TD #21 — Logique (avec solutions)

Exercice 1. *Syntaxe & sémantique d'expressions arithmétiques*

1. Définissez une syntaxe des *expressions arithmétiques* sur un ensemble de variable $\mathcal{V}$ avec deux constructeurs $+$ et $\times$. (Vous pouvez définir cette syntaxe directement sous la forme d'un `type ae` OCaml.)

   On propose :
   ```
   type ae =
       | X of int
       | Add of ae * ae
       | Mul of ae * ae
   ```

2. Définissez la sémantique usuelle pour les expressions arithmétiques entières, sous la forme d'une fonction OCaml de type `ae -> ··· -> int`, où l'on assimilera les valeurs de type `int` aux entiers, et où l'on ignorera tout problème lié à la finitude de ces valeurs.

   On propose :
   ```
   let rec eval_nat v
       = function
       | X i -> v.(i)
       | Add (e1, e2)
             -> (eval_nat v e1) + (eval_nat v e2)
       | Mul (e1, e2)
             -> (eval_nat v e1) * (eval_nat v e2)
   ```
   Où le premier argument représente une valuation pour les variables apparaissant dans l'expression.

3. De même pour les expressions arithmétiques sur $\mathbb{F}_{257}$, le corps fini à 257 éléments, où l'on assimilera les valeurs de type `int` entre `0` et `256` aux éléments de $\mathbb{F}_{257}$, et ignorera tout problème lié au dépassement de capacité lors de calcul avec des valeurs de type `int`.

   On propose :
   ```
   let rec eval_f257 v
       = function
       | X i -> v.(i)
       | Add (e1, e2)
             -> ((eval_nat v e1) + (eval_nat v e2)) mod 257
       | Mul (e1, e2)
             -> ((eval_nat v e1) * (eval_nat v e2)) mod 257
   ```

Exercice 2. *Quelques équivalences sémantiques classiques*

En utilisant des tables de vérité, montrez les équivalences sémantiques suivantes pour la sémantique par tables de vérité de la logique propositionnelle :

1. $\neg\varphi \equiv \varphi \to \bot$
2. $\neg(\varphi_1 \wedge \varphi_2) \equiv \neg\varphi_1 \vee \neg\varphi_2$
3. $\neg(\varphi_1 \vee \varphi_2) \equiv \neg\varphi_1 \wedge \neg\varphi_2$
4. $\neg\neg\varphi \equiv \varphi$
5. $\varphi_1 \to \varphi_2 \equiv \neg\varphi_2 \to \neg\varphi_1$
6. $\varphi_1 \vee (\varphi_2 \wedge \varphi_3) \equiv (\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3)$
7. $\varphi_1 \wedge (\varphi_2 \vee \varphi_3) \equiv (\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \varphi_3)$

où φ, φ_1, φ_2 désignent des formules quelconques sur un ensemble de variable arbitraire.

Pour la sémantique par tables de vérité de la logique propositionnelle, l'on a que quelque soit l'ensemble de variable $\mathcal{V}$ sur lequel φ etc. est définie, pour toute valuation σ, l'évaluation $[\![\varphi]\!]_\sigma$ est par construction ou bien vraie (1), ou bien fausse (0). Pour montrer les équivalences sémantiques ci-dessus, il est donc suffisant de considérer les ensembles des

valuations pour lesquelles (par exemple) $[\![\varphi_1]\!]_\sigma = 0$, $[\![\varphi_2]\!]_\sigma = 0$, pour lesquelles $[\![\varphi_1]\!]_\sigma = 1$, $[\![\varphi_2]\!]_\sigma = 0$, etc. Autrement dit, on peut considérer toutes les formules comme des variables et montrer les équivalences en considérant toutes leurs valuations.

Exercice 3. *Quelques tautologies classiques*

Montrez les tautologies suivantes pour la sémantique par tables de vérité de la logique propositionnelle :

1. $\models \neg(\varphi \wedge \neg\varphi)$
2. $\models \varphi \vee \neg\varphi$
3. $\models \bot \rightarrow \varphi$
4. $\models (\neg\varphi \rightarrow \varphi) \rightarrow \varphi$

où φ etc.

