

Computational Models — Lecture 3¹

Handout Mode

Ronitt Rubinfeld and Iftach Haitner.

Tel Aviv University.

March 25/27, 2015

¹Based on frames by Benny Chor, Tel Aviv University, modifying frames by Maurice Herlihy, Brown University.

Computational Models - Lecture 3

- ▶ Midterm exam is reschedule to **May 15**.
- ▶ **Non**-regular languages: two approaches
 1. Pumping Lemma
 2. Myhill-Nerode Theorem (not in Sipser's book)
- ▶ Closure properties
- ▶ Algorithmic questions for NFAs
- ▶ Sipser, [1.4](#), [2.1](#), [2.2](#)
- ▶ Hopcroft and Ullman, [3.4](#)

Proved Last Time

Theorem 1

A *language* is described by a *regular expression*, iff it is regular.

We have made a lot of progress understanding what finite automata **can** do, but what they **cannot** do?

Negative Results

Is there a DFA that accepting the following languages (over $\{0, 1\}$).

- ▶ $\mathcal{B} = \{0^n 1^n : n \geq 0\}$
- ▶ $\mathcal{C} = \{w : \#_1(w) = \#_0(w)\}$
- ▶ $\mathcal{D} = \{w : \#_{01}(w) = \#_{10}(w)\}$

$\#_s(w)$ – the number of times s appears in w .

Consider \mathcal{B} :

- ▶ DFA must “remember” how many 0’s it has seen
- ▶ Impossible with finite state.

The others languages seem to be exactly the same...

Question: Is this a proof?

Answer: No, \mathcal{D} is regular.....

Part I

Pumping Lemma

Regular languages can be pumped

For every regular language \mathcal{L} , there exists $\ell > 0$, the **pumping length**, s.t.:
Every $s \in \mathcal{L}$ longer than ℓ , can be “pumped” into a **longer string in \mathcal{L}** .

This is a powerful technique for showing that a language is **not regular**.

The Pumping Lemma

Lemma 2

For every regular language \mathcal{L} , exists $\ell > 0$ (the *pumping length*) s.t.
Every $s \in \mathcal{L}$ with $|s| \geq \ell$, can be written as $s = xyz$ s.t.

1. $xy^iz \in \mathcal{L}$ for every $i \geq 0$,
2. $|y| > 0$, and
3. $|xy| \leq \ell$.

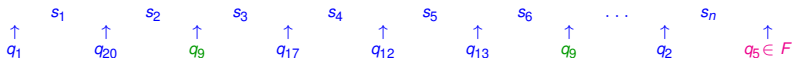
Remarks: Without the second condition, the theorem would be trivial.

The third condition is technical and sometimes useful.

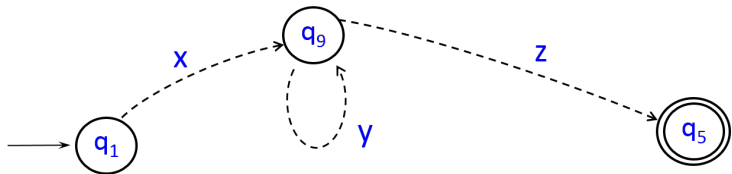
Proving the Pumping Lemma

Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA accepting \mathcal{L} , and let $\ell = |Q|$.

Let $s \in \mathcal{L}$ be with $|s| \geq \ell$, and consider the sequence of states M traverse as it reads $s = s_1 \dots s_n$:



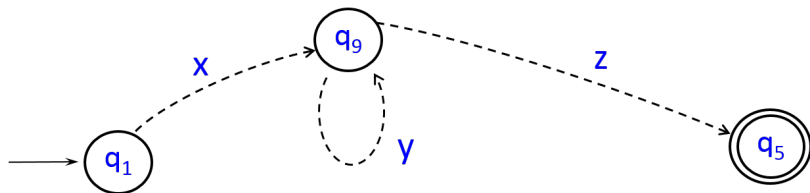
By the pigeonhole principle, at least one of the states in the above sequence repeats.



$$s = xyz$$

Proving the Pumping Lemma, cont.

$$s = xyz$$



- ▶ By inspection, M accepts xy^kz for every $k \geq 0$.
- ▶ $|y| > 0$, because the state q_9 is repeated.
- ▶ To ensure that $|xy| \leq \ell$, pick first state repetition, which must occur no later than $\ell + 1$ states in sequence.

