Exercices sur l'induction

RYAN KAVANAGH

Remarque 1. Ceci n'est pas un travail pratique à remettre. Ces exercices sont fournis simplement pour vous aider à pratiquer comment définir des relations par induction et comment raisonner par induction au sujet de ces définitions inductives.

1 Les arbres de syntaxe abstraite

Soient les arbres de syntaxe abstraite n, l générés par la grammaire suivante :

$$n, l ::= \mathsf{z} \mid \mathsf{s}(n) \mid \mathsf{nil} \mid \mathsf{cons}(n;\ l).$$

Ici on reconnaît deux opérateurs nullaires (z et nil), un opérateur unaire ($\mathsf{s}(n)$) et un opérateur binaire ($\mathsf{cons}(n;\ l)$). On appelle n et l des *métavariables*. On utilise le terme métavariable parce que plus tard nos arbres de syntaxe abstraite contiendront des variables, et le terme métavariable nous permet de distinguer entre le concept de variable qui appartient à notre méta-langage (une variable qui représente un arbre) et une variable qui appartient à notre langage (une variable qui peut paraître dans un arbre).

Exemple 1.1 (Nombres naturels).
— L'arbre z représente le nombre naturel 0.
— L'arbre $\mathsf{s}(\mathsf{s}(\mathsf{z}))$ représente le nombre naturel 2.

Exemple 1.2 (Listes de nombres naturels).
— L'arbre nil représente une liste vide $[]$.
— L'arbre $\mathsf{cons}(\mathsf{z};\ \mathsf{nil})$ représente la liste $[0]$ avec un élément.
— L'arbre $\mathsf{cons}(\mathsf{s}(\mathsf{z});\ \mathsf{cons}(\mathsf{z};\ \mathsf{nil}))$ représente la liste $[1, 0]$ avec deux éléments.
— L'arbre $\mathsf{cons}(\mathsf{z};\ \mathsf{cons}(\mathsf{s}(\mathsf{z});\ \mathsf{cons}(\mathsf{z};\ \mathsf{nil})))$ représente la liste $[0, 1, 0]$ avec trois éléments.

Exercice 1.3. Dessinez ces arbres de syntaxe abstraite en tant qu'arbres, où les nœuds sont les opérateurs.

Exercice 1.4. Est-ce qu'on peut former des listes de listes ? Des successeurs de listes ?

2 Jugements au sujet des nombres naturels et des listes

Où n est un arbre de syntaxe abstraite, on définit le jugement $n\ \mathtt{nat}$ pour signifier qu'il représente un nombre naturel. Ce jugement est défini par induction par les règles suivantes :

$\boxed{n\ \mathtt{nat}}$ l'arbre n représente un nombre naturel

$$\frac{}{\mathsf{z}\ \mathtt{nat}}\ (\textsc{nat-z}) \qquad \frac{n\ \mathtt{nat}}{\mathsf{s}(n)\ \mathtt{nat}}\ (\textsc{nat-s})$$

Il est important de noter que dans la règle ($\textsc{nat-s}$), la métavariable n représente *n'importe quel arbre de syntaxe abstraite* et non un arbre particulier !

Où l est un arbre de syntaxe abstraite, on définit le jugement $l\ \mathtt{liste}$ pour signifier qu'il représente une liste. Ce jugement est défini par induction par les règles suivantes, où les métavariables n et l encore une fois représente n'importe quel arbre de syntaxe abstraite :

INF889J: Fondements pratiques des langages de programmation, automne 2024, Université du Québec à Montréal

$\boxed{l\ \texttt{liste}}$ l'arbre l représente une liste

$$\frac{}{\mathsf{nil}\ \texttt{liste}}\ (\text{LISTE-NIL}) \qquad \frac{l\ \texttt{liste}}{\mathsf{cons}(n;\ l)\ \texttt{liste}}\ (\text{LISTE-CONS})$$

Exemple 2.1. La dérivation suivante démontre que l'arbre $\mathsf{cons}(\mathsf{z};\ \mathsf{cons}(\mathsf{z};\ \mathsf{nil}))$ est une liste :

$$\cfrac{\cfrac{\cfrac{}{\mathsf{nil}\ \texttt{liste}}\ (\text{LISTE-NIL})}{\mathsf{cons}(\mathsf{z};\ \mathsf{nil})\ \texttt{liste}}\ (\text{LISTE-CONS})}{\mathsf{cons}(\mathsf{z};\ \mathsf{cons}(\mathsf{z};\ \mathsf{nil}))\ \texttt{liste}}\ (\text{LISTE-CONS})$$

Exercice 2.2. Donnez une dérivation de $\mathsf{cons}(\mathsf{s}(\mathsf{z});\ \mathsf{nil})\ \texttt{liste}$.

