Categories & Algebras 2

Automata and Monoids I

Graham Campbell

Summer 2017

Alphabets and Words

Definition (Alphabet)

An alphabet is a finite non-empty set A and a letter is an element of A.

Alphabets and Words

Definition (Alphabet)

An alphabet is a finite non-empty set A and a letter is an element of A.

Definition (Word)

Given alphabet A, a word (or string) is a finite sequence of elements of A.

Alphabets and Words

Definition (Alphabet)

An alphabet is a finite non-empty set A and a letter is an element of A.

Definition (Word)

Given alphabet A, a word (or string) is a finite sequence of elements of A.

Definition (Kleene operators)

We define $A^+ = \bigcup_{i \geq 1} A^i$ to be the set of all non-empty words over A, and $A^* = A^+ \cup \{\lambda\}$ where λ is the empty string.

Languages and Monoids

Definition (Language)

A language is L over A is a subset of A^*, is finite iff $|L| < \infty$, and is cofinite iff $|L^c| = |A^* - L| < \infty$.

Languages and Monoids

Definition (Language)

A language is L over A is a subset of A^*, is finite iff $|L| < \infty$, and is cofinite iff $|L^c| = |A^* - L| < \infty$.

Definition (Free monoid)

A^* is a monoid with identity λ, where λ is the empty string, called the free monoid on A.

Languages and Monoids

Definition (Language)

A language is L over A is a subset of A^*, is finite iff $|L| < \infty$, and is cofinite iff $|L^c| = |A^* - L| < \infty$.

Definition (Free monoid)

A^* is a monoid with identity λ, where λ is the empty string, called the free monoid on A.

All the string and languages operations we know from MFCS carry over. We are not going to recap them again.

Automata

Definition (FSA)

Recall from MFCS that a FSA is a 5-tuple:

$$\mathcal{A} = (A, Q, \delta, q_0, F),$$

where transition function $\delta : Q \times A \to Q$ can be extended in the obvious way to give $\delta : Q \times A^* \to Q$.

Automata

Definition (FSA)

Recall from MFCS that a FSA is a 5-tuple:

$$\mathcal{A} = (A, Q, \delta, q_0, F),$$

where transition function $\delta : Q \times A \to Q$ can be extended in the obvious way to give $\delta : Q \times A^* \to Q$.

Definition (Language recognised)

The language recognised by $\mathcal{A}$ is:

$$L(\mathcal{A}) = \{w \in A^* \mid \delta(q_0, w) \in F\}.$$

Recognisable languages

Definition (Recognisable languages)

A language $L \subseteq A^*$ is recognisable iff there exists an FSA $\mathcal{A}$ such that:

$$L = L(\mathcal{A}),$$

and we write $L \in \mathrm{Rec}\, A^*$.

Recognisable languages

Definition (Recognisable languages)

A language $L \subseteq A^*$ is recognisable iff there exists an FSA $\mathcal{A}$ such that:

$$L = L(\mathcal{A}),$$

and we write $L \in \operatorname{Rec} A^*$.

We note that $A^*, \emptyset \in \operatorname{Rec} A^*$, and all singleton languages lie in $\operatorname{Rec} A^*$. This is easy to see by constructing automata.

Lemmas 1

Lemma

$L \in \mathrm{Rec}\, A^* \Rightarrow L^c \in \mathrm{Rec}\, A^*$.

Lemmas 1

Lemma

$L \in \mathrm{Rec}\, A^* \Rightarrow L^c \in \mathrm{Rec}\, A^*$.

Lemma

$L, K \in \mathrm{Rec}\, A^* \Rightarrow L \cup K \in \mathrm{Rec}\, A^*$.

Lemmas 1

Lemma

$L \in \mathrm{Rec}\, A^* \Rightarrow L^c \in \mathrm{Rec}\, A^*$.

Lemma

$L, K \in \mathrm{Rec}\, A^* \Rightarrow L \cup K \in \mathrm{Rec}\, A^*$.

Lemma

$L, K \in \mathrm{Rec}\, A^* \Rightarrow L \cap K \in \mathrm{Rec}\, A^*$.

