St Andrews, September 5–9, 2006

Interpreting graphs in 0-simple semigroups with involution with applications to computational complexity and the finite basis problem

Mikhail Volkov (joint work with Marcel Jackson)

Ural State University, Ekaterinburg, Russia

Recap from Lecture 1

We are interested in:

Recap from Lecture 1

We are interested in:

- Computational complexity of the (Pseudo)variety Membership Problem (PMP)

Recap from Lecture 1

We are interested in:

- Computational complexity of the (Pseudo)variety Membership Problem (PMP)

and

- the Finite Basis Problem (FBP)

Recap from Lecture 1

- Computational complexity of (some instances of) PMP is of interest and "practical" importance

Recap from Lecture 1

- Computational complexity of (some instances of) PMP is of interest and “practical” importance $\Rightarrow$ try to find upper and lower bounds for it!

Recap from Lecture 1

- Computational complexity of (some instances of) PMP is of interest and “practical” importance $\Rightarrow$ try to find upper and lower bounds for it!
- A positive answer to some instance of FBP gives an efficient algorithm for the corresponding instance of PMP

Recap from Lecture 1

- Computational complexity of (some instances of) PMP is of interest and “practical” importance $\Rightarrow$ try to find upper and lower bounds for it!
- A positive answer to some instance of FBP gives an efficient algorithm for the corresponding instance of PMP $\Rightarrow$ systematically investigate FBP!

Recap from Lecture 1

- Computational complexity of (some instances of) PMP is of interest and “practical” importance $\Rightarrow$ try to find upper and lower bounds for it!
- A positive answer to some instance of FBP gives an efficient algorithm for the corresponding instance of PMP $\Rightarrow$ systematically investigate FBP!
- For many important instances, the answer to FBP is negative

Recap from Lecture 1

- Computational complexity of (some instances of) PMP is of interest and “practical” importance $\Rightarrow$ try to find upper and lower bounds for it!
- A positive answer to some instance of FBP gives an efficient algorithm for the corresponding instance of PMP $\Rightarrow$ systematically investigate FBP!
- For many important instances, the answer to FBP is negative $\Rightarrow$ develop some “equational” approach to PMP for nonfinitely based pseudovarieties!

Interpreting Graphs

We will consider a *graph* to be a structure $G := \langle V; \sim \rangle$, where V is a set and $\sim \subseteq V \times V$ is a binary relation.

Interpreting Graphs

We will consider a *graph* to be a structure $G := \langle V; \sim \rangle$, where V is a set and $\sim \subseteq V \times V$ is a binary relation. In other words, we consider all graphs to be directed, and do not allow multiple edges.

Interpreting Graphs

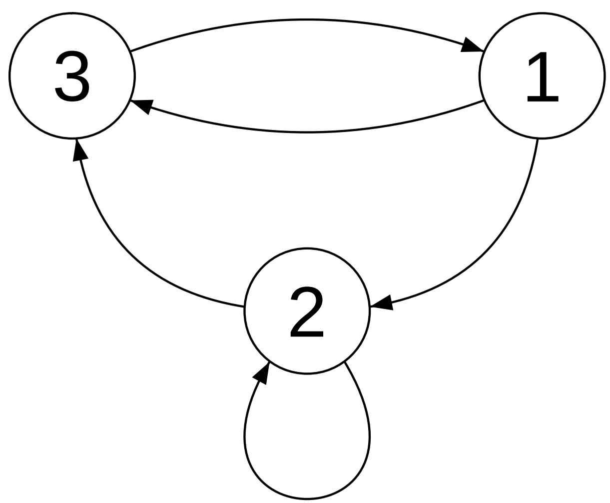
We will consider a *graph* to be a structure $G := \langle V; \sim \rangle$, where V is a set and $\sim \subseteq V \times V$ is a binary relation. In other words, we consider all graphs to be directed, and do not allow multiple edges.

The adjacency matrix P_G of a graph $G = \langle V; \sim \rangle$ is given a $V \times V$-matrix by $P_G(i, j) = 1$ if $i \sim j$ and 0 otherwise.

