Correction TD N°3 : Types inductifs

Exercice 1 (Fonctions et preuves inductives sur les entiers)

1. Écrire la fonction *mult* sur les entiers naturels $\mathcal{N}$.
2. Démontrer que : $\forall n \in \mathcal{N}.\ mult(2, n) = plus(n, n)$.
3. Démontrer que : $\forall n \in \mathcal{N}.\ mult(n, 2) = plus(n, n)$.

1. La fonction *mult* est de type $\mathcal{N} \to \mathcal{N} \to \mathcal{N}$ et peut être définie de la façon suivante :

   ($mult_1$) Pour tout $n \in \mathcal{N}$, $mult(0, n) = 0$

   ($mult_2$) Pour tout $p, n \in \mathcal{N}$, $mult(S\ p, n) = plus(n, mult(p, n))$

2. On rappelle la définition de la fonction *plus* qui peut être définie de la façon suivante :

   ($plus_1$) Pour tout $n \in \mathcal{N}$, $plus(0, n) = n$

   ($plus_2$) Pour tout $p, n \in \mathcal{N}$, $plus(S\ p, n) = S\ plus(p, n)$

   On cherche à prouver que $mult(2, n) = plus(n, n)$.

$$\begin{aligned} mult(2, n) &= plus(n, mult(1, n)) && (mult_2) \\ &= plus(n, plus(n, mult(0, n))) && (mult_2) \\ &= plus(n, plus(n, 0)) && (mult_1) \\ &= plus(n, n) && (plus_1) \end{aligned}$$

   On a bien démontré en utilisant les règles de constructionde *plus* et *mult* que $mult(2, n) = plus(n, n)$.

3. On cherche maintenant à prouver que $mult(n, 2) = plus(n, n)$.
   Pour ce faire, on peut utiliser plusieurs méthodes, la plus générique étant de prouver par induction structurelle la propriété.
   On rappelle le schéma d'induction structurelle sur $\mathcal{N}$ qui est le suivant :

$$\forall P \in \mathcal{N} \to Prop.\ P(0) \Rightarrow (\forall n \in \mathcal{N}.\ P(n) \Rightarrow P(S\ n)) \Rightarrow \forall n \in \mathcal{N}.\ P(n)$$

   Dans notre cas, notre proposition est la suivante :

$$P(n) = mult(n, 2) = plus(n, n)$$

   Pour prouver P nous allons avoir besoin d'une autre propriété de *plus* que nous allons formuler avec le lemme suivant :

Lemme 1. $\forall n, m \in \mathcal{N}.\ S\ plus(n, m) = plus(n, S\ m)$

   Soit $Q(n)$ la propriété représentant cette formule, prouvons Q par induction structurelle.

Base Prouvons la propriété vraie pour le cas $Q(0)$.

$$\begin{aligned} &\forall m \in \mathcal{N}.\ S\ plus(0, m) = S\ m && (plus_1) \\ &\forall m \in \mathcal{N}.\ plus(0, S\ m) = S\ m && (plus_1) \end{aligned}$$

par reflexivité, $Q(0)$ est donc vraie.

Induction Supposons $Q(n)$ et prouvons $Q(S\ n)$.

Nous avons donc l'hypothèse d'induction suivante :

$$\forall m \in \mathcal{N}.\ S\ plus(n, m) = plus(n, S\ m)$$

$$\begin{aligned} \forall m \in \mathcal{N}.\ S\ plus(S\ n, m) &= \forall m \in \mathcal{N}.\ S\ S\ plus(n, m) && (plus_2) \\ &= \forall m \in \mathcal{N}.\ S\ plus(n, S\ m) && \text{(hypothèse d'induction)} \\ &= \forall m \in \mathcal{N}.\ plus(S\ n, S\ m) && (plus_2) \end{aligned}$$

On a bien $Q(n) \Rightarrow Q(S\ n)$.

