On the Complexity of the Cayley Semigroup Membership Problem

Lukas Fleischer

FMI, University of Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany
fleischer@fmi.uni-stuttgart.de

June 24, 2018

The Cayley Semigroup Membership Problem

- CSM:
  - **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
  - **Question:** Is t in the subsemigroup generated by X?

The Cayley Semigroup Membership Problem

- CSM:
  - **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
  - **Question:** Is t in the subsemigroup generated by X?
- input encoding:
  - S given as multiplication table ($N^2 \log(N)$ bits for $N = |S|$)
  - X given as a list of elements ($k \log(N)$ bits for $k = |X|$)
  - t encoded using $\log(N)$ bits

The Cayley Semigroup Membership Problem

- CSM:
  - **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
  - **Question:** Is t in the subsemigroup generated by X?
- input encoding:
  - S given as multiplication table ($N^2 \log(N)$ bits for $N = |S|$)
  - X given as a list of elements ($k \log(N)$ bits for $k = |X|$)
  - t encoded using $\log(N)$ bits
- CSM($\mathbf{C}$): restriction to semigroups from a class $\mathbf{C}$
  - **Com**: commutative semigroups ("$xy = yx$")
  - **G**: groups ("$e^2 = e \implies ex = x = xe$")
  - **N**: nilpotent semigroups ("$e^2 = e \implies ex = e = xe$")

The Cayley Semigroup Membership Problem

- CSM:
  - **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
  - **Question:** Is t in the subsemigroup generated by X?
- input encoding:
  - S given as multiplication table ($N^2 \log(N)$ bits for $N = |S|$)
  - X given as a list of elements ($k \log(N)$ bits for $k = |X|$)
  - t encoded using $\log(N)$ bits
- CSM($\mathbf{C}$): restriction to semigroups from a class $\mathbf{C}$
  - **Com**: commutative semigroups ("$xy = yx$")
  - **G**: groups ("$e^2 = e \implies ex = x = xe$")
  - **N**: nilpotent semigroups ("$e^2 = e \implies ex = e = xe$")
- mostly interested in **varieties** (classes closed under direct products and division)

Motivation

- connections between complexity classes and algebra

Motivation

- connections between complexity classes and algebra
- The **emptiness**, **universality**, **inclusion** and **equivalence problems** for regular languages represented by morphisms to finite semigroups (encoded as multiplication table) are AC^0-Turing-reducible to CSM (and vice versa).

Motivation

- connections between complexity classes and algebra
- The **emptiness**, **universality**, **inclusion** and **equivalence problems** for regular languages represented by morphisms to finite semigroups (encoded as multiplication table) are AC^0-Turing-reducible to CSM (and vice versa).
- The reductions "preserve varieties"!

Historical Background

- First investigated by Jones, Lien and Laaser (1976): CSM is NL-complete.

Historical Background

- First investigated by Jones, Lien and Laaser (1976): CSM is NL-complete.
- Barrington and McKenzie (1991): $\mathrm{CSM}(\mathbf{G})$ is decidable in SL. Suggested that $\mathrm{CSM}(\mathbf{G})$ might be complete for L.

Historical Background

- First investigated by Jones, Lien and Laaser (1976): CSM is NL-complete.
- Barrington and McKenzie (1991): $\mathrm{CSM}(\mathbf{G})$ is decidable in SL. Suggested that $\mathrm{CSM}(\mathbf{G})$ might be complete for L.
- Barrington, Kadau, Lange and McKenzie (2001): $\mathrm{CSM}(\mathbf{Ab})$ and $\mathrm{CSM}(\mathbf{G_{nil}})$ cannot be hard for any class containing $\textsc{Parity}$ (such as ACC^0, TC^0, NC^1, L or NL).

