INFO-F-302 Informatique Fondamentale
Rapport du projet 2016-2017

Nathan Liccardo, Stanislas Gueniffey

21 mai 2017

1 Introduction

La projet détaillé dans ce document avait deux objectifs : la modélisation de problèmes sous formes CSP et l'implémentation de ceux-ci en Java. Les réponses fournies seront formalisées selon le modèle CSP à savoir : variables, domaines et contraintes.

2 Problèmes d'échecs

Nous supposons ici que le lecteur est familier avec le jeu d'échecs. Les problèmes posés par la suite consistent à reproduire des scénarios plausibles dans le cadre d'une partie de ce jeu, dont certains paramètres ont été modifiés et respectant des contraintes supplémentaires (qui elles ne sont pas inhérentes au jeu).

2.1 Formalismes

Afin de pouvoir obtenir le problème sous la forme d'un CSP, nous avons posé certaines variables inhérentes au problème :

n	La taille du tableau d'échecs
k_1	Le nombre de tours
k_2	Le nombre de fous
k_3	Le nombre de cavaliers

2.2 Problème d'indépendance

Le problème d'indépendance consiste à déterminer s'il est possible d'assigner à chacune des pièce une position distincte sur l'échiquier de sorte qu'aucune pièce ne menace une autre pièce.

> Exprimer une instance quelconque du problème d'indépendance par un CSP équivalent, en exposant de manière claire les variables, leurs domaines respectifs et les contraintes à respecter.

2.2.1 Variables

Nous avons défini dans un premier temps les variables représentant respectivement les tours, les fous et les cavaliers :

$$X = \{x_t |\ t \in K1\} \cup \{x_f |\ f \in K2\} \cup \{x_c |\ c \in K3\}$$

Par facilité de notation, nous avons également définit les ensembles suivants qui sont directement liés aux places et pièces du jeu :

$$N = \{1, \ldots, n\}$$
$$K_1 = \{t_1, \ldots, t_{k1}\}$$
$$K_2 = \{f_1, \ldots, f_{k2}\}$$

$$K_3 = \{c_1, \ldots, c_{k3}\}$$
$$K = K_1 \cup K_2 \cup K_3$$
$$K_4 = \{v_1, \ldots, v_{(n*n)-(k_1+k_2+k_3)}\}$$

2.2.2 Domaines

Les pièces seront représentées sous la forme de coordonnées X et Y. Ce sont ces deux valeurs qui formeront le domaine pour les trois types de pièces :

$$D_t = \{(i,j)|1 \leq i \leq n \wedge 1 \leq j \leq n\}$$
$$D_f = \{(i,j)|1 \leq i \leq n \wedge 1 \leq j \leq n\}$$
$$D_c = \{(i,j)|1 \leq i \leq n \wedge 1 \leq j \leq n\}$$

2.2.3 Contraintes

Les contraintes s'appliquants sur les variables de ce problème doivent prendre en compte les deux demandes suivantes :

Une case est occupée par maximum une pièce.

Aucune pièce ne peut en menacer une autre.

Notre ensemble de contraintes final sera l'union entre les contraintes de positions et les contraintes de domination :

$$C = C_p \cap C_d$$

Définissons dans un premier temps l'ensemble correspondant aux contraintes de positions. Ce dernier reprendra alors l'ensemble des contraintes liants les différentes pièces entre-elles :

$$C_p = \{C_{p_l p_m} \mid p_l \in K, p_m \in K \backslash \{p_l\}\}$$

Voici le détail de la contrainte :

$$C_{p_l p_m} = ((x_{p_l}, x_{p_m}), \{(i_{p_l}, j_{p_m}, i_{p_l}, j_{p_m}) \in N^4 \mid \neg((i_{p_l} = i_{p_m}) \wedge (j_{p_l} = j_{p_m}))\})$$

Nous allons à présent définir l'ensemble correspondant aux contraintes de dominations. Ce dernier permettra de s'assurer qu'aucune pièce n'en menace une autre :

$$\begin{aligned} C_d = &\{C_{t_l p_m} |\ t_l \in K_1, p_m \in K \backslash \{t_l\}\} \cap \\ &\{C_{f_l p_m} |\ f_l \in K_2, p_m \in K \backslash \{f_l\}\} \cap \\ &\{C_{c_l p_m} |\ c_l \in K_3, p_m \in K \backslash \{c_l\}\} \end{aligned}$$

