Logique propositionnelle

Exercice 1
On définit les connecteurs NOR et $NAND$ par $PNORQ = \neg(P \vee Q)$ et $PNANDQ = \neg(P \wedge Q)$. On dit qu'un connecteur logique est complet si tout formule logique peut s'exprimer à l'aide des variables propositionnelles et de ce connecteur.
1. Montrer que NOR et $NAND$ sont des connecteurs logiques complets.
2. On note XOR le ou exclusif.
a. Proposer une définition de XOR.
b. Mettre XOR sous forme normale conjonctive et sous forme normale disjonctive.
c. Soit ϕ la distribution constante égale à 0 (autrement dit, la distribution attribuant le booléen faux à toutes les variables logiques). Montrer que, pour toute formule logique P n'utilisant que le connecteur XOR, $\phi(P) = 0$.
d. En déduire que XOR n'est pas un connecteur complet.

Exercice 2
Soient ϕ, ψ, φ des formules logiques. Montrer que $\varphi \to (\psi \to \phi)$, $(\varphi \to \psi) \wedge (\psi \to \phi)$, $(\neg\neg\psi) \to \psi$ sont des tautologies.

Exercice 3
Soient ψ, ϕ, φ des formules logiques. Déterminer si les équivalences suivantes sont vraies :
1. $(\psi \to \phi) \to \varphi \equiv \psi \to (\phi \to \varphi)$.
2. $(\psi \wedge \phi) \to \varphi \equiv \psi \to (\phi \to \varphi)$.

Exercice 4
Un peuple reclus dans une haute-vallée a la coutume suivante d'accueil des voyageurs étrangers : lorsque plusieurs membres de la tribu entament une conversation avec l'étranger, chaque membre ne dira que des vérités ou que des mensonges tout le temps de la conversation. Vous serez ensuite autorisé à séjourner parmi le peuple autochtone si vous arrivez à démêler le vrai du faux parmi leurs propos (notez que nous ne savons pas ce qui arrive à ceux qui n'arrive pas à comprendre qui ment et qui dit la vérité, personne n'ayant jamais pu témoigner dans ce cas de figure).
1. Lors d'une première conversation avec deux membre A et B, A affirme : "notre village se trouve dans la vallée" ; ce à quoi B rétorque "non, il ne s'y trouve pas" ; avant que A ne réplique "ou alors dans les collines".
Nous noterons V l'affirmation "le village est dans la vallée" et C l'affirmation "le village est dans les collines". Nous noterons A et B les affirmations des deux autochtones.
a. Ecrire une formule logique dépendant de A et B décrivant le comportement de vos interlocuteurs.
b. Ecrire des formules logiques décrivant les affirmations des individus.
c. En utilisant les propriétés des opérateurs booléens (notamment les lois de De Morgan), déterminer dans quelle région vous devez vous rendre pour rejoindre le village autochtone.
2. Après avoir survécu à la première conversation, vous rencontrez trois autres sauvages que nous prénommerons X, Y et Z à la croisée de plusieurs chemins. X affirme : "le chemin de gauche conduit au village" ; Y abonde : "tu as raison" ; X poursuit : "le chemin de droite y conduit aussi" ; Y vous lance de son air malicieux tout en se léchant les babines : "si le chemin du milieu y conduit, alors celui de droite n'y conduit pas" ; Z conclut : "le chemin du milieu n'y conduit pas."
On notera X, Y et Z les affirmations des trois hurluberlus, D (resp. M, G) les affirmations "le chemin de droite (resp. du milieu, de gauche) mène au village".
a. Décrire le comportement des individus rencontrés par une formule logique impliquant X, Y et Z.
b. Décrire les affirmations X, Y et Z à l'aide de D, M et G.
c. En utilisant une table de vérité, déterminer un chemin qui vous mènera au village.
d. A supposer que les trois participants aient menti, pouviez-vous prendre d'autres chemins ? Si oui, lesquels ?

Exercice 5
On considère F la formule logique $(p \to (q \to r)) \to (r \vee \neg p)$.
1. Dresser la table de vérité de F.
2. F est-elle une tautologie ? Une antilogie ? Est-elle satisfiable ?
3. Si F possède un modèle, en donner un.
4. Ecrire F sous forme normale conjonctive et sous forme normale disjonctive.

