PSC 2023/24 (375AA, 9CFU)

Principles for Software Composition

Roberto Bruni
http://www.di.unipi.it/~bruni/

http://didawiki.di.unipi.it/doku.php/magistraleinformatica/psc/start

01 - Introduction
Classes

In presence, every

Tuesday: 16:00-18:00, C1

Thursday: 16:00-18:00, M1

Friday: 14:00-16:00, C1
Course material
Who am I?

http://www.di.unipi.it/~bruni

Office hours: by appointment preferably Tuesday 14:00-16:00

bruni@di.unipi.it
Research topics (theses?)

False alarm detection in Abstract Interpretation
Formal approaches to code obfuscation
Quantum Computation and concurrency models
Modelling and analysis of biological systems
Graphical specification languages
Algebraic approaches to structured graphs
Rewrite rules for reversible languages
Who are you?

First Name: ____________________________
Last Name: ____________________________
Enrollment number: ____________________
email: _________________________________
Bachelor degree: _______________________
MSc course of enrollment: _____________

Please fill the form!
Who are you?

First Name: John
Last Name: Smith
Enrollment number: 123456
email: john.smith@email.com
Bachelor degree: Comp. Sci., Pisa, IT
MSc course of enrollment: Comp. Sci. - SW

Please fill the form!
Interaction protocol

Do you know what are the 3 possible answers to the question “does my program c satisfy the property ψ?”

☐ yes
☐ no
☐ don’t know

(you MUST select one of them to answer questions in these classes)
The Course
Some quotes

Computer science is no more about computers than astronomy is about telescopes
- Edsger W. Dijkstra

Studying programming languages without formal semantics would be like studying physics without math
- from the web

All models are wrong, but some are useful
- George Box

Subjects are divided in two categories:
1) too difficult matters, that CANNOT be studied
2) easy matters, that DO NOT NEED to be studied
- back of a t-shirt
Objectives

Programming paradigms
(imperative, declarative, higher order, concurrent, mobile, stochastic)

Mathematical frameworks
(concrete & abstract)
(domains, inference rules, transition systems, λ-calculus, process algebras)

Understand
(recursion, semantics, compositionality)

Reason
(induction, modal and temporal logics, behavioural and logical equivalences)

Explain
(correctness, compliance, performance)
The approach

(in their simplest form, still Turing equivalent)

programming paradigms \textbullet\textbullet\textbullet mathematical frameworks \textbullet\textbullet\textbullet meta-properties + proof techniques

(for all programs or just some classes of programs)

models

programs \textbullet\textbullet\textbullet specifications
Key question

Given two programs $p$ and $q$:

Do they behave the same?

Is it safe to replace one with the other in any context?
Textbooks

Roberto Bruni and Ugo Montanari
Models of Computation
Texts in Theoretical Computer Science (an EATCS series)
Course activities

attend classrooms:
ask questions!
(sleep quietly)

learn theorems:
(drink many coffees)

do some thinking:
solve ALL your homeworaks
(at least try to)

give the exam:
time for a party!
Be proactive!

Let’s spell out definitions together

% find the least (non-unitary) divisor $p$ of $n>1$
$p := 0;$
$x := 2;$
while (_______________________________) do {
    if ( $n \% x == 0$ ) then {
        $p := x;$
    } else {
        $x := x + 1;$
    }
}
Be proactive!

Correct me if I’m wrong

```plaintext
% 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;
}
```
Be proactive!

Sometimes tricky questions!

Can you find the mistake?

1 2 3 4 5 6 7 8 9
Exam

In past years, the evaluation was based on written and oral exams.

Since the covid-19 emergency, and for the current period, the evaluation will be solely based on a final oral exam.

Registration to exams is mandatory: https://esami.unipi.it/esami

The exam will typically consist of:
1. three to four preliminary questions
2. one exercise (analogous to past written exams)
3. redoing one of the proofs seen in the course
4. some additional questions

The list of preliminary questions is available on Microsoft Teams, in the File tab (PSC-questions.txt)
A sample exam

What is a complete partial order?

