Announcements Mar 23

- Cameras on
- HW due Thu 3:30
- Office Hours Fri 2-3, appt
  + makeup 10-11 Wed
- Progress report Apr 2 ~ 1 page
- First draft Apr 9
- Talk to me about makeup points!

<u>Today</u>

- Howson's thm
- Regular languages
- Automata

Howson's Thm

Thm 7.32 (1954)

If G, H f.g. subgps of F_n then $G \cap H$ is f.g.

A "counterexample" with F_n replaced by another group:

Take $F_2 \times \mathbb{Z}$ $\quad F_2 = \langle x, y \rangle \quad \mathbb{Z} = \langle z \rangle$

$G = F_2$ (first factor)

$H = \ker(F_2 \times \mathbb{Z} \to \mathbb{Z})$

all 3 gens $\longmapsto 1$

To check: ① G fg. ✓

② H fg.

③ $G \cap H$ not fg.

② Claim: H is gen by $S = \{xz^{-1}, yz^{-1}\}, \cancel{xy^{-1}}$

Step 1. $\langle S \rangle$ normal.

To show: $gsg^{-1} \in \langle S \rangle$

$g = (\text{gen for } F_2 \times \mathbb{Z})^{\pm 1} \quad s \in S.$

example. $y(x^{-1}z)y^{-1} = (yz^{-1})(x^{-1}z)(y^{-1}z)$

Step 2. $\langle S \rangle \subseteq H$ ✓

Step 3. $(F_2 \times \mathbb{Z}) / \langle S \rangle \cong \mathbb{Z}$

We get $F_2 \times \mathbb{Z}$ subject to $x = z, y = z$ ✓

$F_2 \times \mathbb{Z}$ $\quad F_2 = \langle x, y \rangle$ $\mathbb{Z} = \langle z \rangle$

$G = F_2$ (first factor)

$H = \ker(F_2 \times \mathbb{Z} \to \mathbb{Z})$

all 3 gens $\longmapsto 1$

Remains.

③ $G \cap H$ not fg.

$G \cap H$ is the subgp of F_2 :

$\ker F_2 \to \mathbb{Z}$

$x, y \longmapsto 1$

(exponent sum 0).

<u>Claim</u>. $G \cap H$ is freely gen by

$$x^i y^{-i}$$

example. $x^5 y^{-5} x^3 y^{-3} y^2 x^{-2}$

[Very similar to HW problem:

$\ker F_2 \to \mathbb{Z}^2$

$x \longmapsto (1,0)$

$y \longmapsto (0,1)$

Freely gen by

$\{x^i y^j x^{-i} y^{-j}\}$]

$\square$

Hanna Neumann Conjecture (1957)

$$\mathrm{rk}(G \cap H) - 1 \leq (\mathrm{rk}(G) - 1)(\mathrm{rk}(H) - 1)$$

for $G, H \leq F_n$

Proved in 2011 by Friedman, Mineyev.

Our proof of Howson's thm

uses regular languages, automata.

Today: automaton version of Howson's thm. Thu: Howson's thm.

Languages

$S = \{x_1, \dots, x_n\}$ "alphabet"

$S^* = \{\text{words of finite length in } S\}$

Any subset $L \subseteq S^*$ is called a language

Examples

① $S = \{a, \dots, z\}$ $L = \{\text{words in OED}\}$

② $S = \{a\}$ $L = \{a^n : 3 \mid n\}$

③ $S = \{a, b, c\}$ $L = \{a^i b^j c^k : i > 0, j \geq 0, k \geq 0\}$

④ $S = \{\text{gen set for } G\}^{\pm 1}$ L = words in S that equal id in G.

Automata (= simple computer)

S = alphabet (finite set)

An automaton M over S consists of a directed graph with decorations:

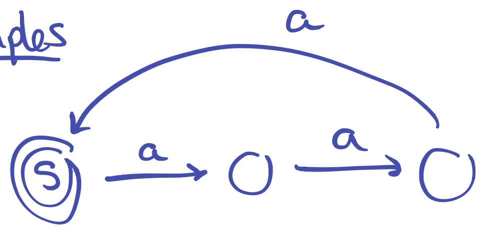
vertices: "states"

- some subset of vertices called start states Ⓢ
- some subset A of vertices called accept states ◎
- edges labeled by elts of S.

If the graph is finite, M is a finite state automaton.

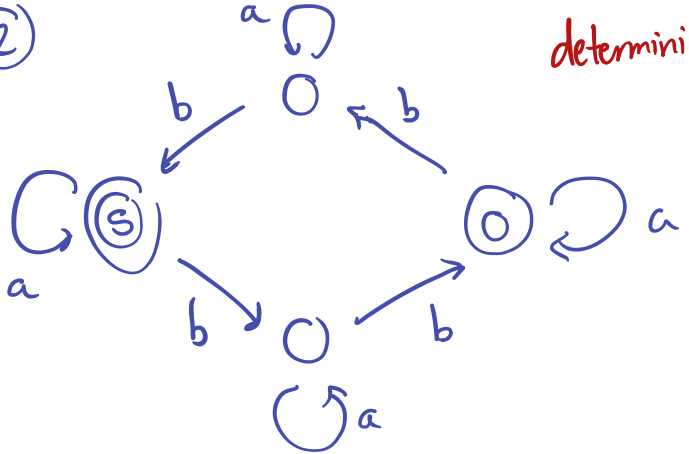
The language accepted by M is $\{w \in S^* : w \text{ given by a directed path in } M\}$

Examples

① 

$\rightsquigarrow L = \{a^i : 3 \mid i\}$

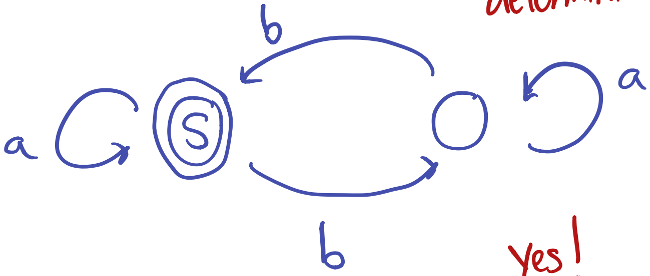
2



deterministic!

