Supplementary Materials

Monoids and Monoid Morphisms

- A monoid is a structure $(M, \circ, 1)$, where
  - M is a base set containing the element "1",
  - $\circ$ is an associative binary operation on M, and
  - 1 is the identity element with respect to $\circ$.
- Examples of monoids: $(N, +, 0)$, $(A^*, \cdot, \epsilon)$.
- Another example: $(X \rightarrow X, \circ, id)$, where
  - $X \rightarrow X$ denotes the set of all functions from a set X to itself,
  - $f \circ g$ is a function composition: $(f \circ g)(x) = f(g(x))$
- A morphism from a monoid $(M, \circ_M, 1_M)$ to $(N, \circ_N, 1_N)$ is a mapping $\psi : M \rightarrow N$, satisfying
  - $\psi(1_M) = 1_N$, and,
  - $\psi(m \circ_M m') = \psi(m) \circ_N \psi(m')$.
- Example: $\psi : A^* \rightarrow N$, given by $\psi(w) = |w|$ is a morphism from $(A^*, \cdot, \epsilon)$ to $(N, +, 0)$.

Language Recognition via Monoid Morphisms

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

$$L = \psi^{-1}(X)$$

- In this case, we say that the monoid M recognizes L.

Example of Language Recognition via Monoid

Consider monoid $M = (\{1, m\}, \circ, 1)$, where $\circ$ is given by:

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

Consider the morphism $\psi : A^* \to M$ given by

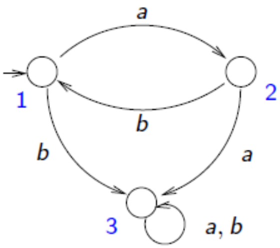
$$\epsilon \to 1 \text{ and } w \to m, \ \text{ for } w \in A^*.$$

Then M recognizes A^+, since $\psi^{-1}(\{m\}) = A^+$. Notice that M also recognizes $\{\epsilon\}$, A^*, and $\emptyset$.

Transition Monoid of a DFA

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

- For $w \in A^*$, define $f_w : Q \to Q$ by $f_w(q) = \delta(q, w)$
- Consider the monoid $M(\mathcal{A}) = (\{f_w \mid w \in A^*\}, \circ, 1)$.
- $M(\mathcal{A})$ is called the transition monoid of $\mathcal{A}$.



Distinct elements of $M(\mathcal{A})$ are $\{f_\epsilon, f_a, f_b, f_{aa}, f_{ab}, f_{ba}\}$, where f_a is $1 \to 2, 2 \to 3, 3 \to 3$.

Algebraic Characterization of Regular Languages

Theorem 1

Let $L \subseteq A^*$. The following are equivalent.

1. *L is regular.*
2. *L is recognized by a finite monoid, i.e., $\exists$ a finite monoid $(M, \circ, 1)$ and a morphism $\psi : (A^*, \cdot, \epsilon) \to (M, \circ, 1)$ and a $X \subseteq M$, $L = \psi^{-1}(X)$.*

Proof.

1. (1)$\Rightarrow$ (2). Let $\mathcal{A} = (Q, A, q_0, \delta, F)$ be a DFA accepting L. Consider the transition monoid $M(\mathcal{A}) = (\{f_w \mid w \in A^*\}, \circ, 1)$, which is clearly finite. Consider the morphism $\psi : (A^*, \cdot, \epsilon) \to (\{f_w \mid w \in A^*\}, \circ, 1)$ with $\psi(w) = f_w, w \in A^*$, and $X = \{f_w \mid f_w(q_0) \in F\}$. It is not hard to see that $L = \psi^{-1}(X)$.
2. (2) $\Rightarrow$ (1). Define a DFA $\mathcal{A} = (M, A, 1, \delta, X)$, where $\delta(m, a) = m \circ \psi(a)$.

