Software Verification: Contracts, Trusted Components and Patterns

ETH Zürich

Date: 15 December 2008

Surname, first name: ....

Student number: ....

I confirm with my signature, that I was able to take this exam under regular circumstances and that I have read and understood the directions below.

Signature: ....

Directions:

- Exam duration: 1 hour 45 minutes.
- Except for a dictionary you are not allowed to use any supplementary material.
- All solutions can be written directly on the exam sheets. If you need more space for your solution ask the supervisors for a sheet of official paper. You are **not** allowed to use other paper. Please write your student number on **each** additional sheet.
- Only one solution can be handed in per question. Invalid solutions need to be crossed out clearly.
- Please write legibly! We will only correct solutions that we can read.
- Manage your time carefully (take into account the number of points for each question).
- Don't forget to include header comments in features.
- Please **immediately** tell the exam supervisors if you feel disturbed during the exam.

Good luck!

Question	Number of possible points	Points
1	20	
2	15	
3	15	
4	10	
5	10	
Total	70	

1 Axiomatic semantics (20 points)

Consider the following Hoare triple:

$\{x > 0\}$
```
y := 1;
z := 0;
while (z != x) do
    z := z + 1;
    y := y * z
end
```
$\{y = x!\}$

The ! in the postcondition denotes the factorial function, i.e. $x! = x \cdot (x-1) \cdot (x-2) \cdot \ldots \cdot 1$ and $0! = 1$. Prove that this triple is a theorem of Hoare's axiomatic system for partial correctness. The proof should be a sequence of lines with three elements on each line: line number; proposition; justification.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

2 Program analysis (15 Points)

The assignment to variable v by statement S of program $Prog$ **reaches** a point p in $Prog$ if there exists a control-flow path from S to p on which no statement reassigns v. This can be formulated as a labelling scheme on control-flow graphs:

- A *label* is a pair $(varname : statementnumber)$, where $varname$ is a variable of $Prog$ and $statementnumber$ the number of a node in the control-flow graph of $Prog$. Each node S is numbered with a unique positive integer, $number(S)$.
- Each node S has two sets of labels: the incoming label set $In(S)$ and the outgoing label set $Out(S)$:

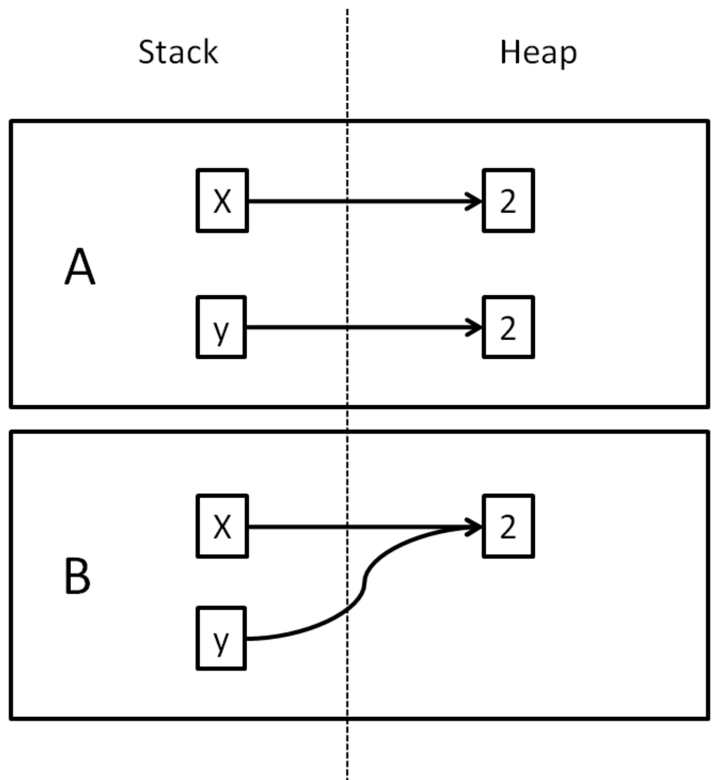
$$
\begin{aligned}
In(S) &= \emptyset \text{ if } S \text{ is the node in } Prog \text{ at which control-flow starts.}\\
&= \bigcup_{S_0 \in pred(S)} Out(S_0) \text{ otherwise, where } pred(S) \text{ denotes the set of all nodes with edges pointing to } S.\\
Out(S) &= (In(S) - \{(varname : n) | n \in \mathbb{N}\}) \cup \{(varname : number(S))\}\\
&\quad \text{if } S \text{ is of the form } varname := expression.\\
&= In(S) \text{ otherwise.}
\end{aligned}
$$

Draw the control-flow graph of the following program fragment and annotate its nodes with reachability labels:

```
a := 2
b := -a
if b <= a then
    a := b * 2
    b := a
else
    b := b + 4
end
b := b + 1
```

3 Separation logic (15 Points)

1. (8 points) Consider program states A and B in the following figure:



Indicate in the table whether or not a given assertion is satisfied by states A and B respectively. Indicate satisfaction with a `T` and non-satisfaction with an `F`.

