Γλώσσες Προγραμματισμού 2
Άσκηση 2: Συστήματα τύπων – σημασιολογία μεγάλων βημάτων
Μπενετόπουλος Αχιλλεύς, 03112614

Η σύνταξη της γλώσσας πάνω στην οποία θα δουλέψουμε έχει ως εξής:

$$\tau ::= \mathrm{Int} \mid Bool \mid \tau_1 \tau_2 \mid Unit \mid Ref\,\tau$$
$$u ::= n \mid true \mid false \mid \lambda x:\tau.e \mid loc_i \mid unit$$
$$e ::= u \mid -e \mid e_1 + e_2 \mid \ldots \mid \neg e \mid e_1 \wedge e_2 \mid \ldots \mid e_1 < e_2 \mid if\,e\,then\,e_1\,else\,e_2 \mid ref\,e \mid !\,e \mid e_1 := e_2$$

Σημασιολογία Μεγάλων Βημάτων

Unary & Binary Operators

$$\frac{(e,m) \Downarrow (u,m') \quad (\square u, m') \Downarrow ([[\square]]u, m')}{(\square e, m) \Downarrow ([[\square]]u, m')}$$

$$\frac{(e_1,m) \Downarrow (u_1,m'') \quad (e_2,m'') \Downarrow (u_2,m') \quad (u_1 \circ u_2, m') \Downarrow ([[\circ]](u_1,u_2), m')}{(e_1 \circ e_2, m) \Downarrow ([[\circ]](u_1,u_2), m')}$$

Conditionals

$$\frac{(e,m) \Downarrow (true,m'') \quad (e_1,m'') \Downarrow (u_1,m')}{(if\,e\,then\,e_1\,else\,e_2, m) \Downarrow (u_1,m')} \qquad \frac{(e,m) \Downarrow (false,m'') \quad (e_2,m'') \Downarrow (u_2,m')}{(if\,e\,then\,e_1\,else\,e_2, m) \Downarrow (u_2,m')}$$

Call by Value

$$\frac{(e_1,m'') \Downarrow (\lambda x:\tau.e, m''') \quad (e_2,m) \Downarrow (u_2,m'') \quad (e[x:=u_2], m''') \Downarrow (u,m')}{(e_1 e_2, m) \Downarrow (u,m')}$$

Call by Name

$$\frac{(e_1,m) \Downarrow (\lambda x:\tau.e, m'') \quad (e[x:=e_2], m'') \Downarrow (u,m')}{(e_1 e_2, m) \Downarrow (u,m')}$$

Memory accesses

$$\frac{(e,m) \Downarrow (loc_i, m') \quad m'(i) = u}{(!e, m) \Downarrow (u,m')}$$

$$\frac{(e_2,m) \Downarrow (u_2,m'') \quad (e_1,m'') \Downarrow (loc_j,m')}{(e_1 := e_2, m) \Downarrow (unit, m\{j \mapsto u\})} \qquad \frac{(e,m) \Downarrow (u,m') \quad j = max(dom(m')) + 1}{(ref\ e, m) \Downarrow (loc_j, m'\{j \mapsto u\})}$$

Για να διατυπώσουμε το θεώρημα ασφάλειας, θα πρέπει πρώτα να διατυπώσουμε τα εξής δύο θεωρήματα:

Θεώρημα Προόδου: Αν $\varnothing ; M \vdash e : \tau$, τότε είτε e τιμή, είτε για κάθε m τέτοιο ώστε $\varnothing \vdash m : M$ υπάρχει (u, m') τέτοιο ώστε $(e,m) \Downarrow (u,m')$.

<u>Θεώρημα Διατήρησης:</u> Αν $\Gamma;M \vdash e:\tau,\ \Gamma \vdash m:M$ και $(e,m) \Downarrow (u,m')$, τότε υπάρχει M' τέτοιο ώστε:

$$M \subseteq M' \text{ και } \Gamma \vdash m':M' \text{ και } \Gamma;M' \vdash u:\tau$$

Το θεώρημα ασφάλειας αποτελεί ουσιαστικά το συνδυασμό των θεωρημάτων προόδου και διατήρησης, και λέει το εξής: αν η έκφραση e έχει τύπο, τότε η αποτίμησή της δε μπορεί να κολλήσει.

