Algorithmic aspects of finite semigroup theory, a tutorial

Jean-Éric Pin

LIAFA, CNRS and University Paris 7

6, 8, 9 September 2006, St Andrews
Outline

(1) What this tutorial is about?
(2) Complexity
(3) Presentation and Cayley graphs
(4) Green’s relations
(5) Idempotents, weak inverses and inverses
(6) Blocks
(7) Syntactic preorder
(8) Other computations

**Warning.** In this tutorial, all semigroups are **finite**.
Part I

What this tutorial is about?

The aim of this tutorial is to present some algorithms to compute finite semigroups.

Programming issues, like data structures, implementation or interface will not be addressed, but most algorithms are implemented in the C-programme semigroupe.

This tutorial is addressed to mathematicians, not to computer scientists. For this reason, I will remind a few basic algorithms, when needed.
Computing finite semigroups

Several questions should be answered:

• How is the semigroup given? (transformation semigroup, semigroup of matrices over some (semi)ring, finite presentation, . . . )

• What does one wish to compute?

• What is the complexity of the algorithms?
How is the semigroup $S$ given?

I assume that $S$ is a subsemigroup of a larger semigroup $U$ (the universe), like:

- the semigroup of all transformations on a set $E$,
- the semigroup of $n \times n$-Boolean matrices,
- the semigroup of $n \times n$-matrices with entries in $\mathbb{Z}$,
- a set of words, with a multiplication defined on it,
- etc.

Then $S$ is given as the subsemigroup of $U$ generated by some set $A$ of generators.
Complexity means **worth case complexity**. The **average complexity** would be too difficult to define: what would be a random semigroup?

Complexity is usually measured in terms of the following **parameters**:

- $|S|$, the number of elements of the semigroup,
- $|A|$, the number of generators.

Occasionally, **other parameters** might be used: number of idempotents, number of $D$-classes, etc.
Two types of complexity

**Space complexity** measures the amount of computer memory required to run the algorithm.

**Time complexity** measures the time spent by the computer to run the algorithm.

Both space and time complexity are measured as a function of the size $n$ of the input data, but are expressed in $O(f(n))$ notation. This makes the notion robust and machine independent.
Meaning of complexity

If an $O(n)$-time algorithm takes 0.1 second on an input of size $10^5$, it will spend roughly 1 second on an input of size $10^6$ and 10 seconds on an input of size $10^7$.

The $O(f(n))$ notation also explains why the cost of elementary operations is irrelevant. Even if your computer is twice faster as mine, computing 1000 additions will take 1000 times as much as a single addition on both computers.

Complexity allows to predict effectiveness and is surprisingly precise in practice.
Usual complexities

<table>
<thead>
<tr>
<th>( \log_2 n )</th>
<th>3</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>16</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>10</td>
<td>( 10^2 )</td>
<td>( 10^3 )</td>
<td>( 10^4 )</td>
<td>( 10^5 )</td>
<td>( 10^6 )</td>
</tr>
<tr>
<td>( n \log_2 n )</td>
<td>33</td>
<td>664</td>
<td>( 10^4 )</td>
<td>1.3 ( 10^5 )</td>
<td>1.6 ( 10^6 )</td>
<td>2 ( 10^7 )</td>
</tr>
<tr>
<td>( n^2 )</td>
<td>( 10^2 )</td>
<td>( 10^4 )</td>
<td>( 10^6 )</td>
<td>( 10^8 )</td>
<td>( 10^{10} )</td>
<td>( 10^{12} )</td>
</tr>
<tr>
<td>( n^3 )</td>
<td>( 10^3 )</td>
<td>( 10^6 )</td>
<td>( 10^9 )</td>
<td>( 10^{12} )</td>
<td>( 10^{12} )</td>
<td>( 10^{12} )</td>
</tr>
<tr>
<td>( 2^n )</td>
<td>( 10^3 )</td>
<td>( 10^{30} )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
</tbody>
</table>

**linear** (time) algorithm = \( O(n) \)-time algorithm

**quadratic** (time) algorithm = \( O(n^2) \)-time algorithm
Practical issues about complexity

In practice, one can run within one minute:
- linear algorithms for data of size $\leq 2 \cdot 10^7$
- $O(n \log n)$-time algorithms for data of size $\leq 10^6$
- quadratic algorithms for data of size $\leq 10^4$

For linear time algorithms, space complexity is often the main issue and most of the time is spent on memory allocation.
Practical issues about semigroup algorithms

Two functions are given:

- one for computing the **product** of an **element** of the universe by a **generator**.
- one for testing the **equality** of two elements of the universe.

Time complexity is usually measured by the **number of accesses** to these two functions.

The **multiplication table** can be computed in **quadratic** time and space. Therefore all algorithms in \( O(|S|^k) \) with \( k > 2 \) may assume that the multiplication table is known.
Part III

Presentation and Cayley graphs

Data: A universe and an ordered set of generators.

Output: A presentation of the semigroup by generators and relations, a confluent rewriting system for this presentation and the right and left Cayley graphs of the semigroup.
An example (input data in red)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>* 1</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>* a</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>* b</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>* c</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>* ab</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>* bc</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>* ca</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

\[
\begin{array}{cccc}
  \ast & 1 & 2 & 3 \\
  a & b & c \\
  ab & d & e \\
  bc & f & g \\
  ca & h & i \\
\end{array}
\]

\[
\begin{array}{ccc}
  a & \ast & ab \\
  b & \ast & bc \\
  c & \ast & ca \\
\end{array}
\]

\[
\begin{array}{c}
  aa \rightarrow a \\
  ac \rightarrow a \\
  ba \rightarrow a \\
  bb \rightarrow b \\
  cb \rightarrow bc \\
  cc \rightarrow c \\
  abc \rightarrow ab \\
  bca \rightarrow ca \\
  cab \rightarrow bc \\
\end{array}
\]
Right Cayley graph: edges of the form $u \xrightarrow{a} ua$
Left Cayley graph: edges of the form $u \xrightarrow{a} au$
Lexicographic order ($\leq_{\text{lex}}$): total order used in a dictionary.