Voici à présent le détail des diverses contraintes :

$$\begin{aligned} C_{t_l p_m} = ((x_{t_l}, x_{p_m}), \{(i_{t_l}, i_{p_m}, j_{t_l}, j_{p_m}) \in N^4 | \neg([(i_{t_l} = i_{p_m}) \vee (j_{t_l} = j_{p_m})] \wedge \\ [(i_{t_l} = i_{p_m}) \rightarrow (\nexists\ x_p, \{p \in K \backslash \{t_l, p_m\} | i_p = i_{t_l} \wedge j_p \in \{j_{t_l}, ..., j_{p_m}\}] \wedge \\ [(j_{t_l} = j_{p_m}) \rightarrow (\nexists\ x_p, \{p \in K \backslash \{t_l, p_m\} | j_p = i_{t_l} \wedge i_p \in \{i_{t_l}, ..., i_{p_m}\}\}])\}) \end{aligned}$$

$$\begin{aligned} C_{f_l p_m} = ((x_{f_l}, x_{p_m}), \{(i_{f_l}, i_{p_m}, j_{f_l}, j_{p_m}) \in N^4 | \neg([abs(i_{f_l} - i_{p_m}) = (abs(j_{f_l} - j_{p_m}))] \wedge \\ [\nexists\ x_p, \{p \in K \backslash \{f_l, p_m\} | i_p \in \{i_{t_l}, ..., i_{p_m}\} \wedge j_p \in \{j_{t_l}, ..., j_{p_m}\}\}])\}) \end{aligned}$$

$$\begin{aligned} C_{c_l p_m} = ((x_{c_l}, x_{p_m}), \{(i_{c_l}, i_{p_m}, j_{c_l}, j_{p_m}) \in N^4 | (abs(i_{c_l} - i_{p_m}) \neq 2 \wedge abs(i_{c_l} - i_{p_m}) \neq 1) \wedge \\ (abs(i_{c_l} - i_{p_m}) \neq 1 \wedge abs(i_{c_l} - i_{p_m}) \neq 2)\}) \end{aligned}$$

Nous pouvons maintenant affirmer que le respect de ces contraintes sur les variables fournies nous permettra d'obtenir le résultat du problème d'indépendance.

2.3 Problème de domination

Le problème de domination consiste à déterminer s'il est possible d'assigner à chacune des pièce une position distincte sur l'échiquier de sorte que chaque case soit occupée ou menacée par au moins une pièce.

Exprimer une instance quelconque du problème de domination par un CSP équivalent, en exposant de manière claire les variables, leurs domaines respectifs, et les contraintes à respecter.

2.3.1 Variables

Les variables seront définies de la même manière que pour l'exercice précédent. Nous créons cependant une nouvelle variable représentant les cases vides :

$$X = \{x_t |\ t \in K1\} \cup \{x_f |\ f \in K2\} \cup \{x_c |\ c \in K3\} \cup \{x_v |\ v \in K4\}$$

Par facilité de notation, nous avons également définit les ensembles suivants qui sont directement liés aux places et pièces du jeu :

$$N = \{1, \ldots, n\}$$
$$K_1 = \{t_1, \ldots, t_{k1}\}$$
$$K_2 = \{f_1, \ldots, f_{k2}\}$$
$$K_3 = \{c_1, \ldots, c_{k3}\}$$
$$K = K_1 \cup K_2 \cup K_3$$
$$K_4 = \{v_1, \ldots, v_{(n*n)-(k_1+k_2+k_3)}\}$$

2.3.2 Domaines

Les domaines sont définis de façon identique au problème précédent pour l'ensemble des pièces mises à disposition :

$$D_t = \{(i,j) \mid 1 \leq i \leq n \wedge 1 \leq j \leq n\}$$
$$D_f = \{(i,j) \mid 1 \leq i \leq n \wedge 1 \leq j \leq n\}$$
$$D_c = \{(i,j) \mid 1 \leq i \leq n \wedge 1 \leq j \leq n\}$$

Etant donné la présence de nouvelles variables, il faut également ajouté le domaine s'y appliquant à savoir :

$$D_v = \{(i,j) \mid 1 \leq i \leq n \wedge 1 \leq j \leq n\}$$

2.3.3 Contraintes

Les contraintes appliquées dans le cas du problème de domination seront quant à elles légèrement différentes. Dans ce nouveau problème, nous avons l'affirmation suivante qui doit être respectée :

L'ensemble des cases libres doivent être menacées.

Une case est occupée par maximum une pièce.