Conclusion $\forall n \in \mathcal{N}.\ Q(n)$

Grâce au **Lemme 1** nous allons maintenant pouvoir prouver P par induction structurelle.

Base Prouvons la propriété vraie pour le cas $P(0)$.

$$\begin{aligned} &mult(0, 2) = 0 && (mult_1) \\ &plus(n, n) = 0 && (plus_1) \end{aligned}$$

$P(0)$ est donc vraie.

Induction Supposons $P(n)$ et prouvons $P(S\ n)$.

Nous avons donc l'hypothèse d'induction suivante :

$$mult(n, 2) = plus(n, n)$$

$$\begin{aligned} mult(S\ n, 2) &= plus(2, mult(n, 2)) && (mult_2) \\ &= plus(2, plus(n, n)) && \text{(hypothèse d'induction)} \\ &= S\ plus(1, plus(n, n)) && (plus_2) \\ &= S\ S\ plus(0, plus(n, n)) && (plus_2) \\ &= S\ S\ plus(n, n) && (plus_1) \\ &= S\ plus(S\ n, n) && (plus_2) \\ &= plus(S\ n, S\ n) && \text{(lemme 1)} \end{aligned}$$

Et donc $P(n) \Rightarrow P(S\ n)$.

Conclusion $\forall n \in \mathcal{N}.\ P(n)$

Ainsi, $\forall n \in \mathcal{N}.\ mult(n, 2) = plus(n, n)$.

Exercice 2 (Fonctions et preuves inductives sur les listes)

1. Écrire la fonction rev qui inverse les éléments d'une liste.
2. Démontrer que : $\forall l \in \mathcal{L}.\ \forall e \in \mathcal{A}.\ rev(app(l, [e])) = e :: rev(l)$.
3. Démontrer que : $\forall n \in \mathcal{L}.\ rev(rev(l)) = l$.

1. La fonction rev est de type $\mathcal{L} \to \mathcal{L}$ et peut être définie de la façon suivante :

   (rev_1) $rev(nil) = nil$

   (rev_2) Pour tout $l \in \mathcal{L}$, pour tout $e \in \mathcal{A}$, $rev(e :: l) = app(rev(l), [e])$

2. On rappelle la définition de la fonction app qui peut être définie de la façon suivante :

   (app_1) Pour tout $l \in \mathcal{L}$, $app(nil, l) = l$

   (app_2) Pour tout $l_1, l_2 \in \mathcal{L}$, pour tout $e \in \mathcal{A}$, $app(e :: l_1, l_2) = e :: app(l1, l_2)$

   On cherche à prouver que $\forall l \in \mathcal{L}.\ \forall e \in \mathcal{A}.\ rev(app(l, [e])) = e :: rev(l)$.
   On va utiliser le schéma d'induction structurelle sur $\mathcal{L}$ pour prouver la propriété. On rappelle le schéma qui est le suivant :

$$\forall P \in \mathcal{L} \to Prop.\ P(nil) \Rightarrow (\forall l \in \mathcal{L}.\ \forall e \in \mathcal{A}.\ P(l) \Rightarrow P(e :: l)) \Rightarrow \forall l \in \mathcal{L}.\ P(l)$$

   Dans notre cas la proposition est la suivante :

$$P(l) = \forall e \in \mathcal{A}.\ rev(app(l, [e])) = e :: rev(l)$$

Base Prouvons $P(nil)$.

$$\begin{aligned}
\forall e \in \mathcal{A}.\ rev(app(nil, [e])) &= rev([e]) = rev(e :: nil) && (app_1)\\
&= app(rev(nil), [e]) && (rev_2)\\
&= app(nil, [e]) && (rev_1)\\
&= [e] = e :: nil && (app_1)\\
&= e :: rev(nil) && (rev_1)
\end{aligned}$$

$P(0)$ est donc vraie.

