Module INF555, Master S&T 2ème année, Spécialité Génie Logiciel

Contrôle Continu (2008 – 2009)

Ce projet s'intéresse à la résolution de grilles de Sudoku. Le sujet comporte des questions demandant des réponses rédigées, ainsi que les spécifications (grossières) de l'implémentation devant être réalisée en langage C.

1 Le Sudoku

Le Sudoku est un casse-tête logique. Le jeu est paramétré par un entier $d \geq 1$ appelé la *dimension* et un entier $n = d^2$. L'objectif est de remplir une grille de taille $n \times n$ avec des éléments d'un *ensemble C de couleurs* de taille $|C| = n$ de sorte que chacune des n lignes, chacune des n colonnes, et chacun des n blocs de taille $d \times d$ contient une et une seule fois la même couleur.

i \ j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0			7					4				3	6		C	
1	2				0	1					7	A				
2	1	8			5		A		E	C	0					
3	C				F				8	4		5	A	0		
4		3		A					C	7	6		B	9	2	
5	4	7		5		9			2					A		
6		2		B					9			F	4			8
7		1	D			4	F				E				5	
8		6				7				E	9			1	8	
9	E			2	9			5					F		6	
10			5					C			3		E		A	4
11		F	8	C		3	1	E					9		7	
12			6	D	A		B	9				E				0
13						5	D	6		2		8			4	1
14					4	8					5	B				D
15		0		8	C				6					F		

FIG. 1 – Grille de Sudoku 16×16.

La grille est initialement pré-remplie (en partie), et le but du casse-tête est de compléter intégralement la grille. Dans ce projet on n'impose aucune condition particulière sur le pré-remplissage de la grille, mais en réalité les grilles de Sudoku ne doivent avoir qu'une et

une seule solution. On pourra consulter par exemple la page `http://en.wikipedia.org/wiki/Sudoku` pour plus d'informations sur le Sudoku.

Exemple. Une grille de Sudoku 16×16 est donnée en figure 1. Cette grille a été obtenue sur `http://www.e-sudoku.fr/`. On notera que l'ensemble des couleurs de cette grille est $C = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$. Afin d'illustrer la règle du jeu, la couleur initiale 9 à la position $(5, 5)$ est marquée en rouge, et les cases sur fond rouge ne peuvent donc pas contenir la couleur 9.

Présentation formelle du Sudoku

Le Sudoku se formalise en introduisant l'ensemble $P = \{0, \ldots, n-1\} \times \{0, \ldots, n-1\}$ des *positions* du jeu. *Une grille* est une fonction $G : P \to C$ associant à chaque position p de P une couleur $G(p)$ de C. Un *remplissage initial* est un ensemble $I \subseteq P \times C$ tel que pour toute position p il existe au plus une couleur c vérifiant $(p, c) \in I$. On dira qu'une grille G complète un remplissage initial I si $G(p) = c$ pour tout $(p, c) \in I$.

Pour contraindre une grille à n'utiliser qu'une seule fois chaque couleur par ligne, colonne et bloc, on introduit trois relations d'équivalence $\sim_l, \sim_c, \sim_b$ sur l'ensemble des positions ; la relation $\sim_l$ est définie par $(i, j) \sim_l (i', j')$ si et seulement si $i = i'$; la relation $\sim_c$ est définie par $(i, j) \sim_c (i', j')$ si et seulement si $j = j'$; la relation $\sim_b$ est définie par $(i, j) \sim_b (i', j')$ si et seulement s'il existe $r, s \in \{0, \ldots, d-1\}$ tels que $rd \leq i, i' < (r+1)d$ et $sd \leq j, j' < (s+1)d$. On appellera respectivement une *ligne*, une *colonne* et un *bloc* une classe d'équivalence respectivement pour $\sim_l$, $\sim_c$ et $\sim_b$. L'ensemble D suivant est appelé *l'ensemble des dépendances* :

$$D = \{(p, q) \in P \times P \mid p \neq q \wedge (p \sim_l q \ \vee \ p \sim_c q \ \vee \ p \sim_b q)\}$$

On dira qu'une grille $G : P \to C$ est *valide* si $G(p) \neq G(q)$ pour toute dépendance $(p, q) \in D$. Une grille valide complétant un *remplissage initial* I sera appelée une *solution* de I.

