Prova Scritta di Linguaggi di Programmazione II

27/03/2006

1. Si rappresenti il codice della macchina di Warren relativo al seguente programma in Prolog.

```
sel( [X,Y], [X,Y] ).
sel( [X,Y,Z|S], L ) :- nxt( [X,Y,Z|S], T ), sel( T, L ).
```

2. Si stabilisca (argomentando opportunamente) se il seguente programma è lineare destro/sinistro, duplicante, eliminante, collassante e se la sua relazione di riscrittura è confluente.

$$
\begin{aligned}
f(c(x), k(y), z) &\to f(x, y, d(d(z))) \\
f(c', k(y), z) &\to f(c', y, d(z)) \\
f(c(x), k', z) &\to f(x, k', d(z)) \\
f(c', k', y) &\to y
\end{aligned}
$$

3. Si calcoli il minimo punto fisso dell'operatore delle conseguenze immediate per le risposte calcolate (T_P^{ca}) del seguente programma:

```
p(X,h(Y)) :- q(X,h(Y),f(Y)).
p(X,Y) :- q(h(X,Y),Y,X).
p(X,f(X)) :- p(f(X),X).

q(h(X,g(X)),Y,0).
q(g(X),X,Y).
```

$$\boxed{\{q(h(X, g(X)), Y, 0), q(g(X), X, Y), p(0, g(0)), p(g(h(X)), h(X))\}}$$

4. Sia dato il seguente TRS R

$$
\begin{aligned}
m(s(0), Y) &\to s(0) && (1)\\
m(s(s(X)), Y) &\to Y + m(s(X), Y) && (2)\\
l(0) &\to [0] && (3)\\
l(s(X)) &\to s(X) : l(X) && (4)
\end{aligned}
$$

Si verifichi che R è induttivamente sequenziale e si calcolino tutte le derivazioni di needed narrowing di lunghezza 1 per il termine $l(m(Z, s(Z))$.

5. Si scriva in Curry una funzione (non-deterministica) *solveknight*(n) che restituisca una (ogni) lista di posizioni che rappresentino una soluzione per il gioco del Giro del Cavallo a dimensione n. Tale gioco consiste nel partire con un cavallo da un angolo di una scacchiera $n \times n$ e riuscire a muoverlo in modo che passi su *ogni* casella *esattamente* una volta.

   Si dia esplicitamente il tipo della funzione.

6. Si scriva un predicato `treepath/2`, in PROLOG, che determina percorsi all'interno di un albero binario. Quindi, a titolo di esempio,

```
treepath(tree(1,tree(2,tree(3,void,void),void),tree(4,void,void)),Xs)
```

   restituisce 2 soluzioni per `Xs`: `[1,2,3]` e `[1,4]`. Si faccia attenzione a *non generare la stessa soluzione più volte.*

   Si scriva il programma con variabili anonime ove possibile.

7. Si consideri il seguente frammento di codice sintatticamente ammissibile sia in Haskell che in Curry.

```
data BST a = Void | Node a (BST a) (BST a)

checkTree :: BST Int -> (Bool,Int,Int)
checkTree Void          = (True, maxBound, minBound)
```

```
checkTree (Node x l r) = check l r
  where
    check Void Void = (True,x,x)
    check Void _    = (x < minr && tr,x,maxr)
    check _ Void    = (maxl <= x && tl,minl,x)
    check _ _       = (maxl <= x && x < minr && tl && tr,minl,maxr)

    (tl,minl,maxl) = checkTree l
    (tr,minr,maxr) = checkTree r
```

Ammettendo di definire opportunamente **maxBound**, **minBound** :: **Int** in Curry, ci si può aspettare di ottenere i medesimi risultati nei 2 linguaggi?

Se no si mostri come andrebbe riscritto per ottenere gli stessi risultati.

Se sì ciò è dovuto a un motivo strutturale o a un "caso fortuito", per cui aggiungendo un caso non definito il comportamento differirebbe?