Interpreting Graphs

We will consider a *graph* to be a structure $G := \langle V; \sim \rangle$, where V is a set and $\sim \subseteq V \times V$ is a binary relation. In other words, we consider all graphs to be directed, and do not allow multiple edges.

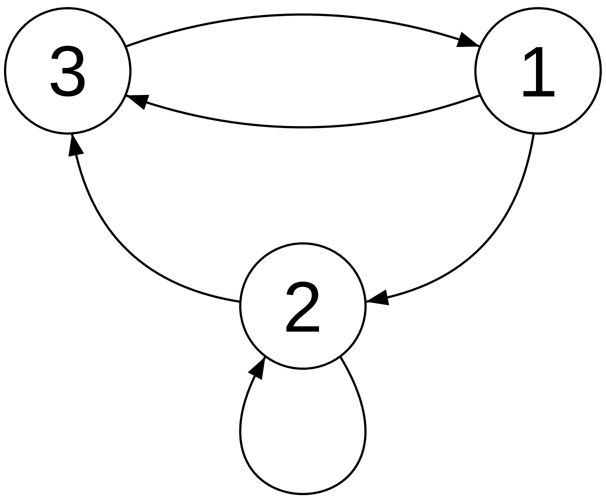
The adjacency matrix P_G of a graph $G = \langle V; \sim \rangle$ is given a $V \times V$-matrix by $P_G(i, j) = 1$ if $i \sim j$ and 0 otherwise.



Interpreting Graphs

We will consider a *graph* to be a structure $G := \langle V; \sim \rangle$, where V is a set and $\sim \subseteq V \times V$ is a binary relation. In other words, we consider all graphs to be directed, and do not allow multiple edges.

The adjacency matrix P_G of a graph $G = \langle V; \sim \rangle$ is given a $V \times V$-matrix by $P_G(i,j) = 1$ if $i \sim j$ and 0 otherwise.



$$\begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

Interpreting Graphs

To each graph G we assign the Rees matrix semigroup over the trivial group using the adjacency matrix P_G as a sandwich matrix. We denote this semigroup by $A(G)$ and call it the adjacency semigroup of G.

Interpreting Graphs

To each graph G we assign the Rees matrix semigroup over the trivial group using the adjacency matrix P_G as a sandwich matrix. We denote this semigroup by $A(G)$ and call it the adjacency semigroup of G.
In other words, $A(G)$ is defined on the set $(V \times V) \cup \{\mathbf{0}\}$ and the multiplication rule is

$$(i,j)(k,\ell) = \begin{cases} (i,\ell) & \text{if } j \sim k, \\ \mathbf{0} & \text{if } j \nsim k; \end{cases}$$

$$a\mathbf{0} = \mathbf{0}a = \mathbf{0} \text{ for all } a \in A(G).$$

Interpreting Graphs

Can some algebraic properties of $A(G)$ capture combinatorial properties of G?

Interpreting Graphs

Can some algebraic properties of $A(G)$ capture combinatorial properties of G?

Example: $A(G)$ is completely 0-simple iff every vertex of G has nonzero indegree and nonzero outdegree.

Interpreting Graphs

Can some algebraic properties of $A(G)$ capture combinatorial properties of G?

Example: $A(G)$ is completely 0-simple iff every vertex of G has nonzero indegree and nonzero outdegree.

In general, however, connections between $A(G)$ and G seem to be far too weak in order to be useful.

Can some algebraic properties of $A(G)$ capture combinatorial properties of G?

Example: $A(G)$ is completely 0-simple iff every vertex of G has nonzero indegree and nonzero outdegree.

In general, however, connections between $A(G)$ and G seem to be far too weak in order to be useful.

New idea: to equip $A(G)$ with an additional unary operation (reversion):

$$(i,j)' = (j,i), \quad 0' = 0$$

Interpreting Graphs

Now many important properties of M are captured!

Interpreting Graphs

Now many important properties of M are captured!

- $x \sim x$ (reflexivity of G) is equivalent to $A(G) \models xx'x = x$;

Interpreting Graphs

Now many important properties of M are captured!

