CSE 311 Lecture 26: Limitations of DFAs, NFAs, and Regular Expressions

Emina Torlak and Sami Davies

Topics

DFAs $\equiv$ NFAs $\equiv$ regular expressions

A quick review of Lecture 25.

Languages and representations

Regular, context-free, and other languages.

Proving irregularity

A proof template for showing that a language is not regular.

DFAs $\equiv$ NFAs $\equiv$ regular expressions

A quick review of Lecture 25.

Equivalence of DFAs, NFAs, and regular expressions

We have shown how to build an optimal DFA for every regular expression.

Build an NFA.

Convert the NFA to a DFA using the subset construction.

Minimize the resulting DFA.

Equivalence of DFAs, NFAs, and regular expressions

We have shown how to build an optimal DFA for every regular expression.

Build an NFA.

Convert the NFA to a DFA using the subset construction.

Minimize the resulting DFA.

Theorem

A language is recognized by a DFA (or NFA) if and only if it has a regular expression.

You need to know this fact but we won't ask you anything about the "only if" direction from DFAs/NFAs to regular expressions.

Equivalence of DFAs, NFAs, and regular expressions

We have shown how to build an optimal DFA for every regular expression.

Build an NFA.

Convert the NFA to a DFA using the subset construction.

Minimize the resulting DFA.

Theorem

A language is recognized by a DFA (or NFA) if and only if it has a regular expression.

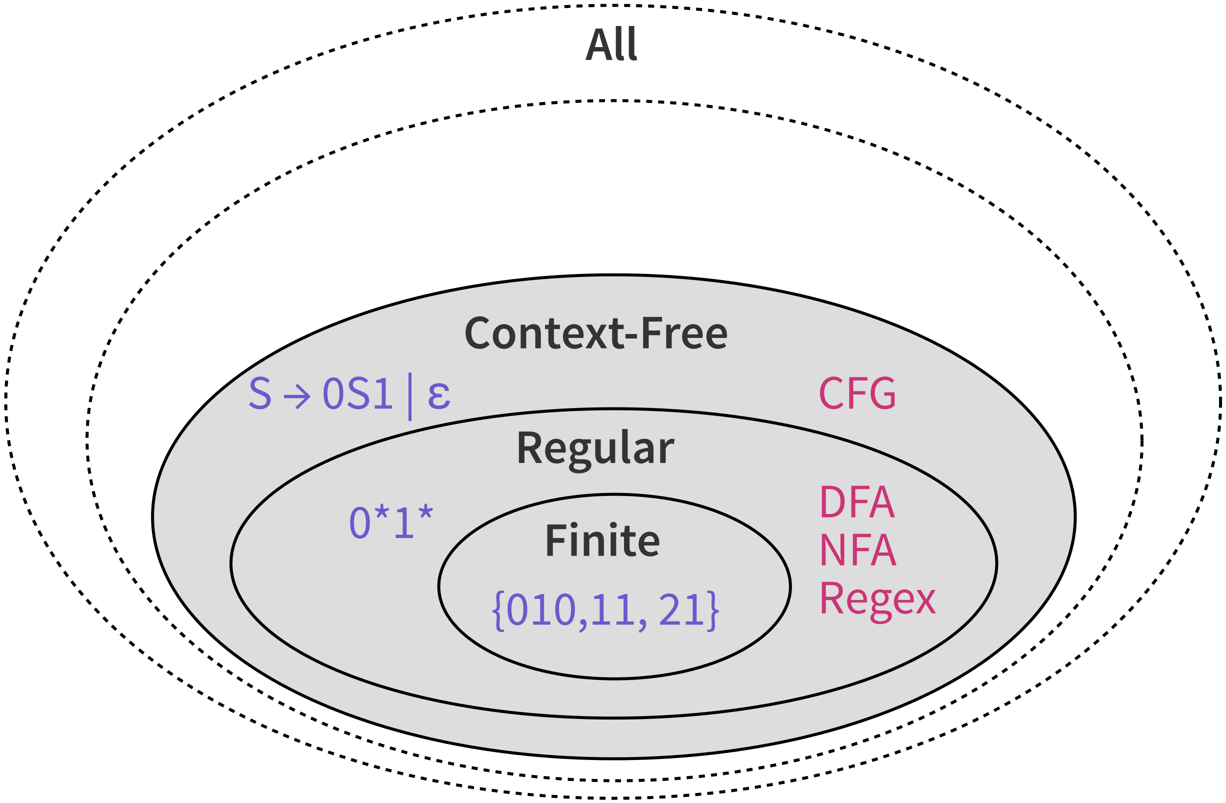
You need to know this fact but we won't ask you anything about the "only if" direction from DFAs/NFAs to regular expressions.

Languages represented by NFAs, DFAs, and regular expressions are called *regular languages*.

