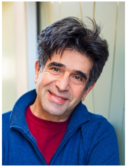
Two-Dimensional Automata Theory: Decidability, Complexity, and Algorithms

One FLAT World Seminar

Taylor J. Smith

Department of Computer Science
St. Francis Xavier University
Antigonish, Nova Scotia, Canada

April 9, 2025



Pantelis Andreou
Dalhousie

Da-Jung Cho
Ajou

Guilhem Gamard
Lorraine

Yo-Sub Han
Yonsei

Stavros Konstantinidis
Saint Mary's

Alastair May
StFX

Gwenaël Richomme
Paul-Valéry

Arto Salomaa
Turku

Kai Salomaa
Queen's

Jeffrey Shallit
Waterloo

Table of Contents

Table of Contents

Two-Dimensional Automata

- A **two-dimensional** (2D) **automaton** is a generalization of a one-dimensional automaton.
- Two major differences:
  1. Different input word
  2. Different transition function

Two-Dimensional Automata

- A **two-dimensional** (2D) **automaton** is a generalization of a one-dimensional automaton.
- Two major differences:
  1. **Different input word**
  2. Different transition function

$$\begin{array}{cccccc}
\# & \# & \# & \cdots & \# & \# \\
\# & a_{1,1} & a_{1,2} & \cdots & a_{1,n} & \# \\
\# & a_{2,1} & a_{2,2} & \cdots & a_{2,n} & \# \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\# & a_{m,1} & a_{m,2} & \cdots & a_{m,n} & \# \\
\# & \# & \# & \cdots & \# & \#
\end{array}$$

Two-Dimensional Automata

- A **two-dimensional** (2D) **automaton** is a generalization of a one-dimensional automaton.
- Two major differences:
  1. Different input word
  2. **Different transition function**

$$\delta : (Q \setminus q_{\text{accept}}) \times (\Sigma \cup \{\#\}) \rightarrow Q \times \{U, D, L, R\}$$

Deterministic four-way (2DFA-4W)

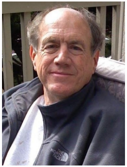
$$\delta : (Q \setminus q_{\text{accept}}) \times (\Sigma \cup \{\#\}) \rightarrow 2^{Q \times \{U,D,L,R\}}$$

Nondeterministic four-way (2NFA-4W)

Two-Dimensional Automata: History

- 2D automata were introduced by Manuel Blum and Carl Hewitt in 1967.

M. Blum



C. Hewitt

- Work on 2D automata has progressed in "waves" since the introduction of the model.

M. Blum and C. Hewitt. Automata on a 2-dimensional tape. In *Proc. of SWAT 1967*, pages 155–160, 1967.

Two-Dimensional Automata: Properties

- 2D automata possess a number of useful properties.

Theorem
Nondeterministic 2D automata are more powerful than deterministic 2D automata.

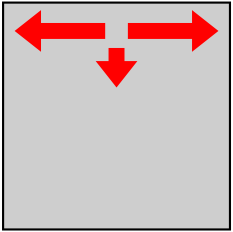
Theorem
Every deterministic 2D automaton can be converted to a halting deterministic 2D automaton.

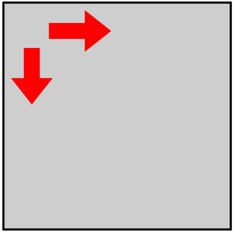
Two-Dimensional Automata: Examples

- Much is known about the kinds of languages recognized by 2D automata.
- Deterministic 2D automata:
  - Given an $m \times n$ input word, does the word contain exactly k occurrences of a given symbol?
  - Given an $m \times n$ input word, are m and n coprime?
  - Given an $n \times n$ input word, is n a power of two?
- Nondeterministic 2D automata:
  - Given an $n \times n$ input word where n is odd, does the word contain a 1 as its center symbol?
  - Given an $n \times n$ input word where n is odd, is the word symmetric about its center column?
- Unknown:
  - Can deterministic 2D automata recognize the language of unary $p \times p$ words where p is prime?