- $x \sim x$ (reflexivity of G) is equivalent to $A(G) \models xx'x = x$;
- $x \nsim x$ (anti-reflexivity of G) is equivalent to $A(G) \models xx'z = zxx' = xx'$ (these laws mean $xx' = 0$);

Interpreting Graphs

Now many important properties of M are captured!

- $x \sim x$ (reflexivity of G) is equivalent to $A(G) \models xx'x = x$;
- $x \nsim x$ (anti-reflexivity of G) is equivalent to $A(G) \models xx'z = zxx' = xx'$ (these laws mean $xx' = 0$);
- $x \sim y \to y \sim x$ (symmetry of G) is equivalent to $A(G) \models (xy)' = y'x'$;

Interpreting Graphs

Now many important properties of M are captured!

- $x \sim x$ (reflexivity of G) is equivalent to $A(G) \models xx'x = x$;
- $x \nsim x$ (anti-reflexivity of G) is equivalent to $A(G) \models xx'z = zxx' = xx'$ (these laws mean $xx' = 0$);
- $x \sim y \to y \sim x$ (symmetry of G) is equivalent to $A(G) \models (xy)' = y'x'$;
- G is an anti-chain (satisfies $x \sim y \to x = y$ and $x \sim x$) if and only if $A(G)$ is a combinatorial Brandt semigroup.

Universal Horn Classes

We encounter identities on the semigroup side.

Universal Horn Classes

We encounter identities on the semigroup side. What is the syntactic nature of conditions appearing at the graph side?

Universal Horn Classes

We encounter identities on the semigroup side. What is the syntactic nature of conditions appearing at the graph side?
A universal Horn sentence is a sentence of one of the following two forms:

$$(\forall x_1 \forall x_2 \ldots) \left(\left(\mathop{\&}_{1 \le i \le n} \Phi_i \right) \to \Phi_0 \right),$$

$$(\forall x_1 \forall x_2 \ldots) \left(\bigvee_{0 \le i \le n} \neg \Phi_i \right)$$

where the Φ_i are of the form $x_j \sim x_k$ or $x_j = x_k$.

Universal Horn Classes

Classes of graphs defined by universal Horn sentences are called **universal Horn classes** (uH-classes).

Universal Horn Classes

Classes of graphs defined by universal Horn sentences are called **universal Horn classes** (uH-classes). Examples:

Universal Horn Classes

Classes of graphs defined by universal Horn sentences are called **universal Horn classes** (uH-classes). Examples:

- Preorders (reflexifity + transitivity)

Universal Horn Classes

Classes of graphs defined by universal Horn sentences are called **universal Horn classes** (uH-classes). Examples:

- Preorders (reflexifity + transitivity) which include
  - equivalence relations (add symmetry)
  - partial orders (add anti-symmetry)
  - anti-chains (add both symmetry and anti-symmetry)
  - complete looped graphs (add $x \sim y$)

Universal Horn Classes

Classes of graphs defined by universal Horn sentences are called **universal Horn classes** (uH-classes). Examples:

- Preorders (reflexifity + transitivity) which include
  - equivalence relations (add symmetry)
  - partial orders (add anti-symmetry)
  - anti-chains (add both symmetry and anti-symmetry)
  - complete looped graphs (add $x \sim y$)
- Simple graphs (anti-reflexifity + symmetry)

Universal Horn Classes

Classes of graphs defined by universal Horn sentences are called **universal Horn classes** (uH-classes). Examples:

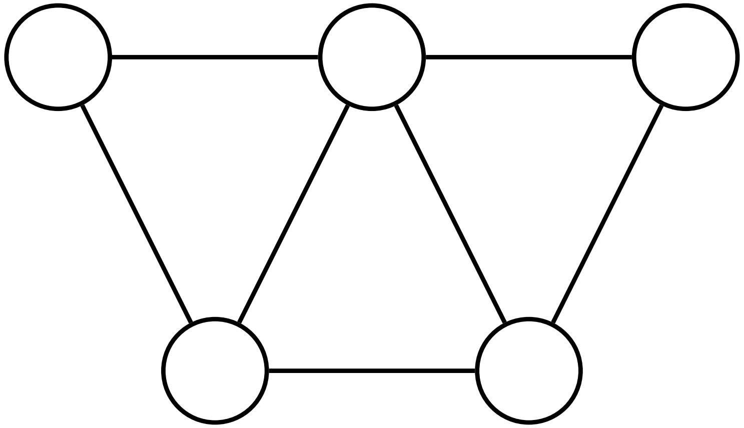
- Preorders (reflexifity + transitivity) which include
  - equivalence relations (add symmetry)
  - partial orders (add anti-symmetry)
  - anti-chains (add both symmetry and anti-symmetry)
  - complete looped graphs (add $x \sim y$)
- Simple graphs (anti-reflexifity + symmetry)
- n-colorable simple graphs (no finite axiomatization)

