Alba Bisante, Michele Ferraro
Corso di Linguaggi di Programmazione - A.A. 2017/18
14 giugno 2018

Relazione Progetto “Mini Rust”

Implementazione in SML del Borrowing

Introduzione

Questo progetto si pone l’obiettivo di implementare in SML i meccanismi di funzionamento del “Borrowing”, un aspetto particolarmente interessante del linguaggio di programmazione Rust. Per rendere possibile tale implementazione, abbiamo scelto di realizzare un modello di valutazione semplificato capace di valutare rappresentazioni algebriche di semplici programmi.

Il primo passo dell’elaborazione del nostro linguaggio, da considerarsi un “Mini Rust” in grado di usufruire del meccanismo di Borrowing, è stato realizzare un’algebra dei programmi, costituiti da istruzioni, variabili, espressioni, valori (costanti e riferimenti).

Nel presente documento esponiamo la grammatica del nostro linguaggio, la semantica statica, la semantica dinamica ed i domini di valutazione utilizzati. Per valutare le capacità del linguaggio proposto, vengono fornite dunque anche una serie di funzioni di valutazione, che associano variabili a locazioni, valori, tipi, oppure espressioni a valori, e così via. Al livello di astrazione più alto, è disponibile la seguente funzione per la valutazione di programmi.

evalProgram : (p, s, b) -> (s’, b’), con p ∈ Program, s, s’ ∈ Storage, b, b’ ∈ Borrows

In secondo luogo, proponiamo una panoramica dell’organizzazione del codice in SML, cui alleghiamo la sorgente in un documento apposito dove vi è una anche una breve trattazione di alcuni esempi per poter al meglio chiarificare il meccanismo di Borrowing implementato.

Sintassi

(Tipi)

T ::= int | mut int | &T | &mut T

(Istruzioni)

I ::= [let] x = e; | [let] mut x = e; | x = e; | *x = e;

(Espressioni)

d, e ::= x | n | &x | &mut x | *x

Il linguaggio di programmazione Rust prevede la possibilità di effettuare binding sia immutabili (di default), sia mutabili. Per ragioni di chiarezza della trasposizione in SML del linguaggio abbiamo deciso di differenziare la mutabilità o meno direttamente nel dominio dei Tipi. Inoltre, per semplicità, ci limitiamo a costanti intere.

Semantica Statica

$$\Gamma 1, z : T, \Gamma 2 \vdash z : T$$

$$\Gamma \vdash n: int^*$$

**Qualunque sia il contesto, le costanti hanno tipo int.*

$$\frac{\Gamma \vdash z : T}{\Gamma \vdash \&z : \&T}$$

$$\frac{\Gamma \vdash z : T}{\Gamma \vdash \&mut\ z : \&mut\ T}$$

$$\frac{\Gamma \vdash x : \&T}{\Gamma \vdash *x : T}$$

$$\frac{\Gamma \vdash x : \&\text{mut } T}{\Gamma \vdash *x : \text{mut } T}$$

Semantica Dinamica

Domini di Valutazione

Val = mut int U ConstRef U MutRef

Types = { TypeI, TypeMutRef, TypeConstRef }

Storage : Loc -> Val

Borrows : Loc -> int

VarLoc : Var -fin-> Loc

VarVal : Var*Storage -> Val (VarVal(v,s) := Storage(VarLoc(v), s))

Relazioni di Valutazione

Distinguiamo dal contesto le relazioni di valutazione di espressioni e di programmi.

$\rightarrow$: Expr * Storage * Borrows -> Val

$\rightarrow$: Programs * Storage -> Storage * Borrows

Regole della Semantica

Per semplicità, assumiamo che B(v), con B ∈ Borrows e v ∈ Var, sia valutata come B(VarLoc(v)).
La nozione di ambiente è rappresentata dalla relazione VarVal.

$$E \vdash z, S \rightarrow v \quad (\text{se } E(z) = v)$$

$$E \vdash n, S \rightarrow n$$

$$\frac{E \vdash e, S \rightarrow v \qquad E\ \{ (x,v) \} \vdash [\ I\ ], S, B \rightarrow S', B'}{E \vdash x = e; [\ I\ ], S, B \rightarrow S', B'}$$

(se $x \in$ mutInt and $B(x) = 0$)

$$\frac{E\ \{ (x, \&y) \} \vdash [\ I\ ], S, B\ \{ (y, B(y) + 1) \} \rightarrow S', B'}{E \vdash x = \&y; [\ I\ ], S, B \rightarrow S', B'}$$

(se $(x \in$ ConstRef) and $B(x) = 0$ and $B(y) >= 0$ and $\nexists z \mid (x,\&z) \in E$)

$$\frac{E\ \{ (x, \&y) \} \vdash [\ I\ ], S, B\ \{ (y, B(y) + 1), (z,n\text{-}1) \} \rightarrow S', B'}{E\ \{ (x, \&z) \} \vdash x = \&y; [\ I\ ], S, B\ \{ (z,n) \} \rightarrow S', B'}$$

(se $(x \in$ ConstRef) and $B(x) = 0$ and $B(y) >= 0$ and $n > 0$)

$$\frac{E\ \{ (x, \&mut\ y) \} \vdash [\ I\ ], S, B\ \{ (y, -1) \} \rightarrow S', B'}{E \vdash x = \&mut\ y; [\ I\ ], S, B \rightarrow S', B'}$$

(se $(x \in$ MutRef) and $B(x) = 0$ and $B(y) = 0$ and $\nexists z \mid (x,\&mut\ z) \in E$)

$$\frac{E\ \{ (x, \&mut\ y) \} \vdash [\ I\ ], S, B\ \{ (y, -1) \} \rightarrow S', B'}{E\{ (x, \&mut\ z) \} \vdash x = \&mut\ y; [\ I\ ], S, B\ \{ (z,-1) \} \rightarrow S', B'}$$

(se $(x \in$ MutRef) and $B(x) = 0$ and $B(y) = 0$)

$$\frac{E \vdash e, S \rightarrow v \qquad E\ \{ (x, \&mut\ y), (y, v) \} \vdash [\ I\ ], S, B \rightarrow S', B'}{E \vdash *x = e; [\ I\ ], S \rightarrow S', B'}$$