Automata Theory for Language

Name: Thomas Graf
Email: mathcamp@thomasgraf.net
Course Website: mathcamp.thomasgraf.net
Personal Website: thomasgraf.net

1 Language as a Mathematical Problem

Every language follows certain rules of grammar. I do not mean the usual grammar-nazi malarkey like "Do not split infinitives! Do not strand prepositions!". Rather, there are words and sentences that are correct, and others that have something wrong with them.

Example 1

Consider the English word *denaturalization*. It is built up from discrete parts:

1. nature
2. nature + al = natural
3. natural + ize = naturalize
4. de + naturalize = denaturalize
5. denaturalize + ation = denaturalization

No other order of these parts produces a good word of English:
denaturizalation, *ationalizenaturde*, *naturalizedeation*, ...

[Exercise 1] How many combinations are mathematically possible?

This restriction to just a tiny fraction of all conceivable patterns also occurs with sentences.

Example 2

The words in the English sentence *These two former presidents seem exhausted* can only be arranged in this order, none of the other 719 combinations are possible (unless you're Yoda).

That's quite surprising if you think about it. Many questions immediately come to mind:

Q1 Are there languages that allow all possible combinations?

A1 No, there aren't. Languages differ in the restrictions they put on words and sentences, but every language allows only a tiny fraction of the possible patterns.

Q2 So why are there are no languages that allow all combinations?

A2 Excellent question, we'll see the answer at the end.

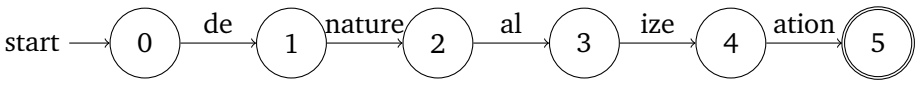
Q3 If languages use different restrictions, how different are those restrictions? For instance, can a language require that words are ordered by length?

A3 Linguists have studied thousands of languages and have determined that their restrictions are very similar. Not a single language does odd things like ordering words by length. We'll see later why this is the case.

Just like any other cognitive ability (seeing, hearing, calculating, learning, ...), language involves the human brain. So there must be some cognitive mechanism in our brains that allows us to produce correct forms while avoiding incorrect ones. Looking at neurons and synapses won't get us far, the brain is just too complicated. Instead, we will use math to model this cognitive mechanism.

2 Graphs for Language: Finite-State Automata

Our mathematical model are *finite-state automata* (FSA), which are a special case of labeled graphs. Here is a simple example of an automaton representing *denationalization*:



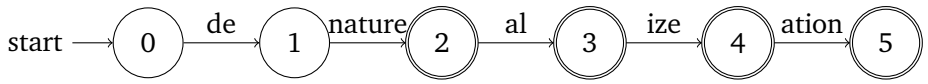
- The circles are the vertices of the graph, which are called *states*.
- The *arcs* connecting the states all must have a label.
- The *start* arrow indicates the beginning point of the automaton, called the *initial state*.
- The double circled state is a *final state*. It marks the end point of the automaton.

An FSA can be used to succinctly describe words and sentences. Every path through the automaton that takes you from an initial state to a final state represents a well-formed structure. In the automaton above, there is only one such path, which takes us from 0 to 1, then to 2, 3, 4, and finally 5. But automata can have multiple paths through them.

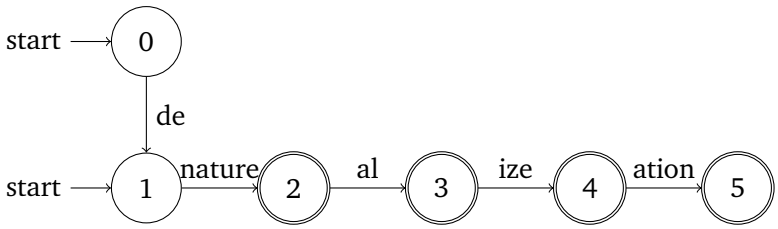
Example 3

Besides *denaturalization*, English also has the words *nature*, *natural*, *naturalize*, *denaturalize*, and *naturalization*. We can modify the previous automaton so that it also allows these other forms. Let's do this step by step.

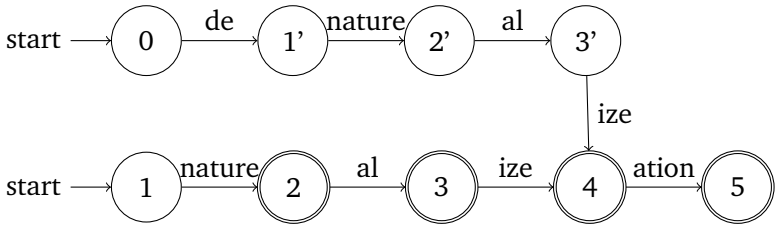
First we want to allow words to also end with *nature*, *al*, and *ize*, so we make the states that are reached through those arcs final.