Exercice 6
Prouver les équivalences suivantes :
1. $\phi \vee \psi \equiv \psi \vee \phi$. 2. $\phi \wedge \psi \equiv \psi \wedge \phi$. 3. $\phi \wedge (\psi_1 \vee \psi_2) \equiv (\phi \wedge \psi_1) \vee (\phi \wedge \psi_2)$.

4. $\phi \vee (\psi_1 \wedge \psi_2) \equiv (\phi \vee \psi_1) \wedge (\phi \vee \psi_2)$. 5. $\top \wedge \phi \equiv \phi$. 6. $\top \vee \phi \equiv \top$.
7. $\bot \wedge \phi \equiv \bot$. 8. $\bot \vee \phi \equiv \phi$ 9. $\phi \wedge \phi \equiv \phi$.
10. $\phi \vee \phi \equiv \phi$. 11. $\phi \wedge \neg\phi \equiv \bot$ 12. $\phi \vee \neg\phi \equiv \top$.
13. $\phi \wedge (\psi \wedge \varphi) \equiv (\phi \wedge \psi) \wedge \varphi$. 14. $\phi \vee (\psi \vee \varphi) \equiv (\phi \vee \psi) \vee \varphi$. 15. $\neg\neg\phi \equiv \phi$.
16. $\neg(\phi \wedge \psi) \equiv \neg\phi \vee \neg\psi$. 17. $\neg(\phi \vee \psi) \equiv \neg\phi \wedge \neg\psi$. 18. $\phi \to \psi \equiv \neg\psi \to \neg\phi$.
19. $\phi \to (\varphi \to \psi) \equiv (\phi \wedge \varphi) \to \psi$.

Exercice 7 (Oral CCINP)
Si v est une variable logique, le littéral v est dit positif et $\neg v$ négatif. Une formule logique est une formule de Horn si elle est sous forme normale conjonctive dans laquelle chaque clause (éventuellement vide) contient au plus un littéral positif. On considérera également des clauses qui ne contiennent au plus qu'une seule fois chaque variable logique.
1. Rappeler la valeur de vérité de la clause vide.
2. Déterminer parmi les formules suivantes lesquelles sont des formules de Horn :
a. $F_1 = (\neg x_0 \vee \neg x_1 \vee \neg x_3) \wedge (x_0 \vee \neg x_1) \wedge (\neg x_2 \vee x_0 \vee \neg x_3) \wedge (\neg x_0 \vee \neg x_3 \vee x_2) \wedge x_2 \wedge (\neg x_3 \vee \neg x_2)$.
b. $F_2 = (x_0 \vee \neg x_1) \wedge (\neg x_2 \vee \neg x_3 \vee \neg x_0) \wedge \neg x_1 \wedge (x_1 \vee \neg x_1 \vee x_0) \wedge (\neg x_0 \vee x_2)$.
c. $F_3 = (\neg x_1 \vee \neg x_4) \wedge x_1 \wedge (\neg x_0 \vee \neg x_3 \vee \neg x_4) \wedge (x_0 \vee \neg x_1) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (x_4 \vee \neg x_0 \vee \neg x_1)$.
On considèrera que les variables logiques sont indexées par des entiers. On assimilera la variable x_i à l'entier i.
Pour implémenter les formules de Horn en OCaml, on définira un type `int option` qui sera `None` ou `Some i` pour un entier `i`. Une clause de Horn un type constituée d'un enregistrement qui sera de type `int` qui sera `None` si la clause ne contient aucun littéral positif ; une formule de Horn sera représentée par une liste de clauses de Horn ; une clause de Horn sera constitué d'un `int option` et d'une liste d'entiers ; le champ `int option` vaudra `None` en l'absence de littéral positif et `Some i` si cette clause contient le littéral positif x_i ; la liste d'entiers sera la liste des entiers correspondant aux littéraux négatifs.
Une formule de Horn sera une liste de clauses de Horn.

```
type int option = None | Some of int
type clause_horn = int option * int list
type formule_horn = clause_horn list
```

3. Ecrire une fonction `avoir_clause_vide : formule_horn -> bool` qui renvoie `true` si et seulement si la formule en entrée contient une clause vide.
On appelle clause unitaire une clause réduite à un littéral positif. On appellera propager une variable x_i dans une formule F sous forme normale conjonctive la modification suivante de F :
- toute clause de F qui ne contient ni x_i ni $\neg x_i$ n'est pas modifiée
- toute clause qui contient x_i est supprimée
- dans toute clause qui contient $\neg x_i$, on supprime $\neg x_i$.

