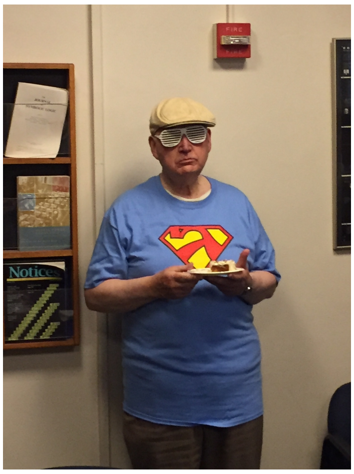
Transducers and Rational Relations

A. Anil & K. Sutner
Carnegie Mellon University

Spring 2018

Wisdom



> The author (along with many other people) has come recently to the conclusion that the functions computed by the various machines are more important–or at least more basic–than the sets accepted by these devices.

D. Scott, "Some Definitional Suggestions for Automata Theory," 1967

Quoi?

Acceptor A machine that checks membership in some language $L \subseteq \Sigma^\star$ (a machine that solves a decision problem).

Transducer A machine that computes a function

$$f : \Sigma^\star \to \Sigma^\star$$

or a relation

$$R \subseteq \Sigma^\star \times \Sigma^\star$$

We will focus on relations (functions are just a special kind).

Example: Lexicographic Order

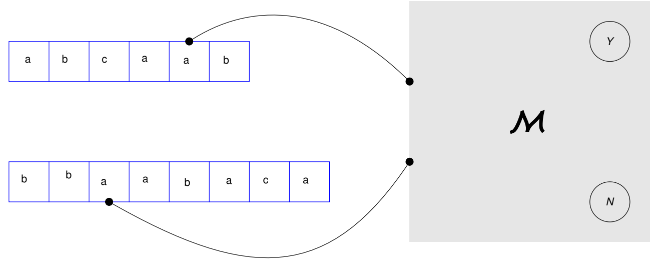
How do we check whether a word u is lexicographically less than v?

- Find the longest common prefix x (so $u = xu'$ and $v = xv'$).
- Handle the case where $u' = \varepsilon$ or $v' = \varepsilon$.
- Otherwise, compare u'_1 and v'_1.

Insight: We scan the input once, left-to-right; our decisions require no extra memory.

This looks just like a DFA, except there are two inputs.

Two-Tape DFA



Details?

We will not give a careful definition of how a k-tape DFA works and appeal to your intuition instead. The main idea is to use transitions of the form

$$p \xrightarrow{x/y} q$$

where the labels are in the following form:

$$a/b \qquad \text{and} \qquad a/\varepsilon \qquad \text{and} \qquad \varepsilon/b.$$

You can think of this as the transducer checking for an a on the first tape, and a b on the second tape.

Or, if you prefer, the machine reads an a and writes a b.

Truth in Advertising

For transducers, nondeterminism is really critical: there are no constraints on these transitions:

$$p \xrightarrow{a/b} q \qquad \text{and} \qquad p \xrightarrow{a/b} q'$$

is perfectly fine for $q \neq q'$.

Acceptance means: there is at least one good path leading from the initial state to a final state that has the right labels.

This is a bit like $\mathbb{NP}$: one good path/witness is enough, all other attempts may well fail.

Rational Relations

So we are trying to come up with a clean definition of the class of rational relations without using explaining all the gory details of a machine model.

As it turns out, it is surprisingly easy to exploit the algebraic characterization of regular languages: the closure of $\emptyset$ and singletons $\{a\}$ under concatenation, union and Kleene star.

Why rational relations rather than regular relations? Historical accident and a bit of tension between the US and Europe.

Kleene's Theorem

Theorem (Kleene 1956)

Every regular language over Σ can be constructed from $\emptyset$ and singletons $\{a\}$, $a \in \Sigma$, using only the operations union, concatenation and Kleene star.

It follows that there is a convenient notation system (regular expressions) for regular languages that is radically different from finite state machines: we can use an algebra (albeit a slightly weird one) to concoct regular languages. Read the `grep` manual to appreciate the importance of this.

