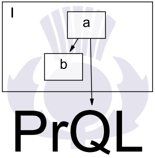
Querying Proofs

David Aspinall, LFCS, Edinburgh
Ewen Denney, NASA Ames, California
Christoph Lüth, DFKI Bremen

STP 2012

1. Motivations
2. Hiproofs
3. Queries
4. Closing



Motivations

Queries

- Mechanised proof tools produce **big proofs**
- What may we do with them?
  - check
  - reuse
  - transform
  - inspect

Queries

- Mechanised proof tools produce **big proofs**
- What may we do with them?
  - check
  - reuse
  - transform
  - inspect

- We propose:
  proof queries to inspect proofs in a uniform way

Example queries

1. Which axioms occur in the proof?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?
6. What are the goal inputs to tactic t at some point?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?
6. What are the goal inputs to tactic t at some point?
7. Show me tactic instances using this axiom?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?
6. What are the goal inputs to tactic t at some point?
7. Show me tactic instances using this axiom?
8. Show me proven goals which rely on this axiom?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?
6. What are the goal inputs to tactic t at some point?
7. Show me tactic instances using this axiom?
8. Show me proven goals which rely on this axiom?
9. Is there a sub-proof that occurs more than once?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?
6. What are the goal inputs to tactic *t* at some point?
7. Show me tactic instances using this axiom?
8. Show me proven goals which rely on this axiom?
9. Is there a sub-proof that occurs more than once?
10. Are there duplicated subproofs in the proof?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?
6. What are the goal inputs to tactic t at some point?
7. Show me tactic instances using this axiom?
8. Show me proven goals which rely on this axiom?
9. Is there a sub-proof that occurs more than once?
10. Are there duplicated subproofs in the proof?
11. Are there steps in the proof which have no effect?

Example queries

1. Which axioms occur in the proof?
2. Which witnesses are used for existentials?
3. Which tactic uses this axiom?
4. Where does this goal come from?
5. Why does this tactic not apply?
6. What are the goal inputs to tactic t at some point?
7. Show me tactic instances using this axiom?
8. Show me proven goals which rely on this axiom?
9. Is there a sub-proof that occurs more than once?
10. Are there duplicated subproofs in the proof?
11. Are there steps in the proof which have no effect?

others. . .

Hiproofs

Use Hiprofs

What is a hiproof?

Abstraction of a proof tree, hierarchy as primary.

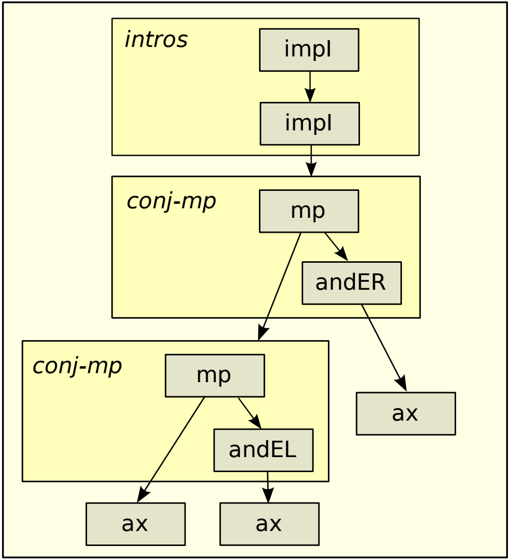
A hierarchical tree with nodes labelled with tactics.

A valid hiproof: edges (implicitly) labelled with goals.

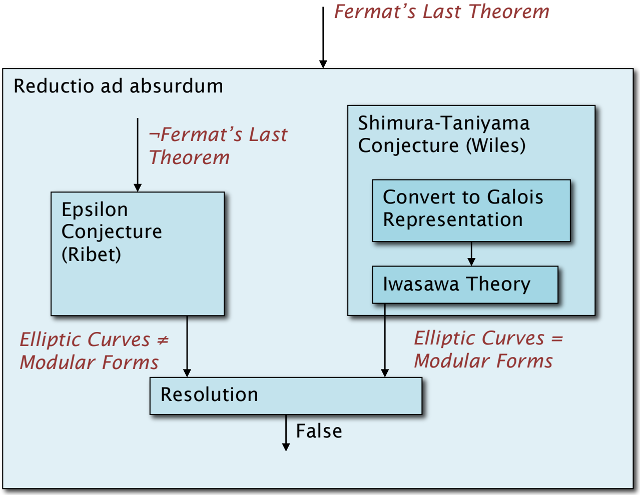
Why use hiproofs?

