Take notes

TECHNOLOGY

To Remember a Lecture Better, Take Notes by Hand

Students do worse on quizzes when they use keyboards in class.

Renato Ganoza

I will make notes / slides available *after* the lectures

I will ask you to work during the lectures

Event-B: introduction and first steps[1]

Manuel Carro
manuel.carro@upm.es

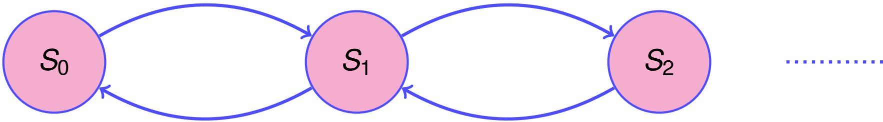
[1] Several slides borrowed from J. R. Abrial

Basic Ideas

- Model: formal description of a discrete system.
  - Formal: tools to decide whether some properties hold
  - Discrete: can be represented as a transition system
- Similar to blueprints in other disciplines
  - We will use techniques similar to other disciplines.
  - But we'll also differ in some key issues.
    - E.g., discrete vs. dense vs. continuous domains.

The State of a Model

- A discrete model is first made of states



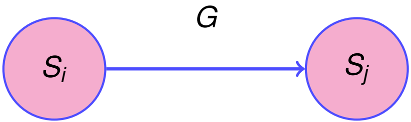
- States are represented by some constants and variables

$$S_i = \langle c_1, \ldots, c_n, v_1, \ldots, v_m \rangle$$

- Relationships among constants and variables written using set-theoretic expressions

Transitions between States

- A discrete model is also made of a number of events
- An event is made of a guard and an action
  - The guard denotes the enabling condition of the event
  - The action denotes the way the state is modified by the event



- Guards and actions are written using set-theoretic expressions

Events

```
Event EventName
  when
    guard:  G(v, c)
  then
    action:  v := E(v, c)
  end
```

```
Event Search
  when
    f(i) != k and i < n
  then
    i := i + 1
  end
```

```
Event Found
  when
    f(i) = k
  then
    skip
  end
```

```
Initialize;
while (some events have true guards) {
  Choose one such event;
  Modify the state accordingly;
}
```

Operational Interpretation

- An event execution is supposed to take no time

- Thus, no two events can occur simultaneously

- When all events have false guards, the discrete system stops

- When some events have true guards, one of them is chosen non-deterministically and its action modifies the state

- The previous phase is repeated (if possible)

- Stopping is not necessary: a discrete system may run for ever

- This interpretation is just given here for informal understanding

- The meaning of such a discrete system will be given by the proofs which can be performed on it (next lectures)

- Formalization contains models of:
  - the future software components
  - the future equipments surrounding these components

- The overall model construction can be very complex

- Three techniques can be used to master this complexity
  - refinement
  - decomposition
  - generic instantiation

- Refinement allows us to build model gradually

- We shall build an ordered sequence of more precise models

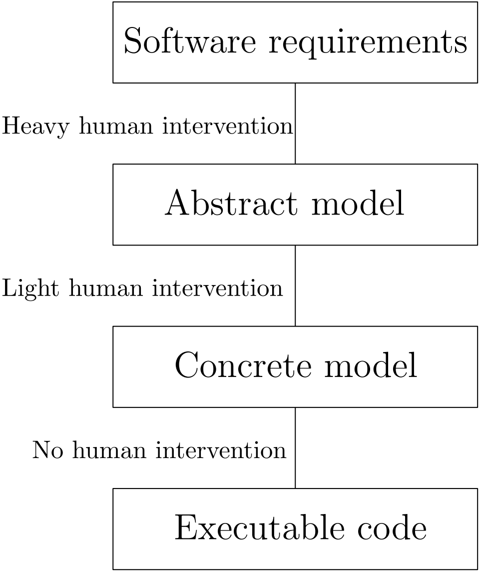
- Each model is a refinement of the one preceding it

