Algebraic Approach to Automata Theory

Kamal Lodaya, adding to slides by Deepak D'Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

September 2021

Outline

Algebraic approach to automata: Overview

An algebra is a set with some operations on it, satisfying some properties.
E.g. a group is a nonempty set with a unit 1, a binary operation satisfying associativity $(x \circ (y \circ z) = (x \circ y) \circ z)$ and a unary operation satisfying an inverse property $(x \circ inv(x) = 1)$

- Define language recognition via morphisms into a monoid
- Analogous result to canonical automaton (Myhill-Nerode)
- Helps in characterising class of FO-definable languages
- Helps in automaton constructions

Monoids

- A monoid is an algebra $(M, \circ, 1)$, where
  - M is a set containing the element 1,
  - $\circ$ is an associative binary operation on M, and
  - 1 is the unit element (sometimes called identity) with respect to $\circ$, so for every $m \in M$, $m \circ 1 = m = 1 \circ m$
  - Sometimes a monoid may have a zero element 0 (if so, unique), such that for every $m \in M$, $m \circ 0 = 0 = 0 \circ m$
- Examples: $(\mathbb{N}, \times)$, $(\mathbb{N}^{\geq 1}, \times)$, $(\mathbb{N}, +)$, $(A^*, \cdot)$. First three are commutative monoids (operation is commutative), first one has a zero, others do not
- Nonexample: $(\mathbb{N}^{\geq 1}, +) = \{1, 2, 3, \dots\}$ with addition (why?)

Monoids

- A monoid is an algebra $(M, \circ, 1)$, where
  - M is a set containing the element 1,
  - $\circ$ is an associative binary operation on M, and
  - 1 is the unit element (sometimes called identity) with respect to $\circ$, so for every $m \in M$, $m \circ 1 = m = 1 \circ m$
  - Sometimes a monoid may have a zero element 0 (if so, unique), such that for every $m \in M$, $m \circ 0 = 0 = 0 \circ m$
- Examples: $(\mathbb{N}, \times)$, $(\mathbb{N}^{\geq 1}, \times)$, $(\mathbb{N}, +)$, $(A^*, \cdot)$. First three are commutative monoids (operation is commutative), first one has a zero, others do not
- Nonexample: $(\mathbb{N}^{\geq 1}, +) = \{1, 2, 3, \dots\}$ with addition (why?)
- More noncommutative examples: $Fun(X) = (X \to X, \circ, id)$, $Rel(X) = (2^{X \times X}, \circ, id)$ where
  - $X \to X$ is the set of all functions from a set X to itself
  - $2^{X \times X}$ is the set of subsets of $X \times X$, that is, all relations on X
  - $f \circ g$ is function composition, $R \circ S$ is relation composition, both are not commutative

Monoid morphisms

- A morphism from a monoid $(M, \circ_M, 1_M)$ to a monoid $(N, \circ_N, 1_N)$ is a map $h : M \to N$, satisfying
  - $h(1_M) = 1_N$, and,
  - $h(m \circ_M m') = h(m) \circ_N h(m')$.
- Example: $h : A^* \to \mathbb{N}$, given by

$$h(w) = |w|$$

is a morphism from $(A^*, \cdot, \epsilon)$ to $(\mathbb{N}, +, 0)$.

Language recognition via monoid morphisms

- A language $L \subseteq A^*$ is said to be recognized by monoid $(M, \circ, 1)$ if there is a morphism h from $(A^*, \cdot, \epsilon)$ to $(M, \circ, 1)$, and a recognizing subset X of M such that

$$L = h^{-1}(X)$$

  Exercise: Show that $L = h^{-1}(h(L))$.
- If L is recognized by some finite M, we say that L is recognizable
- Finite monoids are finitely generated, that is, there is a finite set of elements (which we can take to be the alphabet) such that every element is a multiplication described by words over the alphabet

Example of language recognition via monoid morphisms

Monoid $M = (\{1, m\}, \circ, 1)$ generated by m
Multiplication table shown

$\circ$	1	m
1	1	m
m	m	m

- Consider $h : A^* \to M$ given by $h(w) = m$ iff $w \neq \epsilon$ (why morphism?)
- Then M recognizes A^+, since $h^{-1}(\{m\}) = A^+$.
- M also recognizes $\{\epsilon\}$ (taking $X = \{1\}$), A^* (taking $X = \{1, m\}$), and $\emptyset$ (taking $X = \{\}$).

