

# Lecture: Weighted Automata

Benedikt Bollig

LSV, ENS CACHAN, CNRS

*E-mail address:* `bollig@lsv.ens-cachan.fr`

version: February 1, 2010

## Contents

Chapter 1. Motivation and Preliminaries	1
1.1. Motivation	1
1.2. Semirings and Closed Weighted Systems	3
1.3. Weighted automata	7
1.4. Threshold Languages and Decision Problems	9
Exercises for Chapter 1	10
Further Reading	11
Chapter 2. Word Transducers	13
2.1. Definition	13
2.2. Threshold Problems for Word Transducers	13
Exercises for Chapter 2	18
Further Reading	18
Chapter 3. Probabilistic Automata and Stochastic Languages	19
3.1. Definition	19
3.2. Stochastic Languages	20
3.3. Decision Problems	24
Exercises for Chapter 3	27
Further Reading	27
Chapter 4. Weighted Logic	29
4.1. MSO Logic over Words	29
4.2. Weighted MSO Logic over Words	31
4.3. From Logic to Automata	33
4.4. From Automata to Logic	35
Exercises for Chapter 4	36
Further Reading	37



## Motivation and Preliminaries

### 1.1. Motivation

Weighted graphs and automata offer a unifying framework for treating problems or modeling systems with the same structural properties. Consider the following problems, which all share a common structure:

- computing the language of/a regular expression for a given NFA
- finding shortest paths between all pairs of vertices in a graph
- computing the transitive closure of the edge relation of a graph
- computing probabilities in a Markov chain

**Semantics of an NFA.** Suppose we are given a finite automaton  $\mathfrak{A} = (Q, \Sigma, \Delta, q_0, F)$  with  $Q = \{1, \dots, n\}$  for some  $n \geq 1$ , alphabet  $\Sigma$ , and transition relation  $\Delta \subseteq Q \times \Sigma \times Q$ . Usually, we define the semantics of  $\mathfrak{A}$  to be its *language*, denoted by  $L(\mathfrak{A}) \subseteq \Sigma^*$ . When we ignore  $q_0$  and  $F$ , we obtain a triple  $\mathfrak{B} = (Q, \Sigma, \Delta)$ . As a semantics of  $\mathfrak{B}$ , we could think of a mapping

$$\|\mathfrak{B}\| : \begin{cases} Q \times Q \rightarrow 2^{\Sigma^*} \\ (i, j) \mapsto L(\mathfrak{A}_{i,j}) \end{cases}$$

where  $\mathfrak{A}_{i,j} = (Q, \Sigma, \Delta, i, \{j\})$ .

To compute  $\|\mathfrak{B}\|$ , we let, for  $i, j \in \{1, \dots, n\}$  and  $k \in \{0, \dots, n\}$ ,

$$W_{i,j}^{(k)} := \{w \in \Sigma^* \mid w \text{ leads from } i \text{ to } j \\ \text{without using } k+1, \dots, n \text{ as intermediate states}\}$$

We have

$$W_{i,j}^{(n)} = \|\mathfrak{B}\|(i, j)$$

$$W_{i,j}^{(k)} = \begin{cases} \{a \in \Sigma \mid (i, a, j) \in \Delta\} \cup \{\varepsilon \mid i = j\} & \text{if } k = 0 \\ W_{i,j}^{(k-1)} \cup W_{i,k}^{(k-1)} \left(W_{k,k}^{(k-1)}\right)^* W_{k,j}^{(k-1)} & \text{if } k \geq 1 \end{cases}$$

This characterization, which is illustrated in Figure 1, suggests an algorithm to infer a regular expression for a given NFA.

**Finding shortest paths.** Consider a (directed) graph  $G = (V, E, c)$ . Here,  $V = \{1, \dots, n\}$  is a nonempty finite set of *vertices*,  $E \subseteq V \times V$  is a set of *edges*, and  $c : E \rightarrow \mathbb{N}$  is a *cost function*. If we are interested in shortest paths, a semantics of  $G$  could be given as

$$\|G\| : \begin{cases} V \times V \rightarrow \mathbb{N} \\ (i, j) \mapsto \text{minimal cost of a path from } i \text{ to } j \end{cases}$$

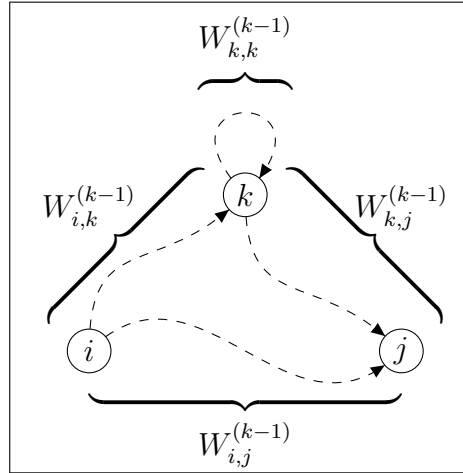


FIGURE 1.  $W_{i,j}^{(k)}$

For  $i, j \in \{1, \dots, n\}$  and  $k \in \{0, \dots, n\}$ , let

$$c_{i,j}^{(k)} := \text{minimal cost of a path from } i \text{ to } j \text{ without using } k+1, \dots, n$$

We have

$$c_{i,j}^{(n)} = \|G\|(i, j)$$

$$c_{i,j}^{(k)} = \begin{cases} 0 & \text{if } i = j \text{ and } k = 0 \\ c((i, j)) & \text{if } i \neq j \text{ and } (i, j) \in E \text{ and } k = 0 \\ \infty & \text{if } i \neq j \text{ and } (i, j) \notin E \text{ and } k = 0 \\ \min(c_{i,j}^{(k-1)}, c_{i,k}^{(k-1)} + c_{k,j}^{(k-1)}) & \text{if } k \geq 1 \end{cases}$$

This characterization suggests an algorithm to determine the minimal cost of a path between two nodes of a graph.

**Computing transitive closure.** Suppose we are interested in  $(V, E^*)$ , i.e., the (reflexive) transitive closure of  $G$ . Then, a semantics of  $G$  could be given as

$$\|G\| : \begin{cases} V \times V \rightarrow \{\text{true}, \text{false}\} \\ (u, v) \mapsto \text{“there is a path from } u \text{ to } v\text{”} \end{cases}$$

Let

$$e_{i,j}^{(k)} := \text{“there is a path from } i \text{ to } j \text{ that does not use } k+1, \dots, n\text{”}$$

We have

$$e_{i,j}^{(n)} = \|G\|(i, j)$$

$$e_{i,j}^{(k)} = \begin{cases} \text{true} & \text{if } (i = j \text{ or } (i, j) \in E) \text{ and } k = 0 \\ \text{false} & \text{if } i \neq j \text{ and } (i, j) \notin E \text{ and } k = 0 \\ e_{i,j}^{(k-1)} \vee (e_{i,k}^{(k-1)} \wedge e_{k,j}^{(k-1)}) & \text{if } k \geq 1 \end{cases}$$

This characterization suggests an algorithm to compute the reflexive transitive closure of a binary relation.

## 1.2. Semirings and Closed Weighted Systems

The semantics that we considered in the previous section rely on a specific interpretation of an edge (or an edge labeling), a path, and the subsumption of a set of paths. In general, these interpretations correspond to operations of a (closed) semiring.

DEFINITION 1.1. A monoid is a structure  $\mathbb{S} = (S, \otimes, \mathbb{1})$  where

- $S$  is a set,
- $\otimes : S \times S \rightarrow S$  is a binary operation that is *associative*, i.e.,  $r \otimes (s \otimes t) = (r \otimes s) \otimes t$  for all  $r, s, t \in S$ , and
- $\mathbb{1} \otimes s = s \otimes \mathbb{1} = s$  for every  $s \in S$ .

We say that  $\mathbb{S}$  is *commutative* if  $\otimes$  is commutative, i.e.,  $s \otimes t = t \otimes s$  for all  $s, t \in S$ . Def.

DEFINITION 1.2. A semiring is a structure  $\mathbb{S} = (S, \oplus, \odot, \mathbb{0}, \mathbb{1})$  where

- $(S, \oplus, \mathbb{0})$  is a commutative monoid,
- $(S, \odot, \mathbb{1})$  is a monoid,
- $\odot$  *distributes over*  $\oplus$ , i.e., for all  $r, s, t \in S$ ,
$$(r \oplus s) \odot t = (r \odot t) \oplus (s \odot t)$$

$$t \odot (r \oplus s) = (t \odot r) \oplus (t \odot s)$$
- $\mathbb{0}$  is an *annihilator* wrt.  $\odot$ , i.e.,  $\mathbb{0} \odot s = s \odot \mathbb{0} = \mathbb{0}$  for every  $s \in S$ .

We call  $\mathbb{S}$  *commutative* if  $\odot$  is commutative.

We call  $\mathbb{S}$  *closed* if

- $\oplus$  is *idempotent*, i.e.,  $s \oplus s = s$  for all  $s \in S$ ,
- for every countable sequence  $s_0, s_1, \dots$  of elements of  $S$ , we have that  $\bigoplus_{n \in \mathbb{N}} s_n$  is well-defined (in  $S$ ),
- associativity, commutativity, and idempotence apply to infinite sums, and
- for every two countable sequences  $s_0, s_1, \dots$  and  $t_0, t_1, \dots$  of elements of  $S$ , we have  $(\bigoplus_{n \in \mathbb{N}} s_n) \odot (\bigoplus_{n \in \mathbb{N}} t_n) = \bigoplus_{i,j \in \mathbb{N}} (s_i \odot t_j)$ . Def.

EXAMPLE 1.3. Prominent semirings are:

- $\mathbb{L}ang_\Sigma = (2^{\Sigma^*}, \cup, \cdot, \emptyset, \{\varepsilon\})$  languages over alphabet  $\Sigma$
- $\mathbb{R}eg_\Sigma = (\mathcal{R}_\Sigma, \cup, \cdot, \emptyset, \{\varepsilon\})$  regular languages over  $\Sigma$   
(i.e.,  $\mathcal{R}_\Sigma \subseteq 2^{\Sigma^*}$  is the set of regular languages over  $\Sigma$ )
- $\mathbb{T}rop = (\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$  tropical semiring
- $\mathbb{B}ool = (\{\text{false}, \text{true}\}, \vee, \wedge, \text{false}, \text{true})$  boolean algebra
- $\mathbb{N}at = (\mathbb{N}, +, \cdot, 0, 1)$  natural numbers
- $\mathbb{P}rob = (\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$  probabilistic semiring<sup>1</sup>
- $(S^{Q \times Q}, \underline{\oplus}, \underline{\odot}, \underline{0}, \underline{1})$  the semiring of  $(Q \times Q)$ -matrices with coefficients in  $\mathbb{S}$  (for some semiring  $\mathbb{S} = (S, \oplus, \odot, 0, 1)$  and some set  $Q$ ). For all  $p, q \in Q$ ,  $\underline{0}_{p,q} = 0$ ,  $\underline{1}_{p,p} = 1$ , and  $\underline{1}_{p,q} = 0$  if  $p \neq q$ . The operations  $\underline{\oplus}$  and  $\underline{\odot}$  are the usual matrix addition and multiplication, i.e., given  $r, s \in S^{Q \times Q}$ ,  $(r \underline{\oplus} s)_{p,q} = r_{p,q} \oplus s_{p,q}$  and  $(r \underline{\odot} s)_{p,q} = \bigoplus_{\hat{q} \in Q} r_{p,\hat{q}} \odot s_{\hat{q},q}$ .

