Fundamenta Informaticae XX (2005) 1–16
IOS Press

Exact Computation of Minimum Feedback Vertex Sets with Relational Algebra

Rudolf Berghammer
Institute of Computer Science and Applied Mathematics
University of Kiel, D-24098 Kiel, Germany
rub@informatik.uni-kiel.de

Alexander Fronk
Software Technology
University of Dortmund, D-44221 Dortmund, Germany
alexander.fronk@uni-dortmund.de

Abstract. A feedback vertex set of a graph is a subset of vertices containing at least one vertex from every cycle of the graph. Given a directed graph by its adjacency relation, we develop a relational algorithm for computing a feedback vertex set of minimum size. In combination with a BDD-implementation of relations, it allows to exactly solve this NP-hard problem for medium-sized graphs.

Keywords: Directed graphs, feedback vertex sets, elementary chordless cycles, relational algebra, RELVIEW tool.

1. Introduction

In this paper we are concerned with the minimum feedback vertex set problem of directed graphs. A feedback vertex set of a graph is a subset of vertices containing at least one vertex from every cycle of the graph, and a minimum feedback vertex set is a feedback vertex set of minimal size. Computing minimum feedback vertex sets is of fundamental importance in combinatorial optimization. The problem originally appeared in the context of switching circuits; cf. [20]. Further applications include, for example, the analysis of signal flow graphs and electrical networks [8], the solution of linear algebraic systems [16], as well as constraint satisfaction problems and Bayesian inference [1].

As shown in [15], the minimum feedback vertex set problem is NP-hard. Due to this negative result, polynomial time approximation algorithms have been developed for computing near optimal feedback vertex sets. In [1] it is shown that a minimum feedback vertex set for undirected graphs can be approximated within a constant factor of 4. In [14], this approximation ratio has been improved to 2, and to the best of our knowledge 2 is still the smallest approximation ratio for the time being. For directed graphs, the case we are concerned with, a solution seems much harder to approximate. To the best of our knowledge, currently the smallest ratio is $O(\log n \log\log n)$ as found in [10].

Our approach, however, is based on relational algebra in the sense of [19]. We stepwise develop an exact algorithm solving the minimum feedback vertex set problem for directed graphs. The algorithm is based on a representation of directed graphs by adjacency relations and on the reduction of the feedback vertex set problem to the enumeration of all vertex sets of elementary chordless cycles. In essence, the enumeration of these vertex sets corresponds to exhaustively testing each subset of vertices for being the vertex set of an elementary chordless cycle. Therefore, one may expect the execution time of our algorithm to grow so rapidly that one can only deal with graphs of fairly small size. Precisely here, a specific implementation of relations based on binary decision diagrams (BDDs for short) appears beneficial. Due to this sophisticated implementation, we can compute exact solutions of the minimum feedback vertex set problem for medium-sized graphs consisting of about 100 vertices in general and more in some advantageous cases. We use the relation-algebraic programming and visualization tool RELVIEW [3, 4] for this computation. It implements relations as well as many operations and tests on them by means of reduced ordered BDDs; see [17, 5, 18]. RELVIEW thereby provides a high-performance and nice mechanization of relational algebra.

The remainder of this paper is organized as follows. Some relation-algebraic preliminaries used throughout this paper are provided in the next Section 2. In particular, we deal with some different possibilities to model sets and direct products with relations. These constructions are of importance for our approach discussed in detail in Sections 3, 4, and 5: Section 3 shows how the minimum feedback vertex set problem can be reduced to the column-wise enumeration of vertex sets of elementary chordless cycles; Sections 4 presents a first relation-algebraic solution of the latter problem, and Section 5 refines it in order to improve performance. In Section 6, we report on results of running practical experiments with RELVIEW and sketch some refinements which in some cases even considerably speed-up the computation time. Section 7 contains concluding remarks and briefly describes some applications in the world of Petri nets where the calculation of vertex sets of elementary chordless cycles can similarly be of advantage.

2. Relation-Algebraic Preliminaries

We assume the reader to be familiar with relational algebra. Details can be found, for example, in [19]. Nonetheless, we provide in the following the notations and definitions as used throughout this paper. In particular, we focus on representations of sets and a relational description of direct products which are not commonly used and require some detailed explanation.

We write $R : X \leftrightarrow Y$ if R is a relation with domain X and range Y, i.e., a subset of $X \times Y$. Both sets X and Y are assumed to be nonempty. Since we are interested in computations, we additionally assume that the sets X and Y of R's *type* $X \leftrightarrow Y$ *are finite*. Hence, we may consider every relation R as a Boolean matrix with $|X|$ rows and $|Y|$ columns. Since this interpretation is well suited for many purposes and Boolean matrices are the main means to depict relations graphically in RELVIEW, we use Boolean matrix notions and notations when appropriate. In particular, we speak of rows and columns of relations and write $R_{x,y}$ instead of $\langle x, y\rangle \in R$ or $x\,R\,y$.

We use the notation R^{T} for *transposition*, $\overline{R}$ for *complement*, $R \cup S$ for *union*, $R \cap S$ for *intersection*, RS for *composition*, and $R \subseteq S$ for *inclusion*. The *empty relation* is denoted by $\mathbb{O}$, the *universal relation* by $\mathbb{L}$, and the *identity relation* by $\mathbb{I}$. Finally, we will use the *transitive closure* $R^+ = \bigcup_{i \geq 1} R^i$, where the powers of R are defined by $R^0 := \mathbb{I}$ and $R^{i+1} := RR^i$.

A relation R is called *univalent* if $R^{\mathsf{T}}R \subseteq \mathbb{I}$, and *total* if $R\mathbb{L} = \mathbb{L}$. A univalent and total relation R is called a *mapping*. It is an *injective mapping* if, in addition, R^{T} is univalent.

Relational algebra provides different possibilities to model sets. The first possibility applied in this paper uses *vectors* (or row-constant relations), that is relations v with $v = v\mathbb{L}$. For a vector v of type $X \leftrightarrow Y$ this condition means: Whatever set Z and universal relation $\mathbb{L} : Y \leftrightarrow Z$ we choose, an element $x \in X$ is in relationship $(v\mathbb{L})_{x,z}$ to either none or every element $z \in Z$. Since for a vector the range is irrelevant, in this paper we mainly consider vectors $v : X \leftrightarrow \mathbf{1}$ with a specific singleton set $\mathbf{1} = \{\bot\}$ as range and omit in such cases the second subscript, i.e., write v_x instead of $v_{x,\bot}$. Such a vector can be considered as a Boolean matrix with exactly one column, i.e., as a Boolean column vector, and it *represents* the subset $\{x \in X : v_x\}$ of its domain X.

