Implements and Compositionality

Shankar

November 6, 2014

Overview

- Definitions and conventions
- Definition of "program A implements program B"
- Compositionality theorem
  - if
    - C uses A implements D
    - A' implements A
  - then
    - $C[A/A']$ implements D
    - $C[A/A']$ satisfies any property of $C - A$
- Program version of "A implements B"
  - B: service w/o internal nondeterminism
- Program version of "A implements B"
  - B: service with internal nondeterminism

Outline

Definitions and Conventions

Implements: evolution-based definition

Compositionality theorem

Program-based implements: internally-deterministic service

Program-based implements: internally-nondeterministic service

Definitions recap – 1

- Input is allowed at a state if its input assumption holds
- Evolution is allowed if each of its inputs is allowed
- Evolution is complete if it satisfies progress assumption
- Input is acceptable at a state if does not cause fault
  - aka accepts input
- Transition is enabled at a state if it is outgoing from the state
- *ext*(x): io sequence of x // same as *io*(x)

- For program B and evolution x of program A
  - x safe wrt B:
    x is fault-free and
    B has evolution y st $ext(x) = ext(y)$
  - x complete wrt B:
    x is fault-free and
    B has complete evolution y st $ext(x) = ext(y)$
- For composite system $C \supseteq$ composite system P
  - state/transition/evolution u of $C \rightarrow$ image $u.P$ on P

Abbreviations

- ff: fault-free
- x.end: end state of evolution x
- A.progress: progress assumption of program A
- swrt: safe wrt
- cwrt: complete wrt
- fin: finite

Outline

Implements definition

- programs A, B
- x: A-evolution swrt B
- t: A-transition

// conventions

- A implements B

  Safety

  - if B's instantiation ff then A's instantiation ff
  - for finite x, input e of B, $x \circ \langle e \rangle$ swrt B:
    A accepts e at x.end
  - for finite x, output or internal t enabled at x.end:
    t is ff and if t outputs e then $x \circ \langle e \rangle$ is swrt B

  Progress

  - for x satisfying A.progress: x cwrt B

Outline

Compositionality theorem

- programs A, C, D, A', C'
- C instantiates A
- C implements D
- A' implements A
- C': C with A replaced by A'
- $\widehat{C}$: composite system $C - A$

- Theorem 7.1
  - C' implements D
  - // C' equivalent to C wrt $\widehat{C}$-properties

    if x' is an evolution of C' safe wrt D,
    then C has an evolution x st $x.\widehat{C}$ equals $x'.\widehat{C}$

    if x' also satisfies C'.progress, then x satisfies C.progress

Outline

Overview

- Let x' be evolution of C' swrt D // $x'.\widehat{C},\ x'.A'$
- Show A has evolution x_A with ioseq of $x'.A'$ // lemma 7.1
- Stitch x_A and $x'.\widehat{C}$ into an evolution x of C
- Show: if x' violates "C' implements D"
  then x violates "C implements D"
- Details for lemma 7.1, outlines for the rest

- Conventions
  - for C: evolutions x, y, transition t
  - for C': x', y', t'
  - for A: x_A, y_A, t_A :
  - for A': x'_A, y'_A, t'_A

Lemma 7.1: if x' is safe wrt D, then $x'.A'$ is safe wrt A

Proof: induction on # transitions in x'

- Base case: C' instantiation is fault-free
  - holds if C' instantiation does not start A' // C-impl-D
  - holds if C' instantiation starts A' // C-impl-D, A'-impl-A
- Let $y' = x' \circ \langle t' \rangle$ be safe wrt D
  - x' safe wrt D
  - $x'.A'$ safe wrt A // induction hyp
  - $\exists\ x_A$ st $ext(x_A) = ext(x'.A')$ // swrt defn
  - $\exists\ x$ st $x.A = x_A$ and $x.\widehat{C} = x'.\widehat{C}$ // stitching x_A and $x'.\widehat{C}$

- Consider different cases of t'
- t' not involving A'
  - $t'.\widehat{C}$ enabled at $x'.\widehat{C}$.end, hence at $x.\widehat{C}$.end
  - $t'.\widehat{C}$ ff if output or internal // C-impl-D, x' swrt D
  - $t'.\widehat{C}$ ff if input // C-impl-D, y' swrt D
  - so y' ff, so $y'.A'$ ff and swrt A
- output t' involving A'
  - $t'.A'$ output transition of A'
  - $y'A'$ ff and swrt A // A'-impl-A, $x'.A'$ swrt A
- internal t' involving A': same as t' output

