Conciseness from Non-Determinism

\[ L_n := \{ s \in (0 + 1)^+ | \text{n-th to the last bit of } s \text{ is 1}\} \]
\[ = (0 + 1)^*1(0 + 1)^{n-1} \quad \text{length is } O(n) \]

**Claim 1.** There is an NFA accepting \( L_n \) with \( n + 1 \) states

**Claim 2.** A DFA accepting \( L_n \) has at least \( 2^n \)-states
Conciseness from Non-Determinism

\[ L_n := \{ s \in (0 + 1)^+ | \text{n-th to the last bit of } s \text{ is } 1 \} \]
\[ = (0 + 1)^*1(0 + 1)^{n-1} \quad \text{length is } O(n) \]

**Claim 1.** There is an NFA accepting \( L_n \) with \( n + 1 \) states

**Sketch.** Initial state \( q_0 \) for \((0 + 1)^*\) plus \( n \) states for \( 1(0 + 1)^{n-1} \)

**Claim 2.** A DFA accepting \( L_n \) has at least \( 2^n \)-states
Conciseness from Non-Determinism

\[ L_n := \{ s \in (0 + 1)^+ \mid \text{n-th to the last bit of s is 1} \} \]
\[ = (0 + 1)^*1(0 + 1)^{n-1} \quad \text{length is } O(n) \]

**Claim 1.** There is an NFA accepting \( L_n \) with \( n + 1 \) states

**Sketch.** Initial state \( q_0 \) for \((0 + 1)^*\) plus \( n \) states for \( 1(0 + 1)^{n-1} \)

**Claim 2.** A DFA accepting \( L_n \) has at least \( 2^n \)-states

**Proof.** Let \( M \) be a DFA with \(< 2^n \) states.

On 2 strings \( s, s' \in (0 + 1)^n \), \( M \) ends up at the same state.

Let \( k \) be a string position where \( s \) and \( s' \) disagree.

Exactly one of \( s0^{k-1} \) and \( s'0^{k-1} \) is in \( L_n \); so \( M \) can't accept \( L_n \).
A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)

- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
L-inseparable histories

A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)

- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
- transitions \(is \xrightarrow{j} sj\) [remember only last \(n\) bits]
**L-inseparable histories**

A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)

- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
- transitions \(i \rightarrow j_s\) [remember only last \(n\) bits]
- set \(1(0 + 1)^{n-1}\) of final states
**L-inseparable histories**

A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)

- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
- transitions \(i \xrightarrow{j} s_j\) [remember only last \(n\) bits]
- set \(1(0 + 1)^{n-1}\) of final states

Given a language \(L\) over an alphabet \(\Sigma\), and strings \(s, s' \in \Sigma^*\),

- a string \(r\) **\(L\)-separates** \(s, s'\) if exactly one of \(sr\) and \(s'r\) is in \(L\)
**L-inseparable histories**

A DFA for $(0 + 1)^* 1(0 + 1)^{n-1}$

- state set $(0 + 1)^n$ with initial state $q_0 = 0^n$
- transitions $i^j \rightarrow s_j$ [remember only last $n$ bits]
- set $1(0 + 1)^{n-1}$ of final states

Given a language $L$ over an alphabet $\Sigma$, and strings $s, s' \in \Sigma^*$,

- a string $r$ **$L$-separates** $s, s'$ if exactly one of $sr$ and $s'r$ is in $L$
- $s \sim^L s'$ iff no string $L$-separates $s, s'$
**L-inseparable histories**

A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)

- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
- transitions \(i \rightarrow j \rightarrow s j\) [remember only last \(n\) bits]
- set \(1(0 + 1)^{n-1}\) of final states

Given a language \(L\) over an alphabet \(\Sigma\), and strings \(s, s' \in \Sigma^*\),

- a string \(r\) is **\(L\)-separates** \(s, s'\) if exactly one of \(sr\) and \(s'r\) is in \(L\)
- \(s \sim^L s'\) iff no string \(L\)-separates \(s, s'\)

**Myhill-Nerode Theorem**

\(L\) is regular iff \(\sim^L\) has finitely many equivalence classes.

\[
 s_L := \{ s' \in \Sigma^* \mid s \sim^L s' \}
\]
**L-inseparable histories**

A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)
- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
- transitions \( s \xrightarrow{j} s_j \) [remember only last \(n\) bits]
- set \(1(0 + 1)^{n-1}\) of final states

Given a language \(L\) over an alphabet \(\Sigma\), and strings \(s, s' \in \Sigma^*\),
- a string \(r\) **\(L\)-separates** \(s, s'\) if exactly one of \(sr\) and \(s'r\) is in \(L\)
- \(s \sim^L s'\) iff no string \(L\)-separates \(s, s'\)

**Myhill-Nerode Theorem**

*\(L\) is regular iff \(\sim^L\) has finitely many equivalence classes.*

*If so, these are the states of a DFA accepting \(L\) with fewest states.*
- states \(s_L := \{s' \in \Sigma^* \mid s \sim^L s'\}\) with \(\epsilon_L\) initial
**L**-inseparable histories

A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)

- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
- transitions \(is \xrightarrow{j} sj\) [remember only last \(n\) bits]
- set \(1(0 + 1)^{n-1}\) of final states

Given a language \(L\) over an alphabet \(\Sigma\), and strings \(s, s' \in \Sigma^*\),

- a string \(r\) \(L\)-separates \(s, s'\) if exactly one of \(sr\) and \(s'r\) is in \(L\)

- \(s \sim^L s'\) iff no string \(L\)-separates \(s, s'\)

**Myhill-Nerode Theorem**

\(L\) is regular iff \(\sim^L\) has finitely many equivalence classes.

If so, these are the states of a DFA accepting \(L\) with fewest states.