Shortlex order ($\leq$): words are ordered by length and words of equal length are ordered by $\leq_{\text{lex}}$.

If $a < b$, then $ababb \leq_{\text{lex}} abba$ but $abba < ababb$.

For each rule $u \rightarrow v$, one has $v < u$.

\begin{align*}
aa & \rightarrow a & ac & \rightarrow a & ba & \rightarrow a \\
bb & \rightarrow b & cb & \rightarrow bc & cc & \rightarrow c \\
abc & \rightarrow ab & bca & \rightarrow ca & cab & \rightarrow bc
\end{align*}
Properties of the shortlex order

Proposition

Let $u, v \in A^*$ and let $a, b \in A$.

(1) If $u < v$, then $au < av$ and $ua < va$.

(2) If $ua \leq vb$, then $u \leq v$.

Therefore $\leq$ is a stable order on $A^*$: if $u \leq v$, then $xuy \leq xvy$ for all $x, y \in A^*$. Further, it is a well order.
Properties of the rewriting system

Let $L$ be the set of left hand sides of the rules.

- All the rules are of the form $u \rightarrow v$ with $v < u$.
- The set $L$ is unavoidable (any sufficiently long word contains a factor in $L$ or, equivalently, the set $A^* \setminus A^*LA^*$ is finite),
- No word of $L$ has a proper factor in $L$.
- The set of proper factors of words in $L$ is the set of reduced words.
- The rewriting system is confluent.
- It depends on the order on $A$. 
Computation of the presentation

Rules:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>c</td>
<td>-</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
Computation of the presentation

<table>
<thead>
<tr>
<th>1</th>
<th>1 2 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>2 2 2</td>
</tr>
<tr>
<td>b</td>
<td>1 3 3</td>
</tr>
<tr>
<td>c</td>
<td>- 2 3</td>
</tr>
</tbody>
</table>

Rules:

\[ aa \rightarrow a \]
Computation of the presentation

Rules:
\[ aa \rightarrow a \]
Computation of the presentation

Rules:

- $aa \rightarrow a$
- $ac \rightarrow a$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>a</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>c</td>
<td>-</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>ab</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Computation of the presentation

Rules:

- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$

<table>
<thead>
<tr>
<th>1</th>
<th>1 2 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>2 2 2</td>
</tr>
<tr>
<td>$b$</td>
<td>1 3 3</td>
</tr>
<tr>
<td>$c$</td>
<td>- 2 3</td>
</tr>
<tr>
<td>$ab$</td>
<td>3 3 3</td>
</tr>
</tbody>
</table>
Computation of the presentation

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>a</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>c</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>ab</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Rules:

- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$
- $bb \rightarrow b$
Computation of the presentation

Rules:

\[ aa \rightarrow a \]
\[ ac \rightarrow a \]
\[ ba \rightarrow a \]
\[ bb \rightarrow b \]
Computation of the presentation

Rules:
- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$
- $bb \rightarrow b$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$b$</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$c$</td>
<td>-</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$ab$</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$bc$</td>
<td>-</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$ca$</td>
<td>-</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Computation of the presentation

**Rules:**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$b$</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$c$</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$ab$</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$bc$</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$ca$</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

**Rules:**

- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$
- $bb \rightarrow b$
- $cb \rightarrow bc$
Computation of the presentation

Rules:
- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$
- $bb \rightarrow b$
- $cb \rightarrow bc$
- $cc \rightarrow c$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$b$</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$c$</td>
<td>-</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$ab$</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$bc$</td>
<td>-</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$ca$</td>
<td>-</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Computation of the presentation

Rules:

- \( aa \rightarrow a \)
- \( ac \rightarrow a \)
- \( ba \rightarrow a \)
- \( bb \rightarrow b \)
- \( cb \rightarrow bc \)
- \( cc \rightarrow c \)
- \( abc \rightarrow ab \)
Computation of the presentation

Rules:
- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$
- $bb \rightarrow b$
- $bc \rightarrow bc$
- $cb \rightarrow bc$
- $cc \rightarrow c$
- $abc \rightarrow ab$
- $bca \rightarrow ca$
Computation of the presentation

**Rules:**

- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$
- $bb \rightarrow b$
- $cb \rightarrow bc$
- $cc \rightarrow c$
- $abc \rightarrow ab$
- $bca \rightarrow ca$
- $cab \rightarrow bc$

**Table:**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$b$</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$c$</td>
<td>-</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$ab$</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$bc$</td>
<td>-</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>$ca$</td>
<td>-</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Computation of the presentation

Rules:

- $aa \rightarrow a$
- $ac \rightarrow a$
- $ba \rightarrow a$
- $bb \rightarrow b$
- $cb \rightarrow bc$
- $cc \rightarrow c$
- $abc \rightarrow ab$
- $bca \rightarrow ca$
- $cab \rightarrow bc$

The end!
Main algorithm

Convention: \(a, b, c, \ldots\) will be \textit{generic} letters and \(p, q, r, \ldots\) will be \textit{generic} words.