Abstract from underlying notions of proof, logic, and derivation. Focus on basic concepts and structure.

Example hiproof



How to understand FLT



Picture credit: Alan Bundy

Hiproofs: story so far

Commonality in systems: multiple connected trees

- Tecton: proof forests, hyperlinks (Kapur, Musser 94)
- tactic trees in NuPrl (Griffin, 98)
- hierarchical plans in Ωmega (Saarbrücken 1997)
- visual notation for λClam proofs (Bundy, Ireland)

Introduction of **Hiproofs** (Denney, Power, Tourlas 05)

- essence of hierarchy, as nested labelled trees
- abstract definitions of syntax, semantics
- equivalent denotational characterisations

Tactics and operational semantics

- LCF-style tactic language (A., Denney, Lüth 08)
- Mizar-style, refactorings (Whiteside et al 11)

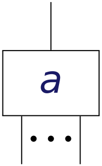
Syntax for hiproofs

s	$::=$	a	atomic
	$\mid$	id	identity
	$\mid$	$[l]\,s$	labelling
	$\mid$	$s_1 \,;\, s_2$	sequencing
	$\mid$	$s_1 \otimes s_2$	tensor
	$\mid$	$\langle\rangle$	empty

- add structure to an underlying derivation system
- map input goals γ to output subgoals $[\gamma_1, \ldots, \gamma_n]$
- have fixed arities (numbers of goals in and out)

Syntax for hiproofs

$$
\begin{array}{rcll}
s & ::= & a & \text{atomic} \\
& | & \mathrm{id} & \text{identity} \\
& | & [l]\, s & \text{labelling} \\
& | & s_1 \,;\, s_2 & \text{sequencing} \\
& | & s_1 \otimes s_2 & \text{tensor} \\
& | & \langle\rangle & \text{empty}
\end{array}
$$



$$\frac{\gamma_1 \cdots \gamma_n}{\gamma}\; a_\gamma$$

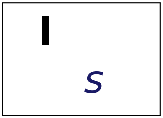
Syntax for hiproofs

s	$::=$	a	atomic
	$\mid$	id	identity
	$\mid$	$[l]\, s$	labelling
	$\mid$	$s_1 \,;\, s_2$	sequencing
	$\mid$	$s_1 \otimes s_2$	tensor
	$\mid$	$\langle\rangle$	empty

wiring

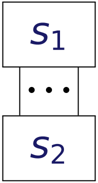
Syntax for hiproofs

s	$::=$	a	atomic
	$\mid$	id	identity
	$\mid$	$[l]\, s$	labelling
	$\mid$	$s_1 \,;\, s_2$	sequencing
	$\mid$	$s_1 \otimes s_2$	tensor
	$\mid$	$\langle\rangle$	empty



Syntax for hiproofs

s	$::=$	a	atomic
	$\mid$	id	identity
	$\mid$	$[l]\, s$	labelling
	$\mid$	$s_1 \,;\, s_2$	sequencing
	$\mid$	$s_1 \otimes s_2$	tensor
	$\mid$	$\langle\rangle$	empty



Syntax for hiproofs

$$
\begin{array}{rcll}
s & ::= & a & \text{atomic} \\
 & | & \mathrm{id} & \text{identity} \\
 & | & [l]\, s & \text{labelling} \\
 & | & s_1 \,;\, s_2 & \text{sequencing} \\
 & | & s_1 \otimes s_2 & \text{tensor} \\
 & | & \langle\rangle & \text{empty}
\end{array}
$$

$$\boxed{s_1} \qquad \boxed{s_2}$$

Syntax for hiproofs

s	$::=$	a	atomic
	$\mid$	id	identity
	$\mid$	$[l]\, s$	labelling
	$\mid$	$s_1 \,;\, s_2$	sequencing
	$\mid$	$s_1 \otimes s_2$	tensor
	$\mid$	$\langle\rangle$	empty