Historical Background

- First investigated by Jones, Lien and Laaser (1976): CSM is NL-complete.
- Barrington and McKenzie (1991): $\mathrm{CSM}(\mathbf{G})$ is decidable in SL. Suggested that $\mathrm{CSM}(\mathbf{G})$ might be complete for L.
- Barrington, Kadau, Lange and McKenzie (2001): $\mathrm{CSM}(\mathbf{Ab})$ and $\mathrm{CSM}(\mathbf{G_{nil}})$ cannot be hard for any class containing $\textsc{Parity}$ (such as ACC^0, TC^0, NC^1, L or NL).
- Group case remained open!

Historical Background

- First investigated by Jones, Lien and Laaser (1976): CSM is NL-complete.
- Barrington and McKenzie (1991): $\mathrm{CSM}(\mathbf{G})$ is decidable in SL. Suggested that $\mathrm{CSM}(\mathbf{G})$ might be complete for L.
- Barrington, Kadau, Lange and McKenzie (2001): $\mathrm{CSM}(\mathbf{Ab})$ and $\mathrm{CSM}(\mathbf{G_{nil}})$ cannot be hard for any class containing $\textsc{Parity}$ (such as ACC^0, TC^0, NC^1, L or NL).
- Group case remained open!
- **This talk**: Both $\mathrm{CSM}(\mathbf{Com})$ and $\mathrm{CSM}(\mathbf{G})$ cannot be hard for any class containing $\textsc{Parity}$.

Historical Background

- First investigated by Jones, Lien and Laaser (1976): CSM is NL-complete.
- Barrington and McKenzie (1991): $\mathrm{CSM}(\mathbf{G})$ is decidable in SL. Suggested that $\mathrm{CSM}(\mathbf{G})$ might be complete for L.
- Barrington, Kadau, Lange and McKenzie (2001): $\mathrm{CSM}(\mathbf{Ab})$ and $\mathrm{CSM}(\mathbf{G_{nil}})$ cannot be hard for any class containing $\textsc{Parity}$ (such as ACC^0, TC^0, NC^1, L or NL).
- Group case remained open!
- **This talk**: Both $\mathrm{CSM}(\mathbf{Com})$ and $\mathrm{CSM}(\mathbf{G})$ cannot be hard for any class containing $\textsc{Parity}$.
- **Also**: NL-completeness of $\mathrm{CSM}(\mathbf{N})$ and further results.

An NL-algorithm for CSM

- **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$

An NL-algorithm for CSM

- **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
- t is in the subsemigroup generated by X if and only if there exist elements $x_1, \ldots, x_k \in X$ with $x_1 \cdots x_k = t$.

An NL-algorithm for CSM

- **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
- t is in the subsemigroup generated by X if and only if there exist elements $x_1, \ldots, x_k \in X$ with $x_1 \cdots x_k = t$.
- Single elements can be stored in log-space, multiplications can be performed in log-space.

An NL-algorithm for CSM

- **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
- t is in the subsemigroup generated by X if and only if there exist elements $x_1, \ldots, x_k \in X$ with $x_1 \cdots x_k = t$.
- Single elements can be stored in log-space, multiplications can be performed in log-space.
- Start by guessing an element $x_1 \in X$ and set $y := x_1$.

An NL-algorithm for CSM

- **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
- t is in the subsemigroup generated by X if and only if there exist elements $x_1, \ldots, x_k \in X$ with $x_1 \cdots x_k = t$.
- Single elements can be stored in log-space, multiplications can be performed in log-space.
- Start by guessing an element $x_1 \in X$ and set $y := x_1$.
- Iterate: Guess an element $x_{i+1} \in X$ and set $y := y \cdot x_{i+1}$.

An NL-algorithm for CSM

- **Input:** finite semigroup S, set $X \subseteq S$, element $t \in S$
- t is in the subsemigroup generated by X if and only if there exist elements $x_1, \ldots, x_k \in X$ with $x_1 \cdots x_k = t$.
- Single elements can be stored in log-space, multiplications can be performed in log-space.
- Start by guessing an element $x_1 \in X$ and set $y := x_1$.
- Iterate: Guess an element $x_{i+1} \in X$ and set $y := y \cdot x_{i+1}$.
- Non-deterministically stop iteration and compare y to t.