On applique alors l'algorithme suivant à une formule de Horn F :
tant qu'il y a une clause unitaire x_i dans F faire
propager x_i dans F
renvoyer le booléen "F ne contient pas de clause vide".

On admet que cet algorithme permet de déterminer la satisfiabilité d'une formule logique.
4. Parmi les formules de Horn de la question 2, déterminer celle qui sont satisfiables.
5. Ecrire une fonction `trouver_clause_unitaire : formule_horn -> int option` renvoyant `None` si la formule en entrée n'a pas de clause unitaire et `Some i` si x_i est une clause unitaire de la formule en entrée.
6. Justifier que la propagation d'une variable dans une formule de Horn renvoie une formule de Horn. Ecrire une fonction `propager : formule_horn -> int -> formule_horn` qui prend en entrée une formule de Horn F et un entier i qui renvoie la propagation de x_i dans F.
7. Déduire des questions précédentes une fonction `etre_satisfiable : formule_horn -> bool` renvoyant la satisfiabilité d'une formule de Horn.
8. Déterminer la complexité de l'algorithme précédente en fonction de la taille de la formule de Horn. On appelle `HORN-SAT` le problème de la satisfiabilité d'une formule de Horn. Comparer les complexités des problèmes `SAT` et `HORN-SAT`.
9. Nous allons prouver la correction de l'algorithme précédent.
a. Si F est une clause de Horn sans clause unitaire ni clause vide, donner une valuation qui satisfait F.
b. Montrer que, si F est une formule de Horn qui possède la clause unitaire x_i et F' est sa propagation selon x_i, alors $F \equiv F'$.
c. Montrer la correction de l'algorithme précédent.
10. Modifier les fonctions précédentes pour déterminer une distribution de vérité qui est un modèle pour une formule de Horn satisfiable.

Exercice 8 (Oral ENS)
On considère un ensemble de variables propositionnelles $V = \{x_1, \cdots, x_n\}$ muni de l'ordre $x_i < x_j$ si et seulement si $i < j$.

Un arbre de décision sur V est un arbre binaire dont les feuilles sont étiquetées par 0 ou 1 et les noeuds internes par des variables logiques sur V. En outre, tout noeud interne x_i doit avoir des fils x_j tels que $x_i < x_j$.
Soit ϕ une distribution. Si T est un arbre de décision réduit à une feuille $x \in \{0,1\}$, on pose $\Phi_T(\phi) = x$. Si x_i est la racine de T, si $\phi(x) = 1$, si D est le sous-arbre droit de T, on pose $\Phi_T(\phi) = \Phi_D(\phi)$; si $\phi(x) = 0$, si G est le sous-arbre gauche de T, on pose $\Phi_T(\phi) = \Phi_G(\phi)$.
1. On considère $T = Noeud(x_1, Noeud(x_3, 1, 0), Noeud(x_2, 1, Noeud(x_3, 0, 1)))$ et $\phi : x_1 \mapsto 0$, $x_2 \mapsto 1$, $x_3 \mapsto 0$. Déterminer $\Phi_T(\phi)$. Donner une distribution pour laquelle Φ_T vaut 0.
2. On considère la formule logique $F = (x_1 \wedge x_2) \vee \neg(x_1 \wedge \neg x_3)$. Déterminer un arbre de décision T sur les variables $x_1 < x_2 < x_3$ telle que Φ_T a la même valeur de vérité que F.
3. Quels arbres de décision représentent des tautologies ? Quels arbres de décision représente des fonctions satisfiables ?
4. Soit un arbre de décision T de fonction associée Φ_T. Expliquer comment construire un arbre de décision T' tel que $\Phi_{T'}$ a même valeur de vérité que Φ_T.
On définit le type "formule logique" suivant

```
type formule =
   | Var of int
   | Et of formule * formule
   | Ou of formule * formule
   | Non of formule
```

On définit aussi le type "arbre de décision" suivant

```
type arbre = F of int | N of int * arbre * arbre
```