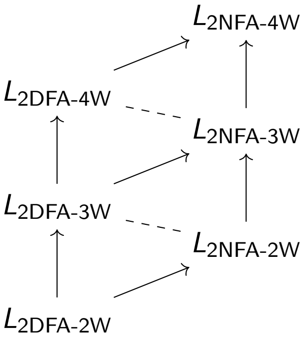
Restricted 2D Automata

- 2D automata do not have to be **four-way automata**.
  - In fact, four-way automata can sometimes be undesirable, since they're Turing-equivalent.
- Restrict the transition function to get:
  - **Three-way (3W) automata**: $\{D, L, R\}$
  - **Two-way (2W) automata**: $\{D, R\}$
- Three-way automata cannot return to a row after moving downward, but they can read symbols multiple times in a row.
- Two-way automata are "read-once".





2D Automata Hierarchy



$L_A \rightarrow L_B$ indicates $L_A \subset L_B$.

L_A - - L_B indicates L_A and L_B are incomparable.

Table of Contents

Decision Problems

- Many of the classic decision problems for 1D languages can be adapted for 2D languages as well.
- Some common decision problems for two 2D languages $L(\mathcal{A})$ and $L(\mathcal{B})$:
  - **Membership:** $w \in L(\mathcal{A})$ for some 2D word w
  - **Emptiness:** $L(\mathcal{A}) = \emptyset$
  - **Universality:** $L(\mathcal{A}) = \Sigma^{**}$ (the set of all 2D words)
  - **Equivalence:** $L(\mathcal{A}) = L(\mathcal{B})$
  - **Inclusion:** $L(\mathcal{A}) \subseteq L(\mathcal{B})$
  - **Disjointness:** $L(\mathcal{A}) \cap L(\mathcal{B}) = \emptyset$

Decision Problems: Decidability

	2DFA-4W	2NFA-4W	2DFA-3W	2NFA-3W	2DFA-2W	2NFA-2W
membership	✓	✓	✓	✓	✓	✓
emptiness	✗	✗	✓	✓	✓	✓
universality	✗	✗	✓	✗	✓	✗
equivalence	✗	✗	?	✗	✓	✗
inclusion	✗	✗	✗	✗	✓	✗
disjointness	✗	✗	✗	✗	✓	?

T. J. Smith and K. Salomaa. Decision problems and projection languages for restricted variants of two-dimensional automata. *Theoret. Comput. Sci.* 870:153–164, 2021.

Decision Problems: Decidability

	2DFA-4W	2NFA-4W	2DFA-3W	2NFA-3W	2DFA-2W	2NFA-2W
membership	✓	✓	✓	✓	✓	✓
emptiness	✗	✗	✓	✓	✓	✓
universality	✗	✗	✓	✗	✓	✗
equivalence	✗	✗	?	✗	✓	✗
inclusion	✗	✗	✗	✗	✓	✗
disjointness	✗	✗	✗	✗	✓	?

Open problems: Are the question marks ✓ or ✗?

T. J. Smith and K. Salomaa. Decision problems and projection languages for restricted variants of two-dimensional automata. *Theoret. Comput. Sci.* 870:153–164, 2021.

Table of Contents

Language Operations

- We can also apply the standard 1D language operations to 2D languages.
- Some of these operations can be applied as-is:
  - **Union:** $L_1 \cup L_2$
  - **Intersection:** $L_1 \cap L_2$
  - **Complement:** $\overline{L}$

Language Operations

- We can also apply the standard 1D language operations to 2D languages.
- Some of these operations can be applied as-is:
  - **Union:** $L_1 \cup L_2$
  - **Intersection:** $L_1 \cap L_2$
  - **Complement:** $\overline{L}$
- Other operations must be adapted to two dimensions:
  - **Concatenation:** $L_1 \circ L_2$ places all words in L_1 *adjacent to* all words in L_2 in some way
  - **Reversal:** L^{R} reverses the order of the rows in all words of L