Universal Horn Classes

More about the last example.

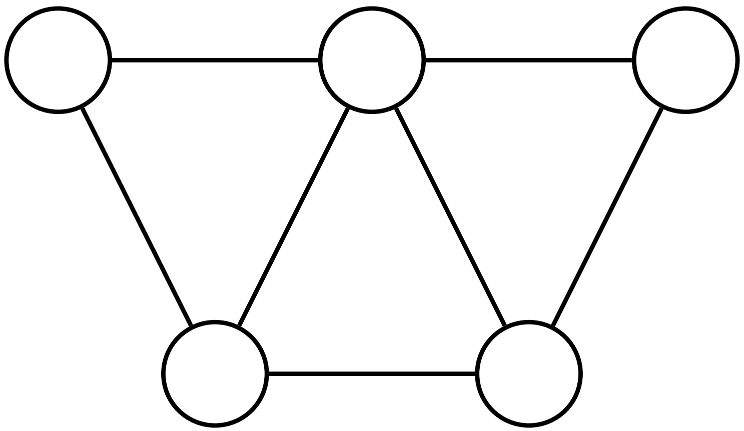
Universal Horn Classes

More about the last example.



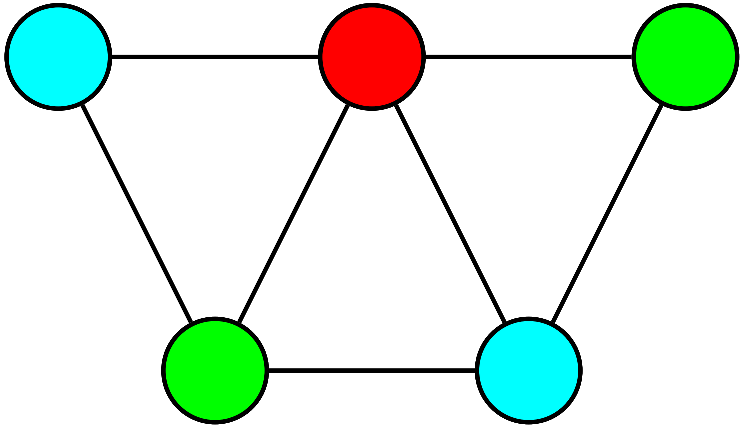
Universal Horn Classes

More about the last example.



The above graph C_3 is 3-colorable but not 2-colorable.

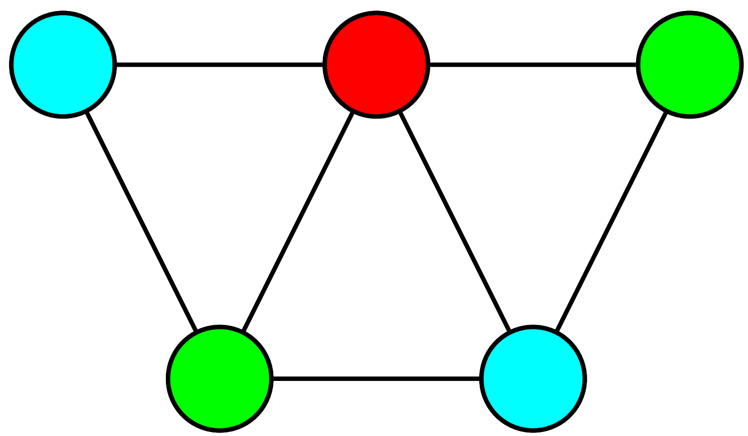
More about the last example.



