Relazione sul secondo progetto intermedio relativo alla realizzazione di un interprete in Ocaml

Programmazione II – A.A. 2020/21

Corso A

Istruzioni per l'esecuzione:

cd src
ocamlc setlang.ml
./a.out

Scelte implementative

La traccia richiedeva di estendere il linguaggio didattico con la possibilità di manipolare insiemi omogenei. Dato che la maggior parte delle operazioni implica il bisogno di confrontare gli elementi degli insiemi tra di loro, si è deciso di permettere la creazione e la gestione di insiemi di tipo comparabile (quindi Bool, String e Int). Per fare ciò, è stato definito un tipo lang_type che permette di dichiarare il tipo degli elementi di un insieme, questo per garantire che gli insiemi siano omogenei, dato che altrimenti sarebbe possibile usare espressioni che potrebbero essere valutate a valori di tipo diverso tra loro.
Si è inoltre deciso di includere il lang_type delle espressioni che valutano a insiemi nel valore stesso degli insiemi, così da poter permettere all'utente di effettuare dei controlli qualora disponesse di diversi tipi di insieme.
Al momento della valutazione di un insieme, tale insieme viene anche validato tramite la funzione validate: si verifica quindi che i tipi delle espressioni fornite valutino a valori di tipo coerente con quanto dichiarato nella formazione dell'espressione e che tali espressioni non valutino a valori uguali tra loro. In tal caso, prendendo ispirazione dal modo in cui linguaggi come C# permettono la creazione di un insieme a partire da una lista (che può contenere duplicati), non si solleva un'eccezione, ma semplicemente si scartano i duplicati.

Operazioni sugli insiemi

Le operazioni richieste dalla traccia sono implementate nella loro ovvia interpretazione. Nel caso degli operatori funzionali (ForAll, Exists, Map e Filter), è stata definita una funzione facente parte dell'RTS chiamata rebuild_set che, dato un valore insiemistico, fornisce la relativa espressione: questo perché la valutazione del predicato o della funzione richiede un'espressione, ma allo stesso tempo è necessario valutare l'espressione insiemistica per poterla validare tramite la funzione validate. Tale approccio permette anche di gestire tutti gli altri costruttori di insieme (Of, Singleton ed Empty) allo stesso modo, senza dover fare pattern matching con ognuno di essi.

Se nella Filter, nella Exists e nella ForAll il tipo dell'insieme di ritorno è ovvio (il tipo dell'insieme passato come parametro nel primo caso e un Bool negli altri), la gestione del tipo del valore insiemistico di ritorno nel caso della Map è più spinosa, dato che deve restituire il tipo di ritorno dell'operatore. In caso di lista non vuota, si valuta l'operazione sulla prima espressione dell'insieme, usando poi la funzione get_lang_type per ottenere il

tipo insiemistico corrispondente; in caso di lista vuota, invece, si è deciso di utilizzare il tipo dell’insieme passato come parametro.