- A useful analogy: looking through a microscope

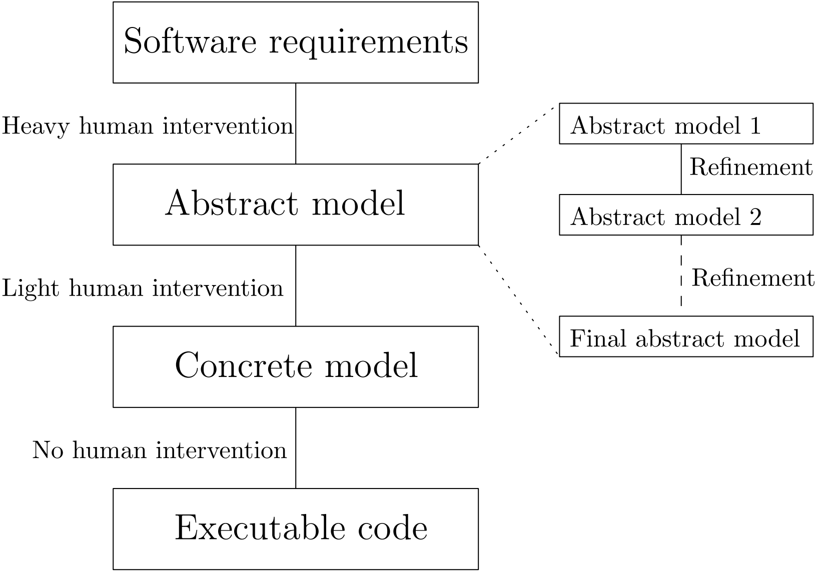
- Spatial as well as temporal extensions

