

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: $\mathbf{O}, \mathbf{L}, \mathbf{I}$.
- Operations: $R \cup S, R \cap S, RS, \overline{R}, R^\top$.
- Tests: $R \subseteq S, R = S$.

The Relation-Algebraic Tool RELVIEW

The screenshot displays the RELVIEW software interface, which is used for relation algebra. It is divided into several panels:

- Evaluate Term:** Contains input fields for 'Result', 'Term', and 'History', along with 'Clear History', 'Eval', and 'Close' buttons.
- Directory:** Shows a list of relations and domains. The 'Relations' tab is active, displaying:

Relation	Definition
R	graph 7 X 7
C	=graph 10 X 10
E	----- 7 X 10
S	=graph 32 X 32
- Matrix View:** A 10x10 grid representing the matrix C. The title bar indicates 'NAME: C DIM: 10 X 10 ORIGIN: (r: 1 / c: 1)'. The grid shows a pattern of shaded cells.
- Graph View:** A graph with 10 nodes (labeled 1 to 10) and directed edges. The title bar indicates 'C with 10 nodes'. The graph shows a complex structure with many edges, representing the cut completion of a partial order.
- Filechooser:** A window showing the file system structure. The current directory is '/home/rub/SYSTEMANWENDUNGEN/RELVIEW/VERBAENDE-ORDNUNGEN/SCH'. The file 'SchnitteEpsI.prog.ddd' is selected. The filter is set to '*.ddd'.

Cut completion of a partial order

Example of a relational function in RELVIEW:

$$\text{Hasse}(C) = C \ \& \ \neg(C * C).$$

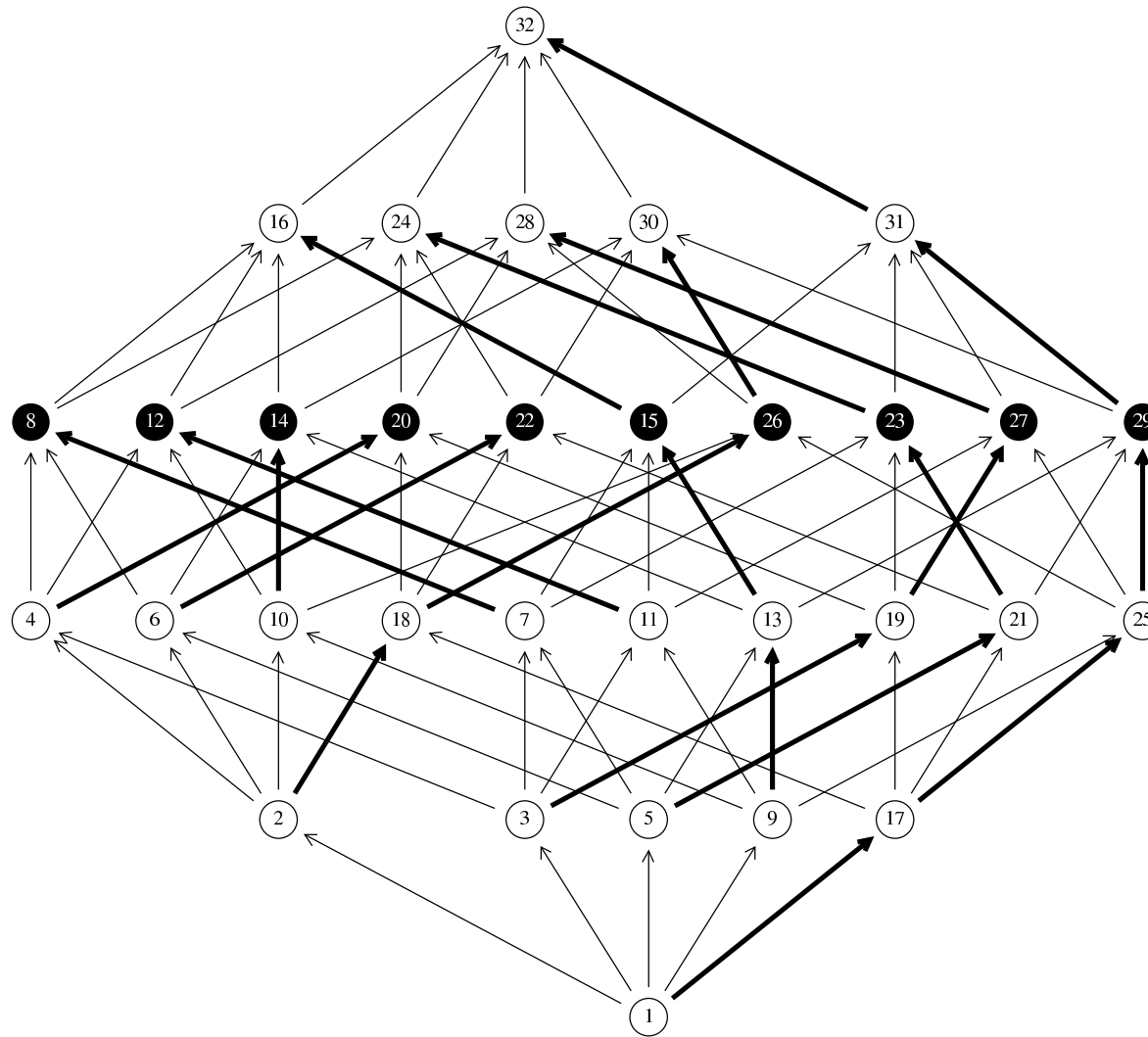
Hasse computes the **Hasse-diagram** $H_C = C \cap \overline{CC}$ of a strict-order relation C .

