Beyond Script Management

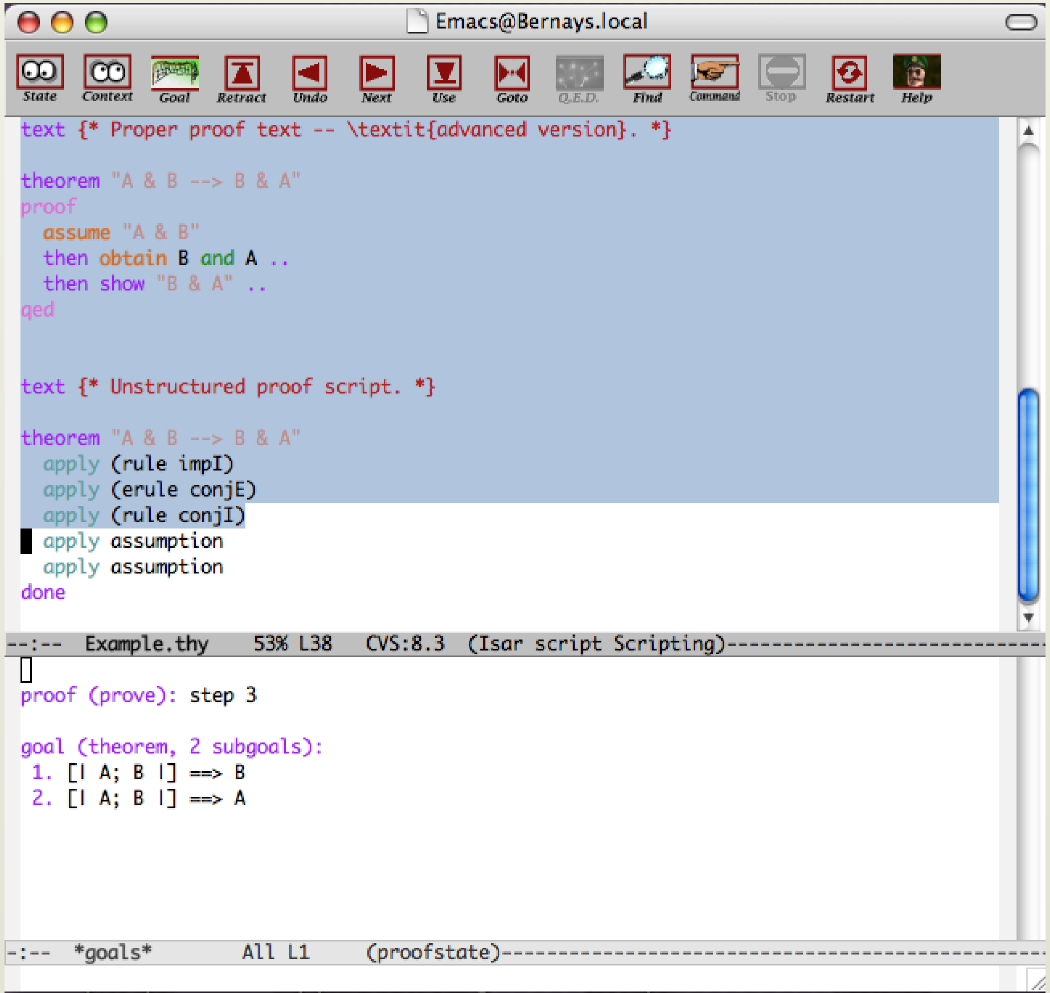
David Aspinall
School of Informatics
University of Edinburgh

