Γλώσσες Προγραμματισμού ΙΙ
Συστήματα Τύπων

Μιλτιάδης Στούρας
ΑΜ: 03116022

Ακ. Έτος 2019-2020 - Ροή Λ

Εισαγωγή

Στην παρούσα εργασία ορίζουμε ένα σύστημα τύπων για μια γλώσσα που περιγράφει την λειτουργία μιας μηχανής στοίβας, με την γραμματική και την λειτουργική σημασιολογία που δίνεται στην εκφώνηση. Οι κανόνες συμπερασμού που θα ορίσουμε θα έχουν την ακόλουθη μορφή

$$\frac{p_2; \sigma_2 : \tau_2}{p_1; \sigma_1 : \tau_1}$$

και σημαίνουν πως αν το πρόγραμα p_1 εκκινεί την εκτέλεσή του με μια στοίβα σ_1 τύπου τ_1, τότε το πρόγραμα p_2 εκκινεί την εκτέλεσή του με μια στοίβα σ_2 τύπου τ_2.
Ακόμα, ορίζουμε τους τύπους Int και $Bool$ για τις αριθμητικές και λογικές σταθερές. Ο τύπος της στοίβας θα ορισθεί αναδρομικά με το παρακάτω fix point, παίρνοντας περιπτώσεις για την κορυφή της στοίβας.

$$Stack = \mu\alpha.\ Unit + Int \times Stack + Bool \times Stack$$

Κανόνες συμπερασμού

Για κάθε εντολή που υποστηρίζει η γλώσσα μας γράφουμε τους κατάλληλους κανόνες συμπερασμού τύπων.

Προσθήκη τιμών στην στοίβα

- $n; \sigma \to \sigma \cdot n$:

$$\frac{(n\ p); \sigma :\ \tau}{p; (\sigma \cdot n) :\ Int \times \tau} \tag{1}$$

- $\mathbf{true}; \sigma \to \sigma \cdot \mathbf{true}$

$$\frac{(\mathbf{true}\ p); \sigma :\ \tau}{p; (\sigma \cdot \mathbf{true}) :\ Bool \times \tau} \tag{2}$$

- $\mathbf{false}; \sigma \to \sigma \cdot \mathbf{false}$

$$\frac{(\mathbf{false}\ p); \sigma :\ \tau}{p; (\sigma \cdot \mathbf{false}) :\ Bool \times \tau} \tag{3}$$

Αριθμητικές πράξεις

- $+; (\sigma \cdot n_1 \cdot n_2) \to \sigma \cdot (n_1 + n_2)$:

$$\frac{(+\ p); (\sigma \cdot n_1 \cdot n_2) :\ Int \times Int \times \tau}{p; (\sigma \cdot (n_1 + n_2)) :\ Int \times \tau} \tag{4}$$

- $-; (\sigma \cdot n) \to \sigma \cdot (-n)$:

$$\frac{(-\ p); (\sigma \cdot n) :\ Int \times \tau}{p; (\sigma \cdot (-n)) :\ Int \times \tau} \tag{5}$$

- $*; (\sigma \cdot n_1 \cdot n_2) \to \sigma \cdot (n_1 * n_2)$:

$$\frac{(*\ p); (\sigma \cdot n_1 \cdot n_2) :\ Int \times Int \times \tau}{p; (\sigma \cdot (n_1 * n_2)) :\ Int \times \tau} \tag{6}$$

- $\dfrac{n_2 \neq 0 \quad (q, r) = \text{quotRem}(n_1, n_2)}{/; (\sigma \cdot n_1 \cdot n_2) \to \sigma \cdot q \cdot r}$

$$\frac{(/\ p); (\sigma \cdot n_1 \cdot n_2) : Int \times Int \times \tau \quad n_2 \neq 0 \quad (q, r) = \text{quotRem}(n_1, n_2)}{p; (\sigma \cdot q \cdot r) : Int \times Int \times \tau} \tag{7}$$