Exercice 4. *Quelques propriétés des modèles pour la sémantique classique*

En utilisant la définition alternative de la sémantique par tables de vérité vue en cours, montrez que, :

1. $\mathbb{M}(\neg\varphi) = \mathbb{V} \setminus \mathbb{M}(\varphi)$

   On a $[\![\neg\varphi]\!]_\sigma := 1 - [\![\varphi]\!]_\sigma$, d'où :

   $$\mathbb{M}(\neg\varphi) := \{\sigma \in \mathbb{V} \mid [\![\neg\varphi]\!]_\sigma = 1\} = \{\sigma \in \mathbb{V} \mid 1 - [\![\varphi]\!]_\sigma = 1\} = \{\sigma \in \mathbb{V} \mid [\![\varphi]\!]_\sigma = 0\} = \mathbb{V} \setminus \mathbb{M}(\varphi)$$

2. $\mathbb{M}(\varphi_1 \wedge \varphi_2) = \mathbb{M}(\varphi_1) \cap \mathbb{M}(\varphi_2)$

   On a $[\![\varphi_1 \wedge \varphi_2]\!]_\sigma := \min([\![\varphi_1]\!]_\sigma, [\![\varphi_2]\!]_\sigma)$, d'où :

   $\mathbb{M}(\varphi_1 \wedge \varphi_2) := \{\sigma \in \mathbb{V} \mid [\![\varphi_1 \wedge \varphi_2]\!]_\sigma = 1\} = \{\sigma \in \mathbb{V} \mid \min([\![\varphi_1]\!]_\sigma, [\![\varphi_2]\!]_\sigma) = 1\} = \{\sigma \in \mathbb{V} \mid [\![\varphi_1]\!]_\sigma = [\![\varphi_2]\!]_\sigma = 1\} = \mathbb{M}(\varphi_1) \cap \mathbb{M}(\varphi_2)$

3. $\mathbb{M}(\varphi_1 \vee \varphi_2) = \mathbb{M}(\varphi_1) \cup \mathbb{M}(\varphi_2)$

   On a $[\![\varphi_1 \vee \varphi_2]\!]_\sigma := \max([\![\varphi_1]\!]_\sigma, [\![\varphi_2]\!]_\sigma)$, d'où :

   $$\mathbb{M}(\varphi_1 \vee \varphi_2) := \{\sigma \in \mathbb{V} \mid [\![\varphi_1 \vee \varphi_2]\!]_\sigma = 1\} = \{\sigma \in \mathbb{V} \mid \max([\![\varphi_1]\!]_\sigma, [\![\varphi_2]\!]_\sigma) = 1\}$$

   Notons $\mathcal{S}$ ce dernier ensemble, alors soit $\sigma \in \mathcal{S}$ l'on a $[\![\varphi_1]\!]_\sigma = 1$ ou $[\![\varphi_2]\!]_\sigma = 1$, d'où $\mathcal{S} \subseteq \mathbb{M}(\varphi_1) \cup \mathbb{M}(\varphi_2)$. Réciproquement, si $\sigma \in \mathbb{M}(\varphi_1) \cup \mathbb{M}(\varphi_2)$ alors $[\![\varphi_1]\!]_\sigma = 1$ ou $[\![\varphi_2]\!]_\sigma = 1$, d'où $\mathcal{S} = \mathbb{M}(\varphi_1) \cup \mathbb{M}(\varphi_2)$ par double inclusion.

où φ etc.

Exercice 5. *Quelques conséquences sémantiques*

Montrez les conséquences sémantiques suivantes (ou non) pour la sémantique par tables de vérité de la logique propositionnelle :

On utilise les propriétés sur les modèles admises en cours ou démontrées à l'exercice précédent.

1. $\varphi_1 \rightarrow \varphi_2 \not\models \varphi_2$

   On a $\mathbb{M}(\varphi_1 \rightarrow \varphi_2) = (\mathbb{V} \setminus \mathbb{M}(\varphi_1)) \cup \mathbb{M}(\varphi_2)$, qui n'est pas inclus dans $\mathbb{M}(\varphi_2)$ en général (ce n'est par exemple pas le cas si $\mathbb{M}(\varphi_1) = \mathbb{M}(\varphi_2) = \varnothing$).

