Secondo progetto intermedio

Samuele Bonini

Sommario

Relazione e documentazione della soluzione presentata al secondo progetto intermedio del corso di Programmazione 2.

Indice

1 Regole operazionali per set

Il tipo `set` è definito nel seguente modo:

⟨*set_*⟩ ::= Empty '`of`' ⟨*TypeLabel*⟩
| Set '`of`' ⟨*evT list*⟩ '`*`' ⟨*TypeLabel*⟩

`TypeLabel` indica l'etichetta del tipo del set.

Sono presenti due espressioni che, quando valutate, si comportano da costruttori per il tipo set. È possibile definire un insieme vuoto specificandone il tipo, con la seguente semantica operazionale:

$$\frac{e_1 \Longrightarrow TypeLabel}{\texttt{empty(e1)} \Longrightarrow (\emptyset, TypeLabel)}$$

È altresì possibile definire un *singleton*, ovvero un insieme contenente un solo elemento, specificandone il tipo e fornendo un'espressione:

$$\frac{e_1 \Longrightarrow TypeLabel \quad e_2 \Longrightarrow v \quad e_2 \overset{\text{typeof}}{\Longrightarrow} TypeLabel}{\texttt{singleton(e1)} \Longrightarrow (\{v\}, TypeLabel)}$$

Dove $e_2 \overset{\text{typeof}}{\Longrightarrow} TypeLabel$ indica che la chiamata a `typeof` (discusso nel **paragrafo 3**) applicata al risultato della valutazione di e_2 restituisce il tipo `TypeLabel` (lo stesso dell'insieme).

2 Regole operazionali per le funzioni su set

Segue la semantica operazionale delle operazioni su insiemi definite nell'interprete.

Appartenenza all'insieme:

$$\frac{e_1 \Longrightarrow v \quad e_2 \Longrightarrow Set}{\texttt{belongsToSet e1 e2} \Longrightarrow v \in Set}$$

Aggiunta di un elemento:

$$\frac{e_1 \Longrightarrow v \quad e_2 \Longrightarrow Set}{\texttt{addToSet e1 e2} \Longrightarrow Set \cup \{v\}}$$

Rimozione di un elemento:

$$\frac{e_1 \Longrightarrow v \quad e_2 \Longrightarrow Set}{\texttt{addToSet e1 e2} \Longrightarrow Set \backslash v}$$

Verifica della relazione di sottoinsieme:

$$\frac{e_1 \Longrightarrow Set_1 \quad e_2 \Longrightarrow Set_2}{\texttt{isSusbset e1 e2} \Longrightarrow Set_1 \subseteq Set_2}$$

Unione insiemistica:

$$\frac{e_1 \Longrightarrow Set_1 \quad e_2 \Longrightarrow Set_2}{\texttt{union e1 e2} \Longrightarrow Set_1 \cup Set_2}$$

Intersezione insiemistica:

$$\frac{e_1 \Longrightarrow Set_1 \quad e_2 \Longrightarrow Set_2}{\texttt{intersection e1 e2} \Longrightarrow Set_1 \cap Set_2}$$

Differenza insiemistica:

$$\frac{e_1 \Longrightarrow Set_1 \quad e_2 \Longrightarrow Set_2}{\texttt{difference e1 e2} \Longrightarrow Set_1 \backslash Set_2}$$

Verifica di vuoto:

$$\frac{e_1 \Longrightarrow Set}{\texttt{isEmpty e1} \Longrightarrow Set = \emptyset}$$

Valore massimo:

$$\frac{e_1 \Longrightarrow Set}{\texttt{maxValue e1} \Longrightarrow \max\{Set\}}$$

Valore minimo:

$$\frac{e_1 \Longrightarrow Set}{\texttt{minValue e1} \Longrightarrow \min\{Set\}}$$

Quantificatore universale:

$$\frac{e_1 \Longrightarrow fx \quad e_2 \Longrightarrow Set}{\texttt{forall e1 e2} \Longrightarrow \forall x \in Set \ . \ f(x)}$$

Quantificatore esistenziale:

$$\frac{e_1 \Longrightarrow fx \quad e_2 \Longrightarrow Set}{\texttt{exists e1 e2} \Longrightarrow \exists x \in Set \ . \ f(x)}$$

Map:

$$\frac{e_1 \Longrightarrow fx \quad e_2 \Longrightarrow Set}{\texttt{map e1 e2} \Longrightarrow Set' \ . \ (\forall x \in Set \ . \ f(x) \in Set')}$$

Filter:

$$\frac{e_1 \Longrightarrow fx \quad e_2 \Longrightarrow Set}{\texttt{filter e1 e2} \Longrightarrow Set' \ . \ (\forall x \in Set \ . \ x \in Set' \Longleftrightarrow f(x))}$$

3 Considerazioni sul type checker

Avendo introdotto `evT` per set e stringhe, la funzione `typecheck` è stata estesa per riflettere questi cambiamenti.

Dato che è stato anche definito un tipo `typename`, che agisce come etichetta per i nomi dei tipi, è stata definita una funzione ausiliaria `typeof`. Essa è utilizzata per ottenere a *run-time* il tipo degli `evT` utilizzati all'interno dei set sotto forma di etichetta di tipo `typename`.

Questa scelta permette di effettuare più agevolmente il controllo dei tipi sugli argomenti delle funzioni di set. In assenza di `typeof`, sarebbe necessario effettuare *pattern matching* sull'etichetta di tipo del set e poi verificare il valore di ritorno di `typecheck` chiamandolo con una stringa opportunamente scelta in base all'etichetta (per esempio, `"string"`, se la funzione *matcha* l'etichetta `StringType`).

4 Nota sui test

Il file `testcases.ml` contiene una ricca suite di test che mostrano le potenzialità delle funzioni implementate. Per mantenere i test leggibili, nel file essi vengono chiamati operando direttamente sugli `evT`, senza passare per la dichiarazione di un'espressione e la successiva chiamata a `eval`.

All'interno del file dell'inteprete sono comunque presenti un paio di esempi di test per ciascuna delle funzionalità implementate, che dimostrano il corretto funzionamento con le espressioni definite. I test in `testcases.ml` vogliono invece essere una serie di test più approfonditi volti a mostrare la correttezza delle funzioni, esplorando casi limite con una varietà di input.