Example of a relational program in RELVIEW:

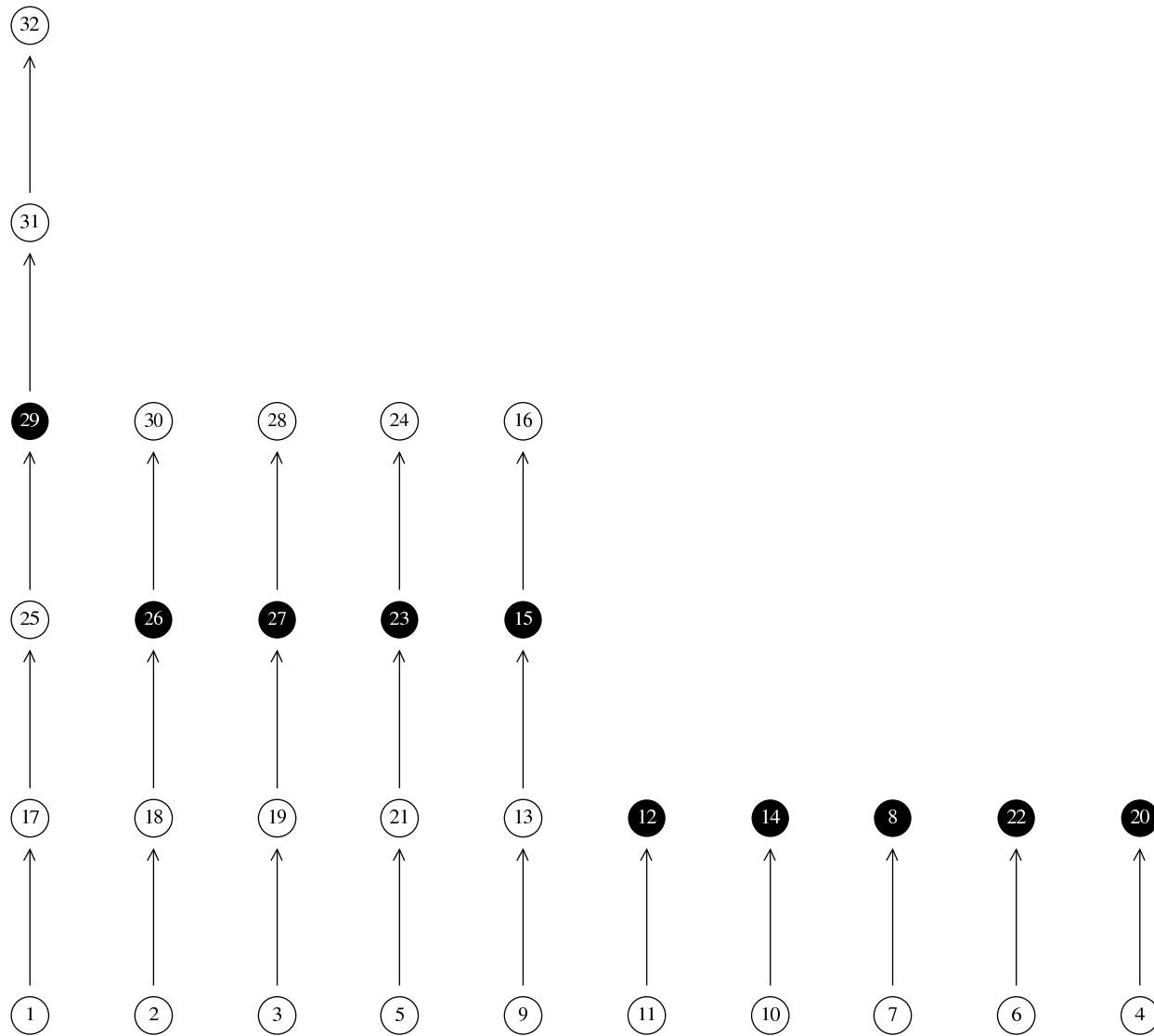
```
Gavril(R)
  DECL E, c, e
  BEG  c = On1(R);
      E = R;
      WHILE -empty(E) DO
        e = edge(E);
        c = c | dom(e);
        E = E & -(e*L(e) & R) & -(L(e)*e & R) OD
      RETURN c
  END.
```