The above graph C_3 is 3-colorable but not 2-colorable.

Universal Horn Classes

More about the last example.



The above graph C_3 is 3-colorable but not 2-colorable. In fact, it is a generator of minimum size for the uH-class of all 3-colorable graphs (Nešetřil and Pultr, 1978).

Main results

Theorem 1. The assignment $G \mapsto A(G)$ induces an injective join-preserving map from the lattice of all universal Horn classes of graphs to the lattice of subvarieties of the variety generated by all (unary) adjacency semigroups.

Main results

Theorem 1. The assignment $G \mapsto A(G)$ induces an injective join-preserving map from the lattice of all universal Horn classes of graphs to the lattice of subvarieties of the variety generated by all (unary) adjacency semigroups.

Theorem 2. When restricted to reflexive graphs, the map preserves meets as well as joins, and is “nearly” surjective in the sense that the lattice of subvarieties of the variety generated by adjacency semigroups satisfying $xx'x = x$ is isomorphic to the lattice obtained from the lattice of all universal Horn classes of reflexive graphs by inserting just one new element.

An application to PMP

Theorem 1 interpreted another way: membership of a unary Rees matrix semigroup $\mathcal{M}_0'[\mathbf{1}, P]$ in the variety generated by another such unary semigroup $\mathcal{M}_0'[\mathbf{1}, Q]$ is equivalent to the membership of the graph whose adjacency matrix is P in the uH-class of the graph whose adjacency matrix is Q.

An application to PMP

Theorem 1 interpreted another way: membership of a unary Rees matrix semigroup $\mathcal{M}_0'[\mathbf{1}, P]$ in the variety generated by another such unary semigroup $\mathcal{M}_0'[\mathbf{1}, Q]$ is equivalent to the membership of the graph whose adjacency matrix is P in the uH-class of the graph whose adjacency matrix is Q.

This allows us to produce numerous examples of finite unary Rees matrix semigroups M such that recognizing membership to the (pseudo)variety generated by M is computationally difficult.

An application to PMP

Theorem 1 interpreted another way: membership of a unary Rees matrix semigroup $\mathcal{M}_0'[\mathbf{1}, P]$ in the variety generated by another such unary semigroup $\mathcal{M}_0'[\mathbf{1}, Q]$ is equivalent to the membership of the graph whose adjacency matrix is P in the uH-class of the graph whose adjacency matrix is Q.

This allows us to produce numerous examples of finite unary Rees matrix semigroups M such that recognizing membership to the (pseudo)variety generated by M is computationally difficult. In order to formulate the results precisely, we need a 5-minute tour in Complexity Theory.

The Classes P *and* NP

Arthur, a normal man

The Classes P *and* NP

Arthur, a normal man

Merlin, a superman

The Classes P *and* NP

A problem is in P if Arthur can solve it in polynomial time (of the size of its input).

The Classes P *and* NP

A problem is in P if Arthur can solve it in polynomial time (of the size of its input). Example: 2-COLOR is in P since Arthur can check in polynomial time whether or not all simple cycles of a given graph are of even length.

The Classes P *and* NP

A problem is in $\mathbf{P}$ if Arthur can solve it in polynomial time (of the size of its input). Example: 2-COLOR is in $\mathbf{P}$ since Arthur can check in polynomial time whether or not all simple cycles of a given graph are of even length.

A problem is in $\mathbf{NP}$ if, *whenever the answer to its instance is YES*, Merlin can convince Arthur that the answer is YES in polynomial time (of the size of the input).

The Classes $\mathbf{P}$ *and* $\mathbf{NP}$

A problem is in $\mathbf{P}$ if Arthur can solve it in polynomial time (of the size of its input). Example: 2-COLOR is in $\mathbf{P}$ since Arthur can check in polynomial time whether or not all simple cycles of a given graph are of even length.

A problem is in $\mathbf{NP}$ if, *whenever the answer to its instance is YES*, Merlin can convince Arthur that the answer is YES in polynomial time (of the size of the input). Example: 3-COLOR is in $\mathbf{NP}$ since, given a 3-colorable graph, Merlin can exhibit its 3-coloring, and Arthur can check in polynomial time that this coloring is correct.