Λογικές πράξεις

- $<; (\sigma \cdot n_1 \cdot n_2) \to \sigma \cdot (n_1 < n_2)$:

$$\frac{(<\ \ p); (\sigma \cdot n_1 \cdot n_2) :\ Int \times Int \times \tau}{p; (\sigma \cdot (n_1 < n_2)) :\ Bool \times \tau} \tag{8}$$

- $=; (\sigma \cdot n_1 \cdot n_2) \to \sigma \cdot (n_1 < n_2)$:

$$\frac{(=\ \ p); (\sigma \cdot n_1 \cdot n_2) :\ Int \times Int \times \tau}{p; (\sigma \cdot (n_1 = n_2)) :\ Bool \times \tau} \tag{9}$$

- $\textbf{and}; (\sigma \cdot n_1 \cdot n_2) \to \sigma \cdot (n_1 \wedge n_2)$:

$$\frac{(\textbf{and}\ p); (\sigma \cdot n_1 \cdot n_2) :\ Int \times Int \times \tau}{p; (\sigma \cdot (n_1 \wedge n_2)) :\ Bool \times \tau} \tag{10}$$

- $\textbf{not}; (\sigma \cdot n) \to \sigma \cdot (\neg\ n)$:

$$\frac{(\textbf{not}\ p); (\sigma \cdot n) :\ Bool \times \tau}{p; (\sigma \cdot (\neg\ n)) :\ Bool \times \tau} \tag{11}$$

Εντολές τροποποίησης της στοίβας

- **dup**; $(\sigma \cdot v) \to (\sigma \cdot v \cdot v)$:

$$\frac{(\textbf{dup}\ p); (\sigma \cdot v) :\ \tau' \times \tau \quad \tau' \in \{Int, Bool\}}{p; (\sigma \cdot v \cdot v) :\ \tau' \times \tau' \times \tau} \tag{12}$$

- **pop**; $(\sigma \cdot v) \to \sigma$:

$$\frac{(\textbf{pop}\ p); (\sigma \cdot v) :\ \tau' \times \tau \quad \tau' \in \{Int, Bool\}}{p; \sigma :\ \tau} \tag{13}$$

- **swap**; $(\sigma \cdot v_1 \cdot v_2) \to (\sigma \cdot v_2 \cdot v_1)$:

$$\frac{(\textbf{swap}\ p); (\sigma \cdot v_1 \cdot v_2) :\ \tau_2 \times \tau_1 \times \tau \quad (\tau_1, \tau_2) \in \{Int, Bool\}^2}{p; (\sigma \cdot v_2 \cdot v_1) :\ \tau_1 \times \tau_2 \times \tau} \tag{14}$$

- **swap2**; $(\sigma \cdot v_1 \cdot v_2 \cdot v_2) \to (\sigma \cdot v_3 \cdot v_1 \cdot v_2)$:

$$\frac{(\textbf{swap2}\ p); (\sigma \cdot v_1 \cdot v_2 \cdot v_3) :\ \tau_3 \times \tau_2 \times \tau_1 \times \tau \quad (\tau_1, \tau_2, \tau_3) \in \{Int, Bool\}^3}{p; (\sigma \cdot v_3 \cdot v_1 \cdot v_2) :\ \tau_2 \times \tau_1 \times \tau_3 \times \tau} \tag{15}$$

Εντολές stall και ελέγχου ροής

Για να διευκολύνουμε τον ορισμό του συστήματος τύπων μας, θα κάνουμε δύο απλουστεύσεις. Αρχικά, θα θεωρήσουμε ότι σε μια εντολή **cond**, και οι δύο κλάδοι της συνθήκης μπορούν να εκκινήσουν από τον ίδιο τύπο στοίβας και όταν εκτελεσθούν πλήρως καταλήγουν στον ίδιο τύπο στοίβας. Ακόμα, για την εντολή **loop** θα θεωρήσουμε ότι το σώμα της loop αφήνει αναλλοίωτο τον τύπο της στοίβας, δηλαδή φροντίζει να επανατοποθετεί μια Bool μεταβλητή στην κορυφή και αφήνει τον τύπο της υπόλοιπης στοίβας ίδιο. Με βάση αυτές τις απλουστεύσεις, παίρνουμε τους παρακάτω κανόνες συμπερασμού.

