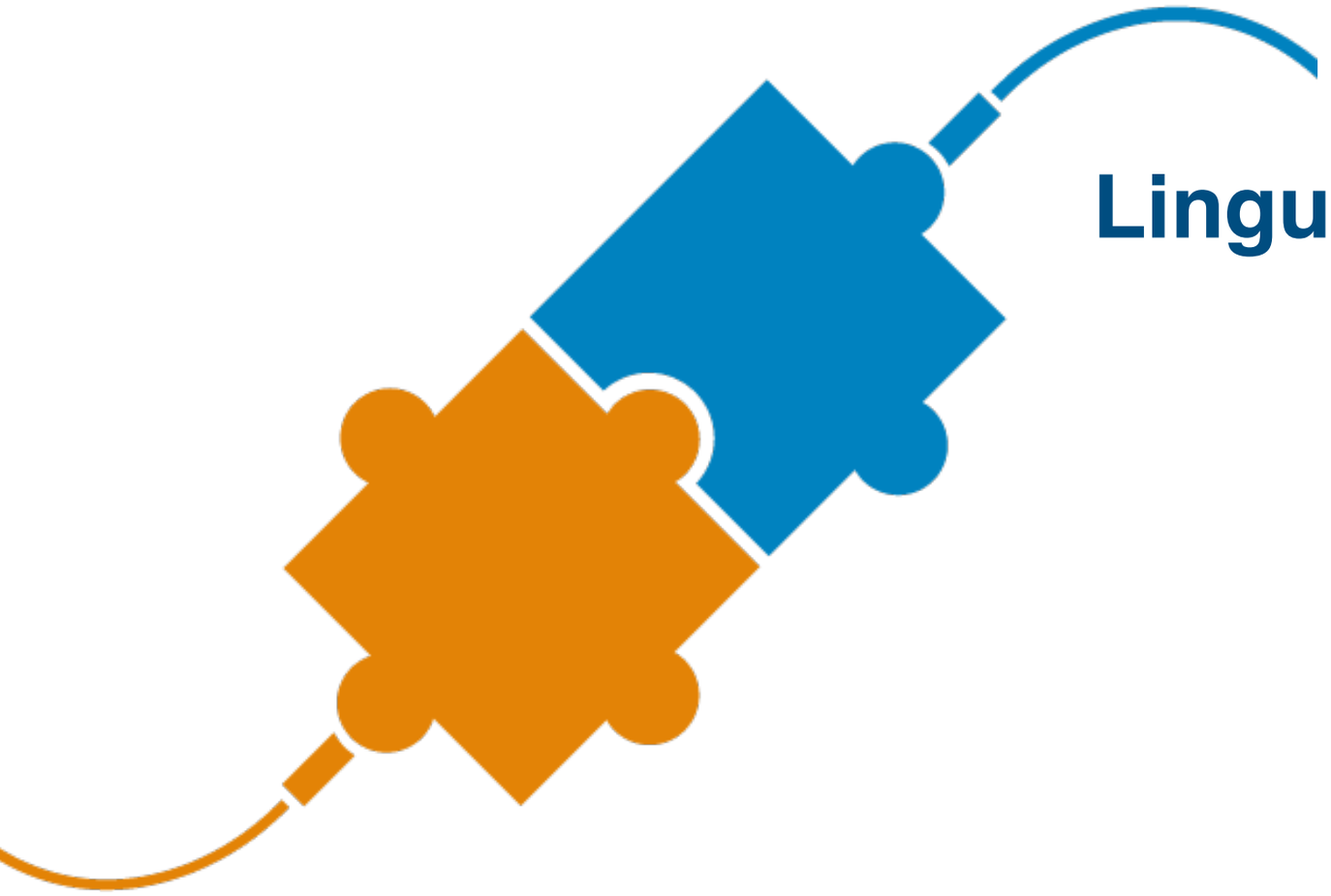


Linguaggi di Programmazione

Roberta Gori
Vincenzo Ciancia

Presentazione

Orario

Lunedi': 11:00-13:00 Fib N

Martedi': 14:00-16:00 Fib H-lab

Mercoledi': 9:00-11:00 Fib G

Giovedi': 14:00-16:00 Fib E1

I prof

Roberta Gori
Dipartimento di Informatica
roberta.gori@unipi.it

Vincenzo Ciancia
Istituto di Scienza e Tecnologie
dell'Informazione "Alessandro Faedo"
vincenzo.ciancia@isti.cnr.it