Question 1 *Montrer que pour toute grille valide $G : P \to C$ chaque couleur $c \in C$ apparaît une et une seule fois dans chaque ligne, chaque colonne et chaque bloc.*

2 Résolution approchée par contraintes

On s'intéresse dans cette partie à une résolution approchée du Sudoku par résolution d'un système de contraintes. Les *variables* du système de contraintes sont $\mathtt{X}_p$ avec $p \in P$. Chaque variable $\mathtt{X}_p$ représente une sur-approximation de l'ensemble des couleurs possibles à la position p.

Pour définir formellement le système de contraintes, on se place dans le treillis complet $(\wp(C), \subseteq)$ où $\wp(C)$ est l'ensemble des parties de C. Les variables $\mathtt{X}_p$ prennent ainsi pour valeur un élément de $\wp(C)$. Formellement, une *valuation* est une fonction $\pi : P \to \wp(C)$ associant à chaque variable un sous-ensemble de C. L'interprétation d'une valuation π est la suivante : pour chaque position $p \in P$, les couleurs possibles en p sont parmi $\pi(p)$. L'ordre partiel $\subseteq$ sur $\wp(C)$ est étendu point-à-point sur l'ensemble $P \to \wp(C)$ des valuations, de

manière classique par : $\pi_1 \subseteq \pi_2$ si $\pi_1(p) \subseteq \pi_2(p)$ pour toute position $p \in P$. Rappelons que l'extension $(P \to \wp(C), \subseteq)$ est également un treillis complet. On observe que l'ordre partiel $\subseteq$ sur les valuations permet de comparer leur précision : si $\pi_1 \subseteq \pi_2$ alors l'information représentée par π_1 est plus précise que celle représentée par π_2. Pour toute grille $G : P \to C$ et pour tout remplissage initial $I \subseteq P \times C$, on note π_G et π_I les valuations définies par :

$$\pi_G(p) \;=\; \{G(p)\} \qquad\qquad \pi_I(p) \;=\; \begin{cases} C & \text{si } I \cap (\{p\} \times C) = \emptyset \\ \{c\} & \text{si } I \cap (\{p\} \times C) = \{(p,c)\} \end{cases}$$

La valuation π_G exprime simplement que pour chaque position $p \in P$, la couleur $G(p)$ est la seule couleur possible en p. La valuation π_I exprime que pour chaque couple $(p,c) \in I$, la couleur c est la seule couleur possible à la position p. Pour définir le système de contraintes, on va utiliser la fonction $f : \wp(C) \to \wp(C)$ définie par :

$$f(X) \;=\; \begin{cases} \emptyset & \text{si } X = \emptyset \\ C \setminus X & \text{si } X \text{ est un singleton} \\ C & \text{dans les autres cas} \end{cases}$$

Observons que la fonction f est croissante. Le *système de contraintes* $\mathcal{C}(\pi_0)$ donnant une solution approchée du Sudoku pour la valuation initiale π_0 est donné ci-dessous.

$$\mathcal{C}(\pi_0) \qquad \begin{cases} \mathtt{X}_p \;\subseteq\; \pi_0(p) & \text{pour chaque } p \in P \quad (1) \\ \mathtt{X}_q \;\subseteq\; f(\mathtt{X}_p) & \text{pour chaque } (p,q) \in D \quad (2) \end{cases}$$

Les contraintes (1) proviennent de la valuation initiale : les couleurs possibles sont restreintes à celle autorisées par la valuation initiale. Rappelons que toute grille valide G vérifie $G(p) \neq G(q)$ pour toutes positions p et q dépendantes. Cette condition est exprimée sous une forme affaiblie dans les contraintes (2). En effet une contrainte de la forme $\mathtt{X}_q \subseteq f(\mathtt{X}_p)$ exprime la propriété suivante : si $\mathtt{X}_p$ est un singleton $\{c\}$ (ce qui signifie que c est l'unique couleur possible à la position p), alors c ne fait pas partie des couleurs possibles à la position q.