Induction Supposons $P(l)$, prouvons $\forall e_1 \in \mathcal{A}.\ P(e_1 :: l)$

Nous avons donc l'hypothèse d'induction suivante :

$$\forall e \in \mathcal{A}.\ rev(app(l, [e])) = e :: rev(l)$$

$$\begin{aligned}
\forall e_1, e \in \mathcal{A}.\ rev(app(e_1 :: l, [e])) &= rev(e_1 :: app(l, [e])) && (app_2)\\
&= app(rev(app(l, [e])), [e_1]) && (rev_2)\\
&= app(e :: rev(l), [e_1]) && \text{(hypothèse d'induction)}\\
&= e :: app(rev(l), [e_1]) && (app_2)\\
&= e :: rev(e_1 :: l) && (rev_2)
\end{aligned}$$

Et donc $P(l) \Rightarrow \forall e_1 \in \mathcal{A}.\ P(e :: l)$.

Conclusion $\forall l \in \mathcal{L}.\ P(l)$.

Ainsi $\forall l \in \mathcal{L}.\ \forall e \in \mathcal{A}.\ rev(app(l, [e])) = e :: rev(l)$.

3. On cherche à prouver que $\forall l \in \mathcal{L}.\ rev(rev(l)) = l$.
   Une nouvelle fois, nous allons le prouver par induction structurelle avec la propriété P suivante :

$$P(l) = rev(rev(l)) = l$$

Base Prouvons $P(nil)$:

$$\begin{aligned} rev(rev(nil)) &= rev(nil) && (rev_1) \\ &= nil && (rev_1) \end{aligned}$$

$P(nil)$ est vraie.

Induction Supposons $P(l)$, prouvons $\forall e \in \mathcal{A}.\ P(e :: l)$

Nous avons donc l'hypothèse d'induction suivante :

$$rev(rev(l)) = l$$

$$\begin{aligned} \forall e \in \mathcal{A}.\ rev(rev(e :: l)) &= rev(app(rev(l), [e])) && (rev_2) \\ &= e :: rev(rev(l)) && \text{(preuve exo 2.2 avec } l = rev(l)) \\ &= e :: l && \text{(hypothèse d'induction)} \end{aligned}$$

Et donc $P(l) \Rightarrow \forall e_1 \in \mathcal{A}.\ P(e :: l)$.

Conclusion $\forall l \in \mathcal{L}.\ P(l)$.

On a bien $\forall l \in \mathcal{L}.\ rev(rev(l)) = l$.

Exercice 3 (Type inductif des formules en logique)

1. Définir le type des formules en logique propositionnelle.
2. Écrire la fonction *sub*, qui rend l'ensemble des sous-formules d'une formule F.
3. Écrire la fonction *nbc*, qui rend le nombre de connecteur d'une formule F.
4. Écrire le schéma d'induction structurelle des formules.
5. Démontrer que : $|sub(F)| \leq 2 \times nbc(F) + 1$, pour toute formule F.

1. On peut définir les formules en logique propositionnelle comme le plus petit ensemble $\mathcal{F}$ vérifiant les propriétés suivantes (on considère l'ensemble $\mathcal{S}$ comme étant l'ensemble des symboles propositionnels) :

   ($prop_1$) Pour tout $s \in \mathcal{S} \cup \{\top, \bot\}$, $s \in \mathcal{F}$
   ($prop_2$) Pour tout $F \in \mathcal{F}$, $\neg F \in \mathcal{F}$
   ($prop_3$) Pour tout $F_1, F_2 \in \mathcal{F}$, $F_1 \wedge F_2 \in \mathcal{F}$
   ($prop_4$) Pour tout $F_1, F_2 \in \mathcal{F}$, $F_1 \vee F_2 \in \mathcal{F}$
   ($prop_5$) Pour tout $F_1, F_2 \in \mathcal{F}$, $F_1 \Rightarrow F_2 \in \mathcal{F}$
   ($prop_6$) Pour tout $F_1, F_2 \in \mathcal{F}$, $F_1 \Leftrightarrow F_2 \in \mathcal{F}$

