On the Use of Relation Algebra and the BDD-based Tool RelView in Algorithm Development

Rudolf Berghammer

Institut für Informatik
Christian-Albrechts-Universität zu Kiel

www.informatik.uni-kiel.de/~progsys/relview.shtml
Introduction

Two well established domains in computer science and applied mathematics are:

- Design of efficient algorithms.
- Formal program development and verification.

In recent years many computer scientists noticed that the design and verification of efficient algorithms which are not only correct “in principle” but in all details can benefit from techniques of formal program specification, derivation and verification.

Therefore, there is an increasing cooperation between the two domains.

For a lot of discrete algorithmic problems relations are a convenient means for the formal development of efficient algorithms.
Main reasons for the success of relations in this application field:

- Many important structures of discrete mathematics are closely related to relations.
- With relations can be calculated very well using the concept of an abstract relation algebra.
- Computer support is possible for theorem proving as well as for the manipulation of concrete finite relations.

Since 1993 at the University of Kiel we have developed an OBDD-based manipulation and visualization tool for relations, called RelView.

Aim of the talk:

To give an impression how a combination of relation algebra and RelView can be useful in algorithm development.
Relation Algebra

Relations:

- $R$ is a relation with domain $X$ and range $Y$:
  \[ R : X \leftrightarrow Y \]

  $X \leftrightarrow Y$ is the type of $R$.

- Instead of $(x, y) \in R$ we use Boolean matrix notation:
  \[ R_{x,y} \] (or $R_x$ if the range is a singleton set)

Signature of relation algebra:

- Constants: $O, L, I$.
- Operations: $R \cup S, R \cap S, RS, \overline{R}, R^T$.
- Tests: $R \subseteq S, R = S$. 
A relation $v : X \leftrightarrow Y$ is a vector or row-constant if $v = vL$.

The normal case is $v : X \leftrightarrow 1$, where $1 := \{\bot\}$ is a singleton set. Then we define for subsets $Y$ of $X$:

$$v \text{ represents } Y \iff Y = \{x \in X : v_x\}$$

Further relational modeling of sets via membership relations $M : X \leftrightarrow 2^X$, such that for all $x \in X$ and subsets $Y$ of $X$:

$$M_{x,Y} \iff x \in Y$$

Example, where $X = \{1, 2, 3, 4, 5\}$ and marked columns (vectors) represent $Y = \{2, 3, 5\}$ and $X$, respectively:
The Relation-Algebraic Tool **RELVIEW**

Cut completion of a partial order
Example of a relational function in \texttt{RelView}:

\[
\text{Hasse}(C) = C \land \neg(C \ast C).
\]

\text{Hasse} computes the \textbf{Hasse-diagram} \( H_C = C \cap \overline{CC} \) of a strict-order relation \( C \).

Example of a relational program in \texttt{RelView}:

\begin{verbatim}
Gavril(R)
    DECL E, c, e
    BEG  c = On1(R);
         E = R;
         WHILE -empty(E) DO
             e = edge(E);
             c = c \cup \text{dom}(e);
             E = E \land -\left( e \ast \text{L}(e) \land R \right) \land -\left( \text{L}(e) \ast e \land R \right) \OD
         RETURN c
    END.
\end{verbatim}

\texttt{Gavril} is a \texttt{RelView}-implementation of the Gavril/Yannakakis approximation algorithm for \textbf{minimum vertex covers}.
Example for \texttt{RELVIEW}'s visualization potential.

Hasse-diagram \( H_{\mathcal{B}_5} \) of the Boolean lattice \( \mathcal{B}_5 \) with a highlighted Dilworth chain partition, a corresponding maximum antichain . . .
... and the single chains as subrelation / subgraph.
Prototyping, Testing, Visualization

• Starting point: More or less formal description of a problem.

  Transformations linking together logic with relation algebra.
  \textbf{RelView}: \textit{Rapid prototyping}.

• Next: Formal relation-algebraic description of it.

  Application of certain development methods.
  \textbf{RelView}: \textit{Algorithm testing and debugging}
  to detect errors, alternative solutions, etc.

• Final result: Relational program for computing a solution.

  \textbf{RelView}: \textit{Clarification} of program executions.
  \textit{Visualization} of computed results.
Support of Relation-algebraic Reasoning

• When developing a relational program frequently relation-algebraic formulae have to be verified.
  – Especially: Maintainance of invariants.

• Use of RelView here:
  – Detection of auxiliary properties by experimenting with expressions and looking at the graphical representations of the results.