Obiettivo del corso

Iniziare a ragionare formalmente sul comportamento dei programmi

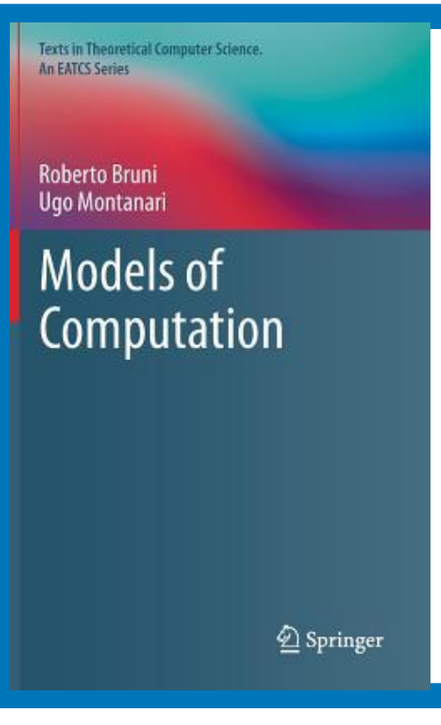
- vi presentero' diverse modelli di calcolo:
  - linguaggi imperativi,
  - linguaggi funzionali di ordine superiore,
  - linguaggi concorrenti

- vedremo i loro paradigmi di programmazione,

- e le loro descrizioni matematiche, sia concrete che astratte,

- ci focalizzeremo su alcuni strumenti/tecniche intellettuali per ragionare sui programmi.

Il libro che seguiremo



Roberto Bruni and Ugo Montanari
Models of Computation
Texts in Theoretical Computer Science (an EATCS series)
https://www.springer.com/book/9783319428987

Informazioni utili

Canale teams per comunicazioni urgenti 063AA 23/24 - LINGUAGGI DI PROGRAMMAZIONE CON LABORATORIO [MAT-L] | General | Microsoft Teams

Pagina web per materiale didattico:
http://didawiki.di.unipi.it/doku.php/matematica/lp/start

Modalita' d'esame

- Compitini
- Prova scritta (obbligatoria)
- Prova orale (facoltativa)

Cercate di partecipare!

cerchiamo di trovare insieme le risposte

```
% find the least (non-unitary) divisor p of n>0
p := 0;
x := 2;
while ( x <= n/2 ) do {
    if ( n%x == 0 ) then {
        p := x;
    } else {
        x := x+1;
    }
}
```

Cercate di partecipare!

Correggetemi se sbaglio

```
% find the index of the last occurrence of n in a
i := length(a)-1;
while ( i>0 && n!=a[i] ) do {
    i := i-1;
}
```

Prerequisiti

Teoria degli insiemi

$\varnothing$ $A \cap B$ $A \cup B$ $A \setminus B$ $\overline{A}$

$a \in A$ $A \subset B$ $A \subseteq B$ $A \times B$

$a \notin A$ $A \not\subseteq B$ $A \cap B = \emptyset$

$\mathbb{N}$ $\mathbb{Z}$ $\mathbb{Q}$ $\mathbb{R}$ $\mathbb{B}$

$N \subseteq \mathbb{N}$ $N \in \wp(\mathbb{N})$ $S \subseteq \wp(\mathbb{N})$

Prerequisiti

teoria degli insiemi: funzioni, relazioni

$f : A \to B$

$R \subset A \times B$

funzioni come relazioni

$$R_f \triangleq \{(a, f(a)) \mid a \in A\}$$

insiemi come funzioni
Dato un insieme N

$$f_N : \mathbb{N} \to \mathbb{B}$$

$$f_N(n) \triangleq \begin{cases} 1 & n \in N \\ 0 & \text{otherwise} \end{cases}$$

$$N = \{n \mid f_N(n) = 1\}$$

Prerequisiti

Logica del primo ordine

ff false 0 F tt true 1 T $P \wedge Q$ $P \vee Q$ $\neg P$

$\exists x.\ P(x)$ $\forall x.\ P(x)$ $P \Rightarrow Q$ $P \Leftrightarrow Q$

significato dell'implicazione!

$$P \Rightarrow Q$$

$$Q \vee \neg P$$

$$\neg Q \Rightarrow \neg P$$

importanza dell'ordine dei quantificatori!

$$\forall n \in \mathbb{N}.\ \exists m \in \mathbb{N}.\ n < m$$

$$\exists m \in \mathbb{N}.\ \forall n \in \mathbb{N}.\ n < m$$

Prerequisiti

Stringhe e grammatiche libere

Alfabeto A $\qquad A^n \triangleq \underbrace{A \times \cdots \times A}_{n} \qquad A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n$

$$\mathbb{B} = \{0, 1\}$$

$$\mathbb{B}^0 = \{\epsilon\}$$
$$\mathbb{B}^1 = \{0, 1\}$$
$$\mathbb{B}^2 = \{00, 01, 10, 11\}$$
$$\mathbb{B}^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$$
$$\ldots$$
$$\mathbb{B}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$$

