Συναρτησιακός Προγραμματισμός

Εισαγωγικές Έννοιες

Κίνητρο

- Οι συναρτησιακές γλώσσες προγραμματισμού προσφέρονται για «μαζικό» προγραμματισμό:
  - Έχουν σαφείς μαθηματικές ιδιότητες που μπορούν να αξιοποιηθούν για να γίνεται τυπική επαλήθευση προγράμματος.
  - Προσφέρονται για παράλληλο υπολογισμό.
  - Προσφέρονται για επαναχρησιμοποίηση κώδικα ή ενσωμάτωση κώδικα.
  - Απλό μοντέλο επικοινωνίας κατά τη διαδικασία του υπολογισμού.
  - Δεν προϋποθέτουν συγκεκριμένη πλατφόρμα επεξεργασίας.
- Εν τέλει, προσφέρονται για την ανάπτυξη Αλγεβρας Προγραμμάτων.

Μερική από την ιστορία …

- Η αφετηρία Church (1930): lampda calculus.
- LISP, ISWIM, Scheme (LISP dialect), ML, Hope, SASL, ML (metalanguage), Hope, **SML** (ML+Hope), LML, Miranda, Haskell, ……
- SML γιατί είναι αμιγώς συναρτησιακή, και πρότυπο.

Συναρτησιακό Πρόγραμμα

- Ένα σύνολο εξισώσεων που περιγράφουν ιδιότητες αντικειμένων ή συσχετίζουν ποσότητες που μας ενδιαφέρουν.
- Δύο αναγνώσεις:
  - Ως κανόνες υπολογισμού.
  - Ως δηλώσεις (ορισμοί) ιδιοτήτων αντικειμένων
- Δύο χρήσεις:
  - Παραγωγή αποτελέσματος
  - Εκτελέσιμος έλεγχος ορθότητας προσδιορισμού

Συναρτήσεις – στα μαθηματικά

- Μαθηματικά, η αντιστοίχιση τιμών που προέρχονται από συγκεκριμένα σύνολα (πεδία ορισμού), σε μια **μοναδική** τιμή ενός συγκεκριμένου συνόλου (πεδίο τιμών).
- double(n)= 2 x n
  - double: N → N.
- even(n) = T if n divisible by 2
  - even: N → {T, F}

- Η εικόνα του ορίσματος μιας συνάρτησης είναι μοναδική.
- Πολλές διαφορετικές τιμές από το πεδίο ορισμού της συνάρτησης μπορεί να αντιστοιχίζονται στην ίδια τιμή στο πεδίο τιμών της. π.χ. Even
- Μια συνάρτηση είναι πλήρης αν αντιστοιχίζει κάθε τιμή του πεδίου ορισμού σε μια τιμή του πεδίου τιμών, αλλιώς είναι μερική.

Άλλες συναρτήσεις

- computer : keyboard strokes → Screen sequences
- compiler : source programs → object code streams
- problem_solver : problems → solutions
- theorem_prover : logic sentences → proofs
- salary: employees → amounts of money
- capital : countries → cities
- mother : people → people
- religious_views: people → religions
- vote : voters → political parties
- lecture_day: courses → weekdays

Ορισμός συνάρτησης

- **Intensionally** (δηλωτικά, έμμεσα, εννοιολογικά)
  - Με κανόνα που περιγράφει το συσχετισμό του πεδίου τιμών με το πεδίο ορισμού
  - π.χ. double (n) = 2 x n
  - Περιγράφει τη διαδικασία με την οποία υπολογίζεται η εικόνα του ορίσματος.
- **Extensionally** (περιγραφικά, αναλυτικά, ρητά, συμπεριφορικά, γράφημα συνάρτησης)
  - Ως σύνολο διατεταγμένων ζευγών της μορφής
    (όρισμα, αποτέλεσμα)
  - double = {(0,0), (1, 2), (2, 4), (3, 6), (4, 8), ….}
  - Δεν ασχολείται με την διαδικασία υπολογισμού, αλλά με τη συμπεριφορά της συνάρτησης όπως αυτή μπορεί να διαπιστωθεί εξωτερικά (τι συσχετίζει με τι).

- Στο FP εκμεταλλευόμαστε αυτή τη διπλή ανάγνωση των συναρτήσεων. Για τη χρήση μιας συνάρτησης ενδιαφέρει μόνο η συμπεριφορά της, όχι ο τρόπος υπολογισμού της
  - Σύνθεση μεγάλων συστημάτων από συστατικά
  - Επαναχρησιμοποίηση λογισμικού
  - Αντικατάσταση λογισμικού
  - Διαλειτουργικότητα λογισμικού
  - Άλγεβρα προγραμμάτων!!!