One direction is easy, given the inductive structure of a regular expression and the closure properties of regular languages we already have: every regular expression denotes a regular language.

Other Direction

The big problem with the other direction is: a finite state machine has no inductive structure, it's just a bunch of labeled edges.

Solution: We hit it hard until it has an inductive structure.

More precisely, suppose we have a DFA that accepts some regular language L. Assume $Q = [n]$. For p, q in Q define

$$L_{p,q} = \mathcal{L}(\langle Q, \Sigma, \delta; p, \{q\} \rangle)$$

Then $L = \bigcup_{q \in F} L_{q_0,q}$ and it suffices to construct regular expressions for the $L_{p,q}$.

The Trick

In order to enable an inductive argument, define a computation from state p to state q to be k-bounded if all intermediate states are no greater than k. Note that we only constrain the intermediate states, p and q themselves are not required to be bounded by k.

In other words: we have erased all states $> k$.

Now consider the approximation languages:

$$L^k_{p,q} = \{\, x \in \Sigma^\star \mid \text{there is a } k\text{-bounded run } p \xrightarrow{x} q \,\}.$$

Note that $L^n_{p,q} = L_{p,q}$.

Proof Sketch, contd.

One can build expressions for $L^k_{p,q}$ by induction on k.

For $k = 0$ the expressions are easy:

$$L^0_{p,q} = \begin{cases} \sum_{\delta(p,a)=q} a & \text{if } q \neq p, \\ \sum_{\delta(p,a)=p} a + \varepsilon & \text{if } p = q. \end{cases}$$

So suppose $k > 0$. The key idea is to use the equality

$$L^k_{p,q} = L^{k-1}_{p,q} + L^{k-1}_{p,k} \cdot (L^{k-1}_{k,k})^* \cdot L^{k-1}_{k,q}$$

Done by induction hypothesis. $\square$

Algebra to the Rescue

The last argument is a perfect example of dynamic programming. Unfortunately, the regular expressions involved grow exponentially, so the algorithm is not practical.

Still, one very nice feature of Kleene's characterization is that a good definition often generalizes. In this case, the monoid $\Sigma^\star$ is perhaps the most natural setting, but there are other plausible choices.

In particular we could use the product monoid $\Sigma^\star \times \Sigma^\star$ instead: since we are dealing with sets of pairs of strings we naturally obtain binary relations this way.

The relevant algebraic structures are called Kleene algebras. We will not study them in any detail and just pull out the pieces that we need for our project.

More Precisely ...

Suppose $\langle M, \cdot, 1 \rangle$ is a monoid. Here is a general way to construct a Kleene algebra on top of M. The carrier set is $\mathfrak{P}(M)$ and the operations are

- set theoretic union,
- pointwise multiplication, and
- Kleene star.

More precisely, define

$$K \cdot L = \{ x \cdot y \mid x \in K, y \in L \}$$

$$K^0 = \{1\} \qquad K^{n+1} = K \cdot K^n$$

$$K^\star = \bigcup_{n \geq 0} K^n$$

Definition

A k-ary rational relation is a relation $R \subseteq M$ where

$$M = \Sigma_1^\star \times \Sigma_2^\star \times \ldots \times \Sigma_k^\star$$

and R is generated in the Kleene algebra over M from elements

$$(\varepsilon, \ldots, \varepsilon, a, \varepsilon, \ldots, \varepsilon)$$

Strictly speaking, this should be a singleton set, but in this context it is best not to distinguish between z and $\{z\}$. Trust me and types be damned.

Note that in the special case $k = 1$ we get back ordinary regular languages.

It is easy to see that we could also allow generators of the form

$$(x_1, x_2, \ldots, x_{k-1}, x_k)$$

where $x_i \in \Sigma_i^\star$.

Using these and customary operation symbols $+$, $\cdot$ (often implicit) and * we obtain rational expressions that provide a notation system for rational relations.