2. La fonction *sub*, de type $\mathcal{F} \to P(\mathcal{F})$, peut être définie de la façon suivante :

   (sub_1) Pour tout $s \in \mathcal{S} \cup \{\top, \bot\}$, $sub(s) = \{s\}$
   (sub_2) Pour tout $F \in \mathcal{F}$, $sub(\neg F) = \{\neg F\} \cup sub(F)$
   (sub_3) Pour tout $F_1, F_2 \in \mathcal{F}$, $sub(F_1 \wedge F_2) = \{F_1 \wedge F_2\} \cup sub(F_1) \cup sub(F_2)$

(sub_4) Pour tout $F_1, F_2 \in \mathcal{F}$, $sub(F_1 \vee F_2) = \{F_1 \vee F_2\} \cup sub(F_1) \cup sub(F_2)$

(sub_5) Pour tout $F_1, F_2 \in \mathcal{F}$, $sub(F_1 \Rightarrow F_2) = \{F_1 \Rightarrow F_2\} \cup sub(F_1) \cup sub(F_2)$

(sub_6) Pour tout $F_1, F_2 \in \mathcal{F}$, $sub(F_1 \Leftrightarrow F_2) = \{F_1 \Leftrightarrow F_2\} \cup sub(F_1) \cup sub(F_2)$

3. La fonction nbc, de type $\mathcal{F} \to \mathcal{N}$, peut être définie de la façon suivante :

   (nbc_1) Pour tout $s \in \mathcal{S} \cup \{\top, \bot\}$, $nbc(s) = 0$

   (nbc_2) Pour tout $F \in \mathcal{F}$, $nbc(\neg F) = S\ nbc(F)$

   (nbc_3) Pour tout $F_1, F_2 \in \mathcal{F}$, $nbc(F_1 \wedge F_2) = S\ plus(nbc(F_1), nbc(F_2))$

   (nbc_4) Pour tout $F_1, F_2 \in \mathcal{F}$, $nbc(F_1 \vee F_2) = S\ plus(nbc(F_1), nbc(F_2))$

   (nbc_5) Pour tout $F_1, F_2 \in \mathcal{F}$, $nbc(F_1 \Rightarrow F_2) = S\ plus(nbc(F_1), nbc(F_2))$

   (nbc_6) Pour tout $F_1, F_2 \in \mathcal{F}$, $nbc(F_1 \Leftrightarrow F_2) = S\ plus(nbc(F_1), nbc(F_2))$

4. Pour construire un schéma d'induction structurelle, il faut prendre en compte les règles de la base, puis les règles de construction. On doit d'abord vérifier la propriété sur les bases, il faut ensuite vérifier la propriété sur les règles de construction sachant la propriété vraie sur les prémisses des règles. Une fois la propriété vérifiée pour toutes les règles, alors la propriété est vérifiée pour tout élément de l'ensemble.

   Dans notre cas présent, le schéma d'induction structurelle des formules est le suivant :

$$\begin{gathered}
\forall P \in \mathcal{F} \to Prop. \\
(\forall s \in \mathcal{S} \cup \{\top, \bot\}).\ P(s)) \Rightarrow \\
(\forall F \in \mathcal{F}.\ P(F) \Rightarrow P(\neg F)) \Rightarrow \\
(\forall F_1, F_2 \in \mathcal{F}.\ P(F_1) \wedge P(F_2) \Rightarrow P(F_1 \wedge F_2)) \Rightarrow \\
(\forall F_1, F_2 \in \mathcal{F}.\ P(F_1) \wedge P(F_2) \Rightarrow P(F_1 \vee F_2)) \Rightarrow \\
(\forall F_1, F_2 \in \mathcal{F}.\ P(F_1) \wedge P(F_2) \Rightarrow P(F_1 \Rightarrow F_2)) \Rightarrow \\
(\forall F_1, F_2 \in \mathcal{F}.\ P(F_1) \wedge P(F_2) \Rightarrow P(F_1 \Leftrightarrow F_2)) \Rightarrow \\
\forall F \in \mathcal{F}.\ P(F)
\end{gathered}$$