Remarquons que chaque contrainte de $\mathcal{C}(\pi_0)$ est de la forme $\mathtt{X}_q \subseteq g(\mathtt{X}_{p_1}, \ldots, \mathtt{X}_{p_k})$ où $q, p_1, \ldots, p_k$ sont des positions et g est une fonction de $\wp(C) \times \cdots \times \wp(C)$ dans $\wp(C)$. Une valuation π *satisfait* une contrainte $\mathtt{X}_q \subseteq g(\mathtt{X}_{p_1}, \ldots, \mathtt{X}_{p_k})$ lorsque $\pi(q) \subseteq g(\pi(p_1), \ldots, \pi(p_k))$. On dit que π *satisfait* le système de contraintes $\mathcal{C}(\pi_0)$ si π satisfait toutes les contraintes de $\mathcal{C}(\pi_0)$. En d'autres termes, en remplaçant dans $\mathcal{C}(\pi_0)$ chaque variable $\mathtt{X}_p$ par l'ensemble $\pi(p)$, les inclusions ensemblistes du système obtenu sont toutes vérifiées.

Question 2 *Montrer que pour toute grille G solution du remplissage I, la valuation π_G satisfait le système de contraintes $\mathcal{C}(\pi_I)$.*

Une *solution maximale* du système de contraintes $\mathcal{C}(\pi_0)$ est une valuation π satisfaisant $\mathcal{C}(\pi_0)$ et vérifiant $\pi' \subseteq \pi$ pour toute valuation π' satisfaisant $\mathcal{C}(\pi_0)$.

Question 3 *Montrer que pour toute valuation initiale π_0, le système de contraintes $\mathcal{C}(\pi_0)$ admet une unique solution maximale.*

Question 4 *Soient π_0 une valuation initiale et π la solution maximale de $\mathcal{C}(\pi_0)$. Montrer que s'il existe une position $p \in P$ telle que $\pi(p) = \emptyset$ alors :*

1. *on a $\pi(q) = \emptyset$ pour toute position $q \in P$, et*
2. *aucune grille n'est solution d'un remplissage initial I tel que $\pi_I \subseteq \pi_0$.*

La solution maximale de $\mathcal{C}(\pi_0)$ ne donne qu'une sur-approximation assez grossière de l'ensemble des couleurs possibles à chaque position $p \in P$. Afin d'améliorer la précision de l'approximation obtenue, on va ajouter à $\mathcal{C}(\pi_0)$ des contraintes plus fines, qui tiennent simultanément compte des dépendances au sein de chaque ligne, de chaque colonne et de chaque bloc. Pour chaque classe d'équivalence E de $\sim_l$, $\sim_c$ et de $\sim_b$, et pour chaque position $p \in E$, on ajoute à $\mathcal{C}(\pi_0)$ la contrainte :

$$\mathtt{X}_p \ \subseteq \ \left\{ H(p) \;\middle|\; H : E \leftrightarrow C \ \wedge \bigwedge_{q \in E} H(q) \in \mathtt{X}_q \right\} \tag{3}$$

Dans cette contrainte, l'expression $E \leftrightarrow C$ désigne l'ensemble des bijections de E dans C. Observons que la restriction d'une grille valide à une classe d'équivalence E (une ligne, une colonne ou un bloc) définit une bijection de E dans C. Ainsi pour tout $p \in E$, les couleurs de $\mathtt{X}_p$ peuvent être restreintes aux $H(p)$ où H est une bijection de E dans C vérifiant $H(q) \in \mathtt{X}_q$ pour tout $q \in E$. Cette propriété est exprimée par l'inclusion (3) ci-dessus.