Nilpotent Semigroups

Theorem

$\mathrm{CSM}(\mathbf{N})$ *is* NL*-complete (under* AC^0 *reductions).*

Nilpotent Semigroups

Theorem

$\mathrm{CSM}(\mathbf{N})$ *is* NL*-complete (under* AC^0 *reductions).*

Proof.

- We reduce STConn to $\mathrm{CSM}(\mathbf{N})$.

Nilpotent Semigroups

Theorem

$\mathrm{CSM}(\mathbf{N})$ *is* NL*-complete (under* AC^0 *reductions).*

Proof.

- We reduce STConn to $\mathrm{CSM}(\mathbf{N})$.
- Let $G = (V, E)$ be a directed graph with n vertices.

Nilpotent Semigroups

Theorem

$\mathrm{CSM}(\mathbf{N})$ *is* NL*-complete (under* AC^0 *reductions).*

Proof.

- We reduce STConn to $\mathrm{CSM}(\mathbf{N})$.
- Let $G = (V, E)$ be a directed graph with n vertices.
- Let $S = V \times \{1, \ldots, n-1\} \times V \cup \{0\}$ with

$$(v, i, w) \cdot (x, j, y) = \begin{cases} (v, i+j, y) & \text{if } w = x \text{ and } i + j < n, \\ 0 & \text{otherwise.} \end{cases}$$

- 0 is a zero element.

Nilpotent Semigroups

Theorem

$\mathrm{CSM}(\mathbf{N})$ *is* NL*-complete (under* AC^0 *reductions).*

Proof.

- We reduce STConn to $\mathrm{CSM}(\mathbf{N})$.
- Let $G = (V, E)$ be a directed graph with n vertices.
- Let $S = V \times \{1, \ldots, n-1\} \times V \cup \{0\}$ with

$$(v, i, w) \cdot (x, j, y) = \begin{cases} (v, i+j, y) & \text{if } w = x \text{ and } i + j < n, \\ 0 & \text{otherwise.} \end{cases}$$

- 0 is a zero element.
- $X = \{\, (v, 1, w) \mid v = w \text{ or } (v, w) \in E \,\}$

Nilpotent Semigroups

Theorem

$\mathrm{CSM}(\mathbf{N})$ *is* NL*-complete (under* AC^0 *reductions).*

Proof.

- We reduce STConn to $\mathrm{CSM}(\mathbf{N})$.
- Let $G = (V, E)$ be a directed graph with n vertices.
- Let $S = V \times \{1, \ldots, n-1\} \times V \cup \{0\}$ with

$$(v, i, w) \cdot (x, j, y) = \begin{cases} (v, i+j, y) & \text{if } w = x \text{ and } i + j < n, \\ 0 & \text{otherwise.} \end{cases}$$

- 0 is a zero element.
- $X = \{\, (v, 1, w) \mid v = w \text{ or } (v, w) \in E \,\}$
- $(s, n-1, t) \in \langle X \rangle$ if and only if t is reachable from s in G.

Nilpotent Semigroups

Theorem

$\mathrm{CSM}(\mathbf{N})$ *is* NL*-complete (under* AC^0 *reductions).*

Proof.

- We reduce STConn to $\mathrm{CSM}(\mathbf{N})$.
- Let $G = (V, E)$ be a directed graph with n vertices.
- Let $S = V \times \{1, \ldots, n-1\} \times V \cup \{0\}$ with

$$(v, i, w) \cdot (x, j, y) = \begin{cases} (v, i+j, y) & \text{if } w = x \text{ and } i + j < n, \\ 0 & \text{otherwise.} \end{cases}$$

- 0 is a zero element.
- $X = \{\, (v, 1, w) \mid v = w \text{ or } (v, w) \in E \,\}$
- $(s, n-1, t) \in \langle X \rangle$ if and only if t is reachable from s in G.
- 0 is the only idempotent of S. □