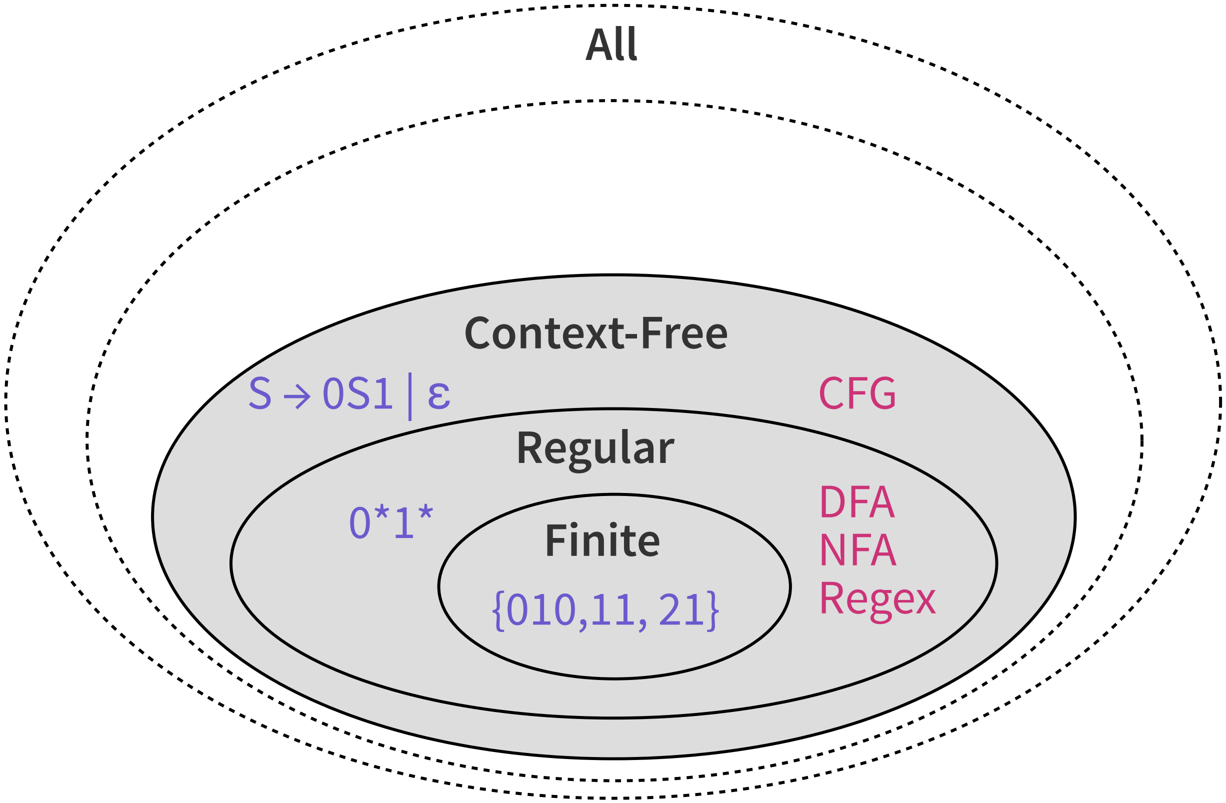
Languages and representations

Regular, context-free, and other languages.

A hierarchy of languages and representations

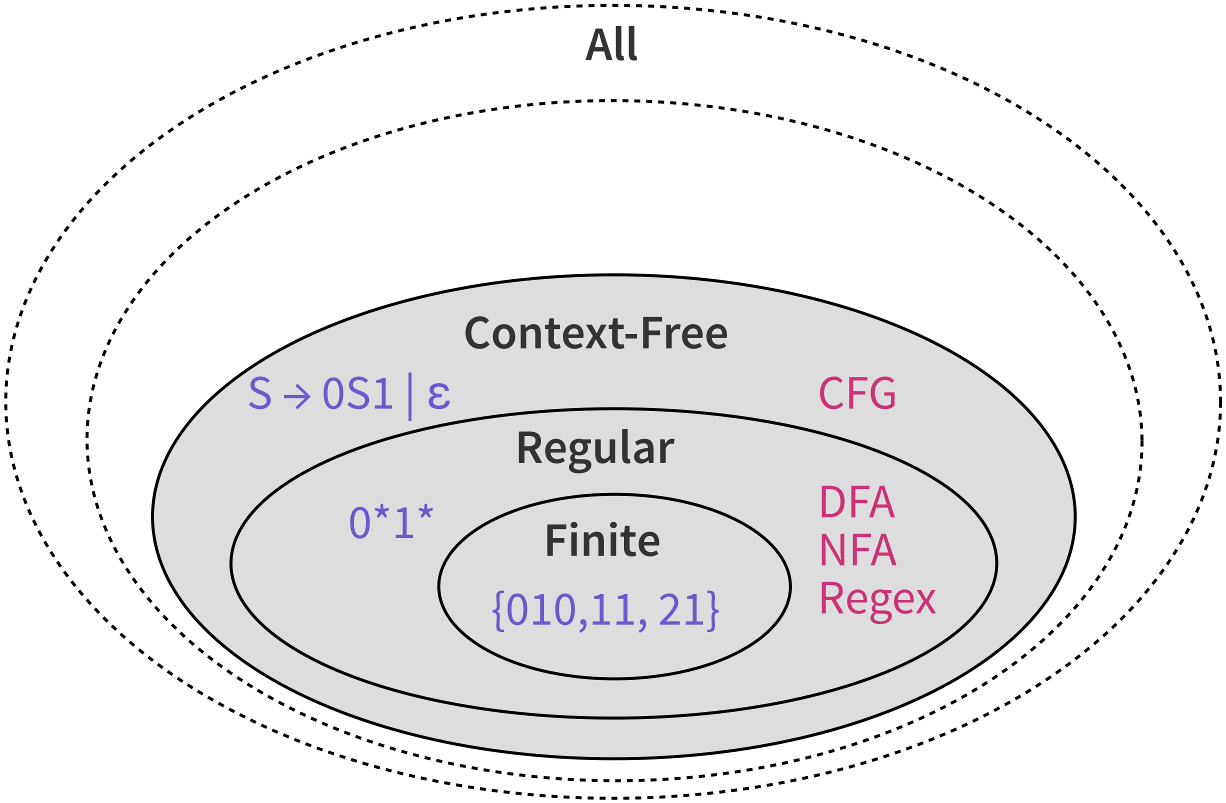


A hierarchy of languages and representations



How do we prove that all finite languages are regular?

A hierarchy of languages and representations



How do we prove that all finite languages are regular? By showing how to construct a DFA/NFA/RegEx for every finite language!

Converting a finite language to a regular expression

Converting a finite language to a regular expression

Convert each string in the language L to a regular expression.

This is just each string itself.

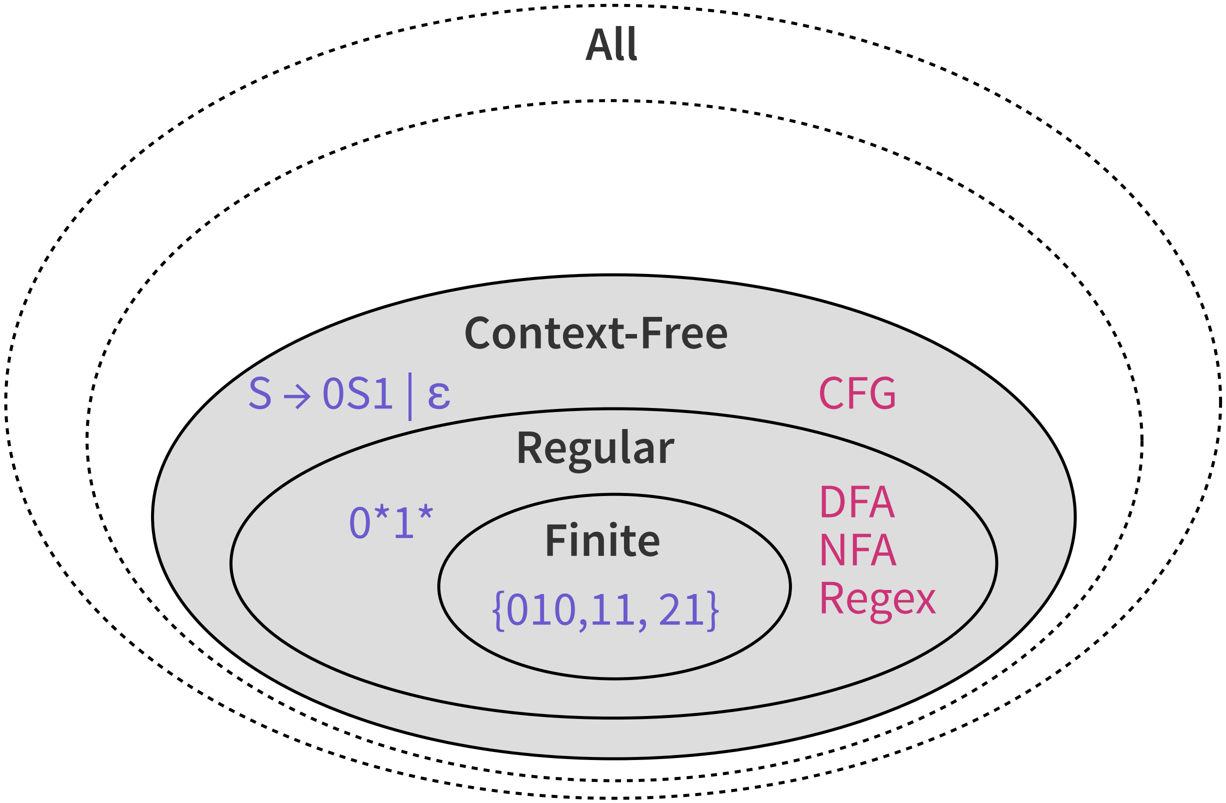
Then put these regular expressions together using the $\cup$ operator.

The resulting regular expression accepts exactly the strings in L.