Syntax for hiproofs

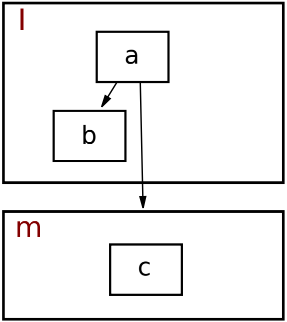
$$
\begin{array}{rcll}
s & ::= & a & \text{atomic} \\
  & \mid & \mathrm{id} & \text{identity} \\
  & \mid & [l]\, s & \text{labelling} \\
  & \mid & s_1 \,;\, s_2 & \text{sequencing} \\
  & \mid & s_1 \otimes s_2 & \text{tensor} \\
  & \mid & \langle\rangle & \text{empty}
\end{array}
$$

- valid hiproofs add structure to a skeleton, e.g.

$$
\frac{\dfrac{}{\gamma_2}\,\mathrm{b} \qquad \dfrac{}{\gamma_3}\,\mathrm{c}}{\gamma_1}\,\mathrm{a}
$$

Syntax for hiproofs

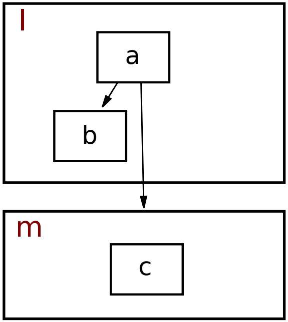
$$
\begin{array}{rcll}
s & ::= & a & \text{atomic} \\
& | & \mathrm{id} & \text{identity} \\
& | & [l]\, s & \text{labelling} \\
& | & s_1 \,;\, s_2 & \text{sequencing} \\
& | & s_1 \otimes s_2 & \text{tensor} \\
& | & \langle\rangle & \text{empty}
\end{array}
$$



$$
\dfrac{\dfrac{}{\gamma_2}\,\mathrm{b} \qquad \dfrac{}{\gamma_3}\,\mathrm{c}}{\gamma_1}\,\mathrm{a}
$$

Syntax for hiproofs

s	$::=$	a	atomic
	$\mid$	id	identity
	$\mid$	$[l]\, s$	labelling
	$\mid$	$s_1 \,;\, s_2$	sequencing
	$\mid$	$s_1 \otimes s_2$	tensor
	$\mid$	$\langle\rangle$	empty



$$([l]\, a \,;\, b \otimes \mathrm{id}) \,;\, [m]\, c$$

HiTac: a hierarchical tactic language

$$
\begin{array}{rcll}
t & ::= & a & \\
& | & \mathrm{id} & \\
& | & [l]\, t & \\
& | & t_1 \,;\, t_2 & \\
& | & t_1 \otimes t_2 & \\
& | & \langle\rangle & \\
& | & t_1 \mid t_2 & \text{alternation} \\
& | & \mu X.t & \text{repetition} \\
& | & \mathrm{assert}\, \phi & \text{testing}
\end{array}
$$

- Big-step semantics: possible final proofs
- Small-step semantics: proof state during evaluation
- Non-determinism: alternation and goal list splitting

Queries

Plan

Design a custom proof query language, PrQL

1. express desired queries succinctly
2. give a direct semantics
3. establish basic “sanity” results
4. make a toy implementation to validate design

Connect to real implementations, other QLs

1. export large-scale proofs from real TPs
   - current export mechanisms (TSTP, ProofRecording)
   - new mechanisms, e.g., views with hierarchy
2. apply existing semi-structured query languages
   - translate PrQL queries into existing language
   - possibly: translate proofs into other formats
   - establish complexity bounds

Plan

Design a custom proof query language, PrQL

1. express desired queries succinctly
2. give a direct semantics
3. establish basic “sanity” results
4. make a toy implementation to validate design

Design of PrQL

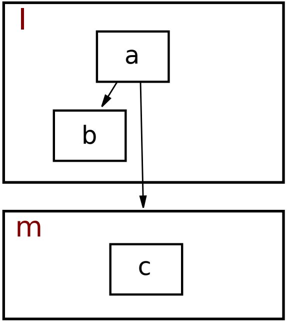
Basic ideas:

1. use schematic terms, with variables and wildcards
2. query has an implicit subject, a single proof
3. result is simply instantiation of variables
4. unimportantly: verbose keywords

Related languages:

- **ASTLOG**, query language for abstract syntax trees
  (Crew 1997)
- **UnQL**, pattern matching and recursion for XML
  (Buneman, Fernandez, Suciu 2000)
- **Graph Logic**, predicates with recursion and separation (Cardelli, Gardner, Ghelli 2002)

Local structure

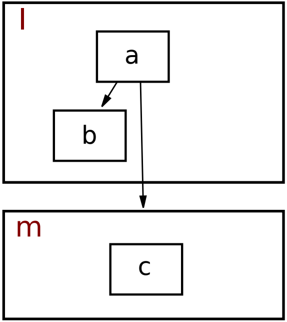


$([l]\, a \,;\, b \otimes \mathrm{id}) \,;\, [m]\, c$

Satisfies this query specifying outer structure:

$$(\textbf{inside}\; l\; *)\; \textbf{then}\; (\textbf{inside}\; m\; *)$$

Local structure

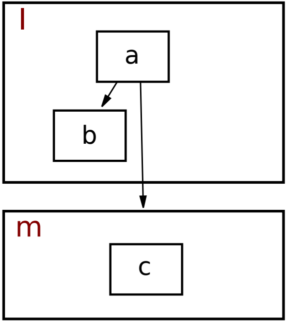


$([\mathsf{l}]\, a \,;\, b \otimes id) \,;\, [\mathsf{m}]\, c$

Satisfies this query specifying some inner structure:

(**inside** $*$ $*$ **then** $*$ **beside** **nothing**) **then** $*$

Local structure



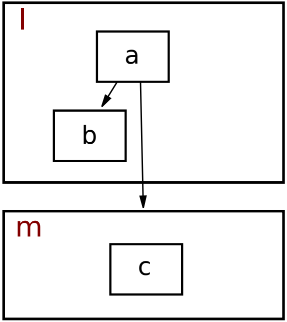
$$([\mathsf{l}]\, a \,;\, b \otimes \mathrm{id}) \,;\, [\mathsf{m}]\, c$$

Satisfies the query with bindings:

$$(\textbf{inside}\, L_1 \; *) \;\textbf{then}\; (\textbf{inside} \; * \; \textbf{atomic}\, A)$$

with instantiation $\{L_1 = \mathsf{l}, A = \mathsf{c}\}$.

Local structure

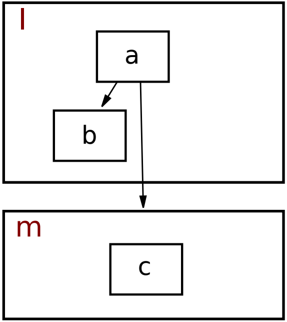


$$\dfrac{\dfrac{}{\gamma_2}\,b \qquad \dfrac{}{\gamma_3}\,c}{\gamma_1}\,a$$

For the validated hiproof, satisfies this goal query:

$$\textbf{ingoals}\,[\gamma_1]$$

Local structure

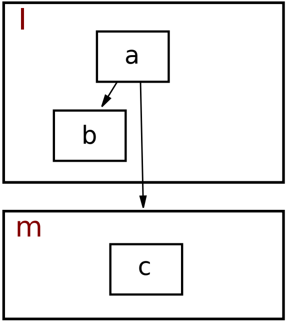


$$\frac{\overline{\gamma_2}\ b \qquad \overline{\gamma_3}\ c}{\gamma_1}\ a$$

For the validated hiproof, satisfies this goal query:

$$\textbf{outgoals}\,[\gamma_3]\;\textbf{then}\;*$$

Local structure



$$\dfrac{\dfrac{}{\gamma_2}\; b \qquad\qquad \dfrac{}{\gamma_3}\; c}{\gamma_1}\; a$$

