Computation Theory over Sets with Atoms

Bartek Klin
University of Oxford

MOVEP Summer School 2022, Aalborg
**Puzzle 1: A graph**

- **nodes:** ordered pairs of distinct natural numbers
  \[
  \{ (n, m) \mid n \neq m \in \mathbb{N} \}
  \]

- **edges (undirected):**
  \[
  \{ (n, m) \sim (m, k) \mid n \neq k \}
  \]

Is it 3-colorable?
Puzzle II : linear equations

- variables: ordered pairs of distinct natural numbers
  \[
  \{ (nm) \mid n \neq m \in \mathbb{N} \}
  \]

- equations:
  \[
  nm + mk + kn = 0
  \]
  \[
  12 + 21 = 1
  \]

Does it have a solution in \( \mathbb{Z}_2 \)?
General theme

Replace *finite* structures with *infinite, but highly symmetric* ones in:

- automata theory
- computability theory
- modelling / verification
- algorithms

...  

- all the way down to *set theory*
Plan

1. Register automata

2. Sets with atoms

3. $\mu$-calculus with atoms
   -- Turing machines with atoms
   -- Constraint satisfaction problems with atoms
   -- Programming with atoms
Register automata
A finite automaton is:
- a set $Q$ of states
- an alphabet $\Sigma$
- initial state $q_0 \in Q$, accepting states $F \subseteq Q$
- transition function $\delta : Q \times \Sigma \rightarrow Q$

(or relation $\delta \subseteq Q \times \Sigma \times Q$)

Example language: \[ \bigcup_{a \in \Sigma} a(\Sigma \setminus a)^* \]
What about infinite alphabets?

**Idea 1**: keep the definition as it is
- a set $Q$ of states
- an alphabet $\Sigma$
- initial state $q_0 \in Q$, accepting states $F \subseteq Q$
- transition function $\delta : Q \times \Sigma \rightarrow Q$
  (or relation $\delta \subseteq Q \times \Sigma \times Q$)

**Problem**: does not recognize $\bigcup_{a \in \Sigma} a(\Sigma \setminus a)^* \bigcup_{a \in \Sigma}$
What about infinite alphabets?

**Idea 11:** allow infinitely many states

- a set $Q$ of states
- an alphabet $\sum$ (infinite)
- initial state $q_0 \in Q$, accepting states $F \subseteq Q$
- transition function $\delta : Q \times \sum \rightarrow Q$
  (or relation $\delta \subseteq Q \times \sum \times Q$)

**Problem:** every language is recognized
A **register automaton** is:
- a set $Q$ of states
- a set $R$ of registers
- an alphabet $A$ (or $\Sigma \times A$)
- initial state $q_0 \in Q$, accepting states $F \subseteq Q$
- configurations: $\Gamma = Q \times (A \cup \{\perp\})^R$
- transition function $\delta : \Gamma \times A \rightarrow \Gamma$

(Or relation $\delta \subseteq \Gamma \times A \times \Gamma$)

that **only checks $A$ for equality**.
Every transition:

\[ q \xrightarrow{a} q' \]

is **guarded** by a Boolean combination of conditions:

\[ a = r_i \quad a = r'_j \quad r_i = r_j \quad r_i = r'_j \]

(so \(a\) is a “letter variable”, not an actual letter)

- \(r_i\) - old \(i\)-th register
- \(r'_i\) - new \(i\)-th register
This is a deterministic register automaton.
“Only checking for equality”, semantically

Every bijection $\pi : \hat{A} \rightarrow \hat{A}$ acts on configurations:

$$(q, a_1, \ldots, a_k) \cdot \pi = (q, \pi(a_1), \ldots, \pi(a_k))$$

This defines a **group action** of $\text{Aut}(\hat{A})$ on $\Gamma$.

