

# LIFLC – Logique classique

## TD3 – Logique propositionnelle

Licence informatique UCBL – Automne 2017–2018

Les (parties d') exercices noté(e)s avec † sont plus difficiles.

### Exercice 1 : Variables d'une formules

1. Donner une définition de "l'ensemble des variables d'une formule".
2. Soient deux interprétations  $I_1$  et  $I_2$ . Montrer que si, pour toutes les variables  $p$  d'une formule  $A$   $I_1(p) = I_2(p)$ , alors  $eval(A, I_1) = eval(A, I_2)$ .
3. Soit  $A$  et  $B$  deux formules. Soit  $P_A$  et  $P_B$  leurs ensembles de variables respectifs. Si  $P_A$  et  $P_B$  sont disjoints, que peut-on dire sur la satisfiabilité de  $A \wedge B$  par rapport à celles de  $A$  et de  $B$  et pourquoi?
4. Peut-on faire la même déduction si  $P_A$  et  $P_B$  ne sont pas disjoints? Donner un exemple.

### Exercice 2 : Additionneur binaire

Un additionneur binaire est un circuit électronique permettant de réaliser des additions sur des entier positifs écrit en base 2. Un additionneur additionnant des nombres codés sur  $n$  bits possède  $2 \times n$  entrées et  $n + 1$  sorties (en effet, en binaire  $10 + 10 = 100$ ).

On souhaite vérifier un additionneur binaire en contrôlant ses sorties en fonction de ses entrées. L'additionneur est représenté par  $n + 1$  formules  $A_1, \dots, A_{n+1}$  ayant  $2n$  variables représentant les entrées du circuit et telle que valeur de vérité de  $A_k$  correspond à la valeur de la  $k^{ieme}$  sortie.

1. Donner la table de vérité de deux fonctions pour l'addition de 3 bits :
  - la première calcule la somme des 3 bits sans retenue (i.e.  $1 + 1 + 1 \mapsto 1$ );
  - la seconde calcule la retenue de cette somme.
2. Donner deux formules réalisant ces fonctions.
3. En déduire les formules spécifiant les sorties d'un additionneur 2 bits. On considère que les entier sont représentés avec le bit de poids faible ayant le plus petit indice, que le premier entier est représenté par  $p_1, p_2$  et que le second est représenté par  $q_1, q_2$ .
4. Généraliser la construction précédente pour donner une manière de construire les formules de spécification des sorties d'un additionneur  $n$  bits.
5. Expliquer comment on peut utiliser ces formules avec les formules  $A_1, \dots, A_{n+1}$  afin de vérifier que l'additionneur est correct.

## Corrections

### Solution de l'exercice 1

1. On définit cet ensemble par la fonction  $Var(A) : \mathcal{F} \rightarrow \mathcal{P}(\mathcal{V})$  définie récursivement par :

- $Var(p) = \{p\}$  si  $p$  est une variable propositionnelle
- $Var(\neg A) = Var(A)$
- $Var(A \diamond B) = Var(A) \cup Var(B)$  où  $\diamond$  peut-être  $\vee$ ,  $\wedge$  ou  $\Rightarrow$ .

2. On le montre par induction et par cas sur  $A$  :

$A = p \in \mathcal{V}$  :

$$eval(A, I_1) = I_1(p) = I_2(p) = eval(A, I_2).$$

$A = \neg B$  :

Hypothèse d'induction :  $eval(B, I_1) = eval(B, I_2)$

$$eval(\neg B, I_1) = non(eval(B, I_1)) = non(eval(B, I_2)) = eval(\neg B, I_2)$$

$A = B \vee C$  :

Hypothèse d'induction ( $\times 2$ ) :  $eval(B, I_1) = eval(B, I_2)$  et  $eval(C, I_1) = eval(C, I_2)$

$$eval(B \vee C, I_1) = ou(eval(B, I_1), eval(C, I_1))$$

$$= ou(eval(B, I_2), eval(C, I_2)) = eval(B \vee C, I_2)$$

$A = B \wedge C$  et  $A = B \Rightarrow C$  : similaire au cas précédent.