5. Ecrire une fonction `arb_vers_for : arbre -> formule` permettant d'associer à un arbre une formule dont la valeur de vérité est celle de la fonction booléenne associée à l'arbre de décision. Analyser sa complexité en temps et en espace.
6. Ecrire une fonction `conjonction : arbre -> arbre -> arbre` telle que, si T_1 et T_2 sont les arbres en argument, la fonction renvoie un arbre T telle que $\Phi_T \equiv \Phi_{T_1} \wedge \Phi_{T_2}$. Déterminer la complexité temporelle de la fonction et la taille de l'arbre construit.
7. Etant donné un arbre de décision T sur X, décrire un algorithme qui calcule combien de distributions satisfont la fonction Φ_T. Déterminer la complexité temporelle de votre algorithme.
8. Peut-on efficacement associer à une formule logique un arbre de décision représentant cette formule ?

Exercice 9 (Oral ENS)
Soit X un ensemble de variables qui seront notées en minuscules : x, y... Un littéral est une variable x ou sa négation $\neg x$. Une clause est une disjonction de littéraux. Une formule booléenne est en forme normale conjonctive (FNC) s'il s'agit d'une conjonction de clauses. On pourra appeler une telle formule une FNC. On pourra également appeler 3-FNC une FNC dont chaque clause contient au plus trois littéraux. Une valuation est une fonction de X dans $\{0,1\}$.
Le problème 3-SAT consiste à dire si une 3-FNC contenant n littéraux et m clauses est satisfiables. On supposera $m = O(n)$. On s'intéresse à des algorithmes efficaces pour résoudre ce problème, en $O(2^{\alpha n})$ où α est une constante.
1. Donner un algorithme en temps $O(2^n)$ pour résoudre 3-SAT.
Etant donnée une 3-FNC ϕ et une variable $x \in X$, on note $\phi|x$ la 3-FNC obtenue en évaluant x à 1 et $\phi|\neg x$ la 3-FNC obtenue en évaluant x à 0. On étend cette notion à un ensemble de variables. Par exemple, $\phi|xy$ est la formule ϕ où x et y s'évaluent en 1.
On admet que le polynôme $X^3 - X^2 - X - 1$ admet une unique racine réelle α et que $\alpha < 1,84$. On admet que la plus grande racine du polynôme $X^3 - 2X - 2$ vérifie $\beta < 1,77$.
2. Soit ϕ une formule logique et x une variable logique.
a. Montrer que ϕ est satisfiable si et seulement si $\phi|x$ ou $\phi|\neg x$ est satisfiable.
b. Supposons que $\neg x$ n'apparaît pas dans ϕ. Montrer que ϕ est satisfiable si et seulement si $\phi|x$ est satisfiable.
3. Soit ϕ une 3-FNC.
a. Soit $x \vee y \vee z$ une clause de ϕ. Montrer que ϕ est satisfiable si et seulement si $\phi|x$ ou $\phi|\neg xy$ ou $\phi|\neg x\neg yz$ est satisfiable.
b. En déduire un algorithme pour résoudre 3-SAT dont la complexité soit $O(1,84^n)$.
4. Soit ϕ une 3-FNC et x une variable logique telle que x et $\neg x$ apparaissent dans ϕ. En considérant une nouvelle disjonction de cas, donner un algorithme plus efficace pour 3-SAT.
On appelle distance de Hamming entre deux n-uplets de $\{0,1\}^n$ le nombre d'éléments où ils diffèrent.
On considère un nouvel algorithme récursif pour 3-SAT qui ne modifie par la 3-FNC ϕ mais qui modifie une valuation f. Voici sa description :
Local(ϕ,f,r)
Si f satisfait ϕ, renvoyer VRAI
Si $r = 0$, renvoyer FAUX
Soient x, y, z les variables de la première clause de ϕ non satisfaite par f
Poser : $\forall t \neq x, f_x(t) = f(t)$ et $f_x(t) - 1 - f(x)$, $\forall t \neq y, f_y(t) = f(t)$ et $f_y(y) = 1 - f(y)$, $\forall t \neq z, f_z(t) = f(t)$ et $f_z(z) = 1 - f(z)$

