettent de mettre en place le concept d'ouvertures et finales.

4.2.1 CaPi

L'approche CaPi (abbréviation de Cases/pieces) est l'approche la plus naive, elle permet de définir quelle piece peut être placée sur telle case et inversement, quelle case peux avoir telle piece.

Cette approche est l'interaction la plus basique de notre problème. C'est aussi celle-ci qui est utilisée en bruteforce.

4.2.2 BoCo

L'approche BoCo (Bordure/Couleur) est bien plus fine : si l'on connait quelle piece est sur telle case, on sait quelle couleur peux se placer sur telle bordure [de la case]. Elle permet d'implémenter un système de mis à jour bien plus performant car ne nombre de couleurs est bien plus petit que le nombre de pièces.

Exemple 4.1. Si une couleur disparait, alors tt les pièces ayant cette couleur ne peuvent plus être placés à cette case, par conséquent, les autres bords de la case ont (probablement) des couleurs qui disparaissent aussi (propagation de la disparition).

4.2.3 Corolles

Grace aux visions CaPi et BoCo, il est possible de pré-calculer des zones du plateau nommés corolles, ceux-ci contiennent tous les cas possibles dans cette zone donnée.

Le nombre de cas possible étant très important, il est nécessaire de le classer. Les corolles peuvent êtres identifiés grâce à plusieurs critères.

- taille du plateau
- l'orientation de la corolle
- La pièce (et sa rotation) à l'origine de la corolle
- La case à l'origine de la corolle
- La taille de la corolle

position et orientation des corolles

4.2.4 BoCoDiag

5 Application

6 Resultats

7 Manuel d'utilisation

8 Manuel Technique

Références

[1] Thierry BENOIST et Eric BOURREAU : La programmation par contraintes à l'attaque d'eternity ii. *In JFPC 2008-Quatrièmes Journées Francophones de Programmation par Contraintes*, pages 105–114, 2008.

[2] Joffrey CUVILLIER et Rémi SZYMKOWIAK : Résolution du jeu eternity 2 avec les technologies sat.

[3] Marijn JH HEULE : Solving edge-matching problems with satisfiability solvers. *SAT*, pages 69–82, 2009.

[4] Daniel B MURRAY et Scott W TEARE : Probability of a tossed coin landing on edge. *Physical Review E*, 48(4):2547, 1993.

[5] Ludovic PATEY et Sylvain GRAVIER : Eternity ii et variantes. 2010.