Ισότητα συναρτήσεων

Εστω f : A → B και g: C → D. Τότε f = g

αν και μόνο αν

A = C και B = D και **έχουν το ίδιο γράφημα**.

(ας έχουν διαφορετικό δηλωτικό ορισμό /τρόπο υπολογισμού).

Π.χ. double (n) = 2 x n, double : N → N

dpl (n) = n + n, dpl : N → N

Αυτό έχει εξαιρετικά ενδιαφέρουσες συνέπειες (άλγεβρα προγραμμάτων)

Σημειογραφία

- Για μια συνάρτηση f : A → B
  - f (a) = b
  - f a = b
  - διαγραμματικά

Συναρτήσεις πολλαπλών ορισμάτων

- Μπορούν να ειδωθούν σαν συναρτήσεις ενός ορίσματος, το οποίο είναι δομημένο αντικείμενο.
- π.χ. Η πρόσθεση φυσικών αριθμών + : N x N → N
  - πεδίο ορισμού το σύνολο N x N που περιέχει δομημένα στοιχεία: διατεταγμένα ζεύγη
- Prefix/infix μορφή συνάρτησης
  - 2 + 3 = + (2, 3)
- Γενικότερα μια συνάρτηση f : A1 x A2 x…xAn → B δέχεται ως όρισμα τη διατεταγμένη πλειάδα (α1, α2, …, αn)
- Διαγραμματική απεικόνιση συναρτήσεων πολλαπλών ορισμάτων, π.χ. +, x, or, and, string concat

Εκφράσεις και σύνθεση

- Μια έκφραση μπορεί να συνδυάζει πολλές συναρτήσεις, καθιστώντας το αποτέλεσμα της εφαρμογής μιας συνάρτησης, όρισμα για την εφαρμογή μιας άλλης συνάρτησης.
  - π.χ. έκφραση even (45 x 30) συνθέτει τις συναρτήσεις even και x.
- Ο συνδυασμός συναρτήσεων μπορεί να ειδωθεί σαν καινούρια συνάρτηση.
  - π.χ. evenprod (a, b) = even (a x b)
- Οποιοσδήποτε συνδυασμός συναρτήσεων περιγράφεται από μια έκφραση είναι σίγουρα συνάρτηση.
  - διαγραμματική απεικόνιση σύνθεσης συναρτήσεων: έξοδος της μιας αποτελεί είσοδο στην άλλη

Σύνθεση και επικοινωνία κατά τον υπολογισμό: η ουσία του FP

- Όλα τα αποτελέσματα που παράγονται σε επιμέρους υπολογισμούς, διαδίδονται σε άλλα μέρη του προγράμματος μόνο μέσω της εφαρμογής συναρτήσεων.
- Αντίθετα, οι διαδικαστικές γλώσσες προγραμματισμού χρησιμοποιούν καθολικές μεταβλητές μνήμης, το περιεχόμενο των οποίων ενημερώνεται και είναι προσβάσιμο από όλα τα μέρη του προγράμματος που χρειάζεται να ενημερωθούν.
- **Αναφορική διαφάνεια (referential transparency)** – για τον καθορισμό του νοήματος μιας σύνθετης έκφρασης απαιτείται μόνο το νόημα των άμεσων συνιστωσών της. Δυο εκφράσεις είναι ισοδύναμες όταν έχουν ίδιο νόημα (τιμή), επομένως λόγω της αναφορικής διαφάνειας, μπορεί να αντικατασταθεί μια συνιστώσα σε μια έκφραση από μια ισοδύναμή της, χωρίς να επηρεαστεί το νόημα της έκφρασης.
- Μπορεί να αποδοθεί αναφορική διαφάνεια σε συναρτήσεις που γράφονται σε διαδικαστικές γλώσσες προγραμματισμού, αλλά με τίμημα την απόδοση συχνά αντιδιαισθητικού νοήματος στα προγράμματα. π.χ η συνάρτηση double μπορεί να έχει εκτός από υπολογισμό τιμής μιας καθολικής μεταβλητής και παρενέργεια την εκτύπωση αποτελέσματος.

