Az előző előadás tartalmából

Típuskikövetkeztetés (ismétlés)

- Típuskikövetkeztetés, a típus rekonstrukciója:
  Adott E esetén létezik-e τ, hogy $\varnothing \vdash E : \tau$?
- Típusozás korlátozásokkal: a korlátozások generálása, azok megoldása
  - Korlátozás (*constraint*): típuskifejezések közti egyenlőség
  - Egységesítés (*unification*): a korlátozások egyenletrendszerének megoldása
- Curry-típusrendszer + `let`-polimorfizmus $\equiv$ Hindley-Milner típusrendszer
- Hindley-Milner típusrendszer + polimorfikus rekurzió $\equiv$ Milner-Mycroft típusrendszer (hamarosan)

Hindley-Milner: szintaktika, típusok (ismétlés)

Definíció. ***Az egyszerű `let`-kifejezés szintaktikája***

<λ-kifejezés>	::=	*<változó>*
	|	*(λ <változó> . <λ-kifejezés>)*
	|	*(<λ-kifejezés> <λ-kifejezés>)*
	|	`let` *<változó>* = *<λ-kifejezés>* `in` *<λ-kifejezés>*

Definíció. ***A Hindley-Milner-típusrendszer típusai***

<típus>	::=	*<egyszerű típus>*
	|	*(∀ <típusváltozó> . <típus>)*
<egyszerű típus>	::=	*<típusváltozó>*
	|	*(<egyszerű típus>* → *<egyszerű típus>)*

τ: egyszerű típus (monotípus), σ: típusséma (politípus)

$\forall\overline{\alpha}.\sigma \equiv \forall\alpha_1.\forall\alpha_2.\ldots.\forall\alpha_n.\sigma \equiv \forall\alpha_1\alpha_2\ldots\alpha_n.\sigma \; (n > 0)$

$gen(\Gamma, \tau) = \forall\alpha_1\alpha_2\ldots\alpha_n.\tau$, $inst(\sigma) = \tau$

$\mathcal{W}$-algoritmus (ismétlés)

1. $\mathcal{W}(\Gamma, x) \leadsto \begin{cases} (Id, inst(\sigma)), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases}$

2. $\mathcal{W}(\Gamma, \lambda x.E) \leadsto (s, \alpha\ s \to \tau)$, ahol

2.1. $\mathcal{W}(\Gamma \cup \{x : \alpha\}, E) = (s, \tau)$ és α új változó

3. $\mathcal{W}(\Gamma, E\ F) \leadsto (s_1\ s_2\ s_3, \alpha\ s_3)$, ahol

3.1. $\mathcal{W}(\Gamma, E) = (s_1, \tau')$,

3.2. $\mathcal{W}(\Gamma\ s_1, F) = (s_2, \tau'')$,

3.3. $s_3 = \mathcal{U}(\{\tau'\ s_2 = \tau'' \to \alpha\}, \varepsilon)$ és α új változó

4. $\mathcal{W}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F) \leadsto (s_1\ s_2, \tau'')$, ahol

4.1. $\mathcal{W}(\Gamma, E) = (s_1, \tau')$,

4.2. $\mathcal{W}(\Gamma\ s_1 \cup \{x : gen(\Gamma\ s_1, \tau')\}, F) = (s_2, \tau'')$

$\Gamma\ s$: Γ minden $x : \sigma$ elemére végrehajtjuk az $x : \sigma$ átalakítást.

$\mathcal{W}(\varnothing, \texttt{let } f = \lambda x.x \texttt{ in } f\ f)$ (ismétlés)

$$
\begin{array}{l}
\mathcal{W}(\varnothing, \texttt{let } f = \lambda x.x \texttt{ in } f\ f) = (Id_1\ s_2, \tau'') \leadsto ([\beta := \delta, \delta := \gamma \to \gamma], \gamma \to \gamma) \\
\quad \mathcal{W}(\varnothing, \lambda x_1.x_2) = (s_1, \tau') \leadsto s_1 = Id_1, \tau' = \alpha \to \alpha \\
\quad\quad \mathcal{W}(\varnothing, \lambda x_1.x_2) = (s, \alpha\ s \to \tau) \leadsto (Id_1, \alpha\ Id_1 \to \alpha) \\
\quad\quad\quad \mathcal{W}(\varnothing \cup \{x_1 : \alpha\}, x_2) \leadsto \mathcal{W}(\{x : \alpha\}, x) = (s, \tau) \leadsto s = Id_1, \tau = \alpha \\
\quad\quad\quad\quad \mathcal{W}(\{x : \alpha\}, x) \leadsto (Id_1, inst(\alpha)) \leadsto (Id_1, \alpha) \\
\quad \mathcal{W}(\varnothing\ Id_1 \cup \{f : gen(\varnothing\ Id_1, \alpha \to \alpha)\}, f\ f) = (s_2, \tau'') \leadsto \\
\quad\quad\quad s_2 = [\beta := \delta, \delta := \gamma \to \gamma], \tau'' = \gamma \to \gamma \\
\quad\quad \mathcal{W}(\{f : \forall \alpha.\alpha \to \alpha\}, f_1\ f_2) = \\
\quad\quad\quad (Id_2\ Id_3\ [\beta := \delta, \delta := \gamma \to \gamma], \delta\ [\delta := \gamma \to \gamma]) \leadsto \\
\quad\quad\quad\quad ([\beta := \delta, \delta := \gamma \to \gamma], \gamma \to \gamma) \\
\quad\quad\quad \mathcal{W}(\{f : \forall \alpha.\alpha \to \alpha\}, f_1) = (s_1, \tau') \leadsto s_1 = Id_2, \tau' = \beta \to \beta \\
\quad\quad\quad\quad \mathcal{W}(\{f : \forall \alpha.\alpha \to \alpha\}, f_1) \leadsto (Id_2, inst(\forall \alpha.\alpha \to \alpha)) \leadsto (Id_2, \beta \to \beta) \\
\quad\quad\quad \mathcal{W}(\{f : \forall \alpha.\alpha \to \alpha\}\ Id_2, f_2) = (s_2, \tau'') \leadsto s_2 = Id_3, \tau'' = \gamma \to \gamma \\
\quad\quad\quad s_3 = \mathcal{U}(\{(\beta \to \beta)\ Id_3 = (\gamma \to \gamma) \to \delta\}, \varepsilon) \leadsto [\beta := \delta, \delta := \gamma \to \gamma]
\end{array}
$$

$$\varnothing \vdash \texttt{let } f = \lambda x.x \texttt{ in } f\ f : \gamma \to \gamma$$