Exercise

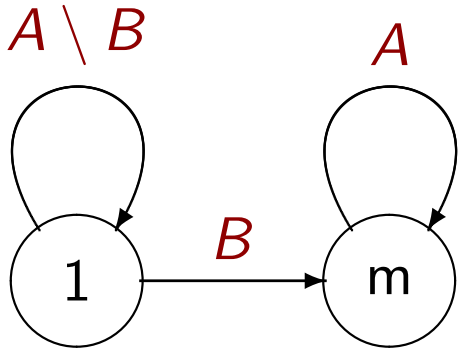
Show that the language of words with an odd number of a's over the alphabet $A = \{a, b\}$ is recognizable.

Finitely generated monoid to deterministic automaton

M generated by m

$\circ$	1	m
1	1	m
m	m	m

Let $B = h^{-1}(m) \cap A$, $h(A \setminus B) = 1$



- Let $L \subseteq A^*$ a language recognized by a monoid $(M, \circ, 1)$.
- That is, a morphism $h : A^* \to M$ and $X \subseteq M$ and the morphism maps subset L to subset X, $L = h^{-1}(X)$.
- Define a DA $\mathcal{A}(M) = (M, \delta, 1, X)$, where

$$\delta(m, a) = m \circ h(a).$$

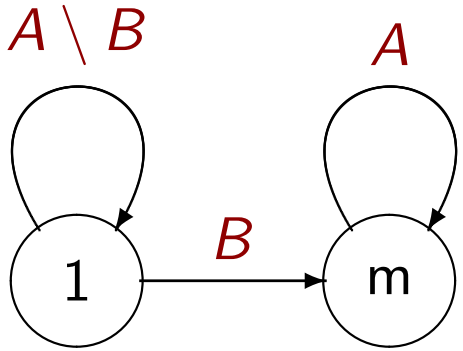
- To prove that: $\mathcal{A}(M)$ accepts L.

Finitely generated monoid to deterministic automaton

M generated by m

$\circ$	1	m
1	1	m
m	m	m

Let $B = h^{-1}(m) \cap A$, $h(A \setminus B) = 1$



- Let $L \subseteq A^*$ a language recognized by a monoid $(M, \circ, 1)$.
- That is, a morphism $h : A^* \to M$ and $X \subseteq M$ and the morphism maps subset L to subset X, $L = h^{-1}(X)$.
- Define a DA $\mathcal{A}(M) = (M, \delta, 1, X)$, where

$$\delta(m, a) = m \circ h(a).$$

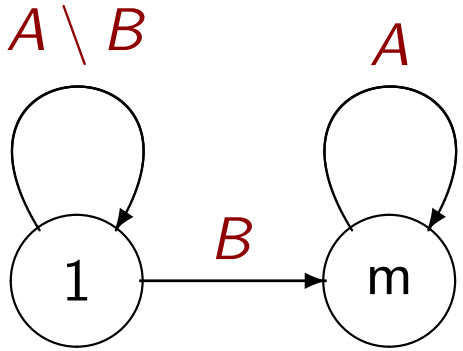
- To prove that: $\mathcal{A}(M)$ accepts L.
- Lemma: $\widehat{\delta}(1, w) = h(w)$. Proof by induction on $|w|$.

Finitely generated monoid to deterministic automaton

M generated by m

$\circ$	1	m
1	1	m
m	m	m

Let $B = h^{-1}(m) \cap A$, $h(A \setminus B) = 1$



- Let $L \subseteq A^*$ a language recognized by a monoid $(M, \circ, 1)$.
- That is, a morphism $h : A^* \to M$ and $X \subseteq M$ and the morphism maps subset L to subset X, $L = h^{-1}(X)$.
- Define a DA $\mathcal{A}(M) = (M, \delta, 1, X)$, where

$$\delta(m, a) = m \circ h(a).$$

- To prove that: $\mathcal{A}(M)$ accepts L.
- Lemma: $\widehat{\delta}(1, w) = h(w)$. Proof by induction on $|w|$.
- Main proof: $\mathcal{A}(M)$ accepts exactly the w such that $h(w) \in X$. So $\mathcal{A}(M)$ accepts $h^{-1}(X) = L$.