Lemmas 1

Lemma

$L \in \mathrm{Rec}\, A^* \Rightarrow L^c \in \mathrm{Rec}\, A^*.$

Lemma

$L, K \in \mathrm{Rec}\, A^* \Rightarrow L \cup K \in \mathrm{Rec}\, A^*.$

Lemma

$L, K \in \mathrm{Rec}\, A^* \Rightarrow L \cap K \in \mathrm{Rec}\, A^*.$

Lemma

$L, K \in \mathrm{Rec}\, A^* \Rightarrow L - K \in \mathrm{Rec}\, A^*.$

Lemmas 2

We remark that $\operatorname{Rec} A^*$ is not closed under infinite union or intersection, however, with finitely many applications, one can see:

1. L finite $\Rightarrow L \in \operatorname{Rec} A^*$,
2. L cofinite $\Rightarrow L \in \operatorname{Rec} A^*$.

Lemmas 2

We remark that $\operatorname{Rec} A^*$ is not closed under infinite union or intersection, however, with finitely many applications, one can see:

1. L finite $\Rightarrow L \in \operatorname{Rec} A^*$,
2. L cofinite $\Rightarrow L \in \operatorname{Rec} A^*$.

We will quote two more results without proof:

Lemma

$L, K \in \operatorname{Rec} A^* \Rightarrow LK \in \operatorname{Rec} A^*$.

Lemma

$L \in \operatorname{Rec} A^* \Rightarrow L^* \in \operatorname{Rec} A^*$.

Rational Languages

Definition (Rational operations)

The rational operations on languages over A are:

1. union $(L, K \mapsto L \cup K)$,
2. product $(L, K \mapsto LK)$,
3. star $(L \mapsto L^*)$.

Rational Languages

Definition (Rational operations)

The rational operations on languages over A are:

1. union $(L, K \mapsto L \cup K)$,
2. product $(L, K \mapsto LK)$,
3. star $(L \mapsto L^*)$.

Definition (Rational languages)

$L \subseteq A^*$ is rational iff L is finite, or L can be obtained by applying rational operations a finite number of times.

We write $\mathrm{Rat}\, A^*$ for the set of rational languages over A.

Kleene's Theorem

From our earlier results, we know that $\operatorname{Rat} A^* \subseteq \operatorname{Rec} A^*$, however, we can take this one step further.

Theorem (Kleene)

$\operatorname{Rat} A^* = \operatorname{Rec} A^*$.

Kleene's Theorem

From our earlier results, we know that $\operatorname{Rat} A^* \subseteq \operatorname{Rec} A^*$, however, we can take this one step further.

Theorem (Kleene)

$\operatorname{Rat} A^* = \operatorname{Rec} A^*$.

Definition (Rational expressions)

A rational expression for a language L over A is one that expresses L using only finite languages and the rational operations a finite number of times.

Monoids

Definition (Monoid)

A monoid is a non-empty set M, together with an associative binary operation, and equipped with a two-sided identity.

Monoids

Definition (Monoid)

A monoid is a non-empty set M, together with an associative binary operation, and equipped with a two-sided identity.

Definition (Submonoid)

Let M be a monoid, and $T \subseteq M$. Then T is a submonoid iff:

1. $1 \in T$,
2. $a, b \in T \Rightarrow ab \in T$,

and we write $T \leq M$.

Generators

Definition (Generator)

Let M be a monoid, and $X \subseteq M$. Then:

$$\langle X \rangle = \{x_1 x_2 \cdots x_n \mid n \geq 0, x_i \in X\}.$$

Generators

Definition (Generator)

Let M be a monoid, and $X \subseteq M$. Then:

$$\langle X \rangle = \{x_1 x_2 \cdots x_n \mid n \geq 0, x_i \in X\}.$$

Notice that $1 \in \langle X \rangle$, and $\langle X \rangle$ is closed under multiplication, so is a submonoid. We say $\langle X \rangle$ is the submonoid of M generated by X.

Generators

Definition (Generator)

Let M be a monoid, and $X \subseteq M$. Then:

$$\langle X \rangle = \{x_1 x_2 \cdots x_n \mid n \geq 0, x_i \in X\}.$$

Notice that $1 \in \langle X \rangle$, and $\langle X \rangle$ is closed under multiplication, so is a submonoid. We say $\langle X \rangle$ is the submonoid of M generated by X.

