L2/LD – TP 8 – Forme normale négative et forme normale conjonctive

Ce TP doit être fini avant la prochaine séance de TP, éventuellement sur votre temps de travail personnel.

Vous devez rendre sous la forme d'un unique fichier source compressé au format .zip avant 23h30 la veille de votre prochain TP l'ensemble des fonctions demandées dans ce TP

Objectif L'objectif de ce TP est de continuer d'implanter en Caml des règles de simplification de formules de la logique des propositions.

1 Introduction

Afin d'écrire plus facilement des algorithmes sur les formules propositionnelles, on les transforme parfois en formules équivalentes, mais plus simples, appelées *formes normales*.

Nous étudions dans ce TP, la forme normale négative et la forme normale conjonctive.

2 Forme normale négative

Une formule est dite en *forme normale négative* lorsque toutes les négations s'appliquent à un atome.

1. Ecrire une fonction qui vérifie qu'une formule est en forme normale négative.

   À taper : `verifFNN (Neg(Vrai));;`
   Réponse : `- : bool = false`
   À taper : `verifFNN (Et(Neg(Atome "D"), Faux));;`
   Réponse : `- : bool = true`
   À taper : `verifFNN (Neg(Neg(Atome "D")));;`
   Réponse : `- : bool = false`
   À taper : `verifFNN (Neg(Et(Atome "D", Atome "P")));;`
   Réponse : `- : bool = false`

2. Utiliser

$$\begin{aligned} \neg\textsc{Faux} &\Leftrightarrow \textsc{Vrai} && (1)\\ \neg\textsc{Vrai} &\Leftrightarrow \textsc{Faux} && (2)\\ \neg\neg F &\Leftrightarrow F && (9)\\ \neg(F \wedge G) &\Leftrightarrow (\neg F) \vee (\neg G) && (10)\\ \neg(F \vee G) &\Leftrightarrow (\neg F) \wedge (\neg G) && (11) \end{aligned}$$

   pour définir une fonction qui met toute formule en forme normale négative.

```
À taper :   formeNN(Neg(Vrai));;
Réponse :   - : pf = Faux
À taper :   # formeNN(Et(Neg(Atome "D"), Faux));;
Réponse :   - : pf = Et (Neg (Atome "D"), Faux)
À taper :   formeNN(Neg(Neg(Atome "D")));;
Réponse :   - : pf = Atome "D"
À taper :   formeNN(Neg(Et(Atome "D", Atome "P")));;
Réponse :   - : pf = Ou (Neg (Atome "D"), Neg (Atome "P"))
```

3. On considère la formule propositionnelle $f5 = P \wedge \neg(\textsc{Faux} \vee \neg(\neg Q \wedge \textsc{Vrai}))$. Tester ces deux fonctions sur $f5$.

```
À taper :   affichePF f5;;
Réponse :   P ^ Non(Faux v Non(Non(Q) ^ Vrai))- : unit = ()
À taper :   verifFNN f5 ;;
Réponse :   - : bool = false
À taper :   affichePF (formeNN f5);;
Réponse :   P ^ (Vrai ^ (Non(Q) ^ Vrai))- : unit = ()
```

3 Composition de simplifications

On peut composer les fonctions précédentes pour obtenir encore plus de simplifications.

1. Ecrire une fonction qui compose l'idempotence, la mise en forme normale négative et l'élimination des constantes de façon optimale.
2. Tester cette fonction de composition, l'idempotence, la mise en forme normale négative et l'élimination des constantes sur la forumle $f6 = (\textsc{Vrai} \wedge (\neg Q \wedge \textsc{Vrai})) \wedge \neg(\textsc{Faux} \vee \neg(\neg Q \wedge \textsc{Vrai}))$

On considère désormais cette fonction pour mettre une formule en forme normale négative.

4 Forme normale conjonctive

Une formule est dite en forme normale conjonctive (CNF, pour *Conjunctive Normal Form*) quand elle est de la forme

$$D_1 \wedge \ldots \wedge D_n$$

où chaque D_i est une disjonction de littéraux.

Un *littéral* est soit une proposition atomique, soit la négation d'une proposition atomique.

Ecrire une fonction `isCNF` qui retourne *vrai* si une formule est sous forme CNF.

