Logique

Cours de Licence de Sciences du Langage (L2)
Alain Lecomte – Professeur, Université Paris 8

4 Logique propositionnelle - II

4.1 Systèmes d'inférences

4.1.1 Démonstration automatisée

Un des buts recherchés en intelligence artificielle est la démonstration automatique de théorèmes. La notion de théorème est dans ce cas très restreinte. Il s'agit d'une assertion dont on se demande si elle est prouvable ou non dans un système donné. Par exemple, supposons que nous ayons validé les hypothèses suivantes:

- les gens qui ont la rougeole doivent prendre le médicament X
- les gens qui ont de la fièvre et des points rouges au fond de la gorge ont la rougeole
- ceux pour qui la température est au-dessus de 38° sont considérés comme ayant de la fièvre
- Armand a des points rouges au fond de la gorge et a une température de 39°5

On se demande si on doit donner à Armand le médicament X.
L'assertion: "Armand doit prendre le médicament X" est considérée comme une formule à démontrer. Si on peut effectivement la démontrer, alors on dira que c'est un théorème. Pour cela, on peut avoir à sa disposition plusieurs systèmes de règles. Par exemple:

- on a le droit d'utiliser seulement la règle du Modus Ponens et l'introduction de la conjonction (la règle: $\{A, B\} \models A \wedge B$).

On voit tout de suite que, en introduisant quelques symboles à titre d'abréviations, nos hypothèses deviennent:

- $r \Rightarrow x$
- $(f \wedge g) \Rightarrow r$
- $sup38 \Rightarrow f$
- $g \wedge sup38$

Question: x?
La déduction est:

• sup38	(prémisse)
• $sup38 \Rightarrow f$	(prémisse)
• f	(MP)
• g	(prémisse)
• $f \wedge g$	(intro $\wedge$)
• $(f \wedge g) \Rightarrow r$	(prémisse)
• r	(MP)
• $r \Rightarrow x$	(prémisse)
• x	(MP)

Ce faisant, on a utilisé une stratégie particulière de démonstration: on est parti des faits connus, et à chaque pas, on a essayé de voir si les faits acquis (connus ou démontrés) figuraient comme antécédents d'implications, auquel cas, on a appliqué la règle du Modus Ponens. Cette démarche, appelée **chaînage avant**, est accomplie plus ou moins à l'aveuglette jusqu'à ce qu'on trouve enfin la conclusion souhaitée. Mais il peut y avoir plusieurs conclusions qu'on tire d'un ensemble de faits. Certaines ne nous intéressent peut-être pas... Si on applique cette méthode, on risque donc de faire beaucoup de travail inutile.

Une autre solution consiste à partir de **la négation** de la conclusion recherchée. Ici, nous aurions alors l'ensemble suivant d'hypothèses:

- $r \Rightarrow x$
- $(f \wedge g) \Rightarrow r$
- $sup38 \Rightarrow f$
- $g \wedge sup38$
- $\neg x$

Et on cherche à prouver que cet ensemble de propositions est inconsistant (c'est-à-dire permet de déduire une contradiction). La démarche est la suivante:

- $\neg x$ (prémisse)
- $r \Rightarrow x$ (prémisse)
- $\neg r$ (Modus Tollens)
- $(f \wedge g) \Rightarrow r$ (prémisse)
- $\neg (f \wedge g)$ (MT)
- $\neg f \vee \neg g$ (de Morgan)
- g (prémisse)
- $\neg f$ (règle $\{A \vee B, \neg B\} \models A$)
- $sup38 \Rightarrow f$ (prémisse)
- $\neg sup38$ (MT)
- $sup38$ (prémisse)
- $\perp$ (contradiction)

