Dokazovanje ekvivalenc med programi

Dostikrat si želimo dokazati, da nek program deluje tako, kot želimo. Običajno je to precej zahtevna naloga, saj so programski jeziki nepredvidljivi. Ker pa je Haskell čist jezik, so v njem izrazi enakovredni svojim definicijam, zato lahko ene zamenjamo z drugimi in obratno.

1 Seznami

Da bomo imeli kaj, s čimer bomo lahko delali, si najprej definirajmo sledeče tri funkcije:

$$\begin{aligned} \texttt{[] ++ ys} &= \texttt{ys} && (1) \\ \texttt{(x:xs) ++ ys} &= \texttt{x:(xs ++ ys)} && (2) \end{aligned}$$

$$\begin{aligned} \texttt{obrni []} &= \texttt{[]} && (3) \\ \texttt{obrni (x:xs)} &= \texttt{obrni xs ++ [x]} && (4) \end{aligned}$$

$$\begin{aligned} \texttt{dolzina []} &= \texttt{0} && (5) \\ \texttt{dolzina (x:xs)} &= \texttt{1 + dolzina xs} && (6) \end{aligned}$$

Trd. 1.1: `obrni [x]` = `[x]`

To enostavno trditev dokažemo zgolj z odvijanjem definicij.

$$\begin{aligned} \texttt{obrni [x]} &= \texttt{obrni (x:[])} && \text{ker je } \texttt{[x]} \text{ okrajšava za } \texttt{x:[]} \\ &= \texttt{[] ++ [x]} && \text{po (3)} \\ &= \texttt{[x]} && \text{po (1)} \end{aligned}$$

Trd. 1.2: `dolzina (xs ++ ys)` = `dolzina xs + dolzina ys`

Tukaj enostavno odvijanje definicij ne bo pomagalo, saj izraz `dolzina (xs ++ ys)` ne ustreza ne levi ne desni strani nobene od zgoraj naštetih definicij. Namesto tega uporabimo načelo indukcije za sezname:

$$P(\texttt{[]}) \wedge (\forall \texttt{x}, \texttt{xs}. P(\texttt{xs}) \Rightarrow P(\texttt{x:xs})) \implies \forall \texttt{ys}. P(\texttt{ys})$$

Torej, lastnost P velja za vse sezname `ys`, kadar (1) velja za prazen seznam `[]` in (2) velja za sestavljen seznam `x:xs` ob predpostavki, da velja za `xs`. Načelo indukcije je podobno načelu indukcije za naravna števila:

$$P(0) \wedge (\forall m. P(m) \Rightarrow P(m^+)) \implies \forall n. P(n)$$

Torej, lastnost P velja za vsa naravna števila n, kadar (1) velja za 0 in (2) velja za naslednika m^+ ob predpostavki, da velja za m.

Izjavo dokažemo z indukcijo na levi seznam. Indukcija poteka v dveh korakih:

Osnovni korak

V osnovnem koraku pokažemo, da enakost velja, kadar je levi seznam prazen, torej oblike `[]`:

$$\begin{aligned}
\texttt{dolzina ([] ++ ys)} &= \texttt{dolzina ys} && \text{po (1)}\\
&= \texttt{0 + dolzina ys} && \text{po Peanovih aksiomih}\\
&= \texttt{dolzina [] + dolzina ys} && \text{po (5)}
\end{aligned}$$

Indukcijski korak

V indukcijskem koraku pokažemo, da enakost velja za sestavljeni seznam `x:xs` ob predpostavki, da enakost velja za seznam `xs`:

$$\begin{aligned}
\texttt{dolzina ((x:xs) ++ ys)} &= \texttt{dolzina (x:(xs ++ ys))} && \text{po (2)}\\
&= \texttt{1 + dolzina (xs ++ ys)} && \text{po (6)}\\
&= \texttt{1 + (dolzina xs + dolzina ys)} && \text{po indukcijski prepostavki}\\
&= \texttt{(1 + dolzina xs) + dolzina ys)} && \text{po Peanovih aksiomih}\\
&= \texttt{dolzina (x:xs) + dolzina ys} && \text{po (6)}
\end{aligned}$$

Trd. 1.3: `xs ++ []` = `xs`

Po (1) vemo, da je `[] ++ xs` = `xs`, torej je `[]` leva enota za `++`. To, da je `[]` tudi desna enota, pa ni samoumevno — za dokaz uporabimo indukcijo.

Osnovni korak

Po (1) velja `[] ++ []` = `[]`, kar dokaže osnovni korak.

Indukcijski korak

Prepostavimo, da velja `xs ++ []` = `xs`. Tedaj velja

$$\begin{aligned}
\texttt{(x:xs) ++ []} &= \texttt{x:(xs ++ [])} && \text{po (2)}\\
&= \texttt{x:xs} && \text{po indukcijski prepostavki,}
\end{aligned}$$

kar zaključi tudi indukcijski korak.

Trd. 1.4: `xs ++ (ys ++ zs)` = `(xs ++ ys) ++ zs`

Operacija stikanja seznamov `++` je tudi asociativna, kar dokažemo z indukcijo na `xs`. Če zapišete dokaz asociativnosti seštevanja, lahko vidite, da poteka podobno, le da se namesto `[]` pojavlja 0, namesto `x:xs` pa naslednik n^+.

Osnovni korak

$$
\begin{aligned}
\texttt{[] ++ (ys ++ zs)} &= \texttt{ys ++ zs} && \text{po (1)}\\
&= \texttt{([] ++ ys) ++ zs} && \text{po (1)}
\end{aligned}
$$

Indukcijski korak

Prepostavimo, da velja `xs ++ (ys ++ zs)` = `(xs ++ ys) ++ zs`. Tedaj velja

$$
\begin{aligned}
\texttt{(x:xs) ++ (ys ++ zs)} &= \texttt{x:(xs ++ (ys ++ zs))} && \text{po (2)}\\
&= \texttt{x:((xs ++ ys) ++ zs)} && \text{po indukcijski predpostavki}\\
&= \texttt{(x:(xs ++ ys)) ++ zs} && \text{po (2)}\\
&= \texttt{((x:xs) ++ ys) ++ zs} && \text{po (2)}
\end{aligned}
$$

Trd. 1.5: `obrni (xs ++ ys)` = `obrni ys ++ obrni xs`

Osnovni korak

$$
\begin{aligned}
\texttt{obrni ([] ++ ys)} &= \texttt{obrni ys} && \text{po (1)}\\
&= \texttt{obrni ys ++ []} && \text{po trditvi 1.3}\\
&= \texttt{obrni ys ++ obrni []} && \text{po (3)}
\end{aligned}
$$

Indukcijski korak

$$
\begin{aligned}
\texttt{obrni ((x:xs) ++ ys)} &= \texttt{obrni (x:(xs ++ ys))} && \text{po (2)}\\
&= \texttt{obrni (xs ++ ys) ++ [x]} && \text{po (4)}\\
&= \texttt{(obrni ys ++ obrni xs) ++ [x]} && \text{po indukcijski predpostavki}\\
&= \texttt{obrni ys ++ (obrni xs ++ [x])} && \text{po trditvi 1.4}\\
&= \texttt{obrni ys ++ obrni (x:xs)} && \text{po (4)}
\end{aligned}
$$

Trd. 1.6: `obrni (obrni xs)` = `xs`

Na kvizu.

Trd. 1.7: `dolzina (obrni xs)` = `dolzina xs`

Domača naloga.

2 Drevesa