On Finite Monoids over Nonnegative Integer Matrices and Short Killing Words

Stefan Kiefer — University of Oxford

Corto Mascle — ENS Paris-Saclay

February 17th 2020

Matrix mortality

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \qquad B = \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}$$

Matrix mortality

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \qquad B = \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix} \qquad C = \begin{pmatrix} 2 & -1 \\ 2 & -1 \end{pmatrix}$$

Matrix mortality

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \qquad B = \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix} \qquad C = \begin{pmatrix} 2 & -1 \\ 2 & -1 \end{pmatrix}$$

No product of As and Bs can be zero, but

$$CBA = C(BA) = \begin{pmatrix} 2 & -1 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Matrix mortality

Given a finite set of matrices $S = \{M_1, ..., M_p\} \subseteq \mathcal{M}_{n\times n}(\mathbb{Z})$, the problem of deciding if the monoid generated by S contains 0 is

- Undecidable in the general case (Paterson, 1970).

Matrix mortality

Given a finite set of matrices $S = \{M_1, ..., M_p\} \subseteq \mathcal{M}_{n \times n}(\mathbb{Z})$, the problem of deciding if the monoid generated by S contains 0 is

- Undecidable in the general case (Paterson, 1970).
- PSPACE-complete when all entries are non-negative (Kao, Rampersad, Shallit, 2009).

Matrix mortality

Given a finite set of matrices $S = \{M_1, ..., M_p\} \subseteq \mathcal{M}_{n\times n}(\mathbb{Z})$, the problem of deciding if the monoid generated by S contains 0 is

- Undecidable in the general case (Paterson, 1970).
- PSPACE-complete when all entries are non-negative (Kao, Rampersad, Shallit, 2009).
- Polynomial-time when all entries are non-negative and the monoid generated by S is finite :
  One can compute the average matrix, and check if its spectral radius is less than 1, but this does not provide a witness.
  $\rightarrow$ Can one compute a 'short' sequence of matrices $M_1, ..., M_p$ such that $M_1 ... M_p = 0$?

Matrix mortality

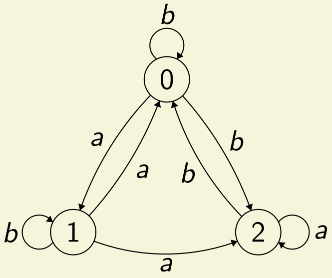
Given a finite set of matrices $S = \{M_1, ..., M_p\} \subseteq \mathcal{M}_{n\times n}(\mathbb{Z})$, the problem of deciding if the monoid generated by S contains 0 is

- Undecidable in the general case (Paterson, 1970).
- PSPACE-complete when all entries are non-negative (Kao, Rampersad, Shallit, 2009).
- Polynomial-time when all entries are non-negative and the monoid generated by S is finite :
  One can compute the average matrix, and check if its spectral radius is less than 1, but this does not provide a witness.

$\rightarrow$ Can one compute a 'short' sequence of matrices $M_{i_1}, ..., M_{i_k}$ such that $M_{i_1}...M_{i_k} = 0$?

Matrix mortality

We can switch from a problem on matrices to a problem on automata :



$\Leftrightarrow$

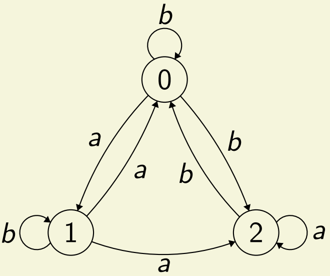
$$M(a) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

$$M(b) = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

$$M(ab) = M(a)M(b) = \begin{pmatrix} 0 & 1 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

Matrix mortality

We can switch from a problem on matrices to a problem on automata :



$$\Leftrightarrow$$

$$M(a) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

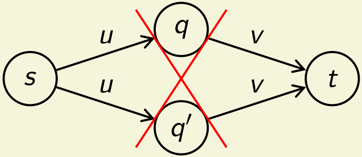
$$M(b) = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

$$M(ab) = M(a)M(b) = \begin{pmatrix} 0 & 1 & 0 \\ 2 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

The *rank* of a word w in $\mathcal{A}$ is the rank of $M(w)$.
A *killing word* is a word of rank 0.

