14/7/2014

Titolo nota 24/09/2015

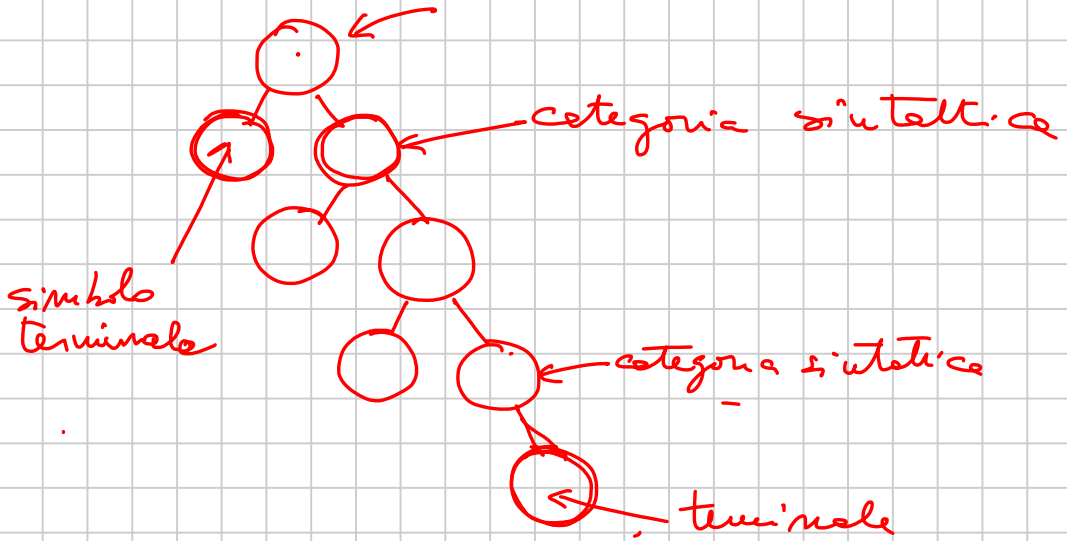
Si definisca una grammatica per il seguente ling. $\Sigma = \{a, b\}$

$$L = \{ a^n b \mid n > 0 \text{ e } n \text{ pari} \}$$

in modo che tutti gli alberi di derivazione siano alberi binari e tutti i nodi non foglia abbiano almeno un figlio costituito da una foglia.

$S \to aA$

$A \to ab \mid aS$



14/7/2014

Titolo nota 24/09/2015

Si definisca una grammatica per il seguente ling. $\Sigma = \{a, b\}$

$$L = \{ a^n b \mid n > 0 \text{ e } n \text{ pari} \}$$

in modo che tutti gli alberi di derivazione siano alberi binari e tutti i nodi non foglia abbiano almeno un figlio costituito da una foglia.

$S \to aA$

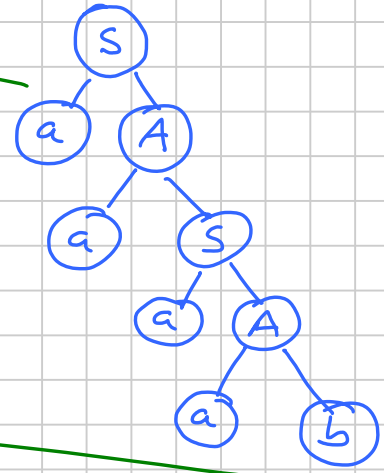
$A \to ab \mid aS$

$S \to aA$

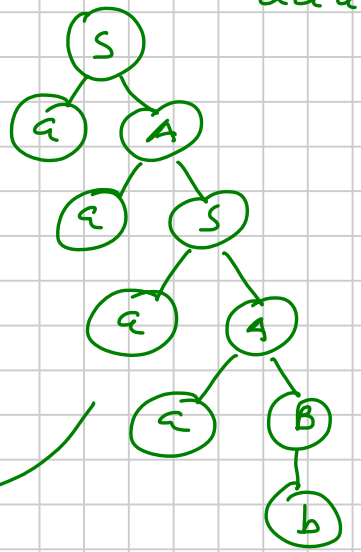
$A \to aB \mid aS$

$B \to b$

aaaab ∈ L



aaaab



Titolo nota 24/09/2015

Una barca sta sulla riva di un fiume. Azioni della barca sono:

↑ caricare un peso

↓ scaricare " "

→ andare dalla riva sinistra a quella destra

← " " " destra " " sinistra

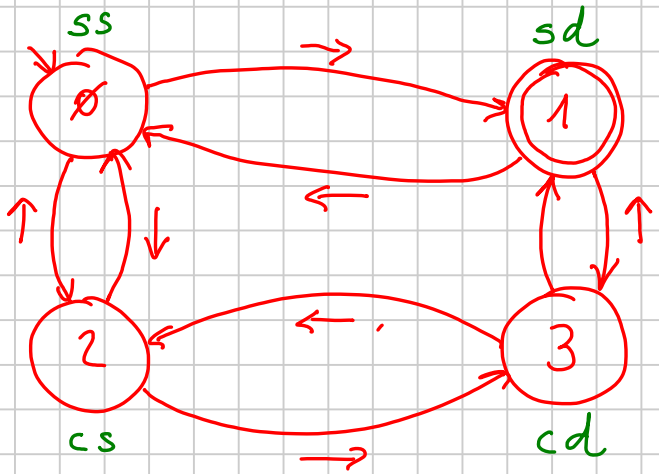
Definire un ASF sull'alfabeto $\Sigma = \{\uparrow, \downarrow, \leftarrow, \rightarrow\}$ che riconosca tutte e solo le stringhe che rappresentano sequenze di azioni della barca in modo che

partendo scarica sulla riva SINISTRA

arrivi scarica sulla riva DESTRA

es: $\rightarrow$, $\uparrow\rightarrow\downarrow\uparrow\downarrow \in L$ ε, $\uparrow\rightarrow \notin L$

Titolo nota 24/09/2015



Titolo nota 24/09/2015

Date una lista di coppie di numeri reali, in cui ogni coppia rappresenta le coordinate di un punto sul piano cartesiano, si definisca una funzione CAML

check : (float * float) list → bool

che controlla che gli elementi della lista siano ordinati in modo crescente rispetto alla loro distanza dall'origine (0,0).

sqrt : float → float (x,y) distanza dall'origine è $\sqrt{x^2+y^2}$

```
let rec check l = match l with
    []  -> true
  | [x] -> true
  | (x1,y1)::(x2,y2)::ys when sqrt(x1*x1 + y1*y1) < sqrt(x2*x2 + y2*y2)
                          -> check ((x2,y2)::ys)
  | (x1,y1)::(x2,y2)::ys when sqrt(  )>=
                              sqrt(  )
                          -> false ;;
```

```
let rec check l = match l with
    [] -> true
  | [x] -> true
  | (x1,y1)::(x2,y2)::ys -> sqrt(x1*x1 + y1*y1) < sqrt(x2*x2 + y2*y2)
                           && check ((x2,y2)::ys);;
```

```
  | (x1,y1)::(x2,y2)::ys -> if sqrt(  "  ) < sqrt(  "  )
                            then check ((x2,y2)::ys)
                            else false ;;
```

Titolo nota 24/09/2015

Si definisce una funzione C

int check (int a[], int b[], int dima, int dimb)

che dati due array di interi e le loro dimensioni, restituisce il valore di verità della seguente formula

$$\exists j \in [0, dima) . \left(\#\{ i \mid i \in [0, dimb) \wedge a[j] = b[i] \} = 3 \right)$$

numero degli elementi di b uguali a a[j]

$\exists j \in [0, dima)$. a[j] è uguale a esattamente 3 elementi di b?

Titolo nota 24/09/2015

```
int check ( . . . . . )
{ int j = 0; int trovato = 0;
  while (j < dima && !trovato)
  { int conta = 0; int i;
    for (i = 0; i < dimb; i++)
      if (b[i] == a[j]) conta = conta + 1;
    if (conta == 3) trovato = 1;
      else j = j + 1;
  }
  return trovato;
}
```

conta = 0;

```
for (i=0; i<dima && !trovato; i++)
```

NO!

