The Logical Approach to Automatic Sequences

Part 2: Logic and Automata

Jeffrey Shallit
School of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1
Canada
shallit@cs.uwaterloo.ca
https://cs.uwaterloo.ca/~shallit

Outline of today's talk

In this talk we will explore the work of Presburger, Büchi, Bruyère, Point, Villemaire, and Hodgson relating automata to logic.

From Monday: Formal definition of our automaton model

Formally, a deterministic finite automaton with output (DFAO) is a 6-tuple $M = (Q, \Sigma, \delta, q_0, \Delta, \tau)$ as follows:.

- Q is a finite nonempty set of states;.
- Σ is the input alphabet (usually Σ_k);.
- $\delta : Q \times \Sigma \to Q$ is the transition function, which is extended to $Q \times \Sigma^*$;.
- q_0 is the initial (or "start") state;.
- Δ is the output alphabet;.
- $\tau : Q \to \Delta$ is the output map.

From Monday: Deterministic finite automata (DFA's)

A different, slightly simpler model:.

- No output function, just a distinguished set of states F called the *final* or *accepting* states.
- An automaton *accepts* a word w if processing w takes M from q_0 to a state of F.
- The *language* accepted by automaton M, written $L(M)$, is the set of all words accepted.
- A language is *regular* if it is accepted by some DFA
- There is an efficient algorithm to find the (unique) minimal DFA equivalent to a given DFA (e.g., Valmari).

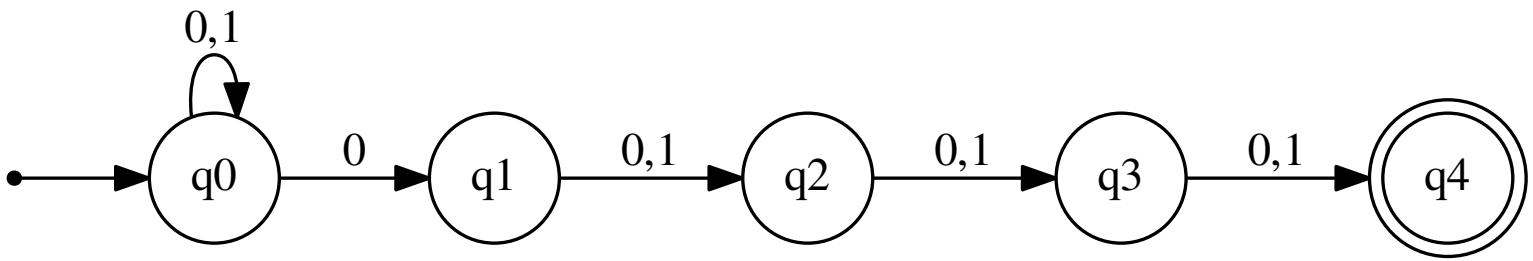
Nondeterministic automata

We will need another model of automata: nondeterministic automata (NFA).

- like deterministic automata, except that from each state there can be 0, 1, or more transitions on a single symbol
- Acceptance is defined by the *existence* of *some* path from the initial state to the final state, labeled by the input
- The transition function δ now has domain $Q \times \Sigma$ and range 2^Q.
- We can convert an NFA to an equivalent DFA with a construction called the "subset construction"; its states are subsets of states of the original automaton.
- In the worst case, an NFA with n states can require as many as 2^n states in an equivalent DFA.

Example of a nondeterministic automaton

A classic example: a nondeterministic machine accepting the set of all binary strings having a 0 symbol in the fourth position from the end:



Small problem: it is not easy to take the complement of a nondeterministic automaton.

One must first convert the NFA to a DFA.