On note $\mathcal{C}'(\pi_0)$ l'ensemble de contraintes obtenu par ajout à $\mathcal{C}(\pi_0)$ des contraintes (3) pour toute classe d'équivalence E de $\sim_l$, $\sim_c$ et de $\sim_b$, et pour toute position $p \in E$.

Question 5 *Montrer que le système de contraintes $\mathcal{C}'(\pi_0)$ vérifie les énoncés des trois questions précédentes.*

3 Résolution exacte en logique propositionnelle

On s'intéresse dans cette partie au calcul d'une formule propositionnelle dénotant l'ensemble des solutions d'un remplissage initial. Les *variables propositionnelles* sont $\mathtt{x}_p$ avec $p \in P$. Chaque variable $\mathtt{x}_p$ représente la couleur à la position p. On considère l'ensemble Φ des formules de la grammaire suivante où $p, q \in P$ et $c \in C$:

$$\varphi \quad := \quad \varphi \vee \varphi \quad | \quad \varphi \wedge \varphi \quad | \quad \neg\varphi \quad | \quad \mathtt{x}_p = c \quad | \quad \mathtt{x}_p = \mathtt{x}_q$$

Les *modèles d'une formule* $\varphi \in \Phi$ seront des grilles $G : P \to C$ et on notera $G \models \varphi$ lorsque une grille G est un modèle de la formule φ. Plus formellement, on définit la relation binaire $\models \subseteq (P \to C) \times \Phi$ de *satisfaction* d'une formule par une grille de la manière suivante : pour toute grille $G : P \to C$, pour toutes formules $\varphi_1, \varphi_2, \varphi \in \Phi$, pour toutes positions $p, q \in P$ et pour toute couleur $c \in C$,

$$\begin{cases} G \models \varphi_1 \vee \varphi_2 & \text{si } G \models \varphi_1 \text{ ou } G \models \varphi_2 \\ G \models \varphi_1 \wedge \varphi_2 & \text{si } G \models \varphi_1 \text{ et } G \models \varphi_2 \\ G \models \neg\varphi & \text{si on n'a pas } G \models \varphi \\ G \models \mathtt{x}_p = c & \text{si } G(p) = c \\ G \models \mathtt{x}_p = \mathtt{x}_q & \text{si } G(p) = G(q) \end{cases}$$

On considère un remplissage initial I. Remarquons qu'une grille G complète I si et seulement si G satisfait la formule suivante :

$$\bigwedge_{(p,c)\in I} \mathtt{x}_p = c$$

Question 6 *Donner une formule satisfaite par une grille G si et seulement si G est valide.*

Question 7 *En déduire une formule satisfaite par une grille G si et seulement si G est une solution de I.*

4 Implémentation

On implémentera la résolution du Sudoku dans un premier temps par la technique de résolution approchée par contraintes développée en section 2, puis par la technique de résolution exacte en logique propositionnelle développée en section 3, et enfin en combinant les deux techniques.