Poll: Is there a simpler automaton for Same language?

deterministic!



yes!

$L = \{$ words with b-exponent sum even $\}$

$a^3 b^5 a b$ ✓

Deterministic automata

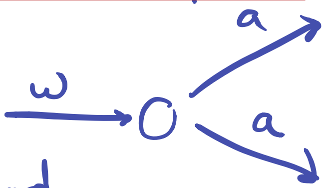
A det. aut. is a FSA with

- exactly one start state
- no two edges leaving same vertex have same label
- no edges with empty label (in Meier: empty = ε)

It is complete if each vertex has departing edges with all possible labels.

What's deterministic about it?

$\boxed{\text{words} \longleftrightarrow \text{paths}}$



The word wa corresponds to more than 1 path

To see if a word is in the accepted language, start at the start state, trace out the word/path, see if land at accept state.

A language is regular if accepted by a det. FSA.

Automaton version of Howson's Thm

<u>Thm 7.11</u> Say $K, L \subseteq S^*$ are reg. languages. Then so are:

1. $S^* \setminus K$
2. $K \cup L$
3. ** $K \cap L$
4. $KL = \{w_K w_L : w_K \in K, w_L \in L\}$
5. $L^* = L \cup LL \cup LLL \cup \cdots$

** reg. lang. is automaton version of f.g.

<u>Lemma 1</u>. L accepted by a det. FSA (i.e. L is regular) $\Longrightarrow$ L accepted by a complete det FSA.

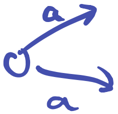
<u>Pf</u>. (exercise: add dead ends/fail states)

<u>Lemma 2</u>. L accepted by a non-det. FSA $\Longrightarrow$ L accepted by a det. FSA.

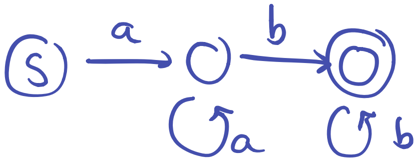
In other words: starting with a non-det FSA, Lemma 2 converts it to a det FSA, Lemma 1 converts to a complete det FSA.

Lemma 2. L accepted by a non-det. FSA $\Rightarrow$ L accepted by a det. FSA.

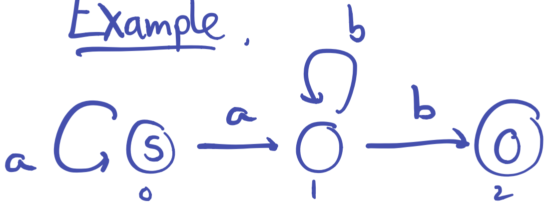
Pf. Two steps:

1. Get rid of arrows with empty labels
2. Get rid of 

3. Get rid of multiple start states.



Example.

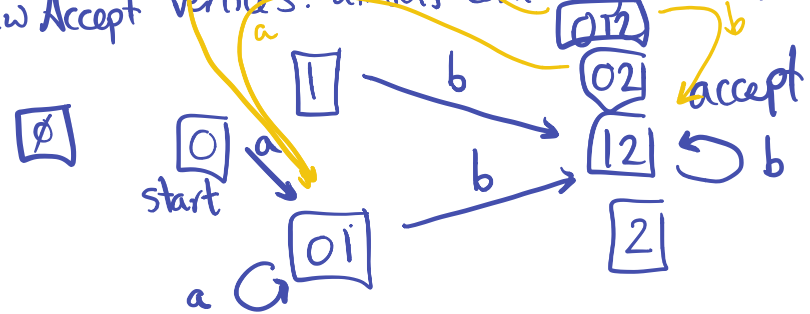


$$L = \{a^i b^j : i,j > 0\}$$

New vertices: subsets of old vertices

New Start vertex: set of all old start vertices.

New Accept vertices: all sets containing an accept



We can now convert $\mathcal{M}_\varepsilon$ into a deterministic automaton, $\mathcal{D}$. The states of $\mathcal{D}$ consist of all the subsets of $V(\mathcal{M}_\varepsilon)$. The single start state of $\mathcal{D}$ is the subset of $V(\mathcal{M}_\varepsilon)$ consisting of all the start states of $\mathcal{M}_\varepsilon$. The accept states of $\mathcal{D}$ are the subsets of $V(\mathcal{M}_\varepsilon)$ that contain at least one accept state of $\mathcal{M}_\varepsilon$. In $\mathcal{D}$ there is an edge from U to U' labelled by x if, for each $v \in U$, there is an edge labelled x from v to some $v' \in U'$, and U' is entirely composed of such vertices. That is, there is an edge labelled x from U to the vertex corresponding to the set

$$U' = \{v' \in V(\mathcal{M}_\varepsilon) \mid v' \text{ is at the end of an edge labelled } x \text{ that begins at some } v \in U\}.$$