Τότε double (3) + double (3) δεν είναι ίδιο με 2 * double (3)

Τύποι Δεδομένων : γιατί τους χρειαζόμαστε;

- Συνήθως ταξινομούμε τις διάφορες τιμές δεδομένων που μεταχειρίζεται ένα πρόγραμμα σε **τύπους**, ανάλογα με το πώς επιτρέπεται να χρησιμοποιούνται, και μετά ελέγχουμε κατά πόσο γίνεται ορθή χρήση των τύπων, προκειμένου να προλάβουμε και να διορθώσουμε λάθη στη χρήση δεδομένων, πριν την πραγματική εκτέλεση του προγράμματος. (automatic type checking)

- Οι περισσότερες γλώσσες προγραμματισμού **εμπεριέχουν** τύπους, και συναρτήσεις που ορίζουν πώς χρησιμοποιούνται σωστά αυτοί οι τύποι.
  - int, bool, real etc.
  - +, -, *, div ορίζονται με πεδίο ορισμού int x int, και όχι bool.
  - even ορίζεται με πεδίο ορισμού int και πεδίο τιμών bool.

- Οι περισσότερες γλώσσες FP συμμερίζονται το πολυμορφικό σύστημα τύπων του Milner: σύνθετοι τύποι καινούριων δεδομένων μπορούν να ορισθούν με βάση τους τύπους των συνιστωσών τους, και υπάρχει μόνο περιορισμένος αριθμός πρωταρχικών τύπων, από τους οποίους ορίζονται όλοι οι άλλοι, με την εφαρμογή τελεστών.

Ένα σύστημα τύπων

- Πρωταρχικοί τύποι : int, bool, string
- Σταθερές : true, false είναι οι μόνες τιμές τύπου bool.
- Οι τιμές τύπου int είναι οι συνήθεις σταθερές ακεραίων,αλλά γράφουμε ~5 αντί για -5 για να μην συγχέεται το αρνητικό πρόσημο με τη συνάρτηση της αφαίρεσης.
- Οι τιμές για τον τύπο string είναι αλφαβητικές ακολουθίες μέσα σε “ ”.

Τελεστές (παραγωγής) τύπων : → και x

- Όπου T1 και T2 είναι τύποι, τότε (T1 → T2) είναι ο τύπος μιας συνάρτησης με πεδίο ορισμού T1 και πεδίο τιμών T2.
  - η double έχει τύπο int → int
  - η even έχει τύπο int → bool
- Μια πλειάδα έχει τύπο το καρτεσιανό γινόμενο των τύπων που συμμετέχουν σ'αυτήν. Δηλαδή, αν E1 είναι έκφραση με τύπο T1, E2 έκφραση με τύπο T2, …, και En έκφραση με τύπο Tn, τότε (E1, …, En) είναι έκφραση τύπου T1 x T2 x…xTn
  - (3, true) έχει τύπο int x bool
  - ((2, "ab"), false, 6) έχει τύπο (int x string) x bool x int

Συνδυασμός τελεστών τύπων

- Οι τελεστές παραγωγής τύπων → και x μπορούν να συνδυαστούν σε πιο σύνθετες εκφράσεις, προκειμένου να παραχθούν πιο σύνθετοι τύποι.
- Π.χ. ο τύπος της συνάρτησης που συνενώνει συμβολοσειρές (string concatenation) είναι

  string x string → string
- Παρατηρήστε ότι οι τελεστές τύπων μπορούν να ειδωθούν ως συναρτήσεις με πεδίο ορισμού και τιμών το σύνολο των τύπων!

Κανόνας ορθής παραγωγής τύπων

- Αν Ε είναι μια έκφραση που περιγράφει μια **συνάρτηση** με τύπο Τ1 → Τ2 και
- Ε' είναι μια έκφραση που περιγράφει μια **τιμή** τύπου Τ1,
- τότε η εφαρμογή της Ε στην Ε' (δηλαδή Ε(Ε') ή Ε Ε') είναι ορθού τύπου και θα έχει τύπο Τ2.

- Γράφουμε Ε:Τ για να δηλώσουμε ότι η Ε είναι ορθού τύπου Τ.

Παραγωγή ορθών τύπων (διαγραμματικά)

- Για εφαρμογή συνάρτησης

$$\frac{E : T1 \rightarrow T2, E' : T1}{E(E') : T2}$$

- Για σχηματισμό πλειάδων

$$\frac{E1 : T1, E2 : T2, \ldots, En : Tn}{(E1, E2, \ldots, En) : T1 x T2 x \ldots x Tn}$$