Fundamentals of Programming Languages

Assignment 3

Records and Patterns

Guilherme João Correia Lopes
fc52761

2022/2023

1 Record Patterns

Assuming all the problem context given for the questions in group 1, we have the following as the answer for each question:

A. Distinct variables in patterns

$$\frac{}{\mathtt{x} : \{\mathtt{x}\}}\ \text{PW-VAR} \qquad \frac{\text{for each } \mathtt{i},\ \mathtt{p_i} : \chi_\mathtt{i} \quad \text{for each } \mathtt{i}, \mathtt{j},\ \mathtt{i} \neq \mathtt{j} \Rightarrow \chi_\mathtt{i} \cap \chi_\mathtt{j} = \emptyset}{\{\mathtt{l_i} = \mathtt{p_i}^{\,\mathtt{i} \in 1..\mathtt{n}}\} : \chi_1 \cup \ldots \cup \chi_\mathtt{n}}\ \text{PW-RCD}$$

A collection of k items are pairwise distinct if no two of them are equal to one another. This is reflected in PW-RCD rule with the logic of intersecting each set of variables χ_i with every other set of variables χ_j, if $i \neq j$. If every intersection results in a empty set ($\emptyset$), then it is pairwise distinct.
For curiosity, I tried to implement these rules in the Haskell program for the implementation section, in the bottom of the code delivered.

B. A good pattern and a not so good pattern

$$\dfrac{\dfrac{}{\mathtt{x} : \{\mathtt{x}\}}\ \text{PW-VAR} \qquad \dfrac{\dfrac{}{\mathtt{z} : \{\mathtt{z}\}}\ \text{PW-VAR}}{\{\mathtt{n} = \mathtt{z}\} : \{\mathtt{z}\}}\ \text{PW-RCD} \qquad \{\mathtt{x}\} \cap \{\mathtt{z}\} = \emptyset \qquad \{\mathtt{z}\} \cap \{\mathtt{x}\} = \emptyset}{\{\mathtt{l} = \mathtt{x},\ \mathtt{m} = \{\mathtt{n} = \mathtt{z}\}\} : \{\mathtt{x}, \mathtt{z}\}}\ \text{PW-RCD}$$

The following derivation is not possible for any χ:

$$\dfrac{\dfrac{}{\mathtt{x} : \{\mathtt{x}\}}\ \text{PW-VAR} \qquad \dfrac{\dfrac{}{\mathtt{x} : \{\mathtt{x}\}}\ \text{PW-VAR}}{\{\mathtt{n} = \mathtt{x}\} : \{\mathtt{x}\}}\ \text{PW-RCD} \qquad \{\mathtt{x}\} \cap \{\mathtt{x}\} \neq \emptyset \qquad \{\mathtt{x}\} \cap \{\mathtt{x}\} \neq \emptyset}{\{\mathtt{l} = \mathtt{x},\ \mathtt{m} = \{\mathtt{n} = \mathtt{x}\}\} : \chi}\ \text{PW-RCD}$$

As we can observe in the derivation attempt, the recursive intersection premises correspondent in $i = 0$ and $i = 1$, in other words, both $\{x\} \cap \{x\} \neq \emptyset$, betray the PW-RCD rule because the intersection doesn't result in a empty set, making this derivation to be "impossible" for any χ and all the variables in the pattern are not pairwise distinct.

For both the derivations, when applying PW-RCD rule a second time, the second "for each" premise doesn't apply, because the Record is a single element only and so there isn't a χ_i and χ_j given that $i \neq j$, and so the application $\chi_i \cap \chi_j = \emptyset$ isn't possible. Regardless, the premise is true.

C. Typing let expressions with patterns

$$\text{P-VAR}\ \frac{}{\vdash \mathtt{x} : \mathsf{T} \Rightarrow \mathtt{x} : \mathsf{T}} \qquad \text{P-RCD}\ \frac{\text{for each } \mathtt{i},\ \ \vdash \mathtt{p_i} : \mathsf{T_i} \Rightarrow \Delta_i}{\vdash \{\mathtt{l_i} = \mathtt{p_i}^{\mathtt{i}\in 1..\mathtt{n}}\} : \{\mathtt{k_j} : \mathsf{T_j}^{\mathtt{j}\in 1..\mathtt{m}}\} \Rightarrow \Delta_1, \ldots, \Delta_n}$$

$$\text{T-LET}\ \frac{\Gamma \vdash \mathtt{t_1} : \mathsf{T_1} \qquad \Gamma, \Delta \vdash \mathtt{t_2} : \mathsf{T_2} \qquad \vdash \mathtt{p} : \mathsf{T_1} \Rightarrow \Delta}{\Gamma \vdash \mathtt{let\ p} = \mathtt{t_1\ in\ t_2} : \mathsf{T_2}}$$

P-VAR and P-RCD rules define the "pattern typing" relation that takes a pattern and a type, returning a context defined here as Δ, with its bindings for the variables in the pattern. In T-LET we add the Δ context to the original context defined by Γ.