- Data refinement

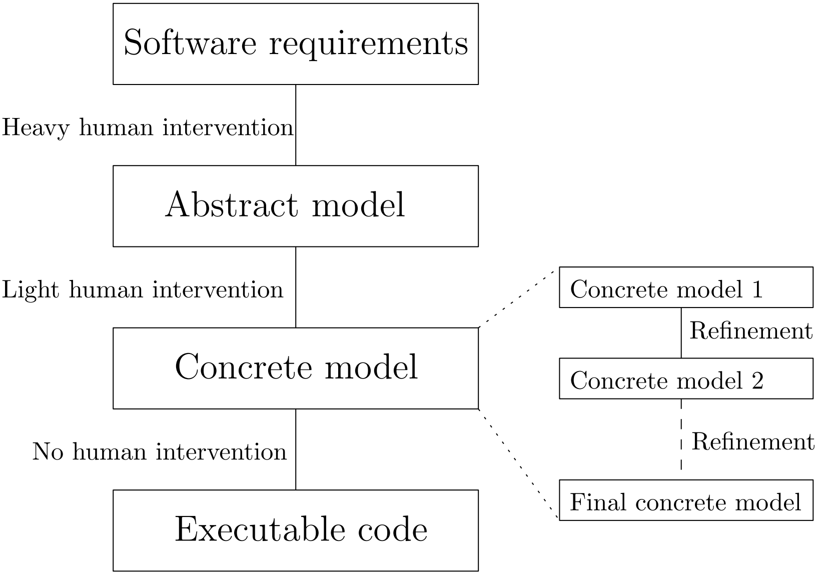
Refinement Phases



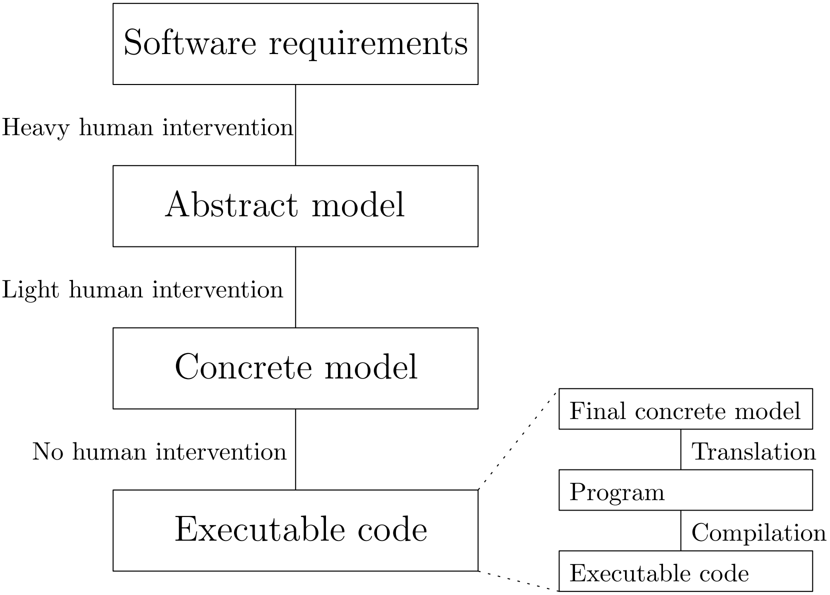
Refinement Phases



Refinement Phases



Refinement Phases



- Three phases:
  - Constructing the abstract model
  - Constructing the concrete model
  - Constructing the executable code

- Two main concepts:
  - Refinement
  - Proof

Running example (sequential code)

- Characterize it

$$a = \left\lfloor \frac{b}{c} \right\rfloor$$

  - we want to define integer division — without division

Q: division spec. (1)

- Input and output?
- Variables and constants?
- Types?

Zero

There is no universal agreement about whether to include zero in the set of natural numbers. Some authors begin the natural numbers with 0, corresponding to the non-negative integers 0, 1, 2, 3, ..., whereas others start with 1, corresponding to the positive integers 1, 2, 3, ... This distinction is of no fundamental concern for the natural numbers as such [...]

I will assume that $0 \in \mathbb{N}$. That is the convention in computer science.

Programming integer division

- We have addition and substracion
- We have a simple procedural language
- Variables, assignment, loops, if-then-else, + & -, arith. operators, …

Q: integer division code (2)

Towards Events

```
Event EventName
  when
    G(v, c)
  then
    v := E(v, c)
  end
```

- $v := E(v, c) \equiv Act_E(v, c, w)$ (*before-after* predicate) where w is renamed to v after the predicate.
- Special initialization event (**INIT**)
- Sequential program (special case):
  - *Final* event, *Progress* events
  - Guards exclude each other
  - Some guard is always true

Q: integer division events (3)

Categorizing elements

Constants Q: constants (4)	Axioms Q: axioms (5)
Variables Q: variables (6)	Invariants Later!

Invariants

- Invariant: formula true before and after event
- State *safety* conditions, prove correctness
  - What must always be true in a physical system
  - Wnat must always be true in an algorithm (ensure that code doesn't go bananas)
  - Necessary to prove (sequential) correctness
  - In non-terminating, reactive systems: capture conditions which must hold always (safety)
- Finding invariants: mixes art and science
- Hint: explore what happens with the variables as the code proceeds

Invariants

- Constants and **variables**
- Proving invariant preservation: For all event i, invariant j

$$A(c), G_i(v,c), I_j(v,c) \vdash I_j(E_i(v,c),c)$$

  - $A(c)$ axioms
  - $G_i(v,c)$ guard of event i
  - $I_j(v,c)$ invariant j
  - $E_i(v,c)$ result of action i

Invariant preservation

If we start with an invariant true and the guards of an event are true and we execute the event's action, the invariant still holds.

- **INIT** case

invariant preservation for **INIT** (7)

Finding invariants

- Which expressions are invariant in our model?

Q: model invariants (8)

- One formula which is an invariant for **any** Event-B model / loop.

Q: eternal invariant (9)

Invariant preservation proofs

- Three invariants & three events: nine proofs
- Named as e.g. $E_{Progress}/I_2/INV$
    - Other types of proofs will be necessary in due time

E_{INIT} / I_1 / INV

INIT I1 inv. proof (10)

E_{INIT} / I_2 / INV

INIT I2 inv. proof (11)

Invariant preservation proofs

E_{INIT} / I_3 / INV

INIT I3 inv. proof (12)