A $\mathcal{J}$-algoritmus

$\mathcal{J}$: bevezetés

Definíció. ***A $\mathcal{J}$-algoritmus***
A $\mathcal{J}$-algoritmus egy adott Γ típuskörnyezethez és egy E kifejezéshez meghatároz egy τ típust, vagy $\mathtt{fail}$ hibajelzéssel megáll:

$$\mathcal{J} : \Gamma \times E \to \begin{cases} \tau \\ \mathtt{fail} \end{cases}$$

Tétel. ***A $\mathcal{J}$-algoritmus helyessége***
Ha $\mathcal{J}(\Gamma, E) = \tau$ és s az eredményül kapott helyettesítés, akkor $\Gamma\ s \vdash E : \tau\ s$.

Tétel. ***A $\mathcal{J}$-algoritmus teljessége***
Ha $\mathcal{J}(\Gamma, E) = \tau$ és a helyettesítés s, és egy τ' típusra és egy s' helyettesítésre $\Gamma\ s' \vdash E : \tau'\ s'$ is fennáll, akkor van olyan s'' helyettesítés, amelyre $\tau' = inst((gen(\Gamma\ s, \tau))\ s'')$ és $\Gamma\ s' = \Gamma\ s\ s''$.

$\mathcal{J}$: szabályok

1. $\mathcal{J}(\Gamma, x) \rightsquigarrow \begin{cases} \mathit{inst}(\sigma), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases}$

2. $\mathcal{J}(\Gamma, \lambda x.E) \rightsquigarrow \alpha \to \tau$, ahol

2.1. $\mathcal{J}(\Gamma \cup \{x : \alpha\}, E) = \tau$ és α új változó

3. $\mathcal{J}(\Gamma, E\ F) \rightsquigarrow \alpha$, ahol

3.1. $\mathcal{J}(\Gamma, E) = \tau'$,

3.2. $\mathcal{J}(\Gamma, F) = \tau''$,

3.3. $s := s\ (\mathcal{U}(\tau'\ s = (\tau'' \to \alpha)\ s, \varepsilon))$ és α új változó

4. $\mathcal{J}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F) \rightsquigarrow \tau''$, ahol

4.1. $\mathcal{J}(\Gamma, E) = \tau'$,

4.2. $\mathcal{J}(\Gamma \cup \{x : \mathit{gen}(\Gamma\ s, \tau'\ s)\}, F) = \tau''$

$s \equiv \mathit{Id}$ kezdetben

A 3. pontban nem kell meghatározni új helyettesítést, csak nyilvántartani a korlátozásokat – az egységesítés elhalasztható a 4. pontig

$\mathcal{J}$: példa: $\mathcal{J}(\varnothing, \lambda x.x)$

$$
\begin{aligned}
&s = Id, \mathcal{J}(\varnothing, \lambda x.x) \rightsquigarrow \mathcal{J}(\varnothing, \lambda x_1.x_2) \rightsquigarrow \alpha \to \tau \rightsquigarrow \alpha \to \alpha \\
&\quad \mathcal{J}(\varnothing \cup \{x_1 : \alpha\}, x_2) = \tau \rightsquigarrow \tau = \alpha \\
&\qquad \mathcal{J}(\{x_1 : \alpha\}, x_2) \rightsquigarrow inst(\alpha) \rightsquigarrow \alpha
\end{aligned}
$$

$$\varnothing \vdash \lambda x.x : \alpha \to \alpha$$

$\mathcal{J}$: szabályok

1. $\mathcal{J}(\Gamma, x) \quad \rightsquigarrow \quad \begin{cases} inst(\sigma), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases}$

2. $\mathcal{J}(\Gamma, \lambda x.E) \quad \rightsquigarrow \quad \alpha \to \tau$, ahol

2.1. $\mathcal{J}(\Gamma \cup \{x : \alpha\}, E) = \tau$ és α új változó

3. $\mathcal{J}(\Gamma, E\ F) \quad \rightsquigarrow \quad \alpha$, ahol

3.1. $\mathcal{J}(\Gamma, E) = \tau'$,

3.2. $\mathcal{J}(\Gamma, F) = \tau''$,

3.3. $s := s\ (\mathcal{U}(\tau'\ s = (\tau'' \to \alpha)\ s, \varepsilon))$ és α új változó

4. $\mathcal{J}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F) \rightsquigarrow \tau''$, ahol

4.1. $\mathcal{J}(\Gamma, E) = \tau'$,

4.2. $\mathcal{J}(\Gamma \cup \{x : gen(\Gamma\ s, \tau'\ s)\}, F) = \tau''$

$s \equiv Id$ kezdetben

$\mathcal{J}$: példa: $\mathcal{J}(\{x : \alpha\}, x\ x)$

$$
\begin{array}{l}
s = \mathit{Id}, \mathcal{J}(\{x : \alpha\}, x\ x) \rightsquigarrow \mathcal{J}(\{x : \alpha\}, x_1\ x_2) \rightsquigarrow \texttt{fail} \\
\quad \mathcal{J}(\{x : \alpha\}, x_1) = \tau' \rightsquigarrow \tau' = \alpha \\
\quad\quad (\{x : \alpha\}, x_1) \rightsquigarrow \mathit{inst}(\alpha) \rightsquigarrow \alpha \\
\quad \mathcal{J}(\{x : \alpha\}, x_2) = \tau'' \rightsquigarrow \tau'' = \alpha \\
\quad\quad (\{x : \alpha\}, x_2) \rightsquigarrow \mathit{inst}(\alpha) \rightsquigarrow \alpha \\
\quad s := \mathit{Id}\ (\mathcal{U}(\alpha\ \mathit{Id} = (\alpha \to \beta)\ \mathit{Id}, \varepsilon)) \rightsquigarrow \mathcal{U}(\{\alpha = \alpha \to \beta\}, \varepsilon) \rightsquigarrow \\
\quad\quad \texttt{fail}
\end{array}
$$

$\mathcal{J}$: szabályok

1. $\mathcal{J}(\Gamma, x) \leadsto \begin{cases} inst(\sigma), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases}$

2. $\mathcal{J}(\Gamma, \lambda x.E) \leadsto \alpha \to \tau$, ahol

2.1. $\mathcal{J}(\Gamma \cup \{x : \alpha\}, E) = \tau$ és α új változó

3. $\mathcal{J}(\Gamma, E\ F) \leadsto \alpha$, ahol

3.1. $\mathcal{J}(\Gamma, E) = \tau'$,

3.2. $\mathcal{J}(\Gamma, F) = \tau''$,

3.3. $s := s\ (\mathcal{U}(\tau'\ s = (\tau'' \to \alpha)\ s, \varepsilon))$ és α új változó

4. $\mathcal{J}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F) \leadsto \tau''$, ahol

4.1. $\mathcal{J}(\Gamma, E) = \tau'$,

4.2. $\mathcal{J}(\Gamma \cup \{x : gen(\Gamma\ s, \tau'\ s)\}, F) = \tau''$

$s \equiv Id$ kezdetben

$\mathcal{J}$: példa: $\mathcal{J}(\varnothing, \texttt{let } f = \lambda x.x \texttt{ in } f\ f)$

$$
\begin{array}{l}
s = Id, \mathcal{J}(\varnothing, \texttt{let } f = \lambda x.x \texttt{ in } f\ f) = \tau'' \rightsquigarrow \delta \\
\quad \mathcal{J}(\varnothing, \lambda x.x) = \tau' \rightsquigarrow \tau' = \alpha \to \alpha \\
\quad\quad \mathcal{J}(\varnothing, \lambda x_1.x_2) = \alpha \to \tau \rightsquigarrow \alpha \to \alpha \\
\quad\quad\quad \mathcal{J}(\varnothing \cup \{x_1 : \alpha\}, x_2) = \tau \rightsquigarrow \mathcal{J}(\{x : \alpha\}, x) \rightsquigarrow inst(\alpha) \rightsquigarrow \alpha \rightsquigarrow \tau = \alpha \\
\quad \mathcal{J}(\varnothing \cup \{f : gen(\varnothing\ Id, (\alpha \to \alpha)\ Id)\}, f\ f) = \tau'' \rightsquigarrow \delta \\
\quad\quad \mathcal{J}(\{f : \forall\alpha.\alpha \to \alpha\}, f\ f) \rightsquigarrow \mathcal{J}(\{f : \forall\alpha.\alpha \to \alpha\}, f_1\ f_2) = \delta \rightsquigarrow \delta \\
\quad\quad\quad \mathcal{J}(\{f : \forall\alpha.\alpha \to \alpha\}, f_1) = \sigma' \rightsquigarrow \sigma' = \beta \to \beta \\
\quad\quad\quad\quad \mathcal{J}(\{f : \forall\alpha.\alpha \to \alpha\}, f_1) \rightsquigarrow inst(\forall\alpha.\alpha \to \alpha) \rightsquigarrow \beta \to \beta \\
\quad\quad\quad \mathcal{J}(\{f : \forall\alpha.\alpha \to \alpha\}, f_2) = \sigma'' \rightsquigarrow \sigma'' = \gamma \to \gamma \\
\quad\quad\quad\quad \mathcal{J}(\{f : \forall\alpha.\alpha \to \alpha\}, f_2) \rightsquigarrow inst(\forall\alpha.\alpha \to \alpha) \rightsquigarrow \gamma \to \gamma \\
\quad\quad\quad s := Id\ (\mathcal{U}((\beta \to \beta)\ Id = ((\gamma \to \gamma) \to \delta)\ Id, \varepsilon)) \rightsquigarrow \\
\quad\quad\quad\quad \mathcal{U}(\{\beta \to \beta = (\gamma \to \gamma) \to \delta\}, \varepsilon) \\
\quad\quad\quad\quad [\beta := \delta, \delta := \gamma \to \gamma]
\end{array}
$$

$$
\begin{array}{l}
\varnothing\ [\beta := \delta, \delta := \gamma \to \gamma] \vdash \texttt{let } f = \lambda x.x \texttt{ in } f\ f : \delta\ [\beta := \delta, \delta := \gamma \to \gamma] \\
\varnothing \vdash \texttt{let } f = \lambda x.x \texttt{ in } f\ f : \gamma \to \gamma
\end{array}
$$

Az $\mathcal{M}$-algoritmus

$\mathcal{M}$: bevezetés

Definíció. ***Az $\mathcal{M}$-algoritmus***
Az $\mathcal{M}$-algoritmus egy adott Γ típuskörnyezethez, egy E kifejezéshez és egy τ típushoz meghatároz egy s helyettesítést, vagy `fail` hibajelzéssel megáll:

$$\mathcal{M} : \Gamma \times E \times \tau \to \begin{cases} s \\ \mathtt{fail} \end{cases}$$

Környezetfüggő, felülről-lefele haladó következtetés: $E\ F$ esetén az E feldolgozása után rögtön ellenőrizni, hogy E függvény-e.

Tétel. ***Az $\mathcal{M}$-algoritmus helyessége***
Ha $\mathcal{M}(\Gamma, E, \tau) = s$, akkor $\Gamma\ s \vdash E : \tau\ s$.

Tétel. ***Az $\mathcal{M}$-algoritmus teljessége***
Ha egy Γ típuskörnyezetre és egy E kifejezésre van olyan τ típus és s' helyettesítés, hogy $\Gamma\ s' \vdash E : \tau'\ s'$, akkor $\mathcal{M}(\Gamma, E, \tau) = s$ létezik, és van olyan s'' helyettesítés, amellyel $s' = s\ s''$.

$\mathcal{M}$: szabályok

1. $\mathcal{M}(\Gamma, x, \tau) \rightsquigarrow \begin{cases} \mathcal{U}(\{\tau = \mathit{inst}(\sigma)\}, \varepsilon), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases}$

2. $\mathcal{M}(\Gamma, \lambda x.E, \tau) \rightsquigarrow s_1\ s_2$, ahol

2.1. $s_1 = \mathcal{U}(\{\tau = \alpha \to \beta\}, \varepsilon)$ és α és β új változók

2.2. $s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : \alpha\ s_1\}, E, \beta\ s_1)$

3. $\mathcal{M}(\Gamma, E\ F, \tau) \rightsquigarrow s_1\ s_2$, ahol

3.1. $s_1 = \mathcal{M}(\Gamma, E, \alpha \to \tau)$, α új változó

3.2. $s_2 = \mathcal{M}(\Gamma\ s_1, F, \alpha\ s_1)$

4. $\mathcal{M}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F, \tau) \rightsquigarrow s_1\ s_2$, ahol

4.1. $s_1 = \mathcal{M}(\Gamma, E, \alpha)$, α új változó

4.2. $s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : \mathit{gen}(\Gamma\ s_1, \alpha\ s_1)\}, F, \tau)$

$\mathcal{M}$: példa: $\mathcal{M}(\varnothing, \lambda x.x, \tau)$

$$
\begin{aligned}
&\mathcal{M}(\varnothing, \lambda x.x, \tau) \rightsquigarrow \mathcal{M}(\varnothing, \lambda x_1.x_2, \tau) \rightsquigarrow [\tau := \alpha \to \beta]\ [\beta := \alpha] \rightsquigarrow \\
&\quad [\tau := \alpha \to \beta, \beta := \alpha] \\
&\quad s_1 = \mathcal{U}(\{\tau = \alpha \to \beta\}, \varepsilon) \rightsquigarrow [\tau := \alpha \to \beta] \\
&\quad s_2 = \mathcal{M}(\varnothing\ [\tau := \alpha \to \beta] \cup \{x_1 : \alpha\ [\tau := \alpha \to \beta]\}, x_2, \beta\ [\tau := \alpha \to \beta]) \rightsquigarrow \\
&\qquad \mathcal{M}(\{x : \alpha\}, x_2, \beta) \rightsquigarrow \\
&\qquad\quad \mathcal{M}(\{x : \alpha\}, x_2, \beta) \rightsquigarrow \mathcal{U}(\{\beta = inst(\alpha)\}, \varepsilon) \rightsquigarrow \mathcal{U}(\{\beta = \alpha\}, \varepsilon) \rightsquigarrow \\
&\qquad\qquad [\beta := \alpha]
\end{aligned}
$$

$$\varnothing\ [\tau := \alpha \to \beta, \beta := \alpha] \vdash \lambda x.x : \tau\ [\tau := \alpha \to \beta, \beta := \alpha]$$

$$\varnothing \vdash \lambda x.x : \alpha \to \alpha$$

$\mathcal{M}$: szabályok

1. $\mathcal{M}(\Gamma, x, \tau) \rightsquigarrow \begin{cases} \mathcal{U}(\{\tau = inst(\sigma)\}, \varepsilon), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases}$

2. $\mathcal{M}(\Gamma, \lambda x.E, \tau) \rightsquigarrow s_1\ s_2$, ahol

2.1. $s_1 = \mathcal{U}(\{\tau = \alpha \to \beta\}, \varepsilon)$ és α és β új változók

2.2. $s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : \alpha\ s_1\}, E, \beta\ s_1)$

3. $\mathcal{M}(\Gamma, E\ F, \tau) \rightsquigarrow s_1\ s_2$, ahol

3.1. $s_1 = \mathcal{M}(\Gamma, E, \alpha \to \tau)$, α új változó

3.2. $s_2 = \mathcal{M}(\Gamma\ s_1, F, \alpha\ s_1)$

4. $\mathcal{M}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F, \tau) \rightsquigarrow s_1\ s_2$, ahol

4.1. $s_1 = \mathcal{M}(\Gamma, E, \alpha)$, α új változó

4.2. $s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : gen(\Gamma\ s_1, \alpha\ s_1)\}, F, \tau)$

$\mathcal{M}$: példa: $\mathcal{M}(\{x : \alpha\}, x\ x, \tau)$

$$
\begin{array}{l}
\mathcal{M}(\{x : \alpha\}, x\ x, \tau) \leadsto \mathcal{M}(\{x : \alpha\}, x_1\ x_2, \tau) \leadsto [\alpha := \beta \to \tau]\ s_2 \leadsto \texttt{fail} \\
\quad s_1 = \mathcal{M}(\{x : \alpha\}, x_1, \beta \to \tau) \leadsto [\alpha := \beta \to \tau] \\
\qquad \mathcal{M}(\{x : \alpha\}, x, \beta \to \tau) \leadsto \mathcal{U}(\{\beta \to \tau = \mathit{inst}(\alpha)\}, \varepsilon) \leadsto \\
\qquad \mathcal{U}(\{\beta \to \tau = \alpha\}, \varepsilon) \leadsto [\alpha := \beta \to \tau] \\
\quad s_2 = \mathcal{M}(\{x : \alpha\}\ [\alpha := \beta \to \tau], x_2, \beta\ [\alpha := \beta \to \tau]) \leadsto \\
\qquad \mathcal{M}(\{x : \beta \to \tau\}, x_2, \beta) \leadsto \mathcal{U}(\{\beta = \mathit{inst}(\beta \to \tau)\}, \varepsilon) \leadsto \\
\qquad \mathcal{U}(\{\beta = \beta \to \tau\}, \varepsilon) \leadsto \texttt{fail}
\end{array}
$$

$\mathcal{M}$: szabályok

1. $\mathcal{M}(\Gamma, x, \tau) \leadsto \begin{cases} \mathcal{U}(\{\tau = inst(\sigma)\}, \varepsilon), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases}$

2. $\mathcal{M}(\Gamma, \lambda x.E, \tau) \leadsto s_1\ s_2$, ahol
2.1. $s_1 = \mathcal{U}(\{\tau = \alpha \to \beta\}, \varepsilon)$ és α és β új változók
2.2. $s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : \alpha\ s_1\}, E, \beta\ s_1)$

3. $\mathcal{M}(\Gamma, E\ F, \tau) \leadsto s_1\ s_2$, ahol
3.1. $s_1 = \mathcal{M}(\Gamma, E, \alpha \to \tau)$, α új változó
3.2. $s_2 = \mathcal{M}(\Gamma\ s_1, F, \alpha\ s_1)$

4. $\mathcal{M}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F, \tau) \leadsto s_1\ s_2$, ahol
4.1. $s_1 = \mathcal{M}(\Gamma, E, \alpha)$, α új változó
4.2. $s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : gen(\Gamma\ s_1, \alpha\ s_1)\}, F, \tau)$

$\mathcal{M}$: példa: $\mathcal{M}(\varnothing, \texttt{let}\ f = \lambda x.x\ \texttt{in}\ f\ f, \tau)$

$$
\begin{array}{l}
\mathcal{M}(\varnothing, \texttt{let}\ f = \lambda x.x\ \texttt{in}\ f\ f, \tau) \leadsto \\
\quad [\alpha := \beta \to \gamma, \gamma := \beta]\ [\tau := \varphi, \delta := \varphi, \varphi := \eta \to \eta] \leadsto \\
\quad [\alpha := \beta \to \gamma, \gamma := \beta, \tau := \varphi, \delta := \varphi, \varphi := \eta \to \eta] \\
\quad s_1 = \mathcal{M}(\varnothing, \lambda x.x, \alpha) \leadsto [\alpha := \beta \to \gamma, \gamma := \beta] \\
\qquad \mathcal{M}(\varnothing, \lambda x_1.x_2, \alpha) \leadsto [\alpha := \beta \to \gamma]\ [\gamma := \beta] \leadsto [\alpha := \beta \to \gamma, \gamma := \beta] \\
\qquad\quad s_1 = \mathcal{U}(\{\alpha = \beta \to \gamma\}, \varepsilon) \leadsto [\alpha := \beta \to \gamma] \\
\qquad\quad s_2 = \mathcal{M}(\varnothing\ [\alpha := \beta \to \gamma] \cup \{x_1 : \beta\ [\alpha := \beta \to \gamma]\}, x_2, \gamma\ [\alpha := \beta \to \gamma]) \leadsto \\
\qquad\qquad \mathcal{M}(\{x : \beta\}, x_2, \gamma) \leadsto \mathcal{U}(\{\gamma = inst(\beta)\}, \varepsilon) \leadsto \mathcal{U}(\{\gamma = \beta\}, \varepsilon) \leadsto [\gamma := \beta] \\
\quad s_2 = \mathcal{M}(\varnothing\ [\alpha := \beta \to \gamma] \cup \{f : gen(\varnothing\ [\alpha := \beta \to \gamma], \alpha\ [\alpha := \beta \to \gamma])\}, f\ f, \tau) \\
\qquad \leadsto \mathcal{M}(\{f : \forall \beta.\beta \to \beta\}, f_1\ f_2, \tau) \leadsto [\tau := \varphi, \delta := \varphi]\ [\varphi := \eta \to \eta] \leadsto \\
\qquad\quad [\tau := \varphi, \delta := \varphi, \varphi := \eta \to \eta] \\
\qquad\quad s_1 = \mathcal{M}(\{f : \forall \beta.\beta \to \beta\}, f_1, \delta \to \tau) \leadsto \\
\qquad\qquad \mathcal{U}(\{\delta \to \tau = inst(\forall \beta.\beta \to \beta)\}, \varepsilon) \leadsto \mathcal{U}(\{\delta \to \tau = \varphi \to \varphi\}, \varepsilon) \leadsto \\
\qquad\qquad [\tau := \varphi, \delta := \varphi] \\
\qquad\quad s_2 = \mathcal{M}(\{f : \forall \beta.\beta \to \beta\}\ [\tau := \varphi, \delta := \varphi], f_2, \delta\ [\tau := \varphi, \delta := \varphi]) \leadsto \\
\qquad\qquad \mathcal{M}(\{f : \forall \beta : \beta \to \beta\}, f_2, \varphi) \leadsto \mathcal{U}(\{\varphi = inst(\forall \beta : \beta \to \beta)\}, \varepsilon) \leadsto \\
\qquad\qquad \mathcal{U}(\{\eta \to \eta\}, \varepsilon) \leadsto [\varphi := \eta \to \eta]
\end{array}
$$

$$\varnothing \vdash \texttt{let}\ f = \lambda x.x\ \texttt{in}\ f\ f : \eta \to \eta$$