Transition Monoid of a DA

Let $\mathcal{A} = (Q, \delta, s)$ be a deterministic transition system.

- For $w \in A^*$, define $f_w : Q \to Q$ by

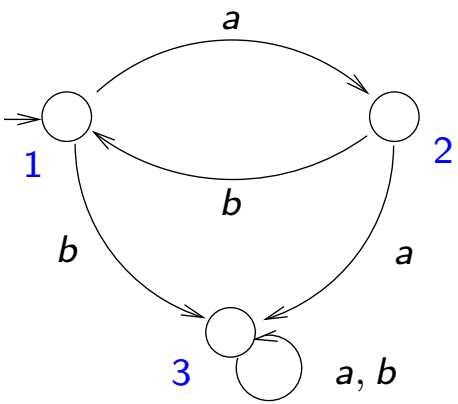
$$f_w(q) = \widehat{\delta}(q, w).$$

- Consider the monoid

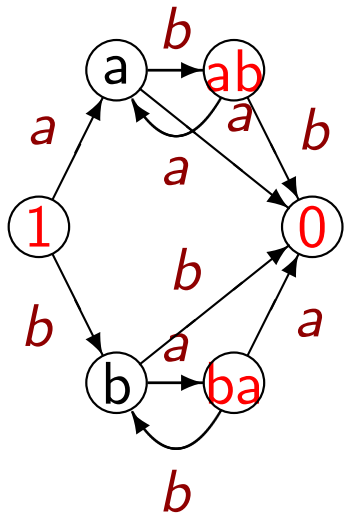
$$M(\mathcal{A}) = (\{f_w \mid w \in A^*\}, \circ, f_\epsilon).$$

  with the morphism $h(w) = f_w$.
- $M(\mathcal{A})$ is a submonoid of $Fun(Q) = (Q \to Q, \circ, id)$, it is called the transition monoid of $\mathcal{A}$.

Example: DFA to Transition Monoid to DFA



∘	1	a	b	ab	ba	0
1	1	a	b	ab	ba	0
a	a	0	ab	0	a	0
b	b	ba	0	b	0	0
ab	ab	a	0	ab	0	0
ba	ba	0	b	0	ba	0
0	0	0	0	0	0	0



Distinct elements of $M(\mathcal{A})$ are
$\{f_\epsilon = 1, f_a, f_b, f_{ab}, f_{ba}, f_{aa} = f_{bb} = 0\}$.
Multiplication table as shown. Idempotents e satisfy $ee = e$.
Right-multiplication by generators a, b is coloured.

Write f_a as $\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 3 \end{pmatrix}$, or (233). $f_b = (313), f_{ab} = (133)$, $f_{ba} = (323), 0 = (333)$.

Question: If Q is finite, how many elements can $M(\mathcal{A})$ have?

Homework

Problem (McNaughton-Papert 1971)

Find transition monoids for two DFA with states $\{s, p, q\}$, initial state s. Also construct the automata from the monoids and compare what you get.

1. $\delta_1 = s \xrightarrow{a} p, p \xrightarrow{a} s, q \xrightarrow{a} q,\ s \xrightarrow{b} s, p \xrightarrow{b} q, q \xrightarrow{b} p$
2. $\delta_2 = s \xrightarrow{a} p, p \xrightarrow{a} q, q \xrightarrow{a} q,\ s \xrightarrow{b} s, p \xrightarrow{b} s, q \xrightarrow{b} p$

- Two automata look similar
- First monoid has 6 elements, second monoid has 8 elements. Using them can say something about languages they recognize
- There is a language recognized by first monoid which is not definable in first-order logic with $<$. All languages recognized by second one are defined by first-order logic sentences

Syntactic Monoid of a language

- Let $\mathcal{A}_{\equiv_L} = (Q, s, \delta, F)$ be the canonical automaton for a language $L \subseteq A^*$.
- The transition monoid of $\mathcal{A}_{\equiv_L}$ is called the syntactic monoid of L, we write $M(L)$ rather than $M(\mathcal{A}_{\equiv_L})$.
- The surjective morphism $h_L : A^* \to M(L)$ is called the syntactic morphism of L.