- **nop**; $\sigma \to \sigma$:

$$\frac{(\textbf{nop}\ p); \sigma :\ \tau}{p; \sigma :\ \tau} \tag{16}$$

- **cond**$[p_1|p_2]; \sigma \cdot \text{true} \to p_1; \sigma$, **cond**$[p_1|p_2]; \sigma \cdot \text{false} \to p_2; \sigma$:

$$\frac{\textbf{cond}[p_1|p_2]\ p; \sigma \cdot v :\ Bool \times \tau \quad p_1; \sigma :\ \tau \longrightarrow^* \sigma' :\ \tau' \quad p_2; \sigma :\ \tau \longrightarrow^* \sigma' :\ \tau'}{p; \sigma' : \tau'} \tag{17}$$

- **loop**$[p]; \sigma \cdot \text{true} \to p$ **loop**$[p]; \sigma$, **loop**$[p]; \sigma \cdot \text{false} \to \sigma$

$$\frac{\textbf{loop}[p]\ p'; \sigma \cdot v :\ Bool \times \tau \quad p; \sigma :\ \tau \longrightarrow^* \sigma' : Bool \times \tau'}{p' : \sigma' :\ \tau'} \tag{18}$$

Type Checking

Ως εφαρμογή του συστήματος τύπων που ορίσαμε, θα προσπαθήσουμε να ελέγξουμε το πρόγραμμα της εκφώνησης:

$p \equiv 1\ 3\ \textbf{true}\ \textbf{loop}[\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]]\ \textbf{pop}\ \textbf{dup}\ 1\ +\ *$

Ξεκινάμε με άδεια στοίβα ($\emptyset : Unit$) και εφαρμόζουμε τον κανόνα συμπερασμού που αντιστοιχεί στην κάθε φορά επόμενη εντολή. Ξεκινάμε με τον κανόνα 1 για την προσθήκη του 1 στην στοίβα:

$$\frac{1\ 3\ \textbf{true}\ \textbf{loop}[\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]]\ \textbf{pop}\ \textbf{dup}\ 1\ +\ *; \emptyset : Unit}{3\ \textbf{true}\ \textbf{loop}[\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]]\ \textbf{pop}\ \textbf{dup}\ 1\ +\ *; \sigma' : Int}$$

Εφαρμόζοντας τους κανόνες 1 και 2 διαδοχικά παίρνουμε:

$$\frac{3\ \textbf{true}\ \textbf{loop}[\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]]\ \textbf{pop}\ \textbf{dup}\ 1\ +\ *; \sigma : Int}{\textbf{true}\ \textbf{loop}[\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]]\ \textbf{pop}\ \textbf{dup}\ 1\ +\ *; \sigma' : Int \times Int}$$

$$\frac{\textbf{true}\ \textbf{loop}[\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]]\ \textbf{pop}\ \textbf{dup}\ 1\ +\ *; \sigma : Int \times Int}{\textbf{loop}[\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]]\ \textbf{pop}\ \textbf{dup}\ 1\ +\ *; \sigma' : Bool \times Int \times Int}$$

