Esercizi ed Altro

Sommario: 10 aprile, 2021

- **Esercizi**
  Funzioni Parz. Valutate e Valutazione (Esterna vs. Stretta).
- **Esercizi**
  Tipi Astratti: Store in AM21.
- **Altro**
  Small21: Un Nucleo per La Macchina a Stati di Minsky-Wang (prepariamoci all'attività di Laboratorio di domani)

Esercizi del 10/05

Esercizio (1: Funzioni Parzialmente Valutate, Valori, Valutazione Esterna vs. Trasmissione per Valore)

```
Si consideri il comportamento delle seguenti espressioni (chiuse = combinatori) in Ocaml:
                let le x y = (x < y) || (x = y);;
                let le5 = le 5;;                    (* nominare f. parzialmente valutata *)
                let rec bot n = bot (succ n);;      (* Chi e' succ? *)
                bot 0;;                             (* Ricordarsi di ctrl-c *)
                let bot0 = bot 0;;                  (* nominare espressione: Perche' questo com-
                                                       portamento? - *)
                let g x y z = if le5 x then y else z;;
                g 0 10 (bot 0);;                    (* Perche'? - Lo sapevamo: OCaml usa ... *)
                                                    (* Il Lambda-Calcolo usa ..... *)
(* Con trasmissione per nome (beta-riduzione) (g 0 10) risulterebbe definita anche su (bot 0) *)
(* Vantaggi: ... (Proprieta' Formali? = (bot 0) non e' un valore del dominio pero' e' un'espres-
                                        sione chiusa = combinatore ed e' un valore del calolo) *)
(* Costo: ... (valore vs. nome: Calcolare tutto vs. Closure per tutto ma
                              i combinatori sono gia' Closure!!) *)
                let k = fun x y -> x;;              (* Come descrivere k ?  *)
                let why = k 10 bot0                 (* Comportamento di k ben diverso da quello
                                                       che ha nel Lambda-Calcolo *)
```

Esercizi del 10/05

Esercizio (2: Progettazione e Sviluppo di ADT)

Si fornisca in OCaml, un tipo astratto per i valori store da utilizzare nella AM21 di Small21. Allo scopo si proceda guardando la sola definizione della AM21 fornita in Small21: Definizione. In particolare, anche se disponibile, Si ignori del tutto l'implementazione data della macchina AM21 o dell'esecutore di Small21.

Esercizio (3: Molularizzazione di Codice e Definizione di ADT)

Si consideri l'implementazione di AM21 nei Listing OCaml prodotti durante l'implementazione di Small21. Si individui la sezione (o le sezioni, in caso di cattiva localizzazione delle definizioni) di codice relativa alla definizione dello store. Si enuclei il codice di tale sezione, rimuovendolo opportunamente, e lo si incapsuli in un modulo Ocaml. Si utilizzi infine, tale modulo per definire un tipo astratto da introdurre nel contesto (anche REPL) di uso ed esecuzione del nuovo listing ottenuto per l'implementazione di Small21.

Esercizio (4: Rifrasare Programmi della Macchina a Stati in Programmi di un Nucleo di Small21)

Si discuta come sia possibile rifrasare ogni Programma della Macchina a Stati di Minsky-Wang in un Programma di un Nucleo di Small21. Il rifrasamento deve consistere di una traduzione (non contestuale) di ogni statement del programma sorgente in una sequenza di statements del programma oggetto.
Allo scopo, si risponda puntualmente alle seguenti richieste:.
(a) Come esprimere in Small21 e come introdurre nel Programma Oggetto i registri utilizzati nel Sorgente?
(b) Come esprimere in Small21 e come introdurre nell' Oggetto le posizioni degli statements del Sorgente?
(c) Cosa utilizzare per esprimere il costrutto: `Zero R`
(d) Cosa utilizzare per esprimere il costrutto: `Inc R`
(e) Cosa utilizzare per esprimere il costrutto: `DJ0 R N`
(f) Cosa utilizzare per esprimere il costrutto: `Halt`
(g) Si mostri (a)-(f) nella traduzione del seguente sorgente:

```
Zero R0
DJ0 R1 4
Inc R2
DJ0 R0 1
Halt
```

Soluzione
punto (a): `Rk`
Usiamo variabili intere, stesso nome Rk, introdutte con dicharazioni ad inizio blocco programma oggetto.
punto (b): `N`
Usiamo label pN introdotte etichettando primo comando sequenza generata per sorgente in tale posizione.
punto (c): `Zero R`
Usiamo assegnamento con 0, etichettato con la posizione del sorgente.
punto (d): `Inc R`
Usiamo assegnamento con variabile R incrementata di 1, etichettato con la posizione del sorgente.

Small21: Un Nucleo per La Macchina a Stati

Esercizio (4: Rifrasare Programmi della Macchina a Stati in Programmi di un Nucleo di Small21)

. . .

(e) Cosa utilizzare per esprimere il costrutto: `DJ0 R N`
(f) Cosa utilizzare per esprimere il costrutto: `Halt`
(g) Si mostri (a)-(f) nella traduzione del seguente sorgente:

```
Zero R0
DJ0 R1 4
Inc R2
DJ0 R0 1
Halt
```

Soluzione(continua)
punto (e): `DJ0 R N`
Usiamo condizionale if-then, etichettato ..., con (R = 0) come predicato e goto pN come comando. Aggiungiamo in sequenza, il comando non etichettato, R = R - 1
punto (f): `Halt`
Usiamo comando goto end, dove etichetta end sia etichetta introdotta dal comando end: ;, aggiunta come ultimo comando dell'oggetto prodotto.

```
punto (g)
    int R0; int R1; int R2;
p0: R0 = 0;
p1: if (R1 == 0) goto p4;
    R1 = R1 - 1;
p2: R2 = R2 + 1;
p3: if (R0 == 0) then goto p1;
    R0 = R0 - 1;
p4: goto end;
end: ;
```

Small21: Un Nucleo per La Macchina a Stati

Esercizio (5: Nucleo Small21 per La Macchina a Stati - Regole di Inferenza in AM21)

Si fornisca il Nucleo Small21 per rifrasare i Programmi della Macchina a Stati. In particolare:
(a) Si elenchino i costrutti da utilizzare;
(b) Si forniscano le regole di inferenza[a] *per i costrutti non ancora definiti in Laboratorio.*

---

[a] `Nel fornire tali regole si proceda nel modo usuale: Prima le regole del costrutto per il sistema dei tipi, quindi le regole per la semantica dinamica che nel nostro caso deve includere il controllo dinamico dei tipi. Prima di procedere si controlli sempre, la funzione semantica relativa al costrutto da considerare.`

Soluzione
punto (a): costrutti Small21 richiesti

- dichiarazioni; (giá definire e implementate)
- espressioni:
  - somma, +, sottrazione, -, ed
  - uguaglianza, ==.
- comandi:
  - Statement etichettato;
  - sequenza di comandi;
  - statement goto;
  - Statement assegnamento;
  - Statement condizionale;

punto (b): Regole di Inferenza - Inserirle in 'Small21 - Definizione'