Dans ce second problème, notre ensemble de contrainte sera alors composé des contraintes de menaces et des contraintes de positions :

$$C = C_m \cap C_p$$

Nous allons commencer par définir l'ensemble de contraintes C_p. Cet ensemble représentera l'impossibilité de positionner deux pièces sur la même case :

$$C_p = \{C_{p_l p_m} \mid p_l \in K, p_m \in K \backslash \{p_l\}\}$$

Il sera alors défini de la façon suivante :

$$C_{p_l p_m} = ((x_{p_l}, x_{p_m}), \{(i_{p_l}, j_{p_m}, i_{p_l}, j_{p_m}) \in N^4 \mid \neg((i_{p_l} = i_{p_m}) \wedge (j_{p_l} = j_{p_m}))\})$$

Nous pouvons à présent détailler l'ensemble permettant d'appliquer les menaces sur chaque cases vides :

$$\begin{aligned} C_m = &\{C_{v_l t_m} |\ v_l \in K_4, t_m \in K_1\} \cup \\ &\{C_{v_l f_m} |\ v_l \in K_4, f_m \in K_2\} \cup \\ &\{C_{v_l c_m} |\ v_l \in K_4, c_m \in K_3\} \end{aligned}$$

Qui seront appliquées de la manière suivante :

$$\begin{aligned} C_{v_l t_m} = ((x_{v_l}, x_{t_m}), \{(i_{v_l}, i_{t_m}, j_{v_l}, j_{t_m}) \in N^4 | &[(i_{v_l} = i_{t_m}) \vee (j_{v_l} = j_{t_m})] \wedge \\ &[(i_{v_l} = i_{t_m}) \rightarrow \nexists\ x_p, \{p \in K \backslash \{t_m\} | i_p = i_{t_m} \wedge j_p \in \{j_{t_m}, ..., j_{v_l}\}] \wedge \\ &[(j_{v_l} = j_{t_m}) \rightarrow \nexists\ x_p, \{p \in K \backslash \{t_m\} | j_p = j_{t_m} \wedge i_p \in \{i_{t_m}, ..., i_{v_l}\}\}]\}) \end{aligned}$$

$$\begin{aligned} C_{v_l f_m} = ((x_{v_l}, x_{f_m}), \{(i_{v_l}, i_{f_m}, j_{v_l}, j_{f_m}) \in N^4 | &[abs(i_{v_l} - i_{f_m}) = (abs(j_{v_l} - j_{f_m}))] \wedge \\ &[\nexists\ x_p, \{p \in K \backslash \{f_m, v_l\} | i_p \in \{i_{f_m}, ..., i_{v_l}\} \wedge j_p \in \{j_{f_m}, ..., j_{v_l}\}\}]\}) \end{aligned}$$

$$C_{v_l c_m} = ((x_{v_l}, x_{c_m}), \{(i_{v_l}, i_{c_m}, j_{v_l}, j_{c_m}) \in N^4 | [(i_{v_l} = i_{c_m}) \vee (j_{v_l} = j_{c_m})]$$

2.4 Implémentation

Notre implémentation des problèmes ci-dessus nous a permis de répondre aux questions 2.3, Bonus et 2.4 de l'énoncé. Le code qui en résulte est fournis en annexe de ce document. Il a été implémenté comme demandé à l'aide de l'outil ChocoSolver. Afin de respecter les consignes imposées, nous avons également ajouté la possibilité de lire les commandes lors du lancement du programme. Voici les différentes options proposées :

```
usage: board solver
 -F <arg>          file representing the board
 -d                solve a domination problem
 -i                solve an independence problem
 -n <size>         size of the board (default is 8)
 -t <rooks>        number of rooks (default is 0)
 -f <bishops>      number of bishops (default is 0)
 -c <knights>      number of knights (default is 0)
 -m,--minimize     minimize the total number of pieces
                   - numbers of pieces are used as upper bounds
```

On retrouve sur cette images l'ensemble des options disponibles étant donné que le programme réalisé est générique pour les trois problèmes donnés. Les programmes seront lancés via les commandes suivantes :

— Domination : -d -n <taille tableau> -c <nombre max> -m
— Indépendance : -i -n <taille> -c <cavaliers> -t <tours> -f <fous>

2.5 Bonus

Notre implémentation répond également à la question bonus. Il suffit d'ajouter une fonction reprenant la façon dont la pièce en menace une autre afin de pouvoir l'ajoutée à notre programme.

3 Surveillance de musée

La résolution du problème de musée a été réalisée en dérivant les résultats obtenus lors des exercices précédents. On s'est donc directement inspirés du problème de domination afin trouver

une réponse adéquate. Avant de commencer le détail du problème sous la forme CSP, détaillons les données fournies avec le problème :

$$O = \{(i_{o1}, j_{o1}), \ldots, (i_{ok}, j_{ok})\}$$

$$V = \{(i_{v1}, j_{v1}), \ldots, (i_{vm}, j_{vm})\}$$

Où l'ensemble O représente les positions des cases occupées et l'ensemble V des cases vides. Notons également que la taille en X et en Y sera également fournie avec le problème (les variables du problème seront dénommées par ces deux lettres).