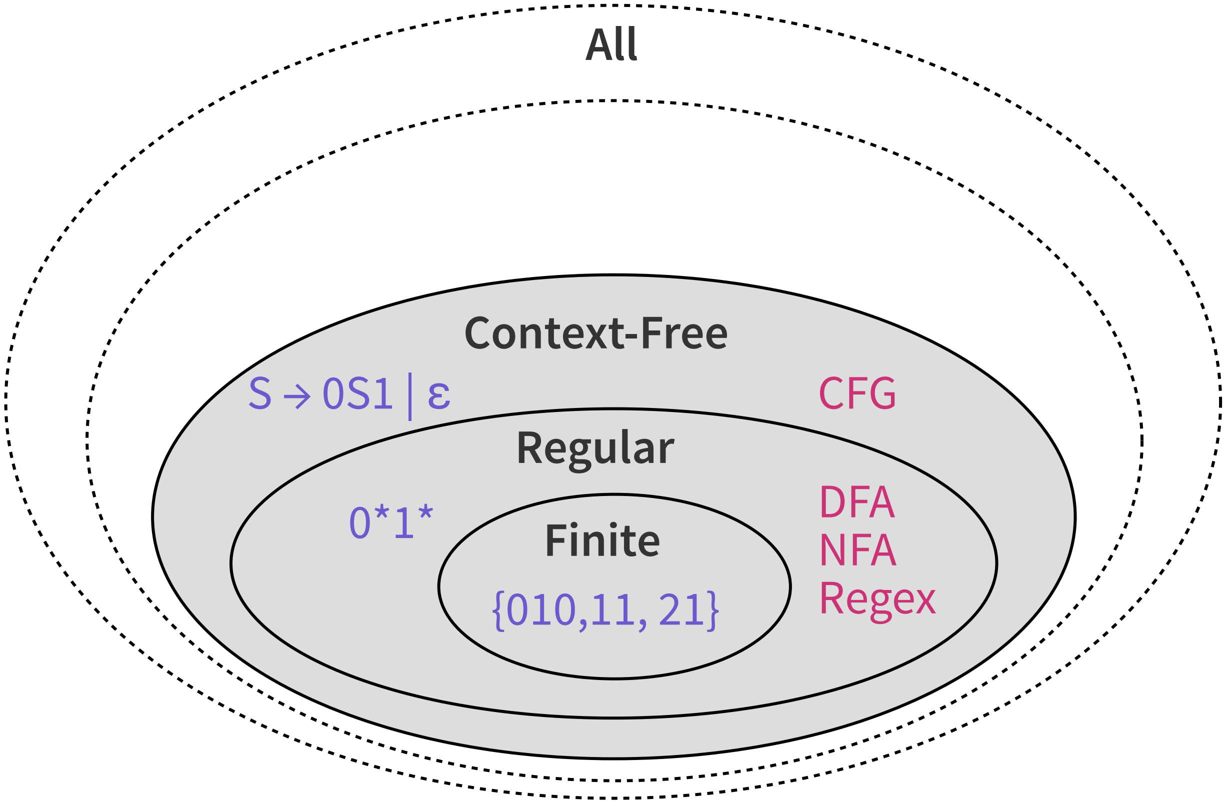
Example

$$\{010, 11, 21\} \longrightarrow 010 \cup 11 \cup 21$$

Back to languages and representations …

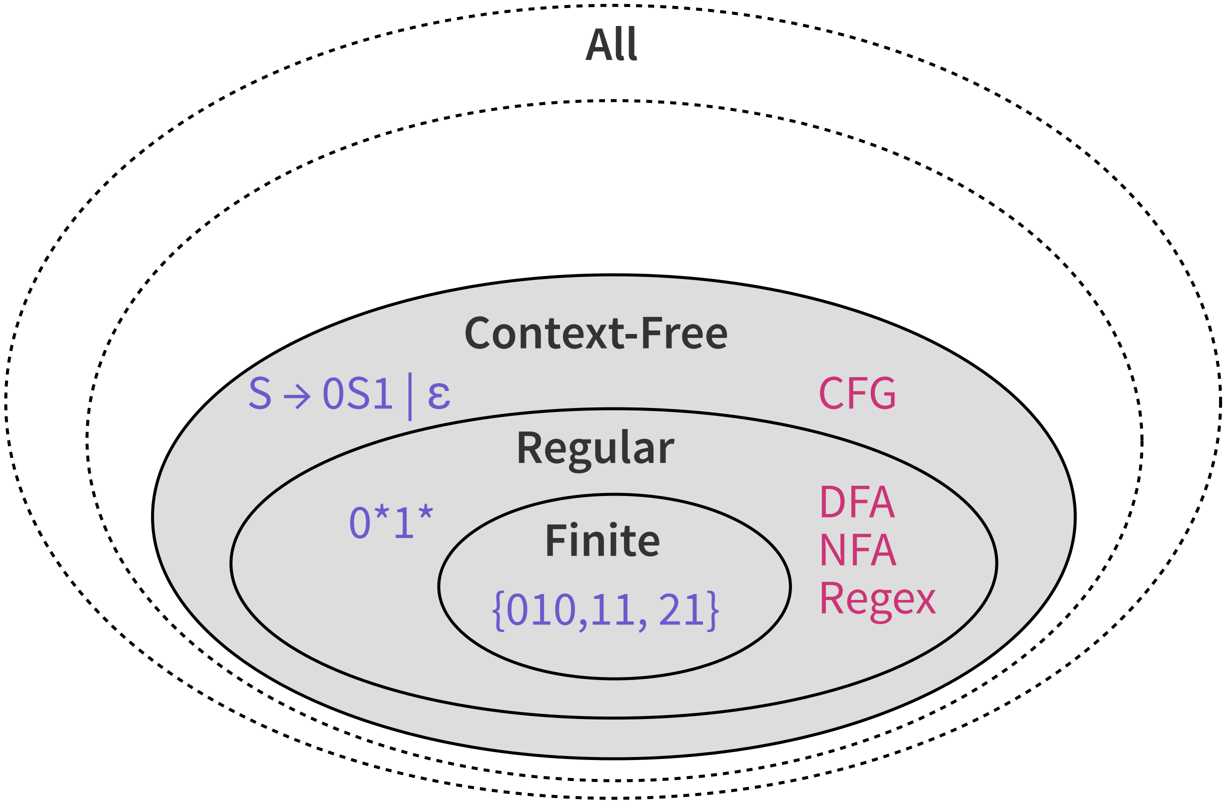


Back to languages and representations ...



How do we prove that all regular languages are context-free?

Back to languages and representations …



How do we prove that all regular languages are context-free? By showing how to construct a CFG for every regular language!