D. Preservation and progress for let expressions with patterns

AUXILIARY LEMMAS

1. Weakening Lemma: If $\Gamma \vdash \mathtt{t} : \mathsf{T}$ and $\vdash \mathtt{p} : \mathsf{T} \Rightarrow \Delta$, then $match(p, t) = \sigma$, with $\Gamma \vdash \sigma \models \Delta$
2. Extending the Substitution Lemma: $\Gamma, \Delta \vdash \mathtt{t} : \mathsf{T}$ and $\Gamma \vdash \sigma \models \Delta$, then $\Gamma \vdash \sigma t : \mathsf{T}$

PROGRESS
[Proof] By induction on a derivation of $t : T$.

Case T-LET:
$t = let\ p = t_1\ in\ t_2,\ \Gamma \vdash t_1 : T_1,\ \Gamma \vdash p : T_1 \Rightarrow \Delta$ and $\Gamma, \Delta \vdash t_2 : T_2$
By induction hypothesis, either t_1 is a value or else there is some t_1' such that $\mathtt{t_1} \rightarrow \mathtt{t_1'}$. With that said, we have the following:

1. If t_1 is a value, then apply E-LETV and $\mathtt{t}$ reduces to $\mathtt{t'}$ by T-LET.
2. If $\mathtt{t_1} \rightarrow \mathtt{t_1'}$, then apply E-LET and $\mathtt{t}$ reduces to $\mathtt{t'}$ by T-LET.

PRESERVATION
[Proof] By induction on a derivation of $t : T$. At each step of the induction, we assume that the desired property holds for all subderivations (i.e., that if $\mathtt{s} : \mathsf{S}$ and $\mathtt{s} \rightarrow \mathtt{s'}$, then $\mathtt{s'} : \mathsf{S}$ whenever $\mathtt{s} : \mathsf{S}$ is proved by a subderivation of the present one) then and proceed by case analysis on the final rule in the derivation.

Case T-LET:
$t = let\ p = t_1\ in\ t_2,\ \Gamma \vdash t_1 : T_1,\ \Gamma \vdash p : T_1 \Rightarrow \Delta$ and $\Gamma, \Delta \vdash t_2 : T_2$
If the last rule in the derivation is T-LET, by inspecting the evaluation rules, we find that there are two rules, E-LET and E-LETV, that can be used to derive $\mathtt{t} \rightarrow \mathtt{t'}$.

1. E-LETV: $\mathtt{t'} = \mathtt{match(p, v_1)\ t_2}$
   Since we know $\Gamma \vdash t_1 : T_1$ and $\Gamma \vdash p : T_1 \Rightarrow \Delta$, we can apply the weakening lemma to obtain $match(p, v1) = \sigma$, with $\Gamma \vdash \sigma \models \Delta$, in other words, if σ is a substitution and Δ is a context with the same domain as σ, then $\Gamma \vdash \sigma \models \Delta$ means that, for each $x \in dom(\Delta)$, we have $\Gamma \vdash \sigma(\mathtt{x}) : \Delta(\mathtt{x})$. Since, it also known that $\Gamma, \Delta \vdash t_2 : T_2$ and $\Gamma \vdash \sigma \models \Delta$, we apply the substitution lemma to obtain $\Gamma \vdash \sigma t : \mathsf{T}$.

2. E-LET: $\mathtt{t}_1 \rightarrow \mathtt{t}_1'$, $\mathtt{t}' = \mathtt{let\ p} = \mathtt{t}_1'\ \mathtt{in}\ \mathtt{t}_2$
   Since we know $\Gamma \vdash t_1 : \mathsf{T}$, we can apply the induction hypothesis to obtain $\Gamma \vdash t_1' : \mathsf{T}$, from which we obtain $\mathtt{let\ p} = \mathtt{t}_1'\ \mathtt{in}\ \mathtt{t}_2 : \mathsf{T}$, i.e., $\mathtt{t}' : \mathsf{T}$, by applying rule T-LET.
   We obtain $\Gamma \vdash [x \rightarrow t_2]t_1' : \mathsf{T}$, i.e., $\mathtt{t}' : \mathsf{T}$, by applying the Preservation of Types Under Substitution, in which its result preserves the type T.

2 Implementation

Reminder: Due to being in doubt about maintaining the original Let binding from the book's section 11.5, it was also implemented in the code. If this was not necessary, this implementation should be ignored for clarity purposes.
As mentioned previously, it was also implemented the vars-in-patterns rule, from exercise 1.A, in the bottom of the code.