We maintain two tables. One contains the list of \textbf{reduced} words with the corresponding elements of \(U\). The other one contains the list of \textbf{relations}.

\[
\begin{array}{c|ccc}
1 & 1 & 2 & 3 \\
\hline
\text{a} & 2 & 2 & 2 \\
\text{b} & 1 & 3 & 3 \\
\text{c} & - & 2 & 3 \\
\text{ab} & 3 & 3 & 3 \\
\text{bc} & - & 3 & 3 \\
\text{ca} & - & 2 & 2 \\
\end{array}
\]

\[
\begin{align*}
\text{aa} & \rightarrow a \\
\text{ac} & \rightarrow a \\
\text{ba} & \rightarrow a \\
\text{bb} & \rightarrow b \\
\text{bc} & \rightarrow bc \\
\text{cc} & \rightarrow c \\
\text{abc} & \rightarrow ab \\
\text{bca} & \rightarrow ca \\
\text{cab} & \rightarrow bc \\
\end{align*}
\]
Main loops

For each length $n$,

Computation of the right Cayley graph

For each word $u$ of length $n$ in the table

For $a$ ranging from the first to the last letter
handle $ua$ [next slide]

Computation of the left Cayley graph

For each word $u$ of length $n$ in the table

For $a$ ranging from the first to the last letter
reduce $au$
For each length $n$,

  For each word $u$ of length $n$ in the table

  For $a$ ranging from the first to the last letter

  try to reduce the word $ua$; [next slide]

  if it can be reduced with the current rules

  switch to the next letter

  else

  compute the associated element of $U$;

  if it corresponds to some word $v$

    add the relation $ua \rightarrow v$

  else

    add this new element to the table;
Reduction of $ua$

Put $u = bs$. If $sa \to r$, then $ua = bsa \to br$.

- If $r = 1$, then $ua \to b$.
- If $r \neq 1$, put $r = tc$. Then $r = tc < sa$ implies $t \leq s$.

- If $t = s$, then $c < a$, $ua \to br = btc = bsc = uc$ and the reduction of $uc$ has been done, since $c < a$.

- If $t < s$, then $|t| \leq |s| < |u|$ and thus the reduction of $bt$ has been done. Assume that $bt \to v$. Then $v \leq bt < bs = u$. Then $ua \to br = btc \to vc$, and the reduction of $vc$ has been done, since $v < u$. 
Left Cayley graph

If $u = pb$ and $a \in A$, then $au = tb$, where $t = ap$.

Since $|t| \leq |u|$, $tb$ has been handled at this stage.
Complexity

Theorem

The number of accesses to the function computing the product in the universe is equal to $|S| + |R| - |A|$ (where $R$ is the set of rules).

Result for $T_n$ ($|A| = 3$).

| $n$ | $|S|$ | Nb of Rules | Nb of Calls |
|-----|------|-------------|-------------|
| 3   | 27   | 13          | 37          |
| 4   | 256  | 83          | 336         |
| 5   | 3,125| 751         | 3,873       |
| 6   | 46,656| 7935       | 54,588      |
## Benchmarks (in seconds)

| Name   | $|A|$ | $|S|$       | $S$    | $D$    | $H$  |
|--------|-----|-----------|--------|--------|------|
| S10    | 2   | 3,628,800 | 11.02  | 15.00  | 0.01 |
| T7     | 3   | 823,543   | 2.95   | 2.38   | 0.74 |
| F7     | 4   | 2,097,152 | 8.87   | 7.36   | 0.64 |
| I8     | 3   | 1,441,729 | 5.44   | 5.63   | 0.42 |
| RB4    | 4   | 63,904    | 0.37   | 0.10   | 0.02 |
| FC13   | 13  | 5,200,300 | 35.63  | 22.61  | 1.29 |
| FIC12  | 12  | 2,704,156 | 20.57  | 11.53  | 0.71 |
| POPI12 | 2   | 16,224,937| 56.00  | 66.73  | 5.90 |
| Tr6    | 21  | 2,097,152 | 33.09  | 10.43  | 1.12 |
| U7     | 21  | 2,097,152 | 40.93  | 9.28   | 1.04 |
Part IV

Green’s relations and blocks

\[
\begin{array}{c}
\ast 1 \\
\ast b & \ast c \\
\ast a & \ast ab & \ast ca & \ast bc
\end{array}
\]
The $\mathcal{R}$-classes are the strongly connected components of the right Cayley graph.
The $\mathcal{L}$-classes are the strongly connected components of the left Cayley graph.
$\mathcal{J}$-classes (loops and labels are omitted)

The $\mathcal{J}$-classes are the strongly connected components of the union of the right and left Cayley graphs.
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Starting from vertex 1
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Reaching the rightmost neighbour, 2
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Vertex 2 has no neighbour, back to 1
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Reaching the rightmost neighbour, 3
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Reaching the rightmost neighbour, 6
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Vertex 6 has no neighbour, back to 3
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Reaching the rightmost neighbour, 5
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Vertex 5 has no free neighbour, back to 3
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Vertex 3 has no free neighbour, back to 1
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Reaching the rightmost neighbour, 4
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

Vertex 4 has no free neighbour, back to 1
Depth first search (DFS)

A directed graph is given by its set of vertices and for each vertex, the ordered set of its successors.

The end.
Tarjan’s algorithm (1)
Tree, Backward, Cross and Forward edges
The tie $t(x)$ of a vertex $x$ is the least $y$ such that there is a path (possibly empty) from $x$ to $y$ containing at most one backward or cross edge.

**Theorem**

If $t(x) = x$ and if $t(y) < y$ for every descendant $y$ of $x$, then the set of descendants of $x$ is a **strongly connected component** (SCC).

**Algorithm** : Find the deepest SCC, remove it and look for the next one.
Computing the ties
Computing the ties

1

B

2

B

3

C

4

5

C

6

7

C

8

9

C

10

11
Computing the ties
Computing the ties
Computing the ties
Computing the ties

Graph diagram with nodes labeled 1 to 11 and edges labeled with letters B, C, and numbers 1.
Computing the ties

![Graph with nodes and edges labeled with 'B', 'C', and '1'. The nodes are connected by arrows indicating direction and distance.]
Computing the ties
Computing the ties
Computing the ties
Computing the ties
Computing the ties
Computing the ties
Computing the ties
Computing the ties
Computing the ties
Computing the ties
Tarjan computes the SCC on $O(|E| + |V|)$ time.
Computing Green’s relations

To obtain the $R$-classes, the $L$-classes and the $D$-classes, it suffices to compute the strongly connected of the right Cayley graph, the left Cayley graph and the union $G$ of the two Cayley graphs.