Hilbert's dreams

- To show that every true statement is provable (killed by Gödel)
- To provide an algorithm to decide if an input statement is provable (killed by Turing)
- Nevertheless, some subclasses of problems are decidable — i.e., an algorithm exists guaranteed to prove or disprove any statement

First-order logic

By *first-order logic*, we mean the set of all formulas formed from

- any finite number of variables that can take values in some domain;
- equality defined on variables;
- possibly other comparison operators that can be applied to variables, such as less than, greater than, etc., depending on domain;
- possibly other functions applied to the variables, such as addition or multiplication;
- logical operations such as `and` ($\wedge$), `or` ($\vee$), `logical implication` ($\implies$), `iff` ($\iff$), and `not` ($\neg$);
- quantifiers, such as `for all` ($\forall$) and `there exists` ($\exists$).

First-order logic

- Variables can be either bound (by a quantifier) or unbound.
- If all variables are bound, then we can assign a truth value to the formula.
- If some variables are unbound, then we can consider the set of all values of the variables for which the formula is true.

A first-order logical theory is *decidable* if there is an algorithm that, given a well-formed formula with all variables bound, will decide its truth.

In the case of unbound variables, we'd like to algorithmically construct the representations of all integers for which the formula is true.

Presburger arithmetic

Presburger arithmetic is $\mathrm{Th}(\mathbb{N}, +)$, the first-order theory of the natural numbers $\mathbb{N} = \{0, 1, 2, \ldots\}$ with addition.

Mojżesz Presburger (1904–1943)
(died in the Holocaust)

Presburger arithmetic

- Sometimes Presburger arithmetic is written to include $<$, the "less-than" operator
- But it is not really needed, since the assertion $x < y$ is equivalent to $\exists z \ (z \neq 0) \ \wedge \ y = x + z$.

Example: The Chicken McNuggets Problem

A famous problem in elementary arithmetic books in the US:

At McDonald's, Chicken McNuggets are available in packs of either 6, 9, or 20 nuggets. What is the largest number of McNuggets that one cannot purchase?

Presburger arithmetic

In Presburger arithmetic we can express the "Chicken McNuggets theorem" that 43 is the largest integer that cannot be represented as a non-negative integer linear combination of $6, 9$, and 20, as follows:

$$(\forall n > 43\ \exists x, y, z \geq 0 \text{ such that } n = 6x + 9y + 20z) \wedge \neg(\exists x, y, z \geq 0 \text{ such that } 43 = 6x + 9y + 20z). \quad (1)$$

Here, of course, "$6x$" is shorthand for the expression "$x + x + x + x + x + x$", and similarly for $9y$ and $20z$.

More examples

x is even:

$$\exists y \; x = y + y$$

definition of the number $x = 1$:

$$(x \neq 0) \; \wedge (\forall y \; (y \neq 0) \implies y \geq x)$$

commutativity of addition:

$$\forall x \; \forall y \; (x + y = y + x)$$

Presburger's theorem

Presburger proved that $\mathrm{Th}(\mathbb{N}, +, 0, 1)$ is *decidable*: that is, there exists an algorithm that, given a well-formed formula in the theory, will decide its truth.

He used quantifier elimination.

Decidability of Presburger arithmetic: Büchi's proof

J. Richard Büchi found a much simpler proof of Presburger's result, based on automata. It gives us automata for the unbound variable case, too!

Ideas:

- represent integers in an integer base $k \geq 2$ using the alphabet $\Sigma_k = \{0, 1, \ldots, k-1\}$.
- represent n-tuples of integers as words over the alphabet Σ_k^n, padding with leading zeroes, if necessary. This corresponds to reading the base-k representations of the n-tuples *in parallel*.
- For example, the pair $(21, 7)$ can be represented in base 2 by the word