We will mostly deal with the case $k = 2$ and consider the monoid

$$M = \Sigma^\star \times \Gamma^\star \qquad \text{and even} \qquad M = \Sigma^\star \times \Sigma^\star$$

We often write x/y for an element of M, so $x \in \Sigma^\star$, $y \in \Gamma^\star$. This is just fancy notation for a pair of words.

Writing rational expression can be a bit confusing, so on occasion we use vector notation, in particular in the case $k = 2$, and write $\binom{x}{y}$. Think of this a two tracks with one word in each track.

Note that multiplication here is componentwise:

$$\binom{x}{y} \cdot \binom{u}{v} = \binom{xu}{yv}$$

Examples

Let $\Sigma = \{a, b\}$.

The universal relation on $\Sigma^\star$ is given by

$$\left(\binom{\varepsilon}{a}+\binom{\varepsilon}{b}+\binom{a}{\varepsilon}+\binom{b}{\varepsilon}\right)^* = \{\, \binom{x}{y} \mid x, y \in \Sigma^\star \,\}$$

The identity relation on $\Sigma^\star$ is given by

$$\left(\binom{a}{a}+\binom{b}{b}\right)^* = \{\, \binom{x}{x} \mid x \in \Sigma^\star \,\}$$

Exercise

Show that the un-equal relation is rational.

Maps versus Relations

By definition, a rational relation has the form

$$\rho \subseteq \Sigma^\star \times \Gamma^\star$$

But we may also think of such a relation as a so-called transduction, a map

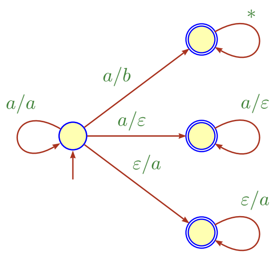
$$\rho : \Sigma^\star \longrightarrow \mathfrak{P}(\Gamma^\star)$$

This is the reason for the x/y notation: think of x as being mapped to y.

Of course, there is no fundamental difference between the two interpretations.
We will switch back and forth as convenient.

Machine Example

We already know that identity $x = y$ is a rational relation. Here is a transducer whose behavior is the relation $x \neq y$.



In the diagram, a and b are supposed to range over Σ, and $a \neq b$.
$*$ means: gobble up the rest of the input, in a state of eternal bliss.

Kleene's Theorem on Steroids

With the proper definitions we can show an analogue to Kleene's theorem:

Theorem

A relation is rational (in the sense of the Kleene algebra approach) if, and only if, it is computed by a (finite) transducer.

The proof is an exact re-run of the argument for regular languages, once all the definitions are just right.

Exercise

Write out a detailed proof of the theorem.

More Wisdom

> Good mathematicians see analogies between theorems or theories; the very best ones see analogies between analogies.
>
> S. Banach

Wurtzelbrunft's Conjecture

Wurtzelbrunft remembers the Banach quote about analogies and immediately concludes:

> Every result about regular languages carries over, *mutatis mutandis*, to rational relations.

After all, it's just about the same Kleene algebra we are working in, so what could possibly change? For example, we should be able to come up with a nice machine model, figure out how to determinize and minimize these devices, and so on.

Fortunately, life is so much more interesting than that.

Some results do indeed carry over, almost verbatim. But others a plain false and one has to be very careful.

Problem 1: Intersection

Consider the binary rational relations

$$A = \binom{a}{c}^\star \binom{b}{\varepsilon}^\star \qquad B = \binom{a}{\varepsilon}^\star \binom{b}{c}^\star$$

Then

$$A \cap B = \{ \binom{a^i b^i}{c^i} \mid i \geq 0 \}$$

It is easy to see that the intersection cannot be recognized by a finite state transducer, essentially for the same reasons that $\{ a^i b^i \mid i \geq 0 \}$ fails to be regular.

Exercise

Prove that $A \cap B$ really fails to be rational.

Disaster Strikes

Rational relations are closed under union by definition.

So the last result shows that we fail to have closure under intersection and complement.

This makes it really difficult to work with rational relations in general. In fact, next class we will see how to adjust our definitions to make this problem go away.

But for the time being, let's stick with rational relations.