It can be done in time linear in the size (number of vertices + number of edges) of the Cayley graphs, that is, in $O(|A||S|)$. Space complexity is also linear.

The depth-first search of $G$ also gives the “deepest” strongly connected component, that is, the minimal ideal.
\( \mathcal{H} \)-classes

At this stage, one gets the table of all \( \mathcal{D} \)-classes, \( \mathcal{R} \)-classes and \( \mathcal{L} \)-classes:

<table>
<thead>
<tr>
<th>Element</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{D} )-class</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>( \mathcal{R} )-class</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>( \mathcal{L} )-class</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

How to find the \( \mathcal{H} \)-classes?
Step one: sorting the elements by $\mathcal{R}$-class number

<table>
<thead>
<tr>
<th>Element</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{R}$-class</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sorted</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>4</td>
<td>7</td>
<td>10</td>
<td>13</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>11</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

This can be done in linear time by first counting the number of elements in each $\mathcal{R}$-class.

<table>
<thead>
<tr>
<th>$\mathcal{R}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

$\mathcal{L}$-class

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

$\mathcal{H}$-class
Step two: browsing the \( \mathcal{L} \)-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{L} )-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>( \mathcal{R} )-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

\( \mathcal{L} \)-class

<table>
<thead>
<tr>
<th>( \mathcal{L} )-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{H} )-class</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

New \( \mathcal{R} \)-class

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{H} )-class</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

$\mathcal{L}$-class

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$\mathcal{H}$-class

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
</tr>
</thead>
</table>

Sorted | 6 | 9 | 12 | 4 | 7 | 10 | 13 | 3 | 5 | 8 | 11 | 1 | 2 |
| $\mathcal{H}$-class | 1 | 2 |

LIAFA, CNRS and University Paris VII
### Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:
**Step two: browsing the \( \mathcal{L} \)-class table**

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{L} )-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>( \mathcal{R} )-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

\[
\begin{array}{ccccccccccc}
\mathcal{L}-class & 1 & 2 & 3 & 4 & 5 & 6 \\
\mathcal{H}-class & 3 & 2 & 0 & 1 & 4 & 0 \\
\end{array}
\]

**New \( \mathcal{R} \)-class**

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{H} )-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{array}{cccc}
\mathcal{L} & 3 & 5 & 2 & 4 \\
\mathcal{H} & 1 & 2 & 3 & 4 \\
\end{array}
\]
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The table is sorted and the classes are represented as follows:

- $\mathcal{L}$-class:
  - Sorted: 6, 9, 12, 4, 7, 10, 13, 3, 5, 8, 11, 1, 2
  - Table:
    - $\mathcal{L}$-class:
      - Sorted: 4, 2, 1, 5, 3, 5, 3, 5, 3, 5, 3, 6, 6
      - $\mathcal{R}$-class:
        - 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4

- $\mathcal{H}$-class:
  - Sorted: 6, 9, 12, 4, 7, 10, 13, 3, 5, 8, 11, 1, 2
  - Table:
    - $\mathcal{L}$-class:
      - 1, 2, 3, 4, 5, 6
    - $\mathcal{H}$-class:
      - 3, 2, 5, 1, 4, 0

The diagram represents the structure of the classes with numbers indicating the order and positions.
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

```
+---+---+---+---
|   | 6 |
+---+---+---+---
| 4 |
+---+---+---+---
| 3 | 5 |
+---+---+---+---
| 2 | 5 | 4 |
+---+---+---+---
| 3 |
+---+---+---+---
| 1 | 3 | 2 | 1 |
```
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

New $\mathcal{R}$-class

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

| $\mathcal{L}$-class | 1  | 2  | 3  | 4  | 5  | 6  |
| $\mathcal{H}$-class  | 3  | 2  | 7  | 1  | 6  | 0  |

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[\begin{array}{cccc}
6 & & & \\
4 & & & \\
3 & 5 & & \\
2 & 5 & 4 & \\
3 & 7 & 6 & \\
1 & 3 & 2 & 1
\end{array}\]
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Diagrams
### Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram of the L-class table]
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

New $\mathcal{R}$-class

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>

\[ \begin{array}{ccc} 
6 & & \\
4 & 8 & \\
3 & 5 & \\
2 & 5 & 4 \\
3 & 7 & 6 \\
1 & 3 & 2 & 1 \\
\end{array} \]
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\mathcal{L}$-class</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

\[
\begin{array}{ccccccccccccc}
6 & & & & & & & & & & & & \\
4 & & & & & & & & & & & & \\
3 & & & & & & & & & & & & \\
2 & & & & & & & & & & & & \\
3 & & & & & & & & & & & & \\
1 & & & & & & & & & & & & \\
\end{array}
\]
Step two: browsing the $\mathcal{L}$-class table

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{L}$-class</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\mathcal{R}$-class</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

| $\mathcal{L}$-class | 1 | 2 | 3 | 4 | 5 | 6 |
| $\mathcal{H}$-class | 3 | 2 | 7 | 1 | 6 | 8 |

<table>
<thead>
<tr>
<th>Sorted</th>
<th>6</th>
<th>9</th>
<th>12</th>
<th>4</th>
<th>7</th>
<th>10</th>
<th>13</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>11</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{H}$-class</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>7</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>
Computing the idempotents can be done by testing whether $x = x^2$ in the universe, or by using the rewriting system.
Possible improvements

• If the $\mathcal{H}$-classes of the minimal ideal are trivial (which is easy to test), then all elements of the minimal ideal are idempotent.

• If there is only one $\mathcal{H}$-class, the semigroup is a group, and $1$ is the unique idempotent.

• If one makes use of the rewriting system, one reads the word $x$ from the node $x$ on the right Cayley graph, but one can stop if one leaves the $\mathcal{R}$-class of $x$. 
Recall that $t$ is a weak inverse of $s$ if $tst = t$. If, further, $sts = s$, then $t$ is an inverse of $s$.