Syntactic Congruence of a language

- Define an equivalence relation $\cong_L$ on A^*, induced by L, as

$$u \cong_L v \text{ iff } \forall x, y \in A^* : \ xuy \in L \text{ iff } xvy \in L.$$

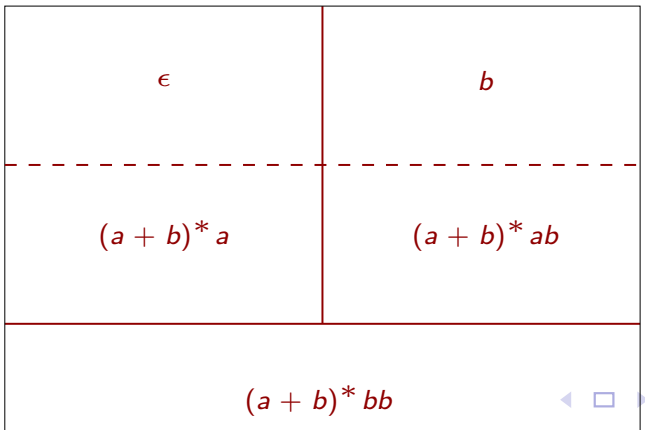
- $\cong_L$ is called the syntactic congruence of L.
- Check that $\cong_L$ is a two-sided congruence:
  - That is, $\cong_L$ is both a left-congruence (i.e $u \cong_L v$ implies $wu \cong_L wv$, for each $w \in A^*$) and a right-congruence (i.e $u \cong_L v$ implies $uw \cong_L vw$).
  - Equivalently, $u \cong_L u'$ and $v \cong_L v'$ imples $uv \cong_L u'v'$.
- $\cong_L$ refines the canonical MN relation, $\equiv_L$, for L.
- Example?

Syntactic Congruence of a language

- Define an equivalence relation $\cong_L$ on A^*, induced by L, as

$$u \cong_L v \text{ iff } \forall x, y \in A^* : \ xuy \in L \text{ iff } xvy \in L.$$

- $\cong_L$ is called the syntactic congruence of L.
- Check that $\cong_L$ is a two-sided congruence:
  - That is, $\cong_L$ is both a left-congruence (i.e $u \cong_L v$ implies $wu \cong_L wv$, for each $w \in A^*$) and a right-congruence (i.e $u \cong_L v$ implies $uw \cong_L vw$).
  - Equivalently, $u \cong_L u'$ and $v \cong_L v'$ imples $uv \cong_L u'v'$.
- $\cong_L$ refines the canonical MN relation, $\equiv_L$, for L.
- Example? Consider the language $(a+b)^*bb$:



Characterization of the syntactic monoid

Claim

For a canonical DA $\mathcal{A} = (Q, \delta, s, F)$,

$$f_u = f_v \text{ iff } u \cong_L v.$$

Proof: ($\Longrightarrow$) By definition the element f_w of the transition monoid of the canonical DA is $\widehat{\delta}(\_, w)$.

$xuy \in L$ iff $(f_x \circ f_u \circ f_y)(s) \in F$ in the canonical DA
(in other words, $\widehat{\delta}(s, xuy) \in F$) in the canonical DA
iff $(f_x \circ f_v \circ f_y)(s) \in F$ in the canonical DA
(in other words, $\widehat{\delta}(s, xvy) \in F$) in the canonical DA
iff $xvy \in L$

Exercise: Prove the ($\Longleftarrow$) direction.

Thus the syntactic congruence $u \cong_L v$ matches the equality $f_u = f_v$ derived from the canonical DA.