For a closed semiring, we can define a closure operator:

DEFINITION 1.4. Let  $\mathbb{S} = (S, \oplus, \odot, 0, 1)$  be a closed semiring and  $s \in S$ . The closure of  $s$  is the infinite sum

$$s^* := s^0 \oplus s^1 \oplus s^2 \oplus \dots$$

where  $s^0 = 1$ . Def.

Revisiting our initial models, we realize that we actually deal with one and the same semantics. Every instance, however, is solved by a computation in a specific closed semiring:

semantics of NFA	$\mathbb{L}ang_\Sigma$
shortest path	$\mathbb{T}rop$
closure	$\mathbb{B}ool$

Let us give a general formalization of our problem.

DEFINITION 1.5. Let  $\mathbb{S} = (S, \oplus, \odot, 0, 1)$  be a semiring. A weighted system over  $\mathbb{S}$  is a pair  $\mathfrak{A} = (Q, \mu)$  where  $Q$  is a nonempty finite set of *states* and  $\mu$  is the *weight function*  $Q \times Q \rightarrow S$ . We call  $\mathfrak{A}$  *closed* if  $\mathbb{S}$  is closed. Def.

Let  $\mathfrak{A} = (Q, \mu)$  be a **closed** weighted system over a (closed) semiring  $\mathbb{S}$ .

A *path* of  $\mathfrak{A}$  is a nonempty finite sequence  $\rho = (q_0, \dots, q_m)$  of states. Hereby,  $m$  is the *length* of  $\rho$ . We extend  $\mu$  to paths as follows:

$$\mu(\rho) := \begin{cases} 1 & \text{if } m = 0 \\ \mu(q_0, q_1) \odot \mu(q_1, q_2) \odot \dots \odot \mu(q_{m-1}, q_m) & \text{if } m \geq 1 \end{cases}$$

<sup>1</sup>Note that  $([0, 1], \max, \cdot, 0, 1)$  is sometimes considered as the probabilistic semiring as its universe restricts to probabilities. It is, however, not suitable for our purposes, as it neglects addition and, thus, does not allow one to model non-determinism.

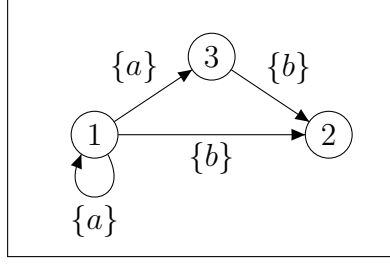


FIGURE 2. Example 1.6

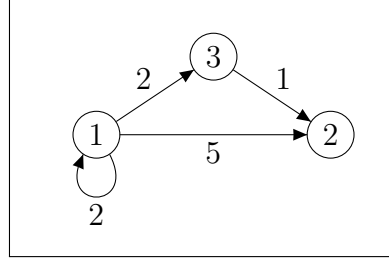


FIGURE 3. Example 1.7

Furthermore, we extend  $\mu$  to sets  $P$  of paths:

$$\mu(P) := \begin{cases} 0 & \text{if } P = \emptyset \\ \bigoplus_{\rho \in P} \mu(\rho) & \text{if } P \neq \emptyset \end{cases}$$

For  $p, q \in Q$ , we denote by  $P_{p,q}$  the set of paths  $(q_0, \dots, q_m)$  with  $q_0 = p$  and  $q_m = q$ . Thus, we are interested in computing  $\mu(P_{p,q})$  for any  $p, q \in Q$ . So, we let

$$\|\mathfrak{A}\| : \begin{cases} Q \times Q \rightarrow S \\ (p, q) \mapsto \mu(P_{p,q}) \end{cases}$$

EXAMPLE 1.6. Let  $\Sigma = \{a, b\}$  and consider Figure 1.6, depicting a closed weighted system  $\mathfrak{A} = (Q, \mu)$  over  $\text{Reg}_\Sigma$  (i.e., an NFA without initial and final states). A labeled edge  $(p, s, q) \in Q \times S \times Q$  indicates that  $\mu(p, q) = s \neq \emptyset$ . Edges  $(p, s, q)$  with  $\mu(p, q) = s = \emptyset$  are omitted. The weight function  $\mu$  and its semantics  $\|\mathfrak{A}\|$  are given as follows:

$\mu$	1	2	3
1	{a}	{b}	{a}
2	$\emptyset$	$\emptyset$	$\emptyset$
3	$\emptyset$	{b}	$\emptyset$

$\ \mathfrak{A}\ $	1	2	3
1	$a^*$	$a^*b$	$a^*a$
2	$\emptyset$	$\emptyset$	$\emptyset$
3	$\emptyset$	$b$	$\emptyset$

EXAMPLE 1.7. Consider Figure 1.7, depicting a closed weighted system  $\mathfrak{A} = (Q, \mu)$  over  $\mathbb{Trop}$  (i.e., a weighted directed graph). We have:

$\mu$	1	2	3
1	2	5	2
2	$\infty$	$\infty$	$\infty$
3	$\infty$	1	$\infty$

$\ \mathfrak{A}\ $	1	2	3
1	0	3	2
2	$\infty$	0	$\infty$
3	$\infty$	1	0

**Floyd-Warshall algorithm.** Let  $\mathfrak{A} = (Q, \mu)$  be a closed weighted system over a closed semiring  $\mathbb{S} = (S, \oplus, \odot, 0, 1)$ .

To compute  $\|\mathfrak{A}\|$ , we follow our above scheme and assume  $Q = \{1, \dots, n\}$ . For  $i, j \in Q$  and  $k \in \{1, \dots, n\}$ , let

$$P_{i,j}^{(k)} \text{ contain the paths } (i_0, \dots, i_m) \in P_{i,j} \text{ with } 1 \leq i_1, \dots, i_{m-1} \leq k$$

$P_{i,j}^{(0)}$  contain the paths from  $P_{i,j}$  of the form  $(i_0)$  or  $(i_0, i_1)$

In particular,  $P_{ij}^{(n)} = P_{ij}$ . We would like to compute  $\|\mathfrak{A}\|(i, j) = \mu(P_{i,j})$ , for all  $i, j \in Q$ . In the following, we use  $h_{i,j}^{(k)} = \mu(P_{i,j}^{(k)})$  as an abbreviation.

From the definitions, we directly deduce

$$h_{i,j}^{(0)} = \begin{cases} \mu(i, j) \oplus \mathbb{1} & \text{if } i = j \\ \mu(i, j) & \text{if } i \neq j \end{cases}$$

For  $k \geq 1$ , we have:

$$\begin{aligned} h_{i,j}^{(k)} = \mu(P_{i,j}^{(k)}) &= \bigoplus_{\rho \in P_{i,j}^{(k)}} \mu(\rho) \\ &= \underbrace{\bigoplus_{\rho \in P_{i,j}^{(k-1)}} \mu(\rho)}_{= h_{i,j}^{(k-1)}} \oplus \underbrace{\bigoplus_{\rho \in P_{i,j}^{(k)} \setminus P_{i,j}^{(k-1)}} \mu(\rho)}_U \end{aligned}$$

$$\begin{aligned} U &= \bigoplus_{\substack{\rho_1 \in P_{i,k}^{(k-1)} \\ \rho_2 \in P_{k,k}^{(k)} \\ \rho_3 \in P_{k,j}^{(k-1)}}} \mu(\rho_1) \odot \mu(\rho_2) \odot \mu(\rho_3) \\ &= \underbrace{\bigoplus_{\rho_1 \in P_{i,k}^{(k-1)}} \mu(\rho_1)}_{= h_{i,k}^{(k-1)}} \odot \underbrace{\bigoplus_{\rho_2 \in P_{k,k}^{(k)}} \mu(\rho_2)}_V \odot \underbrace{\bigoplus_{\rho_3 \in P_{k,j}^{(k-1)}} \mu(\rho_3)}_{= h_{k,j}^{(k-1)}} \end{aligned}$$

$$\begin{aligned} V &= \mathbb{1} \oplus \bigoplus_{\ell \geq 1} \bigoplus_{\substack{\rho_1, \dots, \rho_\ell \\ \in P_{k,k}^{(k-1)}}} \mu(\rho_1) \odot \dots \odot \mu(\rho_\ell) \\ &= \mathbb{1} \oplus \bigoplus_{\ell \geq 1} \left( \underbrace{\bigoplus_{\rho \in P_{k,k}^{(k-1)}} \mu(\rho)}_{h_{k,k}^{(k-1)}} \right)^\ell \\ &= (h_{k,k}^{(k-1)})^* \end{aligned}$$

Altogether, we obtain, for  $k \in \mathbb{N}$ :

$$h_{i,j}^{(k)} = \begin{cases} \mu(i, j) \oplus \mathbb{1} & \text{if } i = j \text{ and } k = 0 \\ \mu(i, j) & \text{if } i \neq j \text{ and } k = 0 \\ h_{i,j}^{(k-1)} \oplus (h_{i,k}^{(k-1)} \odot (h_{k,k}^{(k-1)})^* \odot h_{k,j}^{(k-1)}) & \text{if } k \geq 1 \end{cases}$$

The procedure that is suggested by these equations to compute  $\|\mathfrak{A}\|$  is called Floyd-Warshall algorithm.

**The matrix approach.** Again, let  $\mathbb{S} = (S, \oplus, \odot, 0, \mathbb{1})$  be a **closed** semiring and  $\mathfrak{A} = (Q, \mu)$  be a closed weighted system over  $\mathbb{S}$ . To determine  $\|\mathfrak{A}\|$ , we can work with matrix operations in the closed(!) semiring  $(S^{Q \times Q}, \oplus, \odot, \mathbb{0}, \mathbb{1})$  of  $(Q \times Q)$ -matrices with coefficients in  $\mathbb{S}$ . Note that both  $\|A\|$  and  $\mu$  can be considered as  $(Q \times Q)$ -matrices with entries  $\|A\|_{p,q} = \|A\|(p, q)$  and  $\mu_{p,q} = \mu(p, q)$ , respectively.

**THEOREM 1.8.** *Let  $\mathfrak{A} = (Q, \mu)$  be a closed weighted system. We have that*

$$\|\mathfrak{A}\| = (\mu)^*.$$

**PROOF.** Exercise. □

**Markov chains.** A weighted system that is not closed does not always have this natural semantics. Consider the following classical definition:

**DEFINITION 1.9.** A *discrete-time Markov chain* is a weighted system  $(Q, \mu)$  over  $\mathbb{Prob}$  such that, for all  $p \in Q$ ,  $\sum_{q \in Q} \mu(p, q) \in \{0, 1\}$ . Def.

In the case of a discrete-time Markov chain, however, the matrix  $(\mu)^n$  contains the probabilities of going from  $p$  to  $q$  in exactly  $n$  steps. (Cf. follow-up lecture given by Serge Haddad.)