We will also use injective mappings to model subsets. Given an injective mapping $\imath : Y \leftrightarrow X$, we may consider Y as a subset of X by identifying it with its image under $\imath$. If Y is actually a subset of X and $\imath$ is the identity mapping from Y to X, then the vector $\imath^{\mathsf{T}}\mathbb{L} : X \leftrightarrow \mathbf{1}$ represents Y as subset of X in the sense above. Clearly, it is also possible to generate the injective mapping

$$\mathsf{inj}(v) : Y \leftrightarrow X \qquad\qquad \mathsf{inj}(v)_{y,x} :\iff y = x \tag{1}$$

from a given vector $v : X \leftrightarrow \mathbf{1}$ representing a subset Y of X. We call $\mathsf{inj}(v)$ the *injective mapping generated* by the vector v.

As a third possibility to model subsets of a given set X we will use the set-theoretic *membership relation* defined by

$$\mathbb{M} : X \leftrightarrow 2^X \qquad\qquad \mathbb{M}_{x,Y} :\iff x \in Y\,, \tag{2}$$

where 2^X is the powerset of X. Using matrix terminology, a combination of injective mappings and membership relations lead to a *column-wise representation (enumeration)* of sets of subsets. More specifically, if the vector $v : 2^X \leftrightarrow \mathbf{1}$ represents a nonempty subset $\mathfrak{S}$ of 2^X in the sense above, then for all $x \in X$ and $Y \in \mathfrak{S}$ we get the equivalence of $x \in Y$ and $(\mathbb{M}\,\mathsf{inj}(v)^{\mathsf{T}})_{x,Y}$ due to (1) and (2). This means that the elements of $\mathfrak{S}$ are represented precisely by the columns of the relation $S := \mathbb{M}\,\mathsf{inj}(v)^{\mathsf{T}} : X \leftrightarrow \mathfrak{S}$. A further consequence is that $\overline{S^{\mathsf{T}}\overline{S}} : \mathfrak{S} \leftrightarrow \mathfrak{S}$ is the relation-algebraic specification of set inclusion on $\mathfrak{S}$, that is, for all $Y, Z \in \mathfrak{S}$ we have the relationship $(\overline{S^{\mathsf{T}}\overline{S}})_{Y,Z}$ if and only if $Y \subseteq Z$. Due to $\mathsf{inj}(\mathbb{L}) = \mathbb{I}$, hence, the *set inclusion relation* on the entire powerset 2^X is relation-algebraically given by

$$\mathbb{S} := \overline{\mathbb{M}^{\mathsf{T}}\overline{\mathbb{M}}} : 2^X \leftrightarrow 2^X\,. \tag{3}$$

Besides set inclusion relations, we will in this paper also apply *size-comparison relations* on powersets. Such a relation is defined by

$$\mathbb{C} : 2^X \leftrightarrow 2^X \qquad\qquad \mathbb{C}_{Y,Z} :\iff |Y| \leq |Z|\,. \tag{4}$$

Membership relations and size comparison relations allow for very space-efficient implementations using reduced ordered BDDs. Given a variable ordering such that the variables of the domain of a relation to be implemented via a BDD are tested prior to the variables of the range, the number of vertices of the BDD implementing $\mathbb{M} : X \leftrightarrow 2^X$ is bounded by $3 * |X| + 1$, and the number of vertices of the BDD implementing $\mathbb{C} : 2^X \leftrightarrow 2^X$ is exactly $2 + |X| * (|X| + 1)$; see [17] and [18] for the detailed algorithms. Unfortunately, the size of the BDD for implementing a set inclusion relation $\mathbb{S} : 2^X \leftrightarrow 2^X$ is exponential in $|X|$.

Given a direct product $X \times Y$ of two sets X and Y, there are two projection functions decomposing a pair $u = \langle u_1, u_2 \rangle$ into its first component u_1 and its second component u_2. For a relation-algebraic approach, instead to consider these functions it is more convenient to take the corresponding *projection relations* $\pi : X \times Y \leftrightarrow X$ and $\rho : X \times Y \leftrightarrow Y$ such that for all $u \in X \times Y$, $x \in X$, and $y \in Y$ we have $\pi_{u,x}$ if and only if $u_1 = x$, and $\rho_{u,y}$ if and only if $u_2 = y$. With projection relations we can relation-algebraically describe the well-known pairing operation of functional programming as follows: For relations $R : Z \leftrightarrow X$ and $S : Z \leftrightarrow Y$ we define their *pairing* (frequently also called their *fork* or *tupling*) by

$$[R, S] := R\pi^{\mathsf{T}} \cap S\rho^{\mathsf{T}} : Z \leftrightarrow X \times Y \,. \tag{5}$$

Using specification (5), for all $z \in Z$ and $u \in X \times Y$ a simple reflection shows that $[R, S]_{z,u}$ if and only if R_{z,u_1} and S_{z,u_2}.

3. Reduction of Feedback Vertex Sets to Cycle Enumeration

In the remainder, let $g = (V, R)$ be a *finite and directed graph* where V is its nonempty set of vertices and $R : V \leftrightarrow V$ its *adjacency relation*. The latter assumes each pair $\langle x, y \rangle \in V^2$ to be an arc in g if and only if $R_{x,y}$ holds.

A sequence $c := (x_0, \ldots, x_n)$ of $n > 0$ vertices is a *cycle* of g if each pair of consecutive vertices forms an arc, i.e., $R_{x_i,x_{i+1}}$ for $0 \le i \le n-1$, and $x_0 = x_n$. $\mathbf{V}(c) := \{x_i : 0 \le i \le n\}$ denotes the set of vertices of c. The cycle c is *elementary* if the vertices $x_0, \ldots, x_{n-1}$ are pairwise different. An arc $\langle x, y \rangle$ of g is a *chord* of c if $x, y \in \mathbf{V}(c)$ but $\langle x, y \rangle$ is not an arc of c. A cycle of g is called *chordless* if it does not possess a chord. Finally, a set $F \in 2^V$ is a *feedback vertex set* of g if for each cycle c of g there exists a vertex $x \in F$ with $x \in \mathbf{V}(c)$, i.e., $F \cap \mathbf{V}(c) \neq \emptyset$.

As the following proposition shows, feedback vertex sets can sufficiently be characterized by the vertex sets of elementary chordless cycles.

Proposition 3.1. Let $F \in 2^V$ be a set of vertices of g. Then F is a feedback vertex set of g if and only if $F \cap \mathbf{V}(c) \neq \emptyset$ for all elementary chordless cycles c of g.

Proof:
"$\Longrightarrow$" If F is a feedback vertex set of g, each cycle of g (and, hence, each elementary chordless one) contains a vertex of F.

"$\Longleftarrow$" Let c be a cycle of g. We consider the set

$$\{c' : c' \text{ cycle of } g \text{ and } \mathbf{V}(c') \subseteq \mathbf{V}(c)\}.$$

Then this set is nonempty and contains minimal elements with respect to the preorder on sequences induced by length comparison, since this preorder is Noetherian. Let c' be such a minimal element. Then obviously c' is an elementary chordless cycle of g. By assumption, $F \cap \mathbf{V}(c') \neq \emptyset$. With $\mathbf{V}(c') \subseteq \mathbf{V}(c)$ it follows that F contains a vertex from c. $\square$

In the following, we assume a column-wise representation of the subset

$$\mathfrak{C} := \{\mathbf{V}(c) : c \text{ elementary chordless cycle of } g\}$$

of the powerset 2^V to be given, i.e., a relation $C : V \leftrightarrow \mathfrak{C}$ such that $C_{x,S}$ if and only if S is the vertex set of an elementary chordless cycle of g and $x \in S$. The calculation of the relation C is postponed to Sections 4 and 5.