Gavril is a RELVIEW-implementation of the Gavril/Yannakakis approximation algorithm for **minimum vertex covers**.

Example for 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.

RELVIEW: **Rapid prototyping.**

- Next: Formal relation-algebraic description of it.

Application of certain development methods.

RELVIEW: **Algorithm testing and debugging**
to detect errors, alternative solutions, etc.

- Final result: Relational program for computing a solution.

RELVIEW: **Clarification** of program executions.
Visualization of computed results.

Support of Relation-algebraic Reasoning

- When developing a relational program frequently relation-algebraic formulae have to be verified.
 - Especially: Maintenance of invariants.
- Use of RELVIEW here:
 - Detection of auxiliary properties by experimenting with expressions and looking at the graphical representations of the results.
 - Testing validity of arbitrary Boolean combinations of inclusions.
- 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

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

Equation (1) as RELVIEW-function using the pre-defined equality-test eq:

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

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

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

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

$$\mathbf{L} = \mathbf{L}v = (R \cup \overline{R})v = Rv \cup \overline{R}v.$$

Realistic Appraisal of Approximation Algorithms

- Approximation algorithms: Efficient computation of nearly optimal solutions of hard optimization problems.
- 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, $\mathbf{M} : X \leftrightarrow 2^X$ membership relation, $\mathbf{C} : 2^X \leftrightarrow 2^X$ size comparison relation.

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

$$\forall x, y : R_{x,y} \rightarrow x \in S \vee y \in S.$$

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

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

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

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

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

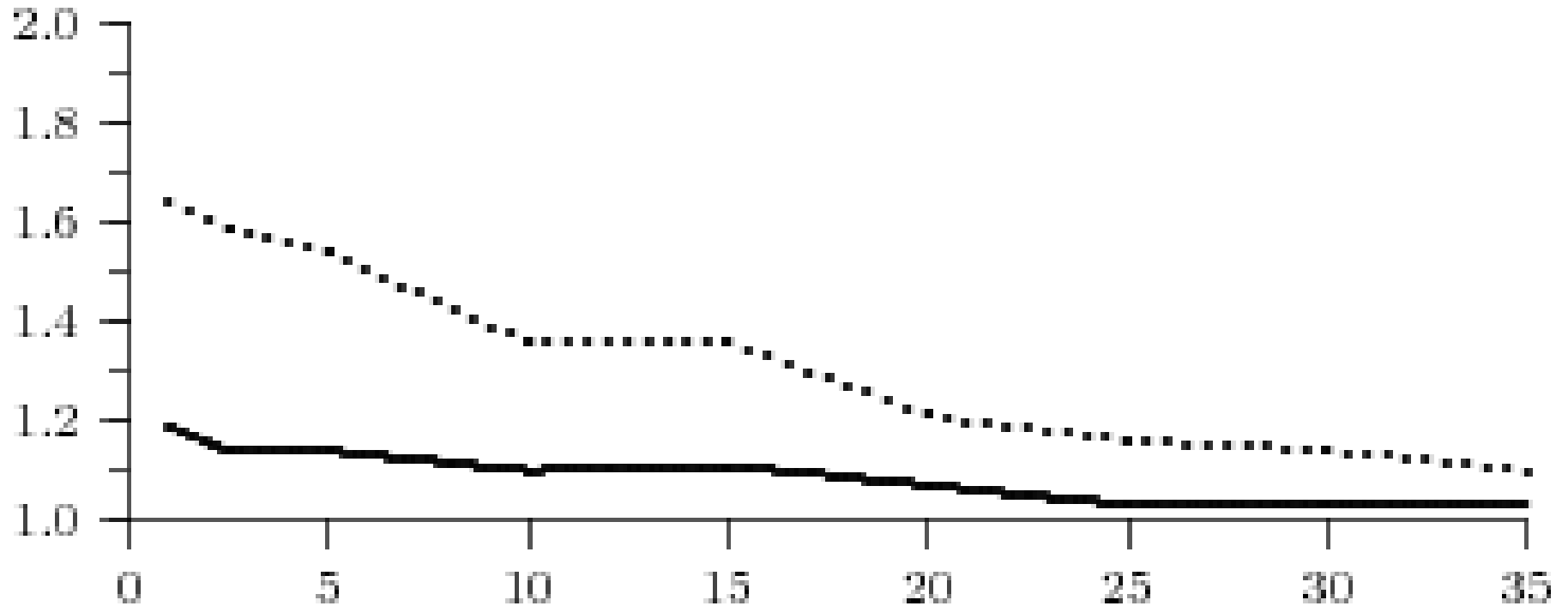
Translation of *Mincover* into RELVIEW-code:

```
Mincover(R)
  DECL least(P,v) = v & -(-P * v);
      M, C
  BEG  M = epsi(0(R));
      C = cardrel(0(R))
      RETURN least(C, -(L1n(R)*(R*-M & -M))^)
  END.
```