Language Operations

- We can also apply the standard 1D language operations to 2D languages.
- Some of these operations can be applied as-is:
  - **Union:** $L_1 \cup L_2$
  - **Intersection:** $L_1 \cap L_2$
  - **Complement:** $\overline{L}$
- Other operations must be adapted to two dimensions:
  - **Concatenation:** $L_1 \circ L_2$ places all words in L_1 *adjacent to* all words in L_2 in some way
  - **Reversal:** L^{R} reverses the order of the rows in all words of L
- Still other operations are unique to two dimensions:
  - **Rotation:** $L^{\circlearrowright}$ rotates all words in L by 90° clockwise
  - **Row Closure:** $L^{\ominus}$ concatenates L with itself row-wise $i \geq 1$ times.
  - **Row Cyclic Closure:** Rearrange the top k rows of each word in L to be shifted to the bottom for some $1 \leq k \leq \#$ of rows.

Language Operations: Set-Theoretic Closure

	2DFA-4W	2NFA-4W	2DFA-3W	2NFA-3W	2DFA-2W	2NFA-2W
$\cup$	✓	✓	✗	✓	✗	✓
$\cap$	✓	✓	✗	✗	✗	✗
$\neg$	✓	✗	✓	✗	✓	✗

T. J. Smith and K. Salomaa. Recognition and complexity results for projection languages of two-dimensional automata. *J. Autom. Lang. Comb.* 28(1–3):201–220, 2023.

Concatenation

- Let's focus on "the" concatenation operation $L_1 \circ L_2$.
- We can concatenate 2D words in two different ways: row-wise or column-wise.

Concatenation

- Let's focus on "the" concatenation operation $L_1 \circ L_2$.
- We can concatenate 2D words in two different ways: **row-wise** or column-wise.

$$w \ominus v = \begin{matrix} w_{1,1} & \cdots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{m,1} & \cdots & w_{m,n} \\ v_{1,1} & \cdots & v_{1,n} \\ \vdots & \ddots & \vdots \\ v_{m',1} & \cdots & v_{m',n} \end{matrix}$$

Concatenation

- Let's focus on "the" concatenation operation $L_1 \circ L_2$.
- We can concatenate 2D words in two different ways: row-wise or **column-wise**.

$$w \mathbin{⦶} v = \begin{matrix} w_{1,1} & \cdots & w_{1,n} & v_{1,1} & \cdots & v_{1,n'} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ w_{m,1} & \cdots & w_{m,n} & v_{m,1} & \cdots & v_{m,n'} \end{matrix}$$

Concatenation

- Let's focus on "the" concatenation operation $L_1 \circ L_2$.
- We can also concatenate two 2D words **diagonally**.

$$w \oslash v = \begin{matrix} w_{1,1} & \cdots & w_{1,n} & x_{1,1} & \cdots & x_{1,n'} \\ \vdots & & \vdots & \vdots & & \vdots \\ w_{m,1} & \cdots & w_{m,n} & x_{m,1} & \cdots & x_{m,n'} \\ y_{1,1} & \cdots & y_{1,n} & v_{1,1} & \cdots & v_{1,n'} \\ \vdots & & \vdots & \vdots & & \vdots \\ y_{m',1} & \cdots & y_{m',n} & v_{m',1} & \cdots & v_{m',n'} \end{matrix}$$

Language Operations: Concatenation Closure

	2DFA-4W	2NFA-4W	2DFA-3W	2NFA-3W	2DFA-2W	2NFA-2W
$\cup$	✓	✓	✗	✓	✗	✓
$\cap$	✓	✓	✗	✗	✗	✗
$\neg$	✓	✗	✓	✗	✓	✗
$\ominus / \varobar$	✗	✗	✗	$✓_{\ominus}$ $✗_{\varobar}$	✗	✗
$\oslash$	?	?	✗	?	✗	✓

(2NFA-2W is closed under $\ominus$ and $\varobar$ for unary alphabets.)