Examples

Example

If $K \subseteq \Sigma^\star$ and $L \subseteq \Gamma^\star$ are regular, then $K \times L$ is rational.

Example

If $\rho \subseteq \Sigma^\star \times \Gamma^\star$ is rational, then spt$(\rho) \subseteq \Sigma^\star$ and rng$(\rho) \subseteq \Gamma^\star$ are regular.

Here spt(ρ) is the support of ρ: $\{\, x \in \Sigma^\star \mid \exists\, y\, \rho(x, y)\,\}$.

Example

All the relations "x is a prefix of y", "x is a suffix of y", "x is a factor of y" and "x is a subword of y" are rational.

Shuffle

Example

Recall the definition of shuffle:

$$\varepsilon \parallel y = y \parallel \varepsilon \;=\; \{y\}$$

$$xa \parallel yb = (x \parallel yb)\, a \cup (xa \parallel y)\, b.$$

So $x \parallel y$ is the set of all possible interleavings of the letters of x and y (preserving relative order).

The map $(x, y) \mapsto x \parallel y$ is rational.

Note that we could also think of shuffle as a ternary relation sh(x, y, z) meaning $z \in x \parallel y$.

$$\left(\left(\begin{smallmatrix} a \\ \varepsilon \\ a \end{smallmatrix}\right) + \left(\begin{smallmatrix} b \\ \varepsilon \\ b \end{smallmatrix}\right) + \left(\begin{smallmatrix} \varepsilon \\ a \\ a \end{smallmatrix}\right) + \left(\begin{smallmatrix} \varepsilon \\ b \\ b \end{smallmatrix}\right)\right)^* = \left\{ \left(\begin{smallmatrix} x \\ y \\ z \end{smallmatrix}\right) \,\middle|\, z \in x \parallel y \right\}$$

Determinism

Disregarding state complexity, in the world of regular languages, there is no real need for nondeterminism: every regular language is already accepted by a deterministic FSM (a famous result by Rabin and Scott in 1959).

One might wonder if there is some notion of deterministic rational relation and a corresponding deterministic transducer.

The basic idea is simple: there should be at most one computation on all inputs.

Unfortunately, the technical details are a bit messy (use of endmarkers) and we'll skip this opportunity to inflict mental pain on the student body.

Problem 2: Determinism and Union

Consider the binary rational relations

$$A = \left(\begin{smallmatrix} aa \\ b \end{smallmatrix}\right)^\star \qquad B = \left(\begin{smallmatrix} a \\ bb \end{smallmatrix}\right)^\star$$

It is clear that both A and B are deterministic rational relations.

Now consider

$$A \cup B = \{\, \left(\begin{smallmatrix} a^i \\ b^j \end{smallmatrix}\right) \mid i = 2j \vee j = 2i \,\}$$

For the union, your intuition should tell you that nondeterminism is critical: initially, we don't know which type of test to apply. This indicates that determinization is not going to work in general for rational relations (which is to be expected since we already know that complementation fails in general).

Example: Word Orders

Consider the binary relation $<_{\text{len}}$ on $\Sigma^\star$ defined by

$$x <_{\text{len}} y \iff |x| < |y|.$$

We obtain a strict pre-order called length order; the corresponding classes of indistinguishable elements are words of the same length.

Given an ordered alphabet Σ consider the binary relation $<_{\text{s}}$ on $\Sigma^\star$ defined by

$$x <_{\text{s}} y \iff \exists\, a < b \in \Sigma, u, v, w \in \Sigma^\star\, (x = uav \wedge y = ubw)$$

This produces another strict pre-order, the so-called split order; this time indistinguishable words are prefixes of one another.

Lexicographic Order

Again assume an ordered alphabet Σ. The lexicographic order is a mix of prefix order and split order:

$$x <_\ell y \iff x \sqsubset y \vee x <_{\text{s}} y$$

Here $x \sqsubset y$ means that x is a proper prefix of y. Note that lexicographic order is a total order, there are no indistinguishable elements.

Proposition

Length order, split order and lexicographic order are all rational.

Exercise

Construct rational expressions that prove the proposition. Then construct the corresponding transducers.