Syntactic monoid via syntactic congruence

For a language $L \subseteq A^*$, consider the monoid $A^*/\cong_L$:

- elements are congruence classes under $\cong_L$,
- take the unit to be $[\epsilon]$,
- multiply two congruences by defining:

$$[u] \circ [v] = [uv].$$

Claim

The monoids $M(L)$ and $A^*/\cong_L$ are isomorphic.

(Use the morphism $f_w \mapsto [w]$.)

Algebraic definition of regular languages

Theorem (Myhill-Nerode)

Let $L \subseteq A^*$. Then the following are equivalent:

1. *L is regular*
2. *The syntactic monoid of L, i.e. $M(L)$, is finite*
3. *L is recognizable*

Proof:

Algebraic definition of regular languages

Theorem (Myhill-Nerode)

Let $L \subseteq A^*$. Then the following are equivalent:

1. *L is regular*
2. *The syntactic monoid of L, i.e. $M(L)$, is finite*
3. *L is recognizable*

Proof:

(1) $\Longrightarrow$ (2): since $\mathcal{A}_{\equiv_L}$ is finite, and hence so is $M(L)$.

Algebraic definition of regular languages

Theorem (Myhill-Nerode)

Let $L \subseteq A^*$. Then the following are equivalent:

1. *L is regular*
2. *The syntactic monoid of L, i.e. $M(L)$, is finite*
3. *L is recognizable*

Proof:

(1) $\implies$ (2): since $\mathcal{A}_{\equiv_L}$ is finite, and hence so is $M(L)$.

(2) $\implies$ (3): Syntactic morphism $h_L : A^* \to M(L)$, given by $w \mapsto f_w$, recognizes L.

Algebraic definition of regular languages

Theorem (Myhill-Nerode)

Let $L \subseteq A^*$. Then the following are equivalent:

1. *L is regular*
2. *The syntactic monoid of L, i.e. $M(L)$, is finite*
3. *L is recognizable*

Proof:

(1) $\implies$ (2): since $\mathcal{A}_{\equiv_L}$ is finite, and hence so is $M(L)$.

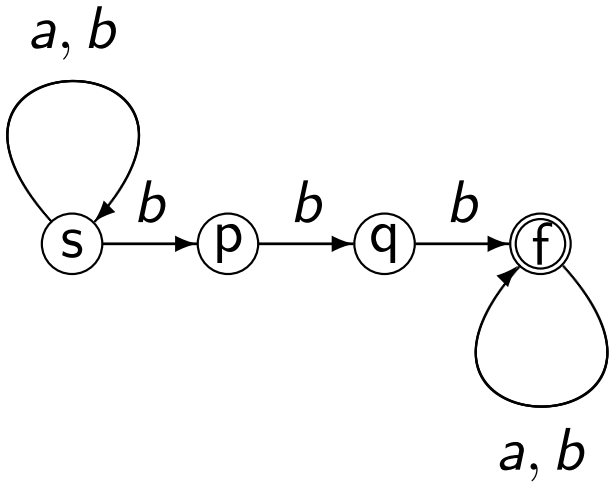
(2) $\implies$ (3): Syntactic morphism $h_L : A^* \to M(L)$, given by $w \mapsto f_w$, recognizes L.

(3) $\implies$ (1): Given finite M recognizing L, finite $\mathcal{A}(M)$ accepts L.

Corollary (Rabin-Scott, Shepherdson-Sturgis)