Our objective is to calculate a vector representing the set $\mathfrak{F} \subseteq 2^V$ of all feedback vertex sets of g. We assume an arbitrary set of vertices $F \in 2^V$ and calculate as follows, where c ranges over the elementary chordless cycles of the directed graph g and S ranges over the set $\mathfrak{C}$ of their vertex sets:

$$\begin{array}{ll}
\ F \text{ is a feedback vertex set of } g & \\
\iff \forall\, c : F \cap \mathbf{V}(c) \neq \emptyset & \text{Proposition 3.1} \\
\iff \forall\, c : \exists\, x : x \in F \wedge x \in \mathbf{V}(c) & \\
\iff \forall\, S : \exists\, x : \mathbb{M}_{x,F} \wedge C_{x,S} & \mathbb{M} : V \leftrightarrow 2^V \\
\iff \neg\exists\, S : \overline{\mathbb{M}^\mathsf{T} C}_{F,S} \wedge \mathbb{L}_S & \mathbb{L} : \mathfrak{C} \leftrightarrow \mathbf{1} \\
\iff \overline{\overline{\mathbb{M}^\mathsf{T} C}\, \mathbb{L}}_F\,. &
\end{array}$$

This derivation merely uses well-known correspondences between logical and relation-algebraic constructions. If we remove the subscript F from the last expression following the representation of sets through vectors as introduced in Section 2, we get $\overline{\overline{\mathbb{M}^\mathsf{T} C}\, \mathbb{L}} : 2^V \leftrightarrow \mathbf{1}$ as the relation-algebraic description of the vector representing the set $\mathfrak{F}$ of all feedback vertex sets of g.

From this vector we obtain a vector representation of the set $\mathfrak{F}_{min} \subseteq 2^V$ of all minimum feedback vertex sets of g as follows. The relational function

$$LeEl(Q, v) = v \cap \overline{\overline{Q}\, v}$$

computes vector representations of the least elements of a vector/set v using a preorder Q (see [19]). With the size comparison relation $\mathbb{C} : 2^V \leftrightarrow 2^V$ taken as the preorder Q and $\overline{\overline{\mathbb{M}^\mathsf{T} C}\, \mathbb{L}}$ as vector v, the vector

$$MinFvs(C) := LeEl(\mathbb{C}, \overline{\overline{\mathbb{M}^\mathsf{T} C}\, \mathbb{L}}) : 2^V \leftrightarrow \mathbf{1} \tag{6}$$

represents the desired set of minimum feedback vertex sets of g. If this vector is nonempty, the column-wise representation of the minimum feedback vertex sets $\mathfrak{F}_{min}$ is represented by the relation

$$\mathbb{M}\, \mathsf{inj}(MinFvs(C))^\mathsf{T} : V \leftrightarrow \mathfrak{F}_{min}$$

as an immediate consequence of (6) and the technique mentioned in Section 2.

4. Enumeration of Elementary Chordless Cycles

In this section, we are concerned with the computation of vertex sets of elementary chordless cycles in the directed graph $g = (V, R)$. The relation $C : V \leftrightarrow \mathfrak{C}$ as used in the previous chapter demands for its relation-algebraic specification to complete the program development. It will deliver a column-wise representation of the set $\mathfrak{C}$ of vertex sets $\mathbf{V}(c)$ of all elementary chordless cycles c of the graph g. For a better understanding of our approach, we start with a simple consideration on progressively infinite vertex sets of g. A set $S \in 2^V$ is *progressively infinite* if for each vertex $x \in S$ there exists a successor in S, i.e. a vertex $y \in S$ with $R_{x,y}$. The following property fundamentally relates these sets to vertex sets of elementary chordless cycles.

Proposition 4.1. Let $S \in 2^V$ be a set of vertices of g. Then S is the vertex set of an elementary chordless cycle of g if and only if S is a minimal (w.r.t. set-inclusion) nonempty and progressively infinite subset of V.

Proof:
"$\Longrightarrow$" Let $c = (x_0, \ldots, x_n)$ be an elementary chordless cycle of g and assume $S = \mathbf{V}(c)$. Then S is nonempty. Furthermore, for each $x_i \in S$, $0 \leq i \leq n-1$, there exists a successor $x_{i+1} \in S$ so that S is progressively infinite. S is also a minimal set with these properties: Assume S' to be a nonempty and progressively infinite proper subset of S. Then, S' is the vertex set $\mathbf{V}(c')$ of a cycle c' of g since V is finite. Now $\mathbf{V}(c') \subset \mathbf{V}(c)$ implies that c must contain a chord, which contradicts the assumption that c is chordless.

"$\Longleftarrow$" From S being nonempty and progressively infinite it follows that there exists a cycle c of g with $\mathbf{V}(c) = S$ due to the finiteness of V. Let c have the form $(x_0, \ldots, x_n)$. Then the cycle c is elementary: Assume $x_i = x_j$ for some $0 \leq i, j \leq n-1$ and w.l.o.g. $i < j$. This implies that $\{x_i, \ldots, x_j\}$ is a proper nonempty and progressively infinite subset of S, which contradicts S to be minimal. The cycle c is also chordless. To prove this by contradiction as well, we assume the arc $\langle x_i, x_j \rangle$ of g to be a chord of c. W.l.o.g. we again assume $i < j$. Then, obviously $\{x_0, \ldots, x_i, x_j, \ldots, x_n\}$ is a proper nonempty and progressively infinite subset of S, which again contradicts S to be minimal. $\square$

This proposition offers a first way to relation-algebraically specify the decisive relation $C : V \leftrightarrow \mathfrak{C}$ we are looking for. In a first step, we develop a vector representation for the nonempty and progressively infinite subsets of V. In a second step, we extract minimal elements therefrom. The relational function

$$Min(Q, v) = v \cap \overline{(Q^{\mathsf{T}} \cap \overline{\mathbb{I}})v}$$

computes the minimal elements of a vector/set v using a preorder Q (again, see [19]). We simply take the set inclusion relation $\mathbb{S} : 2^V \leftrightarrow 2^V$ as preorder Q and the vector representation of the progressively infinite subsets as vector v and yield a vector representation of the set $\mathfrak{C}$. In a third step, we transform this vector representation into a column-wise one following the technique introduced in Section 2 and eventually obtain the relation $C : V \leftrightarrow \mathfrak{C}$.