A group action of $G$ on a set $X$:

$$\_ \cdot \_ : X \times G \rightarrow X$$

such that

$$x \cdot 1 = x$$

$$x \cdot (fg) = (x \cdot f) \cdot g$$
“Only checking for equality”, semantically

Every bijection $\pi : \mathbb{A} \rightarrow \mathbb{A}$ acts on configurations:

$$(q, a_1, \ldots, a_k) \cdot \pi = (q, \pi(a_1), \ldots, \pi(a_k))$$

This defines a group action of $\text{Aut}(\mathbb{A})$ on $\Gamma$.

We require $\delta$ to be equivariant:

if $(\gamma, a, \gamma') \in \delta$ then $(\gamma \cdot \pi, \pi(a), \gamma' \cdot \pi) \in \delta$

for all $\pi$.

Fact: The syntactic and the semantic conditions are equivalent.
It is tempting to write:

A **register automaton** is:
- a set $\Gamma$ of configurations
- a group action of $\text{Aut}(\mathcal{A})$ on $\Gamma$
- an alphabet $\mathcal{A}$ (or $\Sigma \times \mathcal{A}$)
- initial and accepting configurations
- transition function $\delta : \Gamma \times \mathcal{A} \to \Gamma$
  (or relation $\delta \subseteq \Gamma \times \mathcal{A} \times \Gamma$)

that is equivariant.

This is too powerful

(we’ll fix it later)
Questions

Q1: What about other computation models, logics, calculi etc?

Q2: What if we want to check for more than equality?
II
Sets with Atoms
\( X = \text{set, function, relation, automaton, Turing machine, grammar, graph, system of equations...} \)

**X with atoms**

Infinite but with lots of symmetries

Infinite but symbolically finitely presentable

We can compute on them
Von Neumann hierarchy

A hierarchy of universes:

\[ U_0 = \emptyset \]

\[ U_{\alpha + 1} = \mathcal{P}U_\alpha \]

\[ U_\beta = \bigcup_{\alpha < \beta} U_\alpha \]

defined for every ordinal number.

Elements of sets are other sets, in a well founded way

Every set sits somewhere in this hierarchy.
A - a countable set of atoms

A hierarchy of universes:

\[ \mathcal{U}_0 = \emptyset \]

\[ \mathcal{U}_{\alpha + 1} = \mathcal{P} \mathcal{U}_\alpha + A \]

\[ \mathcal{U}_\beta = \bigcup_{\alpha < \beta} \mathcal{U}_\alpha \]

Elements of sets with atoms are atoms or other sets with atoms, in a well founded way

A canonical group action:

\[ \_ \cdot \_ : \mathcal{U} \times \text{Aut}(A) \to \mathcal{U} \]
Finite support

\[ S \subseteq A \text{ supports } X \text{ if } \forall a \in S. \pi(a) = a \implies x \cdot \pi = x \]

A legal set with atoms:
- has a finite support,
- every element has a finite support,
- and so on.

A set is equivariant if it has empty support.
Examples

\[ a \in A \quad \text{is supported by} \quad \{a\} \]

\[ A \quad \text{is equivariant} \]

\[ S \subseteq A \quad \text{is supported by} \quad S \]

\[ A \setminus S \quad \text{is supported by} \quad S \]

**Fact:** \( S \subseteq A \) is fin. supp. iff it is finite or co-finite

\[ A^{(2)} = \{(d, e) \mid d, e \in A, d \neq e\} \quad \text{is equivariant} \]

\[ \begin{pmatrix} A \end{pmatrix}_2 = \\{\{d, e\} \mid d, e \in A, d \neq e\} \quad \text{is equivariant} \]
Closure properties

Legal sets with atoms are closed under:
- unions, intersections, set differences
- Cartesian products
- taking finitely supported subsets
- quotienting by finitely supported equivalence relations

**BUT** not under powersets!

\[ \mathcal{P}(\mathbb{A}) \] is equivariant but not legal.