Application # 1

Corollary 3

$\mathcal{B} = \{0^n 1^n : n > 0\}$ is not regular.

Proof: By contradiction. Suppose \mathcal{B} is regular and let ℓ be its pumping length.

- ▶ Consider the string $s = 0^\ell 1^\ell \in \mathcal{B}$.
- ▶ Let x, y, z be (one possible) strings guaranteed by the pumping lemma (i.e., $s = xyz$)
 1. $xy^i z \in \mathcal{B}$ for every $k \geq 0$,
 2. $|y| > 0$, and
 3. $|xy| \leq \ell$.
- ▶ If y is all 0, then xy^2z has too many 0's.
- ▶ If y is all 1, then xy^2z has too many 1's.
- ▶ If y is mixed, then xy^2z is not of right form. □

We did not use the third property.

Application # 2

Corollary 4

$C = \{w : \#_1(w) = \#_0(w)\}$ is not regular.

Proof: By contradiction. Suppose C is regular. Let ℓ be the pumping length.

- ▶ Consider the string $s = 0^\ell 1^\ell \in C$.
- ▶ Let x, y, z be (possible) set of strings guaranteed by the pumping lemma (i.e., $s = xyz$)
 1. $xy^i z \in C$ for every $k \geq 0$,
 2. $|y| > 0$, and
 3. $|xy| \leq \ell$.
- ▶ Since $|xy| \leq \ell$, the string y is all 0's.
- ▶ Thus, $xy^2 z \notin C$ (more 0's than 1's). □

Could we have used $s = (01)^\ell$?

Application # 3

Corollary 5

$\mathcal{E} = \{0^i 1^j : i > j\}$ is not regular.

Proof: By contradiction. Suppose \mathcal{E} is regular. Let ℓ be its pumping length.

- ▶ Consider the string $s = 0^\ell 1^{\ell-1} \in \mathcal{E}$.
- ▶ By pumping lemma, $s = xyz$, where $xy^kz \in \mathcal{E}$ for every $k \geq 0$, $|y| > 0$ and $|xy| \leq \ell$.
- ▶ But $xy^0z = xz \notin \mathcal{E}$ (at least as much 1's as 0's) □

Application # 4

Corollary 6

The language $Primes \subset 1^*$ – all strings whose length is a prime number – is not regular.

Proof: Suppose $Primes$ is regular, and let ℓ be its pumping length.

- ▶ Let $s = 1^p \in Primes$, where $p \geq \ell$ is a prime (?)
- ▶ By pumping lemma, $s = xyz$, where $xy^kz \in Primes$ for every $k \geq 0$.
- ▶ Let $|y| = m$. Hence, $xy^{p+1}z = 1^{p+mp} \in Primes$
but $p(m+1)$ is not prime... □

Another Example

Consider the language $\mathcal{L} = \{a^i b^n c^n : n \geq 0, i \geq 1\} \cup \{b^n c^m : n, m \geq 0\}$.

Any non-empty $s \in \mathcal{L}$ can be pumped:

- ▶ If $s = a^i b^n c^n$, then set $x = \varepsilon$ and $y = a$.
- ▶ If $s = b^n c^m$, then set $x = \varepsilon$ and $y = b$.
- ▶ If $s = c^m$, then set $x = \varepsilon$ and $y = c$.

(in all cases z is set arbitrarily).

- ▶ Is \mathcal{L} regular? **No**
- ▶ How can we prove it?