For the first step, we assume an arbitrary set $S \in 2^V$ to be given. Using some well-known correspondences between logical and relation-algebraic constructions again, we are able to develop a vector representing the set of all nonempty and progressively infinite sets as given below; in this calculation both x and y range over the set V:

$$
\begin{array}{lll}
& S \neq \emptyset \text{ and progressively infinite} & \\
\iff & \exists\, x : x \in S \wedge \forall\, x : x \in S \to \exists\, y : y \in S \wedge R_{x,y} & \\
\iff & \exists\, x : \mathbb{M}_{x,S} \wedge \forall\, x : \mathbb{M}_{x,S} \to \exists\, y : \mathbb{M}_{y,S} \wedge R_{x,y} & \mathbb{M} : V \leftrightarrow 2^V \\
\iff & \exists\, x : \mathbb{L}_{\perp,x} \wedge \mathbb{M}_{x,S} \wedge \forall\, x : \mathbb{M}_{x,S} \to (R\,\mathbb{M})_{x,S} & \mathbb{L} : \mathbf{1} \leftrightarrow V \\
\iff & (\mathbb{L}\mathbb{M})_{\perp,S} \wedge \neg\exists\, x : \mathbb{L}_{\perp,x} \wedge \mathbb{M}_{x,S} \wedge \overline{R\,\mathbb{M}}_{x,S} & \\
\iff & (\mathbb{L}\mathbb{M})^{\mathsf{T}}_{S} \wedge \overline{\mathbb{L}(\mathbb{M} \cap \overline{R\,\mathbb{M}})}_{\perp,S} & \\
\iff & ((\mathbb{L}\mathbb{M})^{\mathsf{T}} \cap \overline{\mathbb{L}(\mathbb{M} \cap \overline{R\,\mathbb{M}})}^{\mathsf{T}})_S\,. &
\end{array}
$$

The last expression of this derivation leads to

$$
\mathit{PrInf}(R) := (\mathbb{L}\mathbb{M})^{\mathsf{T}} \cap \overline{\mathbb{L}(\mathbb{M} \cap \overline{R\,\mathbb{M}})}^{\mathsf{T}} : 2^V \leftrightarrow \mathbf{1} \tag{7}
$$

as the vector representation of the set of all nonempty and progressively infinite subsets of V. The remaining steps for the column-wise representation of the set $\mathfrak{C}$ via the relation C are now straightforward and lead to the equality

$$
C = \mathbb{M}\,\mathsf{inj}(\mathit{Min}(\mathbb{S}, \mathit{PrInf}(R)))^{\mathsf{T}}\,, \tag{8}
$$

where $\mathbb{S} : 2^V \leftrightarrow 2^V$ is the set inclusion relation on the powerset 2^V.

Each of the relational specifications (6), (7), and (8) we have derived so far easily can be translated into the programming language of the RelView tool. Doing so, we obtain the following RelView-program for computing the vector representation of the set $\mathfrak{F}_m$ of all minimum feedback vertex sets of a directed graph $g = (V, R)$.

```
PrInf(R)
  DECL M, L
  BEG  M = epsi(O(R));
       L = L1n(R)
       RETURN (L * M)^ & -(L * (M & -(R * M)))^
  END.

MinFvs(C)
  DECL LeEl(Q,v) = v & -(-Q * v);
       M, CC
  BEG  M = epsi(O(C));
       CC = cardrel(O(C))
       RETURN LeEl(CC,-dom(-(M^ * C)))
  END.
```

```
Main(R)
  DECL Min(Q,v) = v & -((Q^ & -I(Q)) * v);
       M, SS, C
  BEG  M = epsi(O(R));
       SS = -(M^ * -M);
       C = M * inj(Min(SS,PrInf(R)))^
       RETURN MinFvs(C)
  END.
```

The bottleneck of this program is the set inclusion relation `SS` in `Main`. As already mentioned in Section 2, the size of the reduced ordered BDD of such a relation is exponential in the size of the base set. In applications this means that `Main` only works for adjacency relations of fairly small graphs with approximately 25 vertices. In the following section, we will demonstrate how to avoid a set inclusion relation in computing the decisive relation $C : V \leftrightarrow \mathfrak{C}$ by refining the hitherto approach.

5. A Refined Version which Avoids Set Inclusion

So far, we used the nonempty and progressively infinite subsets of $g = (V, R)$ to approximate the vertex sets of elementary chordless cycles. We computed their vector representation, $\mathit{PrInf}(R) : 2^V \leftrightarrow \mathbf{1}$, and provided in $C : V \leftrightarrow \mathfrak{C}$ the exact result by minimizing $\mathit{PrInf}(R)$ using a set inclusion relation, see Specification (8). Now, we add some further properties to obtain a much better approximation of the set $\mathfrak{C}$ and simultaneously avoid the set inclusion relation. The sets S of vertices of g we are concerned with now are considered to be:

- *Regressively infinite*: S is progressively infinite w.r.t. the transposed graph $g^{\mathsf{T}} = (V, R^{\mathsf{T}})$.
- *Free of branching vertices* (branch-free): There is no $x \in S$ such that $R_{x,y}$ and $R_{x,z}$ for any different vertices $y, z \in S$.
- *Free of joining vertices* (join-free): S is free of branching vertices w.r.t. the transposed graph $g^{\mathsf{T}} = (V, R^{\mathsf{T}})$.

From a column-wise representation $Q : V \leftrightarrow \mathfrak{S}$ of the set

$$\mathfrak{S} := \{S \in 2^V : S \neq \emptyset, \text{progr. and regr. infinite, branching and joining free}\}$$

we compute the relation $C : V \leftrightarrow \mathfrak{C}$ by removing from Q those columns not corresponding to minimal sets of the approximation $\mathfrak{S}$ of $\mathfrak{C}$. As we will demonstrate later, this can be done without using a set inclusion relation. The following proposition tightens Proposition 4.1 and is the crucial justification of the proposed refinement.

Proposition 5.1. Let $S \in 2^V$ be a vertex set. Then S is the vertex set of an elementary chordless cycle of g if and only if S is a minimal (w.r.t. set-inclusion) nonempty, progressively infinite, regressively infinite, branch-free, and join-free subset of V.

Proof:
"$\Longrightarrow$" Let $c = (x_0, \ldots, x_n)$ be an elementary chordless cycle of g and assume $S = \mathbf{V}(c)$. In the proof of Proposition 4.1 we already have shown that S is nonempty and progressively infinite. S is also regressively infinite since for each $x_i \in S$, $1 \leq i \leq n$, there exists a predecessor $x_{i-1} \in S$.

The absence of chords in c implies that S is free of branching vertices: A branching vertex $x \in S$ with two different successors $y, z \in S$ would lead to a chord of c since at least one of the arcs $\langle x, y\rangle$ or $\langle x, z\rangle$ is not an arc of c. In the same way it follows that S is free of joining vertices.

Additionally, S is also minimal. Assume S' to be a proper subset of S fulfilling the properties stated in the proposition. Then, the cycle c must contain a chord as already shown in the proof of Proposition 4.1, direction "$\Longrightarrow$".