L'implémentation prendra en entrée (par exemple sur l'entrée standard) un remplissage initial et retournera (par exemple sur la sortie standard) les grilles solutions du remplissage initial. Les grilles seront lues et écrites dans le format simple suivant : chaque ligne de texte correspondra à une ligne de la grille, et les couleurs d'une même ligne seront séparées par le caractère *espace*. Pour le remplissage initial, on utilisera le caractère _ lorsque la case n'est pas pré-remplie. Par exemple, le texte correspondant au remplissage initial de la figure 1 est le suivant :

```
_ _ 7 _ _ _ _ 4 _ _ _ 3 6 _ C _
2 _ _ _ 0 1 _ _ _ _ 7 A _ _ _ _
1 8 _ _ 5 _ A _ E C 0 _ _ _ _ _
C _ _ _ F _ _ _ 8 4 _ 5 A 0 _ _
_ 3 _ A _ _ _ _ C 7 6 _ B 9 2 _
4 7 _ 5 _ 9 _ _ 2 _ _ _ _ A _ _
_ 2 _ B _ _ _ _ 9 _ _ F 4 _ _ 8
_ 1 D _ _ 4 F _ _ _ E _ _ _ 5 _
_ 6 _ _ _ 7 _ _ _ E 9 _ _ 1 8 _
E _ _ 2 9 _ _ 5 _ _ _ _ F _ 6 _
_ _ 5 _ _ _ _ C _ _ 3 _ E _ A 4
_ F 8 C _ 3 1 E _ _ _ _ 9 _ 7 _
_ _ 6 D A _ B 9 _ _ _ E _ _ _ 0
_ _ _ _ _ 5 D 6 _ 2 _ 8 _ _ 4 1
_ _ _ _ 4 8 _ _ _ _ 5 B _ _ _ D
_ 0 _ 8 C _ _ _ 6 _ _ _ _ F _ _
```

On déduira du remplissage initial fourni en entrée la dimension d et l'entier $n = d^2$. Pour des raisons d'efficacité et de simplicité, l'implémentation sera limitée à $d \leq 4$. On supposera que l'ensemble de couleurs C est $C = \{0, \ldots, n-1\}$, et que chaque couleur du remplissage initial est représentée en hexadécimal, comme dans l'exemple ci-dessus.

4.1 Résolution approchée par contraintes

On implémentera la méthode de recherche de solutions du Sudoku donnée dans l'algorithme récursif Sudoku ci-dessous. Naturellement, la fonction principale du programme appelera Sudoku avec la valuation initiale π_I où I est le remplissage initial donné en entrée. Dans cette sous-section, on se limite au système de contraintes $\mathcal{C}(\pi_0)$, c'est-à-dire aux contraintes (1) et (2).

```
1  Sudoku (π_0 : P → ℘(C))
2      π ← solution maximale de C(π_0)
3      si |π(p)| ≤ 1 pour toute position p ∈ P alors
4          si |π(p)| = 1 pour toute position p ∈ P alors
5              soit G l'unique grille telle que π = π_G
6              afficher G
7      sinon
8          choisir p ∈ P tel que |π(p)| ≥ 2
9          pourchaque c ∈ π(p) faire
10             π'_0 ← π
11             π'_0(p) ← {c}
12             appeler Sudoku (π'_0)
```

Le calcul de la solution maximale de $\mathcal{C}(\pi_0)$ à la ligne 2 sera réalisé par un des algorithmes vu en cours (*round-robin* ou *work list*). Les sous-ensembles X de C seront représentés par des entiers non signés sur 16 bits (type C `uint16_t`), le $i^{\text{ème}}$ bit codant l'appartenance de la couleur i au sous-ensemble X. On utilisera des tableaux d'entiers non signés sur 16 bits pour représenter les valuations. Les opérations ensemblistes seront implémentées de manière efficace avec des opérateurs booléens bit-à-bit. Le choix de $p \in P$ à la ligne 8 est laissé libre dans l'algorithme Sudoku. On décrira le choix réalisé dans l'implémentation.

