' $

RALL: Machine-supported Proofs for Relation Algebra

David von Oheimb[1]
Thomas F. Gritzner[2]

German Research Program: *Deduction*

[1]Fakultät für Informatik,
Technische Universität München,
D–80290 München, Germany.
eMail: `oheimb@informatik.tu-muenchen.de`
URL: `http://www.informatik.tu-muenchen.de/~oheimb/`

[2]Universität der Bundeswehr München,
D–85577 Neubiberg, Germany.
eMail: `Thomas.Gritzner@UniBw-Muenchen.DE`

& %

Theorem Prover *RALL*:

*R*elation *A*lgebraic *L*anguage and *L*ogic

- *Relation Algebra*
  - predicate logic with relations as terms
  - database modeling, graph theory, semantics, ...
- *Isabelle*
  - generic theorem proving system
  - object logics: therories
  - proof scripts: tactics and tacticals
- Components of RALL

<table>
<tr><td colspan="3">RALL</td></tr>
<tr><td rowspan="3">Rela-tion Algebra</td><td colspan="2">HOL</td></tr>
<tr><td colspan="2">Isabelle</td></tr>
<tr><td>Predicate Logic</td><td>SML</td></tr>
</table>

Development Goals for RALL

- Reliability
- Flexibility
- Abstractness (axiomatic theory)
- Type-safety (heterogeneous algebra)
- Convenience
- Automation

Further Structure of this Talk

- Relation Algebra
- Formalization
- Interactive Proofs
- Automatic Proofs
- Conclusion

Relation Algebra

$$\mathtt{R} \quad \sqcap \quad \mathtt{S\hat{}} \quad \sqsubseteq \quad \mathtt{I} \quad \cdot \quad \mathtt{R}$$

$$\begin{pmatrix} O & O & L \\ L & L & L \end{pmatrix} \sqcap \begin{pmatrix} L & L \\ O & O \\ L & L \end{pmatrix}^T \sqsubseteq \begin{pmatrix} L & O \\ O & L \end{pmatrix} \cdot \begin{pmatrix} O & O & L \\ L & L & L \end{pmatrix}$$

Order (Lattice) Properties:

$$\mathtt{O} \sqsubseteq \mathtt{R} \sqsubseteq \mathtt{Sup}(\{\mathtt{R},\mathtt{S}\}) = \mathtt{R} \sqcup \mathtt{S} \sqsubseteq \mathtt{L}$$

Monoidal Properties:

$$\mathtt{I} \cdot (\mathtt{R\hat{}\hat{}} \cdot \mathtt{I}) = (\mathtt{I} \cdot \mathtt{R\hat{}\hat{}}) \cdot \mathtt{I} = \mathtt{R\hat{}\hat{}} = \mathtt{R}$$

Schröder Equivalence(s):

$$(\mathtt{Q} \cdot \mathtt{R} \sqsubseteq \mathtt{S}) \Leftrightarrow (\mathtt{-S} \cdot \mathtt{R\hat{}} \sqsubseteq \mathtt{-Q})$$

Atomicity:

$$\mathtt{R = Sup}(\{\mathtt{a.\ Atom\ a} \wedge \mathtt{a} \sqsubseteq \mathtt{x}\})$$

Isomorphism from Relation Algebra to *Atom Structure*:

$$\mathfrak{A} = (A, i, C, T) \text{ with } i, T : A \to A,\ C \subseteq A \times A \times A$$

$$\langle a, a, i(a) \rangle \in C, \quad \langle c, a, b \rangle \in C \Leftrightarrow \langle a, c, T(b) \rangle \in C$$

Formalization

Hierarchy of Isabelle Theories:

- `Lattice` theory

  $\_ \sqcap \_ :: (\sigma{::}\mathtt{lattice}) \Rightarrow \sigma \Rightarrow \sigma$

  $\mathtt{(x \sqcap y) \sqcap z = x \sqcap (y \sqcap z)}$

  $\mathtt{(\forall x {\in} A.\ l \sqsubseteq x) \longrightarrow l \sqsubseteq Inf(A)}$

- `Relations` theory

  $\_ \cdot \_ :: (\alpha,\beta)\mathtt{rel} \Rightarrow (\beta,\gamma)\mathtt{rel} \Rightarrow (\alpha,\gamma)\mathtt{rel}$

  $\mathtt{I} :: (\alpha,\alpha)\mathtt{rel}$

  $\_\hat{} :: (\alpha,\beta)\mathtt{rel} \Rightarrow (\beta,\alpha)\mathtt{rel}$

  $\mathtt{(R \neq O) = (L \cdot R \cdot L = L)}$

- Applications

  $\mathtt{S \prec R} \equiv \mathtt{-(-S \cdot R\hat{})}$

  $\mathtt{Powerset\ e} \equiv \mathtt{SyQ(e,e) \sqsubseteq I} \wedge$
  $\quad \mathtt{\forall R.\ Total(SyQ(R\hat{},e))}$