Επομένως, ξέρουμε ότι το **loop** θα ξεκινήσει με στοίβα $Bool \times Int \times Int$. Για να εφαρμόσουμε τον κανόνα 18, πρέπει πρώτα να επιβεβαιώσουμε ότι το σώμα της **loop**, ξεκινώντας από στοίβα $Int \times Int$, τελειώνει αφήνοντας **Bool** στην κορυφή της στοίβας. Εφαρμόζουμε τους κανόνες 12,1 και 8 διαδοχικά για το σώμα της **loop**:

$$\frac{\textbf{dup}\ 2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]; \sigma : Int \times Int}{2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]; \sigma' : Int \times Int \times Int}$$

$$\frac{2\ <\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]; \sigma : Int \times Int \times Int}{<\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]; \sigma' : Int \times Int \times Int \times Int}$$

$$\frac{<\ \textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]; \sigma : Int \times Int \times Int \times Int}{\textbf{cond}[\textbf{false} \mid \textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}]; \sigma' : Bool \times Int \times Int}$$

Για να χρησιμοποιήσουμε τον κανόνα 17 πρέπει να υπολογίσουμε ότι και τα δύο branches του **cond** ξεκινώντας από στοίβα $Int \times Int$, μόλις τερματίσουν αφήνουν στοίβα ίδιου τύπου. Για το πρώτο branch, εφαρμόζουμε τον κανόνα 3:

$$\frac{\textbf{false}; \sigma : Int \times Int}{\sigma' : Bool \times Int \times Int}$$

Για το 2ο branch, εφαρμόζουμε διαδοχικά τους κανόνες 12, 1, 5, 4, 15, 6, 14, 2:

$$\frac{\textbf{dup}\ 1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}; \sigma : Int \times Int}{1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}; \sigma' : Int \times Int \times Int}$$

$$\frac{1\ -\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}; \sigma : Int \times Int \times Int}{-\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}; \sigma' : Int \times Int \times Int \times Int}$$

$$\frac{-\ +\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}; \sigma : Int \times Int \times Int \times Int}{+\ \textbf{swap2}\ *\ \textbf{swap}\ \textbf{true}; \sigma' : Int \times Int \times Int \times Int}$$

$$\frac{+\ \textbf{swap2}\ *\ \textbf{swap true}; \sigma : Int \times Int \times Int \times Int}{\textbf{swap2}\ *\ \textbf{swap true}; \sigma' : Int \times Int \times Int}$$

$$\frac{\textbf{swap2}\ *\ \textbf{swap true}; \sigma : Int \times Int \times Int}{*\ \textbf{swap true}; \sigma' : Int \times Int \times Int}$$

$$\frac{*\ \textbf{swap true}; \sigma : Int \times Int \times Int}{\textbf{swap true}; \sigma' : Int \times Int}$$

$$\frac{\textbf{swap true}; \sigma : Int \times Int \times Int}{\textbf{true}; \sigma' : Int \times Int}$$

$$\frac{\textbf{true}; \sigma : Int \times Int \times Int}{\sigma' : Bool \times Int \times Int}$$

Επομένως και τα δύο branch καταλήγουν στον ίδιο τύπο, άρα μπορούμε να εφαρμόσουμε τον 17 και να καταλήξουμε ότι το σώμα της **loop** κάνει type check και τερματίζει με στοίβα τύπου $Bool \times Int \times Bool$. Προχωρόντας μετά την loop, εφαρμόζουμε διαδοχικά τους κανόνες 13, 12, 1, 4, 6 παίρνουμε:

$$\frac{\textbf{pop dup}\ 1\ +\ *; \sigma : Int \times Int}{\textbf{dup}\ 1\ +\ * : \sigma' : Int}$$

$$\frac{\textbf{dup}\ 1\ +\ * : \sigma : Int}{1\ +\ * : \sigma' : Int \times Int}$$

$$\frac{1\ +\ * : \sigma : Int \times Int}{+\ * : \sigma' : Int \times Int \times Int}$$

$$\frac{+\ * : \sigma : Int \times Int \times Int}{* : \sigma' : Int \times Int}$$

$$\frac{* : \sigma : Int \times Int}{\sigma' : Int}$$

δηλαδή όλο το πρόγραμμα κάνει type check και τερματίζει με στοίβα τύπου Int όπως περιμέναμε, αφού όταν τερματίσει στην στοίβα υπάρχει ο αριθμός 42.