Theorem Proving in Isabelle

Lutz Schröder

February 2, 2005

Simplification

Simplification

- Simplification is directed application of equations:
  - replace left-hand side by right-hand side ('rewrite')
  - right-hand side should in some sense be 'simpler'
    $\longrightarrow$ termination
- 'Equations' include equivalences (boolean equality)
- Any subterm of a formula can be rewritten

An Example

Equations $x + 0 = x$, $x * 0 = 0$

$$(((a - b) + 0) + (c * 0)) + 0 \leadsto$$
$$((a - b) + (c * 0)) + 0 \leadsto$$
$$((a - b) + 0) + 0 \leadsto$$
$$(a - b) + 0 \leadsto$$
$$a - b$$

This doesn't always work well!

Equation $x + y = y + x$ is somewhat dangerous:

$$a + b \leadsto$$

$$b + a \leadsto$$

$$a + b \leadsto \ldots$$

(Isabelle's simplifier notices this.)
Look out for non-obvious notions of 'simplicity':
$x + (y + z) = (x + y) + z$ is OK!

Using Isabelle's simplifier

- simpsets are sets of equations for simplification
- Various tactics for simplification:
  - `simp` simplifies conclusion and assumptions, using also the assumptions
  - `simp (no_asm_simp)` uses, but does not simplify assumptions
  - `simp (no_asm_use)` simplifies, but does not use assumptions
  - `simp (no_asm)` ignores assumptions entirely

Examples

Manipulating simpsets

- Locally (in `apply (simp)`):
  - Add theorem (need not be equations): `add: <thm>`
  - Delete theorems: `del: <thm>`
  - Use only a given list of theorems: `only: <thm>`
  - Special types of theorems, e.g. case splits: `split: <thm>`, `split del: <thm>` etc.
- Globally: use attributes `[simp]`, `[split]` etc.
  - in lemma/theorem statements
  - later: `declare <thm> [simp]` etc.

Another example

Constant Definitions

- Syntax: `constdefs f::  <type>`
  `"f x y = ..."`
- Defining equation is not automatically in the simpset (encapsulation!)
- If needed, add `f_def`
- Expand `let`-expressions using `Let_def`

Example

Conditional Simplification

- Isabelle handles also conditional equations $\phi \Longrightarrow t = s$ in simplification
- tries to find ϕ among local assumptions by simplification!

Example

Case splits

- Manually: `case_tac`
- Similarly: `split`, provided with a splitting theorem `split_if` or `t.split`, `t` datatype
- Automatically: add splitting theorems to the simplifier using modifier `split`
- Variants for splitting assumptions: `split_if_asm`, `t.split_asm`

Example

Arithmetic

- Extremely simple arithmetic is handled by simplification:
  - handles $+$, $=$, $<$, $\leq$
  - no object-logical connectives
- If that fails, use `arith`:
  - Handles also $-$, min, max
  - Connectives $\neg$, $\wedge$, $\vee$, $\Rightarrow$
  - No $*$ etc. (except as 'atomic' numbers)

A last example