$\square$

Another Algebraic View of DFA

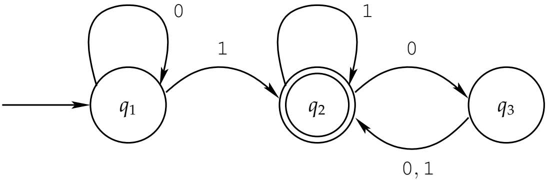


Figure: A Finite Automaton M_1

Consider the following matrix representation:

- Initial state $I = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$; final state $F = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$;

$$M_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}; M_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

Algebraic View of DFA

The computation $q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_3 \xrightarrow{1} q_2$ is represented by

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} =$$

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} =$$

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}^T \cdot \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}^T$$

As $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}^T \cdot F = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = 1$, the input "101" is accepted.

Algebraic View of NFA

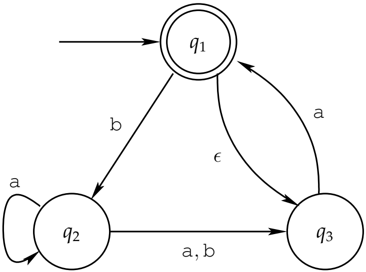


Figure: NFA N_4

$$M_a = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}; M_b = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}; M_\epsilon = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Matrix Multiplication

Question:
How to define matrix multiplication

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \cdot \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix}$$ for the above examples"

- In $(a_{1,1} \cdot b_{1,1} + a_{1,2} \cdot b_{2,1} + a_{1,3} \cdot b_{3,1})$, for instance, the operations "." and "+" stand for integer multiplication and addition, resp.
- Suppose "1" and "0" stand for Boolean "True" and "False", resp., the operations "." and "+" stand for Boolean operations $\wedge$ and $\vee$, resp.
- Hence, conventional FA are with respect to $(\vee, \wedge)$-Semiring.

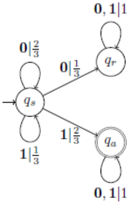
Semiring

A semiring is a set R equipped with two binary operations $+$ and $\cdot$, called addition and multiplication, such that

- $(R, +)$ is a commutative monoid with identity element 0:
  - $(a + b) + c = a + (b + c)$
  - $0 + a = a + 0 = a$
  - $a + b = b + a$
- $(R, \cdot)$ is a monoid with identity element 1:
  - $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
  - $1 \cdot a = a \cdot 1 = a$
- Multiplication left and right distributes over addition:
  - $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
  - $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$
- Multiplication by 0 annihilates R:
  - $0 \cdot a = a \cdot 0 = 0$

Note: a monoid is an algebraic structure with a single associative binary operation and an identity element.

Probabilistic FA: $(+, \times)$-Semiring



PFA A_0: $q_s = q_1, q_r = q_2, q_a = q_3$

$$M_0 = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}; M_1 = \begin{pmatrix} \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

On input 011, we calculate

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} \frac{2}{3} & \frac{1}{3} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} \frac{2}{9} \\ \frac{1}{3} \\ \frac{4}{9} \end{pmatrix}^T \cdot \begin{pmatrix} \frac{1}{3} & 0 & \frac{2}{3} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{2}{27} \\ \frac{1}{3} \\ \frac{16}{27} \end{pmatrix}^T$$

, where $\frac{16}{27}$ corresponds to

- $q_s \overset{0|\frac{2}{3}}{\rightarrow} a_s \overset{1|\frac{1}{3}}{\rightarrow} q_s \overset{1|\frac{2}{3}}{\rightarrow} q_a \Rightarrow$ prob. $= \frac{4}{27}$
- $q_s \overset{0|\frac{2}{3}}{\rightarrow} a_s \overset{1|\frac{2}{3}}{\rightarrow} q_a \overset{1|1}{\rightarrow} q_a \Rightarrow$ prob. $= \frac{4}{9}$

Probabilistic Finite Automaton – Formal Definition

A <u>probabilistic finite automaton</u> (PFA) A is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

- Q is a finite set of states;
- Σ is a finite alphabet;
- $\delta : Q \times \Sigma \times Q \to [0, 1]$ is the transition function, such that $\forall q, \in Q, \forall a \in \Sigma, \sum_{q' \in Q} \delta(q, a, q') = 1$, where $\delta(q, a, q')$ is a rational number;
- $q_0 \in Q$ is the start state; and
- $F \subseteq Q$ is the accept states.