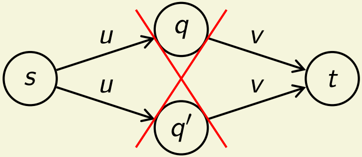
Unambiguous automata

Unambiguous automaton $\rightarrow$ Nondeterministic finite automaton in which for all states s, t, for all word w there is at most one path from s to t labelled by w.
Equivalently, all entries of the matrix monoid associated to this automaton are 0 or 1.



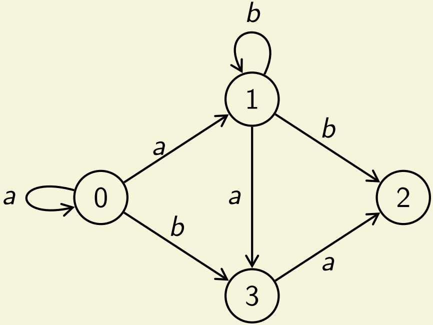
Unambiguous automata

Unambiguous automaton $\rightarrow$ Nondeterministic finite automaton in which for all states s, t, for all word w there is at most one path from s to t labelled by w.
Equivalently, all entries of the matrix monoid associated to this automaton are 0 or 1.



The monoid associated to an unambiguous automaton is finite. The converse is true for strongly connected automata.

Unambiguous automata



$$M(a) = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \; M(b) = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \qquad M(baab) = 0$$

Result

Result

Given an unambiguous automaton with n states one can **compute** a killing word (when there is one) of length at most $\frac{1}{16}n^5 + \frac{15}{16}n^4$ in polynomial time.

Result

Result

Given an unambiguous automaton with n states one can **compute** a killing word (when there is one) of length at most $\frac{1}{16}n^5 + \frac{15}{16}n^4$ in polynomial time.

Main result

Given a set of matrices $S \subseteq \mathcal{M}_{n\times n}(\mathbb{N})$ generating a finite monoid, if this monoid contains 0,then one can **compute** a sequence of matrices $M_1, ..., M_K \in S$ such that $M_1 ... M_K = 0$ in polynomial time, with $K \leq \frac{1}{16}n^5 + \frac{15}{16}n^4$.

When the automaton does not have any killing word, the procedure returns a word of minimal rank.

Result

Main result (generalized version)

Given a set of matrices $S \subseteq \mathcal{M}_{n\times n}(\mathbb{N})$ generating a finite monoid, one can **compute** in polynomial time a sequence of matrices $M_1, ..., M_K \in S$ such that $M_1 ... M_K$ has minimal rank, with $K \leq \frac{1}{16}n^5 + \frac{15}{16}n^4$.

Remarks

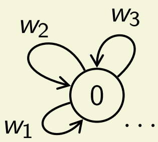
- In 1988, Carpi gave a polynomial-time algorithm to compute minimal-rank words for strongly connected unambiguous automata with positive minimal rank.

Remarks

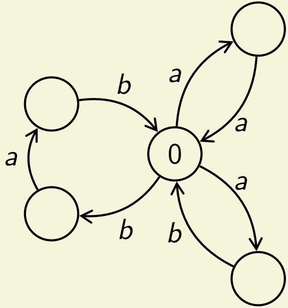
- In 1988, Carpi gave a polynomial-time algorithm to compute minimal-rank words for strongly connected unambiguous automata with positive minimal rank.
- The main contribution here is the extension of Carpi's result to unambiguous automata with minimal rank 0.

Restivo's conjecture

$S = \{w_1, ..., w_p\} \subseteq \Sigma^*$



$S = \{aa, ab, bab\}$



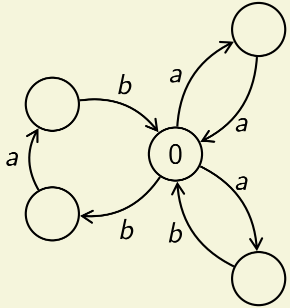
Let $\mathcal{A}(S)$ be the automaton associated to the set of words S.
$\mathcal{A}(S)$ is non-deterministic in general.

Restivo's conjecture

Restivo's conjecture (1981)

$kw(\mathcal{A}(S))$ is bounded by $2m^2$ where $m = max_{w \in S}|w|$

Here $\mathcal{A}(S)$ is not necessarily unambiguous.