The Classes P *and* NP

Clearly $\mathbf{P} \subseteq \mathbf{NP}$. Is the inclusion strict? In other words, is $\mathbf{P} \neq \mathbf{NP}$?

The Classes P *and* NP

Clearly $\mathbf{P} \subseteq \mathbf{NP}$. Is the inclusion strict? In other words, is $\mathbf{P} \neq \mathbf{NP}$? This is a VERY BIG PROBLEM which is worth $1000000

The Classes P *and* NP

Clearly $\mathbf{P} \subseteq \mathbf{NP}$. Is the inclusion strict? In other words, is $\mathbf{P} \neq \mathbf{NP}$? This is a VERY BIG PROBLEM which is worth $1000000 (before tax).

The Classes P *and* NP

Clearly $\mathbf{P} \subseteq \mathbf{NP}$. Is the inclusion strict? In other words, is $\mathbf{P} \neq \mathbf{NP}$? This is a VERY BIG PROBLEM which is worth \$1000000 (before tax). According to the present paradigm, we assume that $\mathbf{P} \neq \mathbf{NP}$.

The Classes $\mathbf{P}$ *and* $\mathbf{NP}$

Clearly $\mathbf{P} \subseteq \mathbf{NP}$. Is the inclusion strict? In other words, is $\mathbf{P} \neq \mathbf{NP}$? This is a VERY BIG PROBLEM which is worth \$1000000 (before tax). According to the present paradigm, we assume that $\mathbf{P} \neq \mathbf{NP}$.

An $\mathbf{NP}$-*hard* problem is a problem to which any problem from $\mathbf{NP}$ can be reduced in polynomial time.

Clearly $\mathbf{P} \subseteq \mathbf{NP}$. Is the inclusion strict? In other words, is $\mathbf{P} \neq \mathbf{NP}$? This is a VERY BIG PROBLEM which is worth $1000000 (before tax). According to the present paradigm, we assume that $\mathbf{P} \neq \mathbf{NP}$.

An $\mathbf{NP}$-*hard* problem is a problem to which any problem from $\mathbf{NP}$ can be reduced in polynomial time.

An $\mathbf{NP}$-*complete* problem is a problem in $\mathbf{NP}$ that at the same time is $\mathbf{NP}$-hard.

The Classes $\mathbf{P}$ *and* $\mathbf{NP}$

Clearly $\mathbf{P} \subseteq \mathbf{NP}$. Is the inclusion strict? In other words, is $\mathbf{P} \neq \mathbf{NP}$? This is a VERY BIG PROBLEM which is worth $1000000 (before tax). According to the present paradigm, we assume that $\mathbf{P} \neq \mathbf{NP}$.

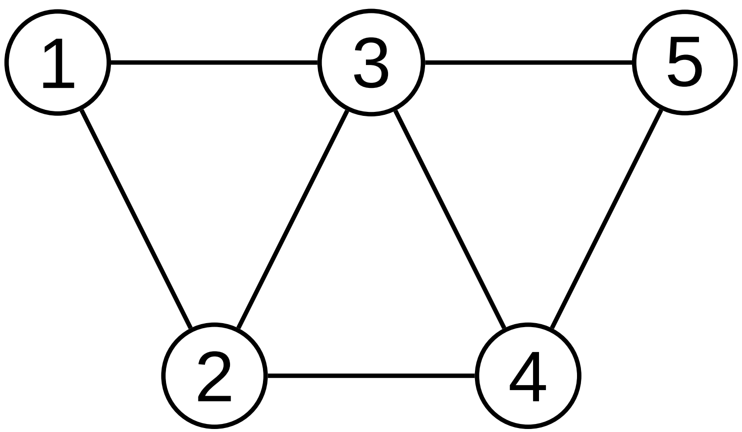
An $\mathbf{NP}$-*hard* problem is a problem to which any problem from $\mathbf{NP}$ can be reduced in polynomial time.

An $\mathbf{NP}$-*complete* problem is a problem in $\mathbf{NP}$ that at the same time is $\mathbf{NP}$-hard.