Prerequisiti

Stringhe e grammatiche libere

Alfabeto A $\qquad A^n \triangleq \underbrace{A \times \cdots \times A}_{n} \qquad A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n$

$$\mathbb{B}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$$

$$\mathsf{A} ::= \epsilon \mid 0\ \mathsf{A} \mid 1\ \mathsf{B}$$

$$\mathsf{B} ::= 0\ \mathsf{B} \mid 1\ \mathsf{A}$$

$$\mathcal{L}(\mathsf{A}) = ?$$

$$\underline{\mathsf{A}} \to \underline{0\ \mathsf{A}} \to 0\ 1\ \underline{\mathsf{B}} \to 0\ 1\ \underline{1\ \mathsf{A}} \to 0\ 1\ 1\ \epsilon = 0\ 1\ 1$$

Prerequisiti

Definizioni ricorsive e induttive

$$0! \triangleq 1$$
$$(n+1)! \triangleq n! \cdot (n+1)$$

$$A^0 \triangleq \{\epsilon\}$$
$$A^{(n+1)} \triangleq A \times A^n$$

$$f(n) \triangleq \begin{cases} 1 & \text{if } n \leq 1 \\ f(n/2) & \text{if } n \neq 0 \wedge n\%2 = 0 \\ f(3n+1) & \text{otherwise} \end{cases}$$

Congettura di Collatz

$$f(12) = f(6) = f(3) = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1$$

Prerequisiti

Congetture vs teoremi

un numero naturale p è primo
se non può essere scritto come il prodotto di due numeri più piccoli

n	Is n prime?	$2^n - 1$	Is $2^n - 1$ prime?
2	yes	3	yes
3	yes	7	yes
4	no: $4 = 2 \cdot 2$	15	no: $15 = 3 \cdot 5$
5	yes	31	yes
6	no: $6 = 2 \cdot 3$	63	no: $63 = 7 \cdot 9$
7	yes	127	yes
8	no: $8 = 2 \cdot 4$	255	no: $255 = 15 \cdot 17$
9	no: $9 = 3 \cdot 3$	511	no: $511 = 7 \cdot 73$
10	no: $10 = 2 \cdot 5$	1023	no: $1023 = 31 \cdot 33$