Interlude with sequents

- Mechanize proofs?
  - Humans "understand"; proving is tiresome and error-prone
  - Computers manipulate symbols
- How can we mechanically construct correct proofs?
  - Every step crystal clear
  - For a computer to perform
- Several approaches
- For Event B: sequent calculus
  - Reading: [Pau] (available through course web page), at least Sect. 3.3 and 6.4, 6.5. Note: when we use $\Gamma \vdash \Delta$, Paulson uses $\Gamma \Rightarrow \Delta$
- ... switching to J.R. Abrial slides for a moment

- An inference rule is a tool to perform a formal proof
- It is denoted by:

$$\frac{\mathbf{A}}{\mathbf{C}}\ \mathbf{R}$$

- **A** is a (possibly empty) collection of sequents: the antecedents
- **C** is a sequent: the consequent
- **R** is the name of the rule

The proofs of each sequent of **A**

——— together give you ———

a proof of sequent **C**

- We are given:
  - a collection $\mathcal{T}$ of inference rules of the form $\frac{A}{C}$
  - a sequent container K, containining S initially

WHILE K is not empty

CHOOSE a rule $\frac{A}{C}$ in $\mathcal{T}$ whose consequent C is in K;

REPLACE C in K by the antecedents A (if any)

This proof method is said to be goal oriented

$$\frac{}{S2}\textbf{r1} \qquad \frac{S7}{S4}\textbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\textbf{r3} \qquad \frac{}{S5}\textbf{r4} \qquad \frac{S5 \quad S6}{S3}\textbf{r5} \qquad \frac{}{S6}\textbf{r6} \qquad \frac{}{S7}\textbf{r7}$$

$S1$

?

Proof of Sequent $S1$

$$\frac{}{S2}\mathbf{r1} \qquad \frac{S7}{S4}\mathbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\mathbf{r3} \qquad \frac{}{S5}\mathbf{r4} \qquad \frac{S5 \quad S6}{S3}\mathbf{r5} \qquad \frac{}{S6}\mathbf{r6} \qquad \frac{}{S7}\mathbf{r7}$$

$$\begin{array}{ccc} & S1 & \\ & \mathbf{r3} & \\ \nearrow & \uparrow & \nwarrow \\ S2 & S3 & S4 \\ \mathbf{?} & \mathbf{?} & \mathbf{?} \end{array}$$

Proof of Sequent $S1$

$$\frac{}{S2}\,\textbf{r1} \qquad \frac{S7}{S4}\,\textbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\,\textbf{r3} \qquad \frac{}{S5}\,\textbf{r4} \qquad \frac{S5 \quad S6}{S3}\,\textbf{r5} \qquad \frac{}{S6}\,\textbf{r6} \qquad \frac{}{S7}\,\textbf{r7}$$

$$\begin{tikzcd}
 & S1 & \\
 & \textbf{r3} & \\
S2 \arrow[ur] & S3 \arrow[u] & S4 \arrow[ul] \\
\textbf{r1} & \textbf{?} & \textbf{?}
\end{tikzcd}$$

Proof of Sequent $S1$

$$\frac{}{S2}\,\mathbf{r1} \qquad \frac{S7}{S4}\,\mathbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\,\mathbf{r3} \qquad \frac{}{S5}\,\mathbf{r4} \qquad \frac{S5 \quad S6}{S3}\,\mathbf{r5} \qquad \frac{}{S6}\,\mathbf{r6} \qquad \frac{}{S7}\,\mathbf{r7}$$

$$\begin{tikzcd}
 & \begin{array}{c} S1 \\ \mathbf{r3} \end{array} & \\
\begin{array}{c} S2 \\ \mathbf{r1} \end{array} \arrow[ur] & \begin{array}{c} S3 \\ \mathbf{r5} \end{array} \arrow[u] & \begin{array}{c} S4 \\ \mathbf{?} \end{array} \arrow[ul] \\
\begin{array}{c} S5 \\ \mathbf{?} \end{array} \arrow[ur] & \begin{array}{c} S6 \\ \mathbf{?} \end{array} \arrow[u] &
\end{tikzcd}$$