5. Maintenant que les fonctions nbc et sub ainsi que le schéma d'induction des formules sont définis. Nous pouvons prouver $\forall F \in \mathcal{F}.\ |sub(F)| \leq 2 \times nbc(F) + 1$.
   Soit la propriété P suivante :

$$P(F) = |sub(F)| \leq 2 \times nbc(F) + 1$$

   Prouvons P par induction structurelle :

Base Prouvons P pour les cas de base, c'est à dire $\forall s \in \mathcal{S} \cup \{\top, \bot\}.\ P(s)$

$$\begin{aligned} \forall s \in \mathcal{S} \cup \{\top, \bot\}.\ |sub(s)| &= |\{sub(s)\}| && (sub_1) \\ &= 1 \end{aligned}$$

$$\begin{aligned} \forall s \in \mathcal{S} \cup \{\top, \bot\}.\ 2 \times nbc(F) + 1 &= 2 \times 0 + 1 && (nbc_1) \\ &= 1 \end{aligned}$$

On a bien $1 \leq 1$ et donc la propriété P est vraie pour la base.

Induction $prop_2$ Supposons $P(F)$, prouvons $P(\neg F)$

Nous avons donc l'hypothèse d'induction suivante :

$$|sub(F)| \leq 2 \times nbc(F) + 1$$

$$\begin{aligned} |sub(\neg F)| &= |\{\neg F\} \cup sub(F)| && (sub_2) \\ &= 1 + |sub(F)| \\ &\leq 2 \times nbc(F) + 2 && \text{(hypothèse d'induction)} \end{aligned}$$

$$\begin{aligned} 2 \times nbc(\neg F) + 1 &= 2 \times (S\ nbc(F)) + 1 && (nbc_2) \\ &= 2 \times nbc(F) + 3 \end{aligned}$$

Il est alors évident que $|sub(\neg F)| \leq 2 \times nbc(\neg F) + 1$
et nous avons bien $P(F) \Rightarrow P(\neg F)$

Induction $prop_3$ Supposons $P(F_1)$ et $P(F_2)$, prouvons $P(F_1 \wedge F_2)$

Nous avons donc les hypothèses d'induction suivantes :

$$|sub(F_1)| \leq 2 \times nbc(F_1) + 1$$
$$|sub(F_2)| \leq 2 \times nbc(F_2) + 1$$

$$\begin{aligned} |sub(F_1 \wedge F_2)| &= |\{F_1 \wedge F_2\} \cup sub(F_1) \cup sub(F_2)| && (sub_3) \\ &= 1 + |sub(F_1)| + |sub(F_2)| \\ &\leq 2 \times nbc(F_1) + 1 + 2 \times nbc(F_2) + 1 + 1 && \text{(hypothèses d'induction)} \\ &\leq 2 \times nbc(F_1) + 2 \times nbc(F_2) + 3 \end{aligned}$$

$$\begin{aligned} 2 \times nbc(F_1 \wedge F_2) + 1 &= 2 \times (S\ plus(nbc(F_1), nbc(F_2))) + 1 && (nbc_3) \\ &= 2 \times nbc(F_1) + 2 \times nbc(F_2)) + 3 \end{aligned}$$

Il est alors évident que $|sub(F_1 \wedge F_2)| \leq 2 \times nbc(F_1 \wedge F_2) + 1$
et nous avons bien $P(F_1) \wedge P(F_2) \Rightarrow P(F_1 \wedge F_2)$

Induction $prop_4$ Supposons $P(F_1)$ et $P(F_2)$, prouvons $P(F_1 \vee F_2)$

Suivre le même raisonnement que l'induction sur $prop_3$.