Lemma 7.1 – 3

- input t' involving A', rcving input e
  - $x' \circ \langle e \rangle$ swrt D, hence $x \circ \langle e \rangle$ swrt D // y' swrt D
  - so A accepts e at $x.A.\mathrm{end}$ // C-impl-D
    so A' accepts e at $x'.A'.\mathrm{end}$ // A'-impl-A
  - so $t'.A'$ ff, so $y'.A'$ swrt A
- composite internal t': A' outputs e to $\widehat{C}$
  - $t'.A'$ output transition of A'
  - $x'.A' \circ \langle e \rangle$ swrt A // A'-impl-A
  - $t'.\widehat{C}$ ff // A can output e at $x.\mathrm{end}$, C-impl-D
  - so $t'.A'$ ff, so $y'.A'$ swrt A
- composite internal t': A' inputs e from $\widehat{C}$
  - similar

Theorem 7.1 proof

- Let x' be swrt D
- Let x_A be st $ext(x_A) = ext(x'.A')$ // $\exists$ by lemma 7.1
- Let x be stitch of x_A and $x'.\widehat{C}$
- Suppose $x' \circ t'$ violates C'-impl-D safety
  - if t' of $\widehat{C}$: not C-impl-D // due to $\widehat{C}$ at $x.\widehat{C}$.end
  - if t' of A'
    - if due to A': not A'-impl-A
    - if not due to A': not C-impl-D
  - if t' is $\widehat{C}$-A' interaction: similar
- Suppose x' satisfies C'-impl-D safety but not progress
  - then x satisfies C-impl-D safety but not progress

Theorem

- B: service program without internal params
- A: candidate implementation program
- $\overline{B}$: service inverse program
- $A\overline{B}$: closed program of A-system a and $\overline{B}$-system $\overline{b}$

- Theorem 7.2
  - "A implements B" safety condition holds
    iff $A\overline{B}$ is fault-free
    iff $A\overline{B}$ satisfies for every ic{P} in $\overline{b}$
    Inv (thread at $\overline{b}$.ic{P}) $\Rightarrow$ $\overline{b}$.P
  - "A implements B" progress condition holds
    iff $A\overline{B}$ satisfies $\overline{b}$.progress

"If" safety

if $A\bar{B}$ ff then "A implements B" safety holds

- assume "A implements B" safety does not hold
- exists finite x_A swrt B with a faulty extension t_A
  - A cannot accept input e swrt B, or
  - A does faulty internal/output transition, or
  - A outputs f not swrt B
- exists finite x_B with same ioseq as x_A // swrt defn
- exchange inputs and outputs in x_B to get $x_{\bar{B}}$
- stitch $x_{\bar{B}}$ and x_A to get $x_{A\bar{B}}$,
  at the end of which the faulty t_A is doable
- so $A\bar{B}$ is faulty

"Only if" safety

if "A implements B" safety holds then $A\overline{B}$ ff

- assume $A\overline{B}$ has ff evolution $x_{A\overline{B}}$ and faulty extension $t_{A\overline{B}}$
- $x_{A\overline{B}}.a$ is ff evol of A
- $x_{A\overline{B}}.\overline{b}$ is ff evol of $\overline{B}$
- exchange inputs and outputs in $x_{A\overline{B}}.\overline{b}$ to get x_B of B
- if $t_{A\overline{B}}.a$ is faulty (input, internal, output) transition:
  "A implements B" does not hold
- if $t_{A\overline{B}}.a$ is ff but outputs e not accepted by $\overline{B}$:
  B does not accept e at $x_B.end$
  no other B-evolution with ioseq of x_B // internally deterministic
  so output not swrt B, so "A implements B" does not hold

Given $A\overline{B}$ ff:
$A\overline{B}$ satisfies $\overline{B}$.progress iff "A implements B" progress holds

- if "A implements B" safety but not progress holds
  then $A\overline{B}$ does not satisfy $\overline{B}$.progress
  - similar to "if" safety proof

- if $A\overline{B}$ does not satisfy $\overline{B}$.progress
  then "A implements B" progress does not hold
  - similar to "only if" safety proof