L'intérêt de cette dernière démarche est qu'elle est guidée par la conclusion à laquelle on veut arriver. On ne s'égare pas en route vers d'autres conclusions. On peut constater que les règles d'inférence utilisées ne sont pas les mêmes: il s'agit ici principalement de la règle du Modus Tollens: $\{A \Rightarrow B, \neg B\} \models \neg A$, ainsi que de la règle $\{A \vee B, \neg B\} \models A$ sur laquelle nous allons revenir dans quelques instants. Cette démarche est appelée: **chaînage arrière**. On part en effet de la négation de la conclusion à prouver et on remonte en passant de la négation d'un conséquent de formule implicative à la négation de l'antécédent correspondant et ainsi jusqu'à ce qu'on déduise un fait contradictoire avec un autre. Ces exemples introduisent deux questions :

1) qu'est-ce qui fonde la deuxième démarche?

2) comment automatiser cette démarche? c'est-à-dire: peut-on trouver un algorithme qui la mette en application de manière à toujours trouver la solution en un nombre fini de pas?

4.1.2 Le théorème de la déduction

- *Théorème* : Si *E* est un ensemble de propositions sur un ensemble de variables propositionnelles P, il revient au même de démontrer que B se déduit de $E \cup \{A\}$ ou que $(A \Rightarrow B)$ se déduit de *E*. Autrement dit :

- $E \cup \{A\} \models B$ *équivalent à* $E \models (A \Rightarrow B)$

Ce théorème est souvent appelé "théorème de la déduction". Sa démonstration est très facile :

Démonstration : nous allons plutôt démontrer la contraposée :

- $E \cup \{A\} \not\models B$ *équivalent à* $E \not\models (A \Rightarrow B)$

$E \cup \{A\} \not\models B$ *équivalent à* : il existe une assignation de valeurs de vérité aux variables propositionnelles de P (autrement dit un monde possible) qui rend vraies les formules de *E* ainsi que A mais qui ne rend pas vraie B *équivalent à* : il existe une assignation de valeurs de vérité aux variables propositionnelles de P qui rend vraies les formules de *E* mais qui rend faux $A \Rightarrow B$ *équivalent à* : $E \not\models (A \Rightarrow B)$.

4.1.3 Le théorème de réfutation

Nous allons maintenant prouver le théorème suivant:

- *Théorème*: Soit *E* un ensemble de propositions : *E* |= A est équivalent à $E \cup \{\neg A\} \models \bot$.

Démonstration:
a) supposons *E* |= A, alors de *E* on peut déduire A, mais de ¬A on peut déduire ¬A, donc de $E \cup \{\neg A\}$, on peut déduire $A \wedge \neg A$, c'est-à-dire: ⊥.
b) supposons $E \cup \{\neg A\} \models \bot$, alors d'après le théorème de la déduction, on a:
E |= (¬A ⇒⊥). Mais $(\neg A \Rightarrow \bot) \equiv \neg\neg A \equiv A$, donc *E* |= A.

4.1.4 Incohérence et réfutation

Définition: un ensemble de propositions *E* est dit **incohérent ou non satisfaisable** s'il n'existe aucun monde possible dans lequel toutes les propositions de *E* prennent simultanément la valeur vrai.

- *Théorème*: *E* est incohérent si et seulement si *E* |= ⊥.

Démonstration: si ⊥ ne se déduit pas de *E*, alors il existe un monde possible où toutes les propositions de *E* sont vraies et où ⊥ est faux, donc un monde possible où toutes les propositions de *E* sont vraies puisque ⊥ est toujours faux par définition! Donc, si *E* est incohérent, *E* |=⊥. Réciproquement, si *E* n'est pas incohérent, autrement dit est satisfaisable, alors il existe un monde possible où toutes les propositions de *E* sont vraies, et évidemment dans un tel monde, comme dans n'importe quel autre, ⊥ ne saurait être vrai, donc ⊥ ne peut pas se déduire de *E.*

4.1.5 La règle de résolution

- *Théorème*: Le schéma suivant est bien une règle d'inférence:
  $\{X \vee A, \neg X \vee B\} \models A \vee B$ (*règle de résolution*)