Examples

1. $A^* = \langle A \rangle$ where A is an alphabet.
2. $\mathbb{N} = \langle P \rangle$ where P is the set of primes.

Transformation Monoids

Definition (Full transformation monoid)

If X is a non-empty set, then:

$$\mathcal{T}_X = \{\alpha \mid \alpha : X \to X\}$$

is a monoid under functional composition, with identity I_X, and is called the "full transformation monoid on X".

Transformation Monoids

Definition (Full transformation monoid)

If X is a non-empty set, then:

$$\mathcal{T}_X = \{\alpha \mid \alpha : X \to X\}$$

is a monoid under functional composition, with identity I_X, and is called the "full transformation monoid on X".

Notation

From now onward, we will be using postfix notation for functional application, and if $X = \{1, 2, ...n\}$, write $\mathcal{T}_n$ for $\mathcal{T}_X$ and I_n for I_X.

It is also convenient to adopt two-row notation for elements of $\mathcal{T}_n$.

Constant Functions

Definition (Constant function)

For fixed $x \in X$, $c_x : X \to X$ is given by:

$$yc_x = x$$

for any $y \in X$, and is called the "constant function on x".

Constant Functions

Definition (Constant function)

For fixed $x \in X$, $c_x : X \to X$ is given by:

$$yc_x = x$$

for any $y \in X$, and is called the "constant function on x".

Remark

Note that:

1. $\forall \alpha \in \mathcal{T}_X, \alpha c_x = c_x$,
2. $\forall \alpha \in \mathcal{T}_X, c_x \alpha = c_{x\alpha}$.

Preliminary Result

Lemma

Let $\mathcal{A} = (A, Q, \delta, q_0, F)$ *be a FSA.*

For each $w \in A^*$, *let* $\sigma_w \in \mathcal{T}_Q$ *be defined by:*

$$q\sigma_w = \delta(q, w).$$

Preliminary Result

Lemma

Let $\mathcal{A} = (A, Q, \delta, q_0, F)$ *be a FSA.*

For each $w \in A^*$, *let* $\sigma_w \in \mathcal{T}_Q$ *be defined by:*

$$q\sigma_w = \delta(q, w).$$

Then, for any $w, v \in A^*$*:*

$$\sigma_w \sigma_v = \sigma_{wv}.$$

Preliminary Result

Lemma

Let $\mathcal{A} = (A, Q, \delta, q_0, F)$ be a FSA.

For each $w \in A^*$, let $\sigma_w \in \mathcal{T}_Q$ be defined by:

$$q\sigma_w = \delta(q, w).$$

Then, for any $w, v \in A^*$:

$$\sigma_w \sigma_v = \sigma_{wv}.$$

Remark

In particular, we note that $q\sigma_\lambda = \delta(q, \lambda) = q = qI_Q$.

Transition Monoids

So, we have an identity σ_λ, and closed multiplication.

Definition (Transition monoid)

Thus, we have submonoid:

$$M(\mathcal{A}) = \{\sigma_w \mid w \in A^*\} \leq \mathcal{T}_Q.$$

We call this the transition monoid of FSA $\mathcal{A}$.

Transition Monoids

So, we have an identity σ_λ, and closed multiplication.

Definition (Transition monoid)

Thus, we have submonoid:

$$M(\mathcal{A}) = \{\sigma_w \mid w \in A^*\} \leq \mathcal{T}_Q.$$

We call this the transition monoid of FSA $\mathcal{A}$.