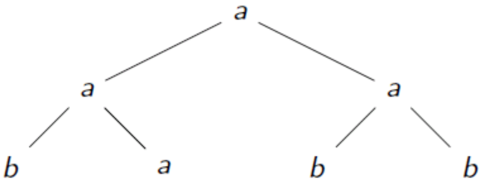
The language $L_{\diamondsuit x}(A) = \{u \in \Sigma^* \mid P_A(u) \diamondsuit x\}$, where $P_A(u)$ is the probability of acceptance on u, $x \in [0, 1]$, and $\diamondsuit \in \{<, \leq, =, \geq, >\}$.

- In general, $L_{\diamondsuit x}(A)$ may not be regular. For instance, $L_{>\frac{1}{2}}(A_0)$ and $L_{\geq\frac{1}{2}}(A_0)$ are not regular.
- $L_{\diamondsuit x}(A)$ is regular, if $x \in \{0, 1\}$.

Why Tree Automata?

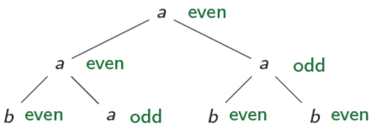
- Foundations of XML type languages (DTD, XML Schema, Relax NG...)
- Provide a general framework for XML type languages
- A tool to define regular tree languages with an operational semantics
- Provide algorithms for efficient validation
- Basic tool for static analysis (proofs, decision procedures in logic)
- ...

E.g. Binary trees with an even number of a's

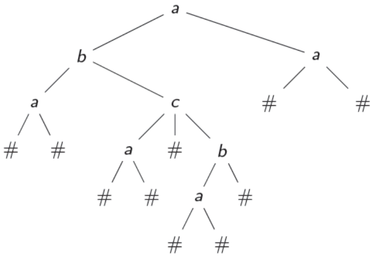


Binary Trees & Ranked Trees

- Binary trees with an even number of a's
- How to write transitions?
  - (even, odd) $\xrightarrow{a}$ even
  - (even, even) $\xrightarrow{a}$ odd
  - ...



- Ranked Tree:
  - Alphabet: $\{a^{(2)}, b^{(2)}, c^{(3)}, \#^{(0)}\}$
  - $a^{(k)}$: symbol a with arity$(a) = k$



Bottom-up (Ranked) Tree Automata

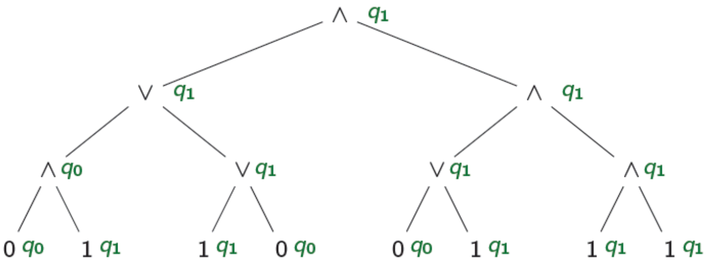
A **ranked bottom-up tree automaton** A consists of:

- $Alphabet(A)$: finite alphabet of symbols
- $States(A)$: finite set of states
- $Rules(A)$: finite set of transition rules
- $Final(A)$: finite set of final states ($\subseteq States(A)$)

where $Rules(A)$ are of the form $(q_1, ..., q_k) \stackrel{a^{(k)}}{\rightarrow} q$;

if $k = 0$, we write $\epsilon \stackrel{a^{(0)}}{\rightarrow} q$

Bottom-up Tree Automata: An Example



Principle

- Alphabet$(A) = \{\wedge, \vee, 0, 1\}$
- States$(A) = \{q_0, q_1\}$
- 1 accepting state at the root: Final$(A) = \{q_1\}$

Rules(A)