$S = \{aa, ab, bab\}$

The word *bbb* is a killing word for this automaton.

History

1981 $\rightarrow$ Restivo's conjecture ($2m^2$ upper bound)

2010 $\rightarrow$ Numerical counterexample by Fici, Pribavnika and Sakarovitch

2011 $\rightarrow$ Family of counterexamples by Gusev and Pribavnika ($5m^2$ lower bound)

2017 $\rightarrow$ Computations hinting at an exponential growth in m by Julia, Malapert and Provillard

2019 $\rightarrow$ Superpolynomial lower bound by Mika and Szykuła ($2^{\frac{m}{4}}$)

Final disproval

Theorem (Maksymilian Mika, Marek Szykuła)

The following problem is PSPACE-complete:
Input: A finite set S of words
Output: $\Sigma^* = S^*$?

While proving this result the authors constructed a family of sets of words whose minimal uncompletable word is of superpolynomial size in the maximal length of their elements.

A particular case

Rather than any finite set of words, we can restrict ourselves to finite codes.

Definition

A code is a set of words S such that for all $u_1, \ldots, u_n, v_1, \ldots, v_p \in S$, if $u_1 \cdots u_n = v_1 \cdots v_p$ then $n = p$ and $u_i = v_i$ for all i.

$S = \{aa, aba\}$ is a code.

A particular case

When S is a finite code, $\mathcal{A}(S)$ is unambiguous (the converse is true).

A version of Restivo's conjecture in codes

When S is a code, $kw(\mathcal{A}(S))$ is polynomially bounded in terms of $m = max_{w \in S}|w|$

This has been shown for some particular cases such as prefix codes (Néraud, Selmi, 1988).

A particular case

When S is a finite code, $\mathcal{A}(S)$ is unambiguous (the converse is true).

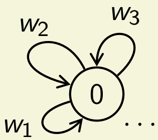
A version of Restivo's conjecture in codes

When S is a code, $kw(\mathcal{A}(S))$ is polynomially bounded in terms of $m = max_{w \in S}|w|$

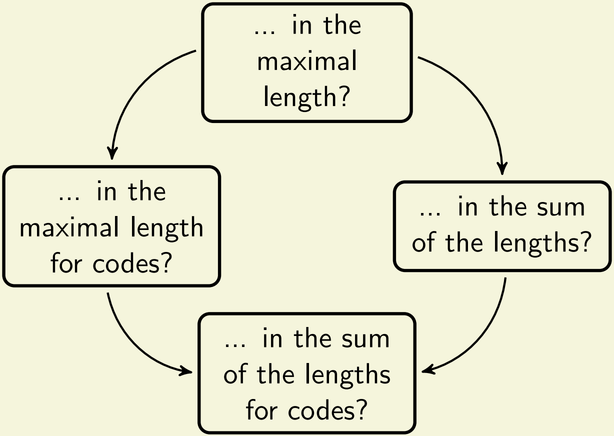
This has been shown for some particular cases such as prefix codes (Néraud, Selmi, 1988).

Weaker version of the conjecture

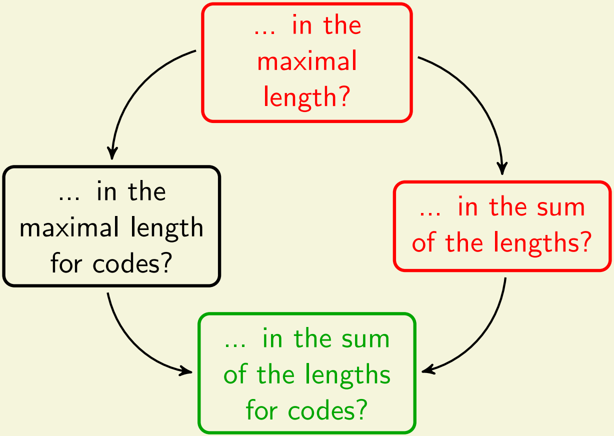
When S is a code, $kw(\mathcal{A}(S))$ is polynomially bounded in terms of the number of states n in $\mathcal{A}(S)$



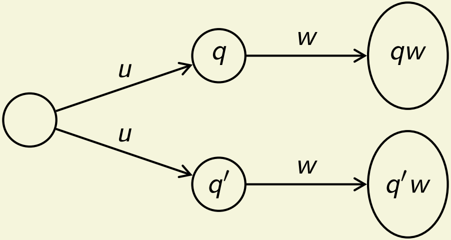
Is there always a killing word of length polynomial in ...