### 1.3. Weighted automata

In this section,  $\mathbb{S}$  will be a semiring  $(S, \oplus, \odot, 0, \mathbb{1})$ . Let  $\Sigma$  be an alphabet, i.e., a nonempty finite set. A formal power series over  $\Sigma$  and  $\mathbb{S}$  is a mapping  $\Sigma^* \rightarrow S$  (we might also write  $\Sigma^* \rightarrow \mathbb{S}$ ). The set of those formal power series is denoted by  $\mathbb{S}\langle\langle \Sigma^* \rangle\rangle$ .

**DEFINITION 1.10.** A weighted automaton over the semiring  $\mathbb{S}$  is a structure  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  where

- $Q$  is the nonempty finite set of *states*,
- $\Sigma$  is the *input alphabet*,
- $\mu : Q \times \Sigma \times Q \rightarrow S$  is the *transition-weight function*,
- $\lambda : Q \rightarrow S$  is the *initial-weight function*, and
- $\gamma : Q \rightarrow S$  is the *final-weight function*. Def.

Let  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  be a weighted automaton over  $\mathbb{S}$ . The semantics of  $\mathfrak{A}$  is a formal power series  $\|\mathfrak{A}\| \in \mathbb{S}\langle\langle \Sigma^* \rangle\rangle$ . To define it, we extend  $\mu$  towards paths and, afterwards, to sets of paths. The weight of a word  $w \in \Sigma^*$  is then the sum over the weights of all paths that are labeled with  $w$ .

A *path* of  $\mathfrak{A}$  is an alternating sequence  $\rho = (q_0, a_1, q_1, \dots, a_n, q_n)$ ,  $n \in \mathbb{N}$ , of states  $q_i \in Q$  and letters  $a_i \in \Sigma$ . We call  $w = a_1 \dots a_n$  the *labeling* of  $\rho$ . The weight of  $\rho$  is

$$\mu(\rho) := \lambda(q_0) \odot \mu(q_0, a_1, q_1) \odot \dots \odot \mu(q_{n-1}, a_n, q_n) \odot \gamma(q_n).$$

For a set  $P$  of paths, we let

$$\mu(P) := \begin{cases} \emptyset & \text{if } P = \emptyset \\ \bigoplus_{\rho \in P} \mu(\rho) & \text{if } P \neq \emptyset \end{cases}$$

For  $w = a_1 \dots a_n \in \Sigma^*$ , let  $P_w$  denote the set of all paths with labeling  $w$ . Then, we set

$$\|\mathfrak{A}\|(w) := \mu(P_w).$$

One can easily verify that

$$\|\mathfrak{A}\|(w) = \lambda \underline{\odot} \mu(a_1) \underline{\odot} \dots \underline{\odot} \mu(a_n) \underline{\odot} \gamma$$

where  $\mu(a_i)$  is the  $(Q \times Q)$ -matrix given by  $\mu(a)_{p,q} = \mu(p, a, q)$ ,  $\lambda$  is considered as a row-, and  $\gamma$  as a column vector.

In the following, we will demonstrate that weighted automata are a generic model that subsumes many important automata classes.

**Finite automata.** A (non-deterministic) finite automaton is simply a weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\mathbb{B}\text{ool}$  such that  $\lambda^{-1}(\text{true})$  is a singleton set (containing the unique initial state). The semantics of  $\mathfrak{A}$  is a mapping  $\|\mathfrak{A}\| : \Sigma^* \rightarrow \{\text{true}, \text{false}\}$  where **true** signals “accepted” and **false** “rejected”. Usually, finite automata come with a set of final states  $F \subseteq Q$  and a transition relation  $\Delta \subseteq Q \times \Sigma \times Q$ , which can be recovered from  $\mathfrak{A}$  by letting  $F = \gamma^{-1}(\text{true})$  and  $\Delta = \mu^{-1}(\text{true})$ .

**Probabilistic automata.** A *probabilistic automaton* is a weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\mathbb{P}\text{rob} = (\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$  such that

- (1) there is  $p \in Q$  such that  $\lambda(p) = 1$  and, for all  $q \in Q \setminus \{p\}$ ,  $\lambda(q) = 0$ ,
- (2) for all  $p \in Q$ ,  $\gamma(p) \in \{0, 1\}$ , and
- (3) for all  $p \in Q$  and  $a \in \Sigma$ , we have  $\sum_{q \in Q} \mu(p, a, q) = 1$ .

In that model,  $\|\mathfrak{A}\|(w)$  can be interpreted as the probability of reaching a final state when  $w$  is used as a *scheduling policy*.

**Generative probabilistic automata.** A *generative probabilistic automaton* is defined like a probabilistic automaton, apart from condition (3), which is replaced with

$$(3') \text{ for every } p \in Q, \text{ we have } \sum_{(a,q) \in \Sigma \times Q} \mu(p, a, q) = 1.$$

In that model,  $\|\mathfrak{A}\|(w)$  can be considered as the probability of executing  $w$  and ending in a final state, under the precondition that we perform  $|w|$  steps.

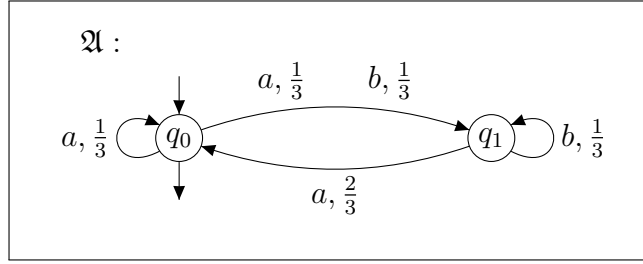


FIGURE 4. A generative probabilistic automaton

EXAMPLE 1.11. Figure 4 depicts a generative probabilistic automaton  $\mathfrak{A}$  over  $\{a, b\}$ . We have  $\|\mathfrak{A}\|(a) = \|\mathfrak{A}\|(aa) = \frac{1}{3}$  and  $\|\mathfrak{A}\|(aaa) = \frac{5}{27}$ . Moreover,  $\|\mathfrak{A}\|(w) = 0$  whenever  $w$  ends with the letter  $b$ . Exm.

**Word transducers.** Let  $\Gamma$  be an alphabet. A word transducer over  $\Gamma$  is a weighted automaton over  $\mathbb{R}\text{eg}_\Gamma = (\mathcal{R}_\Gamma, \cup, \cdot, \emptyset, \{\varepsilon\})$ .

#### 1.4. Threshold Languages and Decision Problems

In classical automata theory, one often raises the question if a given automaton exhibits *some* behavior. Regarding a weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over a semiring  $\mathbb{S} = (S, \oplus, \odot, 0, \mathbb{1})$ , this corresponds to asking if there is some  $w \in \Sigma^*$  such that  $\|\mathfrak{A}\|(w) \neq 0$ .

Let  $\mathcal{C}$  be a class of weighted automata over  $\mathbb{S}$ . The *emptiness problem* for  $\mathcal{C}$  is given as follows:

INPUT: Weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma) \in \mathcal{C}$ .  
 PROBLEM: Do we have  $\|\mathfrak{A}\|(w) \neq 0$  for some  $w \in \Sigma^*$ ?

Under some assumption (e.g., for computable fields) this problem is decidable [1]. The emptiness problem can be refined when the semiring comes with an ordering, which, given a formal power series, allows us to classify words according to a threshold. When we have a (generative) probabilistic automaton, for example, we might be interested in the set of words that are accepted with a probability greater than some  $\theta \in [0, 1]$ . Or, given a word transducer, we might want to compute the set of words that generate sets that subsume a given regular language. In the subsequent chapters, we

will see that both questions are undecidable. Let us give a general formal definition of this problem.

**DEFINITION 1.12.** Let  $\mathbb{S} = (S, \oplus, \odot, \mathbb{0}, \mathbb{1})$  be a semiring,  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  be a weighted automaton over  $\mathbb{S}$ ,  $\bowtie \subseteq S \times S$  be a binary relation, and  $\theta \in S$ . The threshold language of  $\mathfrak{A}$  wrt.  $\bowtie$  and  $\theta$  is given as follows:

$$L_{\theta \bowtie}(\mathfrak{A}) := \{w \in \Sigma^* \mid \theta \bowtie \|\mathfrak{A}\|(w)\}$$

Def.

Two natural, related questions arise in the context of threshold languages:

- (1) Is a threshold language (effectively) regular?
- (2) If not, can we decide if a threshold language is empty/universal?

These questions will be studied in Chapters 2 and 3 for word transducers and probabilistic automata.

Let us be more precise. For a semiring  $\mathbb{S} = (S, \oplus, \odot, \mathbb{0}, \mathbb{1})$ , a relation  $\bowtie \subseteq S \times S$ , and a class  $\mathcal{C}$  of weighted automata over  $\mathbb{S}$ , consider the following two problems:

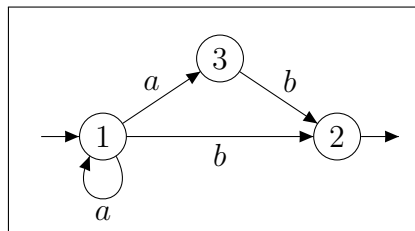
- INPUT: Weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma) \in \mathcal{C}$  and  $\theta \in S$ .
- PROBLEM 1: Do we have  $L_{\theta \bowtie}(\mathfrak{A}) \neq \emptyset$ ?
- PROBLEM 2: Do we have  $L_{\theta \bowtie}(\mathfrak{A}) = \Sigma^*$ ?

These problems are respectively called the

- (1) *threshold emptiness problem* for  $\mathcal{C}$  wrt.  $\bowtie$
- (2) *threshold universality problem* for  $\mathcal{C}$  wrt.  $\bowtie$

### Exercises for Chapter 1

**EXERCISE 1.13.** Let the NFA  $\mathfrak{A}$  with alphabet  $\Sigma = \{a, b\}$  be given as follows:



Using the Floyd-Warshall method, specify a regular expression(!) that is equivalent to  $\mathfrak{A}$ .

Ex.

**EXERCISE 1.14.** Which of the semirings from Example 1.3 is commutative, which of them is closed?

Ex.

**EXERCISE 1.15.** Prove Theorem 1.8

Ex.

EXERCISE 1.16. Apply the matrix approach to Example 1.6.

Ex.

EXERCISE 1.17. Apply the matrix approach to Example 1.7.

Ex.

### Further Reading

- [1] J. Berstel and C. Reutenauer. *Rational series and their languages*. Springer, 1988.
- [2] Th. H. Cormen, Ch. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, September 2009.
- [3] W. Kuich and A. Salomaa. *Semirings, Automata and Languages*. Springer, 1985.
- [4] M. Mohri. Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages, and Combinatorics*, 7(3):321–350, 2002.



## CHAPTER 2

### Word Transducers

#### 2.1. Definition

Word transducers have manifold applications in computer science, e.g., in regular model checking [1], speech recognition, natural language processing [5], etc.

**DEFINITION 2.1.** Let  $\Gamma$  be an alphabet. A word transducer over  $\Gamma$  is a weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\mathbb{R}\text{eg}_\Gamma = (\mathcal{R}_\Gamma, \cup, \cdot, \emptyset, \{\varepsilon\})$ . □<sub>Def.</sub>

Thus,  $\|\mathfrak{A}\|$  assigns to a word  $w \in \Sigma^*$  a language  $\|\mathfrak{A}\|(w) \subseteq \Gamma^*$ . Usually,  $\|\mathfrak{A}\|$  is represented by the binary relation

$$R(\mathfrak{A}) := \{(u, v) \in \Sigma^* \times \Gamma^* \mid v \in \|\mathfrak{A}\|(u)\}.$$

From that point of view, word transducers describe precisely the *rational relations* (cf. [3, 4]).