Rekurzió

Bevezetés: visszatérés a típus nélküli λ-kalkulushoz

Rekurzív matematikai függvény ($x = E$ alakban):
$x = \ldots x \ldots$

ebből készíthető egy λ-kifejezés:
$F \equiv \lambda x.\ \ldots x \ldots$, $F \equiv \lambda x.E$

Erre $x = F\ x$, ahol x az F fixpontja, amelyet például az alábbi fixpont-kombinátorokkal adhatunk meg:

$$Y \equiv \lambda x.(\lambda y.x\ (y\ y))\ (\lambda y.x\ (y\ y))$$
$$\Theta \equiv (\lambda x\ y.y\ (x\ x\ y))\ (\lambda x\ y.y\ (x\ x\ y))$$

$x = Y\ F$
$x = Y\ (\lambda x.\ \ldots x \ldots)$
$x = Y\ (\lambda x.E)$

Rövidítés:
$Y\ (\lambda x.E) \equiv \mathtt{fix}\ x.E$

Bevezetés: a HM-típusrendszer bővítése

Definíció. ***A HM-típusrendszer bővített szintaktikája***

<típus>	::=	*<egyszerű típus>*
	|	*(∀ <típusváltozó> . <típus>)*
<egyszerű típus>	::=	*<típusváltozó>*
	|	*(<egyszerű típus> → <egyszerű típus>)*
<λ-kifejezés>	::=	*<változó>*
	|	*(λ <változó> . <λ-kifejezés>)*
	|	*(<λ-kifejezés> <λ-kifejezés>)*
	|	`let` *<változó> = <λ-kifejezés>* `in` *<λ-kifejezés>*
	|	*<fix-kifejezés>*
<fix-kifejezés>	::=	`fix` *<változó> . <λ-kifejezés>*

`fix`: monomorf

Bevezetés: a HM-típusrendszer `fix` kifejezésekkel

Definíció. ***A HM-típusrendszer új szabálya***

A Hindley-Milner-típusrendszernek a korábbi definícióban megadott szabályait a következő szabállyal bővítjük:

$$\frac{\Gamma, x : \tau \vdash E : \tau}{\Gamma \vdash \mathtt{fix}\; x.E : \tau} \; (\textsc{val fix})$$

τ monomorf: az x, az E és a `fix` $x.E$ típusa is τ, azaz nincs polimorfizmus a rekurzióban.

Bevezetés: a rekurzió operációs szemantikája

Az új $\texttt{fix}\ x.E$ kifejezés operációs szemantikája a következőképpen adható meg:

$$\texttt{fix}\ x.E \rightarrow E[x := \texttt{fix}\ x.E]$$

mivel a fixpont definíciója alapján:

$$Y\ F = F\ (Y\ F)$$

azaz:

$$\texttt{fix}\ x.E = F\ (\texttt{fix}\ x.E) \equiv (\lambda x.E)\ (\texttt{fix}\ x.E)$$

$$(\lambda x.E)\ (\texttt{fix}\ x.E) \rightarrow_\beta E[x := \texttt{fix}\ x.E]$$

Bevezetés: a $\mathcal{W}$-algoritmus bővítése

$$
\begin{array}{lll}
1. & \mathcal{W}(\Gamma, x) & \leadsto \begin{cases} (Id, inst(\sigma)), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases} \\
2. & \mathcal{W}(\Gamma, \lambda x.E) & \leadsto (s, \alpha\ s \to \tau), \text{ ahol} \\
2.1. & & \mathcal{W}(\Gamma \cup \{x : \alpha\}, E) = (s, \tau) \text{ és } \alpha \text{ új változó} \\
3. & \multicolumn{2}{l}{\mathcal{W}(\Gamma, E\ F) \leadsto (s_1\ s_2\ s_3, \alpha\ s_3), \text{ ahol}} \\
3.1. & \multicolumn{2}{l}{\quad \mathcal{W}(\Gamma, E) = (s_1, \tau'),} \\
3.2. & \multicolumn{2}{l}{\quad \mathcal{W}(\Gamma\ s_1, F) = (s_2, \tau''),} \\
3.3. & \multicolumn{2}{l}{\quad s_3 = \mathcal{U}(\{\tau'\ s_2 = \tau'' \to \alpha\}, \varepsilon) \text{ és } \alpha \text{ új változó}} \\
4. & \multicolumn{2}{l}{\mathcal{W}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F) \leadsto (s_1\ s_2, \tau''), \text{ ahol}} \\
4.1. & \multicolumn{2}{l}{\quad \mathcal{W}(\Gamma, E) = (s_1, \tau'),} \\
4.2. & \multicolumn{2}{l}{\quad \mathcal{W}(\Gamma\ s_1 \cup \{x : gen(\Gamma\ s_1, \tau')\}, F) = (s_2, \tau'')}
\end{array}
$$

$$
\boxed{
\begin{array}{ll}
5. & \mathcal{W}(\Gamma, \texttt{fix}\ x.E) \leadsto (s_1\ s_2, \tau'\ s_2), \\
5.1. & \quad \mathcal{W}(\Gamma \cup \{x : \beta\}, E) = (s_1, \tau'), \ \beta \text{ új változó} \\
5.2. & \quad s_2 = \mathcal{U}(\{\beta\ s_1 = \tau'\}, \varepsilon)
\end{array}
}
$$

Bevezetés: az $\mathcal{M}$-algoritmus bővítése

$$
\begin{array}{rl}
1. & \mathcal{M}(\Gamma, x, \tau) \rightsquigarrow \begin{cases} \mathcal{U}(\{\tau = \mathit{inst}(\sigma)\}, \varepsilon), & \text{ha } \{x : \sigma\} \in \Gamma \\ \texttt{fail} & \text{egyébként} \end{cases} \\
2. & \mathcal{M}(\Gamma, \lambda x.E, \tau) \rightsquigarrow s_1\ s_2, \text{ ahol} \\
2.1. & \quad s_1 = \mathcal{U}(\{\tau = \alpha \to \beta\}, \varepsilon) \text{ és } \alpha \text{ és } \beta \text{ új változók} \\
2.2. & \quad s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : \alpha\ s_1\}, E, \beta\ s_1) \\
3. & \mathcal{M}(\Gamma, E\ F, \tau) \rightsquigarrow s_1\ s_2, \text{ ahol} \\
3.1. & \quad s_1 = \mathcal{M}(\Gamma, E, \alpha \to \tau), \alpha \text{ új változó} \\
3.2. & \quad s_2 = \mathcal{M}(\Gamma\ s_1, F, \alpha\ s_1) \\
4. & \mathcal{M}(\Gamma, \texttt{let}\ x = E\ \texttt{in}\ F, \tau) \rightsquigarrow s_1\ s_2, \text{ ahol} \\
4.1. & \quad s_1 = \mathcal{M}(\Gamma, E, \alpha), \alpha \text{ új változó} \\
4.2. & \quad s_2 = \mathcal{M}(\Gamma\ s_1 \cup \{x : \mathit{gen}(\Gamma\ s_1, \alpha\ s_1)\}, F, \tau) \\
\hline
5. & \mathcal{M}(\Gamma, \texttt{fix}\ x.E, \tau) \rightsquigarrow \mathcal{M}(\Gamma \cup \{x : \tau\}, E, \tau) \\
\hline
\end{array}
$$

A Milner-Mycroft-típusrendszer

MM-típusrendszer: polimorf `fix`-kifejezések

Definíció. ***A Milner-Mycroft-típusrendszer szabályai***

Ha a Hindley-Milner-típusrendszernek a korábbi definícióban megadott szabályát a következőképpen módosítjuk:

$$\frac{\Gamma, x : \sigma \vdash E : \sigma}{\Gamma \vdash \texttt{fix}\ x.E : \sigma}\ (\text{VAL FIX POLY})$$

valamint a többi szabályát módosítások nélkül átvesszük, akkor a Milner-Mycroft típusrendszer szabályait kapjuk meg.

A rekurzív függvények definíciójában a típusok típussémák ($\forall\alpha.\tau$ alakúak).

MM-típusrendszer: szabályok összefoglalása (1/2)

$$\frac{}{\varnothing \vdash \mathsf{wf}} \; (\text{ENV } \varnothing) \qquad \frac{\Gamma \vdash \sigma, x \notin dom(\Gamma)}{\Gamma, x : \sigma \vdash \mathsf{wf}} \; (\text{ENV } x)$$

$$\frac{\Gamma \vdash \mathsf{wf}, \alpha \in TV}{\Gamma \vdash \alpha} \; (\text{TYPE VAR}) \qquad \frac{\Gamma \vdash \tau', \Gamma \vdash \tau''}{\Gamma \vdash \tau' \to \tau''} \; (\text{TYPE ARROW})$$

$$\frac{\Gamma \vdash \sigma, \Gamma \vdash \alpha}{\Gamma \vdash \forall \alpha . \sigma} \; (\text{TYPE FORALL}) \qquad \frac{\Gamma', x : \sigma, \Gamma'' \vdash \mathsf{wf}}{\Gamma', x : \sigma, \Gamma'' \vdash x : \sigma} \; (\text{VAL } x)$$

$$\frac{\Gamma, x : \tau' \vdash E : \tau''}{\Gamma \vdash \lambda x . E : \tau' \to \tau''} \; (\text{VAL ABS})$$

MM-típusrendszer: szabályok összefoglalása (2/2)

$$\frac{\Gamma \vdash E : \tau' \to \tau'', \Gamma \vdash F : \tau'}{\Gamma \vdash E\ F : \tau''} \ (\text{VAL APPL})$$

$$\frac{\Gamma \vdash E : \sigma}{\Gamma \vdash E : \mathit{gen}(\Gamma, \sigma)} \ (\text{VAL GEN}) \qquad \frac{\Gamma \vdash E : \sigma}{\Gamma \vdash E : \mathit{inst}(\sigma)} \ (\text{VAL INST})$$

$$\frac{\Gamma \vdash E : \sigma \qquad \Gamma, x : \sigma \vdash F : \tau}{\Gamma \vdash \texttt{let}\ x = E\ \texttt{in}\ F : \tau} \ (\text{VAL LET POLY})$$

$$\frac{\Gamma, x : \sigma \vdash E : \sigma}{\Gamma \vdash \texttt{fix}\ x.E : \sigma} \ (\text{VAL FIX POLY})$$

MM-típusrendszer: szintaktika-vezérelt változat

Definíció. ***A szintaktika-vezérelt Milner-Mycroft-típusrendszer szabályai***

Ha a szintaktika-vezérelt Hindley-Milner-típusrendszer szabályait átvesszük és kiegészítjük a Milner-Mycroft típusrendszer előző definícióban megadott szabályának módosításával:

$$\frac{\Gamma, \{x : gen(\Gamma, \tau)\} \vdash E : \tau}{\Gamma \vdash \mathtt{fix}\; x.E : inst(gen(\Gamma, \tau))} \text{ (VAL FIX POLY}'\text{)}$$

akkor a szintaktika-vezérelt Milner-Mycroft-típusrendszer szabályait kapjuk meg.

