1 Généralités

1)

v_1	v_2	v_3	a	b	c	d
0	0	0	0	0	1	0
0	0	1	0	0	1	1
0	1	0	0	0	1	1
0	1	1	0	1	1	1
1	0	0	1	1	1	1
1	0	1	1	0	1	1
1	1	0	0	0	1	1
1	1	1	1	0	1	1

2) a) $(v_1 \vee v_2) \vee \neg(v_2 \vee v_1)$ est une tautologie, car elle est de la équivalente à une formule de la forme "$p \vee \neg p$". Elle est donc satisfiable et ce n'est pas une antilogie.

b) $(v_1$ xor $v_2)$ xor $\neg v_1$ est satisfiable par exemple par la valuation $v_1 = 0$ et $v_2 = 1$. Ce n'est donc pas une antilogie. Ce n'est pas non plus une tautologie car elle n'est pas satisfaite par la valuation $v_1 = 1$ et $v_2 = 1$.

c) $v_1 \Rightarrow (v_2$ xor$(v_3 \vee \neg v_2))$ est satisfiable par exemple par toute valuation telle que $v_1 = 0$. Ce n'est donc pas une antilogie. Ce n'est pas non plus une tautologie car elle n'est pas satisfaite par la valuation $v_1 = 1$, $v_2 = 1$ et $v_3 = 1$.

d) $(p_1 \Rightarrow p_2) \Rightarrow \big((p_2 \Rightarrow p_3) \Rightarrow (p_1 \Rightarrow p_3)\big)$ est une tautologie. Une façon simple de le justifier est de faire sa table de vérité.

p_1	p_2	p_3	$p_1 \Rightarrow p_2$	$p_2 \Rightarrow p_3$	$p_1 \Rightarrow p_3$	$(p_1 \Rightarrow p_2) \Rightarrow ((p_2 \Rightarrow p_3) \Rightarrow (p_1 \Rightarrow p_3))$
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	1	0	1	1
0	1	1	1	1	1	1
1	0	0	0	1	0	1
1	0	1	0	1	1	1
1	1	0	1	0	0	1
1	1	1	1	1	1	1

3) Si p_1 est une tautologie, alors $p_1 \Rightarrow p_2$ est vraie pour une valuation donnée si et seulement cette valuation satisfait p_2. La formule $p_1 \Rightarrow p_2$ a donc la même table de vérité que p_2. Ainsi, si $p_1 \Rightarrow p_2$ est une tautologie, alors p_2 est également une tautologie (que des 1 dans la table de vérité).

4) a)
```
let rec longueur p = match p with
        |V -> 1
        |F -> 1
        |Var(n) -> 1
        |Non(q) -> (longueur q) + 1
        |Ou(q,r) -> (longueur q) + (longueur r) + 3
        |Et(q,r) -> (longueur q) + (longueur r) + 3;;
```

b) Commençons par l'étude de la suite (m_n).

- $m_1 = 1$ (longueur des constantes ou variable)
- soit $n \in \mathbb{N}*$, $m_{n+1} = m_n + 1$ (pour ajouter un étage, le plus économe en longueur est la négation)

En conséquence, pour tout $n \in \mathbb{N}*$, $m_n = n$.

Étudions maintenant la suite (M_n).

- $M_1 = 1$ (longueur des constantes ou variable)
- soit $n \in \mathbb{N}*$, $M_{n+1} = 2M_n + 3$ (pour ajouter un étage, le meilleur moyen de maximiser la longueur est la conjonction ou la disjonction et dans les deux cas ça ajoute 3)

La suite (M_n) est donc une suite arithmético-géométrique de terme général $M_n = 2^{n+2} - 3$.

5) a) $p|p \equiv \neg p \vee \neg p \equiv \neg p$

b) $(p_1|p_1)|(p_2|p_2) \equiv (\neg p_1|\neg p_2) \equiv (\neg(\neg p_1) \vee \neg(\neg p_2)) \equiv p_1 \vee p_2$

c) $(p_1|p_2)|(p_1|p_2) \equiv \neg(p_1|p_2) \equiv \neg(\neg p_1 \vee \neg p_2) \equiv \neg(\neg(p_1 \wedge p_2)) \equiv p_1 \wedge p_2$

d) Soit p une formule logique. On note $H(p)$ le prédicat « p est équivalente à une formule ne s'écrivant qu'avec les constantes, les variables propositionnelles et le connecteur de Sheffer ».