"$\Longleftarrow$" The argument is the same as in the proof of Proposition 4.1, direction "$\Longleftarrow$": The finiteness of V implies that S is the vertex set of a cycle c and that c is both elementary and chordless since S is minimal. $\Box$

Further considerations lead to the observation that S is nonempty, progressively infinite, regressively infinite, branch-free, and join-free if and only if S is the union of vertex sets of disjoint elementary chordless cycles. This property is not used in any of our formal proofs. Its value is to motivate further refinement steps since it shows how the minima of such sets S look like. Hence, it remains to refine the derivation of regressively infinite, branch- and join-free sets, and to show how to extract the minima from their column-wise representation.

From the vector $\mathit{PrInf}(R) : 2^V \leftrightarrow \mathbf{1}$ as derived in (7), we immediately obtain the vector representation of all nonempty and regressively infinite sets through $\mathit{PrInf}(R^\mathsf{T}) : 2^V \leftrightarrow \mathbf{1}$. Equally, the vector representation of join-free sets can be reduced to the vector representation of branch-free sets. If the latter one is given by $\mathit{FreeBv}(R) : 2^V \leftrightarrow \mathbf{1}$, then obviously $\mathit{FreeBv}(R^\mathsf{T}) : 2^V \leftrightarrow \mathbf{1}$ represents the join-free sets. Thus, it remains to develop a relation-algebraic specification of $\mathit{FreeBv}(R)$. We divide this task into two steps: In the first step, we calculate a relation $B : V \leftrightarrow 2^V$ that relates a vertex $x \in V$ and a set $S \in 2^V$ if and only if x is a branching vertex of S. In the second step, based on B we calculate a vector of type $2^V \leftrightarrow \mathbf{1}$ representing all branch-free vertex sets.

In the derivation of the relation $B : V \leftrightarrow 2^V$, we will use y and z ranging over V, and $u = \langle u_1, u_2\rangle$ ranging over V^2. Furthermore, we will consecutively introduce a membership relation, the two projection relations on V^2, a diversity relation, an identity relation, and an universal relation:

$$
\begin{array}{lll}
 & x \text{ is a branching vertex of } S & \\
\Longleftrightarrow & x \in S \wedge \exists\, y : y \in S \wedge R_{x,y} \wedge \exists\, z : z \in S \wedge R_{x,z} \wedge y \neq z & \\
\Longleftrightarrow & x \in S \wedge \exists\, y, z : y \in S \wedge z \in S \wedge R_{x,y} \wedge R_{x,z} \wedge y \neq z & \\
\Longleftrightarrow & \mathbb{M}_{x,S} \wedge \exists\, u : u_1 \in S \wedge u_2 \in S \wedge R_{x,u_1} \wedge R_{x,u_2} \wedge u_1 \neq u_2 & \mathbb{M} : V \leftrightarrow 2^V \\
\Longleftrightarrow & \mathbb{M}_{x,S} \wedge \exists\, u : (\pi\mathbb{M})_{u,S} \wedge (\rho\mathbb{M})_{u,S} \wedge R_{x,u_1} \wedge R_{x,u_2} \wedge u_1 \neq u_2 & \pi, \rho \text{ project.} \\
\Longleftrightarrow & \mathbb{M}_{x,S} \wedge \exists\, u : [R,R]_{x,u} \wedge (\pi\mathbb{M} \cap \rho\mathbb{M})_{u,S} \wedge u_1 \neq u_2 & (5) \\
\Longleftrightarrow & \mathbb{M}_{x,S} \wedge \exists\, u : [R,R]_{x,u} \wedge (\pi\mathbb{M} \cap \rho\mathbb{M})_{u,S} \wedge (\pi\overline{\mathbb{I}}\rho)_{u,u} & \overline{\mathbb{I}} : V \leftrightarrow V \\
\Longleftrightarrow & \mathbb{M}_{x,S} \wedge \exists\, u : [R,R]_{x,u} \wedge (\pi\mathbb{M} \cap \rho\mathbb{M})_{u,S} \wedge ((\pi\overline{\mathbb{I}}\rho \cap \mathbb{I})\mathbb{L})_{u,S} & \mathbb{I} : V^2 \leftrightarrow v^2 \\
\Longleftrightarrow & (\mathbb{M} \cap [R,R](\pi\mathbb{M} \cap \rho\mathbb{M} \cap (\pi\overline{\mathbb{I}}\rho \cap \mathbb{I})\mathbb{L}))_{x,S} & \mathbb{L} : V^2 \leftrightarrow 2^V.
\end{array}
$$

This yields $\mathbb{M}\cap[R,R](\pi\mathbb{M}\cap\rho\mathbb{M}\cap(\pi\overline{\mathbb{I}}\rho\cap\mathbb{I})\mathbb{L})$ as the relation-algebraic specification of B. To calculate a vector representing all branch-free vertex sets, we assume an arbitrary set $S\in 2^V$, let x range over V, and calculate as follows:

$$
\begin{array}{lll}
 & S \text{ is free of branching vertices} & \\
\iff & \neg\exists\, x : x\in S \wedge B_{x,S} & \\
\iff & \neg\exists\, x : \mathbb{M}_{x,S} \wedge B_{x,S} & \mathbb{M}: V\leftrightarrow 2^V \\
\iff & \neg\exists\, x : \mathbb{L}_{\perp,x} \wedge \mathbb{M}_{x,S} \wedge B_{x,S} & \mathbb{L}: \mathbf{1}\leftrightarrow V \\
\iff & \neg(\mathbb{L}(\mathbb{M}\cap B))_{\perp,S} & \\
\iff & \overline{\mathbb{L}(\mathbb{M}\cap B)}^{\mathsf{T}}_{S}\,. &
\end{array}
$$

A consequence of this is the relation-algebraic specification $\overline{\mathbb{L}(\mathbb{M}\cap B)}^{\mathsf{T}} : 2^V\leftrightarrow\mathbf{1}$ of the branch-free vertex sets of g. Unfolding the relation B within this vector, the vector $\mathit{FreeBv}(R)$ we originally were interested in reads as follows:

$$\mathit{FreeBv}(R) = \overline{\mathbb{L}(\mathbb{M}\cap(\mathbb{M}\cap[R,R](\pi\mathbb{M}\cap\rho\mathbb{M}\cap(\pi\overline{\mathbb{I}}\rho\cap\mathbb{I})\mathbb{L})))}^{\mathsf{T}} : 2^V\leftrightarrow\mathbf{1}\,. \tag{9}$$

To obtain a relation $Q : V\leftrightarrow\mathfrak{S}$ representing in a column-wise manner the set $\mathfrak{S}$ of nonempty, progressively infinite, regressively infinite, branch-free, and join-free vertex sets, we combine the relation-algebraic specifications (7) and (9) with the membership relation $\mathbb{M} : V\leftrightarrow 2^V$ and yield

$$Q = \mathbb{M}\,\mathsf{inj}(\mathit{PrInf}(R)\cap\mathit{PrInf}(R^{\mathsf{T}})\cap\mathit{FreeBv}(R)\cap\mathit{FreeBv}(R^{\mathsf{T}}))^{\mathsf{T}}\,. \tag{10}$$