Length-Lex Order

Another important way of ordering words is the product order of length order and lexicographic order, the so-called length-lex order.

$$x <_{\ell\ell} y \iff x <_{\text{len}} y \lor (|x| = |y| \land x <_{\ell} y)$$

Length-lex order is easily seen to be a well-order and there are many algorithms on strings that are naturally defined by induction on length-lex order.

Needless to say, length-lex order is also rational.

Concatenation is Rational

Usually one thinks of concatenation as a binary operation. But we can also model it as a ternary relation γ:

$$\gamma(x, y, z) \iff x \cdot y = z$$

Proposition

Concatenation is rational.

Proof. For simplicity assume $\Sigma = \{a, b\}$

$$\gamma = (a{:}\varepsilon{:}a + b{:}\varepsilon{:}b)^{\star} \cdot (\varepsilon{:}a{:}a + \varepsilon{:}b{:}b)^{\star}$$

□

Addition is Rational

Consider the ternary relation α on $\mathbf{2}$ defined by

$$\alpha(x, y, z) \iff \text{bin}(x) + \text{bin}(y) = \text{bin}(z)$$

where $\text{bin}(x)$ is the numerical value of x assuming the LSD is first (reverse binary).

Proposition

Binary addition in reverse binary is rational.

Proof. The kindergarten algorithm for addition shows that α is rational. □

Warning: there is no analogous result for multiplication (for reverse binary encoding; but beware of exotic encodings).

Relational Composition

Here is a central result: rational relations are closed under composition. Suppose we have two binary relations $\rho \subseteq \Sigma^\star \times \Gamma^\star$ and $\sigma \subseteq \Gamma^\star \times \Delta^\star$. Their composition $\tau = \rho \circ \sigma \subseteq \Sigma^\star \times \Delta^\star$ is defined to be the binary relation

$$x \,\tau\, y \iff \exists\, z\, (x \,\rho\, z \wedge z \,\sigma\, y)$$

Theorem (Elgot, Mezei 1965)

If both ρ and σ are rational, then so is their composition $\rho \circ \sigma$.

Proof

Assume we have transducers $\mathcal{A}$ and $\mathcal{B}$ for ρ and σ, respectively. We may safely assume that the labels in $\mathcal{A}$ have the form a/ε or ε/b where $a \in \Sigma$, $b \in \Gamma$; likewise for $\mathcal{B}$. Add self-loops labeled ε/ε everywhere.

We construct a product automaton $\mathcal{C}$ with transitions

$$(p, q) \xrightarrow{a/c} (p', q')$$

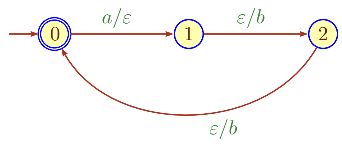
whenever there are transitions $p \xrightarrow{a/b} p'$ and $q \xrightarrow{b/c} q'$ in $\mathcal{A}$ and $\mathcal{B}$, respectively, for some $a \in \Sigma_\varepsilon$, $b \in \Gamma_\varepsilon$ and $c \in \Delta_\varepsilon$.

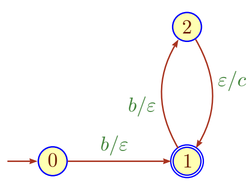
Initial and final states in $\mathcal{C}$ are $I_1 \times I_2$ and $F_1 \times F_2$. It is a labor of love to check that $\mathcal{C}$ accepts x/z if, and only if, $x \,\rho\, y$ and $y \,\sigma\, z$ for some $y \in \Gamma^\star$. $\square$

Example

Let $\rho = \left(\begin{smallmatrix} a \\ bb \end{smallmatrix}\right)^\star$ and $\sigma = \left(\begin{smallmatrix} b \\ \varepsilon \end{smallmatrix}\right)\left(\begin{smallmatrix} b \\ c \end{smallmatrix}\right)^\star$; thus $\rho \circ \sigma = \left(\begin{smallmatrix} a \\ c \end{smallmatrix}\right)\left(\begin{smallmatrix} a \\ cc \end{smallmatrix}\right)^\star$.