**EXAMPLE 2.2.** Let  $\Sigma = \{a, b\}$ . A word transducer  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\Sigma$  is depicted in Figure 1, where  $\lambda(q_0) = \gamma(q_0) = \Sigma^* \in \mathcal{R}_\Sigma$ . For all  $u \in \Sigma^*$ ,  $\|\mathfrak{A}\|(u)$  is the set of words  $v \in \Sigma^*$  such that  $u$  is an infix of  $v$ .

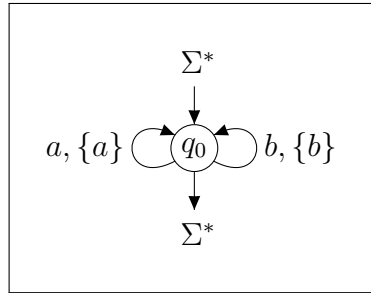


FIGURE 1. A word transducer

#### 2.2. Threshold Problems for Word Transducers

Word transducers can be used to model systems that communicate with an environment. Suppose a system can execute actions from an alphabet  $\Sigma$  that are triggered by input streams from  $\Gamma^*$  where  $\Gamma$  is a set of signals (cf. Figure 2). This can be modeled by a word transducer  $\mathfrak{A}$  over  $\Gamma$  with input alphabet  $\Sigma$ . For a sequence  $w \in \Sigma^*$  of actions,  $\|\mathfrak{A}\|(w)$  contains the sequences

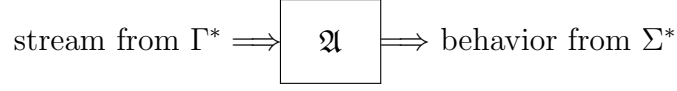


FIGURE 2. A system communicating with its environment

that trigger the behavior  $w$ . Often, the input stream cannot be predicted in advance but only isolated by a, say, regular set  $E \subseteq \Gamma^*$ . Thus, it can be useful to know the set of words  $w \in \Sigma^*$  such that  $E \cap \|\mathfrak{A}\|(w) \neq \emptyset$ . This corresponds to the set  $L_{E \cap_{\neq \emptyset}}(\mathfrak{A})$  where  $\cap_{\neq \emptyset}$  is the set of pairs  $(M_1, M_2)$  with nonempty intersection.

Suppose we are given a set  $Bad \subseteq \Sigma^*$  of bad behaviors that our system must avoid. Then, we would like to have

$$Bad \cap L_{E \cap_{\neq \emptyset}}(\mathfrak{A}) = \emptyset.$$

If, on the other hand, we are given a set  $Good \subseteq \Sigma^*$  of behaviors that the system should exhibit, then we need a statement such as

$$Good \subseteq L_{E \subseteq}(\mathfrak{A}).$$

The first problem, where  $\mathfrak{A}$ ,  $E$ , and  $Bad$  act as the input, is the *model checking problem wrt. safety properties*. The second one, where in the input  $Bad$  is replaced with  $Good$ , is the *model checking problem wrt. liveness properties*. These considerations motivate us to study the threshold emptiness and universality problems for word transducers.

The following result follows from a standard automaton construction.

**THEOREM 2.3.** *Let  $\Gamma$  be an alphabet and let  $\mathfrak{A}$  be a word transducer over  $\Gamma$ . For every regular language  $E \subseteq \Gamma^*$ , the threshold language  $L_{E \cap_{\neq \emptyset}}(\mathfrak{A})$  is effectively regular.*

**PROOF.** Suppose  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  and let  $E$  be represented by the finite automaton  $\mathfrak{B} = (P, \Gamma, \Delta, p_0, F)$ . We construct the finite automaton  $\mathfrak{B}' = (P', \Sigma, \Delta', P'_0, F')$  (here,  $P'_0 \subseteq P'$  is a set of initial states) with  $L(\mathfrak{B}') = L_{E \cap_{\neq \emptyset}}(\mathfrak{A})$  as follows:

- $P' = Q \times P$
- $P'_0 = \{(q, p) \in P' \mid p \in \delta(p_0, \lambda(q))\}$
- $F' = \{(q, p) \in Q \times P \mid \delta(p, \gamma(q)) \cap F \neq \emptyset\}$
- $((q, p), a, (q', p')) \in \Delta'$  if  $p' \in \delta(p, \mu(q, a, q'))$

Hereby, for  $p \in P$  and a regular set  $L \subseteq \Gamma^*$ ,  $\delta(p, L)$  denotes the (computable) set of states of  $\mathfrak{B}$  that are reachable from  $p$  by reading some word from  $L$ .  $\square$

**COROLLARY 2.4.** *For word transducers, both the threshold emptiness problem and the threshold universality problem wrt.  $\cap_{\neq \emptyset}$  are decidable.*

We also conclude that, for word transducers over  $\Gamma$ , the model checking problem wrt. safety properties is decidable.

Unfortunately, both the threshold emptiness and the threshold universality problem become undecidable when we consider the binary relation  $\subseteq$ . We will first show the result for the universality problem.

**THEOREM 2.5.** *Let  $\Gamma$  be an alphabet with at least two letters. Then, the threshold universality problem for word transducers over  $\Gamma$  wrt.  $\subseteq$  is undecidable.*

We will actually show undecidability of a concrete instance of that problem:

**THEOREM 2.6.** *Let  $\Gamma$  be an alphabet with at least two letters. The following problem is undecidable:*

**INPUT:** Word transducer  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\Gamma$ .  
**PROBLEM:** Do we have  $\|\mathfrak{A}\|(w) = \Gamma^*$  for every  $w \in \Sigma^*$ ?

**PROOF.** The proof is by reduction from Post's correspondence problem (PCP), which is given as follow:

**INPUT:** Alphabet  $A$  and morphisms  $f, g : A^* \rightarrow \{0, 1\}^*$ .

**PROBLEM:** Is there  $w \in A^+$  such that  $f(w) = g(w)$ ?

Let an instance of the PCP be given by  $A$  and  $f, g$ . Suppose  $\Gamma = \{0, 1\}$ . Our reduction is based on two binary relations  $R_f, R_g \subseteq A^+ \times \Gamma^*$ :

$$R_f = \{(w, f(w)) \mid w \in A^+\}$$

$$R_g = \{(w, g(w)) \mid w \in A^+\}$$

We can construct a word transducer  $\mathfrak{A} = (Q, A, \lambda, \mu, \gamma)$  over  $\Gamma$  such that

$$R(\mathfrak{A}) = \overline{R_f} \cup \overline{R_g}$$

where, for a relation  $R \subseteq A^* \times \Gamma^*$ , we let  $\overline{R} = (A^* \times \Gamma^*) \setminus R$ . We have

$$\begin{aligned} \|\mathfrak{A}\|(w) = \Gamma^* & \text{ for every } w \in \Sigma^* \\ \text{iff } R(\mathfrak{A}) = A^* \times \Gamma^* & \\ \text{iff there is no } w \in A^+ & \text{ such that } f(w) = g(w) \end{aligned}$$

The construction of  $\mathfrak{A}$  is left as an exercise (Exercise 2.10). This concludes the proof of Theorem 2.6. □

From that result, we deduce that, for word transducers over  $\Gamma$ , the model checking problem wrt. liveness properties is undecidable.

**THEOREM 2.7.** *Let  $\Gamma$  be an alphabet with at least two letters. Then, the threshold emptiness problem for word transducers over  $\Gamma$  wrt.  $\subseteq$  is undecidable.*

Again, we will show undecidability of a more specific problem:

**THEOREM 2.8.** *Let  $\Gamma$  be an alphabet with at least two letters. The following problem is undecidable:*

**INPUT:** Word transducer  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\Gamma$ .  
**PROBLEM:** Do we have  $\|\mathfrak{A}\|(w) = \Gamma^*$  for some  $w \in \Sigma^*$ ?

**PROOF.** Again, the proof is by reduction from the PCP. It is inspired by a result from a timed setting [2].

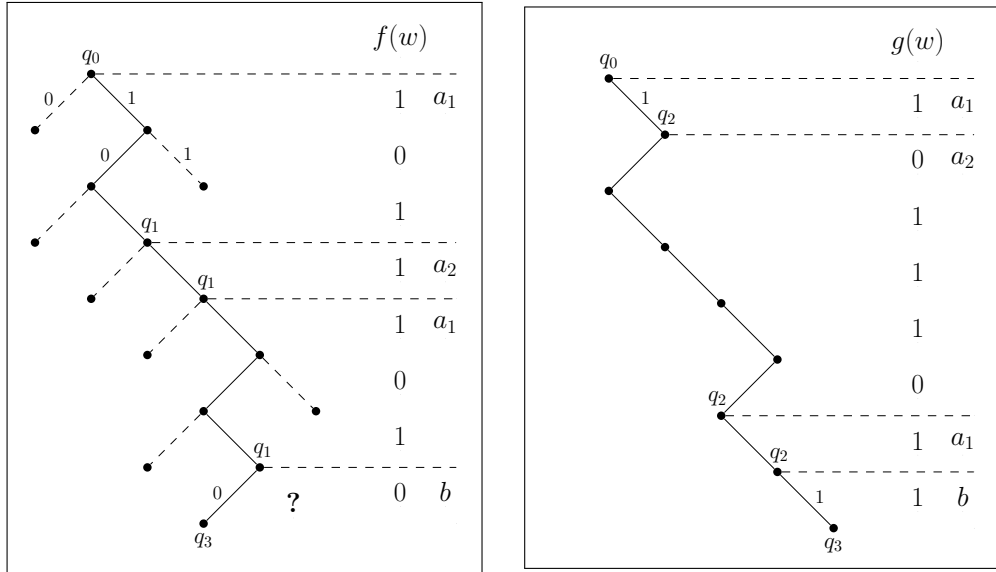


FIGURE 3. The trees generated by  $w = a_1a_2a_1b$

From a PCP instance  $A, f, g$ , we will construct a word transducer  $\mathfrak{A}$  over  $\Gamma = \{0, 1\}$  such that  $L_{\Gamma^* \subseteq}(\mathfrak{A}) = \{u.b \mid u \in A^+ \text{ and } f(u) = g(u)\}$ . The idea of our construction is illustrated in Figure 3. Consider the PCP instance given by  $A = \{a_1, a_2\}$  and

$$\begin{aligned} f(a_1) &= 101 & f(a_2) &= 1 \\ g(a_1) &= 1 & g(a_2) &= 01110 \end{aligned}$$

with the obvious solution  $u = a_1a_2a_1$ . Our transducer will be split into two parts. One part is concerned with  $f$  (cf. the left hand side of the figure). Intuitively, it reads a sequence  $u.b$  with  $u \in A^+$  and generates a tree whose nodes correspond to words over  $\Gamma$ . A node (or, the path to reach the node from the root) will be accepted unless it corresponds to  $f(u)$ . In the latter case, the transducer remains in a state  $q_1$ . When reading  $b$  in  $q_1$ , our transducer will produce 0, and will accept. However, the sequence  $u.b$  is in  $L_{\Gamma^* \subseteq}(\mathfrak{A})$  only if a path labeled  $f(u).1$  is accepted, too. Such a path can be provided by the second part of the automaton, simulating  $g$  (cf. the right hand side of the figure). The missing path needs to follow  $f(u)$  and produce

the letter 1 on that node where the  $f$ -part had produced 0. For this,  $f(u)$  and  $g(u)$  have to coincide, which implies that the PCP instance has a solution.

