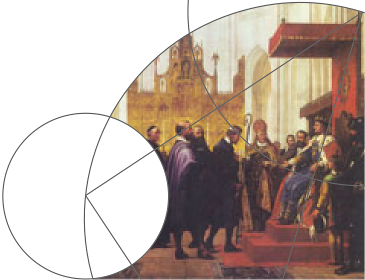
Turtledove

Tool assisted programming in SML, with special emphasis on semi-automatic rewriting to predefined standard forms.

Morten Brøns-Pedersen
Department of Computer Science



Program

- Introduktion.
- Regler for `exists` og `all`.
- Eksempel: `member` er en `exists`-instans.
- Afrunding.

Introduktion

Ideen bag Turtledove

1. Et system til at udvikle front-end-uafhængige refaktoreringer og værktøjer til SML.
2. Et værktøj til at opdage og omskrive funktioner i henhold til et antal foruddefinerede skabeloner.

Nærmeste „konkurrent“: HaRe

1. Implementerer mange forskellige refaktoreringer.
2. Kun „simple“ refaktoreringer.
3. Layoutbevarende.

Regler for `exists` og `all`

exists_a

$$\begin{array}{ll} \mathcal{C}[\overline{x} \texttt{ :: xs}] & \Rightarrow \mathbb{D}(\mathcal{C}[\overline{x}]) \texttt{ orelse self}(\mathcal{C}[\texttt{xs}]) \\ \mathcal{D} & \Rightarrow \texttt{false} \end{array}$$

$$\text{where samedom}(\mathcal{C}, \mathcal{D})$$

$$\Downarrow$$

$$\mathcal{C}[\texttt{xs}] \Rightarrow \texttt{List.exists (fn a => } \mathbb{D}(\mathcal{C}[\text{a}])\texttt{) xs}$$

all_a

$$\begin{array}{ll} \mathcal{C}[\overline{x} \texttt{ :: xs}] & \Rightarrow \mathbb{D}(\mathcal{C}[\overline{x}]) \texttt{ andalso self}(\mathcal{C}[\texttt{xs}]) \\ \mathcal{D} & \Rightarrow \texttt{true} \end{array}$$

$$\text{where samedom}(\mathcal{C}, \mathcal{D})$$

$$\Downarrow$$

$$\mathcal{C}[\texttt{xs}] \Rightarrow \texttt{List.all (fn a => } \mathbb{D}(\mathcal{C}[\text{a}])\texttt{) xs}$$

Eksempel

```
fun member (_, nil) = false
  | member (x, y :: ys) = x = y orelse member (x, ys)

fun sublist (nil, _) = true
  | sublist (x :: xs, ys) = member (x, ys) andalso sublist (xs, ys)
```

kan skrives om til

```
fun member (x, ys) = List.exists (fn y => x = y) ys

fun sublist (xs, ys) = List.all (fn x => member (x, ys)) xs
```

Eksempel — Normalform

```
fun member (_, nil) = false
  | member (x, y :: ys) = x = y orelse member (x, ys)
```

```
fun member (x, y :: ys) = x = y orelse member (x, ys)
  | member x = false
```

```
fun sublist (nil, _) = true
  | sublist (x :: xs, ys) = member (x, ys) andalso sublist (xs, ys)
```

```
fun sublist (x :: xs, ys) = member (x, ys) andalso sublist (xs, ys)
  | sublist x = true
```

Eksempel — Normalform

Live demo

Eksempel — Omskrivning

Først undersøges om funktionen er en instans af reglen. For hvert klausulpar skal eksistere en afledning

$$\frac{\begin{array}{c}\mathcal{H}\\ \sigma(spat) = mpat\end{array} \qquad \begin{array}{c}\mathcal{I}\\ pat : \langle mpat,\ \theta\rangle\end{array} \qquad \begin{array}{c}\mathcal{J}\\ \sigma,\ \theta \vdash sexp : exp\end{array}}{\sigma \vdash pat\ \texttt{=>}\ exp : spat \Rightarrow sexp}$$

Jeg demonstrerer omskrivning af `member`.
Lad

$$\sigma = \left\{ \begin{array}{lcr} \mathcal{C} & \mapsto & (\overline{a},\ \diamond_1) \\ \mathcal{D} & \mapsto & \overline{a} \\ \mathbb{E} & \mapsto & ((\texttt{v},\ \texttt{w}), \texttt{v = w}) \\ \textsf{self} & \mapsto & (\texttt{member}, 1) \end{array} \right\}$$

Eksempel — Omskrivning

```
fun member (x, y :: ys) = x = y orelse member (x, ys)
  | member x = false
```

$$
\begin{aligned}
\mathcal{C}[\overline{x}\ \mathtt{::}\ \mathtt{xs}] &\Rightarrow \mathbb{D}(\mathcal{C}[\overline{x}])\ \mathtt{orelse}\ \mathtt{self}(\mathcal{C}[\mathtt{xs}]) \\
\mathcal{D} &\Rightarrow \mathtt{false}
\end{aligned}
$$

Eksempel — Omskrivning

$$\mathcal{H}: \frac{}{\sigma(\mathcal{C}[\overline{x} \ :: \ \mathtt{xs}]) = (\overline{a},\ \diamond_1)[\overline{x} \ :: \ \mathtt{xs}/\diamond_1]}$$

$$\mathcal{I}: \frac{\dfrac{}{\mathtt{x} : \langle \overline{a},\ \{\overline{a} \mapsto \mathtt{x}\}\rangle} \qquad \dfrac{\dfrac{}{\mathtt{y} : \langle \overline{x},\ \{\overline{x} \mapsto \mathtt{y}\}\rangle} \qquad \dfrac{}{\mathtt{ys} : \langle \mathtt{xs},\ \emptyset\rangle}}{\mathtt{y} \ :: \ \mathtt{ys} : \langle \overline{x} \ :: \ \mathtt{xs},\ \{\overline{x} \mapsto \mathtt{y}\}\rangle}}{(\mathtt{x},\ \mathtt{y} \ :: \ \mathtt{ys}) : \langle (\overline{a},\ \overline{x} \ :: \ \mathtt{xs}),\ \{\overline{a} \mapsto \mathtt{x}, \overline{x} \mapsto \mathtt{y}\}\rangle}$$

Og vi identificerer xs med ys.