Circuit Complexity

- unbounded fan-in Boolean circuits

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth
- FOLL: polynomial size and depth $\mathcal{O}(\log \log n)$

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth
- FOLL: polynomial size and depth $\mathcal{O}(\log \log n)$
- qAC^0: quasi-polynomial size and constant depth

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth
- FOLL: polynomial size and depth $\mathcal{O}(\log \log n)$
- qAC^0: quasi-polynomial size and constant depth
- Håstad (1986) and Yao (1985): none of the classes above contain $\textsc{Parity} = \{ w \in \{0,1\}^* \mid |w|_1 \equiv 0 \bmod 2 \}$.

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth
- FOLL: polynomial size and depth $\mathcal{O}(\log \log n)$
- qAC^0: quasi-polynomial size and constant depth
- Håstad (1986) and Yao (1985): none of the classes above contain $\textsc{Parity} = \{ w \in \{0,1\}^* \mid |w|_1 \equiv 0 \bmod 2 \}$.
- No problem in the classes above can be hard for any class containing $\textsc{Parity}$ such as $\mathsf{ACC}^0, \mathsf{TC}^0, \mathsf{NC}^1, \mathsf{L}, \mathsf{NL}$.

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth
- FOLL: polynomial size and depth $\mathcal{O}(\log \log n)$
- qAC^0: quasi-polynomial size and constant depth
- Håstad (1986) and Yao (1985): none of the classes above contain $\textsc{Parity} = \{ w \in \{0,1\}^* \mid |w|_1 \equiv 0 \bmod 2 \}$.
- No problem in the classes above can be hard for any class containing $\textsc{Parity}$ such as $\mathsf{ACC}^0, \mathsf{TC}^0, \mathsf{NC}^1, \mathsf{L}, \mathsf{NL}$.
- $\mathsf{AC}^0 \subsetneq \mathsf{FOLL} \subsetneq \mathsf{AC}^1$

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth
- FOLL: polynomial size and depth $\mathcal{O}(\log \log n)$
- qAC^0: quasi-polynomial size and constant depth
- Håstad (1986) and Yao (1985): none of the classes above contain $\textsc{Parity} = \{ w \in \{0,1\}^* \mid |w|_1 \equiv 0 \bmod 2 \}$.
- No problem in the classes above can be hard for any class containing $\textsc{Parity}$ such as $\mathsf{ACC}^0, \mathsf{TC}^0, \mathsf{NC}^1, \mathsf{L}, \mathsf{NL}$.
- $\mathsf{AC}^0 \subsetneq \mathsf{FOLL} \subsetneq \mathsf{AC}^1$
- $\mathsf{AC}^0 \subsetneq \mathsf{qAC}^0$

Circuit Complexity

- unbounded fan-in Boolean circuits
- AC^0: polynomial size and constant depth
- FOLL: polynomial size and depth $\mathcal{O}(\log \log n)$
- qAC^0: quasi-polynomial size and constant depth
- Håstad (1986) and Yao (1985): none of the classes above contain $\textsc{Parity} = \{ w \in \{0,1\}^* \mid |w|_1 \equiv 0 \bmod 2 \}$.
- No problem in the classes above can be hard for any class containing $\textsc{Parity}$ such as $\mathsf{ACC}^0, \mathsf{TC}^0, \mathsf{NC}^1, \mathsf{L}, \mathsf{NL}$.
- $\mathsf{AC}^0 \subsetneq \mathsf{FOLL} \subsetneq \mathsf{AC}^1$
- $\mathsf{AC}^0 \subsetneq \mathsf{qAC}^0$
- $\mathsf{qAC}^0 \not\subseteq \mathsf{FOLL}$ and $\mathsf{FOLL} \not\subseteq \mathsf{qAC}^0$

Straight-Line Programs and the PLCP

- S finite semigroup, $X \subseteq S$

Straight-Line Programs and the PLCP

- S finite semigroup, $X \subseteq S$
- $(s_1, \ldots, s_\ell) \in S^\ell$ is an **SLP over** X if for each $i \in \{1, \ldots, \ell\}$, we have $s_i \in X$ or $s_i = s_p s_q$ for some $p, q < i$.