They are closed under finite powersets \( \mathcal{P}_{\text{fin}}(\mathbb{A}) \) and finitely supported powersets \( \mathcal{P}_{\text{fs}}(\mathbb{A}) \)
Relations and functions

Relations and functions are sets too, so:

\[ R \subseteq X \times Y \] is equivariant iff
\[ xRy \quad \text{implies} \quad (x \cdot \pi)R(y \cdot \pi) \quad \text{for all } \pi \]

\[ f : X \rightarrow Y \] is equivariant iff
\[ f(x \cdot \pi) = f(x) \cdot \pi \quad \text{for all } \pi \]
Examples

For fixed $2, 5 \in A$:

$$R = \{(5, 2)\} \cup \{(2, d) \mid d \neq 5\} \cup \{(d, d)\}$$

$R$, $R^*$ are supported by $\{2, 5\}$
Equivariant binary relations on $\mathbb{A}$:

- empty
- total
- equality
- inequality

No equivariant function from $\mathbb{A}^2$ to $\mathbb{A}$, but

$$\{(\{a, b\}, a) \mid a, b \in \mathbb{A}\}$$

is an equivariant relation.

Only equiv. functions from $\mathbb{A}^2$ to $\mathbb{A}$ are projections.

Only equiv. function from $\mathbb{A}$ to $\mathbb{A}^2$ is the diagonal.
The **orbit** of $x$ is the set $\{x \cdot \pi \mid \pi \in \text{Aut}(A)\}$

Every equivariant set is a disjoint union of orbits.

**Orbit-finite set** if the union is finite.

More generally: the $S$-orbit of $x$ is

$\{x \cdot \pi \mid \pi \in \text{Aut}_S(A)\}$

**Fact:** An orbit-finite set is $S$-orbit-finite for every finite $S$. 


Examples

Orbit-finite sets:

\[ \mathbb{A} \quad \mathbb{A}^n \quad \binom{\mathbb{A}}{n} \]

\[ \mathbb{A}^\triangleleft = \{\{(a, b, c), (b, c, a), (c, a, b)\} \mid a, b, c \in \mathbb{A}\} \]

- closed under finite union, intersection difference, finite Cartesian product
- but not under (even finite) powerset!

Not orbit-finite:

\[ \mathbb{A}^* \quad \mathcal{P}_{\text{fin}}(\mathbb{A}) \]
Automata with atoms

A automaton with atoms is:

- a set $Q$ of states
- an alphabet $\sum$
- initial state $q_0 \in Q$, accepting states $F \subseteq Q$
- transition function $\delta : Q \times \sum \rightarrow Q$

(or relation $\delta \subseteq Q \times \sum \times Q$)

Fact: these are expressively equivalent to reg. aut.
Finite presentation

A set-builder expression:

\[ \{ e \mid a_1, \ldots, a_n \in A, \ \phi[a_1, \ldots, a_n, b_1, \ldots, b_m] \} \]

expression  bound variables  free variables  FO(=)-formula

Add also \( \emptyset \) and \( \cup \).

Fact: s.-b. e. + interpretation of free vars. as atoms

= a hereditarily orbit-finite set with atoms

Fact: Every h. o.-f. set is of this form.
Examples

The graph puzzle:

\[ G = (V, E) \]

\[ V = \{ (a, b) \mid a, b \in A, a \neq b \} \]

\[ E = \{ \{ (a, b), (b, c) \} \mid a, b, c \in A, a \neq b \neq c \} \]

(encode pairs with standard set-theoretic trickery)

Descriptions like this can be input to algorithms, for example:

Is 3-colorability of orbit-finite graphs decidable?
Set theory with atoms

Sets with atoms are a topos

A lot of mathematics can be done with atoms

- set $\rightarrow$ set with atoms
- finite $\rightarrow$ orbit-finite
- function $\rightarrow$ equivariant function

EXCEPT:
- axiom of choice fails, even orbit-finite choice
- powerset does not preserve orbit-finiteness
A recipe for adding atoms to everything:

1. Take your favourite definition.
2. Replace all sets (relations, functions etc.) with sets with atoms (equivariant if you wish).
3. Replace every “finite” with “orbit-finite”.
4. Check if your favourite theorems still hold.