Christoph Lüth
DFKI Bremen

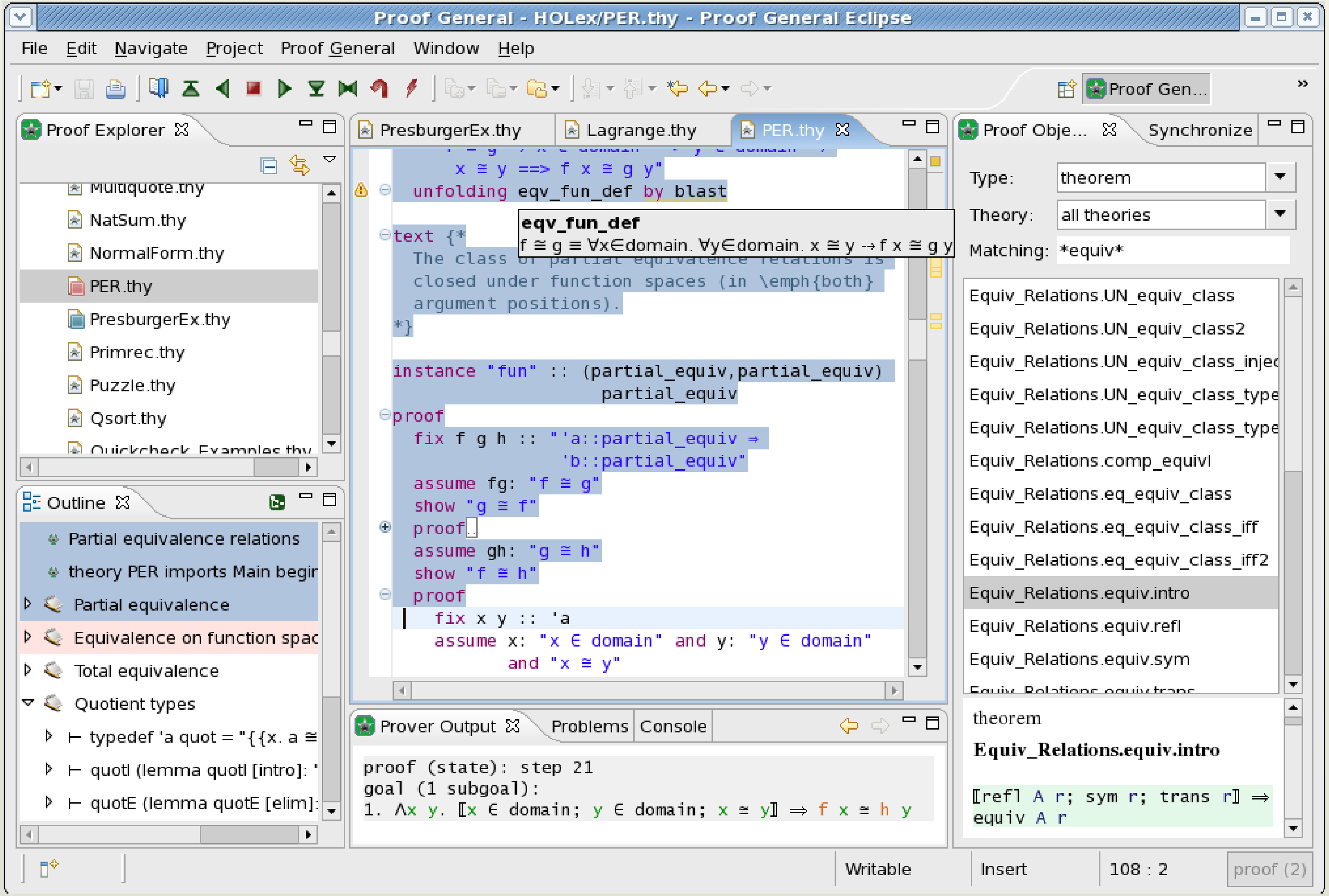
CIAO 2007 · 04.04.2007

Script Management: The Story So Far

Script Managemenent: Recap



Script Management: Recap



Script Management: History

- Beginnings: unclear
- Bertot et al: CtCoq
- Aspinall et al: Proof General
  - First with a toolbar
- Proven technology

Script Management: History

- Beginnings: unclear
- Bertot et al: CtCoq
- Aspinall et al: Proof General
  - First with a toolbar
- Proven technology
- Easy to understand & use, close to prover
- . . . but is it really best we can do?