Converting a regular expression to a CFG

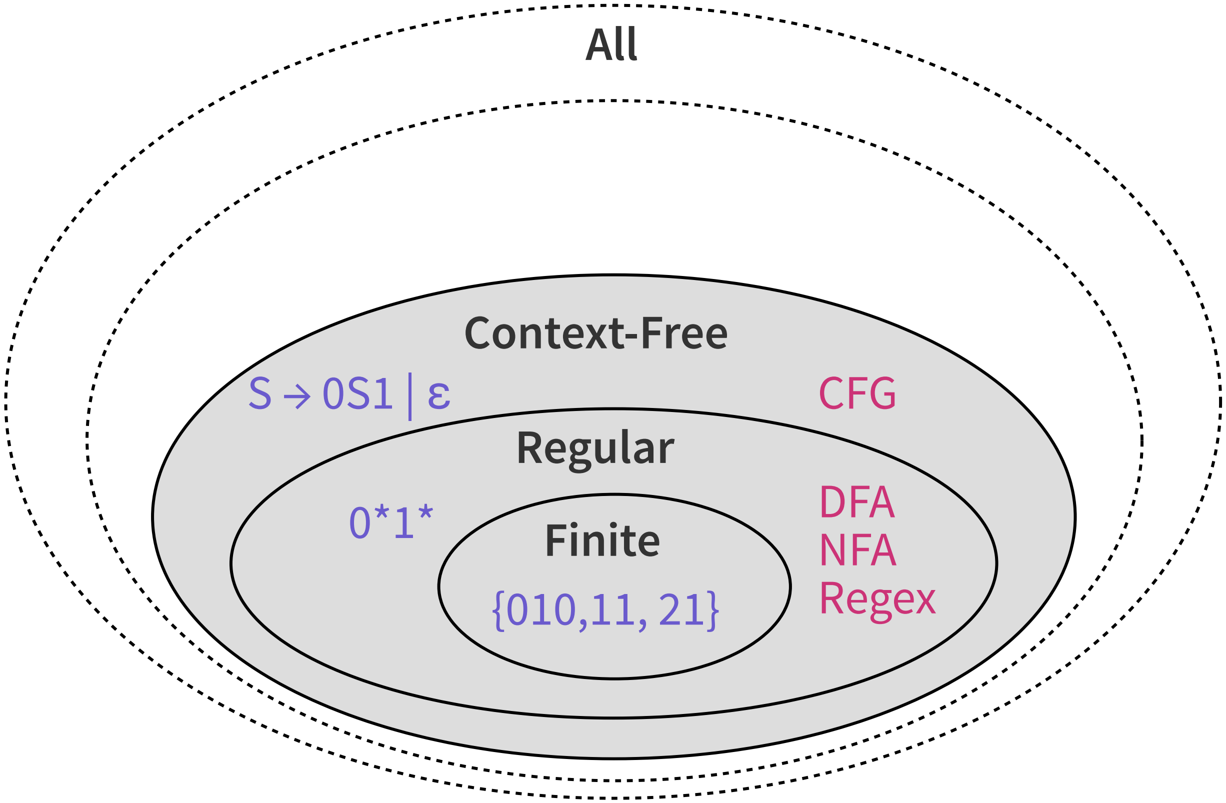
Use the following function on regular expressions:

$\mathsf{cfg}(\emptyset) =$ the CFG with no productions.
$\mathsf{cfg}(\varepsilon) =$ the CFG with just the production $S_\varepsilon \to \varepsilon$.
$\mathsf{cfg}(a) =$ the CFG with just the production $S_a \to a$ for every $a \in \Sigma$.
$\mathsf{cfg}(AB) =$ the CFG with the productions $\mathsf{cfg}(A)$, $\mathsf{cfg}(B)$, and $S_{AB} \to S_A S_B$.
$\mathsf{cfg}(A \cup B) =$ the CFG with the productions $\mathsf{cfg}(A)$, $\mathsf{cfg}(B)$, and $S_{A\cup B} \to S_A \mid S_B$.
$\mathsf{cfg}(A^*) =$ the CFG with the productions $\mathsf{cfg}(A)$ and $S_{A*} \to \varepsilon \mid S_A S_{A*}$.