3.  $A \wedge B$  est satisfiable si et seulement si  $A$  et  $B$  sont satisfiables :

— Si  $A \wedge B$  est satisfiable, alors il existe une interprétation  $I$  telle que  $eval(A \wedge B, I) = 1$ .

Par ailleurs  $eval(A \wedge B, I) = et([A]_I, [B]_I)$ . D'après la table de vérité de *et* cela signifie que  $eval(A, I) = 1$  et  $eval(B, I) = 1$ . Donc  $A$  et  $B$  sont satisfiables.

— Si  $A$  et  $B$  sont satisfiables, il existe  $I_A$  telle que  $eval(A, I_A) = 1$  et il existe  $I_B$  telle que  $eval(B, I_B) = 1 = V$ . Soit l'interprétation  $I_{AB}$  définie comme suit :

—  $I_{AB}(p) = I_A(p)$  si  $p$  est une variable de  $A$

—  $I_{AB}(p) = I_B(p)$  si  $p$  n'est pas une variable de  $A$  (en particulier pour les variables de  $B$  car  $Var(A) \cap Var(B) = \emptyset$ ).

D'après la question précédente  $eval(A, I_{AB}) = eval(A, I_A) = 1$  et  $eval(B, I_{AB}) = eval(B, I_B) = 1$ . Donc  $eval(A \wedge B, I_{AB}) = et(eval(A, I_{AB}), eval(B, I_{AB})) = 1$ . Donc  $A \wedge B$  est satisfiable.

4. Si  $A$  et  $B$  sont satisfiables mais n'ont pas un ensemble de variables disjoints, on a pas forcément  $A \wedge B$  satisfiable : prendre  $A = p$  et  $B = \neg p$ .

### Solution de l'exercice 2

1.

p	q	r	somme	retenue
1	1	1	1	1
1	1	0	0	1
1	0	1	0	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

2. — somme :  $S = (p \wedge ((q \wedge r) \vee (\neg q \wedge \neg r))) \vee (\neg p \wedge ((\neg q \wedge r) \vee (q \wedge \neg r)))$   
 (qui est équivalent à  $p \text{ xor } q \text{ xor } r$ )  
 — retenue :  $R = (p \wedge q) \vee (q \wedge r) \vee (r \wedge p)$
3. —  $B_1 = (p_1 \wedge \neg q_1) \vee (\neg p_1 \wedge q_1)$  (obtenue à partir de la formule pour la somme en remplaçant  $r$  par faux et en propageant).  
 — On pose  $R_1 = p_1 \wedge q_1$  (obtenue à partir de la formule pour la retenue en remplaçant  $r$  par faux et en propageant). On a alors  $B_2 = S[p_2/p, q_2/q, R_1/r] =$   
 $(p_2 \wedge ((q_2 \wedge (p_1 \wedge q_1)) \vee (\neg q_2 \wedge \neg (p_1 \wedge q_1)))) \vee (\neg p_2 \wedge ((\neg q_2 \wedge (p_1 \wedge q_1)) \vee (q_2 \wedge \neg (p_1 \wedge q_1))))$   
 —  $B_3 = R[p_2/p, q_2/q, R_1/r] =$   
 $(p_2 \wedge q_2) \vee (q_2 \wedge p_1 \wedge q_1) \vee (p_1 \wedge q_1 \wedge p_2)$
4. On construit itérativement les formules  $B_k$  et  $R_k$  selon le schéma suivant :
- $B_1 = (p_1 \wedge \neg q_1) \vee (\neg p_1 \wedge q_1)$
  - $R_1 = p_1 \wedge q_1$
  - $B_k = S[p_k/p, q_k/q, R_{k-1}/r]$  pour  $2 \leq k \leq n$
  - $R_k = R[p_k/p, q_k/q, R_{k-1}/r]$  pour  $2 \leq k \leq n+1$
  - $B_{n+1} = R_{n+1}$
5. Il suffit de vérifier que  $A_i \equiv B_i$  pour  $1 \leq i \leq n+1$ .