Is there always a killing word of length polynomial in ...

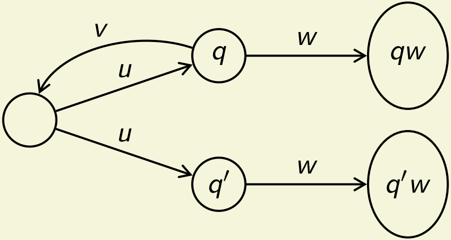


Proof for strongly connected UFA



qw and $q'w$ are disjoint because of unambiguity.

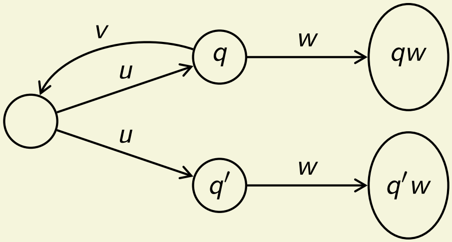
Proof for strongly connected UFA



qw and $q'w$ are disjoint because of unambiguity.

v exists as we assumed strong connectedness.

Proof for strongly connected UFA



qw and $q'w$ are disjoint because of unambiguity.

v exists as we assumed strong connectedness.

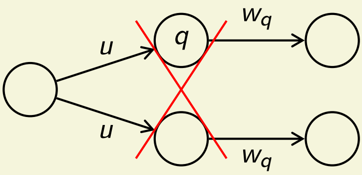
$qw \sqcup q'w \subseteq q(vuw)$

Proof for strongly connected UFA

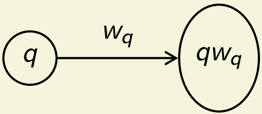
Lemma

For all state q we can compute a word w_q such that for all states $q' \neq q$, if q and q' are coaccessible then either $qw = \emptyset$ or $q'w = \emptyset$.