Si Local(ϕ, f_x,$r-1$), renvoyer VRAI
Si Local(ϕ, f_y,$r-1$), renvoyer VRAI
Renvoyer Local(ϕ,f_z,$r-1$).
5. a. Montrer que, s'il existe une valuation qui satisfait ϕ et qui est à distance de Hamming au plus r de f, alors Local(ϕ, f, r) renvoie VRAI.
b. Donner un algorithme en $O(3^{n/2})$ pour résoudre 3-SAT.
6. a. Soit $V(r)$ le nombre de n-uplets de $\{0,1\}^n$ à distance au plus r d'un n-uplet. Soi $t \in \{0,1\}^n$. Montrer qu'en choisissant $n2^n/V(r)$ n-uplets uniformément au hasard dans $\{0,1\}^n$, la probabilité qu'aucun ne soit à distance au plus r de t est bornée par e^{-n}.
b. En déduire un meilleur algorithme pour 3-SAT (avec une faible probabilité d'erreur).

Déduction naturelle

Exercice 10
Montrer que les séquents suivants sont valides dans la déduction naturelle :
1. $\vdash (p \vee q) \to (q \vee p)$ 2. $\vdash (p \vee (q \wedge r)) \to ((p \vee q) \wedge (p \vee r))$ 3. $\vdash ((p \to q) \wedge (p \to \neg q)) \to \neg p$.
4. $\vdash \neg(p \vee q) \to (\neg p \wedge \neg q)$ 5. $\vdash (\neg p \vee q) \to (p \to q)$ 6. $\vdash (p \to q) \to (\neg p \vee q)$.

Exercice 11
1. Montrer que $\vdash ((p \wedge q) \to r) \to (\neg p \vee q) \to (p \to r)$ est valide.
2. Déterminer si $\vdash ((p \wedge q) \to r) \wedge (\neg p \vee q) \to p$ est valide.

Exercice 12
Justifier la validité de toutes les règles de la déduction naturelle.

Exercice 13
Montrer la validité des séquents
1. $\vdash p \to (q \to p)$ 2. $\vdash p \to (p \to q) \to q$ 3. $\vdash \neg(p \wedge \neg p)$.

Exercice 14 (Oral CCINP)
1. Montrer que le séquent $\vdash \neg p \to (p \to \bot)$ est valide.
2. Montrer que le séquent $\vdash (p \to \bot) \to \neg p$ est valide.
3. Montrer que $\vdash (\neg p \to (p \to \bot)) \wedge ((p \to \bot) \to \neg p)$ est valide.
4. On considère la formule logique $F = ((p \to q) \to p) \to p$ (appelée règle de Peirce). Montrer F est une tautologie. 5. A l'aide de la déduction naturel, montrer la validité de $\vdash F$.

Exercice 15
1. Ecrire, en français, une preuve de la tautologie $\neg(p \vee q) \to (\neg p \wedge \neg q)$.
2. Ecrire en déduction naturelle une preuve de $\vdash \neg(p \vee q) \to (\neg p \wedge \neg q)$.

Exercice 16
On se place dans le système logique de la déduction naturelle. On appelle théorème d'affaiblissement le théorème suivant : si $\Delta \vdash A$ est un séquent valide, alors, pour toute formule B, $\Delta, B \vdash A$ est valide.
1. Rappeler la définition inductive des arbres de preuves.
2. Donner les arbres de preuve à une seule règle.
3. Prouver le théorème d'affaiblissement pour un séquent conclusion d'un arbre de preuve à une seule règle.
4. Montrer le théorème d'affaiblissement pour la déduction naturelle par induction structurelle sur l'ensemble des arbres de preuve.

Exercice 17 (Oral IMT)
1. Prouver le séquent $A \wedge B \vdash B \wedge A$.
2. Prouver le séquent $A \wedge (B \vee C) \vdash (A \wedge B) \vee (A \wedge C)$.
3. Prouver le séquent $\vdash A \vee \neg A$.
4. Soit $\Gamma \vdash C$ un séquent prouvable à l'aide d'un arbre de preuve. Montrer que $\Gamma \vdash C$ est conclusion d'un arbre de preuve qui n'utilise pas la succession de la règle d'élimination de la conjonction et de l'introduction de la conjonction.

Exercice 18 (partie d'Oral ENS)
Donner des preuves des séquent :
1. $\vdash p \leftrightarrow \neg\neg p$ 2. $\vdash \neg(p \vee q) \leftrightarrow (\neg p \wedge \neg q)$
3. $\vdash (\neg p \vee q) \leftrightarrow (p \to q)$ 4. $\vdash p \vee \neg p$.