Σταματελόπουλος Νικόλαος
Α.Μ.: 03116138

Γλώσσες Προγραμματισμού ΙΙ

Άσκηση 6: Συστήματα Τύπων - letrec

1) Αρχικά προσθέτουμε στη σύνταξη της γλώσσας τον κανόνα

$$e ::= \ldots \mid \text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e$$

2) Κανόνας τύπων

$$\frac{\text{for } j = 1, \ldots, n : \ \Gamma, \{x_i : \tau_i\}^{i=1,\ldots,n} \vdash e_j : \tau_j \qquad \Gamma, \{x_i : \tau_i\}^{i=1,\ldots,n} \vdash e : \tau}{\Gamma \vdash \text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e : \tau}$$

3) Λειτουργική Σημασιολογία

• call by value

$$\text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } \lambda x_1 : \tau_1 . (\ldots (\lambda x_n : \tau_n . \tau) \ldots) \longrightarrow$$

$$\longrightarrow e_{[x_1 := \text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e_1]} \ldots [x_n := \text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e_n]$$

$$\frac{e \longrightarrow e'}{\text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e \longrightarrow \text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e'}$$

• **call by name**

$$\text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e \longrightarrow$$

$$e\,[x_1 := \text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e_1]$$
$$\ldots$$
$$[x_n := \text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e_n]$$

4) Παραγόμενη Μορφή

Ο ορισμός του fix που έχουμε στη διάθεσή μας είναι αρκετά γενικός οπότε και θα μπορούσαμε να τον χρησιμοποιήσουμε για ορισμό του letrec ως syntactic sugar, αν αυτό μας επιστρέψει μια πλειάδα εκφράσεων $<e_1, \ldots, e_n>$.

$$\text{letrec } x_1 = e_1 \ldots x_n = e_n \text{ in } e \equiv$$

$$\text{let } <x_1, \ldots, x_n> = \text{fix}\,(\lambda <x_1{:}\tau_1, \ldots, x_n{:}\tau_n>.\,<e_1, \ldots, e_n>) \text{ in } e$$

5) Κανόνας Τύπων

Αρχίζοντας από την ίδια υπόθεση

$$\text{for } j=1,\dots,n: \Gamma, \{x_i:\tau_i\}^{i=1,\dots,n} \vdash e_j:\tau_j$$
$$\Gamma, \{x_i:\tau_i\}^{i=1,\dots,n} \vdash e:\tau$$

$$\dfrac{\dfrac{\dfrac{\text{for } j=1,\dots,n: \Gamma, \{x_i:\tau_i\}^{i=1,\dots,n} \vdash e_j:\tau_j}{\Gamma, \{x_i:\tau_i\}^{i=1,\dots,n} \vdash \langle e_1,\dots,e_n\rangle : \tau_1\times\dots\times\tau_n}\ (\text{Tuple Rule})}{\Gamma \vdash \lambda\langle x_1:\tau_1,\dots,x_n:\tau_n\rangle.\langle e_1,\dots,e_n\rangle : \tau_1\times\dots\times\tau_n \to \tau_1\times\dots\times\tau_n}\ (\text{Function Rule})}{\Gamma \vdash fix(\lambda\langle x_1:\tau_1,\dots,x_n:\tau_n\rangle.\langle e_1,\dots,e_n\rangle) : \tau_1\times\dots\times\tau_n}\ (\text{Fix Rule})$$

Από υπόθεση: $\dfrac{\Gamma, \{x_i:\tau_i\}^{i=1,\dots,n} \vdash e:\tau}{\Gamma \vdash let\ \langle x_1,\dots,x_n\rangle = fix(\lambda\langle x_1:\tau_1,\dots,x_n:\tau_n\rangle.\langle e_1,\dots,e_n\rangle)\ in\ e : \tau}\ (\text{Let Rule})$

Που έχει τελικά τον ίδιο τύπο με τον κανόνα που ορίσαμε για το letrec στο ερώτημα 2.

(Σε ορισμένα σημεία παραλείφθηκε η εφαρμογή του κανόνα πλησιάσας για τα $x_i:\tau_i$ χαριν ευκολίας και συντομίας)

Λειτουργική Σημασιολογία

Αρχικά λόγω της λειτουργίας του let θα έχουμε αντικατάσταση του $<x_1,\dots,x_n>$ tuple εντός της e έκφρασης με το $fix(\lambda<x_1:\tau_1,\dots,x_n:\tau_n>.<e_1,\dots,e_n>)$ υποθέτοντας ότι το e είναι της μορφής $\lambda x_1:\tau_1(\dots(\lambda x_n:\tau_n.e)\dots)$ (αλλιώς πρώτα το φέρνεις σε αυτή τη μορφή) και πως κάθε εμφάνιση x_i στο e αντιστοιχεί στο i-οστό στοιχείο της πλειάδος.
Στη συνέχεια το fix κατά τον αναδρομικό ορισμό του ακολουθεί την ίδια ακριβώς λογική με τη letrec, περνώντας κάθε φορά την πλειάδα με τον ορισμό των συναρτήσεων στο επόμενο βήμα για κάθε $\lambda x_1:\tau_1(\dots(\lambda x_n:\tau_n.e_i)\dots)$ εμφάνιση.
Τα παραπάνω δίνοντας τη λειτουργική σημασιολογία των let και fix που ορίσαμε, συμφωνούν ακριβώς με αυτή του letrec τελικά, γιατί σε κάθε αναδρομικό βήμα αποτίμηση η εμφάνιση της e_i βρίσκεται εντός του scope το ορισμού της που είναι εντός της πλειάδας που μας περνάει το fix σε αντίθεση με τη letrec που τα περνά από τη μία εμφάνιση στην άλλη όλα τα $\lambda x_1(\dots(\lambda x_n.e_n))$ ξεχωριστά.