(take with a pinch of salt)

Has been applied to: automata, grammars, Turing machines, while-programs, functional programs, CSPs, vector spaces, ...

Here: the $\mu$-calculus.
III

$\mu$-Calculus with Atoms
μ-calculus

Formula: \( \varphi \)

Model: \( \mathcal{K} \)

Semantics: \( p \) holds now
\( \mu \)-calculus

Formula: \( \varphi \)

Model: \( \mathcal{K} \)

Semantics: \( p \) holds in some successor

\[ p, q, r, \ldots \in P \]
$\mu$-calculus

Formula: $\varphi$

Model: $\kappa$

Semantics: $p$ holds now or in some successor

$p, q, r, \ldots \in \mathbb{P}$
μ-calculus

Formula: \( \varphi \)

Model: \( \mathcal{K} \)

Semantics: \( p \) holds in some future
μ-calculus

Formula: \( \nu X. (\neg p \land \Box X) \)

Model: \( \mathcal{K} \)

Semantics: \( p \) never holds in any future
Properties

Model checking:

Given \( k \in K \) and \( \varphi \), does \( K, k \models \varphi \)?

is decidable.

Satisfiability:

Given \( \varphi \), are there \( k \in K \) s.t. \( K, k \models \varphi \)?

is decidable.

Useful fragments, e.g. CTL*:

\[
\Phi ::= p \mid \Phi \lor \Phi \mid \neg \Phi \mid \exists \phi \\
\phi ::= \Phi \mid \phi \lor \phi \mid \neg \phi \mid \phi U \phi \mid X \phi
\]
Consider an infinite set of basic predicates:

\[ \mathbb{P} = \{p_0, p_1, p_2, \ldots \} \]

\( p_n \): the number \( n \) has been input

Now let's define the property:

The current input number is input again in some future

\[ \bigvee_{n \in \mathbb{N}} (p_n \land \lozenge \mu X. (p_n \lor \lozenge X)) \]

Problem: infinite disjunction

Practical motivation:

*The system never crashes*

*unless the password generator generates the same password twice...*
Fix an equivariant set $\mathcal{P}$ of basic predicates.  

A model (with atoms): $\mathcal{K} = (K, \rightarrow, \text{pred})$

- a set with atoms $K$,
- a finitely supported relation $\rightarrow \subseteq K \times K$,
- a finitely supported function $\text{pred} : K \rightarrow \mathcal{P}_{fs}\mathcal{P}$,

Example:

- $K = \mathbb{A}^*$,
- $w \rightarrow wa$ for $w \in \mathbb{A}^*$, $a \in \mathbb{A}$
- $\text{pred}(a_1 a_2 \cdots a_n) = \{a_n\}$
Syntax

Formulas of $\mathcal{L}^\mu_\Delta$:

$$\phi ::= p \mid X \mid \bigvee \Phi \mid \neg \phi \mid \diamond \phi \mid \mu X.\phi$$

Positive formula

Orbit-finite disjunction

We write e.g.

$$\forall a \in \Delta \phi_a \text{ for } \bigvee \{\phi_a \mid a \in \Delta\}$$

Standard abbreviations:

$$\top ::= p \lor \neg p$$
$$\bot ::= \neg \bigvee \{\neg \phi \mid \phi \in \Phi\}$$
$$\Box \phi ::= \neg \diamond \neg \phi$$
$$\nu X.\phi ::= \neg \mu X.\neg \phi[\neg^X/x]$$

Example:

$$\forall a \in \Delta (a \land \diamond \mu X.(a \lor \diamond X))$$
For a formula $\phi$ and a model $\mathcal{K}$

(and a valuation $\rho : Variables \rightarrow \mathcal{P}_{fsK}$)

define $\llbracket \phi \rrbracket_\rho \subseteq K$ by induction:

- $\llbracket p \rrbracket_\rho = \{ x \in K \mid p \in \text{pred}(x) \}$,
- $\llbracket X \rrbracket_\rho = \rho(X)$,
- $\llbracket \neg \phi \rrbracket_\rho = K \setminus \llbracket \phi \rrbracket_\rho$,
- $\llbracket \bigvee \Phi \rrbracket_\rho = \bigcup \{ \llbracket \phi \rrbracket_\rho \mid \phi \in \Phi \}$,
- $\llbracket \Diamond \phi \rrbracket_\rho = \{ k \in K \mid \exists s \in \llbracket \phi \rrbracket_\rho. \ k \rightarrow s \}$,
- $\llbracket \mu X. \phi \rrbracket_\rho = \text{lfp}(F')$, where $F(A) = \llbracket \phi \rrbracket_\rho[X \mapsto A]$. 
Examples

- $\bigvee_{a \in A}(a \land \Diamond \mu X.(a \lor \Diamond X))$
  
  some predicate that holds now, holds again in some future

- $\nu X.((\Diamond \bigvee_{a \in A} a) \land \Box X)$
  
  every reachable state has some successor for which some basic predicate holds

- $\neg(\mu X.(\psi \lor \Diamond X))$
  
  on every path, no basic predicate holds more than once

\[ \psi = \bigvee_{a \in A}(a \land \Diamond \mu Y.(a \lor \Diamond Y)) \]
Properties

Fact: Model checking on orbit-finite models is decidable.
   (proof: direct computation of semantics, including fixpoints)

Fact: Satisfiability is undecidable.
   (proof: direct encoding of Turing machine computations)

CTL* with atoms:

\[
\Phi ::= p \mid \bigvee a \Phi_a \mid \neg \Phi \mid \exists \phi \\
\phi ::= \Phi \mid \bigvee a \phi_a \mid \neg \phi \mid \phi U \phi \mid X \phi
\]

Fact: This is not a fragment of $L^A_\mu$.
   (and it has undecidable model checking)
The fresh path property

The property:

on some path
no basic predicate holds more
than once

is not expressible.

Note:

on every path
no basic predicate holds more
than once

is expressible:

$$\neg (\mu X. (\psi \lor \diamond X))$$
$$\psi = \bigvee_{a \in A} (a \land \diamond \mu Y. (a \lor \diamond Y))$$
The history-dependent $\mu$-calculus

Extend the syntax:

$$\phi ::= p \mid X \mid \#a \mid \bigvee \Phi \mid \neg \phi \mid \Diamond \phi \mid \mu X. \phi$$

Idea: $\#a$ says “$a$ has never appeared in any predicate so far”

Semantics evaluated in the context of a history $H \subseteq_{\text{fin}} A$:

$$x \in \llbracket \#a \rrbracket^H_{\rho} \iff a \notin H$$

$$x \in \llbracket \Diamond \phi \rrbracket^H_{\rho} \iff \exists y \in \llbracket \phi \rrbracket^{H \cup \text{pred}(x)}_{\rho} \text{ s.t. } x \rightarrow y$$

This expresses the fresh path property:

$$\nu X. (\bigwedge_{a \in A} (a \rightarrow \#a) \land \Diamond X)$$

Fact: Model checking on orbit-finite models still decidable.
The Needham-Schroeder public-key protocol:

The system (consisting of Alice, Bob and Eve) is represented as an orbit-finite model, and its security is expressed as a formula. (which fails)
A recipe for adding atoms to everything:

1. Take your favourite definition.
2. Replace all sets (relations, functions etc.) with sets with atoms (equivariant if you wish).
3. Replace every “finite” with “orbit-finite”.
4. Check if your favourite theorems still hold.
   (take with a pinch of salt)
Further reading

Books:

- A. Pitts: *Nominal sets. Names and symmetry in Computer Science*
  Cambridge Univ. Press, 2013

- M. Bojańczyk: *Slightly infinite sets*
  to appear, available online: