Cette *trentième* colle vous fera travailler sur la suite du cours de logique.
C'est aussi la dernière planche de colle de l'année 2021/2022 !

Conséquences logiques

Soit P, P', Q et Q' des formules quelconques.

1. Rappeler ce que signifie la notation $P \equiv P'$, en termes de valuations $\phi : \mathcal{V} \to \mathbb{B}$, et leurs fonctions d'évaluation $\overline{\phi}$ associées.
2. Si on suppose avoir $P \equiv P'$ et $Q \equiv Q'$, montrer que $P \wedge Q \equiv P' \wedge Q'$.
3. Soient x et y deux variables. Prouver que $(x \vee y) \wedge z \equiv (x \wedge z) \vee (y \wedge z)$.
   En déduire que pour toutes formules P, Q et R, on a $(P \vee Q) \wedge R \equiv (P \wedge R) \vee (Q \wedge R)$.
   Comment appelle-t-on cette propriété ?

Autres connecteurs logiques

4. Donner la table de vérité des connecteurs usuels $x \vee y$ et $x \wedge y$.
5. Donner la table de vérité des connecteurs $x \to y$ ("x implique y") et $x \leftarrow y$ ("x est impliqué par y").
6. Donner la table de vérité des connecteurs NAND, NOR et XOR.
7. Écrire ces connecteurs en fonction des connecteurs de base ($\neg$ unaire, $\vee$ et $\wedge$ binaires).

Fonctions booléennes associées à une formule

Soit $P = x \vee \neg(\neg y \wedge z)$.

8. Donner la fonction booléenne $\varphi_P : \mathbb{B}^n \to \mathbb{B}$ associée à la formule P.

A l'inverse, on peut chercher à trouver une formule qui est associée à une fonction booléenne donnée. Soit la fonction booléenne $f : \mathbb{B}^n \to \mathbb{B}$ suivante, renseignée par sa table de vérité :

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

9. Proposer une formule P_f telle que $\varphi_{P_f} = f$.

Littéraux, min-termes, max-termes et clauses

10. Rappeler la définition d'un littéral. Donner des exemples.
11. Rappeler la définition d'un min-terme et d'un max-terme. Donner des exemples.
12. Rappeler la définition d'une clause. Donner des exemples.
13. Pour les formules suivantes P_i, lister les variables $\mathcal{V}(P_i)$, les littéraux ℓ_j et les clauses C_k qui y apparaissent.
    - $P_1 = \neg(x \vee (y \wedge \neg\neg z))$
    - $P_2 = (\neg x) \wedge (\neg y \vee z)$
    - $P_3 = (\neg a \vee \neg b \vee c) \wedge (a \vee b \vee \neg c)$
    - $P_4 =$ `beau` $\vee$ `pluie` $\vee$ `neige`

Formes normales conjonctives (CNF) et disjonctives (DNF)

14. Rappeler la définition d'une formule en forme normale conjonctive.
15. Que signifie qu'une CNF (ou une DNF) est canonique ?
16. Pour les formules P_i de la Q13, dire si elles sont en CNF, en DNF (ce n'est pas exclusif), et canonique ou non.

Mise en CNF ou en DNF

17. Mettre en forme CNF (non nécessairement canonique) la formule P_1 de la Q13.
18. Mettre en forme DNF (non nécessairement canonique) la formule P_2 de la Q13.

Implémentation en OCaml

19. Donner un type récursif `formule` qui permet de représenter les formules de la logique propositionnelle, avec les connecteur usuels ($\neg$ unaire, $\vee$ et $\wedge$ binaires). On représentera les variables par un constructeur `Var of int`, tel que $Var(i)$ représente une variable x_i pour $i \in \mathbb{N}$.

On souhaite pouvoir transformer n'importe quelle formule P en une formule équivalente P', telle que l'opérateur de négation $\neg$ soit appliqué uniquement à des constantes $\perp$ ou $\top$ ou des variables x_i.

Par exemple, on peut transformer la formule $P_5 = \neg(x \vee (y \wedge z))$ en $P_5' = (\neg x) \wedge (\neg y \vee \neg z)$.

20. Expliquer un algorithme récursif (par induction structurelle sur la formule P) qui permet de réaliser cette transformation.
21. L'implémenter en une fonction (récursive) OCaml `propagation_negation : formule -> formule`.
22. Tester la sur l'exemple de la formule P_5, et vérifier le résultat de l'exemple.
23. Que peut-on dire sur la hauteur $h(P')$ et la taille $t(P')$, en fonction de $h(P)$ et de $t(P)$?