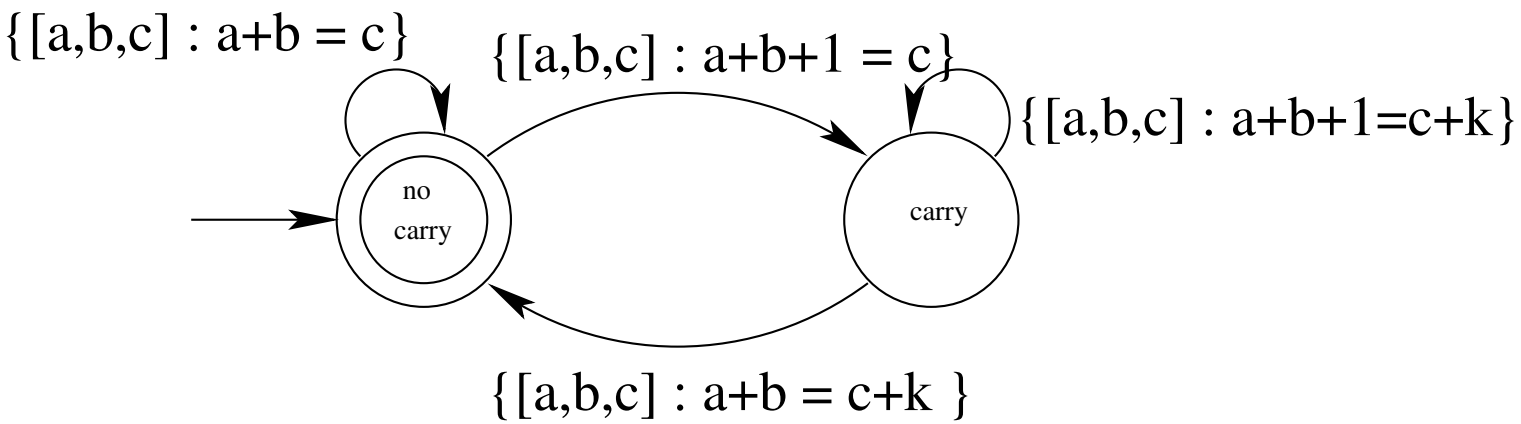
$$[1, 0][0, 0][1, 1][0, 1][1, 1].$$

Büchi's proof

- Automata will accept words over the alphabet Σ_k^n representing n-tuples of integers
- The language accepted is the set of all n-tuples of integers for which the formula (or subformula) is true
- Parsing the formula corresponds to performing operations on automata
- For example, if automaton M corresponds to some formula φ, then $\neg\varphi$ can be obtained by changing the "finality" of M's states: a final state becomes non-final and vice-versa
- Care is needed to handle the "leading zeroes" problem

Decidability of Presburger arithmetic

- The relation $x + y = z$ can be checked by a simple 2-state automaton depicted below, where transitions not depicted lead to a nonaccepting "dead state".



Decidability of Presburger arithmetic: proof sketch

- Relations like $x = y$ and $x < y$ can be checked similarly. (exercise)
- Given a formula with free variables $x_1, x_2, \ldots, x_n$, we construct an automaton accepting the base-k expansion of those n-tuples $(x_1, \ldots, x_n)$ for which the proposition holds.
- If a formula is of the form $\exists x_1, x_2, \ldots x_n \; p(x_1, \ldots, x_n)$, then we use nondeterminism to "guess" the x_i and check them.
- If the formula is of the form $\forall p$, we use the equivalence $\forall p \equiv \neg\exists\neg p$; this may require using the subset construction to convert an NFA to a DFA and then flipping the "finality" of states.
- Ultimately, if all variables are bound, we are left with a single state machine that either accepts (formula is true) or rejects (formula is false)

The bad news

- The worst-case running time of the algorithm above is bounded above by

$$2^{2^{\cdot^{\cdot^{\cdot^{2^{p(N)}}}}}},$$

  where the number of 2's in the exponent is equal to the number of quantifier alternations, p is a polynomial, and N is the number of states needed to describe the underlying automatic sequence.
- The bound for Presburger arithmetic can be improved to double-exponential.

The proof using automata

A couple of additional tricks: if the last quantifiers are $\exists$, all we need to do is check to see if the resulting automaton accepts some word.

In this case, we do not need to convert an NFA to a DFA.

We can check acceptance with depth-first search, by seeing if there is a path in the automaton from the initial state q_0 to a state of F. This can be done in time linear in the size of the automaton.