**Algorithm:** for each $t \in S$, start a depth first search of $G$ from $t$. Note that each visited $s$ is $J$-below $t$. One checks whether:

1. $st$ is idempotent,
2. $st \mathcal{R} s$
3. $s \mathcal{J} t$

Then $s$ is a weak inverse [inverse] of $t$ iff (1–2) [(1–3)] are satisfied.
Part VI

Blocks
# Blocks

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

LIAFA, CNRS and University Paris VII
Blocks
Blocks
Blocks
Blocks
Blocks

\[
\begin{array}{ccc}
\ast & \circ & \circ \\
\ast & \ast & \ast \\
\ast & \ast & \ast \\
\ast & \ast & \ast \\
\end{array}
\]
Blocks
Blocks
Blocks
Blocks
Blocks
Blocks
Blocks

\[
\begin{array}{|c|c|c|}
\hline
\ast & \circ & \circ \\
\hline
\circ & \ast & \circ \\
\hline
\ast & \circ & \ast \\
\ast & \circ & \circ \\
\hline
\end{array}
\]
Blocks

```
*   O   O
    O   *
  *   O   *   *
    O   *   *   O
        *   *
    *   *
    *   *
    *   *
  *   *
    *
```
# Blocks

```plaintext
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>o</td>
<td>o</td>
</tr>
<tr>
<td>o</td>
<td>o</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>o</td>
<td>*</td>
</tr>
<tr>
<td>o</td>
<td>*</td>
<td>o</td>
</tr>
</tbody>
</table>
```

```plaintext
<table>
<thead>
<tr>
<th></th>
<th>*</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td>*</td>
<td></td>
</tr>
</tbody>
</table>
```
## Blocks

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>o</td>
<td>o</td>
<td>o</td>
</tr>
<tr>
<td>o</td>
<td>o</td>
<td>o</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>o</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>o</td>
<td>*</td>
<td>*</td>
<td>o</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>o</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Blocks

\[
\begin{array}{c|c|c}
\ast & \circ & \circ \\
\circ & \circ & \ast \\
\ast & \circ & \ast \\
\circ & \ast & \circ \\
\ast & \ast \\
\ast & \ast \\
\ast & \ast & \ast \\
\ast & \ast \\
\end{array}
\]
Blocks
This computation amounts to finding the connected components of a certain graph.
Computation of the blocks
Computation of the blocks (2)

One could use again Tarjan’s algorithm, but using the "Union-Find" algorithm is a bit simpler.
The vertices $(1, 3), (6, 3), (1, 6), (2, 3), (5, 4), (7, 8), (7, 5)$ and $(2, 5)$ are connected. Are 4 and 6 connected?
Representing a forest by an array

In computer science, trees are represented top-down... The root of each tree is its own parent.

<table>
<thead>
<tr>
<th>Vertex</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>5</td>
<td>10</td>
<td>7</td>
<td>2</td>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((1, 3)\)
Union-find (1)

Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((1, 3)\)
Union-find (1)

Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((6, 3)\)
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((6, 3)\)
Rule: to add $(x, y)$, find the root $x'$ [$y'$] of the tree containing $x$ [$y$]. If $x' \neq y'$, add the edge $(x', y')$.

Adding $(1, 6)$
Union-find (1)

Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((2, 3)\)
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((2, 3)\)
Union-find (1)

Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((5, 4)\)
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((5, 4)\)
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((7, 8)\)
Union-find (1)

Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((7, 8)\)
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((7, 5)\)
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((7, 5)\)
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).
Rule: to add \((x, y)\), find the root \(x' [y']\) of the tree containing \(x [y]\). If \(x' \neq y'\), add the edge \((x', y')\).

Adding \((2, 5)\)
Union-find, union by size

When merging two trees, attach the root of the tree with fewer nodes to the root of the tree with more nodes.
Union-find, union by size

When merging two trees, attach the root of the tree with fewer nodes to the root of the tree with more nodes.

Adding \((1, 3)\)
Union-find, path compression

Do twice the search for the root. The second time, attach all nodes on the path to the root.
Tarjan and van Leeuwen have shown that performing \( m \) Finds and \( n - 1 \) Unions, with \( m \geq n \), can be done in \( O(n + m\alpha(m, n)) \) where \( \alpha \) is a kind of inverse of the Ackermann’s function.

This function is so slow that for \( n < 2^{65536} \) and \( m \geq n \), \( \alpha(m, n) \leq 2 \). Thus, the algorithm is linear in practice.

In particular, the blocks can be computed in (quasi)-linear time in the number of idempotents.
Summary on complexity

The computation of the elements, the right and left Cayley graphs, the Green’s relations, the blocks, the idempotents, the minimal ideal, can be done in time $O(|A||S|)$. 
## Benchmarks (in seconds)

<table>
<thead>
<tr>
<th>Name</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$</td>
<td>A</td>
<td>$</td>
<td>$</td>
<td>S</td>
</tr>
<tr>
<td>S10</td>
<td>2</td>
<td>3,628,800</td>
<td>11.02</td>
<td>15.00</td>
<td>0.01</td>
</tr>
<tr>
<td>T7</td>
<td>3</td>
<td>823,543</td>
<td>2.95</td>
<td>2.38</td>
<td>0.74</td>
</tr>
<tr>
<td>F7</td>
<td>4</td>
<td>2,097,152</td>
<td>8.87</td>
<td>7.36</td>
<td>0.64</td>
</tr>
<tr>
<td>I8</td>
<td>3</td>
<td>1,441,729</td>
<td>5.44</td>
<td>5.63</td>
<td>0.42</td>
</tr>
<tr>
<td>RB4</td>
<td>4</td>
<td>63,904</td>
<td>0.37</td>
<td>0.10</td>
<td>0.02</td>
</tr>
<tr>
<td>FC13</td>
<td>13</td>
<td>5,200,300</td>
<td>35.63</td>
<td>22.61</td>
<td>1.29</td>
</tr>
<tr>
<td>FIC12</td>
<td>12</td>
<td>2,704,156</td>
<td>20.57</td>
<td>11.53</td>
<td>0.71</td>
</tr>
<tr>
<td>POPI12</td>
<td>2</td>
<td>16,224,937</td>
<td>56.00</td>
<td>66.73</td>
<td>5.90</td>
</tr>
<tr>
<td>Tr6</td>
<td>21</td>
<td>2,097,152</td>
<td>33.09</td>
<td>10.43</td>
<td>1.12</td>
</tr>
<tr>
<td>U7</td>
<td>21</td>
<td>2,097,152</td>
<td>40.93</td>
<td>9.28</td>
<td>1.04</td>
</tr>
</tbody>
</table>
The group radical of a finite monoid $M$ is the smallest submonoid $D(M)$ of $M$ containing the idempotents and closed under weak conjugation: if $sts = s$ and $d \in D(M)$, then $sdt, tds \in D(M)$. 
Computation of the radical