$$\frac{}{S2}\,\mathbf{r1} \qquad \frac{S7}{S4}\,\mathbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\,\mathbf{r3} \qquad \frac{}{S5}\,\mathbf{r4} \qquad \frac{S5 \quad S6}{S3}\,\mathbf{r5} \qquad \frac{}{S6}\,\mathbf{r6} \qquad \frac{}{S7}\,\mathbf{r7}$$

$$\begin{tikzcd}
 & \begin{array}{c} S1 \\ \mathbf{r3} \end{array} & \\
\begin{array}{c} S2 \\ \mathbf{r1} \end{array} \arrow[ur] & \begin{array}{c} S3 \\ \mathbf{r5} \end{array} \arrow[u] & \begin{array}{c} S4 \\ \mathbf{?} \end{array} \arrow[ul] \\
\begin{array}{c} S5 \\ \mathbf{r4} \end{array} \arrow[ur] & \begin{array}{c} S6 \\ \mathbf{?} \end{array} \arrow[u] &
\end{tikzcd}$$

$$\frac{}{S2}\,\textbf{r1} \qquad \frac{S7}{S4}\,\textbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\,\textbf{r3} \qquad \frac{}{S5}\,\textbf{r4} \qquad \frac{S5 \quad S6}{S3}\,\textbf{r5} \qquad \frac{}{S6}\,\textbf{r6} \qquad \frac{}{S7}\,\textbf{r7}$$

$$\begin{tikzcd}
 & \begin{array}{c} S1 \\ \textbf{r3} \end{array} & \\
\begin{array}{c} S2 \\ \textbf{r1} \end{array} \arrow[ur] & \begin{array}{c} S3 \\ \textbf{r5} \end{array} \arrow[u] & \begin{array}{c} S4 \\ \textbf{?} \end{array} \arrow[ul] \\
\begin{array}{c} S5 \\ \textbf{r4} \end{array} \arrow[ur] & \begin{array}{c} S6 \\ \textbf{r6} \end{array} \arrow[u] &
\end{tikzcd}$$

Proof of Sequent $S1$

$$\frac{}{S2}\,\textbf{r1} \qquad \frac{S7}{S4}\,\textbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\,\textbf{r3} \qquad \frac{}{S5}\,\textbf{r4} \qquad \frac{S5 \quad S6}{S3}\,\textbf{r5} \qquad \frac{}{S6}\,\textbf{r6} \qquad \frac{}{S7}\,\textbf{r7}$$

$$\begin{tikzcd}
 & \substack{S1 \\ \mathbf{r3}} & \\
\substack{S2 \\ \mathbf{r1}} \arrow[ur] & \substack{S3 \\ \mathbf{r5}} \arrow[u] & \substack{S4 \\ \mathbf{r2}} \arrow[ul] \\
\substack{S5 \\ \mathbf{r4}} \arrow[ur] & \substack{S6 \\ \mathbf{r6}} \arrow[u] & \substack{S7 \\ \mathbf{?}} \arrow[u]
\end{tikzcd}$$

$$\frac{}{S2}\,\mathbf{r1} \qquad \frac{S7}{S4}\,\mathbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\,\mathbf{r3} \qquad \frac{}{S5}\,\mathbf{r4} \qquad \frac{S5 \quad S6}{S3}\,\mathbf{r5} \qquad \frac{}{S6}\,\mathbf{r6} \qquad \frac{}{S7}\,\mathbf{r7}$$

$$\begin{tikzcd}
 & \begin{matrix} S1 \\ \mathbf{r3} \end{matrix} & \\
\begin{matrix} S2 \\ \mathbf{r1} \end{matrix} \arrow[ur] & \begin{matrix} S3 \\ \mathbf{r5} \end{matrix} \arrow[u] & \begin{matrix} S4 \\ \mathbf{r2} \end{matrix} \arrow[ul] \\
\begin{matrix} S5 \\ \mathbf{r4} \end{matrix} \arrow[ur] & \begin{matrix} S6 \\ \mathbf{r6} \end{matrix} \arrow[u] & \begin{matrix} S7 \\ \mathbf{r7} \end{matrix} \arrow[u]
\end{tikzcd}$$