Straight-Line Programs and the PLCP

- S finite semigroup, $X \subseteq S$
- $(s_1, \ldots, s_\ell) \in S^\ell$ is an **SLP over** X if for each $i \in \{1, \ldots, \ell\}$, we have $s_i \in X$ or $s_i = s_p s_q$ for some $p, q < i$.
- ℓ is the **length** of the SLP.

Straight-Line Programs and the PLCP

- S finite semigroup, $X \subseteq S$
- $(s_1, \ldots, s_\ell) \in S^\ell$ is an **SLP over** X if for each $i \in \{1, \ldots, \ell\}$, we have $s_i \in X$ or $s_i = s_p s_q$ for some $p, q < i$.
- ℓ is the **length** of the SLP.
- $s_1, \ldots, s_\ell$ are the elements **computed** by the SLP.

Straight-Line Programs and the PLCP

- S finite semigroup, $X \subseteq S$
- $(s_1, \ldots, s_\ell) \in S^\ell$ is an **SLP over** X if for each $i \in \{1, \ldots, \ell\}$, we have $s_i \in X$ or $s_i = s_p s_q$ for some $p, q < i$.
- ℓ is the **length** of the SLP.
- $s_1, \ldots, s_\ell$ are the elements **computed** by the SLP.
- SLPs can also be seen as algebraic circuits.

Straight-Line Programs and the PLCP

- S finite semigroup, $X \subseteq S$
- $(s_1, \ldots, s_\ell) \in S^\ell$ is an **SLP over** X if for each $i \in \{1, \ldots, \ell\}$, we have $s_i \in X$ or $s_i = s_p s_q$ for some $p, q < i$.
- ℓ is the **length** of the SLP.
- $s_1, \ldots, s_\ell$ are the elements **computed** by the SLP.
- SLPs can also be seen as algebraic circuits.
- $\mathbf{C}$ has the **Poly-Logarithmic Circuits Property** (**PLCP**) if for all $S \in \mathbf{C}$ and $X \subseteq S$: every element t in the subsemigroup generated by X can be computed by an SLP of poly-logarithmic length (in $|S|$) over X.

Straight-Line Programs and the PLCP

- **Note**: we can extend SLPs and allow $s_i = s_p^k$ for $p < i$ and for some fixed $k \in \mathbb{N}$ — can be simulated by an ordinary SLP of length $\log |S|$ using "square and multiply"

Straight-Line Programs and the PLCP

- **Note**: we can extend SLPs and allow $s_i = s_p^k$ for $p < i$ and for some fixed $k \in \mathbb{N}$ — can be simulated by an ordinary SLP of length $\log |S|$ using "square and multiply"
- $s_i = s_p^6 \rightsquigarrow (s_i'', s_i', s_i)$ with $s_i'' = s_p s_p$, $s_i' = s_i'' s_p$, $s_i = s_i' s_i'$

Straight-Line Programs and the PLCP

- **Note**: we can extend SLPs and allow $s_i = s_p^k$ for $p < i$ and for some fixed $k \in \mathbb{N}$ — can be simulated by an ordinary SLP of length $\log |S|$ using "square and multiply"
- $s_i = s_p^6 \rightsquigarrow (s_i'', s_i', s_i)$ with $s_i'' = s_p s_p$, $s_i' = s_i'' s_p$, $s_i = s_i' s_i'$
- for groups, we allow $s_i = s_p^{-1}$ with $p < i$; same as $s_p^{|G|-1}$

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathsf{qAC}^0$.

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathrm{qAC}^0$.

Proof.

- PLCP gives a poly-logarithmic upper bound $\log^c |S|$ on SLPs

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathrm{qAC}^0$.

Proof.