T. J. Smith and K. Salomaa. Concatenation operations and restricted variants of two-dimensional automata. In *Proc. of SOFSEM 2021*, pages 147–158, 2021.

Language Operations: Concatenation Closure

	2DFA-4W	2NFA-4W	2DFA-3W	2NFA-3W	2DFA-2W	2NFA-2W
$\cup$	✓	✓	✗	✓	✗	✓
$\cap$	✓	✓	✗	✗	✗	✗
$\neg$	✓	✗	✓	✗	✓	✗
$\ominus$/⦶	✗	✗	✗	✓$_{\ominus}$ ✗$_{⦶}$	✗	✗
$\oslash$	?	?	✗	?	✗	✓

(2NFA-2W is closed under $\ominus$ and ⦶ for unary alphabets.)

Open problems: Are the question marks ✓ or ✗?

T. J. Smith and K. Salomaa. Concatenation operations and restricted variants of two-dimensional automata. In *Proc. of SOFSEM 2021*, pages 147–158, 2021.

Language Operations: Other Closure Results

	2DFA-4W	2NFA-4W	2DFA-3W	2NFA-3W	2DFA-2W	2NFA-2W
$\cup$	✓	✓	✗	✓	✗	✓
$\cap$	✓	✓	✗	✗	✗	✗
$\neg$	✓	✗	✓	✗	✓	✗
$\ominus$/$\obar$	✗	✗	✗	✓$_{\ominus}$ ✗$_{\obar}$	✗	✗
$\oslash$	?	?	✗	?	✗	✓
R	✓	✓	✗	✓	(✗)	(✗)
$\circlearrowright$	✓	✓	✗	✗	(✗)	(✗)
row/column closure	✗	✗	✗	✓$_R$ ✗$_C$	(✗)	(✗)
row/column cyclic closure	✗	✗	✗	✗	(✗)	(✗)

T. J. Smith. *Closure, Decidability, and Complexity Results for Restricted Variants of Two-Dimensional Automata*. Doctoral thesis, Queen's University, 2021.

Projection Operations

- We can **project** 2D words onto one dimension to produce classical string languages.
- The **row/column projection** of a 2D language L is the 1D language consisting of all first rows/first columns of all 2D words in L.

$$w = \begin{matrix} w_{1,1} & \cdots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{m,1} & \cdots & w_{m,n} \end{matrix}$$

$$\mathrm{pr}_{\mathrm{R}}(w) = w_{1,1} w_{1,2} \cdots w_{1,n}$$
$$\mathrm{pr}_{\mathrm{C}}(w) = w_{1,1} w_{2,1} \cdots w_{m,1}$$

Projection Operations: Space Complexity

	$\mathcal{A}$	$\text{pr}_R(L(\mathcal{A}))$	$\text{pr}_C(L(\mathcal{A}))$
General	-4W	$\text{NSPACE}(O(n))$	$\text{NSPACE}(O(n))$
	-3W	$\text{DSPACE}(O(1))$	**?**
	-2W	$\text{DSPACE}(O(1))$	$\text{DSPACE}(O(1))$
Unary	-4W	**?**	**?**
	-3W	$\text{DSPACE}(O(1))$	$\leq \text{NSPACE}(O(\log(n)))$
	-2W	$\text{DSPACE}(O(1))$	$\text{DSPACE}(O(1))$

- Recall that:
  - $\text{REG} = \text{DSPACE}(O(1))$.
  - $\text{CSL} = \text{NSPACE}(O(n))$.

T. J. Smith and K. Salomaa. Recognition and complexity results for projection languages of two-dimensional automata. *J. Autom. Lang. Comb.* 28(1–3):201–220, 2023.