$$\frac{}{S2}\,\mathbf{r1} \qquad \frac{S7}{S4}\,\mathbf{r2} \qquad \frac{S2 \quad S3 \quad S4}{S1}\,\mathbf{r3} \qquad \frac{}{S5}\,\mathbf{r4} \qquad \frac{S5 \quad S6}{S3}\,\mathbf{r5} \qquad \frac{}{S6}\,\mathbf{r6} \qquad \frac{}{S7}\,\mathbf{r7}$$

$$\begin{tikzcd}
 & \begin{array}{c} S1 \\ \mathbf{r3} \end{array} & \\
\begin{array}{c} S2 \\ \mathbf{r1} \end{array} \arrow[ur] & \begin{array}{c} S3 \\ \mathbf{r5} \end{array} \arrow[u] & \begin{array}{c} S4 \\ \mathbf{r2} \end{array} \arrow[ul] \\
\begin{array}{c} S5 \\ \mathbf{r4} \end{array} \arrow[ur] & \begin{array}{c} S6 \\ \mathbf{r6} \end{array} \arrow[u] & \begin{array}{c} S7 \\ \mathbf{r7} \end{array} \arrow[u]
\end{tikzcd}$$

- The proof is a tree

- We supposedly have a PREDICATE Language (NOT DEFINED YET)

- A sequent is denoted by the following construct: $\boxed{\mathbf{H} \;\vdash\; \mathbf{G}}$

- **H** is a (possibly empty) collection of predicates: the hypotheses

- **G** is a predicate: the goal

Under the hypotheses of collection **H**, prove the goal **G**

- There are three basic inference rules

- These rules are independent of our future Predicate Language

- **HYP**: If the goal belongs to the hypotheses of a sequent,

  then the sequent is proved,

$$\frac{}{\mathbf{H},\ \mathbf{P}\ \vdash\ \mathbf{P}}\quad \mathbf{HYP}$$

- **MON**: Once a sequent is proved, any sequent with the same goal and more hypotheses is also proved,

$$\frac{\mathbf{H} \vdash \mathbf{Q}}{\mathbf{H},\ \mathbf{P} \vdash \mathbf{Q}}\ \mathbf{MON}$$

- **CUT**: If you succeed in proving **P** under **H**, then **P** can be added to the collection **H** for proving a goal **Q**.

$$\frac{\mathbf{H} \vdash \mathbf{P} \qquad \mathbf{H},\ \mathbf{P} \vdash \mathbf{Q}}{\mathbf{H} \vdash \mathbf{Q}}\ \mathbf{CUT}$$

More Rules

- There are many other rules for:
  - Logic itself
    - Look at the slides / documents in the course web page
  - reasoning on arithmetic (Peano axioms),
  - reasoning on sets,
  - reasoning on functions,
  - ...
- We will not list all of them here (see online documentation)
- We will explain them as they appear
- But a mechanical prover has them as “inside knowledge” (plus tactics, strategies)

Previous (unexplained) rules

First Peano axiom

$$\frac{}{\vdash 0 \in \mathbb{N}}\ \mathrm{P0}$$

Previous (unexplained) rules

First Peano axiom

$$\frac{}{\vdash 0 \in \mathbb{N}}\ \text{P0}$$

Term substitution

$$\frac{Q(E), E = F \vdash R(E)}{Q(E), E = F \vdash R(F)}\ \text{EQ-LR}$$

Previous (unexplained) rules

First Peano axiom

$$\frac{}{\vdash 0 \in \mathbb{N}}\ \text{P0}$$

Term substitution

$$\frac{Q(E), E = F \vdash R(E)}{Q(E), E = F \vdash R(F)}\ \text{EQ-LR}$$

Equality

$$\frac{}{\vdash E = E}\ \text{EQL}$$

Invariant preservation proofs

$\mathrm{E}_{\mathrm{Progress}}$ / I_1 / INV

Progress I1 inv. proof (13)

First Peano axiom

$$\frac{}{n \in \mathbb{N} \vdash n + 1 \in \mathbb{N}}\ \mathrm{P1}$$

Invariant preservation proofs

E_{Progress} / I_2 / INV

Progress I2 inv. proof (14)