- PLCP gives a poly-logarithmic upper bound $\log^c |S|$ on SLPs
- **Idea:** in parallel, test for all sequences of $\log^c |S|$ elements of S whether the sequence is an SLP computing t

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathrm{qAC}^0$.

Proof.

- PLCP gives a poly-logarithmic upper bound $\log^c |S|$ on SLPs
- **Idea:** in parallel, test for all sequences of $\log^c |S|$ elements of S whether the sequence is an SLP computing t
- $|S|^{\log^c |S|} \leq n^{\log^c n}$ such sequences

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathrm{qAC}^0$.

Proof.

- PLCP gives a poly-logarithmic upper bound $\log^c |S|$ on SLPs
- **Idea:** in parallel, test for all sequences of $\log^c |S|$ elements of S whether the sequence is an SLP computing t
- $|S|^{\log^c |S|} \leq n^{\log^c n}$ such sequences
- create parallel AC^0 sub-circuits for all such sequences

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathrm{qAC}^0$.

Proof.

- PLCP gives a poly-logarithmic upper bound $\log^c |S|$ on SLPs
- **Idea:** in parallel, test for all sequences of $\log^c |S|$ elements of S whether the sequence is an SLP computing t
- $|S|^{\log^c |S|} \leq n^{\log^c n}$ such sequences
- create parallel AC^0 sub-circuits for all such sequences
  - checking whether an element in the sequence equals t: single comparison (actually, single AND gate with some NOT gates)

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathrm{qAC}^0$.

Proof.

- PLCP gives a poly-logarithmic upper bound $\log^c |S|$ on SLPs
- **Idea:** in parallel, test for all sequences of $\log^c |S|$ elements of S whether the sequence is an SLP computing t
- $|S|^{\log^c |S|} \leq n^{\log^c n}$ such sequences
- create parallel AC^0 sub-circuits for all such sequences
  - checking whether an element in the sequence equals t: single comparison (actually, single AND gate with some NOT gates)
  - checking whether an element belongs to X: n comparisons

The PLCP and qAC^0

Theorem

If $\mathbf{C}$ *has the PLCP, then* $\mathrm{CSM}(\mathbf{C}) \in \mathrm{qAC}^0$.

Proof.

- PLCP gives a poly-logarithmic upper bound $\log^c |S|$ on SLPs
- **Idea:** in parallel, test for all sequences of $\log^c |S|$ elements of S whether the sequence is an SLP computing t
- $|S|^{\log^c |S|} \leq n^{\log^c n}$ such sequences
- create parallel AC^0 sub-circuits for all such sequences
  - checking whether an element in the sequence equals t: single comparison (actually, single AND gate with some NOT gates)
  - checking whether an element belongs to X: n comparisons
  - checking whether an element is a product of two previous elements: at most $(\log^c |S|)^2 \leq \log^{2c}(n)$ parallel table lookups and comparisons

□

Groups have the PLCP

Lemma (Reachability Lemma, Babai and Szemerédi (1984))

Let G be a finite group and let X be a set of generators of G. Then, for each element $t \in G$, there exists an SLP of length $(\log |G| + 1)^2$ over X computing t.

Groups have the PLCP

Lemma (Reachability Lemma, Babai and Szemerédi (1984))

Let G be a finite group and let X be a set of generators of G. Then, for each element $t \in G$, there exists an SLP of length $(\log |G| + 1)^2$ over X computing t.

Lemma

G *has the PLCP.*

Commutative Semigroups have the PLCP

Lemma

Com *has the PLCP.*

Commutative Semigroups have the PLCP

Lemma

Com *has the PLCP.*

Proof.

- Let $S \in \mathbf{Com}$ with $|S| = N$.

Commutative Semigroups have the PLCP

Lemma

Com *has the PLCP.*

Proof.

- Let $S \in \mathbf{Com}$ with $|S| = N$.
- Suppose t is in the subsemigroup generated by X.

Commutative Semigroups have the PLCP

Lemma

Com *has the PLCP.*

Proof.