Prerequisiti

Congetture vs teoremi

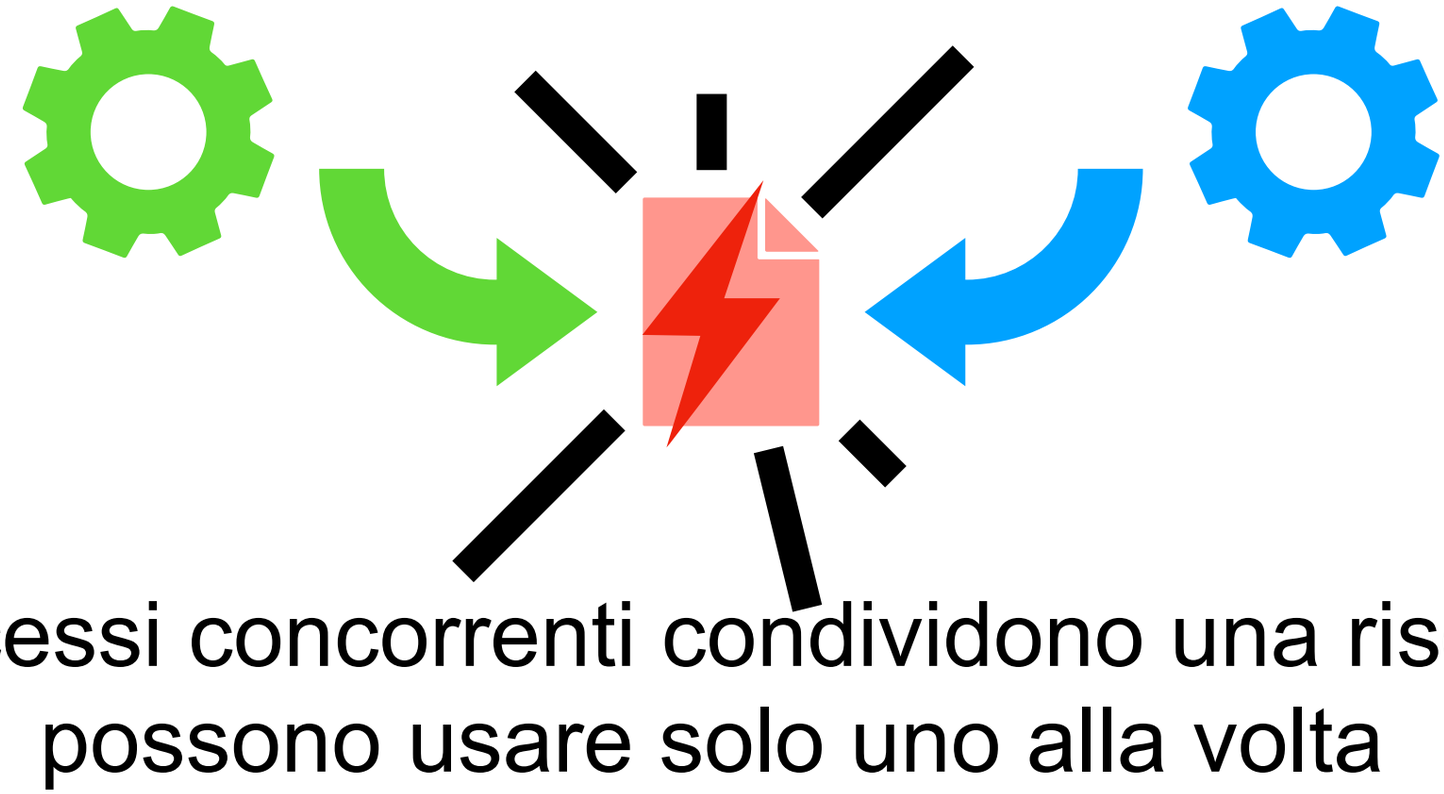
if p is prime
then $2^p - 1$ is prime

if $n > 1$ is not prime
then $2^n - 1$ is not prime

Usate qualsiasi mezzo per provare o confutare le congetture di cui sopra

Un antipasto

Il problema



Due processi concorrenti condividono una risorsa che possono usare solo uno alla volta

Possono comunicare usando la memoria condivisa

Vogliamo garantire che non ci siano conflitti quando i processi accedono alla risorsa

Non vogliamo imporre una rigida alternanza di turni senza motivo

Algoritmo di mutua esclusione di Peterson (1981)

```
% Due processi P1, P2
% Due variabili booleane b1, b2 (valori true o false, all'inizio false)
% quando il processo Pi vuole accedere alla risorsa setta la variabile
% bi a true
% Una variabile Booleana k, che avra' valori in {1,2}
% (inizialmente di valore arbitrario)
% ci dice che  Pk ha prioriza' sull'altro processo%
% Processo P1 in pseudocodice
while (true) {
    ...                          % fa altre cose prima
    b1 := true ;                 % P1 vuole accedere alla risorsa
    k := 2 ;                     % P1 da la priority all'altro processo
    while (b2 && k==2) skip ;    % P1 aspetta il suo turno
    <critical section>           % P1 entra in possesso della risorsa
    b1 := false                  % P1 rilascia la risorsa
}

% P2 e' analogo: i ruoli di b1 e b2 sono scambiati, dove l'algoritmo
setta b1 settera' invece  b2 e dove controlla b2 controllera' invece b1
 e  gli assegnamenti e controlli della variabile k coinvolgeranno il
valore 1 invece del valore 2
```

Domande

L'algoritmo di Peterson funziona?

Cosa significa "funziona"? Cosa ci aspettiamo?

(Progresso)

Se la risorsa è disponibile, nessun processo è costretto ad aspettare

(Attesa limitata)

Nessun processo aspetterà per sempre la risorsa

(*altrimenti la soluzione più semplice è che nessuno entri*)

(Mutua Esclusione)

P1 e P2 non accedono alla risorsa allo stesso tempo

Algoritmo di mutua esclusione di Hyman (1966)

```
% Due processi H1, H2
% Due variabili booleane b1, b2 (valori true o false, all'inizio false)
% quando il processo Hi vuole accedere alla risorsa setta la variabile
% bi a true
% il processoHk ha priorita' sull'altro processo
%
% Processo H1 in pseudocodice
while (true) {
    ...                          % fa altre cose prima
    b1 := true ;                 % H1 vuole accedere alla risorsa
    while (k==2) {               % finche' H2 ha la priorita'
        while (b2) skip ;        % H1 aspetta
        k := 1;                  % H1 setta la priorita' per se stesso
   }
    <critical section>           % H1 accede alla risorsa
    b1 := false                  % H1 rilascia la risorsa
}

% H2 e' analogo: i ruoli di b1 e b2 sono scambiati, dove l'algoritmo
setta b1 settera' invece b2 e dove controlla b2 controllera' invece b1 e
analogamente il ruolo di 1 e 2 sono scambiati negli assegnamenti e
controlli della variabile k
```

La domanda

L'algoritmo di Peterson soddisfa la mutua esclusione?

L'algoritmo di Hyman soddisfa la mutua esclusione?

Per le risposte siate pazienti e aspettate le lezioni di fine corso