Similarly, if we want to know if there are infinitely many integers for which some formula holds (which is sometimes written $\exists_\infty$) we just need to check for which states q there is a nonempty cycle beginning and ending at q (which can be done using depth-first search), and then check to see if there is a path from q_0 to q and q to a final state. Again, linear time.

Some subtleties

Every integer has infinitely many representations!

For example, 5 in base 2 can be written as 101, 0101, 00101, and so forth.

It is best to allow all possible representations in our automata.

(If we do not, then we can run into problems working with k-tuples of integers where one integer has a larger representation than other.)

Augmenting Presburger arithmetic

As described, Presburger arithmetic isn't so interesting (although used, e.g., in system verification).

But if we add DFAO's to the mix, using the same decision procedure, we suddenly can prove theorems people actually want to prove.

For example, we can start with a 2-DFAO M for the Thue-Morse sequence $\mathbf{t}$, write a formula for $\mathbf{t}$ having an overlap, and use the decision procedure to decide it — thus reproving Thue's 1912 result by machine.

But what is the logical theory corresponding to starting with a DFAO?

Büchi's mistake

Julius Richard Büchi (1924–1984) was apparently the first to consider this question.

He thought one should add, to Presburger arithmetic, the function $\nu_k(n)$, which is the function computing the *exponent* of the highest power of k dividing n. For example, $\nu_2(24) = 3$.

This was a mistake.

The correct function to add is $V_k(n)$, the function computing the highest power of k, say k^e, dividing n. For example, $V_2(24) = 8$.

Exercise: show that for $k \geq 2$ the theory $\mathrm{Th}(\mathbb{N}, +, V_k)$ coincides with $\mathrm{Th}(\mathbb{N}, +, V_{k^2})$.

Presburger arithmetic augmented

Theorem. A set of integers is definable in $\mathrm{Th}(\mathbb{N}, +, V_k)$ if and only if its characteristic sequence is k-automatic.

Proof. First we show how to construct a finite automaton M_φ corresponding to any formula φ of $\mathrm{Th}(\mathbb{N}, +, V_k)$.

The idea again is that M_φ will accept the base-k representations of all n-tuples $(x_1, x_2, \ldots, x_n)$ of natural numbers making $\varphi(x_1, x_2, \ldots, x_n)$ true.

We use the *least-significant-digit first* representation for numbers.

We observe that $\mathrm{Th}(\mathbb{N}, R_+, R_{V_k})$ is equivalent to $\mathrm{Th}(\mathbb{N}, +, V_k)$, where $R_+(x, y, z)$ is the relation $x + y = z$ and $R_{V_k}(x, y)$ is the relation $V_k(x) = y$.

We already saw automata for addition, so it suffices to give an automaton for $V_k(x) = y$. (Exercise)

Presburger arithmetic with V_k is decidable

Corollary. The theory $\mathrm{Th}(\mathbb{N}, +, V_k)$ is decidable.

Proof. We can decide if a formula in $\mathrm{Th}(\mathbb{N}, +, V_k)$ is true, just as with Presburger arithmetic, by creating the automaton associated with the formula and checking if it accepts.

Presburger arithmetic augmented

Next we show how to encode a binary automatic sequence $(s(n))_{n\geq 0}$ in $\mathrm{Th}(\mathbb{N}, +, V_k)$. Actually we encode $\{n \ : s(n) = 1\}$ and we use the equivalent theory $\mathrm{Th}(\mathbb{N}, R_+, R_{V_k})$.

The basic idea, given an integer x for which $s(x) = 1$, is to encode another integer y that gives the sequence of states x encounters as it is processed by the automaton.

To do so we need new relations

$$e_{j,k}(x, y)$$

for $0 \leq j < k$. The meaning of this relation is that y is some power of k, say $y = k^e$, and the coefficient of k^e in the base-k representation of x is equal to j.

We also need $\lambda_k(x)$, which is the greatest power of k occurring with a nonzero coefficient in the base-k representation of x. By definition we set $\lambda_k(0) = 1$.

We also need $P_k(x)$, which is true if x is a power of k and false otherwise.