Eksempel — Omskrivning

Lad $\theta = \{\overline{a} \mapsto \mathtt{x}, \overline{x} \mapsto \mathtt{y}\}$ og husk at vi identificerer $\mathtt{xs}$ med $\mathtt{ys}$.

$$\mathcal{J}_1 : \dfrac{\dfrac{\dfrac{\dfrac{}{\sigma,\ \theta \vdash \overline{a} : \mathtt{x}} \qquad \dfrac{}{\sigma,\ \theta \vdash \overline{x} : \mathtt{y}}}{\sigma,\ \theta \vdash (\overline{a},\ \diamond_1)[\overline{x}/\diamond_1] : (\mathtt{x},\ \mathtt{y})}}{\sigma,\ \theta \vdash \mathcal{C}[\overline{x}] : (\mathtt{x},\ \mathtt{y})}}{\sigma,\ \theta \vdash \mathbb{D}(\mathcal{C}[\overline{x}]) : (\mathtt{v = w})[(\mathtt{x},\ \mathtt{y})/(\mathtt{v},\ \mathtt{w})]}$$

$$\mathcal{J}_2 : \dfrac{\dfrac{\dfrac{\dfrac{}{\sigma,\ \theta \vdash \overline{a} : \mathtt{x}} \qquad \dfrac{}{\sigma,\ \theta \vdash \mathtt{xs} : \mathtt{ys}}}{\sigma,\ \theta \vdash (\overline{a},\ \diamond_1)[\mathtt{xs}/\diamond_1] : (\mathtt{x},\ \mathtt{ys})}}{\sigma,\ \theta \vdash \mathcal{C}[\mathtt{xs}] : (\mathtt{x},\ \mathtt{ys})}}{\sigma,\ \theta \vdash \mathtt{self}(\mathcal{C}[\mathtt{xs}]) : (\mathtt{member\ x})[(\mathtt{x},\ \mathtt{ys})/\mathtt{x}]}$$

$$\mathcal{J} : \dfrac{\mathcal{J}_1 \qquad\qquad \mathcal{J}_2}{\sigma,\ \theta \vdash \mathbb{D}(\mathcal{C}[\overline{x}])\ \mathtt{orelse\ self}(\mathcal{C}[\mathtt{xs}]) : \mathtt{x = y\ orelse\ member\ (x,\ ys)}}$$

Eksempel — Resultat

$$
\begin{array}{ll}
\mathcal{C}[\overline{x}\ \texttt{::}\ \texttt{xs}] & \Rightarrow \mathbb{D}(\mathcal{C}[\overline{x}])\ \texttt{orelse}\ \texttt{self}(\mathcal{C}[\texttt{xs}]) \\
\mathcal{D} & \Rightarrow \texttt{false}
\end{array}
$$

$$\text{where samedom}(\mathcal{C}, \mathcal{D})$$

$$\Downarrow$$

$$\mathcal{C}[\texttt{xs}] \Rightarrow \texttt{List.exists (fn a => } \mathbb{D}(\mathcal{C}[\texttt{a}])\texttt{) xs}$$

$$
\frac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{}{\sigma,\ \theta \vdash \overline{a} : \texttt{x}}
\qquad
\dfrac{}{\sigma,\ \theta \vdash \texttt{a} : \texttt{a}}
}{\sigma,\ \theta \vdash (\overline{a},\ \diamond_1)[\texttt{a}/\diamond_1] : (\texttt{x, a})}
}{\sigma,\ \theta \vdash \mathcal{C}[\texttt{a}] : (\texttt{x, a})}
}{\sigma,\ \theta \vdash \mathbb{D}(\mathcal{C}[\texttt{a}]) : (\texttt{v = w})[(\texttt{x, a})/(\texttt{v, w})]}
\qquad
\dfrac{}{\sigma,\ \theta \vdash \texttt{xs} : \texttt{ys}}
}{\sigma,\ \theta \vdash \texttt{List.exists (fn a => } \mathbb{D}(\mathcal{C}[\texttt{a}])\texttt{) xs} : \texttt{List.exists (fn a => x = a) ys}}
$$

```
fun member (x, ys) = List.exists (fn a => x = a) ys
```

Fejl i normalform

```
fun foo (x :: y :: ys) = x + 42 :: foo (y :: ys)
  | foo _ = nil
```

```
fun foo (x :: c) = x + 42 :: foo c
  | foo x = nil
```

Løsning: Sørg for at alle konstruktører fra datatypen eksisterer før et delmønster generaliseres.

```
fun foo (x :: y :: ys) = x + 42 :: foo (y :: ys)
  | foo (x :: nil) = x + 42 :: foo nil
  | foo _ = nil
```

Afrunding

- Forholdsvist let at lave simple værktøjer:
  - Udtræk „todo“-lister.
  - Omdøb variable.
  - Auto-completion.
  - Gå til definition.
  - Indfør ekstra funktionparameter.
  - Generér stubimplementering fra signatur.
  - ...
- Svært at lave skabelonbaseret omkskrivninger:
  - Semantikbevarelse.
  - Det samme program kan skrives på mange måder. Skabeloner skal ramme en balance hvor flest mulige programmer med samme betydning dækkes, men ingen med forskellig.