Interactive Proofs

General Technique

Backward chaining with rules and tactics

- Predicate-logical reasoning
- Algebraic manipulation
  - Substitution: $\dfrac{\texttt{s = t; P s}}{\texttt{P t}}$
  - Estimation: $\dfrac{\texttt{R} \sqsubseteq \texttt{R'; R'}\cdot\texttt{S} \sqsubseteq \texttt{T}}{\texttt{R}\cdot\texttt{S} \sqsubseteq \texttt{T}}$

Examle

```
> goal Relations.thy "R'=R ⟶ (I⊓S)·R' ⊑ R";
    R'=R ⟶ (I⊓S)·R' ⊑ R
> by(safe_tac HOL_cs);  (* predicate logic *)
    (I⊓S)·R ⊑ R
> by(mtac meet_lb1 1);  (* rule "x⊓y ⊑ x" *)
    I·R ⊑ R
> by(sstac I_def1 1);   (* rule "I·R = R" *)
    R ⊑ R
> by(rtac incl_refl 1); (* rule "x ⊑ x" *)
    No subgoals!
```

’ $

Automatic Proofs

Atomization

Observation: Direct automation of "algebraic" derivations is difficult

Key idea: Transformation from algebraic to propositional level

Solution *Atomization*, i.e. transformation to atom structure

Auxiliary notions

$$\begin{array}{lcl}
\texttt{Atom a} & \equiv & \texttt{a}\neq\texttt{0} \wedge \forall\texttt{x}.\ \texttt{x}\neq\texttt{0} \wedge \texttt{x}\sqsubseteq\texttt{a} \longrightarrow \texttt{x=a} \\
\texttt{a}\mapsto\texttt{x} & \equiv & \texttt{Atom a} \wedge \texttt{a}\sqsubseteq\texttt{x} \\
\texttt{i(a)} & \equiv & \texttt{a\^{}}\cdot\texttt{a}\sqcap\texttt{I} \\
\texttt{c}\leadsto\texttt{a}\cdot\texttt{b} & \equiv & \texttt{c}\mapsto\texttt{a}\cdot\texttt{b} \wedge \texttt{Atom a} \wedge \texttt{Atom b}
\end{array}$$

& %

Automatic Proofs: *Transformation*

Rules: (Isomomorphism!)

$\mathtt{x = y}$	$= \mathtt{x} \sqsubseteq \mathtt{y} \wedge \mathtt{y} \sqsubseteq \mathtt{x}$
$\mathtt{x} \sqsubseteq \mathtt{y}$	$= \forall \mathtt{a.}\ \mathtt{a} \mapsto \mathtt{x} \longrightarrow \mathtt{a} \mapsto \mathtt{y}$
$\mathtt{a} \mapsto \mathtt{x} \sqcap \mathtt{y}$	$= \mathtt{a} \mapsto \mathtt{x} \wedge \mathtt{a} \mapsto \mathtt{y}$
$\mathtt{a} \mapsto \mathtt{x} \sqcup$	$= \mathtt{a} \mapsto \mathtt{x} \vee \mathtt{a} \mapsto \mathtt{y}$
$\mathtt{a} \mapsto \mathtt{-x}$	$= \mathtt{Atom\ a} \wedge \neg \mathtt{a} \mapsto \mathtt{x}$
$\mathtt{a} \mapsto \mathtt{Inf(A)}$	$= \mathtt{Atom\ a} \wedge \forall \mathtt{z} \in \mathtt{A.}\ \mathtt{a} \mapsto \mathtt{z}$
$\mathtt{a} \mapsto \mathtt{Sup(A)}$	$= \exists \mathtt{z} \in \mathtt{A.}\ \mathtt{a} \mapsto \mathtt{z}$
$\mathtt{a} \mapsto \mathtt{R} \cdot \mathtt{S}$	$= \exists \mathtt{r\ s.}\ \mathtt{a} \leadsto \mathtt{r} \cdot \mathtt{s} \wedge \mathtt{r} \mapsto \mathtt{R} \wedge \mathtt{s} \mapsto \mathtt{S}$
$\mathtt{a} \mapsto \mathtt{R\hat{}}$	$= \mathtt{a\hat{}} \mapsto \mathtt{R}$
$\mathtt{a} \mapsto \mathtt{O}$	$= \mathtt{False}$
$\mathtt{a} \mapsto \mathtt{I}$	$= \mathtt{Atom\ a} \wedge \mathtt{a=i(a)}$
$\mathtt{a} \mapsto \mathtt{L}$	$= \mathtt{Atom\ a}$