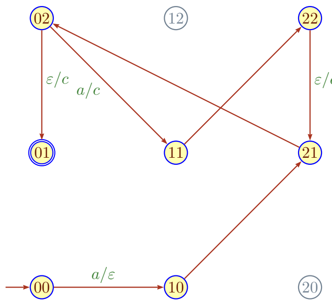
Here are the two machines, without the ε/ε self-loops.





Example

And here is the product; unlabeled transitions carry ε/ε.



Of course, there is a "better" transducer, but this is the one obtained by blind application of the algorithm.

Projections

Here is another important closure property. Suppose ρ is a k-ary relation on words. We define the projection of ρ to be

$$\rho'(x_2, \ldots, x_k) \iff \exists\, z\; \rho(z, x_2, \ldots, x_k)$$

Lemma

Whenever ρ is rational, so is its projection ρ'.

Proof.

Erase the first track in the k-track alphabet:

$$p \xrightarrow{a_1, a_2, \ldots, a_k} q \quad \rightsquigarrow \quad p \xrightarrow{a_2, \ldots, a_k} q$$

That's it! Of course, the new machine will be nondeterministic in general. $\square$

Transitive Closure

One might wonder what happens when we move to the transitive reflexive closure $\mathsf{tcl}(\rho)$. Recall that

$$\mathsf{tcl}(\rho) = \bigsqcup_k \rho^{\circ k}$$

where $\rho^{\circ k}$ indicates the standard iterate, the k-fold composition of ρ with itself.

Mental Health Warning: Unfortunately, the transitive closure is often written $\rho^\star$, in direct clash with the standard notation for the Kleene star of a relation.

Alas, the two are quite incompatible. For example, let ρ be lexicographic order. Clearly, $\mathsf{tcl}(\rho) = \rho$.

But $ab\ \rho^\star\ aabb$ since $a\ \rho\ aa$ and $b\ \rho\ bb$. So Kleene star clobbers the order completely.

Transitive Closure is Semidecidable

Theorem

The transitive closure $\mathsf{tcl}(\rho)$ *of a rational relation is semidecidable.*

Proof.

By definition $x \ \mathsf{tcl}(\rho) \ y$ iff $\exists\, k\, (x \ \rho^{\circ k} \ y)$.

Obviously, $\rho^{\circ k}$ is easily decidable, uniformly in k.

So we are conducting an unbounded search over a decidable relation; semidecidability follows.

□

Semidecidability

What would happen if we add tcl to the closure operations that produce the rational relations?

Theorem

Adding tcl *to the closure operations produces precisely all semidecidable relations.*

Proof.

Clearly every relation obtained this way is semidecidable.

For the opposite direction, note that the one-step relation of a Turing machine is rational.

Then transitive closure is all that is needed to produce any semidecidable relation.

□

Quoi?

We use the old trick of coding configurations of a Turing machine as words of the form $\Gamma^\star Q \Gamma^\star$:

$$x_m x_{m-1} \ldots x_1 \, p \, a \, y_2 \ldots y_n$$

Then the next configuration could look like

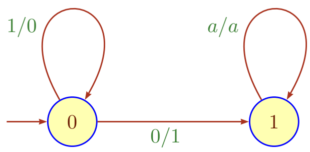
$$x_m x_{m-1} \ldots x_1 \, b \, q \, y_2 \ldots y_n$$

For the most part, we just copy the tape symbols, but there is a little bit of hanky panky right next to the state symbol.

A transducer can easily handle this type of update operation.

By taking the transitive closure we get arbitrary computations, and thus all semidecidable relations.

How About Functions?

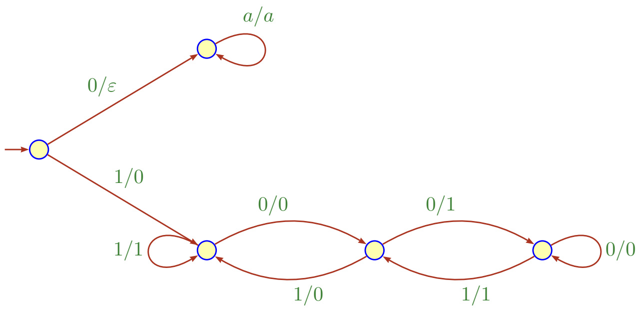


Theorem (Schützenberger 1975)

It is decidable whether a transducer defines a function.