Exercice 2.3. Définissez par induction une paire de jugements $n\ \texttt{pair}$ et $n\ \texttt{impair}$ qui signifient que l'arbre n est un nombre naturel pair ou impair. Pouvez-vous le faire en utilisant un seul axiome (une règle sans prémisses) ? Donnez des dérivations qui montrent que $\mathsf{s}(\mathsf{s}(\mathsf{z}))\ \texttt{pair}$ et $\mathsf{s}(\mathsf{z})\ \texttt{impair}$.

Exercice 2.4. Définissez par induction un jugement $l\ \texttt{listeDeListes}$ qui signifie que l'arbre l est une liste de listes.

Le jugement $\texttt{concat}(l;\ r;\ s)$, défini par induction par les règles suivantes, est une relation ternaire qui signifie que la liste s est la concaténation des listes l et r.

$\boxed{\texttt{concat}(l;\ r;\ s)}$ s est la concaténation de l er r

$$\frac{}{\texttt{concat}(\mathsf{nil};\ l;\ l)}\ (\text{CONCAT-NIL})$$

$$\frac{\texttt{concat}(t;\ l;\ r)}{\texttt{concat}(\mathsf{cons}(h;\ t);\ l;\ \mathsf{cons}(h;\ r))}\ (\text{CONCAT-CONS})$$

Exercice 2.5. Donnez une dérivation qui montre que la concaténation des listes $[0]$ et $[0]$ est $[0,0]$, c'est-à-dire, du jugement $\texttt{concat}(\mathsf{cons}(\mathsf{z};\ \mathsf{nil});\quad \mathsf{cons}(\mathsf{z};\ \mathsf{nil});\quad \mathsf{cons}(\mathsf{z};\ \mathsf{cons}(\mathsf{z};\ \mathsf{nil})))$.

Exercice 2.6. Définissez par induction un jugement $\texttt{len}(l;\ n)$ qui signifie que la longueur de la liste l est n. Donnez quelques dérivations qui utilisent ce jugement pour vérifier qu'il capte ce que vous vouliez capter.

3 Raisonnement par induction

Définition 3.1 (Principe d'induction pour le jugement $n\ \texttt{nat}$). Pour démontrer qu'une propriété P tient pour tout jugement $m\ \texttt{nat}$, il suffit de démontrer :

- si $m\ \texttt{nat}$ est formé par (NAT-Z) (il a la forme $\mathsf{z}\ \texttt{nat}$), alors P est vrai pour $\mathsf{z}\ \texttt{nat}$;
- si $m\ \texttt{nat}$ est formé par (NAT-S) (il a la forme $\mathsf{s}(n)\ \texttt{nat}$ où $n\ \texttt{nat}$), alors P est vrai pour $\mathsf{s}(n)\ \texttt{nat}$ en supposant que P est vrai pour $n\ \texttt{nat}$.

Définition 3.2 (Principe d'induction pour le jugement $\texttt{concat}(l;\ r;\ s)$). Pour démontrer qu'une propriété P tient pour tout jugement $\texttt{concat}(l;\ r;\ s)$, il suffit de démontrer :

- si $\texttt{concat}(l;\ r;\ s)$ est formé par (CONCAT-NIL), alors P est vrai pour $\texttt{concat}(\mathsf{nil};\ l;\ l)$;
- si $\texttt{concat}(l;\ r;\ s)$ est formé par (CONCAT-CONS), alors P est vrai pour $\texttt{concat}(\mathsf{cons}(h;\ t);\ l;\ \mathsf{cons}(h;\ r))$ en supposant que P est vrai pour $\texttt{concat}(t;\ l;\ r)$.

PROPOSITION 3.3 (EXEMPLE DE PREUVE PAR INDUCTION). *Si $\texttt{concat}(l;\ r;\ s)$, alors $l\ \texttt{liste}$.*

Démonstration. Par induction sur la dérivation $\texttt{concat}(l;\ r;\ s)$.