Let an instance of the PCP be given by the alphabet  $A = \{a_1, \dots, a_k\}$  with  $k \geq 1$  and two corresponding morphisms  $f$  and  $g$ . We will construct a word transducer  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\Gamma = \{0, 1\}$ , with  $\Sigma = \{a_1, \dots, a_k, b\}$ , such that

$$L_{\Gamma^* \subseteq}(\mathfrak{A}) = \{u.b \mid u \in A^+ \text{ and } f(u) = g(u)\}.$$

The automaton  $\mathfrak{A}$  is given by Figure 4. Hereby, for  $i \in \{1, \dots, k\}$ , we let

$$\begin{aligned} F_i &= \{f(a_i)\} & \bar{F}_i &= \{v \in \Gamma^* \mid f(a_i) \notin \text{Pref}(v)\} \\ G_i &= \{g(a_i)\} \end{aligned}$$

where  $\text{Pref}(v) = \{v_1 \in \Gamma^* \mid v = v_1.v_2 \text{ for some } v_2 \in \Gamma^*\}$  is the set of *prefixes* of  $v$ .

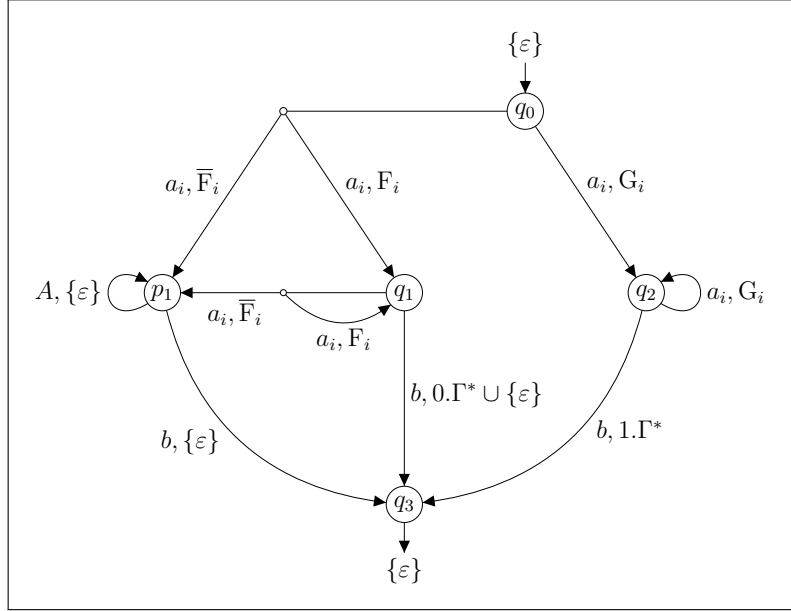


FIGURE 4. Encoding of PCP in terms of a word transducer

For a state  $q \in Q$ , let in the following  $\mathfrak{A}_q$  refer to the word transducer  $(Q, \Sigma, \lambda, \mu, \gamma')$  where, for  $p \in Q$ ,

$$\gamma'(p) = \begin{cases} \{\varepsilon\} & \text{if } p = q \\ \emptyset & \text{if } p \neq q \end{cases}$$

CLAIM 2.9. For every  $u \in A^+$ , the following hold:

- (1)  $\|\mathfrak{A}_{q_1}\|(u) = \{f(u)\}$
- (2)  $\|\mathfrak{A}_{q_2}\|(u) = \{g(u)\}$
- (3)  $\|\mathfrak{A}_{p_1}\|(u) = \{v \in \Gamma^* \mid f(u) \notin \text{Pref}(v)\}$

We can now show that  $L_{\Gamma^* \subseteq}(\mathfrak{A}) = \{u.b \mid u \in A^+ \text{ and } f(u) = g(u)\}$ .

Let  $u \in A^+$  with  $f(u) = g(u)$  and let  $v \in \Gamma^*$ . To prove that  $v \in \|\mathfrak{A}\|(u.b)$ , we distinguish three cases:

- (1) If  $v = f(u)$  or  $v \in f(u).0.\Gamma^*$ , then  $v \in \|\mathfrak{A}\|(u.b)$  by Claim 2.9 (1).
- (2) If  $v \in f(u).1.\Gamma^*$ , then  $v \in \|\mathfrak{A}\|(u.b)$  by Claim 2.9 (2).
- (3) If  $v \notin f(u).\Gamma^*$ , then  $v \in \|\mathfrak{A}\|(u.b)$  by Claim 2.9 (3).

Thus, we have  $v \in \|\mathfrak{A}\|(u.b)$ .

Conversely, let  $u \in A^+$  and suppose  $\|\mathfrak{A}\|(u.b) = \Gamma^*$ . We have  $f(u) \in \|\mathfrak{A}_{q_1}\|(u)$  and  $f(u) \notin \|\mathfrak{A}_{p_1}\|(u)$  (Claims 2.9 (1) and (3)). As  $f(u).1 \in \|\mathfrak{A}\|(u.b)$ , we must have  $f(u) \in \|\mathfrak{A}_{q_2}\|(u)$ . By Claim 2.9 (2), we deduce  $f(u) = g(u)$ .  $\square$

## Exercises for Chapter 2

EXERCISE 2.10. Construct the word transducer  $\mathfrak{A}$  over  $\Gamma = \{0, 1\}$  from the proof of Theorem 2.6. Recall that  $\mathfrak{A}$  should satisfy  $R(\mathfrak{A}) = \overline{R_f} \cup \overline{R_g}$ . Note that  $\overline{R_f}$  (similarly for  $\overline{R_g}$ ) is the union of the following relations:

$$\begin{aligned} R_0 &= \{(u, v) \in A^* \times \Gamma^* \mid u = \varepsilon\} \\ R_1 &= \{(u, v) \in A^+ \times \Gamma^* \mid |v| < |f(u)|\} \\ R_2 &= \{(u, v) \in A^+ \times \Gamma^* \mid |v| > |f(u)|\} \\ R_3 &= \{(u, v) \in A^+ \times \Gamma^* \mid |v| = |f(u)| \text{ and } v \neq f(u)\} \end{aligned}$$

The respective word transducers can then be combined towards  $\mathfrak{A}$ .  $\square$  Ex.

EXERCISE 2.11. Show Theorem 2.6 by means of a variant of the proof of Theorem 2.8.  $\square$  Ex.

EXERCISE 2.12. Show Claim 2.9 (3).  $\square$  Ex.

## Further Reading

- [1] P. Aziz Abdulla, B. Jonsson, M. Nilsson, and M. Saksena. A survey of regular model checking. In *Proceedings of CONCUR'04*, volume 3170 of *Lecture Notes in Computer Science*, pages 35–48. Springer, 2004.
- [2] S. Akshay, B. Bollig, P. Gastin, M. Mukund, and K. Narayan Kumar. Distributed timed automata with independently evolving clocks. In *Proceedings of the 19th International Conference on Concurrency Theory (CONCUR'08)*, volume 5201 of *Lecture Notes in Computer Science*, pages 82–97. Springer, 2008.
- [3] C. Choffrut. Rational relations as rational series. In *Theory Is Forever, Essays Dedicated to Arto Salomaa on the Occasion of His 70th Birthday*, volume 3113 of *Lecture Notes in Computer Science*, pages 29–34. Springer, 2004.
- [4] C. Choffrut. A short introductory course to rational relations, 2009. Lecture notes, <http://www.liafa.jussieu.fr/~cc/DEA/Introduction-MPRI.pdf>.
- [5] M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.

## CHAPTER 3

# Probabilistic Automata and Stochastic Languages

### 3.1. Definition

Remember that  $\mathbb{P}\text{rob}$  denotes the probabilistic semiring  $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$ . Let us recall the definition of probabilistic automata from Chapter 1.

**DEFINITION 3.1.** A probabilistic automaton<sup>1</sup> is a weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\mathbb{P}\text{rob}$  such that

- (1) there is  $p \in Q$  such that  $\lambda(p) = 1$  and, for all  $q \in Q \setminus \{p\}$ ,  $\lambda(q) = 0$ ,
- (2) for all  $q \in Q$ ,  $\gamma(q) \in \{0, 1\}$ , and
- (3) for every  $p \in Q$  and  $a \in \Sigma$ , we have  $\sum_{q \in Q} \mu(p, a, q) = 1$ .

The unique state  $p$  with  $\lambda(p) = 1$  can be considered to be an initial state, and those states  $q$  with  $\gamma(q) = 1$  are the final states. Indeed, the weight  $\|\mathfrak{A}\|(w)$  of a word  $w \in \Sigma^*$  can be seen as its probability of acceptance. More precisely,  $\|\mathfrak{A}\|(w)$  can be interpreted as the probability of reaching a final state when  $w$  is used as a scheduling policy. We have  $\|\mathfrak{A}\|(\varepsilon) \in \{0, 1\}$ . Note that replacing (3) in Definition 3.1 with

$$(3') \text{ for every } p \in Q \text{ and } a \in \Sigma, \text{ we have } \sum_{q \in Q} \mu(p, a, q) \in \{0, 1\}.$$

is not more general, as one can always introduce a non-accepting sink state.

In this section, it will be useful to consider transition matrices rather than one transition function. Recall that, for  $w = a_1 \dots a_n \in \Sigma^*$ ,

$$\|\mathfrak{A}\|(w) = \lambda \cdot \mu(w) \cdot \gamma$$

where we define  $\mu(a_1 \dots a_n) := \mu(a_1) \cdot \dots \cdot \mu(a_n)$  and  $\mu(a_i)$  is the  $(Q \times Q)$ -matrix given by  $\mu(a)_{p,q} = \mu(p, a, q)$ . Moreover,  $\lambda$  is considered as a row-, and  $\gamma$  as a column vector. Note that  $\mu(w)$  is a *stochastic matrix*: for all  $p \in Q$ ,  $\sum_{q \in Q} \mu(w)_{p,q} = 1$ .

---

<sup>1</sup>Other common terms for this model are simply *probabilistic finite automaton* [7] or *reactive probabilistic automata* [8]. When we neglect final states and consider the unfolding semantics rather than formal power series, then they correspond to the classical model of a *Markov decision process* (MDP) [6].