Part II

Characterization of Regular Languages

The equivalence relation $\sim_{\mathcal{L}}$

Definition 7

For $\mathcal{L} \subseteq \Sigma^*$, define the equivalence relation $\sim_{\mathcal{L}}$ over words in Σ^* , by $x \sim_{\mathcal{L}} y$ if for every $z \in \Sigma^*$, it holds that $xz \in \mathcal{L} \iff yz \in \mathcal{L}$.

Easy to see that $\sim_{\mathcal{L}}$ is indeed an equivalence relation (reflexive, symmetric, transitive) on Σ^* .

Hence, $\sim_{\mathcal{L}}$ partitions Σ^* into equivalence classes.

For $x \in \Sigma^*$, let $[x] \subseteq \Sigma^*$ denote its equivalence class with respect to $\sim_{\mathcal{L}}$

How many equivalence classes does $\sim_{\mathcal{L}}$ induce? finite or infinite?

Could be either (depends on \mathcal{L}).

Fact 8 (right invariance)

If $x \sim_{\mathcal{L}} y$, then $xw \sim_{\mathcal{L}} yw$ for every $w \in \Sigma^*$

Proof: $(xw)z \in \mathcal{L} \iff x(wz) \in \mathcal{L} \iff y(wz) \in \mathcal{L} \iff (yw)z \in \mathcal{L}$ □

Three examples

▶ $\mathcal{L}_1 = \{w : \#_1(w) \bmod 4 = 0\}$

$\approx^{\mathcal{L}_1}$ has **finitely many** equivalence classes.

The equivalent classes are: $[1]$, $[11]$, $[111]$, $[1111]$

▶ $\mathcal{L}_2 = \{0^n 1^n : n \in \mathbb{N}\}$

$\approx^{\mathcal{L}_2}$ has **infinitely many** equivalence classes.

$$[0] \neq [0^2] \neq [0^3] \dots$$

▶ $\mathcal{L}_3 = \{a^i b^n c^n : n \geq 0, i \geq 1\} \cup \{b^n c^m : n, m \geq 0\}$

$\approx^{\mathcal{L}_3}$ has **infinitely many** equivalence classes.

$$[ab] \neq [ab^2] \neq [ab^3] \neq \dots$$

The above statements required a proof...

Myhill-Nerode Theorem

Theorem 9 (Myhill-Nerode Theorem)


$\mathcal{L} \subseteq \Sigma^*$ is *regular* iff $\sim_{\mathcal{L}}$ finitely many equivalence classes.

Hence

- ▶ $\mathcal{L}_1 = \{w \in \{0, 1\}^* : \#_1(w) \bmod 4 = 0\}$ is regular.
- ▶ $\mathcal{L}_2 = \{0^n 1^n : n \in \mathbb{N}\}$ is *not* regular.
- ▶ $\mathcal{L}_3 = \{a^i b^n c^n : n \geq 0, i \geq 1\} \cup \{b^n c^m : n, m \geq 0\}$ is *not* regular.

Proving Myhill-Nerode Theorem \implies

Let \mathcal{L} be a regular language and let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA accepting it.

- ▶ Define the binary relation $\overset{M}{\sim}$ by $x \overset{M}{\sim} y$ if $\widehat{\delta}(q_0, x) = \widehat{\delta}(q_0, y)$.
- ▶ $\overset{M}{\sim}$ is an equivalence relation.
- ▶ $x \overset{M}{\sim} y \implies xz \overset{M}{\sim} yz$ for every $z \in \Sigma^*$.
 $\implies xz \in \mathcal{L}$ iff $yz \in \mathcal{L}$.
- ▶ Hence, $x \overset{M}{\sim} y \implies x \overset{\mathcal{L}}{\sim} y$.
- ▶ Each equivalence class of $\overset{\mathcal{L}}{\sim}$ corresponds to **union** of classes of $\overset{M}{\sim}$.
Namely, $\overset{M}{\sim}$ is a **refinement** of $\overset{\mathcal{L}}{\sim}$. (see drawing on board)
- ▶ Specifically, $\#$ of equivalence classes of $\overset{\mathcal{L}}{\sim}$ is **less or equal than** $\#$ of equivalence classes of $\overset{M}{\sim}$.
- ▶ $\overset{M}{\sim}$ has **finitely many** equivalence classes. (?)
- ▶ Therefore, $\overset{\mathcal{L}}{\sim}$ has finitely many equivalence classes. 