Initialisation: \[ D(M) = E(M) \]

For each \( d \) in \( D(S) \)

For each weakly conjugate pair \( (s, t) \)

add \( sdt \) and \( tds \) to \( D(S) \)
add \( D(S)d \) to \( D(S) \).

Time complexity in \( 0(|S|^3) \).
If \( P \) is a subset of a monoid \( M \), the syntactic preorder \( \leq_P \) is defined on \( M \) by \( u \leq_P v \) iff, for all \( x, y \in M \),

\[
xvy \in P \Rightarrow xuy \in P
\]

Denote by \( \bar{P} \) the complement of \( P \). Then \( u \not\leq_P v \) iff there exist \( x, y \in M \) such that

\[
xuy \in \bar{P} \text{ and } xvy \in P
\]
The syntactic ordered monoid of $ab$ in $B^1_2$
An algorithm for the syntactic preorder

Let $G$ be the graph with $M \times M$ as set of vertices and edges of the form $(ua, va) \rightarrow (u, v)$ or $(au, av) \rightarrow (u, v)$.

We have seen that $u \not\leq_P v$ iff there exist $x, y \in M$ such that

$$xuy \in \overline{P} \text{ and } xvy \in P$$

Therefore, $u \not\leq_P v$ iff the vertex $(u, v)$ is reachable in $G$ from some vertex of $\overline{P} \times P$. 
The algorithm (2)

(1) Label each vertex \((u, v)\) as follows:

\[
\begin{align*}
(0, 1) & \quad \text{if } u \notin P \text{ and } v \in P \quad [u \not\leq_P v] \\
(1, 0) & \quad \text{if } u \in P \text{ and } v \notin P \quad [v \not\leq_P u] \\
(1, 1) & \quad \text{otherwise}
\end{align*}
\]

(2) Do a depth first search (starting from each vertex labeled by \((0, 1)\)) and set to 0 the first component of the label of all visited vertices.
Constraint propagation

(3) Do a **depth first search** (starting from each vertex labeled by \((0, 0)\) or \((1, 0)\)) and set to 0 the **second** component of the label of all visited vertices.

(4) The label of each vertex now encodes the **syntactic preorder** of \(P\) as follows:

\[
\begin{align*}
(1, 1) & \quad \text{if } u \sim_P v \\
(1, 0) & \quad \text{if } u \leq_P v \\
(0, 1) & \quad \text{if } v \leq_P u \\
(0, 0) & \quad \text{if } u \text{ and } v \text{ are incomparable}
\end{align*}
\]
Let $M = \{1, a, b\}$ with $aa = ba = a$ and $ab = bb = b$. Let $P = \{a\}$. 

Computation of the syntactic preorder
Computation of the syntactic preorder

Let $M = \{1, a, b\}$ with $aa = ba = a$ and $ab = bb = b$. Let $P = \{a\}$.

Initial labels

![Diagram showing the computation of the syntactic preorder]
Computation of the syntactic preorder

Let \( M = \{1, a, b\} \) with \( aa = ba = a \) and \( ab = bb = b \). Let \( P = \{a\} \).

DFS from \((b, a)\)
Computation of the syntactic preorder

Let $M = \{1, a, b\}$ with $aa = ba = a$ and $ab = bb = b$. Let $P = \{a\}$.

DFS from $(b, a)$
Let $M = \{1, a, b\}$ with $aa = ba = a$ and $ab = bb = b$. Let $P = \{a\}$. 

DFS from $(b, a)$
Computation of the syntactic preorder

Let $M = \{1, a, b\}$ with $aa = ba = a$ and $ab = bb = b$. Let $P = \{a\}$.

DFS from $(a, b)$
Computation of the syntactic preorder

Let $M = \{1, a, b\}$ with $aa = ba = a$ and $ab = bb = b$. Let $P = \{a\}$.

DFS from $(a, b)$
Let $M = \{1, a, b\}$ with $aa = ba = a$ and $ab = bb = b$. Let $P = \{a\}$.

Thus $a \leq_P 1 \leq_P b$
Complexity of the algorithm

The syntactic preorder can be computed in $O(|A||M|^2)$ time and space.
Aperiodicity

**Theorem (Cho-Huynh 1991)**

Testing *aperiodicity* of a deterministic $n$-state automaton is *P-space complete*.

**Proposition**

*One can test in* $O(|A||S|)$*-time whether an $A$-generated finite semigroup $S$ is aperiodic.*

It suffices to test whether the $\mathcal{H}$-classes are trivial.
Other varieties

**Proposition**

One can test in $O(|A||S|)$-time whether an $A$-generated finite semigroup $S$ is $\mathcal{R}$-trivial ($\mathcal{L}$-trivial, $\mathcal{J}$-trivial, commutative, idempotent, nilpotent, a group, a block-group).
This is a **difficult problem** for several reasons:

- It may happen that testing whether a set of identities is satisfied is **much easier** than testing whether any of the individual identities is satisfied.
- Identities for finite semigroups are **profinite identities**. The operations $x^\omega$ and $x^{\omega-1}$ are frequently needed, but other operators might be needed.
- There might be some tricky **tree pattern-matching problems** to solve.
Tree pattern-matching problems

A simple example: the variety $\mathbb{DS}$ is defined by the identity

$((xy)^\omega(yx)^\omega(xy)^\omega)^\omega = (xy)^\omega$
Semigroup theory might help...