Hindley-Milner kontra Milner-Mycroft: példa

Határozzuk meg az alábbi kifejezés típusait a Hindley-Milner és a Milner-Mycroft típuskikövetkeztetéssel:

$F' \equiv \texttt{fix}\ x.((\lambda y\ z.y)\ (x\ (\lambda u.u))\ (x\ (\lambda v\ w.v)))$

Legyen $u : \tau_u$, $v : \tau_v$, $w : \tau_w$

Hindley-Milner:

$$\begin{array}{lcl} \lambda u.u & : & \tau_u \to \tau_u \\ \lambda v\ w.v & : & \tau_v \to \tau_w \to \tau_v \\ x & : & \alpha \to \beta \end{array}$$

$$\begin{array}{lcl} \tau_u \to \tau_u & \equiv & \tau_v \to \tau_w \to \tau_v \\ \tau_u & = & \tau_v \\ \tau_u & = & \tau_w \to \tau_v \end{array}$$

Hindley-Milner kontra Milner-Mycroft: példa

$$F' \equiv \mathtt{fix}\ x.((\lambda y\ z.y)\ (x\ (\lambda u.u))\ (x\ (\lambda v\ w.v)))$$

Hindley-Milner:

$$\begin{array}{lll} \lambda u.u & : & \tau_u \to \tau_u \\ \lambda v\ w.v & : & \tau_v \to \tau_w \to \tau_v \\ x & : & \alpha \to \beta \end{array} \qquad \begin{array}{lll} \tau_u \to \tau_u & \equiv & \tau_v \to \tau_w \to \tau_v \\ \tau_u & = & \tau_v \\ \tau_u & = & \tau_w \to \tau_v \end{array}$$

Milner-Mycroft:

$$\begin{array}{lll} x & : & \forall\alpha\forall\beta.\alpha \to \beta \\ \lambda y\ z.y & : & (\gamma \to \delta) \to \tau_z \to (\gamma \to \delta) \\ x\ (\lambda u.u) & : & x : (\tau_u \to \tau_u) \to (\gamma \to \delta) \rightsquigarrow \gamma \to \delta \\ x\ (\lambda v\ w.v) & : & x : (\tau_v \to \tau_w \to \tau_v) \to \tau_z \rightsquigarrow \tau_z \end{array}$$

$$F' : \gamma \to \delta \rightsquigarrow \forall\alpha\forall\beta.\alpha \to \beta$$

Hindley-Milner kontra Milner-Mycroft: példa

$$F'' \equiv \texttt{fix}\ x.(\lambda y\ z.(x\ (x\ z\ (\lambda u\ v.u))\ y))$$

Hindley-Milner:

$$\begin{array}{lll} x & : & \tau_z \to (\tau_u \to \tau_v \to \tau_u) \to \tau_1 \\ x & : & \tau_1 \to \tau_y \to \tau_2 \end{array} \qquad \begin{array}{lll} \tau_z & \equiv & \tau_1 \\ \tau_u \to \tau_v \to \tau_u & \equiv & \tau_y \\ \tau_1 & \equiv & \tau_2 \end{array}$$

$$\begin{array}{lll} x\ (x\ z\ (\lambda u\ v.v))\ y & : & \tau_z \\ \lambda y\ z.(x\ (x\ z\ (\lambda u\ v.u))\ y) & : & \tau_y \to \tau_z \to \tau_z \\ \tau_z \to (\tau_u \to \tau_v \to \tau_u) \to \tau_z & \equiv & \tau_y \to \tau_z \to \tau_z \end{array}$$

$$\tau_y = \tau_z = \tau_u \to \tau_v \to \tau_u$$

$$F'' : (\tau_u \to \tau_v \to \tau_u) \to (\tau_u \to \tau_v \to \tau_u) \to (\tau_u \to \tau_v \to \tau_u)$$

Hindley-Milner kontra Milner-Mycroft: példa

$$F'' \equiv \texttt{fix}\ x.(\lambda y\ z.(x\ (x\ z\ (\lambda u\ v.u))\ y))$$

Hindley-Milner:

$$\begin{array}{lll} x & : & \tau_z \to (\tau_u \to \tau_v \to \tau_u) \to \tau_1 \\ x & : & \tau_1 \to \tau_y \to \tau_2 \end{array} \qquad \begin{array}{lll} \tau_z & \equiv & \tau_1 \\ \tau_u \to \tau_v \to \tau_u & \equiv & \tau_y \\ \tau_1 & \equiv & \tau_2 \end{array}$$

$$\begin{array}{lll} x\ (x\ z\ (\lambda u\ v.v))\ y & : & \tau_z \\ \lambda y\ z.(x\ (x\ z\ (\lambda u\ v.u))\ y) & : & \tau_y \to \tau_z \to \tau_z \\ \tau_z \to (\tau_u \to \tau_v \to \tau_u) \to \tau_z & \equiv & \tau_y \to \tau_z \to \tau_z \end{array}$$

$$\tau_y = \tau_z = \tau_u \to \tau_v \to \tau_u$$

$$F'' : (\tau_u \to \tau_v \to \tau_u) \to (\tau_u \to \tau_v \to \tau_u) \to (\tau_u \to \tau_v \to \tau_u)$$

Milner-Mycroft:

$$x \quad : \quad \forall\alpha\forall\beta\forall\gamma.\alpha \to \beta \to \gamma$$

$$\begin{array}{lll} x\ z\ (\lambda u\ v.u) & : & x : \tau_z \to (\tau_u \to \tau_v \to \tau_u) \to \delta_1 \rightsquigarrow \delta_1 \\ x\ (x\ z\ (\lambda u\ v.u))\ y & : & x : \delta_1 \to \tau_y \to \delta_2 \rightsquigarrow \delta_2 \end{array}$$

$$F'' : \tau_y \to \tau_z \to \delta_2 \rightsquigarrow \forall\alpha\forall\beta\forall\gamma.\alpha \to \beta \to \gamma$$