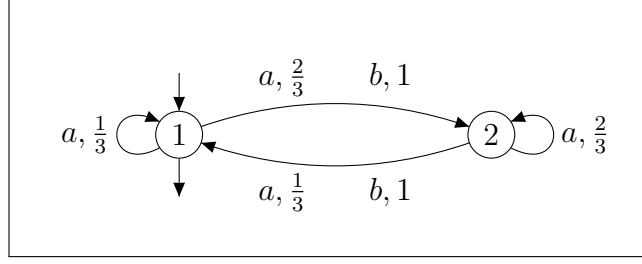


FIGURE 1. A probabilistic automaton

EXAMPLE 3.2. A probabilistic automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  with  $Q = \{1, 2\}$  and  $\Sigma = \{a, b\}$  is depicted in Figure 1. Hereby,

$$\lambda = \begin{pmatrix} 1 & 0 \end{pmatrix} \quad \mu(a) = \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} \end{pmatrix} \quad \mu(b) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \gamma = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

For  $n \in \mathbb{N}$ , we have

$$\|\mathfrak{A}\|(ab^n) = \begin{pmatrix} 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}^n \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{cases} \frac{1}{3} & \text{if } n \text{ is even} \\ \frac{2}{3} & \text{if } n \text{ is odd} \end{cases}$$

Moreover, the image of  $\|\mathfrak{A}\|$  is finite, i.e.,  $\|\mathfrak{A}\|(\Sigma^*) = \{0, \frac{1}{3}, \frac{2}{3}, 1\}$ . Finally, note that

$$L_{0 <}(\mathfrak{A}) = \{a, b\}^* \setminus \{b^n \mid n \text{ is odd}\}.$$

Exm.

### 3.2. Stochastic Languages

As in the previous section, we will consider threshold languages. Again, this study can be motivated by problems that we encounter in the context of verification. So suppose that, for some alphabet  $\Sigma^*$ , we are given a set  $Bad \subseteq \Sigma^*$  of bad behaviors, which our system  $\mathfrak{A}$  should accept with a very low probability 0.001. Then, we would like to have

$$Bad \cap L_{0.001 <}(\mathfrak{A}) = \emptyset.$$

If we are given a liveness property in terms of a set  $Good \subseteq \Sigma^*$ , then we would need a statement such as

$$Good \subseteq L_{0.999 <}(\mathfrak{A}).$$

So let us study the expressiveness of probabilistic automata in terms of threshold languages.

DEFINITION 3.3. Let  $\Sigma$  be an alphabet and  $L \subseteq \Sigma^*$  be a language. We say that  $L$  is stochastic if there are a probabilistic automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  and  $\theta \in [0, 1]$  such that  $L = L_{\theta <}(\mathfrak{A})$ . Def.

In the following, we will show three fundamental results:

- (1) Every regular language is stochastic.
- (2) There is a stochastic language that is not recursively enumerable.
- (3) For certain thresholds, a threshold language is regular.

**THEOREM 3.4.** *For every regular language  $L$ , there is a probabilistic automaton  $\mathfrak{A}$  such that  $L = L_{0<}(\mathfrak{A})$ .*

**PROOF.** Exercise. □

**THEOREM 3.5** (Rabin [7]). *There is a stochastic language over the alphabet  $\{0, 1\}$  that is not recursively enumerable.*

**PROOF.** We will determine a probabilistic automaton  $\mathfrak{A}$  such that, for thresholds  $\theta_1, \theta_2 \in [0, 1]$  with  $\theta_1 \neq \theta_2$ , we have  $L_{\theta_1<}(\mathfrak{A}) \neq L_{\theta_2<}(\mathfrak{A})$ .

Consider  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  that is given by  $Q = \{1, 2\}$ ,  $\Sigma = \{0, 1\}$ , and

$$\lambda = \begin{pmatrix} 1 & 0 \end{pmatrix} \quad \mu(0) = \begin{pmatrix} 1 & 0 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \quad \mu(1) = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 1 \end{pmatrix} \quad \gamma = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Let  $w = b_1 \dots b_n \in \Sigma^*$ . We set  $\langle w \rangle$  to be the real number  $0.b_n \dots b_1$  in binary expansion, i.e.,  $\langle \varepsilon \rangle = 0$  and, if  $n \geq 1$ ,

$$\langle w \rangle = \frac{b_n}{2^1} + \frac{b_{n-1}}{2^2} + \dots + \frac{b_1}{2^n}$$

We will show that, for all  $w \in \Sigma^*$ ,

$$(*) \quad \|\mathfrak{A}\|(w) = \langle w \rangle.$$

As  $\langle \Sigma^* \rangle$  is dense in the whole interval  $[0, 1]$ , we then have  $L_{\theta_1<}(\mathfrak{A}) \not\subseteq L_{\theta_2<}(\mathfrak{A})$  for all  $\theta_1, \theta_2 \in [0, 1]$  with  $\theta_2 < \theta_1$ . Thus, there are more stochastic languages than recursively enumerable languages, which proves our theorem.

We show (\*) by induction on  $n = |w|$ . The claim clearly holds for  $n = 0$ . Moreover, in the case  $n = 1$ ,

$$\begin{aligned} \|\mathfrak{A}\|(0) &= \mu(0)_{1,2} = 0.0 = \langle 0 \rangle \\ \text{and } \|\mathfrak{A}\|(1) &= \mu(1)_{1,2} = 0.1 = \langle 1 \rangle. \end{aligned}$$

Now suppose, for  $w = b_1 \dots b_n$  with  $n \geq 1$ , that  $\|\mathfrak{A}\|(w) = \langle w \rangle$  holds. Moreover, assume

$$\mu(w) = \begin{pmatrix} 1-p & p \\ 1-q & q \end{pmatrix}$$

for suitable  $p, q \in [0, 1]$ . In particular,

$$(+)$$

$$p = \|\mathfrak{A}\|(w) = \langle w \rangle.$$

Let  $b \in \{0, 1\}$ . We have

$$\begin{aligned}
\|\mathfrak{A}\|(b_1 \dots b_n b) &= \mu(b_1 \dots b_n b)_{1,2} \\
&= \left( \begin{pmatrix} 1-p & p \\ 1-q & q \end{pmatrix} \cdot \mu(b) \right)_{1,2} \\
&= \begin{cases} \frac{p}{2} & \text{if } b = 0 \\ \frac{1+p}{2} & \text{if } b = 1 \end{cases} \\
&= \frac{b}{2} + \frac{p}{2} \\
&\stackrel{(+)}{=} 0.b + 0.0b_n \dots b_1 \\
&= 0.bb_n \dots b_1 = \langle b_1 \dots b_n b \rangle
\end{aligned}$$

This concludes the proof of Theorem 3.5.  $\square$

REMARK 3.6. The proof of Theorem 3.5 is a pure existence proof. However, one can come up with a concrete counterexample. Let  $w_0, w_1, \dots$  be any enumeration of  $\Sigma^*$ . For  $\theta = \langle w_0 w_1 \dots \rangle$  (with a suitable extension of the encoding to infinite sequences),  $L_{\theta <}(\mathfrak{A})$  is not recursively enumerable (without proof).

In the following, we will show that, in some cases, the threshold language is regular. This is obviously the case if the threshold is 0:

THEOREM 3.7. *Let  $\mathfrak{A}$  be a probabilistic automaton. Then,  $L_{0 <}(\mathfrak{A})$  is regular.*

PROOF. Let  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  be a probabilistic automaton. Consider the NFA  $\mathfrak{B} = (Q, \Sigma, \Delta, Q_0, F)$  given by

- $Q_0 = \{q \in Q \mid \lambda(q) = 1\}$ ,
- $F = \{q \in Q \mid \gamma(q) = 1\}$ , and
- $\Delta = \{(p, a, q) \in Q \times \Sigma \times Q \mid \mu(p, a, q) > 0\}$ .

We have  $L(\mathfrak{B}) = L(\mathfrak{A})$ .  $\square$

Note that Theorem 3.7 does no longer hold when one considers automata over infinite words [1].

Next, we show that threshold languages are regular if the threshold is a certain *isolated cut point*. Intuitively, the formal power series of an automaton does not converge against an isolated cut point.

DEFINITION 3.8. Let  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  be a probabilistic automaton and let  $\theta \in [0, 1]$ . We say that  $\theta$  is an isolated cut point of  $\mathfrak{A}$  if there is  $\delta > 0$  such that, for all  $w \in \Sigma^*$ , we have

$$|\|\mathfrak{A}\|(w) - \theta| \geq \delta.$$

Def.

THEOREM 3.9 (Rabin [7]). *Let  $\mathfrak{A}$  be a probabilistic automaton and  $\theta \in [0, 1]$  be an isolated cut point of  $\mathfrak{A}$ . Then,  $L_{\theta <}(\mathfrak{A})$  is regular.*

PROOF. Let  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  be a probabilistic automaton. We assume  $Q = \{1, \dots, n\}$  and  $\lambda(1) = 1$  (hence, 1 is the initial state). Let  $\theta$  be an isolated cut point of  $\mathfrak{A}$  and let  $\delta > 0$  such that  $|\|\mathfrak{A}\|(w) - \theta| \geq \delta$ . We set  $L = L_{\theta <}(\mathfrak{A})$ .

Consider the Myhill-Nerode congruence  $\equiv_L \subseteq \Sigma^* \times \Sigma^*$  given as follows: for  $u, v \in \Sigma^*$ ,

$$u \equiv_L v \quad \text{iff} \quad \forall w \in \Sigma^* : (uw \in L \iff vw \in L)$$

It is well-known that  $\equiv_L$  has finite index iff  $L$  is regular. So let us show that  $\equiv_L$  has indeed finitely many equivalence classes.

Let  $u, v \in \Sigma^*$  and set

$$\begin{aligned} \xi^u &= (\mu(u)_{1,1}, \dots, \mu(u)_{1,n}) \\ \text{and } \xi^v &= (\mu(v)_{1,1}, \dots, \mu(v)_{1,n}). \end{aligned}$$

These vectors contain the probabilities to go, via  $u$  and  $v$ , respectively, from the initial state 1 into states  $1, \dots, n$ .

We will show that

$$(*) \quad u \not\equiv_L v \implies |\xi^u - \xi^v| \geq 4\delta$$

where, for an  $n$ -dimensional vector  $\xi$ , we let  $|\xi| = \sum_{i=1}^n |\xi_i|$ . Indeed, (\*) implies the theorem: Suppose  $u \not\equiv_L v$ . Due to (\*), there is  $i \in \{1, \dots, n\}$  such that

$$|\xi_i^u - \xi_i^v| \geq \frac{4\delta}{n}.$$

Consider the  $n$ -dimensional space  $[0, 1]^n$ , covered by “cubes” of side length  $\frac{2\delta}{n}$ , starting with  $[0, \frac{2\delta}{n}]^n$ . If  $u \not\equiv_L v$ , then  $\xi^u$  and  $\xi^v$  are not in the same cube. Conversely, if  $\xi^u$  and  $\xi^v$  share the same cube, then  $u \equiv_L v$ . Thus, “sharing the same cube” is a refinement of  $\equiv_L$ . As only finitely many cubes are necessary to cover  $[0, 1]^n$ ,  $\equiv_L$  has only finitely many equivalence classes.

So let us show (\*). Suppose  $u \not\equiv_L v$ . There is  $w \in \Sigma^*$  such that  $uw \in L$  and  $vw \notin L$  (or vice versa, which is analogous). As  $\theta$  is an isolated cut point with bound  $\delta$ , we have

$$\lambda \cdot \mu(uw) \cdot \gamma \geq \theta + \delta$$

and  $\lambda \cdot \mu(vw) \cdot \gamma \leq \theta - \delta$ .