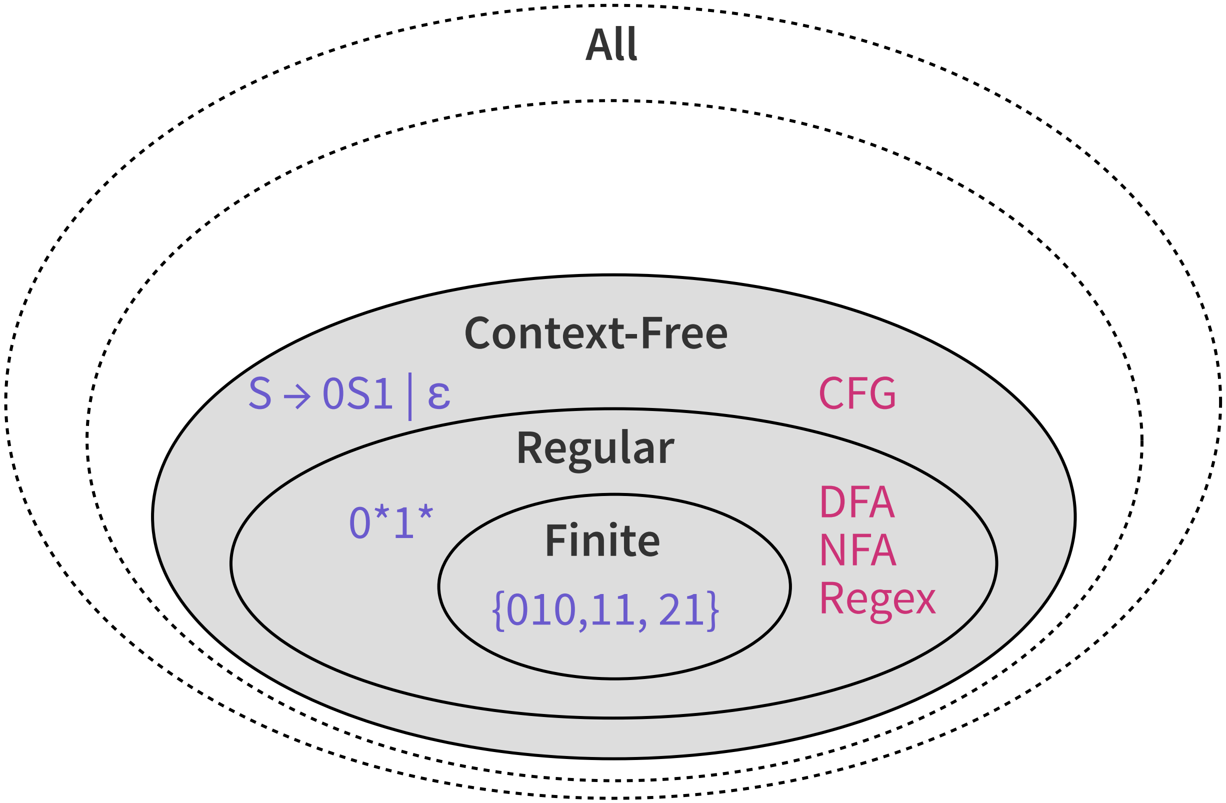
Example: $\mathsf{cfg}((1 \cup 0)^*0)$

$$\begin{aligned}
S_{(1\cup 0)^*0} &\to S_{(1\cup 0)^*} S_0 \\
S_{(1\cup 0)^*} &\to \varepsilon \mid S_{1\cup 0} S_{(1\cup 0)^*} \\
S_{(1\cup 0)} &\to S_1 \mid S_0 \\
S_1 &\to 1 \\
S_0 &\to 0
\end{aligned}$$

Back again to languages and representations …



Back again to languages and representations …



We saw in Lecture 21 that the language B of all binary palindromes can be represented by a CFG. We also said that B can't be represented by any regular expression. How would you prove that?

Why isn't B (binary palindromes) a regular language?

If B were regular, we could express it as a DFA/NFA/RegEx.

Let's choose a DFA as our hypothetical representation.

Why isn't B (binary palindromes) a regular language?

If B were regular, we could express it as a DFA/NFA/RegEx.

Let's choose a DFA as our hypothetical representation.

Now, recall that B consists of infinitely many strings $wv\overline{w}$

where $\overline{w}$ is the reverse of w and $v \in \{\varepsilon, \text{“}0\text{”}, \text{“}1\text{”}\}$.

Why isn't B (binary palindromes) a regular language?

If B were regular, we could express it as a DFA/NFA/RegEx.

Let's choose a DFA as our hypothetical representation.

Now, recall that B consists of infinitely many strings $wv\overline{w}$

where $\overline{w}$ is the reverse of w and $v \in \{\varepsilon, \text{“0”}, \text{“1”}\}$.

What would a DFA need to keep track of to decide B?

Why isn't B (binary palindromes) a regular language?

If B were regular, we could express it as a DFA/NFA/RegEx.

Let's choose a DFA as our hypothetical representation.

Now, recall that B consists of infinitely many strings $wv\overline{w}$

where $\overline{w}$ is the reverse of w and $v \in \{\varepsilon, \text{“0”}, \text{“1”}\}$.

What would a DFA need to keep track of to decide B?

It would need to keep track of w in order to check $\overline{w}$ against it.

Why isn't B (binary palindromes) a regular language?

If B were regular, we could express it as a DFA/NFA/RegEx.

Let's choose a DFA as our hypothetical representation.

Now, recall that B consists of infinitely many strings $wv\overline{w}$

where $\overline{w}$ is the reverse of w and $v \in \{\varepsilon, \text{``0''}, \text{``1''}\}$.

What would a DFA need to keep track of to decide B?

It would need to keep track of w in order to check $\overline{w}$ against it.

But there are infinitely many possible w's and finitely many DFA states!

Why isn't B (binary palindromes) a regular language?

If B were regular, we could express it as a DFA/NFA/RegEx.

Let's choose a DFA as our hypothetical representation.

Now, recall that B consists of infinitely many strings $wv\overline{w}$

where $\overline{w}$ is the reverse of w and $v \in \{\varepsilon, \text{“0”}, \text{“1”}\}$.

What would a DFA need to keep track of to decide B?

It would need to keep track of w in order to check $\overline{w}$ against it.

But there are infinitely many possible w's and finitely many DFA states!

This is the intuition for why B is not regular. Let's see how to turn this intuition into a formal proof.

A strategy for proving that B is not regular

Proof by contradiction:

Assume that B is regular.

Therefore, there is a DFA M that recognizes B.

Show that M accepts or rejects a string it shouldn't.

A strategy for proving that B is not regular

Proof by contradiction:

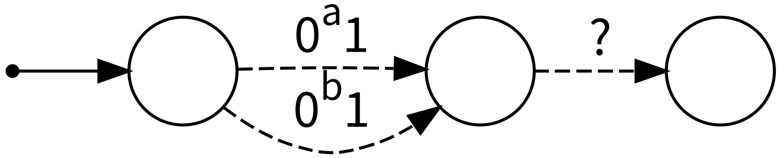
Assume that B is regular.

Therefore, there is a DFA M that recognizes B.

Show that M accepts or rejects a string it shouldn't.

Key Idea 1

If two string prefixes collide by reaching the same state, a DFA can no longer distinguish their suffixes.



A strategy for proving that B is not regular

Proof by contradiction:

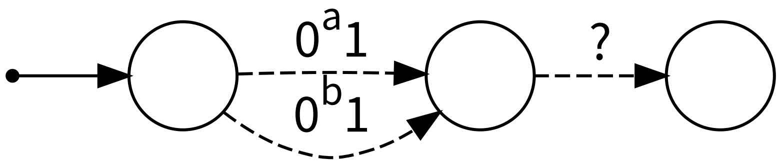
Assume that B is regular.

Therefore, there is a DFA M that recognizes B.

Show that M accepts or rejects a string it shouldn't.

Key Idea 1

If two string prefixes collide by reaching the same state, a DFA can no longer distinguish their suffixes.



Key Idea 2

The machine M has finitely many states, and since the strings in B have infinitely many distinct prefixes, two of them must collide!