Script Management: Disadvantages

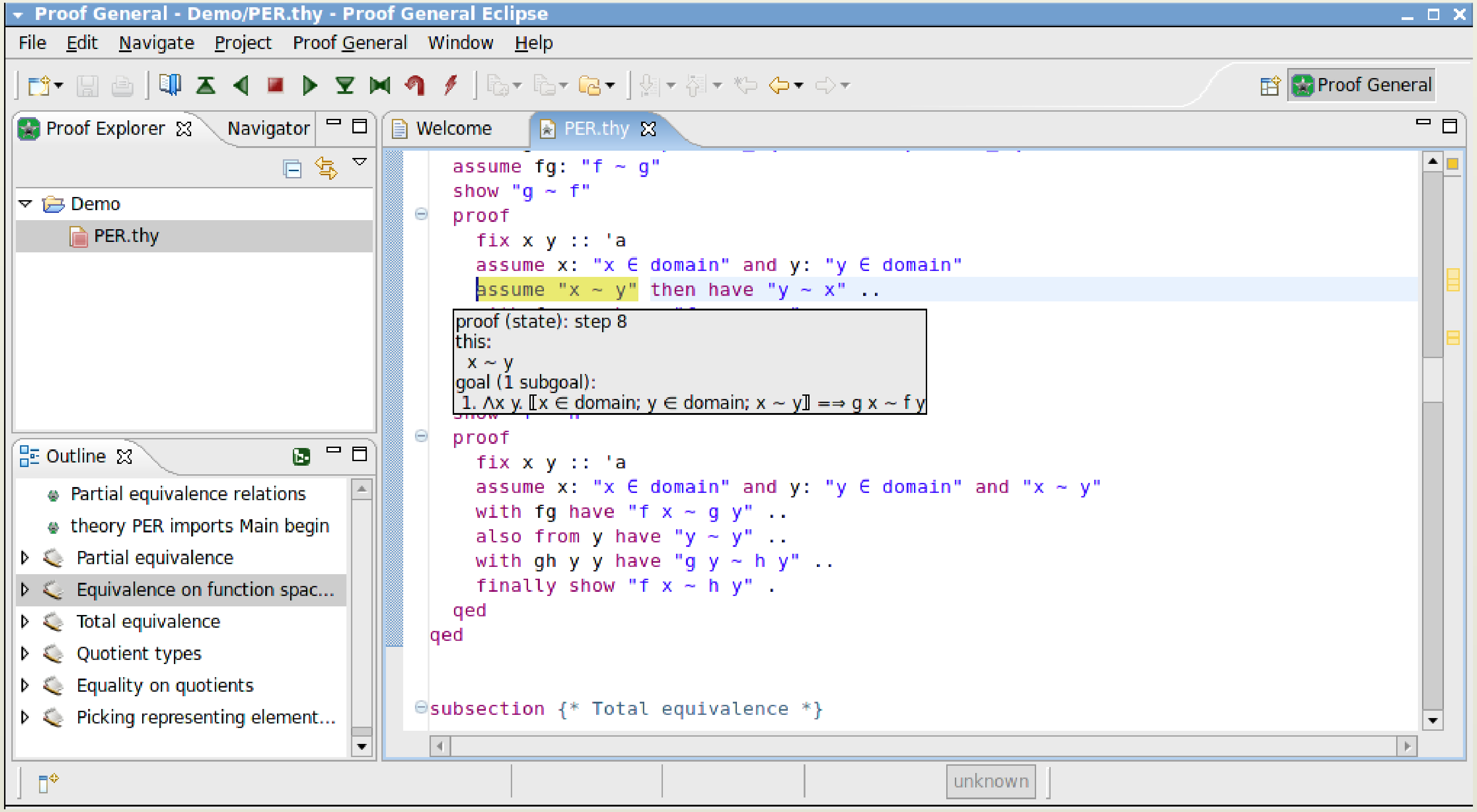
- Highly linear
  - Forced to work one error at a time
  - Interface directs user's attention
- Confusing undo
  - Theorem history discarded upon completion
  - Nested proofs?
  - End of theories?
- Too restrictive
  - Background colouring is invasive
  - Locking is frustrating
  - Explorative proof development restricted