Proving Myhill-Nerode Theorem \Leftarrow

Assume \mathcal{L} has **finitely many** equivalence classes and let $x_1, \dots, x_n \in \Sigma^*$ be their representatives.

We'll construct a DFA $M = (Q, \Sigma, \delta, q_0, F)$ that accepts \mathcal{L} .

For $x \in \Sigma^*$, let $C(x)$ be the index $i \in \{1, \dots, n\}$ with $x \in [x_i]$.

- ▶ $Q = \{1, \dots, n\}$.
- ▶ $\delta(i, \sigma) = C(x_i \sigma)$.
- ▶ $q_0 = C(\varepsilon)$.
- ▶ $F = \{i: x_i \in \mathcal{L}\}$.

Claim. Let $x \in [x_i]$, then $\widehat{\delta}(q_0, x) = i$.

Proof: By induction on word length.

- ▶ Let $y = w\sigma \in \Sigma^*$, and assume $w \in [x_i]$ and $y \in [x_j]$.
- ▶ $\delta(i, \sigma) := C(x_i \sigma) =$ (by right invariance) $C(w\sigma) = C(y) = j$.

$\implies \widehat{\delta}(q_0, w\sigma) := \delta(\widehat{\delta}(q_0, w), \sigma) =$ (by i.h) $\delta(i, \sigma) = j$.

Therefore, $y \in \mathcal{L}(M) \iff C(y) \in F \iff [y] \subseteq \mathcal{L} \iff y \in \mathcal{L}$. □

This is **the** optimal DFA, number of states wise, for \mathcal{L} .(?)

Example

Construct a DFA for $\{w \in \{0, 1\}^* : \#_1(w) \equiv 0 \pmod{5}\}$, via the latter proof method.

- ▶ Equivalent class representatives: $\{\varepsilon, 1, 11, 111, 1111\}$
- ▶ $Q = \{0, 1, 2, 3, 4\}$
- ▶ $q_0 = 0$
- ▶ $F = \{0\}$
- ▶ $\delta(i, 0) = i$ and $\delta(i, 1) = i + 1 \pmod{5}$

Part III

Closure Properties of Regular Languages

Simple closure properties

- ▶ Regular languages are closed under complement.
 1. Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA that accepts \mathcal{L} .
 2. Then $M' = (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a DFA that accepts $\overline{\mathcal{L}} = \Sigma^* \setminus \mathcal{L}$.
 3. NFA ?!
- ▶ Regular languages are closed under intersection.
 1. $\mathcal{L}_1 \cap \mathcal{L}_2 = \overline{\overline{\mathcal{L}_1} \cup \overline{\mathcal{L}_2}}$.
 2. Proof with automata ?

Division

For languages $\mathcal{L}_1, \mathcal{L}_2 \in \Sigma^*$, define

$$\mathcal{L}_1/\mathcal{L}_2 = \{x \in \Sigma^* : \exists y \in \mathcal{L}_2, xy \in \mathcal{L}_1\}$$

Examples:

- ▶ $\mathcal{L}_1 = (01 \cup 1)^*$ and $\mathcal{L}_2 = 00$. Then $\mathcal{L}_1/\mathcal{L}_2 = \emptyset$
- ▶ $\mathcal{L}_3 = a^*b^*c^*$ and $\mathcal{L}_4 = b$. Then $\mathcal{L}_3/\mathcal{L}_4 = a^*b^*$

Closure under division

Recall, $\mathcal{L}_1/\mathcal{L}_2 = \{x : \exists y \in \mathcal{L}_2, xy \in \mathcal{L}_1\}$

Theorem 10

Regular languages are closed under division with **any** language (i.e., \mathcal{L}_1 is regular $\implies \mathcal{L}_1/\mathcal{L}_2$ is regular).