Example: 3-COLOR is $\mathbf{NP}$-complete (Levin, 1973).

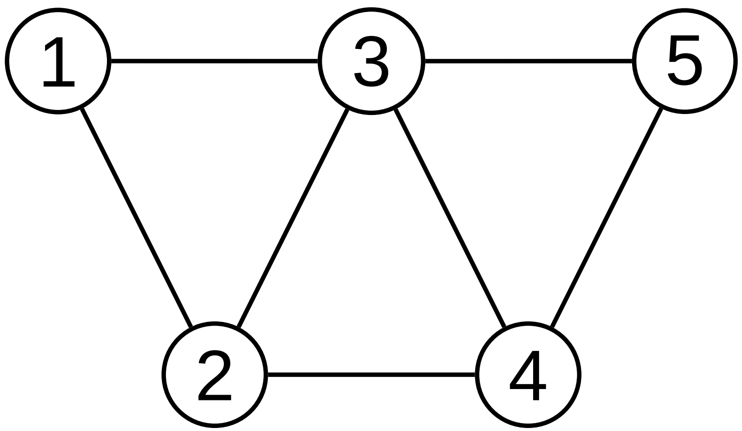
An application to PMP

Recall that C_3 is a five element graph generating a uH-class equal to the class of all 3-colorable graphs:



An application to PMP

Recall that C_3 is a five element graph generating a uH-class equal to the class of all 3-colorable graphs:

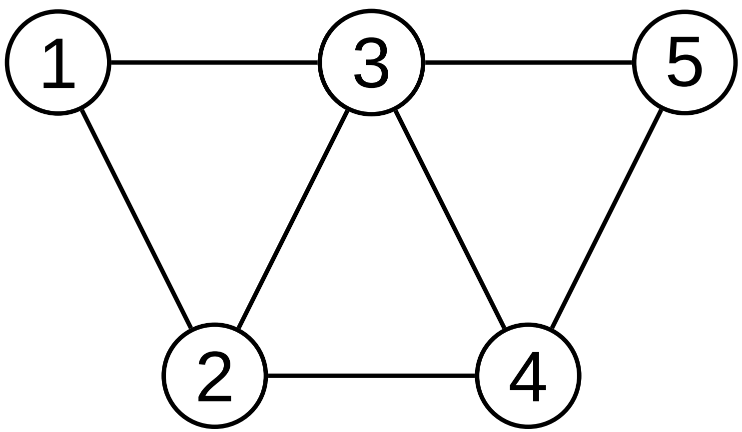


$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Corollary. The membership problem for the pseudovariety generated by $A(C_3)$ is $\mathbf{NP}$-hard.

An application to PMP

Recall that C_3 is a five element graph generating a uH-class equal to the class of all 3-colorable graphs:



$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Corollary. The membership problem for the pseudovariety generated by $A(C_3)$ is $\mathbf{NP}$-hard.

(One can't claim that the problem is $\mathbf{NP}$-complete because it is not yet clear that it belongs to the class $\mathbf{NP}$.)

Further Applications

- $A(C_3)$ is nonfinitely based;

Further Applications

- $A(C_3)$ is nonfinitely based;
- the identity checking problem for $A(C_3)$ is co-$\mathbf{NP}$-complete.

Further Applications

- $A(C_3)$ is nonfinitely based;
- the identity checking problem for $A(C_3)$ is co-$\mathbf{NP}$-complete.

A problem is in co-$\mathbf{NP}$ if, *whenever the answer to its instance is NO*, Merlin can convince Arthur that the answer is NO in polynomial time (of the size of the input).

Further Applications

- $A(C_3)$ is nonfinitely based;
- the identity checking problem for $A(C_3)$ is co-$\mathbf{NP}$-complete.

A problem is in co-$\mathbf{NP}$ if, *whenever the answer to its instance is NO*, Merlin can convince Arthur that the answer is NO in polynomial time (of the size of the input).

In the plain semigroup setting similar examples were produced by Jackson and McKenzie, 2006. Their smallest semigroup with analogous properties consists of 55 elements.