Nondeterministic and two-way automata accept regular languages

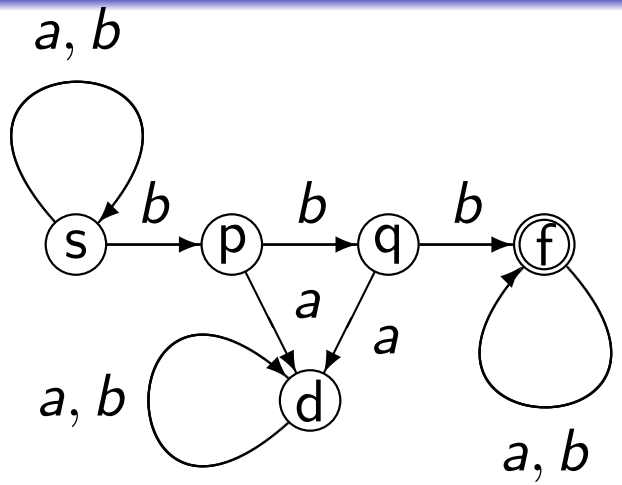
NFA to monoid of relations



$$\Delta(a) = \{(s,s),(f,f)\}$$
$$\Delta(b) = \{(s,s),(s,p),(p,q),(q,f),(f,f)\}$$
$$\widehat{\Delta}(\epsilon) = id = \{(s,s),(p,p),(q,q),(f,f)\}$$
$$\widehat{\Delta}(bb) = \{(s,s),(s,p),(s,q),(p,f),(f,f)\}$$
$$\widehat{\Delta}(bbb) = \{(s,s),(s,p),(s,q),(s,f),(f,f)\}$$
$$\widehat{\Delta}(ab) = \{(s,s),(s,p),(f,f)\}$$
$$\widehat{\Delta}(abb) = \{(s,s),(s,p),(s,q),(f,f)\}$$

- Think of (Q,Δ), where every letter of the alphabet defines a transition relation $\Delta : A \to 2^{Q\times Q}$
- $\widehat{\Delta}$ as mapping a word to a relation on states $A^* \to 2^{Q\times Q}$
- $\widehat{\Delta}(\epsilon) = id, \quad \widehat{\Delta}(wa) = \widehat{\Delta}(w) \circ \Delta(a)$
- That is, $\widehat{\Delta}$ is a morphism into monoid of relations $Rel(Q)$

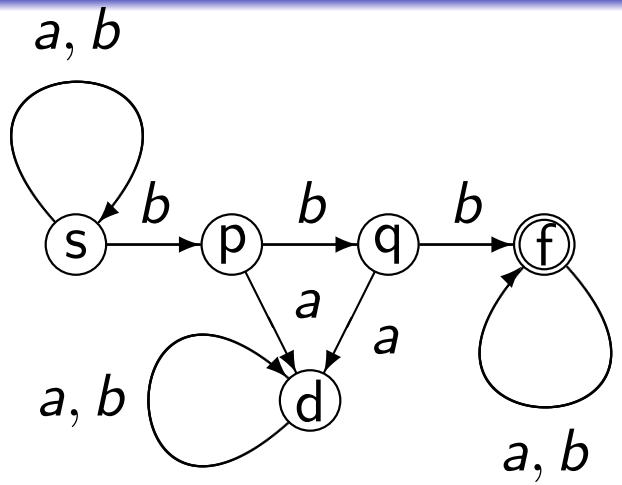
Rabin-Scott construction as a monoid construction



∘	a	b	ab	bb	abb	bbb
a	a	ab	ab	abb	abb	bbb
b	a	bb	b	bbb	bb	bbb
ab	a	abb	b	bbb	bb	bbb
bb	a	bbb	b	bbb	bb	bbb
abb	a	bbb	b	bbb	bb	bbb
bbb	a	bbb	b	bbb	bb	bbb

- Complete the NFA to make it total, then take $Rel(Q)$
- The monoid has 8 elements $\{1, a, b, ab, bb, abb, bbb, 0\}$
- 1,0 are not shown in the multiplication table
- Follow abstractly how the monoid is calculated