Proof: Let \mathcal{L}_1 be a regular language and let \mathcal{L}_2 be an arbitrary language.

- ▶ $\sim^{\mathcal{L}_1}$ is a refinement of $\sim^{\mathcal{L}_1/\mathcal{L}_2}$. Proof: Assume $x \sim^{\mathcal{L}_1} y$. For $z \in \Sigma^*$:
$$xz \in \mathcal{L}_1/\mathcal{L}_2 \leftrightarrow xzW \in \mathcal{L}_1 \leftrightarrow yzW \in \mathcal{L}_1 \leftrightarrow yz \in \mathcal{L}_1/\mathcal{L}_2. \implies x \sim^{\mathcal{L}_1/\mathcal{L}_2} y. \quad \square$$
- ▶ Hence, $\mathcal{L}_1/\mathcal{L}_2$ has finite number of equivalent states, and thus regular.
- ▶ Another proof.
- ▶ Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA for \mathcal{L}_1 .
- ▶ Let $F' = \{q \in Q : \exists y \in \mathcal{L}_2, \delta(q, y) \in F\}$
- ▶ The DFA $M' = (Q, \Sigma, \delta, q_0, F')$ accepts $\mathcal{L}_1/\mathcal{L}_2$. □

F' is well defined, but might be hard to compute – “non constructive proof”.

Homomorphism

Definition 11 (Homomorphism)

An **homomorphism** from alphabet Δ to **words** over alphabet Σ , is a function $h: \Delta \mapsto \Sigma^*$.

For $w \in \Delta^*$, let $h(w = w_1, \dots, w_n) = h(w_1) \cdots h(w_n)$.

For $\mathcal{L} \subseteq \Delta^*$, let $h(\mathcal{L}) = \{h(w) : w \in \mathcal{L}\}$.

Examples:

- ▶ Let $h: \{0, 1\} \mapsto \{a, b\}^*$ be defined by $h(1) = aba$ and $h(0) = aa$.
 $h(010) = aaaba$. For $\mathcal{L}_1 = (01)^*$, $h(\mathcal{L}_1) = (aaaba)^*$.
- ▶ Let $h(0) = a$, $h(1) = a$. For $\mathcal{L}_2 = \{0^n 1^n : n \geq 0\}$, $h(\mathcal{L}_2) = \{a^{2n} : n \geq 0\}$.

Theorem 12

Regular languages are closed under homomorphism.

Proof: two options:

- ▶ Using regular expressions
- ▶ Using Automata (board)

Inverse homomorphism

Definition 13 (Inverse homomorphism)

For homomorphism $h: \Delta \mapsto \Sigma^*$, define its **inverse homomorphism** $h^{-1}: \Sigma^* \mapsto P(\Delta^*)$, by $h^{-1}(w) = \{x \in \Delta^* : h(x) = w\}$.

For $\mathcal{L} \subseteq \Sigma^*$, let $h^{-1}(\mathcal{L}) = \bigcup_{x \in \mathcal{L}} h^{-1}(x) = \{x \in \Delta^* : h(x) \in \mathcal{L}\}$

Example: $h(1) = aba$, $h(0) = aa$ and $\mathcal{L}_2 = (ab \cup ba)^* a$.

Then $h^{-1}(\mathcal{L}_2) = \{1\}$. (\mathcal{L}_2 has no words starting with $h(0)$ or $h(1\sigma)$).