Montrons par induction structurelle que pour toute formule logique p, $H(p)$ est vraie.

- Cas de base : il est clair que $H(V)$ et $H(F)$ sont vraies. De même si v est une variable propositionnelle, $H(v)$ est vérifiée.
- Induction : soit p une formule logique.
  - On suppose que p est de la forme $p = \neg q$ où $H(q)$ est vérifiée. De ce fait, $q \equiv q_1$ où q_1 ne s'écrit qu'avec les constantes, les variables propositionnelles et le connecteur de Sheffer. On a alors
  $$p = \neg q \equiv q|q \equiv q_1|q_1$$
  Donc $H(p)$ est vraie.
  - On suppose que p est de la forme $p = q \vee q'$ où $H(q)$ et $H(q')$ sont vérifiées. De ce fait, $q \equiv q_1$ et $q' \equiv q'_1$ où q_1 et q'_1 ne s'écrivent qu'avec les constantes, les variables propositionnelles et le connecteur de Sheffer. On a alors
  $$p = q \vee q' \equiv (q|q)|(q'|q') \equiv (q_1|q_1)|(q'_1|q'_1)$$
  Donc $H(p)$ est vraie.
  - De même, si on suppose que p est de la forme $p = q \wedge q'$ où $H(q)$ et $H(q')$ sont vérifiées alors $H(p)$ est vérifiée
- Conclusion : pour tout formule logique p, $H(p)$ est vérifiée.

6) a) On peut écrire une fonction `antilogie : formule -> bool`.

```
let antilogie p = not(sat p);;
```

b) On peut écrire une fonction `tautologie : formule -> bool`.

```
let tautologie p = antilogie(Non(p));;
```

c) On peut écrire une fonction `semantiquementEgales : formule -> bool` telle que `semantiquementEgales p q` renvoie `true` si et seulement si `p` et `q` sont sémantiquement égales. On utilise pour cela que $p \equiv q$ si et seulement si $p \Leftrightarrow q$ est une tautologie où

$$p \Leftrightarrow q = (p \Rightarrow q) \wedge (q \Rightarrow p) = (q \vee \neg p) \wedge (p \vee \neg q)$$

```
let semantiquementEgales p q  =
     tautologie Et(Ou(q,Non(p)),Ou(p,Non(q)));;
```

7) a) Montrons que l'on peut simuler les connecteurs $\neg$ et $\vee$ avec le connecteur G.
- $G(p,p,p) = (\neg p \wedge p \wedge p) \vee (p \wedge \neg p) \vee (\neg p \wedge \neg p) \equiv F \vee F \vee \neg p \equiv \neg p$
- $G(p_1, G(p_1,p_1,p_1), p_2) \equiv G(p_1, \neg p_1, p_2) \equiv (\neg p_1 \wedge \neg p_1 \wedge p_2) \vee (p_1 \wedge \neg(\neg p_1)) \vee (\neg(\neg p_1) \wedge \neg p_2) \equiv (\neg p_1 \wedge p_2) \vee p_1 \vee (p_1 \wedge \neg p_2) \equiv (\neg p_1 \wedge p_2) \vee p_1 \equiv p_1 \vee p_2$
- Soit v une variable propositionnelle, $V \equiv v \vee \neg$. Donc V est équivalente à une formule ne s'écrivant qu'avec G et les variables propositionnelles.
- $F \equiv\neq V$. Donc F est équivalente à une formule ne s'écrivant qu'avec G et les variables propositionnelles.
- Soit p une formule de la forme $p = p_1 \wedge p_2$. D'après les loi de De Morgan, $p \equiv \neg(\neg p_1 \vee \neg p_2)$.

