Examen MIF06 - 8 janvier 2019 – 2h (Tiers Temps + 40 min)
Documents Interdits

PARTIE 1 - MODELISATION DE PROBLEMES – 6 PTS

Le but du Sudoku est de remplir une grille avec une série de symboles tous différents, qui ne se trouvent jamais plus d'une fois sur une même ligne, sur une même colonne ou dans une même sous-grille.

Nous souhaitons remplir une grille de 9 × 9 où les symboles sont des chiffres allant de 1 à 9. Les sous-grilles sont donc des carrés de 3 × 3. Quelques chiffres sont déjà disposés dans la grille. Ci-contre, un exemple de Sudoku à résoudre.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Question 1 : Proposez une modélisation de ce problème, avec les données précises de l'exemple, sous forme d'un CSP.

========== Indices de correction ==========

Exemple de formalisation (pas la seule) :

(1 pts) Variables : Xij représentant une case (i, j) avec $1 \leq i \leq 9$, $1 \leq j \leq 9$, soit 81 variables

(1 pts) Domaines : Dxij = {1, . . . , 9}, $\forall$(i, j), sauf pour les cases fixées initialement où le domaine n'a qu'une unique valeur.

(4 pts) Contraintes :

* Les chiffres sur chaque ligne sont différents : $\forall i \ \forall j \ \forall k \in [1,9]$ Xij ≠ Xik

* Les chiffres dans chaque colonne sont différents : $\forall i \ \forall j \ \forall k \in [1,9]$ Xji ≠ Xki

* Les chiffres dans chaque bloc 3×3 sont différents :

$\forall i,j,k,l \in [1,3] \ \forall n,m \in [0,2]$ (i≠k et j≠l) Xi+3n,j+3m ≠ Xk+3n,l+3m

==========

PARTIE 2 – RAISONNEMENT LOGIQUE – 5 PTS

Soit le raisonnement suivant :

1. Aucun singe n'est soldat.
2. Tous les singes sont malicieux.
3. donc : Quelques créatures malicieuses ne sont pas des soldats.

Question 2 : Ce raisonnement est-il valide ? Que votre réponse soit positive ou négative, justifiez-la à l'aide du principe de réfutation.

========== Indices de correction ==========

(1 pts) On considère les prédicats :

- s(x) : « x est un singe » ;
- sdt(x) : « x est un soldat » ;
- mal(x) : « x est malicieux ».

(2 pts) La formalisation du raisonnement donne :

1. $\forall x\,(s(x) \rightarrow \neg sdt(x)) \equiv \forall x\,(\neg s(x) \vee \neg sdt(x))$
2. $\forall x\,(s(x) \rightarrow mal(x)) \equiv \forall x\,(\neg s(x) \vee mal(x))$
3. $\exists x\,(mal(x) \wedge \neg sdt(x))$ dont la négation est : $\forall x\,(\neg mal(x) \vee sdt(x))$

(1 pts) La forme clausale est : $C = \{\neg s(x1) \vee \neg sdt(x1), \neg s(x2) \vee mal(x2), \neg mal(x3) \vee sdt(x3)\}$.

(1 pts) On a $H\infty = \{a\}$.

On ne trouve pas d'ensemble de clauses contradictoire, ce raisonnement n'est donc pas valide.

Pour que ce raisonnement soit valide, il manque s(a) i.e. avoir au moins un élément qui soit un singe.

==========

PARTIE 3 – SYSTEME A BASE DE CONNAISSANCES – 4 PTS

Soient les quatre règles suivantes :

(1) Si musicien alors aime les mathématiques
(2) Si grand et brun alors musicien
(3) Si non lunettes alors brun
(4) Si lunettes alors musicien

Question 3 : Soit le fait : « grand ». Quel type de moteur d'inférences faudrait-il utiliser pour monter « aime les mathématiques » ? Quel serait le résultat ? Déroulez les inférences sur votre copie.

========== Indices de correction ==========

(2 pts) Moteur en chainage arrière

But : « aime-maths »

- R1 > nouveau-but « musicien »
  - R2 > nouveau-but « grand » et « brun »
  - « grand » est un fait
  - « brun »
    - R3 > nouveau-but « non lunettes »

        - échec sur « non lunettes »
- Retour au but « musicien
    - R4 > nouveau but « lunette »
    - « lunette » pas dans partie droite
- Retour au but « musicien

Arrêt en échec car aucune règle ne permet musicien et pas d'autre règle pour aime-maths

==========

Question 4 : Soient les faits : « grand » et « lunettes ». Que peut-on découvrir comme information en lançant le moteur en chainage avant ? Déroulez l'exécution du moteur sur votre copie.

========== Indices de correction ==========

(2 pts) Moteur en chainage avant

F1 grand, F2 lunette

- R1 non
- R2 non
- R3 non
- R4 oui => F3 musicien
- R1 oui => F4 aime-maths
- R2 non
- R3 non

Arrêt

==========

PARTIE 4 – PROLOG – 5 PTS

Six flèches sont dans la position de la figure 1. On souhaite les positionner comme dans la figure 2.

Figure 1

Figure 2

On symbolise ces deux états par « `hhhbbb` » et « `bbbhhh` » avec « `h` » pour « flèche haute » et « `b` » pour « flèche basse ».

On définit quatre opérateurs de transition :

- R1 : retournement de deux flèches hautes adjacentes (« `hh` » devient « `bb` »)
- R2 : retournement d'une flèche haute et d'une flèche basse adjacentes (« `hb` » devient « `bh` »)

- R3 : retournement d'une flèche basse et d'une flèche haute adjacentes (« `bh` » devient « `hb` »)
- R4 : retournement de deux flèches basses adjacentes (« `bb` » devient « `hh` »)

Question 5 : Définissez le prédicat `rempl(S1,S2,L1,L2)` qui est satisfait si le remplacement de la sous-liste S1 par S2 dans la liste L1 donne L2.

Question 6 : Définissez les prédicats `initial, final` et `operateur` pour le problème des flèches qui permettent au prédicat `resoudre` ci-dessous de trouver des solutions au problème.

```
:- dynamic interdit/1.

/* recherche dans un graphe d'etats */

/*recherche(EtatCourant,EtatFinal,ListeTaboue,ListeOp) tous donnes sauf
ListeOp */
recherche(Ec,Ec,_,[]):-!.
recherche(Ec,Ef,Ltaboue,[Op|Lop]) :-
      operateur(Ec,Op,Es),
      not(interdit(Es)),
      not(member(Es,Ltaboue)),
      %write(Ec), write(' '), write(Op), write(' '),write(Es),nl,
      recherche(Es,Ef,[Es|Ltaboue],Lop).

resoudre(Sol) :- initial(Ei), final(Ef), recherche(Ei,Ef,[Ei],Sol).
```

========== Indices de correction ==========

```
Correction : /* probleme des fleches */


(1 pts)
initial([b,h,h,b,b,h]).
final([b,b,b,h,h,h]).

(4 pts)
operateur(E1,r1,E2) :- remplace([b,b],E1,[h,h],E2).
operateur(E1,r2,E2) :- remplace([h,b],E1,[b,h],E2).
operateur(E1,r3,E2) :- remplace([b,h],E1,[h,b],E2).
operateur(E1,r4,E2) :- remplace([h,h],E1,[b,b],E2).

/* remplace(L1,L2,L,R) tous donnes sauf R */
remplace(L1,L,L2,R) :-
append(Deb,L3,L),
append(L1,Fin,L3),
append(Deb,L2,L4),
append(L4,Fin,R).
```

==========