For the validated hiproof, satisfies this goal query:

$$\textbf{inside}\ *\ (\textbf{outgoals}\,[\gamma_2, \gamma_3]\,\textbf{then}\ *)\ \textbf{then}\ *$$

Basic queries

q	$::=$	*		non-empty
	|	**atomic** *nm*		rule instance
	|	**nothing**		identity
	|	**inside** *nm* q		label match
	|	q_1 **then** q_2		successive nodes
	|	q_1 **beside** q_2		adjacent nodes
	|	**ingoals** *gm*		goals into sub-proof
	|	**outgoals** *gm*		goals out of sub-proof

- Matching is against names *nm* and goals *gm*
- Allows variables, wildcards, predicates, negation
- Semantics given by satisfaction relation

$$s \models_\sigma q$$

for match variable assignment σ.

Connectives

$$
\begin{array}{rcl}
q & ::= & \ldots \\
  & | & q_1 \wedge q_2 \\
  & | & q_1 \vee q_2 \\
  & | & \neg q
\end{array}
$$

- Expected meanings
- Disjunction introduces multiple results

Quantifiers

$$
\begin{array}{rcl}
q & ::= & \ldots \\
 & | & \textbf{somewhere}\ q \\
 & | & \textbf{everywhere}\ q
\end{array}
$$

- domain is subterms, which are valid hiproofs
- **somewhere** allows search in the tree
- **everywhere** checks a property holds globally

Quantifiers

$$
\begin{array}{rcl}
q & ::= & \ldots \\
& | & \textbf{somewhere}\ q \\
& | & \textbf{everywhere}\ q
\end{array}
$$

Example:

somewhere inside mytac $*$

is satisfied if the proof uses mytac.

Quantifiers

$$
\begin{array}{lcl}
q & ::= & \ldots \\
 & | & \textbf{somewhere}\ q \\
 & | & \textbf{everywhere}\ q
\end{array}
$$

Example:

$$
\begin{array}{l}
(\textbf{somewhere inside}\ \mathsf{m}\ \textbf{ingoals}\ G) \\
\vee(\textbf{somewhere atomic}\ \mathsf{b} \wedge \textbf{ingoals}\ G)
\end{array}
$$

means: find the goals input to m or atomic rule b.

For proof shown before, $([\mathsf{l}]\,\mathsf{a} \,;\, \mathsf{b} \otimes \mathsf{id}) \,;\, [\mathsf{m}]\,\mathsf{c}$

query is satisfied by $\{G \mapsto [\gamma_2]\}, \{G \mapsto [\gamma_3]\}$

Recursion

$$\begin{array}{rcl} q & ::= & \ldots \\ & | & \mu Q.q \end{array}$$

- Introduces regular patterns
- For particular input, can unfold to depth of tree

Recursion

$$\begin{array}{rcl} q & ::= & \dots \\ & | & \mu Q.q \end{array}$$

Example:

$$\mu Q.\,(\textbf{atomic}\ \mathsf{a}\ \textbf{then}\ (\textbf{ingoals}\ [\gamma_2]\ \textbf{beside}\ Q)) \\ \vee(\textbf{inside}\ \mathsf{m}\ *)$$

is satisfied by proofs that repeatedly apply the atomic rule a, until reaching a box named m.

Recursion

$$
\begin{array}{rcl}
q & ::= & \ldots \\
 & | & \mu Q.q
\end{array}
$$

With pattern matching, the quantifiers can be derived:

$$
\begin{aligned}
\textbf{somewhere}\, q \stackrel{def}{=} \mu Q.\, q \vee & \\
& (\textbf{inside} \ * \ Q) \vee \\
& (Q\, \textbf{then} \, *) \vee (*\, \textbf{then}\, Q) \vee \\
& (Q\, \textbf{beside} \, *) \vee (*\, \textbf{beside}\, Q)
\end{aligned}
$$

Recursion

$$\begin{array}{rcl} q & ::= & \ldots \\ & | & \mu Q.q \end{array}$$

With pattern matching, the quantifiers can be derived:

$$\begin{aligned} \textbf{everywhere}\, q \stackrel{def}{=} \mu Q.\, q \wedge \; & (\textbf{atomic} \ast \;\vee \textbf{nothing} \;\vee \\ & \;(\textbf{inside} \;\ast\; Q) \vee \\ & \;(Q\, \textbf{then}\, Q) \vee (Q\, \textbf{beside}\, Q)\,) \end{aligned}$$