$\epsilon \xrightarrow{0} q_0$ $\quad$ $\epsilon \xrightarrow{1} q_1$

$(q_1, q_1) \xrightarrow{\wedge} q_1$ $\quad$ $(q_0, q_1) \xrightarrow{\vee} q_1$

$(q_0, q_1) \xrightarrow{\wedge} q_0$ $\quad$ $(q_1, q_0) \xrightarrow{\vee} q_1$

$(q_1, q_0) \xrightarrow{\wedge} q_0$ $\quad$ $(q_1, q_1) \xrightarrow{\vee} q_1$

$(q_0, q_0) \xrightarrow{\wedge} q_0$ $\quad$ $(q_0, q_0) \xrightarrow{\vee} q_0$

Top-down (Ranked) Tree Automata

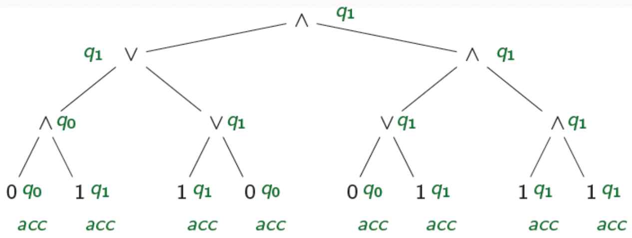
A **ranked top-down tree automaton** A consists of:

- $Alphabet(A)$: finite alphabet of symbols
- $States(A)$: finite set of states
- $Rules(A)$: finite set of transition rules
- $Final(A)$: finite set of final states ($\subseteq States(A)$)

where $Rules(A)$ are of the form $q \overset{a^{(k)}}{\rightarrow} (q_1, ..., q_k)$;
if $k = 0$, we write $\epsilon \overset{a^{(0)}}{\rightarrow} q$

Top-down tree automata also recognize all regular tree languages

Top-down Tree Automata: An Example



Principle

- starting from the root, guess correct values
- check at leaves
- 3 states: q_0, q_1, acc
- initial state at the root: q_1
- accepting if all leaves labeled acc

Transitions

$$
\begin{array}{ll}
q_1 \xrightarrow{\wedge} (q_1, q_1) & q_1 \xrightarrow{\vee} (q_0, q_1) \\
q_0 \xrightarrow{\wedge} (q_0, q_1) & q_1 \xrightarrow{\vee} (q_1, q_0) \\
q_0 \xrightarrow{\wedge} (q_1, q_0) & q_1 \xrightarrow{\vee} (q_1, q_1) \\
q_0 \xrightarrow{\wedge} (q_0, q_0) & q_0 \xrightarrow{\vee} (q_0, q_0) \\
q_1 \xrightarrow{1} \mathsf{acc} & q_0 \xrightarrow{0} \mathsf{acc}
\end{array}
$$

Expressive Power of Tree Automata

Theorem 2

The following properties are equivalent for a tree language L:

(a) *L is recognized by a* **bottom-up non-deterministic** *tree automaton*

(b) *L is recognized by a* **bottom-up deterministic tree** *automaton*

(c) *L is recognized by a* **top-down non-deterministic** *tree automaton*

(d) *L is generated by a* **regular** *tree grammar*

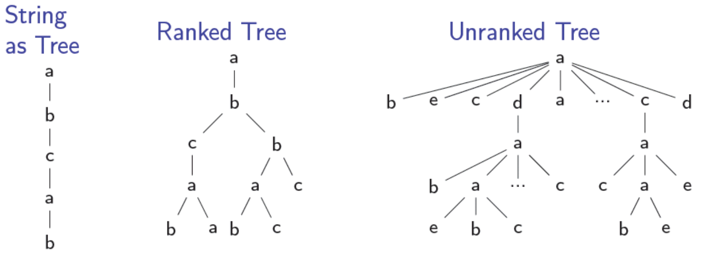
Deterministic Top-down Tree Automata

Deterministic top-down tree automata do not recognize all regular tree languages

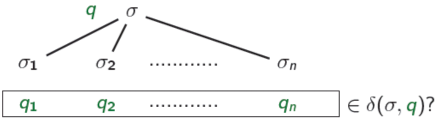
- Example:

Initial$(A) = q_0$
$q_0 \xrightarrow{a} (q, q)$
$q \xrightarrow{b} \epsilon$
$q \xrightarrow{c} \epsilon$
also accepts...