Démonstration:
De manière informelle, on peut dire la chose suivante: si on a X, alors on n'a pas ¬ X, donc ayant ¬ X ∨ B, on a nécessairement B, donc A ∨ B. De même si on a ¬ X, on n'a pas X, donc ayant X ∨ A, on a nécessairement A donc A ∨ B. Dans les deux cas on a ainsi A ∨ B, comme on est forcément dans un de ces deux cas..., on a bien nécessairement A ∨ B.
Ou bien aussi: noter que (¬ X ∨ B) ≡ (X ⇒ B) et que (A ∨ X) ≡ (¬ A ⇒ X), on a ainsi: {¬ A ⇒ X, X ⇒ B}, qui donne par règle du syllogisme: ¬ A ⇒ B, équivalent à A ∨ B.

- *Proposition*: les règles du modus ponens et du modus tollens sont des cas particuliers de la règle de résolution.

Démonstration: MP est la règle de résolution avec X remplacé par A et A par ⊥. MT est la règle de résolution avec X remplacé par B, A par ¬ A et B par ⊥. (Se rappeler que ⊥ est l'élément neutre de ∨).

- *Proposition*: la règle {X, ¬ X} |= ⊥ est un cas particulier de la règle de résolution.

(avec A et B remplacés par ⊥.)

4.1.6 Application de la règle de résolution à un ensemble de clauses

Définition : on appelle:

• **littéral**: une lettre (p, q, r, ...) ou une négation de lettre (¬p, ¬q, ¬r, ...)

• **clause**: une disjonction (ou « somme ») de littéraux (p∨q∨¬r, ¬p∨¬q∨¬r, p∨q, p, ¬p etc.)

Noter qu'une disjonction vide est, comme dans tous les cas similaires, l'élément neutre de la disjonction, à savoir: ⊥. C'est la raison pour laquelle on appellera souvent ⊥: la **clause vide**.

• Un **ensemble de clauses**, par exemple: {p∨q∨¬r, ¬p∨¬q∨¬r, p∨q, p, ¬p} est considéré comme la conjonction de toutes les clauses qu'il contient. (On prendra donc soin de ne pas confondre la clause vide, qui est ⊥, et l'ensemble vide de clauses, qui est T (le vrai) !), c'est aussi ce qu'on appelle une Forme Normale Conjonctive.

• **résolvante de deux clauses**: soit deux clauses, s_1 et s_2, telles que le littéral l figure dans l'une et le littéral ¬l figure dans l'autre (on écrira par exemple: $l \in s_1$, $\neg l \in s_2$), on appelle **résolvante de s1 et s2** la clause obtenue en supprimant le littéral l de l'une et le littéral ¬l de l'autre et en réunissant tout ce qui reste en une seule clause. Exemple: si $s_1 = \neg p \vee q \vee r$ et $s_2 = p \vee q \vee s$, alors $Res(s_1, s_2) = q \vee r \vee s$ (on supprimera aussi en même temps les littéraux redondants).

Méthode de résolution appliquée à un ensemble de clauses (chaînage avant): elle consiste à partir d'un ensemble de clauses et à appliquer la règle de résolution un certain nombre de fois, en enrichissant chaque fois l'ensemble de départ avec les déductions qui sont faites, jusqu'à obtenir la conclusion désirée.

Exemple : (en gras: les formules choisies à chaque étape pour appliquer la règle de résolution)

{p∨q∨r, ¬p∨q∨r, ¬q∨r} |= r

Ensemble de départ: {p∨q∨r, ¬p∨q∨r, ¬q∨r}
1er pas: {**p∨q∨r**, **¬p∨q∨r**, ¬q∨r, q∨r}
2eme pas : {p∨q∨r, ¬p∨q∨r, **¬q∨r**, **q∨r**, r}
stop! on a trouvé r.