Presburger arithmetic augmented

Now we show how to express $e_{j,k}(x, y)$ and $\lambda_k(x)$ and $P_k(x)$ in $\mathrm{Th}(\mathbb{N}, +, V_k)$.

$P_k(x)$ is the easiest. We have $P_k(x)$ is the same as $V_k(x) = x$.

$\lambda_k(x) = y$ is the next easiest. The basic idea is to observe that if we trap x between two powers of k, say $k^e \leq x < k^{e+1}$, then $\lambda_k(x) = k^e$.

So $\lambda_k(x) = y$ is the same as

$$(P_k(y) \ \wedge \ (y \leq x) \ \wedge \ x < ky) \ \vee \ ((x = 0) \ \wedge \ (y = 1)).$$

Finally, we can express $e_{j,k}(x, y)$ as follows: we group the powers of k appearing in x as follows: those appearing in y, those of exponent less than the one occurring in y, and those of exponent greater.

Presburger arithmetic augmented

So $e_{j,k}(x, y)$ is equivalent to

$$P_k(y) \wedge (\exists \ell \, \exists g \, (x = \ell + jy + g) \wedge (\ell < y) \wedge ((y < V_k(g)) \vee (g = 0))).$$

Now that we have these relations, we can encode the computation of a DFAO with a large formula (similar to the way we encode a Turing machine with a SAT formula):

To simplify things, we assume the DFAO has at most k states. If it has more, another trick is needed.

The idea is to create a base-k integer y that encodes the series of states encountered as we process the base-k digits of the input integer x.

Presburger arithmetic augmented

If $x = \sum_{0 \leq i \leq l} a_i k^i$, the input is $a_0 a_1 \cdots a_l$ and the series of states encountered is $p_0, p_1, \ldots, p_{l+1}$. Our formula should say that
(i) $p_0 = 0$
(ii) $\delta(p_i, a_i) = p_{i+1}$ for $0 \leq i \leq l$
(iii) $p_{l+1} \in F$.

This is

(i) $e_{1,k}(y, 1)$

(ii) $\forall t \ P_k(t) \ \wedge \ (t < z) \ \wedge$
$\bigwedge_{\delta(q,b)=q'} \left(e_{q,k}(y, t) \ \wedge \ e_{b,k}(x, t) \implies e_{q',k}(y, kt) \right)$

(iii) $\bigvee_{q \in F} e_{q,k}(y, z)$

Finally, the formula is

$$\exists y \ \exists z \ P_k(z) \ \wedge \ (z > y) \ (z > x) \ \wedge \ (i) \ \wedge \ (ii) \ \wedge \ (iii).$$

Not all morphic sequences have decidable theories

- Consider the morphism $a \to abcc$, $b \to bcc$, $c \to c$.
- The fixed point of this morphism is

$$\mathbf{s} = abccbccccbcccccccbccccccccb\cdots$$

- It encodes, in the positions of the b's, the characteristic sequence of the squares.
- So the first-order theory $\mathrm{Th}(\mathbb{N}, +, 0, 1, n \to \mathbf{s}[n])$ is powerful enough to express the assertion that "n is a square"
- With that, one can express multiplication, and so it is undecidable (Church, 1936).

Open Problems

Is the logical theory $(\mathbb{N}, +, P_2, P_3)$ decidable? Here P_k is the predicate "is a power of k".

We know the theory $(\mathbb{N}, +, \cdot)$ is undecidable (Church, 1936).
Is the logical theory $(\mathbb{N}, +, n \to p(n))$ decidable? Here $p(n)$ is the primality predicate, which is true if n is prime and false otherwise.

Is the logical theory $(\mathbb{N}, +, n \to \varphi(n))$ decidable? Here $\varphi(n)$ is Euler's phi function, counting the number of integers $\leq n$ and relatively prime to it.

Is the following problem decidable? Given two k-automatic sequences $(a(n))_{n \geq 0}$ and $(b(n))_{n \geq 0}$, are there integers $c \geq 1$ and $d \geq 0$ such that $a(n) = b(cn + d)$ for all n?