Claim 14

For any $h: \Delta \mapsto \Sigma^*$:

1. $h(h^{-1}(\mathcal{L})) \subseteq \mathcal{L}$, for any $\mathcal{L} \subseteq \Sigma^*$
2. $\mathcal{L} \subseteq h^{-1}(h(\mathcal{L}))$, for any $\mathcal{L} \subseteq \Delta^*$

Proof:

1. Immediate
2. Holds since $w \in h^{-1}(h(w))$ for any $w \in \Delta^*$

Closure under inverse homomorphism

Theorem 15

Regular languages are closed under inverse homomorphism.

Proof idea: Let \mathcal{L} be a regular language, let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA for \mathcal{L} and let $h: \Delta \mapsto \Sigma^*$.

- ▶ For each $a \in \Delta$, we advance in M using $h(a)$.
- ▶ Formally, we define $M' = (Q, \Delta, \delta', q_0, F)$, where $\delta'(q, a) = \widehat{\delta}(q, h(a))$.
- ▶ Hence, $\widehat{\delta}'(q, w) = \widehat{\delta}(q, h(w))$
- ▶ $w \in \mathcal{L}(M') \iff h(w) \in \mathcal{L}(M)$ □

Part IV

Algorithmic Questions for NFAs

Algorithmic Questions for NFAs

Q: Given an NFA, N , and a string w , is $w \in \mathcal{L}(N)$?

Answer: Construct the DFA equivalent to N and run it on w .

Q: Is $\mathcal{L}(N) = \emptyset$?

Answer: This is a **reachability** question in graphs: Is there a path in the states' graph of N from the start state to some accepting state?

There are simple, efficient algorithms for this task.

More Questions

Q.: Is $\mathcal{L}(N) = \Sigma^*$?

Answer: Check if $\overline{\mathcal{L}(N)} = \emptyset$.

Q.: Given N_1 and N_2 , is $\mathcal{L}(N_1) \subseteq \mathcal{L}(N_2)$?

Answer: Check if $\overline{\mathcal{L}(N_2)} \cap \mathcal{L}(N_1) = \emptyset$.

Q.: Given N_1 and N_2 , is $\mathcal{L}(N_1) = \mathcal{L}(N_2)$?

Answer: Check if $\mathcal{L}(N_1) \subseteq \mathcal{L}(N_2)$ and $\mathcal{L}(N_2) \subseteq \mathcal{L}(N_1)$.

In the future, we will see that for **stronger models** of computations, many of these problems **cannot be solved** by any algorithm.

Part V

Summary — Regular Languages

Summary - Regular Languages

So far we saw

- ▶ Finite automata,
- ▶ Regular languages,
- ▶ Regular expressions,
- ▶ Myhill-Nerode theorem and pumping lemma for **regular languages**.

Next class we introduce stronger machines and languages with more expressive power:

- ▶ pushdown automata,
- ▶ context-free languages,
- ▶ context-free grammars

Part VII

Appendix Not Taught in Class

Section 1

Finding the Minimal Automata

Finding the minimal automata

Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$, find the **minimal** (with respect to # of states) DFA M' with $\mathcal{L}(M') = \mathcal{L}(M)$.

Definition 16 (Equivalent states)

For DFA $(Q, \Sigma, \delta, q_1, F)$, states $q_1, q_2 \in Q$ are **equivalent**, if for all $x_1, x_2 \in \Sigma^*$ with $\widehat{\delta}(q_0, x_i) = q_i$, it holds that $x_1 \stackrel{\mathcal{L}}{\sim} x_2$.

Idea: keep **merging** equivalent states in Q , until all states are **non-equivalent**.

Actual idea:

1. Start with the two sets F and $Q \setminus F$.
2. Keep *splitting* the sets until **all** states in the same set are **equivalent**.

To check whether states q and q' are **equivalent**, check if $\delta(q, \sigma)$ and $\delta(q', \sigma)$ are **in the same set**, for all $\sigma \in \Sigma$. (?)

3. Merge all states in the same set.

Is this a minimal automate? Yes, if M has no **unreachable** states.

But we can assume that wlg. (?)