- Let $S \in \mathbf{Com}$ with $|S| = N$.
- Suppose t is in the subsemigroup generated by X.
- Then $t = x_1^{i_1} \cdots x_k^{i_k}$ for some $k \leq \log(N)$, some $x_1, \ldots, x_k \in X$ and $i_1, \ldots, i_k \in \{1, \ldots, N\}$.

Commutative Semigroups have the PLCP

Lemma

Com *has the PLCP.*

Proof.

- Let $S \in \mathbf{Com}$ with $|S| = N$.
- Suppose t is in the subsemigroup generated by X.
- Then $t = x_1^{i_1} \cdots x_k^{i_k}$ for some $k \leq \log(N)$, some $x_1, \ldots, x_k \in X$ and $i_1, \ldots, i_k \in \{1, \ldots, N\}$.
- To see this, note that if k is minimal, then all products of the form $y_1 \cdots y_\ell$ with $y_i \in \left\{x_1^{i_1}, \ldots, x_k^{i_k}\right\}$ and $y_i \neq y_j$ correspond to pairwise different elements of S.

Commutative Semigroups have the PLCP

Lemma

Com *has the PLCP.*

Proof.

- Let $S \in \mathbf{Com}$ with $|S| = N$.
- Suppose t is in the subsemigroup generated by X.
- Then $t = x_1^{i_1} \cdots x_k^{i_k}$ for some $k \leq \log(N)$, some $x_1, \ldots, x_k \in X$ and $i_1, \ldots, i_k \in \{1, \ldots, N\}$.
- To see this, note that if k is minimal, then all products of the form $y_1 \cdots y_\ell$ with $y_i \in \left\{x_1^{i_1}, \ldots, x_k^{i_k}\right\}$ and $y_i \neq y_j$ correspond to pairwise different elements of S.
- $t = x_1^{i_1} \cdots x_k^{i_k}$ can be rewritten as an SLP of size $\log N$. □

The Complexity Landscape of CSM

- For arbitrary semigroups and for nilpotent semigroups, CSM is NL-complete.

The Complexity Landscape of CSM

- For arbitrary semigroups and for nilpotent semigroups, CSM is NL-complete.
- For commutative semigroups and for groups, CSM is “easier”.

The Complexity Landscape of CSM

- For arbitrary semigroups and for nilpotent semigroups, CSM is NL-complete.
- For commutative semigroups and for groups, CSM is "easier".
- Can we identify a maximal variety $\mathbf{V}$ such that $\mathrm{CSM}(\mathbf{V})$ is not NL-hard?

The Complexity Landscape of CSM

- For arbitrary semigroups and for nilpotent semigroups, CSM is NL-complete.
- For commutative semigroups and for groups, CSM is "easier".
- Can we identify a maximal variety $\mathbf{V}$ such that $\mathrm{CSM}(\mathbf{V})$ is not NL-hard? Likely not.

Proposition

If $\mathbf{V}$ *is a variety with* $\mathbf{Com} \subseteq \mathbf{V}$ *and* $\mathbf{G} \subseteq \mathbf{V}$, *then* $\mathbf{N} \subseteq \mathbf{V}$.

Connections to FOLL

- $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \in \mathsf{qAC}^0$.

Connections to FOLL

- $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \in \mathrm{qAC}^0$.
- Earlier approach by Barrington, Kadau, Lange and McKenzie (2001): FOLL instead of qAC^0.

Connections to FOLL

- $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \in \mathrm{qAC}^0$.
- Earlier approach by Barrington, Kadau, Lange and McKenzie (2001): FOLL instead of qAC^0.
- FOLL-algorithms using the PLCP approach?

Connections to FOLL

- $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \in \mathrm{qAC}^0$.
- Earlier approach by Barrington, Kadau, Lange and McKenzie (2001): FOLL instead of qAC^0.
- FOLL-algorithms using the PLCP approach?
- If $\mathbf{C}$ has the PLCP and the SLPs have **bounded width** (when considered as algebraic circuits), then $\mathrm{CSM}(\mathbf{C}) \in$ FOLL.