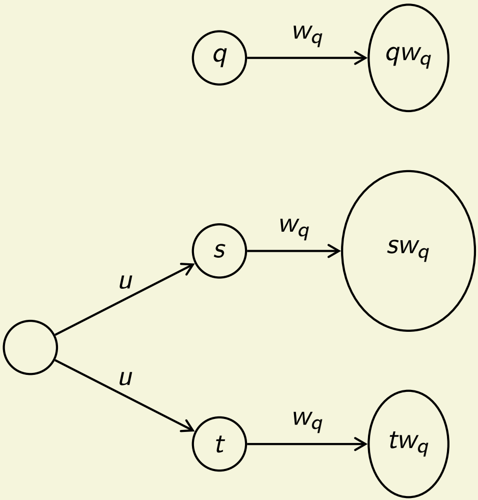
For all q there exists w_q such that



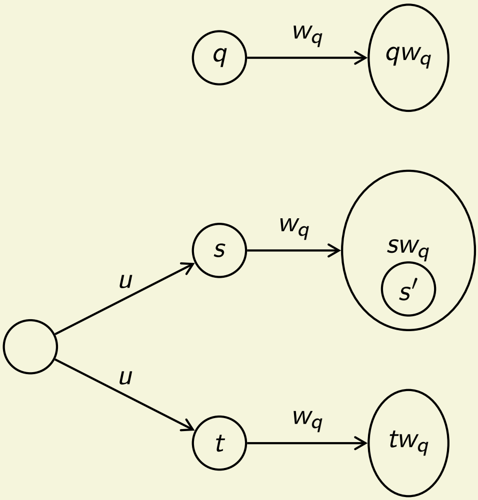
Proof for strongly connected UFA



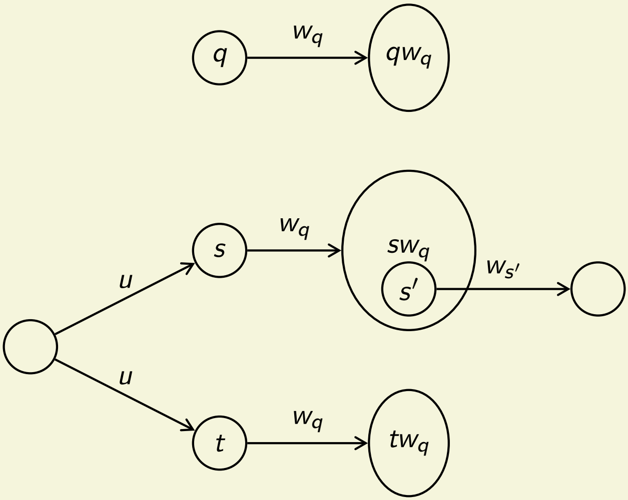
Proof for strongly connected UFA



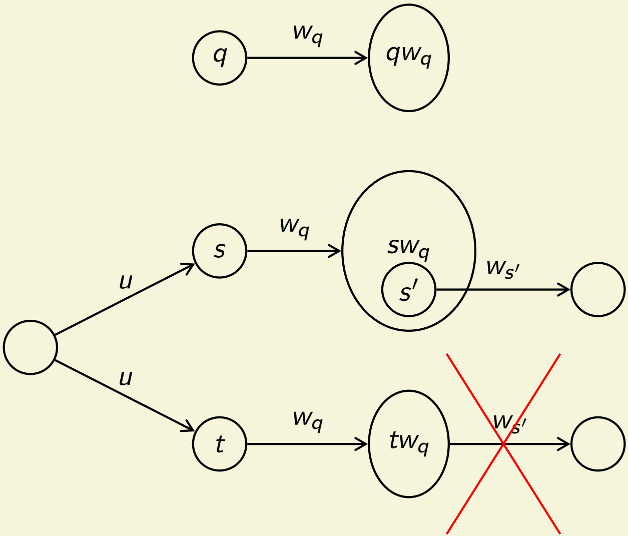
Proof for strongly connected UFA



Proof for strongly connected UFA



Proof for strongly connected UFA

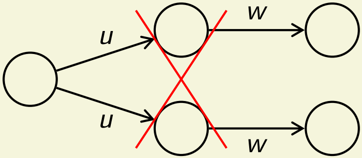


Proof for strongly connected UFA

Lemma

We can compute a word w such that for all states $q \neq q'$, if q and q' are coaccessible then either $\delta(q, w) = \emptyset$ or $\delta(q', w) = \emptyset$.

There exists w such that

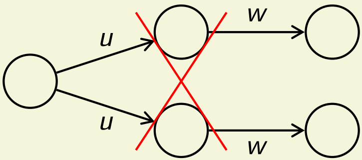


Proof

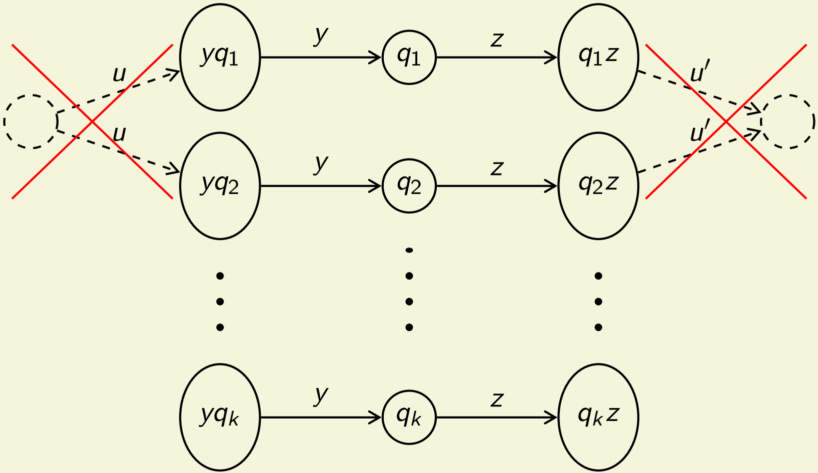
Lemma

We can compute a word w such that for all states $q \neq q'$, if q and q' are coaccessible then either $\delta(q, w) = \emptyset$ or $\delta(q', w) = \emptyset$.