Projection Operations: Space Complexity

	$\mathcal{A}$	$\text{pr}_R(L(\mathcal{A}))$	$\text{pr}_C(L(\mathcal{A}))$
General	-4W	NSPACE($O(n)$)	NSPACE($O(n)$)
	-3W	DSPACE($O(1)$)	**?**
	-2W	DSPACE($O(1)$)	DSPACE($O(1)$)
Unary	-4W	**?**	**?**
	-3W	DSPACE($O(1)$)	$\leq$ NSPACE($O(\log(n))$)
	-2W	DSPACE($O(1)$)	DSPACE($O(1)$)

- Recall that:
  - REG = DSPACE($O(1)$).
  - CSL = NSPACE($O(n)$).

Open problems: What is the space complexity of each of the question mark entries?

T. J. Smith and K. Salomaa. Recognition and complexity results for projection languages of two-dimensional automata. *J. Autom. Lang. Comb.* 28(1–3):201–220, 2023.

Table of Contents

State Complexity

- Why should we care about projections from 2D to 1D?
- Observe all of the 2D projection language classes that are in DSPACE($O(1)$):
  - 2DFA-3W row projection
  - 2DFA-3W-1Σ row projection
  - 2DFA-2W and 2NFA-2W row/column projection
  - 2DFA-2W-1Σ and 2NFA-2W-1Σ row/column projection
- Since each of these projection languages is regular, we can apply standard techniques and obtain **state complexity** results for these languages.

- State complexity tradeoff:
  - n-state 2NFA-2W $\to$ NFA:
    $2n-1 \leq \text{nsc}(\cdot) \leq 2n$
- Operational state complexity:
  - $\text{pr}_{\text{R}}(L(\mathcal{A}) \cup L(\mathcal{B}))$ for 2NFA-2W:
    $2(m+n-1) \leq \text{nsc}(\cdot) \leq 2(m+n+1)$
  - $\text{pr}_{\text{R}}(L(\mathcal{A}) \oslash L(\mathcal{B}))$ for 2NFA-2W:
    $m+n-1 \leq \text{nsc}(\cdot) \leq 2m+n$

T. J. Smith and K. Salomaa. Recognition and complexity results for projection languages of two-dimensional automata. *J. Autom. Lang. Comb.* 28(1–3):201–220, 2023.

State Complexity

- State complexity tradeoff:
  - n-state 2NFA-2W $\to$ NFA:
    $2n-1 \leq \mathrm{nsc}(\cdot) \leq 2n$
- Operational state complexity:
  - $\mathrm{pr}_{\mathrm{R}}(L(\mathcal{A}) \cup L(\mathcal{B}))$ for 2NFA-2W:
    $2(m+n-1) \leq \mathrm{nsc}(\cdot) \leq 2(m+n+1)$
  - $\mathrm{pr}_{\mathrm{R}}(L(\mathcal{A}) \oslash L(\mathcal{B}))$ for 2NFA-2W:
    $m+n-1 \leq \mathrm{nsc}(\cdot) \leq 2m+n$

Open problem: Can these bounds be tightened?

T. J. Smith and K. Salomaa. Recognition and complexity results for projection languages of two-dimensional automata. *J. Autom. Lang. Comb.* 28(1–3):201–220, 2023.

State Complexity

- State complexity tradeoff:
  - n-state 2NFA-2W $\rightarrow$ NFA:
    $2n-1 \leq \text{nsc}(\cdot) \leq 2n$
- Operational state complexity:
  - $\text{pr}_{\text{R}}(L(\mathcal{A}) \cup L(\mathcal{B}))$ for 2NFA-2W:
    $2(m+n-1) \leq \text{nsc}(\cdot) \leq 2(m+n+1)$
  - $\text{pr}_{\text{R}}(L(\mathcal{A}) \oslash L(\mathcal{B}))$ for 2NFA-2W:
    $m+n-1 \leq \text{nsc}(\cdot) \leq 2m+n$

Open problem: Can these bounds be tightened?

