TD6

Systèmes à base de règles d'ordre 0

Exercice 1. Règles conjonctives positives (chaînage avant)

Soit la base de faits **BF** = {S,A} et les règles suivantes :

R1 : $B \wedge R \rightarrow C$
R2 : $A \rightarrow T$
R3 : $R \wedge U \rightarrow E$
R4 : $C \wedge E \rightarrow U$
R5 : $S \wedge T \rightarrow U$
R6 : $U \wedge T \rightarrow R$
R7 : $S \wedge U \rightarrow A$

1. Calculer la saturation de BF par les règles en utilisant l'algorithme de chaînage avant naïf.

2. Calculer la saturation de BF par les règles en utilisant l'algorithme de chaînage avant à base de compteurs (rappelé ci-dessous).

```
Algorithme FC(K) // saturation de la base K
// Données : K = (BF, BR)
// Résultat : BF* : BF saturée par application des règles de BR
Début
  àTraiter ← BF
  BF* ← BF
  Pour toute règle R de BR
     Compteur(R)← | hypothèse(R)|
  Tant que àTraiter ≠ ∅
     Retirer F de àTraiter
     Pour toute règle (R : H -> C) ∈ BR telle que F ∈ H          (a)
        Décrémenter Compteur(R)
        Si Compteur(R) = 0              // R est applicable
              Si C ∉ BF*    // l'application de R est utile       (b)
                     Ajouter C à àTraiter
                     Ajouter C à BF*
     FinPour
  FinTantQue
  Retourner BF*
Fin
```

Exercice 2. Implémentation efficace d'un algorithme de chaînage avant

On considère l'algorithme à base de compteurs de l'exercice précédent. On suppose que les symboles propositionnels sont codés par des *entiers* (de 1 à n si n symboles apparaissent dans BF et BR).

1) Imaginer des structures de données qui permettent de :
   a) ne considérer que les règles dont l'hypothèse contient F, lorsque F est traité
   b) tester en temps constant si C appartient à BF*.

2) Étant données ces structures, montrer que l'algorithme a une complexité *linéaire* en la taille de K, c'est-à-dire effectue un nombre d'opérations "élémentaires" borné par *constante* x *taille(K)*.

[Si les symboles sont des chaînes de caractères, comment adapter les structures précédentes de façon à conserver une complexité "presque linéaire" ?]

Exercice 3. Règles conjonctives positives (chaînage arrière)

En considérant la base de connaissances de l'exercice 1, on veut prouver U en chaînage arrière. Dessiner l'arbre de recherche correspondant à la remontée du graphe ET-OU, en supposant que l'algorithme considère les règles par numéro croissant. Vous indiquerez sur chaque feuille traitée : *échec*, *boucle*, ou *BF*.

Exercice 4. Règles conjonctives positives (chaînage arrière)

On considère la base de faits BF = {D, E} et la base de règles BR suivante :

R1 : $B \wedge C \rightarrow A$	R2 : $E \wedge F \rightarrow B$	R3 : $C \rightarrow F$	R4 : $H \rightarrow C$
R5 : $B \rightarrow C$	R6 : $E \wedge G \rightarrow C$	R7 : $D \rightarrow G$	

1) Que contient BF* (saturation de BF par BR) ?
2) Dessiner l'arbre de recherche visant à prouver A en chaînage arrière (comme dans l'exercice précédent, par numéro croissant de règle). Vous pouvez utiliser l'information qu'un certain atome a *déjà* été prouvé.
3) Adapter l'algorithme de chaînage arrière du cours (BC3) de façon à gérer l'information "déjà prouvé".

```
Algorithme BC3(K,Q,L)  // Données : K = (BF,BR), Q un atome
                       //           L un ensemble d'atomes (à ne pas générer)
                       // Résultat : vrai ssi Q est prouvable
                                                              Appel BC3(K,Q,ø)
Début
Si Q ∈ BF, retourner vrai
Pour toute règle R = H1 ∧ … ∧ Hn → Q de BR
    Si aucun des H1 … Hn n'appartient à L // sinon on va boucler
        i ← 1
        Tant Que i ≤ n  et BC3(K, Hi, L ∪ {Q})
            incrémenter i
        Si i > n, retourner vrai   // hypothèse de R prouvée, donc Q aussi

Retourner faux      // aucun des faits et aucune des règles ne permet de prouver Q
Fin
```

[Indication : maintenir une liste des atomes déjà prouvés et conclure directement lorsque Q appartient à cette liste]

Exercice 5. Règles conjonctives positives (chaînage arrière)

On considère la base de faits BF = {A, B, G} et la base de règles BR suivante :

R1 : $J \wedge H \rightarrow I$	R2 : $A \wedge J \wedge C \rightarrow L$	R3 : $B \wedge C \rightarrow L$
R4 : $I \wedge F \rightarrow H$	R5 : $E \wedge J \rightarrow K$	R6 : $G \wedge F \rightarrow H$
R7 : $A \wedge C \rightarrow J$	R8 : $H \wedge K \rightarrow D$	R9 : $A \wedge B \rightarrow C$
R10 : $I \wedge H \rightarrow J$	R11 : $B \wedge L \wedge J \rightarrow F$	

ainsi que l'algorithme de chaînage arrière optimisé construit à l'étape précédente. Dessiner les arbres de recherche obtenus en chaînage arrière pour les buts suivants : **L**, **I** puis **E**. Vous indiquerez sur chaque feuille traitée : *échec*, *boucle*, *BF ou déjà prouvé.*