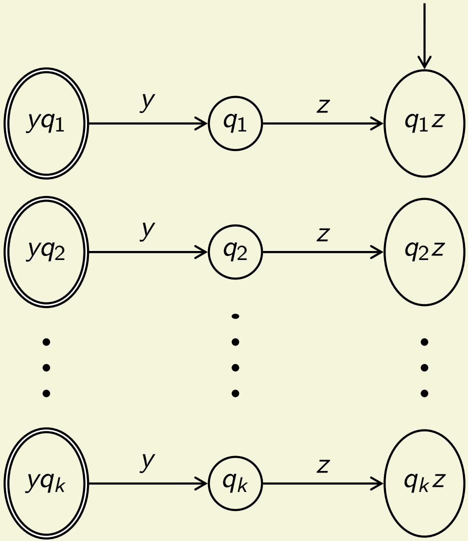
There exists w such that



Idea of the proof



Idea of the proof

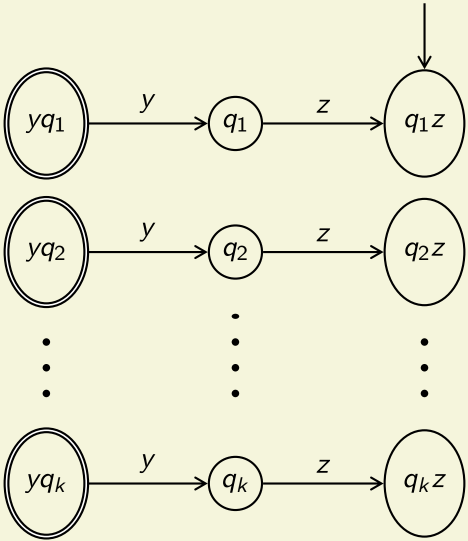


Proof for strongly connected UFA

Lemma

Given a non universal automaton $\mathcal{A}$ with n states such that any word has at most one accepting run, one can compute in polynomial time a word $w \notin L(\mathcal{A})$, with $|w| \leq n$.

Idea of the proof



A counter-example

Main result (again)

Given an unambiguous automaton $\mathcal{A}$, one can compute a killing word w in polynomial time, if it exists, with $|w| \leq \frac{1}{16}n^5 + \frac{15}{16}n^4$

We now know there exists a minimal-rank matrix which is a product of polynomially many matrices of S.

A counter-example

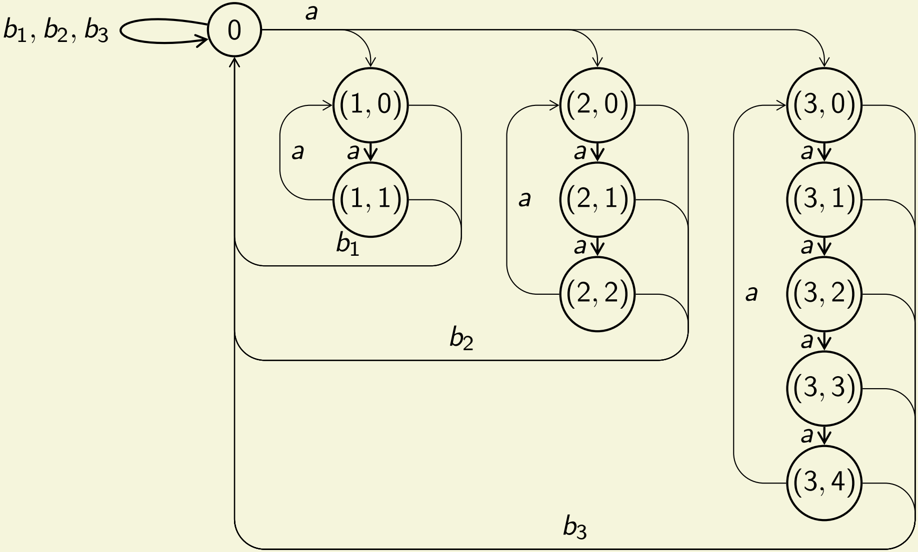
Main result (again)

Given an unambiguous automaton $\mathcal{A}$, one can compute a killing word w in polynomial time, if it exists, with $|w| \leq \frac{1}{16}n^5 + \frac{15}{16}n^4$

We now know there exists a minimal-rank matrix which is a product of polynomially many matrices of S.

Are all minimal-rank matrices products of polynomially many matrices of S ?

A counter-example



Conclusion

- Applications in the theory of codes
- Some work to improve the degree of the bound
- Possible extensions to finite monoids of integer matrices for instance.