• Important in respect thereof:
  The relations needed for tests can be randomly generated.
  – General relations.
  – Specific relations (symmetric, acyclic, functional, etc).
  – Choice of dimension and density.
Example: A Property of Vectors

**Conjecture:** For all relations $R$ and vectors $v$ we have

$$ (1) \quad \overline{R}v = \overline{R}v. $$

Equation (1) as \text{RELVIEW}-function using the pre-defined equality-test $\text{eq}$:

$$ \text{test}(R,v) = \text{eq}(-((R*v),-R*v)) $$

Result of tests: Equation (1) is not valid in general but seems to hold if the vector has exactly one 1-entry.

**Refined conjecture:** For all relations $R$ and vectors $v$ we have

$$ (2) \quad Lv = L, vv^\top \subseteq I \implies \overline{R}v = \overline{R}v. $$

A relation-algebraic proof of (2) is simple. E.g., $\overline{R}v \subseteq \overline{R}v$ is shown by

$$ L = Lv = (R \cup \overline{R})v = Rv \cup \overline{R}v. $$
Realistic Appraisal of Approximation Algorithms


- Standard approach:
  To prove a worst-case bound for the distance between an optimal and a computed solution.

- Necessary for a realistic appraisal:
  To ascertain the actual quality of solutions in practical applications.

- Use of RelView here:
  Its OBDD-implementation frequently allows to compute optimal solutions also for non-trivial cases (like graphs mit more than 100 vertices, in fortunate cases even more than 1000 vertices) using simple programs.
Example: Gavril-Yannikakis-Algorithm

Let $R : X \leftrightarrow X$ adjacency relation of an undirected graph, $M : X \leftrightarrow 2^X$ membership relation, $C : 2^X \leftrightarrow 2^X$ size comparison relation.

Logical specification of $S \in 2^X$ to be a vertex cover:

$$\forall x, y : R_{x,y} \rightarrow x \in S \lor y \in S.$$ \[  \]

Relation specification of the vector representing the set of vertex covers:

$$L(R \overline{M} \cap \overline{M})^T : 2^X \leftrightarrow 1$$

Relational function for the vector representing the set of minimum vertex covers:

$$\text{mincover}(R) = \text{least}(C, L(R \overline{M} \cap \overline{M})^T),$$

where $\text{least}(P, v) = v \cap \overline{P}v$. 

Workshop on Boolean Problems, 2008, Freiberg
Translation of *Mincover* into *RelView*-code:

\[
\begin{align*}
\text{Mincover}(R) \\
\text{DECL} \quad \text{least}(P,v) &= v \& -(-P \ast v); \\
M, C \\
\text{BEG} \quad M &= \text{epsi}(O(R)); \\
C &= \text{cardrel}(O(R)) \\
\text{RETURN} \quad \text{least}(C, -(L1n(R) \ast (R \ast -M \& -M))^) \\
\text{END}.
\end{align*}
\]

*RelView*-program for a single vertex cover (as vector of type \(X \leftrightarrow 1\)):

\[
\begin{align*}
\text{SingleMincover}(R) \\
\text{DECL} \quad M, p \\
\text{BEG} \quad M &= \text{epsi}(O(R)); \\
p &= \text{point}(\text{Mincover}(R)) \\
\text{RETURN} \quad M \ast p \\
\text{END}.
\end{align*}
\]
Results of one test series for 100 vertices using SingleMincover and RelView-implementations of the Gavril/Yannakakis-algorithm.

$x$-axis: Densities of the random graphs.

$y$-axis: Ratio of the result sizes and $c_{opt}$.

Dotted curve: Gavril (original algorithm).
Continuous curve: Additional removal of redundant vertices.
Use in Teaching

At the University of Kiel, the \texttt{RelView} tool has been extensively used in teaching.

- Lectures „Orders and Lattices“ and „Relational Methods in Computer Science“.
- Exercises accompanied with them.
- Seminars. practical exercises, Diploma theses, etc.

We find it very attractive to use the tool for producing and depicting good examples and counter-examples, which frequently have been proven to be the key of fully understanding an advanced concept.

Experience has shown that the visualization and animation possibilities of \texttt{RelView} also qualify it for the demonstration how certain algorithms work.
Example: Linear Extension and Jumps

Strict-order relation $R$

$R \cup H_E$, where $E$ linear extension of $R$

dotted: relation $H_E \cap \overline{R} \cap \overline{R^T}$ of jumps
Some Recent Applications


- Computation and visualization of **lattices of subgroups** and normal subgroups (B., RelMiCS 2006, LNCS 4136).


- Exact computation of **minimum feedback vertex sets** (B., Fronk, Fundamenta Informaticae 70, 2006).


- **Multi-objective optimization** (Dietrich, Kehden, Neumann, RelMiCS 2008, LNCS 4988).