Résolution + réfutation : partir d'un ensemble de clauses contenant la négation de la proposition à démontrer et appliquer la règle de résolution un certain nombre de fois, en enrichissant chaque fois l'ensemble de départ avec les déductions qui sont faites, jusqu'à obtenir ⊥ si possible.

Exemple:

Ensemble de départ: {p∨q∨r, ¬p∨q∨r, ¬q∨r, ¬r}
1er pas: {p∨q∨r, ¬p∨q∨r, **¬q∨r**, **¬r**, ¬q}
2eme pas : {**p∨q∨r**, **¬p∨q∨r**, ¬q∨r, ¬r, ¬q, q∨r}
3eme pas: {p∨q∨r, ¬p∨q∨r, ¬q∨r, ¬r, **¬q**, **q∨r**, r}
4eme pas: {p∨q∨r, ¬p∨q∨r, ¬q∨r, **¬r**, ¬q, q∨r, **r**, ⊥}
stop! on a trouvé ⊥.

On peut formuler ce qui précède sous forme d'un algorithme.

4.1.7 Algorithme de démonstration d'incohérence d'un ensemble de clauses:

Soit $\mathbb{S}$ l'ensemble de clauses
Tant que $\perp \notin \mathbb{S}$
- choisir l, s_1, s_2 tels que $l \in s_1$, $\neg l \in s_2$;
- calculer la résolvante r de s_1 et s_2;
- remplacer $\mathbb{S}$ par $\mathbb{S} \cup \{r\}$

S'il n'y a plus de nouveau choix possible et que $\perp \notin \mathbb{S}$, alors $\mathbb{S}$ est cohérent,
Si $\perp \in \mathbb{S}$, alors $\mathbb{S}$ est incohérent.

L'inconvénient de cet algorithme est son *non-déterminisme*. A chaque étape, un choix doit être fait et l'algorithme ne nous guide pas dans ce choix. Il doit donc être complété par une stratégie de choix explicite. Le problème est alors que *sa terminaison* (propriété selon laquelle l'algorithme arrive nécessairement à une fin) dépend de la stratégie choisie. Supposons par exemple que la stratégie de choix consiste à toujours prendre les deux premières clauses qui se présentent dans l'ensemble, puis à mettre la résolvante à la fin, l'algorithme va indéfiniment refaire le même mouvement et ajouter la même résolvante une infinité de fois! Il y a des stratégies astucieuses qui permettent d'éviter cet inconvénient, mais alors se présente un autre problème: le nombre de résolvantes à calculer peut être très élevé, c'est-à-dire de l'ordre de 2^n où n est le nombre d'occurrences de littéraux dans le problème. C'est le temps d'exécution (exponentiel en n) qui devient alors l'obstacle majeur.
Il peut sembler qu'existe aussi un autre inconvénient: il ne s'applique qu'aux propositions qui ont la forme de clauses. Mais ceci n'est pas un véritable inconvénient car, comme nous allons le voir, toute proposition peut se mettre sous forme de clause. Ce problème sera donc résolu. En revanche, le problème évoqué plus haut (celui de la complexité en temps de l'algorithme) ne sera résolu qu'en restreignant la forme des clauses admises. C'est dire que tous les problèmes de logique ne seront pas résolus (on dit que la méthode est *incomplète*).

4.2 Formes normales

4.2.1 Formes normales disjonctives et conjonctives

Définition : on appelle **Forme Normale Disjonctive** (FND) toute proposition qui est une disjonction de conjonctions de littéraux (ex: $(p \wedge q \wedge \neg r) \vee (\neg p \wedge q \wedge \neg r) \vee (\neg q \wedge r)$).

- *Proposition* : toute proposition P admet une FND qui lui est tautologiquement équivalente (c'est-à-dire une FND Q telle que $P \equiv Q$).