3.1 Variables

Les variables qui seront utilisées pour la suite de l'exercice seront définies ici. On retrouvera les caméras est, ouest, nord et sud mais également les cases vides et occupées. Voici les ensembles et variables utilisés :

$$\begin{aligned} E &= \{e_1, .., e_n\} \\ W &= \{w_1, .., w_p\} \\ N &= \{n_1, .., n_q\} \\ S &= \{s_1, .., s_r\} \\ T &= E \cup W \cup N \cup S \end{aligned}$$

$$X = \{x_v \in V\} \cup \{x_e | e \in E\} \cup \{x_w | w \in W\} \cup \{x_n | n \in N\} \cup \{x_s | s \in S\} \cup \{x_o \in O\}$$

3.2 Domaines

Les domaines sont fortement comparables à ceux utilisés dans les exercices précédents :

$$\begin{aligned} D_e &= \{(i,j) \in V\} \\ D_w &= \{(i,j) \in V\} \\ D_n &= \{(i,j) \in V\} \\ D_s &= \{(i,j) \in V\} \end{aligned}$$

3.3 Contraintes

Les contraintes appliquées sur ce problème seront encore une fois découpées en multiples parties à savoir :

Il n'existe pas de case vide qui soit non occupée et non visée
Il n'existe pas de case ayant deux pièces sur la même case

On va alors commencer par déterminer qu'il n'existe pas deux pièces au même endroit :

$$C_p = \{C_{p_l p_m} |\ p_l \in T, p_m \in T \backslash \{p_l\}\}$$

$$C_{p_l p_m} = ((x_{p_l}, x_{p_m}), \{(i_{p_l}, i_{p_m}, j_{p_l}, j_{p_m}) \in V^2 |\ \neg((i_{p_l} = i_{p_m}) \wedge (j_{p_l} = j_{p_m}))\})$$

A ce premier ensemble de contraintes, on vient alors ajouter que toutes les cases doivent être occupées ce qui donne :

$$\begin{aligned} C_c = &\{C_{e_1 c_2} |\ c_1 \in E, c_2 \in V\} \cup \\ &\{C_{w_1 c_2} | w_1 \in W, c_2 \in V\} \cup \\ &\{C_{n_1 c_2} | n_1 \in N, c_2 \in V\} \cup \\ &\{C_{s_1 c_2} | s_1 \in S, c_2 \in V\} \end{aligned}$$

Ce qui nous donne alors pour chacune des contraintes les résultats suivants :

$$C_{e_1c_2} = ((i_{e_1}, i_{c_1}, j_{e_1}, j_{c_1}) \in V^2 | [j_{e_1} = j_{c_1}]) \wedge [i_{e_1} \leq i_{c_1}] \wedge [\nexists x_o | (j_{e_1} = j_o) \wedge (i_{e_1} \leq i_o \leq i_{c_1})]$$
$$C_{w_1c_2} = ((i_{w_1}, i_{c_1}, j_{w_1}, j_{c_1}) \in V^2 | [j_{w_1} = j_{c_1}]) \wedge [i_{c_1} \leq i_{w_1}] \wedge [\nexists x_o | (j_{w_1} = j_o) \wedge (i_{c_1} \leq i_o \leq i_{w_1})]$$
$$C_{s_1c_2} = ((i_{s_1}, i_{c_1}, j_{s_1}, j_{c_1}) \in V^2 | [i_{s_1} = i_{c_1}]) \wedge [j_{s_1} \leq j_{c_1}] \wedge [\nexists x_o | (i_{s_1} = i_o) \wedge (j_{s_1} \leq j_o \leq j_{c_1})]$$
$$C_{n_1c_2} = ((i_{n_1}, i_{c_1}, j_{n_1}, j_{c_1}) \in V^2 | [i_{n_1} = i_{c_1}]) \wedge [j_{c_1} \leq j_{n_1}] \wedge [\nexists x_o | (i_{n_1} = i_o) \wedge (j_{c_1} \leq j_o \leq j_{n_1})]$$

Ces quatres contraintes nous assurent donc que pour chaque cases il existe au moins un capteur la visant. Etant donné que nous sommes dans un problème à minimum, il va falloir minimiser l'ensemble des solutions. Pour ce faire, il faut ajouter :

$$C_s = \{\nexists\ T_1 |\ (|T_1| < |T|) \wedge (T \neq T_1)\}$$

Où T et T_1 sont des ensembles formés de E,W,N,S. On obtient alors au final :

$$C = C_p \cup C_c \cup C_s$$

L'ensemble du code correspondant sera également trouvable en annexe de ce document.