Therefore,

$$\begin{aligned} & (\lambda \cdot \mu(uw) \cdot \gamma) - (\lambda \cdot \mu(vw) \cdot \gamma) \geq 2\delta \\ \iff & \lambda \cdot (\mu(uw) - \mu(vw)) \cdot \gamma \geq 2\delta \\ \iff & \lambda \cdot (\mu(u) \cdot \mu(w) - \mu(v) \cdot \mu(w)) \cdot \gamma \geq 2\delta \\ \iff & \lambda \cdot (\mu(u) - \mu(v)) \cdot \mu(w) \cdot \gamma \geq 2\delta \\ \iff & \sum_{i=1}^n (\xi_i^u - \xi_i^v) \cdot (\mu(w) \cdot \gamma)_i \geq 2\delta \end{aligned}$$

Let

$$\begin{aligned} \text{P} &= \Sigma \{ \xi_i^u - \xi_i^v \mid i \in \{1, \dots, n\} \text{ and } \xi_i^u - \xi_i^v > 0 \} \\ \text{and } \text{N} &= \Sigma \{ |\xi_i^u - \xi_i^v| \mid i \in \{1, \dots, n\} \text{ and } \xi_i^u - \xi_i^v < 0 \}. \end{aligned}$$

With this,  $\text{P} \geq 2\delta$ . As  $\sum_{i=1}^n \xi_i^u = \sum_{i=1}^n \xi_i^v = 1$ , it holds  $\text{P} = \text{N}$ . Therefore,

$$\sum_{i=1}^n |\xi_i^u - \xi_i^v| = 2\text{P} \geq 4\delta.$$

We conclude  $|\xi^u - \xi^v| \geq 4\delta$ . □

### 3.3. Decision Problems

Unfortunately, basic problems around isolated cut points are undecidable:

**THEOREM 3.10** (Bertoni et al. [2]). *The following problem is undecidable:*

**INPUT:** Probabilistic automaton  $\mathfrak{A}$  and  $\theta \in [0, 1]$ .  
**PROBLEM:** Is  $\theta$  an isolated cut point of  $\mathfrak{A}$ ?

**PROOF.** We show that the problem is already undecidable for fixed  $\theta = \frac{1}{2}$ .

For  $u, v \in \{0, 1\}^*$ , let  $u \wedge v$  denote the longest common suffix of  $u$  and  $v$ .

The proof is by reduction from the following undecidable variant of the PCP:

**INPUT:** Alphabet  $\Sigma$  and morphisms  $f_1, f_2 : \Sigma^* \rightarrow \{0, 1\}^*$  such that,  
for all  $a \in \Sigma$ ,  $f_1(a) = 1\dots$  and  $f_2(a) = 1\dots$

**PROBLEM:** Is  $\{f_1(w) \wedge f_2(w) \mid w \in \Sigma^*\}$  finite?

Let  $\Sigma$  and  $f_1, f_2$  constitute an instance of that problem. As in the proof of Theorem 3.5, we define  $\langle \varepsilon \rangle = 0$  and, for  $w = b_1 \dots b_n \in \{0, 1\}^+$ ,

$$\langle w \rangle = \frac{b_n}{2^1} + \frac{b_{n-1}}{2^2} + \dots + \frac{b_1}{2^n}.$$

Moreover, we set, for  $i = 1, 2$  and  $w \in \Sigma^*$ ,  $\varphi_i(w) = \langle f_i(w) \rangle$ .

CLAIM 3.11. The following statements are equivalent:

- (1) There is  $\delta > 0$  such that, for all  $w \in \Sigma^+$ ,  $|\varphi_1(w) - \varphi_2(w)| \geq \delta$ .
- (2) The set  $\{f_1(w) \wedge f_2(w) \mid w \in \Sigma^*\}$  is finite.

PROOF of Claim 3.11.

(1)  $\implies$  (2): Assume that (2) does not hold. Pick any  $n \in \mathbb{N}$ . There is  $w \in \Sigma^*$  such that  $|f_1(w) \wedge f_2(w)| \geq n$ . The latter implies that  $|\varphi_1(w) - \varphi_2(w)| < \frac{1}{2^n}$ . We conclude that (1) does not hold.

(1)  $\impliedby$  (2): Assume that (1) does not hold.

Case 1: Suppose there is  $w \in \Sigma^+$  such that  $|\varphi_1(w) - \varphi_2(w)| = 0$ . Then,  $f_1(w) = f_2(w)$ . Thus,  $\{f_1(w^i) \wedge f_2(w^i) \mid i = 1, 2, \dots\}$  is infinite, and so is  $\{f_1(u) \wedge f_2(u) \mid u \in \Sigma^*\}$ .

Case 2: Suppose that Case 1 does not apply. Then, for all  $n \in \mathbb{N}$ , there is  $w \in \Sigma^+$  such that  $0 < |\varphi_1(w) - \varphi_2(w)| < \frac{1}{2^n}$ . The latter implies  $|f_1(w) \wedge f_2(w)| \geq n$ . We conclude that  $\{f_1(w) \wedge f_2(w) \mid w \in \Sigma^*\}$  is infinite.  $\blacksquare$

For  $i = 1, 2$ , consider the probabilistic automaton  $\mathfrak{A}_i = (Q, \Sigma, \lambda, \mu_i, \gamma_i)$  that is given by  $Q = \{1, 2\}$ ,

$$\lambda = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \gamma_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \gamma_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

and, for  $i \in \{1, 2\}$  and  $a \in \Sigma$ ,

$$\mu_i(a) = \begin{pmatrix} 1 - \varphi_i(a) & \varphi_i(a) \\ 1 - \varphi_i(a) - \frac{1}{2|f_i(a)|} & \varphi_i(a) + \frac{1}{2|f_i(a)|} \end{pmatrix}$$

One can readily verify that, for all  $w \in \Sigma^*$ ,

$$\begin{aligned} \|\mathfrak{A}_1\|(w) &= \varphi_1(w) \\ \text{and } \|\mathfrak{A}_2\|(w) &= 1 - \varphi_2(w). \end{aligned}$$

We combine  $\mathfrak{A}_1$  and  $\mathfrak{A}_2$  towards a probabilistic automaton  $\mathfrak{A}$  such that  $\|\mathfrak{A}\|(\varepsilon) = 0$  and, for all  $w \in \Sigma^+$ ,

$$\begin{aligned} \|\mathfrak{A}\|(w) &= \frac{1}{2}\|\mathfrak{A}_1\|(w) + \frac{1}{2}\|\mathfrak{A}_2\|(w) \\ &= \frac{1}{2}\varphi_1(w) + \frac{1}{2}(1 - \varphi_2(w)) \\ &= \frac{1}{2} + \frac{1}{2}(\varphi_1(w) - \varphi_2(w)) \end{aligned}$$

The following statements are equivalent due to Claim 3.11:

- (1) There is  $\delta > 0$  such that, for all  $w \in \Sigma^*$ ,  $|\|\mathfrak{A}\|(w) - \frac{1}{2}| \geq \delta$ .
- (2) There is  $\delta > 0$  such that, for all  $w \in \Sigma^+$ ,  $|\varphi_1(w) - \varphi_2(w)| \geq \delta$ .
- (3) The set  $\{f_1(w) \wedge f_2(w) \mid w \in \Sigma^*\}$  is finite.

Thus,  $\frac{1}{2}$  is an isolated cut point of  $\mathfrak{A}$  iff  $\{f_1(w) \wedge f_2(w) \mid w \in \Sigma^*\}$  is finite. As the latter is undecidable, we have shown Theorem 3.10.  $\square$

Let us consider some threshold problems for probabilistic automata.

**THEOREM 3.12** (Gimbert & Oualhadj [3]). *The following problem is undecidable:*

**INPUT:** Probabilistic automaton  $\mathfrak{A}$ .  
**PROBLEM:** Do we have  $L_{\frac{1}{2}}(\mathfrak{A}) \neq \emptyset$ ?

**PROOF.** The proof [3] is by reduction from the following undecidable variant of the PCP:

**INPUT:** Alphabet  $\Sigma$  and morphisms  $f_1, f_2 : \Sigma^* \rightarrow \{0, 1\}^*$  such that, for all  $a \in \Sigma$ ,  $f_1(a) = 1\dots$  and  $f_2(a) = 1\dots$ .

**PROBLEM:** Is there  $w \in \Sigma^+$  such that  $f_1(w) = f_2(w)$ ?

Let an instance of that problem be given by an alphabet  $\Sigma$  and morphisms  $f_1, f_2 : \Sigma^* \rightarrow \{0, 1\}^*$ . Consider the mappings  $\varphi_1, \varphi_2$  and the probabilistic automaton  $\mathfrak{A}$  from the proof of Theorem 3.10. Recall that  $\|\mathfrak{A}\|(\varepsilon) = 0$  and, for all  $w \in \Sigma^*$ ,

$$\|\mathfrak{A}\|(w) = \frac{1}{2} + \frac{1}{2}(\varphi_1(w) - \varphi_2(w))$$

Then, for all  $w \in \Sigma^*$ ,

$$\begin{aligned} \|\mathfrak{A}\|(w) &= \frac{1}{2} \\ \text{iff } w \in \Sigma^+ \text{ and } \varphi_1(w) &= \varphi_2(w) \\ \text{iff } w \in \Sigma^+ \text{ and } f_1(w) &= f_2(w). \end{aligned}$$

Note that the latter step requires the fact that  $f_1(w)$  and  $f_2(w)$  both start with 1. This concludes the proof.  $\square$

**THEOREM 3.13** (Madani et al. [4]). *The following problem is undecidable:*

**INPUT:** Probabilistic automaton  $\mathfrak{A}$  and  $\theta \in [0, 1]$ .  
**PROBLEM:** Do we have  $L_{\theta}(\mathfrak{A}) \neq \emptyset$ ?

An elegant proof of Theorem 3.13 (even for fixed threshold  $\frac{3}{8}$ ), using Theorem 3.12, can be found in [3].

### Exercises for Chapter 3

EXERCISE 3.14. Consider the probabilistic automaton  $\mathfrak{A}$  from Example 3.2. Determine the (finite) class  $\mathcal{L} = \{L_{\theta <}(\mathfrak{A}) \mid \theta \in [0, 1]\}$ . Ex.

EXERCISE 3.15. Prove Theorem 3.4. (Hint: probabilistic automata with one positive entry in every row of a transition matrix are deterministic finite automata.) Ex.

EXERCISE 3.16. Let  $\mathfrak{A}$  be a probabilistic automaton and  $\theta \in [0, 1]$  be an isolated cut point of  $\mathfrak{A}$ . Recall that  $L_{\theta <}(\mathfrak{A})$  is regular (Theorem 3.9). Determine an upper bound on the number of states of a finite automaton recognizing  $L_{\theta <}(\mathfrak{A})$ . Ex.