But every path must still start with *de*, so the automaton does not allow the patterns *nature*, *natural*, or *naturalize*. To fix this, we make 1 an initial state, too.

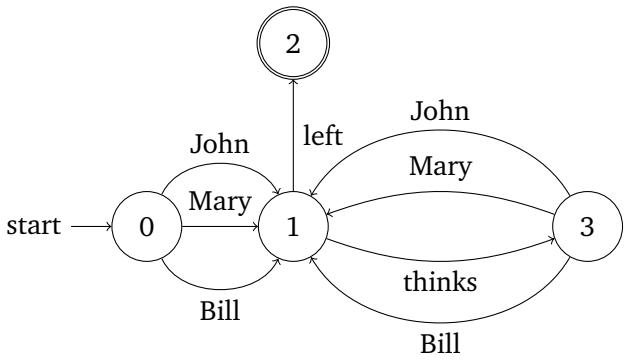


But this automaton is too general, it also allows patterns like *denature* or *denatural*. This is a little tricker to fix: we add a few more states to keep track of the fact that after *de*, the first final state can only be one reached by *ize*.



When you now try all possible paths from an initial state to a final state, you'll see that they are exactly *denaturalize*, *denaturalization*, *nature*, *natural*, *naturalize*, and *naturalization*.

FSAs can be much more complex than this. As long as the number of states is finite, pretty much any graph you can imagine is an FSA. You can have multiple arcs leaving a node, multiple arcs leading to the same node, or arcs that lead back to a previous node, possibly even the one they came from (*loops*). That is to say, you can have convoluted automata like the one below:



[Exercise 2] What collection of sentences is described by the automaton above?

[Exercise 3] Your grandfather's father is your *great grandfather*, whose father is your *great great grandfather*, whose father is your *great great great grandfather*, and so on. Try to write an FSA that captures all those patterns, and only those.

[Exercise 4] Of course we also have *great grandmother*, *great great grandmother*, and so on. How should the automaton from the previous example be modified to also allow these words?

[Exercise 5] Obviously *I like myself* is a sentence of English. Somewhat less obviously, *I like myself and myself* is also a sentence of English. Imagine the following exchange:

Mary: John, you're such a misanthropist. Are there even two people that you like?
John: Sure! I like myself and myself.

If Mary asks for three people, John can sarcastically reply *I like myself and myself and myself*, and so on for any number. Write an FSA that allows all sentences of the form *I like myself*, *I like myself and myself*, *I like myself and myself and myself*, and so on. The automaton should not allow any other sentences.

[Exercise 6] Pretty much any proper name of English can be substituted in the *I like myself* sentences above. If we ignore all proper names except *John* and *Mary*, then we can also get patterns like *I like John*, *I like Mary*, *I like John and Mary*, *I like myself and John*, *I like myself and Mary and Mary*, and so on. Modify the FSA from the previous exercise so that it also allows these patterns.

3 The Limits of Finite-State Automata (aka the Hard Part)

FSAs are fairly powerful devices, and they have found many applications in the real world. They control elevators, help biologists with genome sequencing, give you word suggestions when you write a text message, and are even involved in the automatic creation of subtitles for Youtube videos. Tons of technology uses FSAs, including language technology.

But FSAs cannot do everything. There is a very strict bound on the complexity of the patterns they can handle. Remember that we noticed that

1. no language allows words or parts of words to be freely arranged, and
2. no language requires words to be linearly ordered by their length.

Both properties follow from a crucial limitation of FSAs, which is expressed by the *Myhill Nerode Theorem*.

Before we state the theorem, let us think about how FSAs actually work. In particular, what do the states stand for? Suppose you have taken a path that leads to state 1 in the convoluted automaton from the previous section. How can you continue to get a well-formed pattern? Well, by taking any path that leads you from 1 to a final state. There may be multiple paths to choose from, but anyone will do. It does not matter how you actually got into 1. Whether you came from 0 to 1 via *John*, *Mary*, or *Bill* is completely irrelevant, the only thing that matters is that you are in state 1 now and may take any path that gets you from 1 to 2.

Crucial Insight 1 If you have paths that take you from an initial state to the same state, then those paths can be continued in exactly the same fashion towards a final state.

FSAs have only finitely many states. Suppose we have some pattern, and we compare all paths that could possibly arise in that pattern. We put two paths in the same bin if they can be continued in exactly the same fashion. Then the pattern can be handled by an FSA only if we end up with a finite number of distinct bins.