En mettant en place un raisonnement par induction structurelle similaire à celui de l'exercice 5, on montre que toute formule logique est équivalente à une formule ne s'écrivant qu'avec G et les variables propositionnelles.

b) Montrons que l'on peut simuler les connecteurs $\neg$ et $\wedge$ avec le connecteur $-$ et V.
- $V - p \equiv \neg p \wedge V \equiv \neg p$
- $p_2 - (V - p_1) \equiv p_2 - \neg p_1 \equiv \neg(\neg p_1) \wedge p_2 \equiv p_1 \wedge p_2$

Les connecteurs $\neg$ et $\wedge$ forment un système complet de connecteur ($\vee$ peut être simulé avec $\neg$ et $\wedge$), donc toutes les formules peuvent être écrites en utilisant uniquement les connecteurs $-$ et V.

c) Montrons qu'il n'existe pas de formule propositionnelle s'écrivant uniquement avec le connecteur $-$ et les variables propositionnelles qui soit équivalente à V.

Par l'absurde, supposons que p soit une formule s'écrivant uniquement avec le connecteur $-$ et les variables propositionnelles qui soit équivalente à V. On suppose de plus que p soit de hauteur minimale parmi toutes les formules logiques vérifiant cela. Comme p n'est pas réduite à une variable propositionnelle, on a $p = p_1 - p_2$. Comme p_2 s'écrit uniquement avec le connecteur $-$ et les variables propositionnelles et que p_2 est de hauteur strictement inférieure à p, la formule p_2 n'est pas équivalente à V (ce qui signifie que p_2 n'est pas une tautologie). Il existe donc une distribution de vérités μ telle que $\mathscr{E}_\mu(p_2) = 0$ où $\mathscr{E}_\mu$ est la fonction d'évaluation. On en déduit que $\mathscr{E}_\mu(p) = 0$ ce qui est absurde car $p \equiv V$ par hypothèse.

On a bien montré qu'il n'existe pas de formule propositionnelle s'écrivant uniquement avec le connecteur $-$ et les variables propositionnelles qui soit équivalente à V.

8) a) Soit une valuation μ de $(x_1, ..., x_n)$. Si μ satisfait la formule f, soit $\mu(x_i) = 1$ et μ satisfait le résidu f_{x_i}, soit $\mu(x_i) = 0$ et μ satisfait le résidu $f_{\overline{x_i}}$. Ainsi, $\mu \models f \Rightarrow \mu \models (x_i \wedge f_{x_i}) \vee (\overline{x_i} \wedge f_{\overline{x_i}})$.
Réciproquement, si $\mu(x_i) = 1$ et μ satisfait le résidu f_{x_i} ou si $\mu(x_i) = 0$ et μ satisfait le résidu $f_{\overline{x_i}}$, alors μ satisfait f. Ainsi, $\mu \models (x_i \wedge f_{x_i}) \vee (\overline{x_i} \wedge f_{\overline{x_i}}) \Rightarrow \mu \models f$.
En conclusion : $\mu \models f \Leftrightarrow \mu \models (x_i \wedge f_{x_i}) \vee (\overline{x_i} \wedge f_{\overline{x_i}})$.

b) Notons $f' = (x_i \wedge f_{x_i}) \vee (\overline{x_i} \wedge f_{\overline{x_i}})$.
$(x_i \vee f_{\overline{x_i}}) \wedge (\overline{x_i} \vee f_{x_i}) \equiv (x_i \wedge \overline{x_i}) \vee (f_{\overline{x_i}} \wedge \overline{x_i}) \vee (x_i \wedge f_{x_i}) \vee (f_{\overline{x_i}} \wedge f_{x_i})$
$(x_i \vee f_{\overline{x_i}}) \wedge (\overline{x_i} \vee f_{x_i}) \equiv F \vee f' \vee (f_{\overline{x_i}} \wedge f_{x_i})$
$(x_i \vee f_{\overline{x_i}}) \wedge (\overline{x_i} \vee f_{x_i}) \equiv f' \vee (f_{\overline{x_i}} \wedge f_{x_i})$

Or $(f_{\overline{x_i}} \wedge f_{x_i}) \equiv (f_{\overline{x_i}} \wedge f_{x_i} \wedge x_i) \vee (f_{\overline{x_i}} \wedge f_{x_i} \wedge \overline{x_i})$.
Donc $(f_{\overline{x_i}} \wedge f_{x_i}) \Rightarrow f'$, ainsi $(f_{\overline{x_i}} \wedge f_{x_i}) \vee f' \equiv f'$.
On a donc bien $(x_i \vee f_{\overline{x_i}}) \wedge (\overline{x_i} \vee f_{x_i}) \equiv f' \equiv f$.