	A	B
$x \mapsto 2$		
$y \mapsto 2 * true$		
$x \mapsto 2 * y \mapsto 2$		
$x \mapsto 2 \land y \mapsto 2$		

2. (4 points) Do the following implications hold? If an implication holds, explain why. If it does not hold, provide a counterexample.

$$(P \land Q) \Rightarrow (P * Q) \quad (1)$$
$$(P * Q) \Rightarrow (P \land Q) \quad (2)$$

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

3. (3 points) Consider the following derivation attempt:

$$\frac{\{a \mapsto 30\} b := [a] \{a \mapsto 30 \wedge b = 30\}}{\{(a \mapsto 30) * b \mapsto 45\} b := [a] \{(a \mapsto 30 \wedge b = 30) * b \mapsto 45\}}$$

Explain why the frame rule was wrongly applied.

. . . .

. . . .

. . . .

. . . .

. . . .

. . . .

4 Abstract interpretation (10 Points)

Consider the grammar of integer expressions

$$e \quad ::= \quad i \quad | \quad e_1 + e_2 \quad | \quad e_1 - e_2 \quad | \quad e_1 * e_2$$

where $i \in I$ and $I = \{-1000, -999, \ldots, 999, 1000\}$.

Devise an abstract interpretation scheme to determine whether a given e represents an even or odd integer. You may assume the existence of a function $f : I \rightarrow \{even, odd\}$ that maps i to *even* if i is even and i to *odd* if i is odd.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

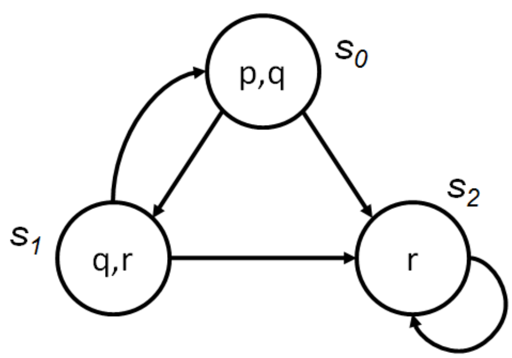
5 Model checking (10 Points)

Here is the semantics of a subset of LTL formulas:

For a path $\pi = s_1 \rightarrow s_2 \rightarrow \ldots$ in a model $M = (S, \rightarrow, L)$ and an LTL formula ϕ:
$\pi \vDash true$
$\pi \nvDash false$
$\pi \vDash p$ iff $p \in L(s_1)$
$\pi \vDash \neg\phi$ iff $\pi \nvDash \phi$
$\pi \vDash \phi_1 \wedge \phi_2$ iff $\pi \vDash \phi_1$ and $\pi \vDash \phi_2$
$\pi \vDash \phi_1 \vee \phi_2$ iff $\pi \vDash \phi_1$ or $\pi \vDash \phi_2$
$\pi \vDash \phi_1 \Rightarrow \phi_2$ iff $\pi \vDash \phi_2$ whenever $\pi \vDash \phi_1$
$\pi \vDash \mathrm{X}\ \phi$ iff $\pi^2 \vDash \phi$ $\quad (\pi^i = s_i \rightarrow s_{i+1} \rightarrow \ldots)$
$\pi \vDash \mathrm{G}\ \phi$ iff for all $i \geq 1$, $\pi^i \vDash \phi$
$\pi \vDash \mathrm{F}\ \phi$ iff there is some $i \geq 1$ such that $\pi^i \vDash \phi$
$\pi \vDash \phi_1\ \mathrm{U}\ \phi_2$ iff there is some $i \geq 1$ such that $\pi^i \vDash \phi_2$ and for all $1 \leq j < i$, $\pi^j \vDash \phi_1$

$M, s \vDash \phi$ for a state $s \in S$ iff for every path π in M starting at s we have $\pi \vDash \phi$.

1. (6 points) Consider the transition system M:



Do the following statements hold? If yes, provide a brief justification, if no, provide a counterexample path.

(a) $M, s_0 \vDash \mathrm{X}\ (q \wedge r)$

. . . .

. . . .

. . . .

. . . .

(b) $M, s_0 \models \mathrm{G}\ \neg(p \wedge r)$

. . . .

. . . .

. . . .

. . . .

(c) $M, s_0 \models \mathrm{G}\ \mathrm{F}\ p$

. . . .

. . . .

. . . .

. . . .

2. (4 points) Express the following specifications as LTL formulas:

(a) A certain process will eventually be permanently `deadlock`ed.

. . . .

. . . .

(b) A downwards travelling lift at the fifth floor with passengers wishing to go to the second floor does not change its direction until it reaches the second floor.

. . . .

. . . .

. . . .

. . . .