Σύστημα Τύπων

$$\frac{\Gamma;M \vdash e_1:\mathrm{Int} \quad \Gamma;M \vdash e_2:\mathrm{Int}}{\Gamma;M \vdash e_1+e_2:\mathrm{Int}}, \text{αντίστοιχα για } (-),(*),(/)$$

$$\frac{\Gamma;M \vdash e_1:\mathrm{Int} \quad \Gamma;M \vdash e_2:\mathrm{Int}}{\Gamma;M \vdash e_1<e_2:\mathrm{Bool}}, \text{αντίστοιχα για } (>),(=),(\leqslant),(\geqslant),(\neq)$$

$$\frac{\Gamma;M \vdash e_1:\mathrm{Bool} \quad \Gamma;M \vdash e_2:\mathrm{Bool}}{\Gamma;M \vdash e_1 \wedge e_2:\mathrm{Bool}}, \text{αντίστοιχα για } (\vee)$$

$$\frac{\Gamma;M \vdash e:\mathrm{Int}}{\Gamma;M \vdash -e:\mathrm{Int}} \qquad \frac{\Gamma;M \vdash e:\mathrm{Bool}}{\Gamma;M \vdash \neg e:\mathrm{Bool}} \qquad \frac{\Gamma;M \vdash e:\mathrm{Bool} \quad \Gamma;M \vdash e_1:\tau \quad \Gamma;M \vdash e_2:\tau}{\Gamma;M \vdash \mathit{if}\ e\ \mathit{then}\ e_1\ \mathit{else}\ e_2:\tau}$$

$$\frac{\Gamma;M,x:\tau \vdash e:\tau'}{\Gamma;M \vdash \lambda x:\tau.e:\tau \to \tau'} \qquad \frac{\Gamma;M \vdash e_1:\tau \to \tau' \quad \Gamma;M \vdash e_2:\tau}{\Gamma;M \vdash e_1\,e_2:\tau'}$$

$$\Gamma;M \vdash \mathit{unit}:\mathrm{Unit} \qquad \frac{\Gamma;M \vdash e_1:\mathrm{Unit} \quad \Gamma;M \vdash e_2:\tau}{\Gamma;M \vdash e_1;e_2:\tau}$$

$$\frac{M(i)=\tau}{\Gamma;M \vdash \mathit{loc}_i:\mathrm{Ref}\,\tau} \qquad \frac{\Gamma;M \vdash e:\tau}{\Gamma;M \vdash \mathrm{ref}\,e:\mathrm{Ref}\,\tau} \qquad \frac{\Gamma;M \vdash e:\mathrm{Ref}\,\tau}{\Gamma;M \vdash !e:\tau} \qquad \frac{\Gamma;M \vdash e_1:\mathrm{Ref}\,\tau \quad \Gamma;M \vdash e_2:\tau}{\Gamma;M \vdash e_1:=e_2:\mathrm{Unit}}$$

Συγκρίνοντας τα παραπάνω με τα semantics που περιέχονται στις διαφάνειες του μαθήματος, παρατηρούμε τα εξής:

- Small-Step Semantics:
  + Λόγω του ότι το evaluation παρουσιάζεται βήμα προς βήμα, είμαστε σε θέση να εκφράσουμε "έννοιες" μεγάλης πολυπλοκότητας σε κάθε βήμα, όπως για παράδειγμα διαχείριση σφαλμάτων ή εναλλακτικά μονοπάτια εκτέλεσης
  - … αλλά απ'την άλλη, το evaluation παρουσιάζεται βήμα προς βήμα, και σπάνια χρειάζεται να εκφράσουμε τέτοιες έννοιες όταν ορίζουμε τα semantics μιας γλώσσας, οπότε καταλήγουμε να καταβάλουμε κόπο πέραν του δέοντος στο όνομα της εκφραστικότητας.

- Big-Step semantics:
  + Μπορούμε να παρουσιάσουμε τα semantics με τέτοιο τρόπο, ώστε το νόημά τους να είναι εύκολα κατανοητό στον αναγνώστη. Αυτό είναι εμφανές για παράδειγμα στην περίπτωση της κλήσης συναρτήσεων, όπου ο κανόνας γραμμένος σε big-step semantics μεταφέρει άμεσα την πρόθεσή μας, σε αντιδιαστολή με τους δύο (!) κανόνες σε small-step semantics που περιγράφουν το ίδιο πράγμα.

+ Επειδή δεν έχουμε “λεπτό” granularity, και επειδή σε γενικές γραμμές δε μας χρειάζεται, γλυτώνουμε περιττό κόπο, αλλά
- Επειδή δεν έχουμε “λεπτό” granularity, δεν έχουμε την ίδια εκφραστικότητα όπως με τα small-step semantics.