Démonstration: écrire la table de vérité de P. Partout où figure un "1", représenter le monde possible correspondant par une conjonction de littéraux (par exemple si 1 figure à la ligne où p est vrai, q est faux et r est vrai, associer à cette ligne la conjonction: $p \wedge \neg q \wedge r$). Faire ensuite la disjonction de toutes ces représentations de mondes possibles. On obtient une FND (la FND dite "canonique") qui, par construction, est équivalente à la proposition P. Noter que si P est la proposition $\perp$, la disjonction est vide, ce qui est cohérent.

Définition : on appelle **Forme Normale Conjonctive** (FNC) toute proposition qui est une conjonction de disjonctions de littéraux (ex: $(p \vee q \vee \neg r) \wedge (\neg p \vee q \vee \neg r) \wedge (\neg q \vee r)$).

La notion de FNC est évidemment identique à celle d'ensemble de clauses (puisque nous avons convenu de considérer un ensemble de clauses comme une conjonction de clauses).

- *Proposition* : toute proposition P admet une FNC qui lui est tautologiquement équivalente (c'est-à-dire une FNC Q telle que P ≡ Q).

Démonstration: d'après la proposition précédente, ¬P a une FND qui lui est équivalente:

$$\neg P \equiv (x_{11} \wedge x_{12} \wedge ...) \vee (x_{21} \wedge x_{22} \wedge ...) \vee .... \vee (x_{p1} \wedge x_{p2} \wedge ...)$$

où x_{11}, x_{12}, x_{21}, x_{22}, ..., x_{p1}, etc. sont des littéraux fabriqués à partir d'un stock de lettres donné. Comme $P \equiv \neg\neg P$, on a:

$$P \equiv \neg\{(x_{11} \wedge x_{12} \wedge ...) \vee (x_{21} \wedge x_{22} \wedge ...) \vee .... \vee (x_{p1} \wedge x_{p2} \wedge ...)\}$$

ce qui, d'après les règles de de Morgan, donne exactement une FNC.
Corollaire: toute proposition peut se mettre sous forme d'un ensemble de clauses.
En général, on n'utilise pas la table de vérité pour cela, mais on fait appel à un algorithme qui est le suivant:

1) Eliminer les $\Leftrightarrow$ et les $\Rightarrow$ (au moyen de lois telles que: $(p \Rightarrow q) \equiv (\neg p \vee q)$)
2) Faire porter les $\neg$ uniquement sur les lettres (par les règles de de Morgan)
3) Appliquer distributivité et lois d'absorption (ie le fait que : $a \wedge (a \vee b) \equiv a$ et $a \vee (a \wedge b) \equiv a$)

Exemple :
mettre sous FNC:

$$(p \Rightarrow (q \Leftrightarrow \neg r)) \wedge (r \Leftrightarrow \neg s))$$

1) equiv. à:

$$(\neg p \vee (q \Leftrightarrow \neg r)) \wedge (r \Rightarrow \neg s) \wedge (\neg s \Rightarrow r))$$
$$(\neg p \vee ((q \Rightarrow \neg r) \wedge (\neg r \Rightarrow q)) \wedge (r \Rightarrow \neg s) \wedge (\neg s \Rightarrow r))$$
$$(\neg p \vee ((\neg q \vee \neg r) \wedge (r \vee q)) \wedge (\neg r \vee \neg s) \wedge (s \vee r))$$

2) est inutile ici
3) noter que $((\neg q \vee \neg r) \wedge (r \vee q)) \equiv (\neg q \wedge r) \vee (\neg r \wedge q)$
on obtient donc:

$$(\neg p \vee (\neg q \wedge r) \vee (\neg r \wedge q)) \wedge (\neg r \vee \neg s) \wedge (s \vee r))$$

après distributivité et élimination de termes comme $\neg q \vee q$ (neutres pour $\wedge$)

$$(\neg p \vee \neg q \vee \neg r) \wedge (\neg p \vee r \vee q) \wedge (\neg r \vee \neg s) \wedge (s \vee r)$$