Cas où $\texttt{concat}(l;\ r;\ s)$ est formé par (concat-nil) (il a la forme $\texttt{concat}(\mathsf{nil};\ l;\ l)$). Il faut démontrer que $\mathsf{nil}\ \texttt{liste}$. La démonstration est donnée par la dérivation suivante :

$$\frac{}{\mathsf{nil}\ \texttt{liste}}\ (\textsc{liste-nil})$$

Cas où $\texttt{concat}(l;\ r;\ s)$ est formé par (concat-cons). Le jugement a la forme

$$\texttt{concat}(\mathsf{cons}(h;\ t);\ l;\ \mathsf{cons}(h;\ r))$$

où $\texttt{concat}(t;\ l;\ r)$. Il faut démontrer que $\mathsf{cons}(h;\ t)\ \texttt{liste}$ en supposant que $t\ \texttt{liste}$ (l'hypothèse de récurrence). La démonstration est donnée par la dérivation suivante, où la prémisse de (liste-cons) est justifiée par l'hypothèse de récurrence :

$$\frac{t\ \texttt{liste}}{\mathsf{cons}(h;\ t)\ \texttt{liste}}\ (\textsc{liste-cons})$$

On conclut le résultat par induction. □

Exercice 3.4. Complétez le schéma suivant pour le principe d'induction pour le jugement $l\ \texttt{liste}$:

Définition 3.5 (Principe d'induction pour le jugement $l\ \texttt{liste}$). Pour démontrer qu'une propriété P tient pour tout jugement $l\ \texttt{liste}$, il suffit de démontrer :

- si $l\ \texttt{liste}$ est formé par (liste-nil) (il a la forme ______________), alors P est vrai pour ______________ (la conclusion de (liste-nil));
- si $l\ \texttt{liste}$ est formé par (liste-cons) (il a la forme ______________ où ______________), alors P est vrai pour ______________ en supposant que P est vrai pour ______________.

Exercice 3.6. Utilisez votre principe pour démontrer : si $l\ \texttt{liste}$ est de la forme $\mathsf{cons}(n;\ r)\ \texttt{liste}$, alors $r\ \texttt{liste}$. Il se peut que vous n'ayez pas besoin d'utiliser l'hypothèse de récurrence !

Exercice 3.7. Prouvez ou trouvez un contre-exemple : si $\texttt{concat}(l;\ r;\ s)$, alors $s\ \texttt{liste}$. Si l'éconcé est faux, comment pourrions-nous modifier les règles pour assurer le résultat ?

Les deux exercices suivants nous permettront de savoir si le jugement $\texttt{concat}(l;\ r;\ s)$ défini une fonction en l et r :

Exercice 3.8. Prouvez ou trouvez un contre-exemple : si $l\ \texttt{liste}$, alors pour tout r il existe un s tel que $\texttt{concat}(l;\ r;\ s)$.

Exercice 3.9. Prouvez ou trouvez un contre-exemple : si $\texttt{concat}(l;\ r;\ s)$ et $\texttt{concat}(l;\ r;\ s')$, alors $s = s'$.

Exercice 3.10. Donnez le principe d'induction pour le jugement $\texttt{len}(l;\ n)$.

Exercice 3.11. Prouvez que si $\texttt{len}(l;\ \mathsf{z})$, alors $l = \mathsf{nil}$, et que si $\texttt{len}(l;\ \mathsf{s}(n))$, alors $l = \mathsf{cons}(m;\ l')$.

Exercice 3.12. Prouvez ou trouvez un contre-exemple : si $l\ \texttt{liste}$ alors il existe un n tel que $\texttt{len}(l;\ n)$.

Exercice 3.13. Prouvez ou trouvez un contre-exemple : si $\texttt{len}(l;\ n)$ et $\texttt{len}(l;\ m)$, alors $n = m$.

Exercice 3.14. Prouvez ou trouvez un contre-exemple : si $\texttt{len}(l;\ n)$ et $\texttt{len}(l';\ n)$, alors $l = l'$.

Exercice 3.15. Donnez un jugement qui donne la somme des nombres naturels dans une liste.

Exercice 3.16. Prouvez que le jugement $m\ \mathsf{mq}\ n$ donné en classe définit une relation transitive, c'est-à-dire que si $m\ \mathsf{mq}\ n$ et $n\ \mathsf{mq}\ r$, alors $m\ \mathsf{mq}\ r$.

Exercice 3.17. Prouvez ou trouvez un contre-exemple : si $\texttt{concat}(l;\ r;\ s)$, alors $l\ \texttt{liste}$. Si l'éconcé est faux, comment pourrions-nous modifier les règles pour assurer le résultat ?