Example 4

Consider the pattern *grandfather*, *great grandfather*, *great great grandfather*, and so on. If the pattern starts with *grandfather*, then there is nothing else that can be added, so there are no continuation paths at all. For *great*, one possible continuation path is *grandfather*, producing *great grandfather*. But we could also continue with *great grandfather*, *great great grandfather*, and so on. Here's a table:

Incoming Path	Continuation Paths
grandfather	—
great	grandfather, great grandfather, ...
great great	grandfather, great grandfather, ...
great great great	grandfather, great grandfather, ...
⋮	

So there's only two bins for paths. One is for *grandfather*, which cannot be continued. The other one is for any path that starts with *great*, as they can all be continued by an arbitrary number of *greats* followed by *grandfather*. Since we only have two bins, the pattern can be handled by an FSA.

Crucial Insight 2 A pattern can be captured by an FSA only if its paths can be classified into finitely many bins.

The Myhill-Nerode theorem is simply a more precise formulation of these two insights. It allows us to explain the two properties of linguistic patterns we observed earlier.

Example 5 No Free Ordering

Every language has sentences like English *And John slept and Mary yawned and Bill cooked and* . . . . This is an FSA-pattern.

If the words in these sentences could be freely arranged, this would give us many new sentences, including those where we first have all instances of *and*, then the proper names, and then the *verbs*. We can once again analyze the paths in the form of a table. For simplicity, let us represent all proper names as N and all verbs as V.

Incoming Path	Continuation Paths
and	N V, and N N V V, . . .
and and	N N V V, and N N N V V V, . . .
and and and	N N N V V V, and N N N N V V V V. . .

As you can see, no two rows in the table have the same continuation paths. Since there is upper bound on how many instances of *and* occur at the beginning of a sentence, we do not get a finite number of distinct bins for the path. Hence this pattern cannot be created by any FSA, no matter how large and complicated it is.

Any FSA that tries to produce this pattern will eventually make mistakes and end up with the wrong number of names or verbs, for instance, *and and and John Mary slept yawned cooked ate*. Changing the order of the words has made the original FSA-pattern too hard for FSAs.

Example 6 No Ordering by Length

Now suppose that we want to order the words in a sentence by length. We already saw that English contains the words *grandfather*, *great grandfather*, *great great grandfather*, and so on. Even though these are spelled as separated words, one can show that each one is actually one large word. For the sake of clarity, let us use an underscore instead of space for these words. Among the sentences of English, we find the pattern *My grandfather likes his great_grandfather*, *My great_grandfather likes his great_great_grandfather*, and so on. Ordered by lengths, these sentences would be *my his likes grandfather great_grandfather*, *my his likes great_grandfather great_great_grandfather*, . . .

Once again we draw a table to compare the paths. We skip the paths *my*, *my his*, and *my his likes*, as they do not matter for the argument.

Incoming Path	Continuation Paths
my his likes grandfather	great_grandfather
my his likes great_grandfather	great_great_grandfather
my his likes great_great_grandfather	great_great_great_grandfather

We have the same problem as before: each row has different continuation paths, so we end up with infinitely many bins and the pattern cannot be captured by an FSA. Any attempt to model the pattern with an FSA will also produce sentences of the form *my his likes great*^{n} *grandfather great*^{m} *grandfather*, $m < n$.

We have accomplished something that few people would consider possible: we have studied language from a mathematical perspective, and we were able to explain properties that are shared by all human languages by purely mathematical means.

The Take Home Message. Mathematics explains language!

[Exercise 7] Word structure indeed seems to be fully captured by FSAs. But there are ways to show that the structure of English sentences is more complicated then what FSAs can handle. Do you have an idea what the argument might be?

Hint: The sentences below play a crucial role.

- A man left.
- A man that a woman saw left.
- A man that a woman that a man saw saw left.

4 I Wanna Learn More!

Cool, here's a few pointers. If you have any questions, just shoot me a line!

- **Automata Theory**
  - *Introduction to the Theory of Computation*
    undergraduate, computer science
  - Michael Sipser. 2012. 3rd edition.
    Introduction to the Theory of Computation. (yep, that's the same name)

    Pro tip: The 2nd edition is almost the same and used copies are much cheaper. A decent scan can easily be found on the first Google page, I'm not sure about its legality.
- **Computational Linguistics**
  - *Language and Technology*
    undergraduate, linguistics
  - *Computational Linguistics*
    undergraduate, linguistics
  - *Natural Language Processing*
    undergraduate, computer science
  - over 10 graduate-level courses
  - Markus Dickinson, Chris Brew, and Detmar Meurers. 2012.
    Language and Computers.

    Used copies are available for 10 bucks on Amazon.
- **Language from a Formal and Cognitive Perspective**
  - Lecture notes for my graduate-level course Computational Linguistics: lin637.thomasgraf.net

  - Steven Pinker. *The Language Instinct: How the Mind Creates Language*.

- **The Human Mind as a Computer**
  - Douglas Hofstadter. *Gödel, Escher, Bach: An Eternal Golden Braid*.