Result: pure predicate-logical formulas with predicates $\mathtt{Atom\ X}$, $\mathtt{X} \mapsto \mathtt{R}$, $\mathtt{X} \leadsto \mathtt{Y} \cdot \mathtt{Z}$, or $\mathtt{X=Y}$, where $\mathtt{X}$, $\mathtt{Y}$, $\mathtt{Z}$ of the form $\mathtt{a}$, $\mathtt{a\hat{}}$, $\mathtt{i(a)}$, or $\mathtt{i(a\hat{})}$

Automatic Proofs: *Atom-level Rules*

Pre-Schröder Rules: $\dfrac{\mathtt{q} \leadsto \mathtt{r} \cdot \mathtt{s}}{\mathtt{r} \leadsto \mathtt{q} \cdot \mathtt{s\hat{}}}$

Pre-Associativity: $\dfrac{\mathtt{a} \leadsto \mathtt{f} \cdot \mathtt{z};\ \mathtt{f} \leadsto \mathtt{x} \cdot \mathtt{y}}{\exists \mathtt{g}.\ \mathtt{a} \leadsto \mathtt{x} \cdot \mathtt{g} \wedge \mathtt{g} \leadsto \mathtt{y} \cdot \mathtt{z}}$

Pre-Identity: $\dfrac{\mathtt{Atom\ a}}{\mathtt{a} \leadsto \mathtt{a} \cdot \mathtt{i(a)}}$, $\dfrac{\mathtt{q} \leadsto \mathtt{k} \cdot \mathtt{i(j)};\ \mathtt{Atom\ j}}{\mathtt{q = k}}$

Automatic Proofs: *Proof Procedure*

1. Preparation:
   (a) unfolding of definitions
   (b) simplification: `--x = x`, `0·R = 0`
   (c) atomization

2. Main Part: depth-first search with
   (a) simplification: `a^^ = a`, `i(i(a)) = i(a)`
   (b) predicate-logical steps
   (c) atom lemmas: $\dfrac{\mathtt{q} \leadsto \mathtt{r} \cdot \mathtt{s}}{\mathtt{Atom\ s}}$, $\dfrac{\mathtt{Atom\ j}}{\mathtt{i(j)} \mapsto \mathtt{I}}$
   (d) pre-identity rules
   (e) special tactics

Automatic Proofs: *Special Tactics*

1. Generation of all association variants of incidence pairs within the premises, applying the pre-associativity rules
2. If there is a premise `a↦I` : rewriting of all atoms `a` to `i(a)`, making the pre-identity rules applicable
3. Solution of subgoals by applying the pre-Schröder rules (modulo involution of `^`)

Automatic Proofs: *Examples*

```
> fpg "Q·R⊓S ⊑ (Q⊓S·R^)·(R⊓Q^·S)";

> fpg "SyQ(Q,R)·SyQ(R,S) ⊑ SyQ(Q,S)";
```

Automatic Proofs: *Evaluation*

Soundness: Yes

Completeness: No

- Lattice-theoretic fraction: satisfactory
- Monoidal fraction: to be improved

```
> rpg "True ∧ R⊓S^ ⊑ I·R";
val it = "True ∧ ?R⊓?S^ ⊑ I·?R" : thm


> g "True ∧ R⊓S^ ⊑ I·R";
 1. R⊓S^ ⊑ I·R

> by(simp_tac AtomRA_ss 1);
 1. ∀a. a↦R ∧ a^↦S ⟶
      ∃r s. a⇝r·s ∧ r↦I ∧ s↦R

> by(rstep_tac 1 THEN ... rstep_tac 1);
 1. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ a⇝?r3(a)·?s4(a)
 2. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ ?r3(a)↦I
 3. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ ?s4(a)↦R

> by(rstep_tac 1); (* "Atom r ==> r⇝i(r^)·r" *)
 1. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ Atom a
 2. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ i(a^)↦I
 3. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ a↦R

by(rstep_tac 1); (* "a↦x ==> Atom a" *)
 1. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ i(a^)↦I

by(rstep_tac 1); (* "Atom j ==> i(j^)↦I" *)
 1. ⋀a. ⟦ a↦R; a^↦S ⟧ ⟹ Atom a^

by(rstep_tac 1); (* "Atom (a^) = Atom a" *)
No subgoals!
```

Conclusion

Goals reached:

Reliable, flexible, type-safe theorem prover for abstract Relation Algebra

Goals partially reached:

- Convenient interactive reasoning
- Automated reasoning

Statistics

- Size of RALL: 25 axioms, 35 definitions,
- 160 lemmas, 180 theorems (62% autom.)
- Total time of development: 12 man-weeks

Experience, Future Work?

- Atomization enables automation
- Further investigation necessary
- Transfer to automatic 1st-order provers?