After removing from Q all columns not corresponding to minimal sets of $\mathfrak{S}$, Proposition 5.1 shows that the resulting relation coincides with the column-wise representation $C : V\leftrightarrow\mathfrak{C}$ of the set $\mathfrak{C}$. It takes two steps to remove such columns: First, we calculate a vector of type $\mathfrak{S}\leftrightarrow\mathbf{1}$ representing the minimal sets of $\mathfrak{S}$, i.e. $\mathfrak{C}$ as a subset of $\mathfrak{S}$. Second, we simply restrict the range $\mathfrak{S}$ of the relation Q to the set $\mathfrak{C}$. For the first step, we assume S to be an arbitrary set in $\mathfrak{S}$. Then we can proceed as follows, where T ranges over $\mathfrak{S}$ and x ranges over V:

$$
\begin{array}{lll}
 & S \text{ is a minimal set of } \mathfrak{S} & \\
\iff & \neg\exists\, T : S\neq T \wedge \forall\, x : x\in T\rightarrow x\in S & \\
\iff & \neg\exists\, T : S\neq T \wedge \forall\, x : Q_{x,T}\rightarrow Q_{x,S} & Q \text{ column-wisely repr. } \mathfrak{S} \\
\iff & \neg\exists\, T : \overline{\mathbb{I}}_{S,T} \wedge \neg\exists\, x : Q_{x,T}\wedge\overline{Q}_{x,S} \quad \overline{\mathbb{I}}:\mathfrak{S}\leftrightarrow\mathfrak{S} & \\
\iff & \neg\exists\, T : \overline{\mathbb{I}}_{S,T} \wedge \neg\exists\, x : \overline{Q}^{\mathsf{T}}_{S,x}\wedge Q_{x,T} & \\
\iff & \neg\exists\, T : \overline{\mathbb{I}}_{S,T} \wedge (\overline{\overline{Q}^{\mathsf{T}}Q})_{S,T} \wedge \mathbb{L}_T & \mathbb{L}:\mathfrak{S}\leftrightarrow\mathbf{1} \\
\iff & (\overline{(\overline{\mathbb{I}}\cap\overline{\overline{Q}^{\mathsf{T}}Q})\mathbb{L}})_S\,. &
\end{array}
$$

As a consequence, the subset $\mathfrak{C}$ of $\mathfrak{S}$ is represented by $\overline{(\overline{\mathbb{I}}\cap\overline{\overline{Q}^{\mathsf{T}}Q})\mathbb{L}} : \mathfrak{S}\leftrightarrow\mathbf{1}$. Now, the range restriction is trivially achieved by multiplying to Q from the right the transposed injective mapping generated by this vector. This results in:

$$C = Q\,\mathsf{inj}(\overline{(\overline{\mathbb{I}}\cap\overline{\overline{Q}^{\mathsf{T}}Q})\mathbb{L}})^{\mathsf{T}}\,. \tag{11}$$

In comparison to the specification provided in (8) above, in (11) we do not use the very expensive set inclusion relation $\mathbb{S} : 2^V \leftrightarrow 2^V$.

Similar to the relation-algebraic specifications (6), (7), and (8), the specifications shown in (9), (10), and (11) can immediately be translated into the programming language of RELVIEW. Here, we only present the code for (9) to show how the RELVIEW-language allows to introduce direct products, projection relations, and pairing. Specifications (10) and (11) merely require simple syntactic adoptions.

```
FreeBv(R)
  DECL Prod = PROD(R,R);
       M, pi, rho, I1, I2, L1, L2, A
  BEG  M = epsi(O(R));
       pi = p-1(Prod);
       rho = p-2(Prod);
       I1 = I(R);
       I2 = I(pi * pi^);
       L1 = L1n(R);
       L2 = L(pi * M^);
       A = pi * M & rho * M & (pi * -I1 * rho^ & I2) * L2
       RETURN -(L1 * (M & ([R,R] * A)))^
  END.
```

Of course, in this program the computations of `I2` and `L2` can be improved by replacing the compositions `pi * pi^` and `pi * M^` by compositions of corresponding empty relations (e.g., `O(pi) * O(pi)^` in the first case) since in a BDD-implementation of relations the generation and composition of empty relations require constant execution time (see [17] for details).

6. Some Further Improvements

The algorithms developed in the previous sections formed the basis for divers experiments to estimate the runtime behavior and the size of the graphs treatable. We used RELVIEW as the surrounding software system executed on a Sun-Fire 880 workstation running Solaris 9 at 750 MHz with 32 GByte of main memory. We constructed some specific graphs by hand to ensure reasonable data to experiment with. Additionally, we generated graphs randomly, a feature provided by RELVIEW. Randomization is controllable through the number of vertices and the density of a graph, i.e. its number of edges. Further, one can compel relations which are adjacency relations of directed graphs consisting of a single cycle or the disjoint union of cycles.

The algorithm as refined in (11) proved to be of much more performance than the original one as provided in (8). Whereas for small graphs the computation turned out to be very fast, for medium-sized graphs it took minutes or even more than an hour to obtain all minimum feedback vertex sets. However, we frequently achieved impressive results on medium-sized graphs. For example, in a sparse graph with 100 vertices, RELVIEW computed precisely 1 103 872 minimum feedback vertex sets within 2479 seconds. Each of them consisted of 11 vertices. The graph is depicted in Figure 1, and one of its minimum feedback vertex sets is emphasized in black.

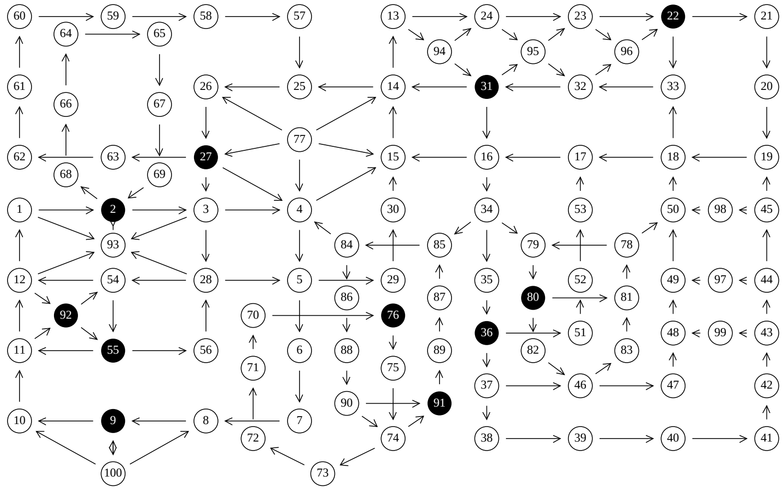

Figure 1. A graph and a minimum feedback vertex set

In case of a large set of solutions, of course, it is recommendable to compute only a single minimum feedback vertex set to reduce storage space. This can easily be accomplished utilizing the vector representation of all minimum vertex sets of a graph $g = (V, R)$. One simply has to select a point $p : 2^V \leftrightarrow \mathbf{1}$ from the vector representation; a suitable base-operation `point` is provided by RelView. A point is a nonempty and injective vector or, in terms of Boolean matrices, a Boolean column vector in which exactly one component is true. The composition $\mathbb{M}\,p$ of the membership relation $\mathbb{M} : V \leftrightarrow 2^V$ and p delivers a vector of type $V \leftrightarrow \mathbf{1}$ representing a single minimum feedback vertex set of g.