The argument is tricky, and it took 25 years to find a polynomial algorithm for functionality.

A Holy Terror



Collatz

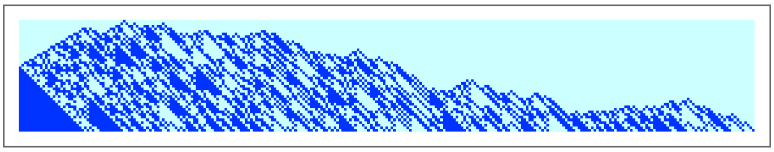
Recall the infamous Collatz problem: Does the following program halt for all $x \geq 1$?

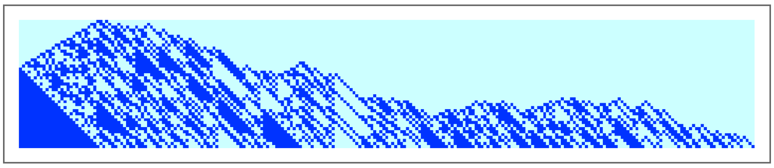
```
while( x > 1 )              // x positive integer
  if( x even )
      x = x/2;
  else
      x = 3 * x + 1;
```

If we write x in reverse binary, and right-pad with 00, the transducer on the last slide computes one execution of the loop body.

So iterating the composition of the trivial $x \mapsto x00$ and the transducer leads to an open problem in number theory.

Collatz Mountains





Off to ∞

If you think the padding operation $x \mapsto x00$ is an ugly kludge, you can easily get rid of it: just write your numbers as infinite strings in $\mathbf{2}^\omega$.

Stop when the string 10^ω is reached.

This may sound weird, but automata on infinite strings are actually hugely important and help to build decision algorithms that are used in program verification.

Unfortunately, they don't seem to be of any use when it comes to settling the Collatz conjecture.

Length-Preserving Iteration

Life should be much easier with length-preserving relations:

$$x \; \tau \; y \Rightarrow |x| = |y|$$

In fact, let's only consider the functional case: we are just iterating a map $y = \tau(x)$.

But if τ is length-preserving then all orbits must be finite, in fact they cannot be longer than $|\Sigma|^{|x|}$.

Theorem

For length-preserving transductions, transitive closure is PSPACE*-complete in general.*

Length-Preserving Example

Recall that x^{op} is the word x written backwards.

It is clear that the map $x \mapsto x^{\text{op}}$ cannot be computed by a FSM.

But iteration of a length-preserving transduction can be used to "compute" x^{op} as follows.

Define a new alphabet $\Gamma = \Sigma \cup \{\, \overline{a} \mid a \in \Sigma \,\}$.

There is a length-preserving rational function τ such that $\tau(\varepsilon) = \varepsilon$ and

$$\tau(a\,uv) = u\,\overline{a}\,v$$

where $au \in \Sigma^\star$ and $v \in \overline{\Sigma}^\star$. Let f be the homomorphism $f(\overline{a}) = a$. Then

$$x^{\text{op}} = f\left(x\,\mathsf{tcl}(\tau) \cap \overline{\Sigma}^\star\right)$$

Fixed-Points

But as soon as we try to make a global statement, decidability vanishes. For example:

Proposition

It is undecidable whether all orbits of a functional length-preserving transduction end in a fixed point.

Sketch of proof.

Simulate a Turing machine without input, operating on bounded memory. Set things up so that all orbits end in a fixed point iff the Turing machine computation diverges.

So the fixed point means: the computation has run out of space. If, on the other hand, the computation converges for some sufficiently long initial setup, then we periodically repeat the whole computation.

□

In fact this problem turns out to be co-r.e.-complete.