c)

f_{x_i}	$f_{\overline{x_i}}$	$\frac{\partial f}{\partial x_i}$
0	0	0
0	1	1
1	0	1
1	1	0

Sur cette table de vérité, on voit que pour les deux lignes où $\frac{\partial f}{\partial x_i}$ vaut 1, la valuation dépend de la valeur de x_i (la valeur des résidus de f avec x_i et $\overline{x_i}$ sont opposés). Réciproquement, les deux lignes où $\frac{\partial f}{\partial x_i}$ vaut 0, la valuation dépend de la valeur de x_i (la valeur des résidus de f avec x_i et $\overline{x_i}$ reste la même).

d)

f_{x_i}	$f_{\overline{x_i}}$	$\frac{\partial f}{\partial x_i}$	$\overline{f}_{x_i}$	$\overline{f}_{\overline{x_i}}$	$\frac{\partial \overline{f}}{\partial x_i}$
0	0	0	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

Sur cette table de vérité étendue, on voit que le fait que les rôles du 0 et du 1 étant symétriques dans le calcul de la dérivée booléenne, $\frac{\partial f}{\partial x_i} = \frac{\partial \overline{f}}{\partial x_i}$.

e) Posons $C_1 = \left(f \wedge \frac{\partial g}{\partial x_i}\right)$, $C_2 = \left(g \wedge \frac{\partial f}{\partial x_i}\right)$, $C_3 = \left(\frac{\partial f}{\partial x_i} \wedge \frac{\partial g}{\partial x_i}\right)$ et $p = C_1 \oplus C_2 \oplus C_3$.
Dressons une table de vérité des deux formules pour vérifier l'égalité sémantique.

x_i	f_{x_i}	$f_{\overline{x_i}}$	g_{x_i}	$g_{\overline{x_i}}$	$\frac{\partial f \wedge g}{\partial x_i}$	C_1	C_2	C_3	p
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	1	1	1	1	1
0	0	1	1	0	0	1	0	1	0
0	0	1	1	1	1	0	1	0	1
0	1	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	1	1	0
0	1	0	1	0	1	0	0	1	1
0	1	0	1	1	1	0	1	0	1
0	1	1	0	0	0	0	0	0	0
0	1	1	0	1	1	1	0	0	1
0	1	1	1	0	1	1	0	0	1
0	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	1	0	0	1	1
0	0	1	1	0	0	0	1	1	0
0	0	1	1	1	1	0	1	0	1
0	1	0	0	0	0	0	0	0	0
0	1	0	0	1	0	1	0	1	0
0	1	0	1	0	1	1	1	1	1
0	1	0	1	1	1	0	1	0	1
0	1	1	0	0	0	0	0	0	0
0	1	1	0	1	1	1	0	0	1
0	1	1	1	0	1	1	0	0	1
0	1	1	1	1	0	0	0	0	0

2 Résolution d'une énigme par la logique des propositions

9) a) $A = OD \vee OG$ et $B = \neg OD$

b) $C = A \wedge B \vee \neg A \wedge \neg B$

c)

OD	OG	A	B
0	0	0	1
0	1	1	1
1	0	1	0
1	1	1	0

Aucune situation ne peut correspondre au fait que les deux sphynx mentent. Les sphynx disent donc tous les deux la vérité. La seule situation possible est donc la seconde ligne : la voie de droite se perd dans le désert et la voie de gauche mène à une oasis.

d) $C = A \wedge B \vee \neg A \wedge \neg B = (OD \vee OG) \wedge \neg OD \vee \neg(OD \vee OG) \wedge \neg\neg OD$
En utilisant les formules de De Morgan :
$C = OD \wedge \neg OD \vee OG \wedge \neg OD \vee \neg(OD \vee OG) \wedge OD$
$C = F \vee OG \wedge \neg OD \vee (\neg OD \wedge \neg OG) \wedge OD$
$C = OG \wedge \neg OD \vee \neg OD \wedge \neg OG \wedge OD$
$C = OG \wedge \neg OD \vee F$
$C = OG \wedge \neg OD$
On retrouve bien la même conclusion : la voie de droite se perd dans le désert et la voie de gauche mène à une oasis.

10) a) $R = A_1 \wedge \neg A_2 \wedge \neg A_3 \vee \neg A_1 \wedge A_2 \wedge \neg A_3 \vee \neg A_1 \wedge \neg A_2 \wedge A_3$

b) $A_1 = C \wedge \neg S$, $A_2 = \neg C \vee \neg S$ et $A_3 = \neg S$

c) $R = C\wedge\neg S\wedge\neg(\neg C\vee\neg S)\wedge\neg\neg S\vee\neg(C\wedge\neg S)\wedge(\neg C\vee\neg S)\wedge\neg\neg S\vee\neg(C\wedge\neg S)\wedge\neg(\neg C\vee\neg S)\wedge\neg S$
En utilisant les loi de De Morgan :
$R = C\wedge\neg S\wedge(\neg\neg C\wedge\neg\neg S)\wedge S\vee(\neg C\vee\neg\neg S)\wedge(\neg C\vee\neg S)\wedge S\vee(\neg C\vee\neg\neg S)\wedge(\neg\neg C\wedge\neg\neg S)\wedge\neg S$
$R = C \wedge \neg S \wedge C \wedge S \wedge S \vee (\neg C \vee S) \wedge (\neg C \vee \neg S) \wedge S \vee (\neg C \vee S) \wedge (C \wedge S) \wedge \neg S$
$R = F \vee (\neg C \vee S) \wedge (\neg C \wedge S \vee \neg S \wedge S) \vee F$
$R = (\neg C \vee S) \wedge (\neg C \wedge S \vee F)$
$R = (\neg C \vee S) \wedge \neg C \wedge S$
$R = \neg C \wedge \neg C \wedge S \vee S \wedge \neg C \wedge S$
$R = F \vee \neg C \wedge S$
$R = \neg C \wedge S$
En conclusion, il faut porter uniquement des vêtements sombres.

d) $A_1 = \neg R \vee B$, $A_2 = V \vee \neg B$ et $A_3 = B \vee ...$

e)

R	B	V	A_1	A_2	A_3
0	0	0	1	1	0
0	0	1	1	1	?
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	1	?
1	0	1	0	1	1
1	1	0	1	0	1
1	1	1	1	1	1

On a rempli la table simplement pour A_1 et A_2. Pour A_3 qui est incomplète, on a procédé comme suit :
- B à vrai suffit à ce que A_3 soit vrai.
- B à faux et les deux autres couleurs à faux implique A_3 à faux parce que la couleur manquante dans la consigne n'est pas portée.
- B à faux et les deux autres couleurs à vrai implique A_3 à vrai parce que la couleur manquante dans la consigne est pas portée.

Pour conclure, on cherche une ligne où un seul A_i est à vrai. C'est donc forcément la quatrième ligne et cela permet de savoir que :
- la couleur manquante dans la consigne est le vert
- il ne faut porter que du rouge.

3 Formes normales

11) a) i) On associe à toute distribution de vérité μ un minterme

$$p^\mu = v_1^{\mu(v_1)} \wedge \cdots \wedge v_n^{\mu(v_n)}$$

où $v_i^0 = \neg v_i$ et $v_i^1 = v_i$. Par construction,

$$\forall \nu \in \mathcal{B}^{\mathcal{V}}, \nu \vDash p^\mu \iff \nu = \mu.$$

Notons de plus que réciproquement, pour tout minterme q il existe une unique distribution de vérité μ tel que $q = p^\mu$. En effet $\mu \mapsto p^\mu$ est une application injective entre deux ensembles ayant 2^n éléments.