Some possibilities to enhance runtime or space consumption remain open. In the remainder, we briefly sketch two of them. We need to mention that these possibilities of course do no affect each and every graph. Our experiments have shown, however, that a large number of graphs are considerably affected.

First of all, it is reasonable to consider a subgraph $h = (W, S)$ of $g = (V, R)$ generated by the set W of vertices of g lying on a cycle. The set W can be represented by the vector $w := (R^+ \cap \mathbb{I})\mathbb{L} : V \leftrightarrow \mathbf{1}$, as a simple calculation proves. Consequently, we obtain $\mathsf{inj}(w)\, R\, \mathsf{inj}(w)^\mathsf{T} : W \leftrightarrow W$ as the adjacency relation S of h, and with the column-wise representation $M : W \leftrightarrow \mathfrak{F}_{min}$ of the minimum feedback vertex sets of h we yield $\mathsf{inj}(w)^\mathsf{T} M : V \leftrightarrow \mathfrak{F}_{min}$ as the column-wise representation of the minimum feedback vertex sets of g. Obviously, the less vertices of g are lying on a cycle the more effective this improvement is.

The second improvement is based on the following property: Let x be a “pipeline vertex” of g with exactly one predecessor $y \neq x$ and exactly one successor $z \neq x$ such that $y \neq z$. If we modify g by deleting the arcs $\langle y, x \rangle$ and $\langle x, z \rangle$ as well as by inserting the “bypass arc” $\langle y, z \rangle$, then each minimum feedback vertex set of the modification will also be a minimum feedback vertex set of the original graph. In case the number of elementary chordless cycles of g is not too large, this fact suggests to compute the relation $S := \bigcup_v R \cap vv^\mathsf{T} : V \leftrightarrow V$, where v ranges over the columns of $C : V \leftrightarrow \mathfrak{C}$. Using a graph-theoretic terminology, S exactly consists of the arcs of the elementary chordless cycles of g. Therefore, the minimum feedback vertex set problem for g can be reduced to the same problem with respect to the

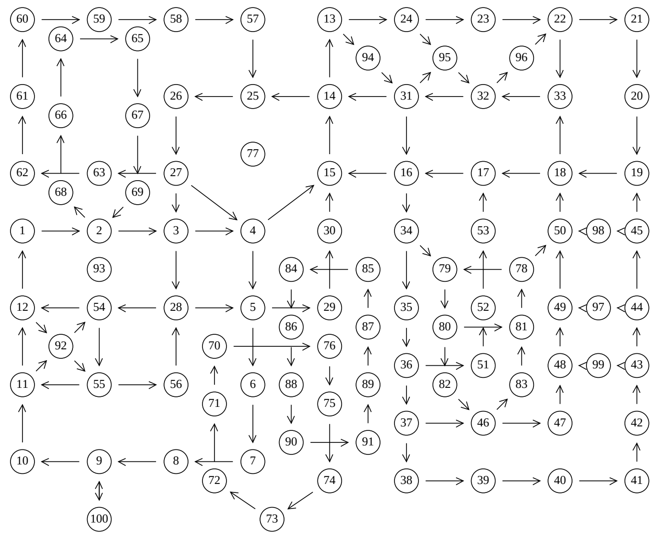

Figure 2. Subgraph consisting of the arcs of all elementary chordless cycles

subgraph $h = (V, S)$. Since we construct feedback vertex sets from elementary cycles, this subgraph may often contain a considerable number of pipeline vertices. Consequently, repeatedly bypassing them leads to a directed graph with many isolated vertices to which, eventually, the first improvement can successfully be applied.

We have combined both possibilities to compute a single minimum feedback vertex set of the graph shown in Figure 1. This graph possesses exactly 35 elementary chordless cycles, as a computation with RELVIEW showed. The subgraph consisting only of the arcs of these cycles is depicted in Figure 2. Repeatedly applying bypassing to this graph produces a graph with 58 isolated vertices, presented in Figure 3 together with the minimum feedback vertex set as shown in Figure 1. The combination of repeatedly bypassing and restriction to the subgraph generated by the set of vertices lying on a cycle reduces the computation of a single minimum feedback vertex set of the original graph shown in Figure 1 to the same problem for a graph with only 42 vertices. It took RELVIEW 18.45 seconds to reduce the graph, to compute a single minimum feedback vertex set of the 42-vertex graph, and to extend this vector to a vector representing a minimum feedback vertex set of the original graph.

The runtime behavior of algorithms solving the minimum feedback vertex set problem is often tested using the ISCAS89 or ITC99 benchmarks. These benchmarks offer sets of hardware circuits of various complexities, some with up to 20.000 gates. It is of course interesting to see how our algorithm performs on these benchmarks and how it compares to related approaches tested on them as well. As future work, we need to convert the benchmark data into a relational style for use with the RELVIEW tool such that these data are processable by our relational algorithm.

7. Conclusion

In this paper, we have developed a relational algorithm (a RELVIEW-program) for the exact enumeration of minimum feedback vertex sets. We started with the observation that these sets can be obtained by

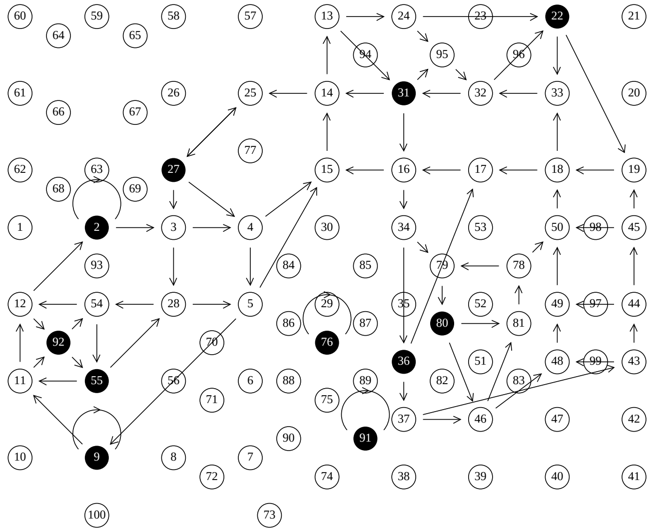

Figure 3. Result after repeatedly bypassing

enumerating the vertex sets of all elementary chordless cycles. To solve the latter problem, we first used minimal nonempty and progressively infinite vertex sets. The resulting program is merely applicable to fairly small graphs, even if relations are implemented by reduced ordered BDDs as in the RELVIEW tool. This arises from using a set inclusion relation, the reduced ordered BDD of which is exponential in the size of the base set. Therefore, we have refined our program in such a way that the set inclusion relation is avoided. Using the refined program, however, we are able to deliver minimum feedback vertex sets for medium-sized graphs in reasonable time. Such graphs frequently appear in practical applications such as analysing signal flow graphs.