Approach

- Given
  - B: service program with internal params (in output parts)
  - A: candidate implementation program
- Construct B-externalized ($\hat{B}$)
  - augment output with any internal parameter values
  - call/ret (*extparam*) $\rightarrow$ call/ret (*extparam*, *intparam*)
- Construct A-externalized ($\hat{A}$)
  - call/ret (*extparam*) $\rightarrow$ call/ret (*extparam*, *intparam*)
  - add auxiliary code to compute *intparam* value
    - i.e., augment A with an auxiliary externalizer fn Ψ
    - call/ret (*extparam*) $\rightarrow$ call/ret (*extparam*, Ψ)
- Check for "$\hat{A}$ implements $\hat{B}$" using theorem 7.2

- Theorem 7.3
  - if safety condition of "$\hat{A}$ implements $\hat{B}$" holds
    then safety condition of "A implements B" holds
  - if progress condition of "$\hat{A}$ implements $\hat{B}$" holds
    then progress condition of "A implements B" holds
- Theorem 7.4
  - if A implements B then
    there is an externalizer Ψ s.t. $\hat{A}$ implements $\hat{B}$

Theorem 7.3: if $\hat{A}$ implements $\hat{B}$ then A implements B

- Externalizer Ψ is auxiliary
  - so evol x of A maps 1-1 to evol $\hat{x}$ of $\hat{A}$ $\star$
- Let x be evolution of A swrt B
- Show that $\hat{x}$ is swrt $\hat{B}$ // lemma 7.6
- So extension of $\hat{x}$ is swrt $\hat{B}$ // $\hat{A}$-impl-$\hat{B}$
- So extension of x is swrt B // $\star$
- Similarly for x cwrt B

- x: evolution of A
- quantity z in A $\longleftrightarrow$ quantity $\hat{z}$ in $\hat{A}$

// conventions

Let $\hat{A}$ implement $\hat{B}$. If evol x of A swrt B then $\hat{x}$ swrt $\hat{B}$

Proof

- Let evol x be swrt B
- x is ff // swrt defn
- $\hat{x}$ ff // $\star$
- inputs in $\hat{x}$ swrt $\hat{B}$ // x swrt B, inputs same in x and $\hat{x}$
- $\hat{x}$ swrt $\hat{B}$ // $\hat{A}$-impl-$\hat{B}$, $\hat{x}$ inputs swrt $\hat{B}$

if $\hat{A}$-impl-$\hat{B}$ safety holds then A-impl-B safety holds

- let x be swrt B
- $\hat{x}$ swrt $\hat{B}$ // lemma 7.6
- let e be input of B st $x \circ \langle e \rangle$ swrt B
  - $\hat{x} \circ \langle e \rangle$ swrt $\hat{B}$ // $\hat{x}$ swrt $\hat{B}$, inputs same for B and $\hat{B}$
  - $\hat{A}$ accepts e at $\hat{x}$.end // $\hat{A}$-impl-$\hat{B}$
- let u be an output transition enabled at x.end
  - $\hat{u}$ enabled at $\hat{x}$.end // $\star$
  - $\hat{x} \circ \langle \hat{u} \rangle$ swrt $\hat{B}$ // $\hat{x}$ swrt $\hat{B}$, $\hat{A}$-impl-$\hat{B}$
  - $x \circ \langle u \rangle$ swrt B // $\hat{x} \circ \langle \hat{u} \rangle$ swrt $\hat{B}$
- let u be an internal transition enabled at x.end
  - like output transition w/o output e

if $\hat{A}$-impl-$\hat{B}$ holds then A-impl-B progress holds

- let x be swrt B
- $\hat{x}$ swrt $\hat{B}$ // lemma 7.6
- let x satisfy A.progress
- $\hat{x}$ satisfies $\hat{A}$.progress // A and $\hat{A}$ have same progress
- $\hat{x}$ cwrt $\hat{B}$ // $\hat{x}$ satisfies $\hat{A}$.progress, $\hat{A}$-impl-$\hat{B}$
- x cwrt B // B and $\hat{B}$ have same progress

Theorem 7.4: if A implements B then $\exists$ Ψ st $\hat{A}$ implements $\hat{B}$

- Let Ψ maintain an instance $\hat{b}$ of $\hat{B}$,
  kept synchronized to $\hat{A}$'s ioseq thus far
  - initially, when $\hat{A}$ instantiated, instantiate $\hat{b}$
  - when $\hat{A}$ rcvs input $\hat{e}$, execute corresponding input part of $\hat{b}$
  - when $\hat{A}$ does an output whose "unhatted" part is e
    set $\hat{e}$ to any output that $\hat{b}$ can output
    - at least one such exists // A-impl-B, ioseq thus far is swrt B

- Such an externalizer ensures that $\hat{A}$-impl-$\hat{B}$ holds