- **Existence** Voici deux preuves du fait que toute formule logique est équivalente à une disjonction de mintermes.
  Preuve par construction : Soit p une proposition logique. On considère $\mathcal{D}$ l'ensemble des distributions de vérité μ satisfiant p. Par construction

$$p \equiv \bigvee_{\mu \in \mathcal{D}} p^\mu.$$

  Cela signifie que pour déterminer la forme normale disjonctive, on peut (à la main), construire la table de vérité de p puis écrire p comme la disjonction des mintermes qui correspondent "aux lignes vérifiant p".
  Prenons l'exemple de $p = (v_1 \vee F) \wedge (v_3 \vee \neg v_2)$.

v_1	v_2	v_3	$v_1 \vee F$	$v_3 \vee \neg v_2$	p
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	0	0
1	1	1	1	1	1

  On a alors

$$p \equiv (v_1 \wedge \neg v_2 \wedge \neg v_3) \vee (v_1 \wedge \neg v_2 \wedge v_3) \vee (v_1 \wedge v_2 \wedge v_3).$$

  Preuve par induction :
  Soit p une formule logique on pose :
  $H(p)$: "*la formule p est équivalente à une disjonction de mintermes*".
  — La constante V est équivalente à la conjonction de tous les mintermes :

$$V \equiv \bigvee_{\mu \in \mathcal{B}^{\mathcal{V}}} p^\mu.$$

  — La constante F est équivalente à la conjonction d'aucun minterme :

$$F \equiv \bigvee_{\mu \in \emptyset} p^\mu.$$

— La variable v_i est équivalente à la conjonction de tous les mintermes associés aux distributions μ telles que $\mu(v_i) = 1$:

$$v_i \equiv \bigvee_{\substack{\mu \in \mathcal{B}^{\mathcal{V}} \\ \mu(v_i)=1}} p^\mu.$$

Soit p une formule logique :
— Si $p = \neg q$ avec q une formule vérifiant $H(q)$. On note alors

$$q \equiv \bigvee_{\mu \in \mathcal{D}} p^\mu$$

où $\mathcal{D}$ est une partie de $\mathcal{B}^{\mathcal{V}}$. En l'évaluant sur toutes les distributions, on voit que

$$p = \neg q \equiv \bigvee_{\mu \notin \mathcal{D}} p^\mu.$$

De ce fait $H(p)$ est vraie.
Soit p une formule logique :
— Si $p = q_1 \wedge q_2$ avec q_1, q_2 des formules vérifiant $H(q_1)$ et $H(q_2)$. On note alors

$$q_1 \equiv \bigvee_{\mu \in \mathcal{D}_1} p^\mu \text{ et } q_2 \equiv \bigvee_{\mu \in \mathcal{D}_2} p^\mu.$$

On a alors,

$$p = q_1 \wedge q_2 \equiv \bigvee_{\mu \in \mathcal{D}_1 \cap \mathcal{D}_2} p^\mu.$$

De ce fait $H(p)$ est vraie.
Soit p une formule logique :
— Si $p = q_1 \vee q_2$ avec q_1, q_2 des formules vérifiant $H(q_1)$ et $H(q_2)$. On note alors

$$q_1 \equiv \bigvee_{\mu \in \mathcal{D}_1} p^\mu \text{ et } q_2 \equiv \bigvee_{\mu \in \mathcal{D}_2} p^\mu.$$

On a de même,

$$p = q_1 \vee q_2 \equiv \bigvee_{\mu \in \mathcal{D}_1 \cup \mathcal{D}_2} p^\mu.$$

De ce fait $H(p)$ est vraie.
Par induction structurelle, on a bien montré que toute formule logique est équivalente à une disjonction de mintermes.

- **Unicité** Nous allons montrer que, pour toute formule logique p, à répétition près et à l'ordre des termes près, il existe une unique disjonction de mintermes qui soit équivalente à p.
Comme nous travaillons à l'ordre près et que nous n'autorisons pas de répétition de mintermes, une disjonction de mintermes q est uniquement caractérisée par **l'ensemble** $Z \subset \mathcal{B}^{\mathcal{V}}$ tel que $q = \bigvee_{\mu \in Z} p^\mu$. Maintenant, pour une proposition p, si on suppose que $p \equiv \bigvee_{\mu \in Z} p^\mu$ et $p \equiv \bigvee_{\mu \in Z'} p^\mu$ alors, pour toute distribution ν,

$$\nu \in Z \iff \nu \vDash p \iff \nu \in Z'.$$

Donc $Z = Z'$.