RELVIEW-program for a single vertex cover (as vector of type $X \leftrightarrow \mathbf{1}$):

```
SingleMincover(R)
  DECL M, p
  BEG  M = epsi(0(R));
      p = point(Mincover(R))
      RETURN M*p
  END.
```

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

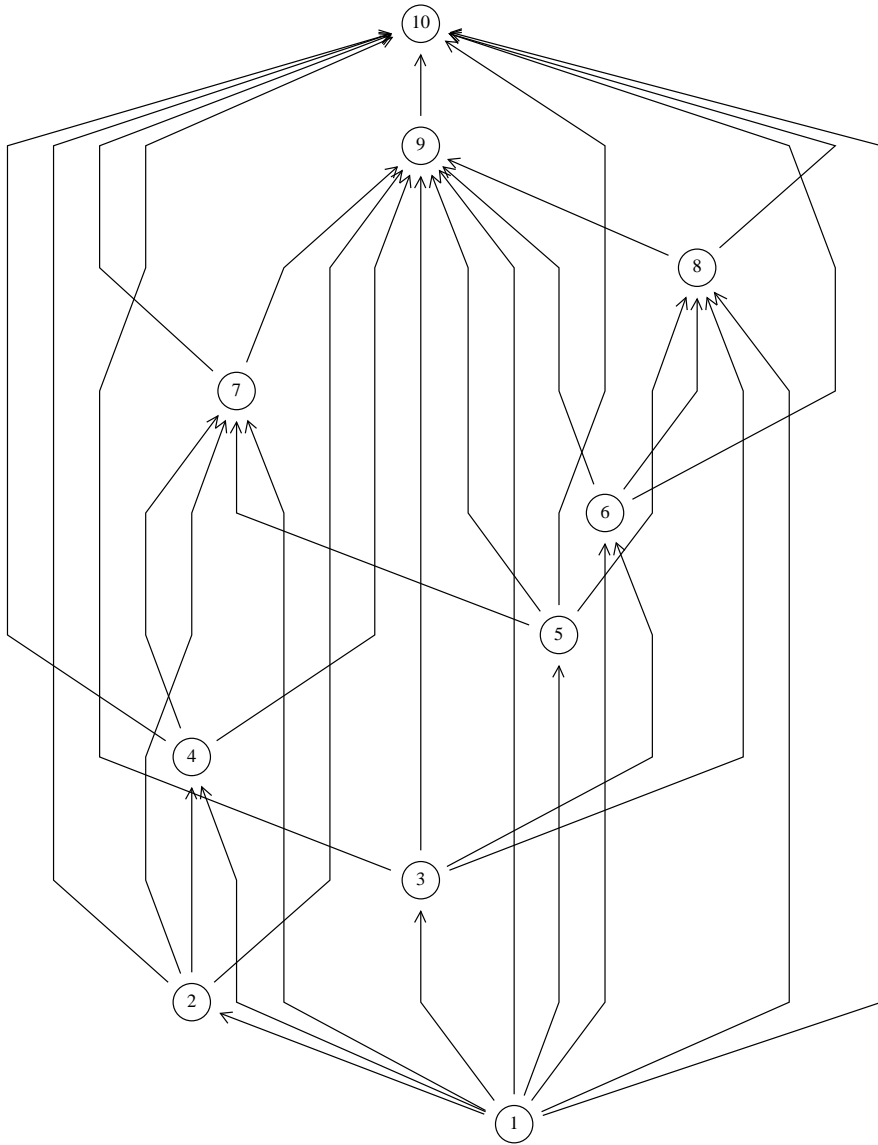
At the University of Kiel, the RELVIEW tool has been extensively used in teaching.