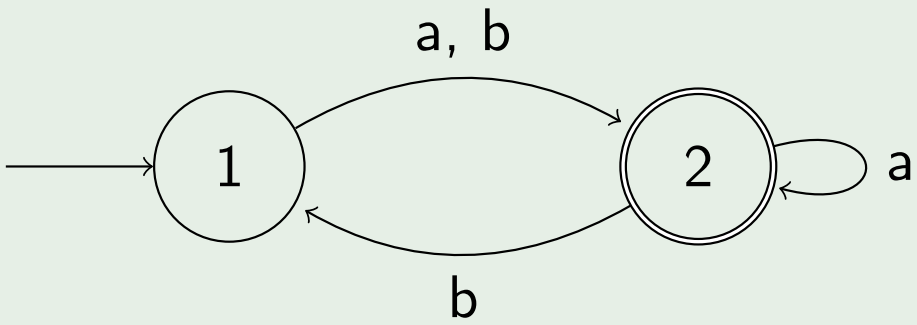
Lemma

$M(\mathcal{A}) = \langle \sigma_a \mid a \in A \rangle$, and is finite.

Example 1

Let $A = \{a, b\}$ and $Q = \{1, 2\}$.

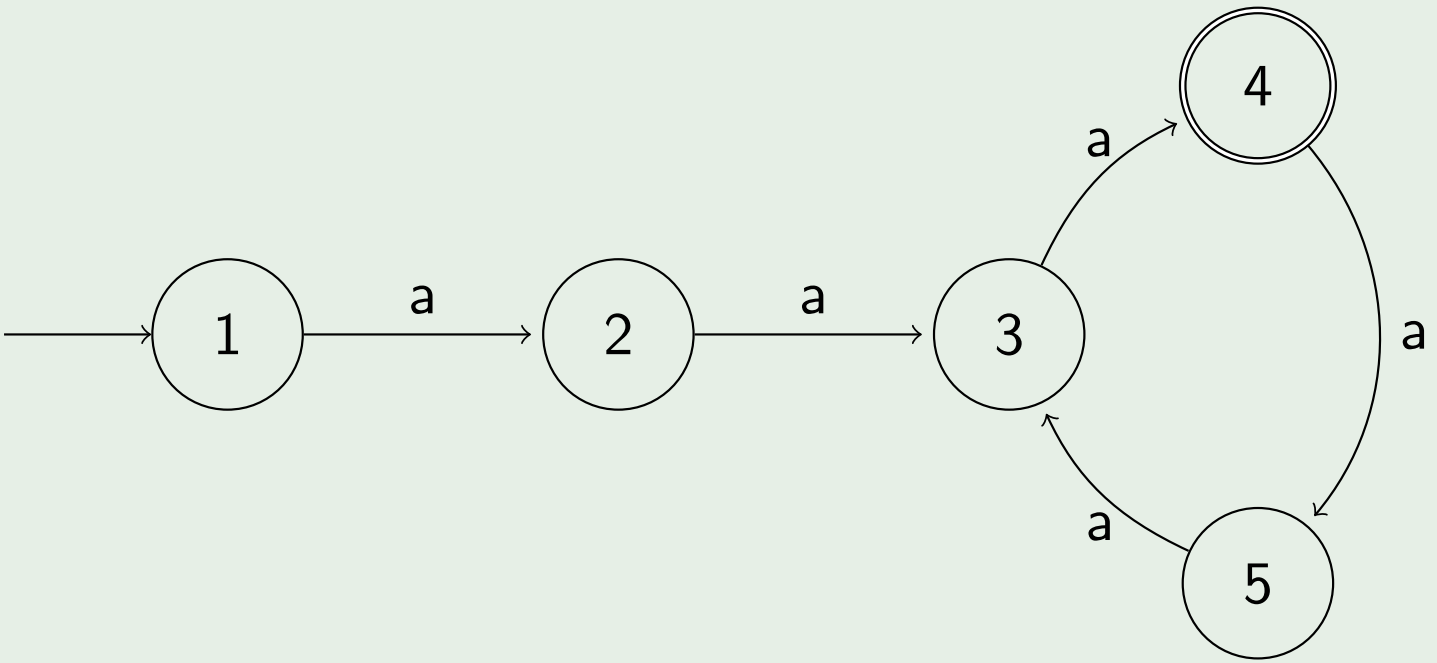
State Diagram



Example 2

Let $A = \{a\}$ and $Q = \{1, 2, 3, 4, 5\}$.

State Diagram



Example 3

Let $A = \{a, b\}$ and $Q = \{1, 2, 3\}$.

State Diagram