More rules

$$\frac{H \vdash Q \qquad H \vdash P}{H \vdash P \wedge Q}\ \text{AND-L}$$

$$\frac{H, Q \vdash R \qquad H, P \vdash R}{H, P \vee Q \vdash R}\ \text{OR-L}$$

$$\frac{H \vdash P}{H \vdash P \vee Q}\ \text{OR-R1} \qquad \frac{H \vdash Q}{H \vdash P \vee Q}\ \text{OR-R2}$$

NB: The two last ones are shorthands for the chaining of

$$\frac{H \vdash P, Q}{H \vdash P \vee Q}\ \text{OR} \qquad \frac{H \vdash P}{H \vdash P, Q}\ \text{W-R}$$

Invariant preservation proofs

$\mathsf{E}_{\mathsf{Progress}}$ / I_3 / INV

Progress I3 inv. proof (15)

Invariant preservation proofs

Proofs for `Finish`

- $E_{Finish}/I_1/INV$
- $E_{Finish}/I_2/INV$
- $E_{Finish}/I_3/INV$

are trivial (`Finish` does not change anything)

Sequential correctness

- Postcondition P must be true at the end of execution
- End of execution associated to special event `Finish`:

$$A(c), G_{\mathsf{Finish}}(v, c), I_{1..n}(v, c) \vdash P(v, c)$$

Q: corr. cond. for example (16)

- Not applicable to non-terminating systems (other proofs required)
- $I_{1..n}$ and G_{Finish} related to P; not necessarily identical
- Correctnes condition if termination = model stopping?

Q: alternative corr. cond. (17)

Sequential correctness: invariant strength

- $I_{1\ldots n}$ together with A and G_{Finish} should imply P
- A correct $I_{1\ldots n}$ may not be strong enough P
- If $A \vdash B$ or $A \Rightarrow B$, then A is stronger than B.
- What are the strongest and weakest possible formulæ?

Q: strogest / weakest flæ. (18)

- Information amount
- Role as invariant
- Balance between extremes
  - Too weak: easy as invariant, maybe not enough information
  - Too strong: maybe not an invariant

Termination

- "*Postcondition P must be true* ***at the end*** *of execution*"
- General strategy: look for a *ranking function* / *progress measure*
- In Event B lingo: a *variant* $V(v, c)$
  - An expression V (with $V \in \mathbb{N}$ or $V \subseteq S$) that is reduced by each *non-terminating* event

$$A(v), I_{1\dots n}, G_i(v, c) \vdash V(v, c) > V(E_i(v, c), c)$$

Q: variant expression (19)

  - We do not say how it is reduced: has to be proven

Term. proof (20)

Event B, Homework #1, by Wed. Sep. 13th, 7pm

1. Which proof(s) would have failed, and where, had we:
   1.1 Added the invariant $k > 0$
   1.2 Replaced the invariant $a \times c + k = b$ for $a \times c - k = b$
   1.3 **Not** included $c > 0$ in the axioms

   Note: every item above is a separate question.

2. Given $n \in \mathbb{N}$, calculate $r = n^2$ as $r = \overbrace{1 + 3 + \cdots + (2n+1)}^{n}$ with

```
Event INIT              Event Finish       Event Progress
  i,r,a := n,0,1          when i = 0         when i > 0
                          then skip          then r,a,i := r+a,a+2,i-1
                        end                end
```

   Identify constants and variables. Determine axioms and invariants, and termination and correctness conditions. Prove invariant preservation for `Progress` event, but not for the type invariants. Prove termination and correctness.

Remarks on homework

- To be corrected correct in the next lecture
- Hand them in by Wed. Sep. 14, 19:00
- Give me handwritten solutions at the beginning of the lecture, or
- Send me a PDF file. **Please do not send me Word files.**

Michael Huth and Mark Ryan.
Logic in Computer Science: Modelling and Reasoning About Systems.
Cambridge University Press, New York, NY, USA, 2004.

Lawrence C. Paulson.
Logic and Proof.
Lecture notes, U. of Cambridge.