Computing with Semigroup Congruences

Michael Torpey

University of St Andrews

2016-03-17
A **congruence** on a semigroup $S$ is an equivalence relation $\rho \subseteq S \times S$ such that

$$(x, y) \in \rho \implies (ax, ay), (xa, ya) \in \rho,$$

or equivalently,

$$(x, y), (s, t) \in \rho \implies (xs, yt) \in \rho,$$

for all $x, y, a, s, t \in S$.

(we may write $x \rho y$ for $(x, y) \in \rho$)
Simple ways to represent congruences

- List of pairs: \( \{(x_1, x_3), (x_1, x_9), (x_{42}, x_{11}), \ldots \} \)
- Partition: \( \{\{x_1, x_3, x_9, x_{14}\}, \{x_2\}, \{x_4, x_5, x_8\}, \ldots \} \)
- ID list: \( (1, 2, 1, 3, 3, 4, 5, 3, 1, \ldots) \)
Generating pairs

- Let \( R \subseteq S \times S \) be a set of pairs.
- Let \( \rho \) be the least congruence on \( S \) containing all the pairs in \( R \).
- We call \( R \) a **generating set** for \( \rho \).
- Two elements \( a \) and \( b \) are \( \rho \)-related if and only if there exists a sequence
  \[
  a = a_1 \rightarrow a_2 \rightarrow \cdots \rightarrow a_n = b
  \]
such that for each \( i \) there exist \( x, y, z, t \) such that
  \[
  a_i = xyz, \quad a_{i+1} = xty,
  \]
and either \((z,t)\) or \((t,z)\) is in \( \rho \).
- Finding whether two elements are \( \rho \)-related has worst-case complexity \( O(|S|^2) \).
Simple and 0-simple semigroups

Definition
A semigroup $S$ without zero is **simple** if it has no proper ideals.

Definition
A semigroup $S$ with zero is **0-simple** if its only ideals are $\{0\}$ and $S$. 
Theorem (Rees)

Every completely 0-simple semigroup is isomorphic to a Rees 0-matrix semigroup

\[ M^0[G; I, \Lambda; P], \]

where \( G \) is a group and \( P \) is regular. Conversely, every such Rees 0-matrix semigroup is completely 0-simple.
Definition

For a finite 0-simple Rees 0-matrix semigroup $\mathcal{M}^0[G; I, \Lambda; P]$, a **linked triple** is a triple

$$(N, S, T)$$

consisting of a normal subgroup $N \triangleleft G$, an equivalence relation $S$ on $I$ and an equivalence relation $T$ on $\Lambda$, such that the following are satisfied:

1. $S$ only relates columns which have zeroes in the same places,
2. $T$ only relates rows which have zeroes in the same places,
3. For all $i, j \in I$ and $\lambda, \mu \in \Lambda$ such that $p_{\lambda i}, p_{\lambda j}, p_{\mu i}, p_{\mu j} \neq 0$ and either $(i, j) \in S$ or $(\lambda, \mu) \in T$, we have that $q_{\lambda \mu ij} \in N$, where

$$q_{\lambda \mu ij} = p_{\lambda i}p_{\mu i}^{-1}p_{\mu j}p_{\lambda j}^{-1}.$$
A finite 0-simple semigroup $S$ has a bijection $\Gamma$ between its *non-universal* congruences and its linked triples, 

$$\Gamma : \rho \mapsto (N, S, T)$$

We may write $\rho$ as $[N, S, T]$.

Two non-zero elements $(i, a, \lambda)$ and $(j, b, \mu)$ are $\rho$-related if and only if

1. $(i, j) \in S$;
2. $(\lambda, \mu) \in T$;
3. $(p_{\xi i} a p_{\lambda x})(p_{\xi j} b p_{\mu x})^{-1} \in N$ for some $x \in I, \xi \in \Lambda$ such that $p_{\xi i}, p_{\xi j}, p_{\lambda x}, p_{\mu x} \neq 0$.

This can be determined in constant time.

$(i, a, \lambda)$ is related to 0 only in the universal congruence $S \times S$. 