Titolo nota 24/09/2015

Senza utilizzare "ricorsione esplicite" definire una funzione CAML

Utilizzando una delle funzioni di ordine superiore: map, filter, foldr

forall, exists

subst : int list → int list

che, data una liste di interi, sostituisce con ø tutti gli elementi che precedono l'ultima occorrenza del valore ø.

subst [1; 2; ø; 4; 1; ø; 2; 3] = [ø; ø; ø; ø; ø; ø; 2; 3]

Titolo nota 24/09/2015

```
let subst l =
    let f x y = match y with
                  [] -> [x]
                | s::ss -> if s=0 then 0::s::ss
                           else x::s::ss
    in foldr f [] l;;
```

risultato sulla parte rimanente (quella che segue l'elemento x)

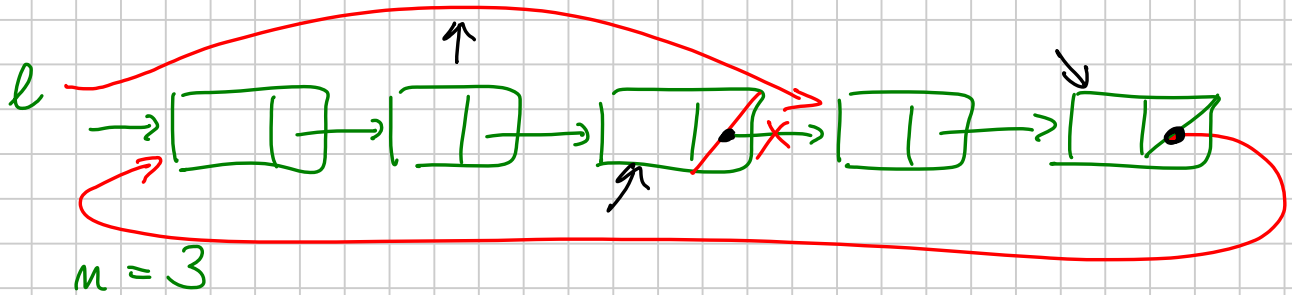
foldr f a [0;1;0;7] = f 0 (f 1 (f 0 (f 7 [])))

[0;7]

[7]

Titolo nota 24/09/2015

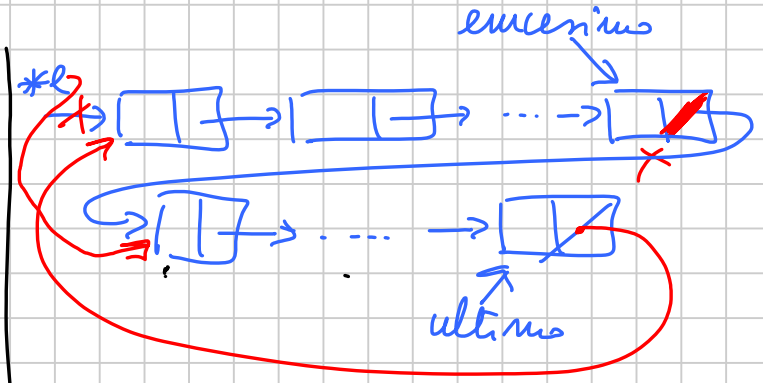
Scrivere in C una procedura che, dati in ingresso, attraverso opportuni parametri, una lista di interi e un valore intero positivo n, sposta in fondo alla lista i primi n elementi. Se n è maggiore o uguale alla lunghezza della lista, questa rimane inalterata.




```
int lunghezza (ListaDiElementi l)
{ int conta = 0; while (l != NULL) { conta = conta + 1; l = l->next; }
                 return conta;
}
```

Titolo nota 24/09/2015

```
void sposte (ListeDiElement *l, int n)
{ if (lunghezza (*l) > n)
  { ListeDiElementi ennesimo = *l;
    ListeDiElementi ultimo = *l;
    int conteelementi = 1;
    while (conteelementi < n)
      { ennesimo = ennesimo -> next;
        Conteelement = conteelementi +1;
      }
    while ( ultimo -> next != NULL)
      ultimo = ultimo -> next;
```




```
    ultimo -> next = *l
    *l = ennesimo -> next;
    ennesimo -> next = NULL;
  }
}
```