What are the rules of the type system of HOFL?

How is iteration achieved in CCS?

Why only positive normal forms are considered in the mu-calculus?

Consider the HOFL term

\[
t \overset{\text{def}}{=} \text{rec } f. \lambda x. \text{ if } x \text{ then } (x, \text{fst}(f \ x)) \text{ else } (\text{snd}(f \ x), x)
\]

1. Find the principal type of \( t \).

2. Find the denotational semantics of \( t \).

Prove the Switch Lemma

Given the initial state distribution and a DTMC, how do we compute the state distribution at time 3?
Badges

No mid-terms
No self-evaluation tests
During the course: some “badge” exercises

Submit your solutions by email to earn bronze / silver / gold badges
(no extra scores, but be proud of yourselves)
# Prerequisites

## Basic set theory

<table>
<thead>
<tr>
<th>$\emptyset$</th>
<th>$A \cap B$</th>
<th>$A \cup B$</th>
<th>$A \setminus B$</th>
<th>$\overline{A}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a \in A$</td>
<td>$A \subset B$</td>
<td>$A \subseteq B$</td>
<td></td>
<td>$A \times B$</td>
</tr>
<tr>
<td>$a \not\in A$</td>
<td>$A \not\subset B$</td>
<td></td>
<td>$A \cap B = \emptyset$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathbb{N}$</th>
<th>$\mathbb{Z}$</th>
<th>$\mathbb{Q}$</th>
<th>$\mathbb{R}$</th>
<th>$\mathbb{B}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N \subseteq \mathbb{N}$</td>
<td>$N \in \wp(\mathbb{N})$</td>
<td></td>
<td>$S \subseteq \wp(\mathbb{N})$</td>
<td></td>
</tr>
</tbody>
</table>
Prerequisites

Basic set theory: functions, relations

\[ f : A \rightarrow B \]

\[ R \subseteq A \times B \]

functions as relations

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

sets as functions (characteristic function)

\[ f_N : \mathbb{N} \rightarrow \mathbb{B} \]

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

\[ N = \{n \mid f_N(n) = 1\} \]
## Prerequisites

First order logic

<table>
<thead>
<tr>
<th>false</th>
<th>true</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

\( \exists x. P(x) \quad \forall x. P(x) \)

\( P \Rightarrow Q \quad P \iff Q \)

**meaning of implication!**

\[ P \Rightarrow Q \]

\[ Q \lor \neg P \]

\[ \neg Q \Rightarrow \neg P \]

**order of quantifiers matters!**

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

\[ \exists m \in \mathbb{N}. \forall n \in \mathbb{N}. n < m \]
Prerequisites

Strings and context-free grammars

Alphabet

\[ A \]

\[ A^n \triangleq A \times \cdots \times A \]

\[ A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n \]

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

Strings and context-free grammars

Alphabet

\[ A^n \triangleq A \times \cdots \times A \]

\[ A^* \triangleq \bigcup_{n \in \mathbb{N}} A^n \]

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

\[
A ::= \epsilon \mid 0 A \mid 1 B \\
B ::= 0 B \mid 1 A
\]

\[
A \rightarrow 0 A \rightarrow 0 1 B \rightarrow 0 1 1 A \rightarrow 0 1 1 \epsilon = 0 1 1
\]

\[ \mathcal{L}(A) = ? \]

\[ \mathcal{L}(B) = ? \]
Prerequisites

Inductive and recursive definitions

\[
\begin{align*}
0! & \triangleq 1 \\
(n + 1)! & \triangleq n! \cdot (n + 1)
\end{align*}
\]

\[
\begin{align*}
A^0 & \triangleq \{\epsilon\} \\
A^{n+1} & \triangleq A \times A^n
\end{align*}
\]

\[
f(n) \triangleq \begin{cases} 
1 & \text{if } n \leq 1 \\
f(n/2) & \text{if } n > 1 \land n \% 2 = 0 \\
f(3n + 1) & \text{otherwise}
\end{cases}
\]

\[
f(12) = f(6) = f(3) = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1
\]
Prerequisites