- states \(s_L := \{s' \in \Sigma^* \mid s \sim^L s'\}\) with \(\epsilon_L\) initial
- transitions \(s_L \xrightarrow{a} (sa)_L\) for \(a \in \Sigma\)
**L-inseparable histories**

A DFA for \((0 + 1)^*1(0 + 1)^{n-1}\)

- state set \((0 + 1)^n\) with initial state \(q_0 = 0^n\)
- transitions \(s \xrightarrow{j} s_j\) [remember only last \(n\) bits]
- set \(1(0 + 1)^{n-1}\) of final states

Given a language \(L\) over an alphabet \(\Sigma\), and strings \(s, s' \in \Sigma^*\),

- a string \(r\) \(L\)-separates \(s, s'\) if exactly one of \(sr\) and \(s'r\) is in \(L\)
- \(s \sim^L s'\) iff no string \(L\)-separates \(s, s'\)

**Myhill-Nerode Theorem**

\(L\) is regular iff \(\sim^L\) has finitely many equivalence classes.
If so, these are the states of a DFA accepting \(L\) with fewest states.

- states \(s_L := \{s' \in \Sigma^* | s \sim^L s'\}\) with \(\epsilon_L\) initial
- transitions \(s_L \xrightarrow{a} (sa)_L\) for \(a \in \Sigma\)
- \(s_L\) is final iff \(s \in L\)
Exercise

Consider again the regular languages $L_n = (0 + 1)^*1(0 + 1)^{n-1}$.

Define a function $f_n : (0 + 1)^* \rightarrow (0 + 1)^n$ s.t. for $s \in (0 + 1)^*$,

\[ s \sim_{L_n} f_n(s) \]

and for all $s' \in (0 + 1)^*$,

\[ f_n(s) = f_n(s') \iff s \sim_{L_n} s'. \]
Consider again the regular languages $L_n = (0 + 1)^*1(0 + 1)^{n-1}$.

Define a function $f_n : (0 + 1)^* \rightarrow (0 + 1)^n$ s.t. for $s \in (0 + 1)^*$,

$$s \sim^{L_n} f_n(s)$$

and for all $s' \in (0 + 1)^*$,

$$f_n(s) = f_n(s') \iff s \sim^{L_n} s'.$$

Hint: The $f_n(s)$’s are the states of a DFA accepting $L_n$. 
Exercise

Consider again the regular languages $L_n = (0 + 1)^*1(0 + 1)^{n-1}$.

Define a function $f_n : (0 + 1)^* \rightarrow (0 + 1)^n$ s.t. for $s \in (0 + 1)^*$,

$$s \sim^{L_n} f_n(s)$$

and for all $s' \in (0 + 1)^*$,

$$f_n(s) = f_n(s') \iff s \sim^{L_n} s'.$$

Hint: The $f_n(s)$'s are the states of a DFA accepting $L_n$.

How does this DFA compare to the determinization of the $(n + 1)$-state NFA accepting $L_n$ given by the subset construction?
From finite automata to Turing machines

For finite automata, determinism can have exponential cost ($L_n$).
From finite automata to Turing machines

For finite automata, determinism can have exponential cost ($L_n$).

For Turing machines (and time), many suspect this is also the case

$$P \neq NP \quad \text{[i.e., } \bigcup_{k \geq 1} \text{DTIME}(n^k) \neq \bigcup_{k \geq 1} \text{NTIME}(n^k)\text{]}$$

although settling $P=NP$ remains an open problem (the most celebrated in theoretical computer science).
From finite automata to Turing machines

For finite automata, determinism can have exponential cost ($L_n$).

For Turing machines (and time), many suspect this is also the case

$$P \neq NP$$

although settling $P=NP$ remains an open problem (the most celebrated in theoretical computer science).

**Satisfiability (SAT):**
Given a Boolean expression $\varphi$ with variables $X_1, \ldots, X_n$, can we make $\varphi$ true by assigning true/false to $X_1, \ldots, X_n$?
From finite automata to Turing machines

For finite automata, determinism can have exponential cost ($L_n$).

For Turing machines (and time), many suspect this is also the case

$$P \neq NP$$

although settling $P=NP$ remains an open problem (the most celebrated in theoretical computer science).

**Satisfiability** (SAT):
Given a Boolean expression $\varphi$ with variables $X_1, \ldots, X_n$, can we make $\varphi$ true by assigning true/false to $X_1, \ldots, X_n$?

Checking that a particular assignment makes $\varphi$ true is easy ($P$).
From finite automata to Turing machines

For finite automata, determinism can have exponential cost ($L_n$).

For Turing machines (and time), many suspect this is also the case

$$P \neq NP$$

although settling $P=NP$ remains an open problem (the most celebrated in theoretical computer science).

**Satisfiability** (SAT):
Given a Boolean expression $\varphi$ with variables $X_1, \ldots, X_n$, can we make $\varphi$ true by assigning true/false to $X_1, \ldots, X_n$?

Checking that a particular assignment makes $\varphi$ true is easy (P). Non-determinism (guessing the assignment) puts SAT in NP.
From finite automata to Turing machines

For finite automata, determinism can have exponential cost ($L_n$).

For Turing machines (and time), many suspect this is also the case

$$P \neq NP$$

although settling $P=NP$ remains an open problem (the most celebrated in theoretical computer science).

**Satisfiability** (SAT):
Given a Boolean expression $\varphi$ with variables $X_1, \ldots, X_n$, can we make $\varphi$ true by assigning true/false to $X_1, \ldots, X_n$?

Checking that a particular assignment makes $\varphi$ true is easy (P). Non-determinism (guessing the assignment) puts SAT in NP. But is SAT in P? There are $2^n$ assignments to try.