Open problem: What bounds exist for other 2D language operations and models?

T. J. Smith and K. Salomaa. Recognition and complexity results for projection languages of two-dimensional automata. *J. Autom. Lang. Comb.* 28(1–3):201–220, 2023.

Table of Contents

Decision Problems Revisited

- Recall some decision problems for 2D automaton models:
  - 2DFA-4W: emptiness undecidable, universality undecidable.
  - 2NFA-4W: emptiness undecidable, universality undecidable.
  - 2DFA-3W: emptiness **decidable**, universality **decidable**.
  - 2NFA-3W: emptiness **decidable**, universality undecidable.
  - 2DFA-2W: emptiness **decidable**, universality **decidable**.
  - 2NFA-2W: emptiness **decidable**, universality undecidable.

Decision Problems Revisited

- Recall some decision problems for 2D automaton models:
  - 2DFA-4W: emptiness undecidable, universality undecidable.
  - 2NFA-4W: emptiness undecidable, universality undecidable.
  - 2DFA-3W: emptiness **decidable**, universality **decidable**.
  - 2NFA-3W: emptiness **decidable**, universality undecidable.
  - 2DFA-2W: emptiness **decidable**, universality **decidable**.
  - 2NFA-2W: emptiness **decidable**, universality undecidable.
- For every 2D automaton model, membership is decidable.
  - In fact, membership is in NL.

K. Lindgren et al. Complexity of two-dimensional patterns. *J. Stat. Phys.* 91(5/6):909–951, 1998.

Decision Problems Revisited

- Recall some decision problems for 2D automaton models:
  - 2DFA-4W: emptiness undecidable, universality undecidable.
  - 2NFA-4W: emptiness undecidable, universality undecidable.
  - 2DFA-3W: emptiness **decidable**, universality **decidable**.
  - 2NFA-3W: emptiness **decidable**, universality undecidable.
  - 2DFA-2W: emptiness **decidable**, universality **decidable**.
  - 2NFA-2W: emptiness **decidable**, universality undecidable.
- For every 2D automaton model, membership is decidable.
  - In fact, membership is in NL.
- However, emptiness and universality for restricted 2D automata are PSPACE-hard.

T. J. Smith. *Closure, Decidability, and Complexity Results for Restricted Variants of Two-Dimensional Automata*. Doctoral thesis, Queen's University, 2021.

PRAX Algorithms

- How might we get answers to these decision problems more efficiently?
  - Use **randomization** and **approximation**!

PRAX Algorithms

- How might we get answers to these decision problems more efficiently?
  - Use **randomization** and **approximation**!
- **Polynomial randomized approximation** (PRAX) **algorithms** were introduced by Konstantinidis et al. to decide approximate versions of NFA decision problems.
- Key idea:
  - Treat the decision problem as an estimation of the parameter of some population.
  - Use existing parameter estimation tools to obtain approximate solutions.

S. Konstantinidis et al. Approximate NFA universality and related problems motivated by information theory. *Theoret. Comput. Sci.* 972:114076, 2023.

PRAX Algorithms

- Goal:
  - Take a (possibly infinite) subset L of an infinite domain X and test whether L is ϵ-close to being empty or full for some $\epsilon \in (0, 1)$.

PRAX Algorithms

- Goal:
  - Take a (possibly infinite) subset L of an infinite domain X and test whether L is ϵ-close to being empty or full for some $\epsilon \in (0,1)$.
- Technical points:
  - How do we sample from a finite distribution?
    Take the distribution to be **polynomially samplable**.
  - How do we sample from an infinite distribution?
    Take the distribution to be **tractable**: use an algorithm to "cut" the infinite tail such that the remaining finite events can be sampled within a tolerance δ of the infinite distribution.
  - How many samples are sufficient?
    A linear amount relative to $1/\delta$. (Previously quadratic!)

P. Andreou, S. Konstantinidis, and T. J. Smith. Improved randomized approximation of hard universality and emptiness problems. *J. Autom. Lang. Comb.* To appear.