Induction $prop_5$ Supposons $P(F_1)$ et $P(F_2)$, prouvons $P(F_1 \Rightarrow F_2)$

Suivre le même raisonnement que l'induction sur $prop_3$.

Induction $prop_6$ Supposons $P(F_1)$ et $P(F_2)$, prouvons $P(F_1 \Leftrightarrow F_2)$

Suivre le même raisonnement que l'induction sur $prop_3$.

Conclusion $\forall F \in \mathcal{F}.\ P(F)$.

Exercice 4 (Relations inductives sur les listes)

1. Spécifier la relation « être une permutation de » pour deux listes.
2. Démontrer que la liste $[1;2;3]$ est une permutation de $[3;2;1]$.
3. Spécifier la relation « être triée » pour une liste.
4. Démontrer que la liste $[1\,;\,2\,;\,3]$ est triée.

1. On peut définir la relation inductive « être une permutation » de type $\mathcal{L} \to \mathcal{L} \to \mathit{Prop}$ de la façon suivante :

   (is_perm_1) Pour tout $l \in \mathcal{L}$, $is\_perm(l, l)$.

   (is_perm_2) Pour tout $l_1, l_2, l_3 \in \mathcal{L}$, pour tout $e \in \mathcal{A}$,
   si $is\_perm(l_1, app(l_2, l_3))$,
   alors on a : $is\_perm(e :: l_1, app(l_2, e :: l_3))$

2. On cherche à prouver $is\_perm([1;2;3],[3;2;1])$
   — Soit $l_1 = [2;3]$, $l_2 = [3;2]$, $l_3 = nil$, $a = 1$.
   On utilise is_perm_2 et on cherche à prouver $is\_perm([2;3],[3;2])$.
   — Soit $l_1 = [3]$, $l_2 = [3]$, $l_3 = nil$, $a = 2$.
   On utilise is_perm_2 et on cherche à prouver $is\_perm([3],[3])$.
   — Soit $l = [3]$.
   On utilise is_perm_1 et on a $is\_perm([3],[3]) \equiv \top$.

3. On peut définir la relation inductive « être triée » de type $\mathcal{L} \to \mathit{Prop}$ de la façon suivante avec $\mathcal{A} = \mathcal{N}$:

   (is_sort_1) On a $is\_sort(nil, nil)$.

   (is_sort_2) Pour tout $n \in \mathcal{N}$, on a $is\_sort([n])$.

   (is_sort_3) Pour tout $l \in \mathcal{L}$, pour tout $n_1, n_2 \in \mathcal{N}$,
   si on a $is\_sort(n_1 :: l)$ et $n_1 \geq n_2$,
   alors on a : $is\_sort(n_2 :: n_1 :: l)$.

4. On cherche à prouver $is\_sort([1;2;3])$
   — Soit $l = [3]$, $n_1 = 1$, $n_2 = 2$.
   On utilise is_sort_3, on a bien $n_1 \leq n_2$ et on cherche à prouver $is\_sort([2;3])$.
   — Soit $l = nil$, $n_1 = 2$, $n_2 = 3$.
   On utilise is_sort_3, on a bien $n_1 \leq n_2$ et on cherche à prouver $is\_sort([3])$.
   — Soit $n = 3$.
   On utilise is_sort_2, et on a $is\_sort([3]) \equiv \top$.

Exercice 5 (Preuves en Coq)

Faire les exercices 1, 2 et 3 en `Coq`.

◇ Voir TP3.

Exercice 6 (Preuves en Coq)

1. Écrire la relation inductive is_even (vue en cours).
2. Écrire une tactique qui démontre des buts de la forme $is\_even(n)$.
3. Écrire une tactique qui démontre des buts de la forme $\neg is\_even(n)$.
4. Écrire une tactique qui démontre les buts précédents indifféremment.
5. Écrire la fonction f_{is_even} qui teste si un entier est pair.
6. Démontrer que la fonction f_{is_even} est correcte vis-à-vis de la relation is_even.

◇ Voir TP3.