Proposition

One can test in $O(|A||S|)$-time whether an $A$-generated finite semigroup $S$ belongs to $DS$.

Indeed, a semigroup belongs to $DS$ iff every regular $D$-class is union of groups. Therefore, it suffices to test whether the number of regular $H$-classes is equal to the number of idempotents.
A stamp is a morphism from a finitely generated free monoid onto a finite monoid. An ordered stamp is a stamp onto an ordered monoid.

\[ \varphi : A^* \rightarrow M \]
Stable subsemigroup

Let $\varphi : A^* \to M$ be a stamp and let $Z = \varphi(A)$. Then $Z$ belongs to the monoid $\mathcal{P}(M)$ of subsets of $M$.

Since $\mathcal{P}(M)$ is finite, $Z$ has an idempotent power. The stability index of $\varphi$ is the least positive integer such that $\varphi(A^s) = \varphi(A^{2s})$.

The set $\varphi(A^s)$ is a subsemigroup of $M$ called the stable semigroup of $\varphi$ and the monoid $\varphi(A^s) \cup \{1\}$ is called the stable monoid of $\varphi$. 
Applications to logic

**Theorem (McNaughton-Paper 1971, Schützenberger 1965)**

A language is $\text{FO}[<]$-definable iff its syntactic semigroup is *aperiodic*.

**Theorem (Barrington, Compton, Straubing, Thérien 1992)**

A language is $\text{FO}[< + \text{MOD}]$-definable iff the *stable semigroup* of its syntactic stamp is *aperiodic*.
A bit of logic

To each nonempty word \( u \) is associated a structure

\[ \mathcal{M}_u = (\{0, 1, \ldots, |u| - 1\}, <, (a)_{a \in A}) \]

where \( a \) is interpreted as the set of integers \( i \) such that the \( i \)-th letter of \( u \) is an \( a \), and \(<\) as the usual order on integers.

If \( u = abbaab \), then \( \text{Dom}(u) = \{0, 1, 2, 3, 4, 5\} \), \( a = \{0, 3, 4\} \) and \( b = \{1, 2, 5\} \).
Modular predicates

Let $d > 0$ and $r \in \mathbb{Z}/d\mathbb{Z}$. We define two new symbols (the modular symbols):

- The unary symbol $\text{MOD}_r^d$:
  \[
  \text{MOD}_r^d(n) = \{ i < n \mid i \text{ mod } d = r \}
  \]

- A constant symbol $m$ for the last position in a word
Fragments of first order logic

\( \text{FO}[\langle]\) denotes the set of first order formulas in the signature \( \{\langle, (a)_{a \in A}\}\}. \)

\( \text{FO}[\langle + \text{MOD}] \) denotes the logic obtained by adjoining all modular symbols.
Fragments of first order logic

\( \text{FO}[<] \) denotes the set of first order formulas in the signature \( \{<, (a)_{a \in A}\} \).

\( \text{FO}[< + \text{MOD}] \) denotes the logic obtained by adjoining all modular symbols.

\( \Sigma_1 \) denotes the set of existential formulas:

\[ \exists x_1 \cdots \exists x_n \, \varphi(x_1, \ldots, x_n) \]

where \( \varphi \) is quantifier-free.

\( \mathcal{B}\Sigma_1 \) denotes the set of Boolean combinations of \( \Sigma_1 \)-formulas.
Some examples

The formula $\exists x \ ax$ is interpreted as:

There exists an integer $x$ such that, in $u$, the letter in position $x$ is an $a$.

This defines the language $A^*aA^*$. 
Some examples

The formula $\exists x \ ax$ is interpreted as:

There exists an integer $x$ such that, in $u$, the letter in position $x$ is an $a$.

This defines the language $A^*aA^*$.

The formula $\exists x \ \exists y \ (x < y) \land ax \land by$ defines the language $A^*aA^*bA^*$. 
Some examples

The formula $\exists x \ ax$ is interpreted as:

There exists an integer $x$ such that, in $u$, the letter in position $x$ is an $a$.

This defines the language $A^*aA^*$.

The formula $\exists x \ \exists y \ (x < y) \land ax \land by$ defines the language $A^*aA^*bA^*$.

The formula $\exists x \ \forall y \ (x < y) \lor (x = y) \land ax$ defines the language $aA^*$.
Simple languages

A **simple** language is a language of the form

\[ A^{*}a_{1}A^{*}a_{2}A^{*} \cdots a_{k}A^{*} \]

where \( k \geq 0 \) and \( a_{1}, a_{2}, \ldots, a_{k} \in A \).

A **modular simple** language is a language of the form

\[ (A^{d})^{*}a_{1}(A^{d})^{*}a_{2}(A^{d})^{*} \cdots a_{k}(A^{d})^{*} \]

where \( d > 0, k \geq 0 \) and \( a_{1}, a_{2}, \ldots, a_{k} \in A \).
Logical description of simple languages

The language $A^*a_1A^*a_2A^* \cdots a_kA^*$ can be defined by the $\Sigma_1$-formula

$$\exists x_1 \ldots \exists x_k \ (x_1 < \ldots < x_k) \land (a_1x_1 \land \cdots \land a_kx_k)$$
Logical description of simple languages

The language $A^*a_1A^*a_2A^*\cdots a_kA^*$ can be defined by the $\Sigma_1$-formula

$$\exists x_1 \ldots \exists x_k (x_1 < \ldots < x_k) \land (a_1x_1 \land \cdots \land a_kx_k)$$

The language $(A^d)^*a_1(A^d)^*a_2(A^d)^*\cdots a_k(A^d)^*$ can be defined by the $\Sigma_1$-formula

$$\exists x_1 \ldots \exists x_k (x_1 < \ldots < x_k) \land (a_1x_1 \land \cdots \land a_kx_k) \land (\text{MOD}_0^d x_1 \land \text{MOD}_1^d x_2 \land \cdots \land \text{MOD}_{k-1}^d x_k \land \text{MOD}_k^d m)$$
First order

**Theorem (McNaughton-Paper 1971, Schützenberger 1965)**

A language is \( \text{FO}[\lt] \)-definable iff its syntactic semigroup is \textit{aperiodic}.