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

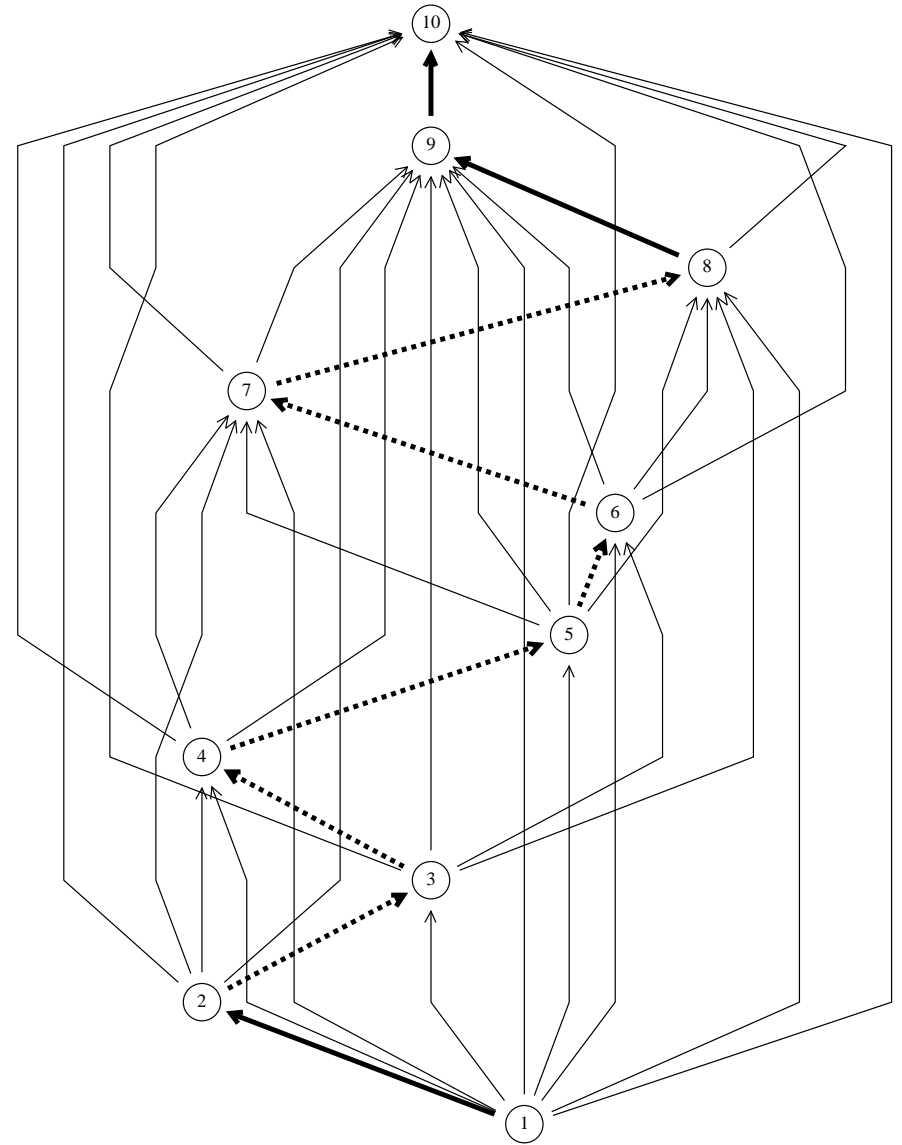
We find it very attractive to use the tool for producing and depicting good examples and counter-examples, which frequently have been proven to be the key of fully understanding an advanced concept.

Experience has shown that the visualization and animation possibilities of 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

- **Evolutionary algorithms** (Kehden, Neumann, EvoCop 2006, LNCS 3906; Kehden, Ph.D. thesis, 2008).
- Computation and visualization of **lattices of subgroups** and normal subgroups (B., RelMiCS 2006, LNCS 4136).
- Solution of **order- and lattice-theoretic problems** (B., CASC 2006, LNCS 4194; B., Schmidt, CASC 2007, LNCS 4770; B., Acta Informatica 45, 2008).
- Exact computation of **minimum feedback vertex sets** (B., Fronk, Fundamenta Informaticae 70, 2006).
- Formation of **coalitions and alliances** (B., Rusinowska, de Swart, Europ. J. Operational Research 176, 2007).
- Construction of **timetables** (B., Kehden, OR 2007; B., Kehden, RelMiCS 2008, LNCS 4988; Kehden, Ph.D. thesis, 2008).
- **Multi-objective optimization** (Dietrich, Kehden, Neumann, RelMiCS 2008, LNCS 4988).