Question 8 *Montrer que pour tout remplissage I, l'exécution de Sudoku(π_I) termine et affiche toutes les grilles solutions de I une et une seule fois.*

4.2 Résolution symbolique par diagrammes de décision

On souhaite construire un *diagramme de décision (DD)* représentant symboliquement l'ensemble des solutions complétant un remplissage initial I. L'idée est de traduire la formule propositionnelle calculée à la question 7 en opérations élémentaires sur les DD. On utilisera la bibliothèque CUDD disponible à l'adresse `http://vlsi.colorado.edu/~fabio/CUDD/` pour implémenter ces opérations. On notera que CUDD permet de manipuler des *Algebraic Decision Diagrams (ADD)*, une extension des *Binary Decision Diagram (BDD)* permettant d'avoir des variables valuées dans un ensemble énuméré. Pour des raisons de simplicité, on utilisera les ADD. Ainsi, on n'utilisera qu'une variable par position p de P valuée dans l'ensemble des couleurs C. Pour l'implémentation, on choisira l'interface C de CUDD. On remarquera expérimentalement que la construction d'un ADD représentant symboliquement l'ensemble des solutions d'une grille de taille 4×4 est possible mais que le calcul ne termine pas toujours (en temps raisonnable) sur des grilles de plus grande taille.

Question 9 *Quel est le nombre de grilles valides lorsque $d = 2$?*

L'efficacité de l'implémentation peut être améliorée en réordonnant les opérations sur les ADD. En effet, on remarque que la formule calculée à la question 7 est une conjonction $\varphi_1 \wedge \ldots \wedge \varphi_k$ de plusieurs sous-formules : une sous-formule pour chaque $(p, c) \in I$ et une sous-formule pour chaque ligne, colonne et bloc. En réordonnant les formules $\varphi_1, \ldots, \varphi_k$, on ne modifie pas le ADD final. On expérimentera au moins deux ordres différents sur plusieurs grilles et on présentera les résultats obtenus (temps de calcul) dans un tableau. On pourra chercher d'autres heuristiques.

4.3 Combinaison des deux techniques

On souhaite remplacer le système de contraintes $\mathcal{C}(\pi_0)$ utilisé à la ligne 2 de l'algorithme Sudoku par le système étendu $\mathcal{C}'(\pi_0)$, c'est-à-dire avec les contraintes (3). Ainsi, pour chaque classe d'équivalence E de $\sim_l, \sim_c$ et de $\sim_b$, pour chaque position $p \in E$ et pour chaque valuation $\pi : P \to \wp(C)$, on souhaite calculer l'ensemble suivant avec des ADD :

$$\left\{ H(p) \;\middle|\; H : E \leftrightarrow C \;\wedge \bigwedge_{q \in E} H(q) \in \pi(q) \right\}$$

Question 10 *Montrer qu'une couleur c' est dans l'ensemble si et seulement la formule suivante admet un modèle :*

$$\mathtt{x}_p = c' \quad \wedge \quad \bigwedge_{q \in E} \left(\bigvee_{c \in \pi(\mathtt{x}_q)} \mathtt{x}_q = c \right) \quad \wedge \quad \bigwedge_{\substack{q, q' \in E \\ q \neq q'}} \neg(\mathtt{x}_q = \mathtt{x}_{q'})$$

En traduisant la formule propositionnelle donnée à la question précédente en opérations élémentaires sur les ADD, on implémentera avec CUDD la résolution du système de contrainte $\mathcal{C}'(\pi_0)$. Observons que la propagation des contraintes (2) est nettement moins coûteuse que celle des contraintes (3). Pour optimiser le calcul de la solution maximale de $\mathcal{C}'(\pi_0)$, on pourra par exemple privilégier la propagation des contraintes (2).

On cherchera à améliorer ce système de contraintes avec de nouvelles contraintes de manière à améliorer la précision de la solution maximale. Les contraintes (3) utilisées précédemment se limitent à une ligne, une colonne ou un bloc. On pourrait par exemple considérer des contraintes plus fines que les contraintes (3) en considérant simultanément toutes les positions dépendantes de la position p, c'est à dire l'ensemble des positions $q \in P$ telles que $(p, q) \in D$. Le calcul effectif de la solution maximale du nouveau système de contraintes passerait à nouveau par la construction d'ADD pour ces nouvelles contraintes.