**Theorem (Barrington, Compton, Straubing, Thérien 1992)**

A language is \( \text{FO}[\lt + \text{MOD}] \)-definable iff the \textit{stable semigroup} of its syntactic stamp is \textit{aperiodic}.
Existential formulas ($\Sigma_1$)

**Proposition**

A language is definable in $\Sigma_1[<]$ iff it is a finite union of simple languages.

**Proposition**

A language is definable in $\Sigma_1[< + \text{MOD}]$ iff it is a finite union of modular simple languages.
Algebraic characterization

**Theorem (Thomas 1982, Perrin-Pin 1986)**

* A language is definable in $\Sigma_1[<]$ iff its ordered syntactic monoid satisfies the identity $x \leq 1$.

**Theorem (Chaubard, Pin, Straubing 2006)**

* A language is definable in $\Sigma_1[< + \text{MOD}]$ iff the stable ordered monoid of its ordered syntactic stamp satisfies the identity $x \leq 1$. 
A morphism $f : A^* \rightarrow B^*$ is length-multiplying ($lm$ for short) if there exists an integer $k$ such that the image of each letter of $A$ is a word of $B^k$.

For instance, if $A = \{a, b\}$ and $B = \{a, b, c\}$, the morphism defined by $\varphi(a) = abca$ and $\varphi(b) = cbba$ is length-multiplying.
Let $u, v$ be two words on the alphabet $B$. A morphism $\varphi : A^* \to M$ satisfies the $lm$-identity $u = v$ if, for every $lm$-morphism $f : B^* \to A^*$, $\varphi \circ f(u) = \varphi \circ f(v)$.

For instance, $\varphi : A^* \to M$ satisfies the $lm$-identity $xyx = xy$ if for any pair of words of the same length $x, y$ of $A^*$, $\varphi(xy) = \varphi(xy)$.
Let $u, v$ be two words on the alphabet $B$. A morphism $\varphi : A^* \to M$ satisfies the \textit{lm}-identity $u = v$ if, for every \textit{lm}-morphism $f : B^* \to A^*$, $\varphi \circ f(u) = \varphi \circ f(v)$.

For instance, $\varphi : A^* \to M$ satisfies the \textit{lm}-identity $xyx = xy$ if for any pair of words of the same length $x, y$ of $A^*$, $\varphi(xyxy) = \varphi(xy)$.

If $M$ is ordered, we say that $\varphi$ satisfies the \textit{lm}-identity $u \leq v$ if, for every \textit{lm}-morphism $f : B^* \to A^*$, $\varphi \circ f(u) \leq \varphi \circ f(v)$.
Characterization by $lm$-identities

**Theorem (Thomas 1982, Perrin-Pin 1986)**

A language is definable in $\Sigma_1[<]$ iff its ordered syntactic monoid satisfies the identity $x \leq 1$.

**Theorem (Chaubard, Pin, Straubing 2006)**

A language is definable in $\Sigma_1[< + \text{MOD}]$ iff its ordered syntactic stamp satisfies the $lm$-identities $x^{\omega-1}y \leq 1$ and $yx^{\omega-1} \leq 1$. 
Boolean combination of existential formulas

**Theorem (Thomas 1982)**

A language is definable in $\mathcal{B}\Sigma_1[<]$ iff it is a Boolean combination of simple languages.

**Theorem (Chaubard, Pin, Straubing 2006)**

A language is definable in $\mathcal{B}\Sigma_1[< + \text{MOD}]$ iff it is a Boolean combination of modular simple languages.
Algebraic characterization

**Theorem (Simon 1972, Thomas 1982)**

A language is definable in $\mathcal{B}\Sigma_1[<]$ iff its syntactic monoid is $\mathcal{J}$-trivial.

**Theorem (Chaubard, Pin, Straubing 2006)**

A language is definable in $\mathcal{B}\Sigma_1[< + \text{MOD}]$ iff its syntactic stamp belongs to the $\mathcal{lm}$-variety $\mathcal{J} \ast \text{MOD}$. 
Derived category of a stamp $\varphi : A^* \to M$

Let $\pi_n(u) = |u| \mod n$.

Let $C_n(\varphi)$ be the category whose objects are elements of $\mathbb{Z}/n\mathbb{Z}$ and whose arrows from $i$ to $j$ are the triples $(i, m, j)$ where $j - i \in \pi_n(\varphi^{-1}(m))$.

Composition is given by $(i, m_1, j)(j, m_2, k) = (i, m_1m_2, k)$. 
A decidable characterization

**Theorem** (Chaubard, Pin, Straubing 2006)

Let \( \varphi \) be a stamp of stability index \( s \). Then \( \varphi \) belongs to \( J \star \text{MOD} \) iff \( C_s(\varphi) \) is in \( gJ \).

**Corollary**

Let \( \varphi \) be the syntactic stamp of a language \( L \) and let \( s \) be its stability index. Then \( L \) is definable in \( B\Sigma_1[< + \text{MOD}] \) iff \( C_s(\varphi) \) is in \( gJ \).

No characterization by \( lm \)-identities is known at the moment.
What would be useful in GAP 4... 

- Define **stamps** as a basic object.

- Compute **stable semigroups** and monoids of stamps.

- Test for **length-preserving** and **length-multiplying identities**.

- Compute **derived categories**
References I