Connections to FOLL

- $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \in \mathrm{qAC}^0$.
- Earlier approach by Barrington, Kadau, Lange and McKenzie (2001): FOLL instead of qAC^0.
- FOLL-algorithms using the PLCP approach?
- If $\mathbf{C}$ has the PLCP and the SLPs have **bounded width** (when considered as algebraic circuits), then $\mathrm{CSM}(\mathbf{C}) \in$ FOLL.
- **Idea**: repeated squaring

Connections to FOLL

- $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \in \mathrm{qAC}^0$.
- Earlier approach by Barrington, Kadau, Lange and McKenzie (2001): FOLL instead of qAC^0.
- FOLL-algorithms using the PLCP approach?
- If $\mathbf{C}$ has the PLCP and the SLPs have **bounded width** (when considered as algebraic circuits), then $\mathrm{CSM}(\mathbf{C}) \in$ FOLL.
- **Idea**: repeated squaring
- SLPs for **Com** have bounded width, so $\mathrm{CSM}(\mathbf{Com}) \in$ FOLL!

Open Problems

- Are there poly-logarithmic bounded-width SLPs for arbitrary groups? Is $\mathrm{CSM}(\mathbf{G}) \in$ FOLL?

Open Problems

- Are there poly-logarithmic bounded-width SLPs for arbitrary groups? Is $\mathrm{CSM}(\mathbf{G}) \in$ FOLL?
- Prove that $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \notin \mathsf{AC}^0$?

Open Problems

- Are there poly-logarithmic bounded-width SLPs for arbitrary groups? Is $\mathrm{CSM}(\mathbf{G}) \in$ FOLL?
- Prove that $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \notin \mathsf{AC}^0$?
- Find other classes of semigroups for which CSM is in qAC^0.

Open Problems

- Are there poly-logarithmic bounded-width SLPs for arbitrary groups? Is $\mathrm{CSM}(\mathbf{G}) \in$ FOLL?
- Prove that $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \notin \mathsf{AC}^0$?
- Find other classes of semigroups for which CSM is in qAC^0.
- Characterize the semigroups for which CSM is in AC^0.

Open Problems

- Are there poly-logarithmic bounded-width SLPs for arbitrary groups? Is $\mathrm{CSM}(\mathbf{G}) \in$ FOLL?
- Prove that $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \notin \mathsf{AC}^0$?
- Find other classes of semigroups for which CSM is in qAC^0.
- Characterize the semigroups for which CSM is in AC^0.
- Design a "natural" deterministic log-space algorithm for $\mathrm{CSM}(\mathbf{G})$.

Open Problems

- Are there poly-logarithmic bounded-width SLPs for arbitrary groups? Is $\mathrm{CSM}(\mathbf{G}) \in$ FOLL?
- Prove that $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \notin \mathsf{AC}^0$?
- Find other classes of semigroups for which CSM is in qAC^0.
- Characterize the semigroups for which CSM is in AC^0.
- Design a "natural" deterministic log-space algorithm for $\mathrm{CSM}(\mathbf{G})$.
- Is there a natural class of semigroups for which CSM is L-complete (NC^1-complete, ...)?

Open Problems

- Are there poly-logarithmic bounded-width SLPs for arbitrary groups? Is $\mathrm{CSM}(\mathbf{G}) \in$ FOLL?
- Prove that $\mathrm{CSM}(\mathbf{G}), \mathrm{CSM}(\mathbf{Com}) \notin \mathsf{AC}^0$?
- Find other classes of semigroups for which CSM is in qAC^0.
- Characterize the semigroups for which CSM is in AC^0.
- Design a "natural" deterministic log-space algorithm for $\mathrm{CSM}(\mathbf{G})$.
- Is there a natural class of semigroups for which CSM is L-complete (NC^1-complete, ...)?
- What exactly causes hardness of CSM?

Thank you for your attention!