A strategy for proving that B is not regular

Proof by contradiction:

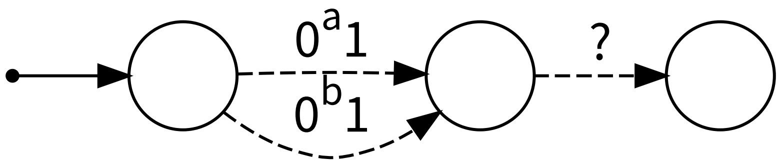
Assume that B is regular.

Therefore, there is a DFA M that recognizes B.

Show that M accepts or rejects a string it shouldn't.

Key Idea 1

If two string prefixes collide by reaching the same state, a DFA can no longer distinguish their suffixes.



Key Idea 2

The machine M has finitely many states, and since the strings in B have infinitely many distinct prefixes, two of them must collide!

We choose an infinite set S of prefixes. This choice must ensure that for *every pair* of prefixes in $s_a \neq s_b \in S$, there is a suffix t such that one of of $s_a t$, $s_b t$ is in B but not the other.

$$S = \{1, 01, 001, 0001, \ldots\}$$
$$= \{0^n 1 : n \geq 0\}$$

Proving that B is not regular

Suppose that some DFA M recognizes B. We show M accepts or rejects a string it shouldn't. Consider the set $S = \{0^n 1 : n \geq 0\}$.

Proving that B is not regular

Suppose that some DFA M recognizes B. We show M accepts or rejects a string it shouldn't. Consider the set $S = \{0^n 1 : n \geq 0\}$.

Since there are finitely many states in M and infinitely many strings in S, there exist strings $0^a 1 \in S$ and $0^b 1 \in S$ with $a \neq b$ that end in the same state of M.

Proving that B is not regular

Suppose that some DFA M recognizes B. We show M accepts or rejects a string it shouldn't. Consider the set $S = \{0^n1 : n \geq 0\}$.

Since there are finitely many states in M and infinitely many strings in S, there exist strings $0^a1 \in S$ and $0^b1 \in S$ with $a \neq b$ that end in the same state of M.

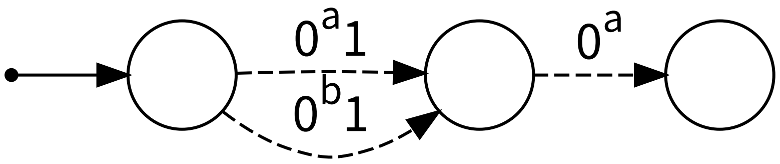
Important: We don't get to choose a and b! We just know they exist.

Proving that B is not regular

Suppose that some DFA M recognizes B. We show M accepts or rejects a string it shouldn't. Consider the set $S = \{0^n 1 : n \geq 0\}$.

Since there are finitely many states in M and infinitely many strings in S, there exist strings $0^a 1 \in S$ and $0^b 1 \in S$ with $a \neq b$ that end in the same state of M.

Now, consider appending 0^a to both strings.

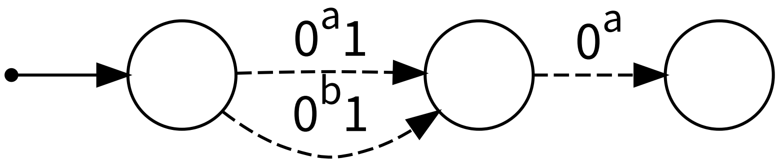


Proving that B is not regular

Suppose that some DFA M recognizes B. We show M accepts or rejects a string it shouldn't. Consider the set $S = \{0^n 1 : n \geq 0\}$.

Since there are finitely many states in M and infinitely many strings in S, there exist strings $0^a 1 \in S$ and $0^b 1 \in S$ with $a \neq b$ that end in the same state of M.

Now, consider appending 0^a to both strings.



Since $0^a 1$ and $0^b 1$ end in the same state, so do $0^a 10^a$ and $0^b 10^a$, call it q. But then M must make a mistake: q needs to be an accept state since $0^a 10^a \in B$, but then M would accept $0^b 10^a \notin B$, which is an error. $\square$

Proving irregularity

A proof template for showing that a language is not regular.

A template for proving that a language L is not regular

① Suppose for contradiction that some DFA M recognizes L.

A template for proving that a language L is not regular

① Suppose for contradiction that some DFA M recognizes L.

② Consider the set $S = \{\dots\}$.

S must be infinite, and for *every pair* of prefixes $s_a \neq s_b \in S$, there is a suffix t such that one of of $s_a t, s_b t$ is in L but not the other.

A template for proving that a language L is not regular

① Suppose for contradiction that some DFA M recognizes L.

② Consider the set $S = \{\dots\}$.

③ Since S is infinite and M has finitely many states, there must be two strings $s_a, s_b \in S$ such that $s_a \neq s_b$ and both end in the same state of M.

S must be infinite, and for *every pair* of prefixes $s_a \neq s_b \in S$, there is a suffix t such that one of of $s_a t, s_b t$ is in L but not the other.

We don't get to choose s_a and s_b!

A template for proving that a language L is not regular

① Suppose for contradiction that some DFA M recognizes L.

② Consider the set $S = \{\dots\}$.

③ Since S is infinite and M has finitely many states, there must be two strings $s_a, s_b \in S$ such that $s_a \neq s_b$ and both end in the same state of M.

④ Consider appending t to both s_a and s_b.

S must be infinite, and for *every pair* of prefixes $s_a \neq s_b \in S$, there is a suffix t such that one of of $s_a t, s_b t$ is in L but not the other.

We don't get to choose s_a and s_b!

t should be an accept suffix for s_a but not s_b.

A template for proving that a language L is not regular

① Suppose for contradiction that some DFA M recognizes L.

② Consider the set $S = \{\ldots\}$.