2. $\{\varphi_1, \varphi_1 \rightarrow \varphi_2\} \models \varphi_2$

   $$\mathbb{M}(\varphi_1) \cap ((\mathbb{V} \setminus \mathbb{M}(\varphi_1)) \cup \mathbb{M}(\varphi_2)) = (\mathbb{M}(\varphi_1) \cap (\mathbb{V} \setminus \mathbb{M}(\varphi_1)) \cup (\mathbb{M}(\varphi_1) \cap \mathbb{M}(\varphi_2)) = \mathbb{M}(\varphi_1) \cap \mathbb{M}(\varphi_2) \subseteq \mathbb{M}(\varphi_2)$$

3. $\{\varphi_1, \neg\varphi_1\} \models \varphi_2$

   $$\mathbb{M}(\varphi_1) \cap (\mathbb{V} \setminus \mathbb{M}(\varphi_1)) = \varnothing \subseteq \mathbb{M}(\varphi_2)$$

   pour toute formule φ_2.

où φ_1 etc.

Exercice 6. *Universalité de $\bar{\wedge}$ pour la sémantique T.V.*

1. Montrez que le connecteur binaire infixe $\bar{\wedge}$ (« non-et », ou « NAND ») dont la table de vérité pour la sémantique par tables de vérité est :

— $a \bar{\wedge} b = 0$ si $a = b = 1$, et 1 sinon ;

est *universel* pour la sémantique par tables de vérité de la logique propositionnelle. C'est à dire, montrez que toute formule propositionnelle s'écrivant avec les connecteurs $\neg$, $\wedge$, $\vee$, $\rightarrow$ est sémantiquement équivalente (pour la sémantique T.V. usuelle de ces derniers) à une formule utilisant uniquement le connecteur $\bar{\wedge}$.

Remarque. On peut montrer la même propriété pour le connecteur binaire infixe $\bar{\vee}$ (« non-ou », ou « NOR ») dont la table de vérité est $a \bar{\vee} b = 1$ si $a = b = 0$, et 0 sinon. L'universalité de ces connecteurs peut être intéressante dans le cadre de l'*électronique des circuits* : il suffit de disposer de *portes logiques* NAND ou NOR pour pouvoir implémenter l'évaluation de n'importe quelle formule propositionnelle pour la sémantique T.V. (dans ce cadre, on parlerait plutôt de *circuits* ou *fonctions booléennes*).

Exercice 7. *Additionneur 2-bit*

1. Donnez trois formules φ_0, φ_1, φ_2 définies sur les variables x_0, x_1, y_0, y_1 telles que pour x & y deux entiers naturels $\in [\![0, 3]\!]$ et σ une affectation pour laquelle x_i (resp. y_i) vaut vrai ssi. le i-ème bit de x (resp. y) vaut 1 dans son écriture en base deux, l'on a toujours $[\![\varphi_i]\!]_\sigma$ valant vrai ssi. le i-ème bit de l'écriture en base deux de $x + y$ vaut 1.

Il est pratique d'exprimer (certaines de) ces formules en termes d'un nouveau connecteur infixe $\oplus$ (« ou exclusif » ou « XOR »), dont la sémantique par table de vérité est définie comme $a \oplus b = 1$ ssi. $a \neq b$. (Autrement dit, $\varphi_1 \oplus \varphi_2 \equiv (\varphi_1 \wedge \neg\varphi_2) \vee (\neg\varphi_1 \wedge \varphi_2)$.)
En abusant les notations et notant z_0, z_1, z_2 les chiffres de l'écriture en base deux sur trois bits de $x + y$, on peut alors remarquer que $z_0 = x_0 \oplus y_0$, $z_1 = x_1 \oplus y_1 \oplus c_0$ avec c_0 valant 1 si une retenue est propagée par l'addition de x_0 et y_0 et 0 sinon, et $z_2 = c_1$ avec c_1 valant 1 si une retenue est propagée par l'addition de x_1, y_1 et c_0, et 0 sinon. Il suffit alors de remarquer que $c_0 \equiv x_0 \wedge y_0$ et $c_1 \equiv (x_1 \wedge y_1) \vee ((x_0 \wedge y_0) \wedge (x_1 \vee y_1))$ (par exemple), ce qui donne (par exemple) :

— $\varphi_0 = x_0 \oplus y_0$
— $\varphi_1 = (x_1 \oplus y_1) \oplus (x_0 \wedge y_0)$
— $\varphi_2 = (x_1 \wedge y_1) \vee ((x_0 \wedge y_0) \wedge (x_1 \vee y_1))$

Ce que l'on peut par exemple vérifier en calculant leurs tables de vérité.