PRAX Algorithms: Applications

- Where have PRAX algorithms been used?
  - Deciding approximate NFA universality.

S. Konstantinidis et al. Approximate NFA universality and related problems motivated by information theory. *Theoret. Comput. Sci.* 972:114076, 2023.

PRAX Algorithms: Applications

- Where have PRAX algorithms been used?
  - Deciding approximate NFA universality.
  - Deciding approximate NFA (in)equivalence.

S. Konstantinidis et al. On the difference set of two transductions. *Theoret. Comput. Sci.* 1016:114780, 2024.

PRAX Algorithms: Applications

- Where have PRAX algorithms been used?
  - Deciding approximate NFA universality.
  - Deciding approximate NFA (in)equivalence.
  - Deciding block NFA universality.
  - Deciding 2D automaton emptiness and universality.
  - Testing whether a CNF formula is a tautology.
  - Testing whether a Diophantine equation has integer solutions.

P. Andreou, S. Konstantinidis, and T. J. Smith. Improved randomized approximation of hard universality and emptiness problems. *J. Autom. Lang. Comb.* To appear.

PRAX Algorithms: Applications

- Where have PRAX algorithms been used?
  - Deciding approximate NFA universality.
  - Deciding approximate NFA (in)equivalence.
  - Deciding block NFA universality.
  - **Deciding 2D automaton emptiness and universality.**
  - Testing whether a CNF formula is a tautology.
  - Testing whether a Diophantine equation has integer solutions.

Theorem

There exists a PRAX algorithm for the 2D emptiness and universality decision problems (relative to a "2D word" Dirichlet distribution $\langle \mathrm{D}^2_{t,d} \rangle$) that runs in time $O(1/\epsilon \cdot \sqrt[t-1]{1/\epsilon^2} \cdot s)$, where s is the number of states of the input 2D automaton.

P. Andreou, S. Konstantinidis, and T. J. Smith. Improved randomized approximation of hard universality and emptiness problems. *J. Autom. Lang. Comb.* To appear.

PRAX Algorithms: Applications

- Where have PRAX algorithms been used?
  - Deciding approximate NFA universality.
  - Deciding approximate NFA (in)equivalence.
  - Deciding block NFA universality.
  - Deciding 2D automaton emptiness and universality.
  - Testing whether a CNF formula is a tautology.
  - Testing whether a Diophantine equation has integer solutions.

Open problem: Where else can we use PRAX algorithms?

Table of Contents

Conclusions

- 2D automata are a natural extension of the finite automaton model, with many different variants or "flavours" possessing different properties.
- Almost no problems are decidable for four-way 2D automata, but more problems are decidable for three- and two-way variants.
- Some language operations have positive closure results for four-way 2D automata, while almost no operations are closed for two-way 2D automata.
- Projection operations allow us to "convert" 2D languages to 1D and apply standard techniques (e.g., state complexity).
- We can obtain approximate solutions to 2D decision problems using PRAX algorithms.

Future Work

- Resolve the decidability status of the remaining decision problems.
- Resolve the closure status of diagonal concatenation for all 2D models.
- Determine the space complexity of other 2D projection language classes.
- Investigate state complexity bounds for other 2D language operations and models.
- Investigate applications of PRAX algorithms.
- **Lots to be done with 2D automata!**

Shameless Advertisement

An open educational resource for undergrad and grad courses in theoretical computer science

- Regular languages
- Context-free languages
- Decidable/semidecidable languages
- Decision problems
- Proving undecidability
- Foundations of complexity
- Time and space complexity
- Hardness, completeness, and complements
- *And more!*

Available for free, forever, at `taylorjsmith.xyz/tocopen/`

Thank you!

StFXCS

FLAReLab

Formal Languages and Automata Research Lab
St. Francis Xavier University

taylorjsmith.xyz   taylorjsmith.xyz/flarelab/