Unranked Trees



$\delta(\sigma, q)$: specified by a regular expression (i.e., regular language).



Quantum Entanglement

- An n-qubit system can exist in any superposition of the 2^n basis states.

$$\alpha_0|000...000\rangle + \alpha_1|000...001\rangle + \cdots + \alpha_{2^n-1}|111...111\rangle$$

- Sometimes such a state can be decomposed into the states of individual bits

$$\frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) = |0\rangle \otimes \frac{1}{\sqrt{2}}((|0\rangle + |1\rangle))$$

- But,

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

is not decomposible, which is called an <u>entangled state</u>.

Unitary Evolution

- A quantum system that is not measured (i.e. does not interact with its environment) evolves in a unitary fashion.
- That is, it's evolution in a time step is given by a unitary linear operation.
- Such an operator is described by a matrix U such that

$$UU^* = I$$

where U^* is the conjugate transpose of U.

$$\begin{pmatrix} 3 & 3+i \\ 2-i & 2 \end{pmatrix}^* = \begin{pmatrix} 3 & 2+i \\ 3-i & 2 \end{pmatrix}$$

Quantum Automata

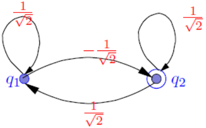
- Quantum finite automata are obtained by letting the matrices M_σ have complex entries. We also require each of the matrices to be unitary. E.g.

$$M_\sigma = \begin{pmatrix} -1 & 0 \\ 0 & i \end{pmatrix}$$

- If all matrices only have 0 or 1 entries and the matrices are unitary, then the automaton is deterministic and reversible.

Quantum Automata

Consider the automaton in a one letter alphabet as:



$$M_a = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix}$$

- The initial state $|\psi_0\rangle = 1 \cdot |0\rangle + 0 \cdot |1\rangle = (1,0)^T$
- $M_{aa} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$. Hence, upon reading *aa*, *M*'s state is

  $|\psi\rangle = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} = 0 \cdot |0\rangle + -1 \cdot |1\rangle$
- There are two distinct paths labelled *aa* from q_1 back to itself, and each has non-zero probability, the net probability of ending up in q_1 is 0.
- The automaton accepts a string of odd length with probability 0.5 and a string of even length with probability 1 if its length is not a multiple of 4 and probability 0 otherwise.

Measure-once Quantum Automata

- The accept state of the automaton is given by an $N \times N$ projection matrix P, so that, given a N-dimensional quantum state $|\psi\rangle$, the probability of $|\psi\rangle$ being in the accept state is $\langle\psi|P|\psi\rangle = \||P|\psi\rangle\|^2$.
  In the previous example, $P = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$
- The probability of the state machine accepting a given finite input string $\sigma = (\sigma_0, \sigma_1, \cdots, \sigma_k)$ is given by
  $Pr(\sigma) = \|PU_{\sigma_k} \cdots U_{\sigma_1} U_{\sigma_0}|\psi\rangle\|^2$. In the previous example, Pr(aa)=
  $\begin{pmatrix} 0 \\ -1 \end{pmatrix}^T \cdot \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} = 1$
- A regular language is accepted with probability p by a quantum finite automaton, if, for all sentences σ in the language, (and a given, fixed initial state $|\psi\rangle$), one has $p < Pr(\sigma)$.

Language Accepted

- Measure Many 1-way QFA: Measurement is performed after each input symbol is read.
- Measure-many model is more powerful than the measure-once model, where the power of a model refers to the acceptance capability of the corresponding automata.
- MM-1QFA can accept more languages than MO- 1QFA.
- Both of them accept proper subsets of regular languages.