On considère la formule propositionnelle $f7 = (P \vee Q) \wedge ((P \vee \neg R) \wedge (P \vee (Q \vee R)))$

```
À taper :   affichePF f7;;
Réponse :   (P v Q) ^ ((P v Non(R)) ^ (P v (Q v R)))- : unit = ()
À taper :   isCNF f7;;
Réponse :   - : bool = true
```

5 Mise en forme normale conjonctive

La première étape pour mettre une formule en forme normale consiste à mettre la formule sous forme normale négative.

Pour que les disjonctions ne portent plus que sur des littéraux, la deuxième étape consiste à appliquer systématiquement, de gauche à droite, l'axiome

$$A \vee (B \wedge C) \quad \equiv \quad (A \vee B) \wedge (A \vee C) \qquad \text{(Axiome)}$$

qui distribue les disjonctions ($\vee$) sur les conjonctions ($\wedge$).

Dans ce but, coder d'abord une fonction `distrib : pf -> pf` qui applique cet axiome chaque fois qu'un $\vee$ précède un $\wedge$.

On considère la formule propositionnelle $f8 = (P \vee ((\neg R \vee (Q \wedge R)) \wedge Q)$

```
À taper :   affichePF f8;;
Réponse :   P v ((Non(R) v (Q ^ R)) ^ Q)- : unit = ()
À taper :   affichePF (distrib f8);;
Réponse :   (P v ((Non(R) v Q) ^ (Non(R) v R))) ^ (P v Q)- : unit = ()
```

En utilisant la fonction `distrib`, écrire une fonction `formCNF` qui permet de faire la mise en forme CNF.

Reprendre les formules propositionnelles déjà définies et déterminer leur forme normale conjonctive :

- $f = P \wedge (\text{Faux} \vee (\text{Vrai} \wedge Q))$
- $f1 = (\text{Vrai} \wedge \text{Faux}) \vee (\text{Vrai} \wedge (\text{Faux} \wedge ((\text{Vrai} \wedge \text{Vrai}) \wedge \text{Faux})))$
- $f2 = (((\text{P} \wedge \text{Faux}) \wedge (\text{Faux} \vee \text{Vrai})) \wedge ((\text{P} \vee \text{Vrai}) \vee (\text{Vrai} \wedge \text{Q}))) \wedge \text{Vrai}$
- $f3 = \text{Faux} \vee ((\text{R} \wedge \text{Vrai}) \wedge \text{P}$
- $f4 = (\text{Faux} \vee (\text{Vrai} \wedge Q)) \wedge ((\text{Faux} \vee (\text{Vrai} \wedge Q)) \wedge (\text{Faux} \vee (\text{Vrai} \wedge Q)))$
- $f5 = P \wedge \neg(\text{Faux} \vee \neg(\neg Q \wedge \text{Vrai}))$
- $f6 = (\text{Vrai} \wedge (\neg Q \wedge \text{Vrai})) \wedge \neg(\text{Faux} \vee \neg(\neg Q \wedge \text{Vrai}))$
- $f7 = (P \vee Q) \wedge ((P \vee \neg R) \wedge (P \vee (Q \vee R)))$
- $f8 = (P \vee ((\neg R \vee (Q \wedge R)) \wedge Q)$

```
À taper :   affichePF (formCNF (f));;
Réponse :   P ^ Q- : unit = ()
À taper :   affichePF (formCNF (f1));;
Réponse :   Faux- : unit = ()
À taper :   affichePF (formCNF (f2));;
Réponse :   Faux- : unit = ()
À taper :   affichePF (formCNF (f3));;
Réponse :   R ^ P- : unit = ()
À taper :   affichePF (formCNF (f4));;
Réponse :   Q- : unit = ()
À taper :   affichePF (formCNF (f5));;
Réponse :   P ^ Non(Q)- : unit = ()
À taper :   affichePF (formCNF (f6));;
Réponse :   Non(Q)- : unit = ()
À taper :   affichePF (formCNF (f7));;
Réponse :   (P v Q) ^ ((P v Non(R)) ^ (P v (Q v R)))- : unit = ()
À taper :   affichePF (formCNF (f8));;
Réponse :   (P v ((Non(R) v Q) ^ (Non(R) v R))) ^ (P v Q)- : unit = ()
```