Finding the minimal automata

Algorithm 17

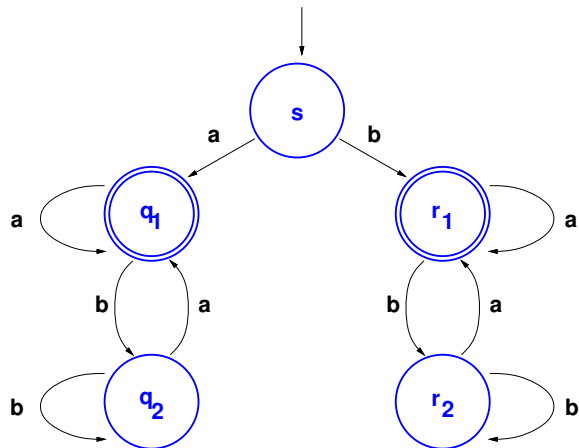
Input: DFA $M = (Q, \Sigma, \delta, q_0, F)$

1. Let $\mathcal{T} = \{F, Q \setminus F\}$.
2. While $\exists \mathcal{S} \in \mathcal{T}$, $q_1, q_2 \in \mathcal{S}$ and $\sigma \in \Sigma^*$ s.t.,
 $\delta(q_1, \sigma) \in \mathcal{S}'$ and $\delta(q_2, \sigma) \notin \mathcal{S}'$ for some $\mathcal{S}' \in \mathcal{T}$:
 - 2.1 Let $\mathcal{S}_{sp} = \{q \in \mathcal{S} : \delta(q, \sigma) \in \mathcal{S}'\}$.
 - 2.2 Set $\mathcal{T} = \mathcal{T} \cup \mathcal{S}_{sp} \cup (\mathcal{S} \setminus \mathcal{S}_{sp}) \setminus \mathcal{S}$.
3. Output DFA $M' = (Q', \delta', q'_0, F')$, where
 - ▶ $Q' = \mathcal{T}$
 - ▶ $q'_0 = \mathcal{S}_0 \in \mathcal{T}$, where $q_0 \in \mathcal{S}_0$.
 - ▶ $F' = \{\mathcal{S} \in \mathcal{T} : \mathcal{S} \subseteq F\}$
 - ▶ $\delta'(\mathcal{S}, \sigma) = \mathcal{S}' \in \mathcal{T}$, s.t. $\delta(q, \sigma) \in \mathcal{S}'$ for (every) $q \in \mathcal{S}$.

Claim 18

The above algorithm outputs **the** minimal automata for $\mathcal{L}(M)$.

Example



On board...

Section 2

Using Homomorphism

Using homomorphism

We know that $\mathcal{L}_1 = \{0^n 1^n : n \geq 1\}$ is **not** regular, show that $\mathcal{L}_2 = \{a^n b a^n : n \geq 1\}$ is **not** regular.

We will prove using homomorphism and inverse homomorphism. Let

$$\triangleright h_1(a) = a, h_1(b) = b, h_1(c) = a. \quad (h_1 : \{a, b, c\} \mapsto \{a, b, c\}^*)$$

$$\triangleright h_2(a) = 0, h_2(b) = \epsilon, h_2(c) = 1. \quad (h_2 : \{a, b, c\} \mapsto \{0, 1\}^*)$$

We prove $h_2(h_1^{-1}(\mathcal{L}_2) \cap a^* b^* c^*) = \mathcal{L}_1$. Thus, \mathcal{L}_2 is **not** regular (?)

$$\triangleright h_1^{-1}(\mathcal{L}_2) = (a \cup c)^n b (a \cup c)^n$$

$$\triangleright h_1^{-1}(\mathcal{L}_2) \cap a^* b^* c^* = \{a^n b c^n : n \geq 1\}$$

$$\triangleright h_2(h_1^{-1}(\mathcal{L}_2) \cap a^* b^* c^*) = \{0^n 1^n : n \geq 1\}$$