Michael Torpey (University of St Andrews)  Computing with Semigroup Congruences  2016-03-17  8 / 15
Finding a congruence’s linked triple

- Clearly, if we have a congruence’s linked triple, we should use it for all calculations. But what if we do not?
- We want an algorithm to find a linked triple \((N, S, T)\) from a set of generating pairs \(R\).
- First observe the following:

**Lemma**

If \((N_1, S_1, T_1)\) and \((N_2, S_2, T_2)\) are linked triples such that

\[
N_1 \leq N_2, \quad S_1 \subseteq S_2, \quad T_1 \subseteq T_2,
\]

then \([N_1, S_1, T_1] \subseteq [N_2, S_2, T_2]\).
Finding a congruence’s linked triple

Our strategy:

- Find any elements that must be in $\mathcal{N}$ because of $\mathcal{R}$.
- Find any pairs of columns that must be in $\mathcal{S}$ because of $\mathcal{R}$.
- Find any pairs of rows that must be in $\mathcal{T}$ because of $\mathcal{R}$.
- Add any elements and pairs necessary for $(\mathcal{N}, \mathcal{S}, \mathcal{T})$ to be linked.
- Watch out for anything that would force this to be the universal congruence $\mathcal{S} \times \mathcal{S}$.
The algorithm

Require: \( S = \mathcal{M}^0[G; I, \Lambda; P] \) is a finite 0-simple Rees 0-matrix semigroup

procedure LinkedTriple(\( R \))

\[ N := \emptyset \]
\[ S := \Delta_I \]
\[ T := \Delta_\Lambda \]

for \((x, y) \in R\) do
    if \( x = y \) then
        Skip this pair
    else if \( x = 0 \) or \( y = 0 \) then
        return “Universal Congruence” (no linked triple)
    end if
    Let \( x = (i, a, \lambda) \)
    Let \( y = (j, b, \mu) \)
    if \((i, j) \notin \varepsilon_I \) or \((\lambda, \mu) \notin \varepsilon_\Lambda\) then
        return “Universal Congruence” (no linked triple)
    end if
The algorithm

Require: \( S = \mathcal{M}^0[G; I, \Lambda; P] \) is a finite 0-simple Rees 0-matrix semigroup

procedure \textsc{LinkedTriple}(R)

\[ \ldots \]

\begin{algorithmic}
\State for \((x, y) \in R\) do
\State \ldots
\State \Comment{Combine row and column classes}
\State \textsc{Union}(\(S, i, j\))
\State \textsc{Union}(\(T, \lambda, \mu\))
\State \ldots
\State \Comment{Add generators for normal subgroup}
\State Choose \(\nu \in \Lambda\) such that \(p_{\nu i} \neq 0\)
\State Choose \(k \in I\) such that \(p_{\lambda k} \neq 0\)
\State Add \((p_{\nu i} a p_{\lambda k})(p_{\nu j} b p_{\mu k})^{-1}\) to \(N\)
\State \ldots
\end{algorithmic}
The algorithm

Require: \( S = \mathcal{M}^0[G; I, \Lambda; P] \) is a finite 0-simple Rees 0-matrix semigroup

procedure \text{LINKEDTRIPLE}(R)

\[ \ldots \]

\[ \text{for } (x, y) \in R \text{ do} \]

\[ \ldots \]

\[ \triangleright \text{ Add more generators for normal subgroup} \]

\[ \text{for } \xi \in \Lambda \setminus \{\nu\} \text{ such that } p_{\xi i} \neq 0 \text{ do} \]

\[ \text{Add } q_{\nu \xi ij} \text{ to } N \]

\[ \text{end for} \]

\[ \text{for } x \in I \setminus \{k\} \text{ such that } p_{\lambda x} \neq 0 \text{ do} \]

\[ \text{Add } q_{\lambda \mu kx} \text{ to } N \]

\[ \text{end for} \]

\[ \text{end for} \]

\[ N := \langle \langle N \rangle \rangle \]

\[ \text{return } (N, S, T) \]

end procedure
The algorithm

- Finding the linked triple is fast.
- Doesn’t require enumerating $S$.
- Transforms an $O(|S|^2)$ time problem into $O(1)$.
- Other information can be found from $(N, S, T)$: number of congruence classes, size of congruence classes, etc.
- A list of all congruences on $S$ can be found.
Generic semigroups: generating pairs.
Simple & 0-simple semigroups: linked triples.
Inverse semigroups: kernel and trace.
Rees congruences are also implemented.