Rabin-Scott construction as a monoid construction

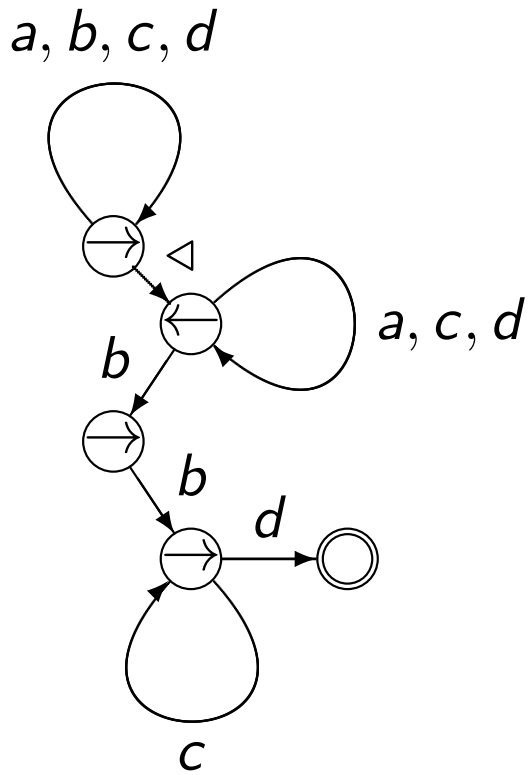


∘	a	b	ab	bb	abb	bbb
a	a	ab	ab	abb	abb	bbb
b	a	bb	b	bbb	bb	bbb
ab	a	abb	b	bbb	bb	bbb
bb	a	bbb	b	bbb	bb	bbb
abb	a	bbb	b	bbb	bb	bbb
bbb	a	bbb	b	bbb	bb	bbb

- Complete the NFA to make it total, then take $Rel(Q)$
- The monoid has 8 elements $\{1, a, b, ab, bb, abb, bbb, 0\}$
- 1,0 are not shown in the multiplication table
- Follow abstractly how the monoid is calculated
- Exercise: Which monoid elements correspond to initial and final states of the automaton?

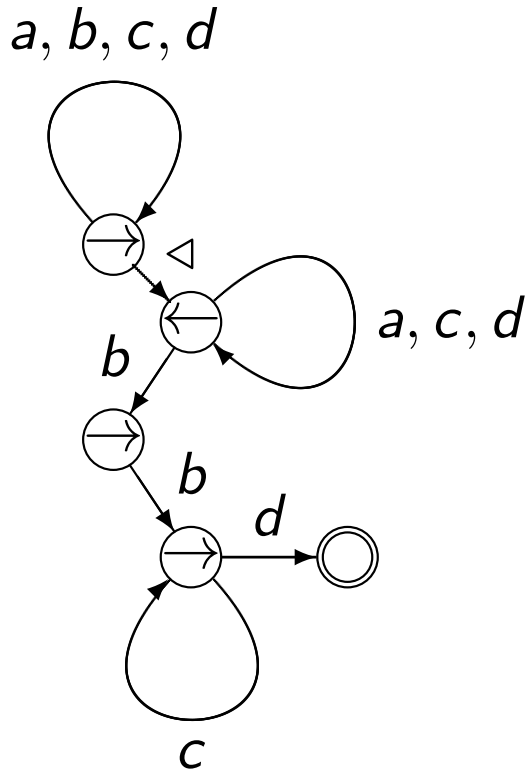
Shepherdson construction for two-way automata

- Left and right endmarkers $\triangleright w \triangleleft$ in alphabet A for input word w
- Q is union of left-moving states $\overleftarrow{Q}$ and right-moving states $\overrightarrow{Q}$
- Transition $\delta : \overleftarrow{Q} \times (A \setminus \{\triangleright\}) \to Q$ reads letter and moves left, $\delta : \overrightarrow{Q} \times (A \setminus \{\triangleleft\}) \to Q$ reads letter and moves right as usual
- Partial functions (2DFA): moving left of left endmarker and moving right of right endmarker are undefined



Birget monoids for two-way automata

- Have to consider four types of morphic extensions for δ:
  - Either $h(\rightarrow w \rightarrow) : \overrightarrow{Q} \rightarrow \overrightarrow{Q}$ enters from left exits from right
  - Or $h(\overrightarrow{\leftarrow} w) : \overrightarrow{Q} \rightarrow \overleftarrow{Q}$ enters from left exits from left
  - Either $h(\leftarrow w \leftarrow) : \overleftarrow{Q} \rightarrow \overleftarrow{Q}$ enters from right exits from left
  - Or $h(w \overleftarrow{\rightarrow}) : \overleftarrow{Q} \rightarrow \overrightarrow{Q}$ enters from right exits from right



Problem

Construct a monoid recognizing the language accepted by a 2DFA

(Warning: there are missing cases above)