Motivating Examples

Find the axioms

$$\begin{array}{l}\textbf{select}\ A\ \textbf{from}\ s\ \textbf{where}\\ \qquad\textbf{somewhere}\ \textbf{axiom}\ A\end{array}$$

$$\textbf{axiom}\ nm \stackrel{def}{=} \textbf{atomic}\ nm \wedge \textbf{outgoals}\ []$$

- Select notation to describe the overall result
- Could extend to transformations, updates, etc.

Find the witnesses

$$\textbf{select}\ A\ \textbf{from}\ s\ \textbf{where}\\ \quad \textbf{somewhere}\ \textbf{atomic}\ A \wedge \textbf{atomic}\ ex_{lt}$$

- ex_l is annotated with witness t
- ex_{lt} denotes predicate selecting all such rule names

Which tactics use atomic rule a?

$$\begin{array}{l}\textbf{select}\ L\ \textbf{from}\ s\ \textbf{where}\\ \quad \textbf{somewhere}\ \textbf{inside}\ L\ \textbf{nearby}\ \textbf{atomic}\ a\end{array}$$

$$\textbf{nearby}\ q \stackrel{def}{=} \mu Q.\ \ q \quad \begin{array}{l}\lor\ (Q\ \textbf{then}\ *)\ \lor\ (*\ \textbf{then}\ Q)\\ \lor\ (Q\ \textbf{beside}\ *)\ \lor\ (*\ \textbf{beside}\ Q)\end{array}$$

Are there bits of the proof that have no effect?

select L **from** s **where**
somewhere inside L **ingoals** $G \wedge$ **outgoals** G

Are there duplicated subproofs?

$$\textbf{select}\ L_1, L_2, G_i, G_o\ \textbf{from}\ s\ \textbf{where}$$
$$\textbf{separately}\ \textbf{inside}\ L_1\ qG\ \textbf{and inside}\ L_2\ qG$$

where $qG\ =\ \textbf{ingoals}\ G_i \wedge \textbf{outgoals}\ G_o$

$$\begin{aligned}&\textbf{separately}\ q_1\ \textbf{and}\ q_2 \stackrel{def}{=}\\ &\textbf{somewhere}\ \ ((\textbf{somewhere}\ q_1\ \textbf{then}\ \textbf{somewhere}\ q_2)\\ &\qquad\qquad \vee (\textbf{somewhere}\ q_1\ \textbf{beside}\ \textbf{somewhere}\ q_2))\end{aligned}$$

Closing

Status of PrQL

- Semantics
  - query satisfaction, notions of query equivalence
  - expected meaning for derived operators
- Decidability of satisfaction for given σ
  - solve for sets of σ by unification
- Expressivity
  - any hiproof has characteristic query
  - most of motivating examples
  - non-examples: no measurement/counting/position
- Implementation experiments
  - abstract hiproof objects
  - **Isabelle**: views on proof objects (labels=rules)

Summary

- PrQL, a query language for (hi)proofs
  - uses hiproofs for structure: labels and nesting
  - paper in LPAR next month
- In progress or yet to do:
  - **global structured** queries with paths
  - denotational data model
  - full scale experiments (**HOL Light** and **Isabelle**)
  - query evaluation algorithms (perhaps translation)
  - expressivity, regular trees, nested words
  - connections to other QLs
- Related work:
  - query languages: databases, graphs, programs
  - proof inspection: data mining, TSTP queries
  - Proof Markup Language: interlingua for justifications
  - proof manipulation: . . .

Advertisement

Advertisement

10th International Workshop on
User Interfaces for Theorem Provers (UITP 2012)
11th July 2012, Bremen, Germany, in CICM 2012

- Innovations in UI design for theorem proving
- Proof construction, presentation, manipulation
- Visualisation and diagrams
- User-oriented TP tools and frameworks

Deadline: 1st May 2012

Visit: uitp-ig.org