ii) Soit p une formule logique. On considère $\neg p$. D'après ce qui précède, il existe une unique forme normale disjonctive qui est équivalente à $\neg p$:

$$\neg p \equiv m_1 \vee \cdots \vee m_r$$

où $m_1, \ldots, m_r$ sont des mintermes. On en déduit par les formules de De Morgan que

$$p \equiv (\neg m_1) \wedge \cdots \wedge (\neg m_r).$$

Maintenant, pour un minterme

$$m = v_1^{\varepsilon_1} \wedge \cdots \wedge v_n^{\varepsilon_n}$$

où $\varepsilon_i \in \{0, 1\}$ et où $v_i^0 = \neg v_i$ et $v_i^1 = v_i$,

$$\neg m \equiv v_1^{1-\varepsilon_1} \vee \cdots \vee v_n^{1-\varepsilon_n}$$

qui est un maxterme.
On a bien obtenu p comme étant équivalent à une conjonction de maxtermes.
Pour $p = (v_1 \vee \neg v_2) \wedge (v_3 \vee \neg v_2)$.

v_1	v_2	v_3	$v_1 \vee \neg v_2$	$v_3 \vee \neg v_2$	p	$\neg p$
0	0	0	1	1	1	0
0	0	1	1	1	1	0
0	1	0	0	0	0	1
0	1	1	0	1	0	1
1	0	0	1	1	1	0
1	0	1	1	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	1	0

On a alors

$$\neg p \equiv (\neg v_1 \wedge v_2 \wedge \neg v_3) \vee (\neg v_1 \wedge v_2 \wedge v_3) \vee (v_1 \wedge v_2 \wedge \neg v_3).$$

Comme

$$\neg p \equiv (\neg v_1 \wedge v_2 \wedge \neg v_3) \vee (\neg v_1 \wedge v_2 \wedge v_3) \vee (v_1 \wedge v_2 \wedge \neg v_3).$$

on obtient

$$p \equiv (v_1 \vee \neg v_2 \vee v_3) \wedge (v_1 \vee \neg v_2 \vee \neg v_3) \wedge (\neg v_1 \vee \neg v_2 \vee v_3).$$

Remarques

- On utilise plus souvent les formes normales conjonctives, sans imposer que les conjonctions soient des maxtermes. On parle alors de **clauses**.
- En théorie, pour déterminer si deux propositions sont équivalentes, on peut comparer leur forme normale conjonctive. Cependant, quand on a une proposition sur n variables, sa forme normale conjonctive peut avoir 2^n clauses. On a donc, en général, une **explosion combinatoire** lors du calcul de la forme normale conjonctive.

b) Pour $p = (v_1 \wedge v_2) \iff (v_1 \vee v_3)$ on a

v_1	v_2	v_3	$v_1 \wedge v_2$	$v_1 \vee v_3$	p	$\neg p$
0	0	0	0	0	1	0
0	0	1	0	1	0	1
0	1	0	0	0	1	0
0	1	1	0	1	0	1
1	0	0	0	1	0	1
1	0	1	0	1	0	1
1	1	0	1	1	1	0
1	1	1	1	1	1	0

On en déduit la forme normale disjonctive

$$p \equiv (\neg v_1 \wedge \neg v_2 \wedge \neg v_3) \vee (\neg v_1 \wedge v_2 \wedge \neg v_3) \vee (v_1 \wedge v_2 \wedge \neg v_3) \vee (v_1 \wedge v_2 \wedge v_3).$$

De même

$$\neg p \equiv (\neg v_1 \wedge \neg v_2 \wedge v_3) \vee (\neg v_1 \wedge v_2 \wedge v_3) \vee (v_1 \wedge \neg v_2 \wedge \neg v_3) \vee (v_1 \wedge \neg v_2 \wedge v_3)$$

et donc la forme normale conjonctive

$$p \equiv (v_1 \vee v_2 \vee \neg v_3) \wedge (v_1 \vee \neg v_2 \vee \neg v_3) \wedge (\neg v_1 \vee v_2 \vee v_3) \wedge (\neg v_1 \vee v_2 \vee \neg v_3)$$

Si on suppose que $n = 4$ on remplace, dans la forme normale disjonctive

$$v_1^{\varepsilon_1} \wedge v_2^{\varepsilon_2} \wedge v_3^{\varepsilon_3}$$

par

$$(v_1^{\varepsilon_1} \wedge v_2^{\varepsilon_2} \wedge v_3^{\varepsilon_3} \wedge v_4) \vee (v_1^{\varepsilon_1} \wedge v_2^{\varepsilon_2} \wedge v_3^{\varepsilon_3} \wedge \neg v_4).$$

De même pour les formes normales conjonctives.