4.2.2 Clauses de Horn (facultatif)

On peut avoir un algorithme ayant de meilleures qualités en se restreignant à une famille de clauses particulières appelées clauses de Horn.
Définition : un littéral est dit positif s'il consiste en une lettre (sans négation), il est dit négatif dans le cas contraire.
Définition : on appelle **clause de Horn** toute clause qui possède au plus un littéral positif.
Exemples de clauses de Horn:

$$\neg p \vee \neg q \vee r$$
$$r$$
$$\neg p \vee \neg q \vee \neg r$$

Définition : on appellera **règle**, ou clause de Horn **stricte**, toute clause qui possède exactement un littéral positif, on appellera clause de Horn **négative** toute clause sans littéral positif, et on appellera **fait**, ou clause **unitaire positive** toute clause consistant en un littéral positif.

Algorithme de démonstration d'incohérence d'un ensemble de clauses de Horn:

Soit $\mathbb{S}$ l'ensemble de clauses
Tant que $\bot \notin \mathbb{S}$
- choisir p et c tels que:
  - p soit une clause unitaire positive de $\mathbb{S}$
  - c soit une clause de $\mathbb{S}$ contenant $\neg p$;
- calculer la résolvante r de p et c;
- remplacer $\mathbb{S}$ par $(\mathbb{S}-\{c\}) \cup \{r\}$

S'il n'y a plus de nouveau choix possible et que $\bot \notin \mathbb{S}$, alors $\mathbb{S}$ est consistant,
Si $\bot \in \mathbb{S}$, alors $\mathbb{S}$ est inconsistant.

On peut démontrer que cet algorithme se termine dans tous les cas, quelle que soit la stratégie de choix choisie. Ceci tient à ce que, comme on peut s'en rendre compte, à chaque pas une occurrence de littéral disparaît. Donc, comme on a au départ nécessairement un nombre fini de telles occurrences de littéraux, si à chaque pas, ce nombre diminue, on est sûr qu'il y aura un moment où l'algorithme s'arrête, ne serait-ce que parce que l'ensemble $\mathbb{S}$ est devenu vide !
Appliquons cet algorithme sur l'ensemble $\mathbb{S}$ suivant, en prenant pour stratégie de choix celle qui consiste à toujours prendre les deux premières clauses vérifiant les conditions données dans l'algorithme:

$\mathbb{S} = \{p \vee \neg r \vee \neg t,\ q,\ r,\ t \vee \neg p \vee \neg r,\ t \vee \neg q,\ \neg p \vee \neg q \vee \neg r\}$

1er pas : $\{\mathbf{p \vee \neg r \vee \neg t,}\ q,\ \mathbf{r},\ t \vee \neg p \vee \neg r,\ t \vee \neg q,\ \neg p \vee \neg q \vee \neg r\}$ donne:
$\{p \vee \neg t,\ q,\ r,\ t \vee \neg p \vee \neg r,\ t \vee \neg q,\ \neg p \vee \neg q \vee \neg r\}$

2eme pas : $\{p \vee \neg t,\ q,\ \mathbf{r},\ \mathbf{t \vee \neg p \vee \neg r},\ t \vee \neg q,\ \neg p \vee \neg q \vee \neg r\}$ donne:
$\{p \vee \neg t,\ q,\ r,\ t \vee \neg p,\ t \vee \neg q,\ \neg p \vee \neg q \vee \neg r\}$

3eme pas : $\{p \vee \neg t,\ \mathbf{q},\ r,\ t \vee \neg p,\ \mathbf{t \vee \neg q},\ \neg p \vee \neg q \vee \neg r\}$ donne:
$\{p \vee \neg t,\ q,\ r,\ t \vee \neg p,\ t\ ,\ \neg p \vee \neg q \vee \neg r\}$