At this place, a new application of minimum feedback vertex sets in coalition formation should also be mentioned. It allows to determine a stable government even in the case the dominance graph of feasible governments has no sources [6]. In practical applications, such dominance graphs are small or at most of medium size, and hence RELVIEW can effectively be used.

Aside from feedback vertex sets, calculating vertex sets of elementary chordless cycles is also beneficial in other domains. For example, Petri nets, as the most prominent representative of bipartite graphs in Computer Science, offer a nice playground for cycle calculation. State machines and marked graphs are common classes of Petri nets. They model systems built out of cycles [2]. On a visual analysis level, a designer wants to identify these building cycles, since they form the processes the system consists of. Identifying the cycles visually helps to understand the net topology and is thus vital for working with the underlying system. On the level of machine-based analysis, as for example proposed in [11], the enumeration of cycles is of particular interest for strongly connected conservative Petri nets. There, the support of a minimal semi-positive place invariant of the net is also the support of an elementary cycle [7]. A theory considering so called *handles* is elaborated in [9]. It relies on the absence of certain cyclic structures and then allows to deduce interesting net properties. For a brief explanation, let an elementary cycle $c = (x_0, \dots, p, \dots, t, \dots, x_n)$ be given with p and t any place and transition, respectively, of the net. Further, let p be a branching point and t be a joining point with successors of p and predecessors of t not in the vertex set of c. A path from p to t using no elements from the support of c is called a PT-handle (the dual is called TP-handle). Then, a strongly connected Petri net in which no elementary cycle has a PT- or TP-handle is structurally live, consistent, and conservative. With our algorithms, the detection of such structures is feasible, and they provide an alternative to costly proving the latter properties individually.

With a high-performance mechanization of relational algebra at hand, we were able to implement our algorithms within the RELVIEW system and within RELCLIPSE, an object-oriented version of RELVIEW implemented as an Eclipse Plug-In [13] using our KURE-Java library [12] providing this mechanization. The algorithms and ideas discussed in this paper can of course be realized in any other programming language using specialized data types instead of relations only. Nonetheless, we believe that our approach and the computation and visualization of the results by RelView delivers some new insights into the NP-hard problem of minimum feedback vertex set enumeration for medium-sized graphs.

References

[1] Bar-Yehuda, R., Geiger, D., Naor, J., Roth, R.: Approximation Algorithms for the Feedback Vertex Set Problem with Applications to Constraint Satisfaction and Bayesian Inference, *SIAM Journal on Computing*, (27), 1998, 942–959.

[2] Baumgarten, B.: *Petri-Netze: Grundlagen und Anwendungen*, BI-Wissenschafts-Verlag, 1990.

[3] Behnke, R., Berghammer, R., Meyer, E., Schneider, P.: RELVIEW – A system for calculating with relations and relational programming, *Proceedings of the 1st International Conference on Fundamental Approaches to Software Engineering* (E. Astesiano, Ed.), Lecture Notes in Computer Science 1382, Springer, 1996.

[4] Berghammer, R., Hoffmann, T.: Modeling sequences within the RELVIEW system, *Journal of Universal Computer Science*, **7**(2), 2001, 107–123.

[5] Berghammer, R., Leoniuk, B., Milanese, U.: Implementation of relational algebra using binary decision diagrams, *Proceedings of the 6th International Workshop on Relational Methods in Computer Science (RelMiCS6)* (H. de Swart, Ed.), Lecture Notes in Computer Science 2561, Springer, 2002.

[6] Berghammer, R., Rusinowska, A., de Swart H.: Graphs, RELVIEW, and Coalition Formation. In preparation.

[7] Desel, J., Esparza, J.: *Free Choice Petri Nets*, vol. 40 of *Cambridge Tracts in Theoretical Computer Science*, Cambridge University Press, 1995.

[8] Diaz, M., Richard, J., Courvoisier, M.: A Note on Minimal and Quasi-minimal Essential Sets in Complex Directed Graphs, *IEEE Transactions on Circuit Theory*, (19), 1972, 512–513.

[9] Esparza, J., Silva, M.: Circuits, Handles, Bridges and Nets, *Advances in Petri Nets* (G. Rozenberg, Ed.), Lecture Notes in Computer Science 483, Springer, 1990.

[10] Even, G., Naor, J., Schieber, B., Sudan, M.: Approximating Minimum Feedback Sets and Multi-cuts in Directed Graphs, *Proceedings of the 4th Conference on Integer Programming and Combinatorial Optimization* (E. Balas, J. Clausen, Eds.), Lecture Notes in Computer Science 920, Springer, 1995.

[11] Fronk, A.: Using relation algebra for the analysis of Petri nets in a CASE tool based approach, *2nd IEEE International Conference on Software Engineering and Formal Methods (SEFM), Beijing*, IEEE, 2004.

[12] Fronk, A., Pleumann, J., Schönlein, R., Szymanski, O.: KURE-Java, available via `http://ls10-www.cs.uni-dortmund.de/index.php?id=136`, 2005.

[13] Fronk, A., Schönlein, R.: RelClipse, available via `http://ls10-www.cs.uni-dortmund.de/index.php?id=137`.

[14] Hafna, V., Berman, P., Fujito, T.: A 2-Approximation Algorithm for the Undirected Feedback Vertex Set Problem, *SIAM Journal on Discrete Mathematics*, (17), 1999, 289–297.

[15] Karp, R.: Reducibility among combinatorial problems, *Proceedings of a Symposium on the Complexity and Computer Computation* (R. Miller, J. Thatcher, Eds.), Plenum Press, New York, 1972.

[16] Kevorkian, A.: The Principal Maxmin Matrix Transversal Strategy, *Journal on Mathematics and Operations Research*, (4), 1979.

[17] Leoniuk, B.: *ROBDD-based implementation of relational algebra with applications*, Ph.D. Thesis, Institut für Informatik und Praktische Mathematik, Universität Kiel, 2001, In German.

[18] Milanese, U.: *On the implementation of a ROBDD-based tool for the manipulation and visualization of relations*, Ph.D. Thesis, Institut für Informatik und Praktische Mathematik, Universität Kiel, 2003, In German.

[19] Schmidt, G., Ströhlein, T.: *Relations and graphs*, EATCS Monographs on Theoretical Computer Science, Spinger, 1993.

[20] Seshu, S., Reed, M.: *Linear Graphs and Electrical Networks*, Addison-Wesley, 1961.