③ Since S is infinite and M has finitely many states, there must be two strings $s_a, s_b \in S$ such that $s_a \neq s_b$ and both end in the same state of M.

④ Consider appending t to both s_a and s_b.

⑤ Since s_a and s_b end in the same state of M, then $s_a t$ and $s_b t$ also end in the same state q of M. Since $s_a t \in L$ and $s_b t \notin L$, M does not recognize L.

S must be infinite, and for *every pair* of prefixes $s_a \neq s_b \in S$, there is a suffix t such that one of of $s_a t, s_b t$ is in L but not the other.

We don't get to choose s_a and s_b!

t should be an accept suffix for s_a but not s_b.

A template for proving that a language L is not regular

① Suppose for contradiction that some DFA M recognizes L.

② Consider the set $S = \{\ldots\}$.

③ Since S is infinite and M has finitely many states, there must be two strings $s_a, s_b \in S$ such that $s_a \neq s_b$ and both end in the same state of M.

④ Consider appending t to both s_a and s_b.

⑤ Since s_a and s_b end in the same state of M, then $s_a t$ and $s_b t$ also end in the same state q of M. Since $s_a t \in L$ and $s_b t \notin L$, M does not recognize L.

⑥ Since M was arbitrary, no DFA recognizes L.

S must be infinite, and for *every pair* of prefixes $s_a \neq s_b \in S$, there is a suffix t such that one of of $s_a t, s_b t$ is in L but not the other.

We don't get to choose s_a and s_b!

t should be an accept suffix for s_a but not s_b.

Example: prove that $L = \{0^n 1^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S =$

Example: prove that $L = \{0^n 1^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{0^n : n \geq 0\}$.

Example: prove that $L = \{0^n 1^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{0^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $0^a, 0^b \in S$ with $a \neq b$ that both end in the same state of M.

Example: prove that $L = \{0^n 1^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{0^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $0^a, 0^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending   to both 0^a and 0^b.

Example: prove that $L = \{0^n 1^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{0^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $0^a, 0^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending 1^a to both 0^a and 0^b.

Example: prove that $L = \{0^n 1^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{0^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $0^a, 0^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending 1^a to both 0^a and 0^b.

Since 0^a and 0^b end in the same state of M, then $0^a 1^a$ and $0^b 1^a$ also end in the same state q of M. Since $0^a 1^a \in L$ and $0^b 1^a \notin L$, M does not recognize L.

Example: prove that $L = \{0^n 1^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{0^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $0^a, 0^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending 1^a to both 0^a and 0^b.

Since 0^a and 0^b end in the same state of M, then $0^a 1^a$ and $0^b 1^a$ also end in the same state q of M. Since $0^a 1^a \in L$ and $0^b 1^a \notin L$, M does not recognize L.

Since M was arbitrary, no DFA recognizes L.

Example: prove that $L = \{(^n)^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S =$

Example: prove that $L = \{(^n)^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{(^n : n \geq 0\}$.

Example: prove that $L = \{(^n)^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{(^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $(^a, (^b \in S$ with $a \neq b$ that both end in the same state of M.

Example: prove that $L = \{(^n)^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{(^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $(^a, (^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending   to both $(^a$ and $(^b$.

Example: prove that $L = \{(^n)^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{(^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $(^a, (^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending $)^a$ to both $(^a$ and $(^b$.

Example: prove that $L = \{(^n)^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{(^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $(^a, (^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending $)^a$ to both $(^a$ and $(^b$.

Since $(^a$ and $(^b$ end in the same state of M, then $(^a)^a$ and $(^b)^a$ also end in the same state q of M. Since $(^a)^a \in L$ and $(^b)^a \notin L$, M does not recognize L.

Example: prove that $L = \{(^n)^n : n \geq 0\}$ is not regular

Suppose for contradiction that some DFA M recognizes L.

Consider the set $S = \{(^n : n \geq 0\}$.

Since S is infinite and M has finitely many states, there must be two strings $(^a, (^b \in S$ with $a \neq b$ that both end in the same state of M.

Consider appending $)^a$ to both $(^a$ and $(^b$.

Since $(^a$ and $(^b$ end in the same state of M, then $(^a)^a$ and $(^b)^a$ also end in the same state q of M. Since $(^a)^a \in L$ and $(^b)^a \notin L$, M does not recognize L.

Since M was arbitrary, no DFA recognizes L.

A fun fact about this proof method

Suppose that for a language L, the set S is a *largest* set of prefix strings with the property that for every pair $s_a \neq s_b \in S$, there is some string t such that one of $s_a t$, $s_b t$ is in L but the other isn't.

A fun fact about this proof method

Suppose that for a language L, the set S is a *largest* set of prefix strings with the property that for every pair $s_a \neq s_b \in S$, there is some string t such that one of $s_a t$, $s_b t$ is in L but the other isn't.

If S is infinite, then L is not regular.

A fun fact about this proof method

Suppose that for a language L, the set S is a *largest* set of prefix strings with the property that for every pair $s_a \neq s_b \in S$, there is some string t such that one of $s_a t$, $s_b t$ is in L but the other isn't.

If S is infinite, then L is not regular.

If S is finite, then the minimal DFA for L has precisely $|S|$ states, one reached by each member of S.

Summary

DFAs $\equiv$ NFAs $\equiv$ regular expressions.

We've shown how to go from a regular expression to an NFA to a DFA.
(And going from an NFA to a DFA can lead to exponential blow-up.)
But we won't show how to go from an NFA/DFA to a regular expression.

Finite $\subset$ regular $\subset$ context-free languages.

To show that a language is regular, construct a DFA/NFA/RegEx for it.
To show that it is not regular, use the proof method from this lecture.