4eme pas : $\{\mathbf{p \vee \neg t},\ q,\ r,\ t \vee \neg p,\ \mathbf{t}\ ,\ \neg p \vee \neg q \vee \neg r\}$ donne:
$\{p,\ q,\ r,\ t \vee \neg p,\ t\ ,\ \neg p \vee \neg q \vee \neg r\}$

5eme pas : $\{\mathbf{p},\ q,\ r,\ \mathbf{t \vee \neg p},\ t\ ,\ \neg p \vee \neg q \vee \neg r\}$ donne:
$\{p,\ q,\ r,\ t\ ,\ t\ ,\ \neg p \vee \neg q \vee \neg r\}$

par élimination des littéraux redondants:
$\{\mathbf{p},\ q,\ r,\ t\ ,\ \mathbf{\neg p \vee \neg q \vee \neg r}\}$, qui donne:

6eme pas : $\{p,\ q,\ r,\ t\ ,\ \neg q \vee \neg r\}$

7eme pas : $\{p,\ \mathbf{q},\ r,\ t\ ,\ \mathbf{\neg q \vee \neg r}\}$ donne:
$\{p,\ q,\ r,\ t\ ,\ \neg r\}$

8eme pas : $\{p,\ q,\ \mathbf{r},\ t\ ,\ \mathbf{\neg r}\}$ donne:
$\{p,\ q,\ r,\ t\ ,\ \bot\}$

stop !
On a bien prouvé que l'ensemble était inconsistant.
On remarquera sur cet exemple la décroissance stricte de l'ensemble $\mathbb{S}$, qui entraine la terminaison.

- **Proposition** : l'algorithme précédent est en temps polynômial, et plus précisément en $O(n^2)$ où n est le nombre d'occurrences de littéraux dans l'ensemble S de départ.

(NB: cela signifie que son temps d'exécution est fonction de la taille des données n, cette fonction étant un polynôme du second degré).

Démonstration: la recherche des deux clauses qui s'apparient, à un stade donné de l'algorithme se fait en un temps proportionnel à la longueur de la liste représentant l'ensemble S, cette longueur est toujours inférieure à n. Cette recherche se fait d'autre part un nombre de fois borné par n (puisqu'à chaque pas la longueur diminue de 1). Donc au total, le temps sera toujours borné par une fonction de l'ordre de n^2.
On peut d'autre part démontrer qu'il est complet relativement aux clauses de Horn (c'est-à-dire: quand il s'arrête parce qu'il a trouvé $\perp$, l'ensemble est vraiment inconsistant et quand il s'arrête parce qu'il n'y a plus de choix possible, l'ensemble S est vraiment consistant. Comme de plus il s'arrête nécessairement on est sžr que quand S est inconsistant, l'algorithme s'arrête avec $\perp \in S$ (car si c'était pour l'autre raison, on aurait : S consistant) et que quand il est consistant, l'algorithme s'arrête parce qu'il n'y a plus de choix possible). Mais il n'est pas complet par rapport à la logique propositionnelle dans son ensemble pour la simple raison que toutes les propositions ne peuvent pas se mettre sous forme de clauses de Horn.

Notation "Prolog":
Toute clause de Horn permet de distinguer deux ensembles de lettres: celle qui occurre positivement en elle (si elle existe) et celles qui occurrent négativement. On peut convenir de les mettre de part et d'autre du signe ":-" de sorte que la positive soit à gauche et les négatives à droite.
Exemple: l'ensemble précédent de clauses de Horn aurait la représentation suivante:

p :- r, t
q :-
r :-
t :- p, r
t :- q
:- p, q, r

Ceci ressemble beaucoup à un petit programme Prolog (sans variables). Le signe ":-" peut alors s'interpréter comme un "si". Une règle se lit: p **si** r **et** t. Un fait est une clause n'ayant qu'une partie gauche. Cela signifie que la vérité de cette partie gauche n'est conditionnée par rien. Une clause négative (appelée *requête* en Prolog) telle que la dernière a pour interprétation: FAUX **si** p **et** q **et** r. (Exercice: justifier ces interprétations!)

