The problem's plain to see: too much technology.
Machines to save our lives. Machines dehumanize.
Review

Current topic:
• How to reason about correctness of code
• Started with informal arguments
• Developed formal logic

Today:
• A proof assistant called Coq
Question #1

How much of PS5 have you finished?
A. None
B. About 25%
C. About 50%
D. About 75%
E. I’m done!!!
## Review: Proof rules of IPC, part 1

<table>
<thead>
<tr>
<th>Rule name</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>/\ intro</td>
<td>if $F \vdash f_1$ and $F \vdash f_2$ then $F \vdash f_1 \land f_2$</td>
</tr>
<tr>
<td>/\ elim L</td>
<td>if $F \vdash f_1 \land f_2$ then $F \vdash f_1$</td>
</tr>
<tr>
<td>/\ elim R</td>
<td>if $F \vdash f_1 \land f_2$ then $F \vdash f_2$</td>
</tr>
<tr>
<td>=&gt; elim</td>
<td>if $F \vdash f$ and $F \vdash f \Rightarrow g$ then $F \vdash g$</td>
</tr>
<tr>
<td>=&gt; intro</td>
<td>if $F, f \vdash g$ then $F \vdash f \Rightarrow g$</td>
</tr>
<tr>
<td>assume</td>
<td>$f \vdash f$</td>
</tr>
<tr>
<td>weak</td>
<td>if $F \vdash f$ then $F, g \vdash f$</td>
</tr>
<tr>
<td>set assume</td>
<td>$F, f \vdash f$</td>
</tr>
</tbody>
</table>
### Review: Proof rules of IPC, part 2

<table>
<thead>
<tr>
<th>Rule name</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>∨ intro L</td>
<td>if $F</td>
</tr>
<tr>
<td>∨ intro R</td>
<td>if $F</td>
</tr>
<tr>
<td>∨ elim</td>
<td>if $F</td>
</tr>
<tr>
<td>true intro</td>
<td>$F</td>
</tr>
<tr>
<td>false elim</td>
<td>if $F</td>
</tr>
<tr>
<td>~ intro</td>
<td>if $F</td>
</tr>
<tr>
<td>~ elim</td>
<td>if $F</td>
</tr>
</tbody>
</table>
## Review: Proof rules of IQC

<table>
<thead>
<tr>
<th>Rule name</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
<td>All rules of IPC</td>
</tr>
<tr>
<td>forall intro</td>
<td>if $F</td>
</tr>
<tr>
<td>forall elim</td>
<td>if $F</td>
</tr>
<tr>
<td>exists intro</td>
<td>if $F</td>
</tr>
<tr>
<td>exists elim</td>
<td>if $F</td>
</tr>
</tbody>
</table>
Theories

- IQC reaches its full power when augmented with *theories*
- Collections of
  - names of relations and functions, and
  - new proof rules for those
Theory of equality

• Relation: $equals(t_1, t_2)$
  – normally written $t_1 = t_2$

• Proof rules:
  – reflexivity: $t = t$
  – symmetry: if $t_1 = t_2$ then $t_2 = t_1$
  – transitivity: if $t_1 = t_2$ and $t_2 = t_3$ then $t_1 = t_3$
  – eq-fn: if $t_1 = u_1$ and...and $t_n = u_n$ then $fn(t_1, \ldots, t_n) = fn(u_1, \ldots, u_n)$
  – eq-rel: if $t_1 = u_1$ and...and $t_n = u_n$ then $R(t_1, \ldots, t_n) = R(u_1, \ldots, u_n)$
Theory of rings

• **Ring**: mathematical structure that abstracts addition and multiplication
  – see Math 4320

• Relies on theory of equality

• Functions:
  – `plus(t1, t2)` and `mult(t1, t2)` and `neg(t)`
    • written `t1 + t2` and `t1 * t2` and `-t`
  – `zero` and `one`
    • written `0` and `1`
Theory of rings

• Proof rules (all are axioms):
  – forall a b c, \((a+b)+c = a+(b+c)\)
  – forall a b, \(a+b = b+a\)
  – forall a, \(0+a = a\)
  – forall a, \(a + (-a) = 0\)
  – forall a b c, \(a*(b+c) = (a*b)+(a*c)\)
  – forall a b c, \((b+c)*a = (b*a)+(c*a)\)
  – forall a b c, \((a*b)*c = a*(b*c)\)
  – forall a b, \(a*b = b*a\)
  – forall a, \(1*a = a\)

• Syntactic sugar:
  \(\forall a b, f\)  
  means \(\forall a, (\forall b, f)\)
Prelim 2

- One week from today
- Covers everything from Oct 2 through Nov 12 (inclusive)
  - People with Thursday recitations, note that today’s recitation is included
- Sample prelim posted on Piazza
- Review session in recitation day before prelim
- Cancel lecture on day of prelim
- You can take prelim at your choice of 5:30-7:00 pm or 7:30-9:00 pm; no need to reserve in advance
- Three rooms, will be assigned by netid next week
- Closed book
  - But you may have one page of notes
  - 8.5x11” two-sided 😊
Why formal logic?

• Humans make mistakes in writing proofs
• Humans make mistakes in checking proofs
• Formal logic:
  – Reduces proof to symbolic manipulation
  – Maybe a machine could check that manipulation
• Analogy:
  – Compiler type checks program
  – Proof checker uses proof rules we've given to check proof
Mechanized proof

• Automated theorem provers
  – You give tool a theorem
  – Tools finds a proof or a counterexample
    • Or runs out of time
  – e.g., Z3, developed at Microsoft
    • Ships with the Windows 7 device driver developer's kit
• Proof assistants
  – You give tool a theorem
  – You and tool cooperatively find proof
    • Human guides the construction
    • Machine does the low-level details
  – e.g., Coq, Isabelle/HOL, NuPRL
    • NuPRL: Prof. Constable (Cornell)
    • Coq: used to verify compiler, OS kernel, etc.
Coq

- **1984**: Coquand and Huet first begin implementing a new theorem prover Coq based on *calculus of inductive constructions*
- **1992**: Coq ported to Caml
- **2012**: Coq version 8.4
  - Implemented in OCaml
  - Can produce verified OCaml code
Coq's full system
Subset of Coq we'll use
Coq3110.v

• We went through the file up through and including implication and forall.
Please hold still for 1 more minute.

WRAP-UP FOR TODAY
Upcoming events

• PS5 due tonight
• Prelim 2 in one week

This is mechanized.

THIS IS 3110