Beyond Script Management: Four Ideas

Eager Script Management, Lazy Locking

- Do not colour, instead highlight next command
- Process next command behind the scenes
- See good/bad effect by marker highlight
- Examine detail by hover or forward

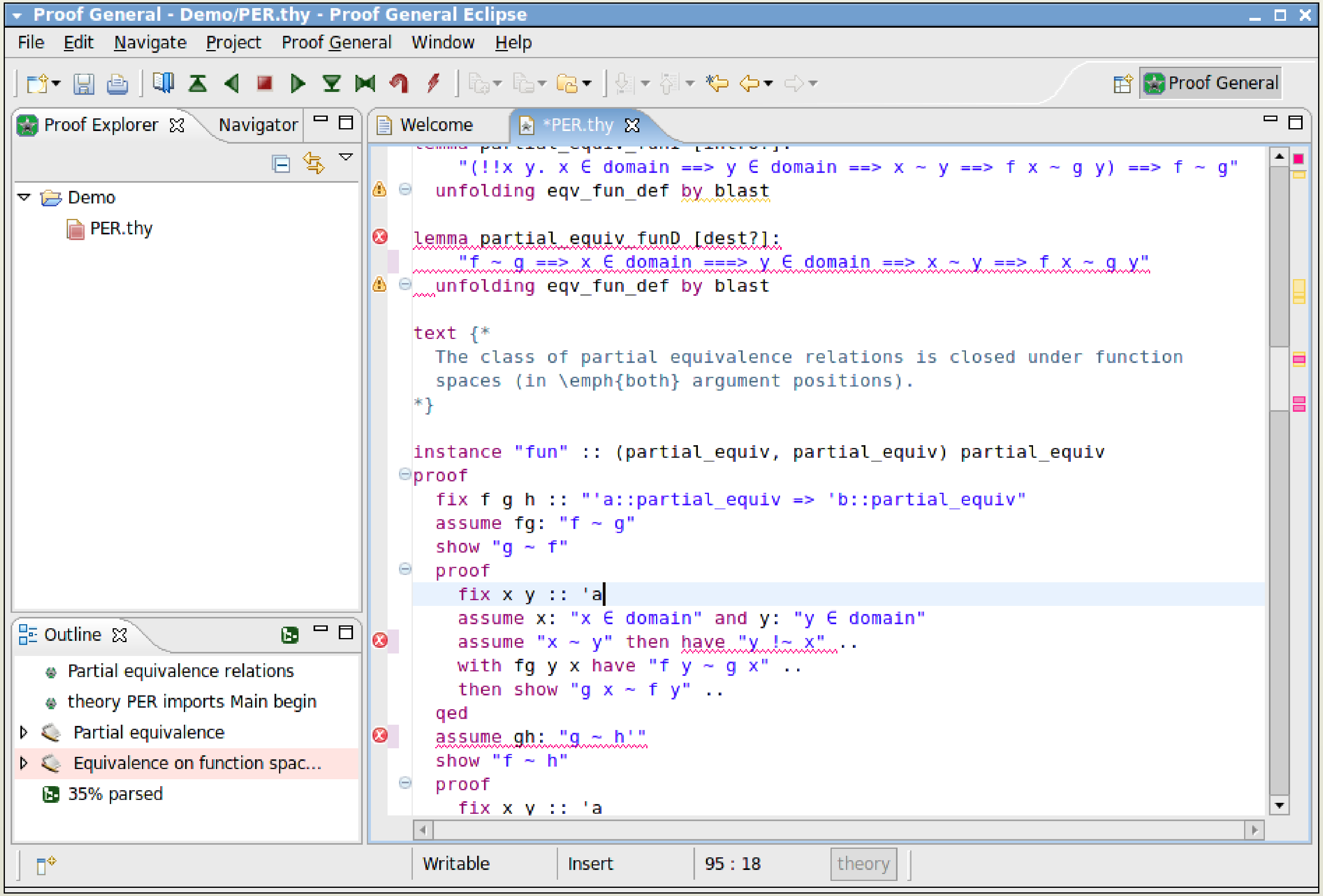
Eager Script Management, Lazy Locking



Whole-Document Editing

- Do not stop at first error
- Use `<postponegoal>` and proceed
  - Prover should record proof obligation
- How to deal with dependencies?
  - Maybe not at all: reprove when necessary

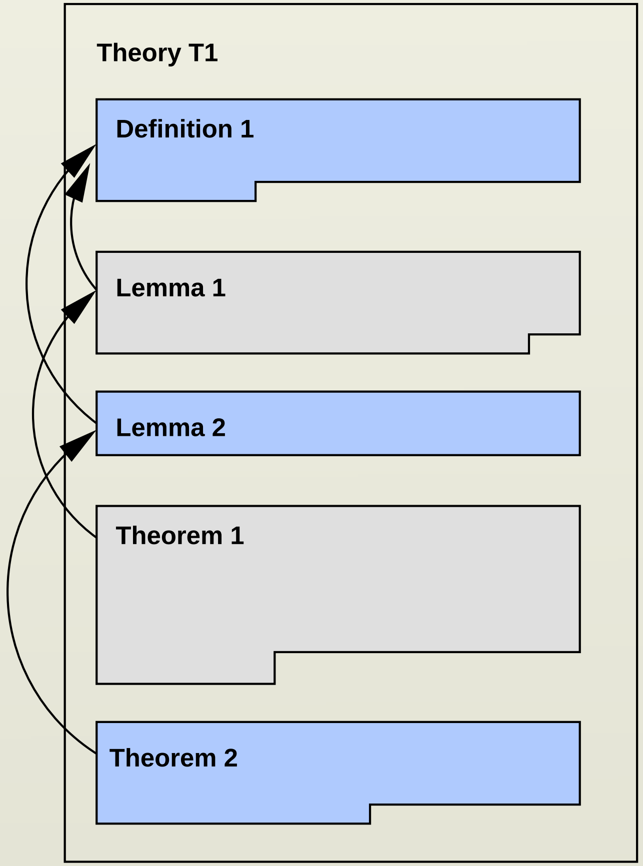
Whole-Document Editing



Non-linear dependencies

- #1 complaint about script management: it's too linear
  - Must step through proof from top to bottom
  - To work on later stages, must process early
  - To change early, must undo later
- Plan:
  - Use approximate dependency reporting from theorem prover
  - Augment with user-level annotations
  - Script management uses independent states for regions (already in PGIP display protocol)

Non-linear dependencies



Explicit dependencies:

- Requires accurate prover assistance
- Better change management

Alternative paths in proofs

- Allow easy exploration (and recording) of alternatives in proof scripts
- Add `<alternatives>` element with named subtrees.
- Add marker indicating processed state of sub-tree
- Indicate visually in the interface
  - notion of active choice
  - does some successful proof path exist?
  - are all proof paths correct (ideal)?
- Export operation to flatten active choices

Conclusions

Conclusions

- Four incremental and achievable extensions to usual script management
- Work with existing provers out-of-the-box

Conclusions

- Four incremental and achievable extensions to usual script management
- Work with existing provers out-of-the-box
- Alternative interface metaphors exist:
  - graphical presentation: graph, tree, box; desktop direct manipulation
  - whole document/constant validation
    - most obvious & possibly most productive (cf WYSIWYG)
    - disadvantage: short UI opn $\longrightarrow$ long-running prover opn
    - tight prover integration required