Exercices :

1- Démontrer par la Méthode de Résolution les relations de déduction suivantes:

$p \vee q \vee r \; ; \; \neg p \vee q \vee r \; ; \neg q \vee r \; \vdash \; r$

$p \vee q \; ; \; p \vee \neg q \; ; \; \neg p \vee q \; ; \; \neg p \vee \neg q \; \vdash \; \perp$

$\neg p \vee \neg q \; ; \; p \vee \neg r \; ; \; r \; ; \; q \vee \neg s \; ; \; s \vee q \; \vdash \perp$

Parmi les ensembles précédents de clauses, lesquels sont des ensembles de clauses de Horn ? Pourquoi ?

2- Démontrer que l'ensemble suivant de clauses est cohérent (on dit aussi satisfiable). Trouver une interprétation au moins qui le satisfait.

$C = \{p \vee q \vee r \; , \; p \vee \neg q \vee r \vee s \; , \; \neg p \vee q \vee s, \; p \vee \neg q \vee \neg r \vee s\}$

3- Faire la déduction suivante par la méthode de résolution :

$\{r \wedge t \Rightarrow p, \neg s \Rightarrow \neg p \vee \neg r, r, t, s \Rightarrow (t \Rightarrow q)\} \vdash q$

4. Soit E une expression booléenne (c'est-à-dire une formule propositionnelle qui ne contient que des $\neg$, des $\vee$ et des $\wedge$). On dit qu'une conjonction élémentaire P est un *implicant premier* de E si:

a) $P \vee E \equiv E$

(noter que cela correspond à : $P \models E$)

et b) il n'existe pas de conjonction élémentaire incluse dans P possédant cette propriété.

a) Démontrer que $P=a \wedge \neg c$ est un implicant premier de $E = (a \wedge \neg b) \vee (a \wedge b \wedge \neg c) \vee (\neg a \wedge b \wedge \neg c)$

b) Soit P_1 et P_2 deux conjonctions élémentaires telles que l'une contienne p_i et l'autre $\neg p_i$. On appelle *consensus* de P_1 et P_2 la conjonction élémentaire obtenue en prenant tous les littéraux qui figurent dans P_1 et P_2, une fois supprimés p_i et $\neg p_i$. Déterminer les consensus de:

(1) $p_0 \wedge p_1 \wedge \neg p_2 \wedge p_3$ et $p_0 \wedge \neg p_1 \wedge p_4$

(2) $p_0 \wedge \neg p_1$ et p_1

(3) $\neg p_0 \wedge p_1 \wedge p_2$ et $\neg p_0 \wedge p_1 \wedge p_3$

c) Démontrer que si Q est le consensus de P_1 et P_2, alors: $P_1 \vee P_2 \vee Q \equiv P_1 \vee P_2$

d) Montrer que dans une expression $E = P_1 \vee P_2 \vee ... \vee P_n$, où les P_i sont des conjonctions élémentaires, si un des P_i contient comme partie un autre P_j, alors on peut le supprimer de E.

e) Montrer qu'on peut aussi ajouter tout consensus Q de P_i et P_j qui n'inclut aucun des autres P_k.

L'application de ces deux règles s'appelle *méthode du consensus*. Par cette méthode, on peut arriver à exprimer toute expression booléenne comme la disjonction de ses implicants premiers (ce qui s'appelle la mettre sous forme de *somme réduite*).

f) Appliquer cette méthode à:

$E= (p_0 \wedge p_1 \wedge p_2) \vee (\neg p_0 \wedge \neg p_2) \vee (p_0 \wedge p_1 \wedge \neg p_2) \vee (\neg p_0 \wedge \neg p_1 \wedge p_2) \vee (\neg p_0 \wedge p_1 \wedge \neg p_2)$