Conjectures vs theorems

A natural number $p$ is **prime** if it cannot be written as the product of two smaller numbers.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Is $n$ prime?</th>
<th>$2^n - 1$</th>
<th>Is $2^n - 1$ prime?</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>yes</td>
<td>3</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>yes</td>
<td>7</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>no: $4 = 2 \cdot 2$</td>
<td>15</td>
<td>no: $15 = 3 \cdot 5$</td>
</tr>
<tr>
<td>5</td>
<td>yes</td>
<td>31</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>no: $6 = 2 \cdot 3$</td>
<td>63</td>
<td>no: $63 = 7 \cdot 9$</td>
</tr>
<tr>
<td>7</td>
<td>yes</td>
<td>127</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>no: $8 = 2 \cdot 4$</td>
<td>255</td>
<td>no: $255 = 15 \cdot 17$</td>
</tr>
<tr>
<td>9</td>
<td>no: $9 = 3 \cdot 3$</td>
<td>511</td>
<td>no: $511 = 7 \cdot 73$</td>
</tr>
<tr>
<td>10</td>
<td>no: $10 = 2 \cdot 5$</td>
<td>1023</td>
<td>no: $1023 = 31 \cdot 33$</td>
</tr>
</tbody>
</table>
Prerequisites

Conjectures vs theorems

if \( p \) is prime
then \( 2^p - 1 \) is prime

if \( n > 1 \) is not prime
then \( 2^n - 1 \) is not prime

Use any mean to prove or disprove the above conjectures
Your background?

Please fill the form about “Familiar subjects”
An Appetiser
Two concurrent processes share a single-use resource

They can communicate using shared memory

We want to guarantee that there are no conflicts when the processes access the resource

No strict alternation of naive turn taking is imposed
Peterson’s mutual exclusion algorithm (1981)

% Two processes P1, P2
% Two boolean variables b1, b2 (both initially false)
% when Pi wants to enter the critical section, then it sets bi to true
% An integer variable k, taking values in {1,2}
% (initial value is arbitrary)
% the process Pk has priority over the other process
%
% Process P1 in pseudocode
while (true) {
    ... % non critical section
    b1 := true ; % P1 wants to enter the critical section
    k := 2 ; % P1 gives priority to the other process
    while (b2 && k==2) skip ; % P1 waits its turn
    ... % P1 enters the critical section
    b1 := false % P1 leaves the critical section
}

% Process P2 is analogous to P1
Which question?

Does Peterson’s algorithm work?

What does it mean that “it works”? What do we expect?

(Progress)
If the resource is available, no process is forced to wait

(Bounded Waiting)
No process will wait forever for the resource
(otherwise the easiest solution is no one gets in)

(Mutual Exclusion)
P1 and P2 are never in the critical section at the same time
Hyman’s mutual exclusion algorithm (1966)

% Two processes H1, H2
% Two boolean variables b1, b2 (both initially false)
% when Hi wants to enter the critical section, then it sets bi to true
% An integer variable k, taking values in {1,2}
% (initial value is arbitrary)
% the process Hk has priority over the other process
%
% Process H1 in pseudocode
while (true) {
  ...
  b1 := true ; % H1 wants to enter the critical section
  while (k==2) {
    while (b2) skip ; % H1 waits
    k := 1; % H1 sets priority to itself
  }
  ...
  b1 := false % H1 enters the critical section
}

% Process H2 is analogous to H1
The question

Does Peterson’s algorithm satisfy mutual exclusion?

Does Hyman’s algorithm satisfy mutual exclusion?

For the answers be patient and wait early-May lectures