12) a)
```
let rec descend_negations p = match p with
    |Non(Non(q)) -> descend_negations q ;
    |Non(Et(q,r)) -> Ou(descend_negations (Non(q)),descend_negations (Non(r)))
    |Non(Ou(q,r)) -> Et(descend_negations (Non(q)),descend_negations (Non(r)))
    |Et(q,r) -> Et(descend_negations q,descend_negations r) ;
    |Ou(q,r) -> Ou(descend_negations q,descend_negations r) ;
    |_ -> p;;
```

b)
```
let rec descend_ou p = match p with
    |Et(q,r) -> Et(descend_ou q,descend_ou r)
    |Ou(q,Et(r1,r2)) -> Et(descend_ou (Ou(q,r1)),descend_ou (Ou(q,r2)))
    |Ou(Et(q1,q2),r) -> Et(descend_ou (Ou(q1,r)),descend_ou (Ou(q2,r)))
    |Ou(q,r) -> Ou(descend_ou q,descend_ou r)
    |Non(q) -> Non(descend_ou q)
    |_ -> p;;
```

Le nombre de connecteurs dans les formules en argument des appels récursifs diminue strictement et le cas de base où il n'y a plus de connecteur est traité par la dernière ligne.

c) Contrairement au cas des négations, en appliquant la fonction on n'obtient pas directement ce que l'on veut. En effet les $\wedge$ ne remontent qu'un peu à chaque fois. Par exemple pour

`p = Ou(Var(0),Ou(Var(1),Et(Var(2),Var(3))));;`

la fonction `descend_ou p` renvoie

`formule = Ou (Var (0), Et (Ou (Var (1), Var (2)), Ou (Var (1), Var (3))))`

Noter de plus que la taille de la formule augmente.

On va donc appliquer la fonction `descend_ou` jusqu'à l'obtention d'un point fixe. On écrit des fonctions `pfiter` et `pfrec` de type `'a -> ('a -> 'a) -> 'a = <fun>` qui prennent en paramètre une fonction `f`, une valeur de départ `x0` et qui itère la fonction jusqu'à l'obtention d'un point fixe (on suppose qu'il existe).

```
let pfiter x0 f =
    let z = ref x0 and fz = ref (f x0) in
    while !z <> !fz do
        z := !fz;
        fz := f !fz ;
    done ;
    !z ;;
```

Alternative récursive :

```
let pfrec x0 f =
    let rec iteration x fx =
        if x = fx then x else iteration fx ( f fx )
    in iteration x0 ( f x0 ) ;;

let et_ou_non p = pfrec (descend_negations p) descend_ou;;
```

d) On commence par écrire une fonction `construit_clause` qui construit la clause associée à une formule ne contenant pas de `Et`. On peut penser à :

```
let n=4 ;;

let construit_clause_fausse c =
    let t = Array.make n 0 in
    let rec parcours c  = match c with
        |Ou(p1,p2) -> parcours p1 ; parcours p2
        |Var(i) -> t.(i) <- 1
        |Non(Var(i)) -> t.(i) <- -1
        |_ -> ()
    in parcours c  ;
    t ;;
```

Cette fonction ne donne pas nécessairement le bon résultat si un littéral et son contraire est dans la clause (auquel cas il faut remplacer Ou((Var i), Non (Var i)) par V.

```
let construit_clause c =
    let t = Array.make n 0 in
```

```
let rec parcours c  =
    if t.(0) <> 2 then match c with
        |Ou(p1,p2) -> parcours p1 ; parcours p2
        |Var(i) -> t.(i) <- 1
        |Non(Var(i)) -> t.(i) <- -1
        |_ -> ()
in parcours c ;
t ;;
```

Il ne reste plus qu'à concaténer les formes clausales obtenues :

```
let rec fc p = match p with
    |Et(p1,p2) -> (fc p1) @ (fc p2)
    |_ -> [construit_clause p];;
```