### Further Reading

- [1] C. Baier, N. Bertrand, and M. Größer. On decision problems for probabilistic Büchi automata. In *Proceedings of FoSSaCS'08*, volume 4962 of *Lecture Notes in Computer Science*, pages 287–301. Springer, 2008.
- [2] A. Bertoni, G. Mauri, and M. Torelli. Some recursively unsolvable problems relating to isolated cutpoints in probabilistic automata. In *Proceedings of ICALP'77*, volume 52 of *Lecture Notes in Computer Science*, pages 87–94. Springer, 1977.
- [3] H. Gimbert and Y. Oualhadj. Automates probabilistes: problèmes décidables et indécidables. Rapport de Recherche RR-1464-09 LaBRI, 2009. <http://hal.archives-ouvertes.fr/hal-00422888/en/>.
- [4] O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence*, 147(1-2):5–34, 2003.
- [5] A. Paz. *Introduction to probabilistic automata*. Academic Press, 1971.
- [6] M. L. Puterman. *Markov Decision Processes*. John Wiley & Sons, Inc., New York, NY, 1994.
- [7] M. O. Rabin. Probabilistic automata. *Information and Control*, 6:230–245, 1963.
- [8] R. Segala. Probability and nondeterminism in operational models of concurrency. In *Proceedings of CONCUR'06*, volume 4137 of *Lecture Notes in Computer Science*, pages 64–78. Springer, 2006.



## CHAPTER 4

### Weighted Logic

In this chapter, we present a logical characterization of weighted automata in terms of a monadic second-order logic [2,3]. This characterization extends one of the fundamental theorems in computer science, which goes back to Büchi and Elgot: the precise correspondence between finite automata and monadic second-order formulas [1,4]. A detailed exposition of weighted logics and their connection with weighted automata can be found in [2,3].

#### 4.1. MSO Logic over Words

We fix infinite supplies  $Var = \{x, y, x_1, x_2, \dots\}$  of *first-order* and  $VAR = \{X, Y, X_1, X_2, \dots\}$  of *second-order variables*.

Let us recall monadic second-order logic in the traditional boolean setting. Fix an alphabet  $\Sigma$ .

**DEFINITION 4.1.** The set of monadic second-order formulas (MSO formulas) over  $\Sigma$  is given by the grammar

$$\varphi ::= P_a(x) \mid x \leq y \mid x \in X \mid \varphi_1 \vee \varphi_2 \mid \neg\varphi \mid \exists x.\varphi \mid \exists X.\varphi$$

where  $a \in \Sigma$ ,  $x, y \in Var$ , and  $X \in VAR$ . The set of all those formulas is denoted by  $MSO(\Sigma)$ . Def.

Furthermore, we may use usual abbreviations such as  $\varphi_1 \wedge \varphi_2$ ,  $x = y$ ,  $succ(x, y)$ ,  $\varphi_1 \rightarrow \varphi_2$ ,  $\varphi_1 \leftrightarrow \varphi_2$ ,  $\forall x.\varphi$ , and  $\forall X.\varphi$ .

For  $\varphi \in MSO(\Sigma)$ , let  $Free(\varphi)$  denote the set of variables that are free in  $\varphi$ . If  $Free(\varphi) = \emptyset$ , then  $\varphi$  is called a *sentence*.

For a finite set  $\mathcal{V} \subseteq Var \cup VAR$  and a word  $w = a_1 \dots a_n \in \Sigma^*$ , a  $(\mathcal{V}, w)$ -*assignment* is a function  $\sigma$  that maps a first-order variable in  $\mathcal{V}$  to an element of  $\{1, \dots, n\}$  and a second-order variable in  $\mathcal{V}$  to a subset of  $\{1, \dots, n\}$ . For  $x \in Var$ ,  $i \in \{1, \dots, n\}$ , and  $I \subseteq \{1, \dots, n\}$ ,  $\sigma[x \rightarrow i]$  will denote the  $(\mathcal{V} \cup \{x\}, w)$ -assignment that maps  $x$  to  $i$  and, otherwise, coincides with  $\sigma$ . The  $(\mathcal{V} \cup \{X\}, w)$ -assignment  $\sigma[X \rightarrow I]$  is defined accordingly.

A word  $w = a_1 \dots a_n \in \Sigma^*$  can be considered as the mathematical structure

$$\underline{w} := (\{1, \dots, n\}, \leq, (R_a)_{a \in \Sigma})$$

where, for  $a \in \Sigma$ ,  $R_a = \{i \in \{1, \dots, n\} \mid a_i = a\}$ .

For a formula  $\varphi \in \text{MSO}(\Sigma)$ , a finite set  $\mathcal{V} \subseteq \text{Var} \cup \text{VAR}$  with  $\text{Free}(\varphi) \subseteq \mathcal{V}$ , a word  $w \in \Sigma^*$ , and a  $(\mathcal{V}, w)$ -assignment  $\sigma$ , the satisfaction relation  $(w, \sigma) \models \varphi$  is defined as usual, by induction on the basis of the structure  $\underline{w}$ .

EXAMPLE 4.2. Let  $\Sigma = \{a, b, c\}$ . Here are some formulas from  $\text{MSO}(\Sigma)$ :

- $\varphi_1 = \forall x.(P_a(x) \rightarrow \exists y.(x \leq y \wedge P_b(y)))$   
“every  $a$  is eventually followed by a  $b$ ”
- $\varphi_2 = \exists x.\forall y.(x \leq y \rightarrow \neg P_a(y))$   
“from some time on, there are only  $b$ 's or  $c$ 's”
- $\varphi_3 = \exists X.\exists x.\exists y.(\neg \exists z.(z < x \vee y < z)$   
 $\wedge x \in X \wedge y \in X$   
 $\wedge \forall z.(z \in X \leftrightarrow \forall z'.(\text{succ}(z, z') \rightarrow \neg(z' \in X)))$   
“the word has odd length”

In the following, it will be convenient to encode a pair  $(w, \sigma)$  where  $\sigma$  is a  $(\mathcal{V}, w)$ -assignment as a word over the extended alphabet  $\Sigma_{\mathcal{V}} := \Sigma \times \{0, 1\}^{\mathcal{V}}$  (setting  $\Sigma_{\emptyset} = \Sigma$ ). We write a word  $(a_1, \sigma_1) \dots (a_n, \sigma_n) \in \Sigma_{\mathcal{V}}^*$  as  $(w, \sigma)$  where  $w = a_1 \dots a_n \in \Sigma^*$  and  $\sigma = \sigma_1 \dots \sigma_n \in \{0, 1\}^{\mathcal{V}}$ . We call  $(w, \sigma)$  *valid* if, for each first-order variable  $x \in \mathcal{V}$ , the  $x$ -row of  $\sigma$  contains exactly one 1. If  $(w, \sigma)$  is valid, then  $\sigma$  can be considered as the  $(\mathcal{V}, w)$ -assignment that maps a first-order variable  $x \in \mathcal{V}$  to the unique position carrying 1 in the  $x$ -row and a second-order variable  $X \in \mathcal{V}$  to the set of positions carrying 1 in the  $X$ -row.

EXAMPLE 4.3. Let  $\Sigma = \{a, b, c\}$  and  $\mathcal{V} = \{x, y, X\}$ . Consider the pair  $(w, \sigma) \in \Sigma_{\mathcal{V}}^*$  given as follows:

$$\begin{array}{l} w \{ \\ \sigma \left\{ \begin{array}{llll} & a & b & a & c \\ x & 1 & 0 & 0 & 0 \\ y & 0 & 0 & 0 & 1 \\ X & 1 & 0 & 1 & 1 \end{array} \right. \end{array}$$

Then,  $(w, \sigma)$  is valid, i.e.,  $\sigma$  can be considered as a  $(\mathcal{V}, w)$ -assignment. □ Ex.

Let  $\varphi \in \text{MSO}(\Sigma)$  and  $\mathcal{V}$  be a finite set of variables such that  $\text{Free}(\varphi) \subseteq \mathcal{V}$ . It is easy to see that  $N_{\mathcal{V}} := \{(w, \sigma) \in \Sigma_{\mathcal{V}}^* \mid (w, \sigma) \text{ is valid}\}$  is recognizable in terms of a finite automaton. Moreover, the set  $L_{\mathcal{V}}(\varphi) := \{(w, \sigma) \in N_{\mathcal{V}} \mid (w, \sigma) \models \varphi\} \subseteq \Sigma_{\mathcal{V}}^*$  of valid pairs satisfying  $\varphi$  form a recognizable language. Let  $L(\varphi)$  abbreviate  $L_{\text{Free}(\varphi)}(\varphi)$ .

THEOREM 4.4 (Büchi and Elgot [1, 4]). *Let  $L \subseteq \Sigma^*$ . The following statements are equivalent:*

- (1) *There is a finite automaton  $\mathfrak{A}$  such that  $L(\mathfrak{A}) = L$ .*
- (2) *There is a sentence  $\varphi \in \text{MSO}(\Sigma)$  such that  $L(\varphi) = L$ .*

## 4.2. Weighted MSO Logic over Words

For the rest of this chapter, we fix a *commutative* semiring  $\mathbb{S} = (S, \oplus, \odot, 0, \mathbb{1})$  and an alphabet  $\Sigma$ .

DEFINITION 4.5. The set of weighted monadic second-order formulas (or, simply, wMSO formulas) over  $\mathbb{S}$  and  $\Sigma$  is given by the grammar

$$\begin{aligned} \varphi ::= & s \mid P_a(x) \mid \neg(P_a(x)) \mid x \leq y \mid \neg(x \leq y) \mid \\ & x \in X \mid \neg(x \in X) \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \\ & \exists x.\varphi \mid \exists X.\varphi \mid \forall x.\varphi \mid \forall X.\varphi \end{aligned}$$

where  $s \in \mathbb{S}$ ,  $a \in \Sigma$ ,  $x, y \in \text{Var}$ , and  $X \in \text{VAR}$ . The set of those formulas is denoted by  $\text{wMSO}(\mathbb{S}, \Sigma)$ . Def.

Negated formulas and formulas of the form  $s$ ,  $P_a(x)$ ,  $x \leq y$ , and  $x \in X$  as well as are called *atomic*.

DEFINITION 4.6. Let  $\varphi \in \text{wMSO}(\mathbb{S}, \Sigma)$  and  $\mathcal{V}$  be a finite set of first-order and second-order variables such that  $\text{Free}(\varphi) \subseteq \mathcal{V}$ . The semantics of  $\varphi$  wrt.  $\mathcal{V}$  is a formal power series  $\llbracket \varphi \rrbracket_{\mathcal{V}} \in \mathbb{S}\langle\langle \Sigma_{\mathcal{V}}^* \rangle\rangle$ , which is given as follows: If  $(w, \sigma)$  is not valid, we set  $\llbracket \varphi \rrbracket_{\mathcal{V}}(w, \sigma) = 0$ . Otherwise,  $\llbracket \varphi \rrbracket_{\mathcal{V}}(w, \sigma)$ , where  $w = a_1 \dots a_n$ , is determined inductively as shown in Table 1. Def.

We will simply write  $\llbracket \varphi \rrbracket$  for  $\llbracket \varphi \rrbracket_{\text{Free}(\varphi)}$ .

Let us consider some examples.

EXAMPLE 4.7. Recall that  $\mathbb{Bool} = (\{\mathbf{false}, \mathbf{true}\}, \vee, \wedge, \mathbf{false}, \mathbf{true})$  is the boolean algebra. The logic  $\text{wMSO}(\mathbb{Bool}, \Sigma)$  actually reduces to  $\text{MSO}(\Sigma)$ .

EXAMPLE 4.8. Let  $\Sigma = \{a, b, c\}$  and consider the semiring of natural numbers  $\mathbb{Nat} = (\mathbb{N}, +, \cdot, 0, 1)$ . Let  $\varphi = \exists x.P_a(x) \in \text{wMSO}(\mathbb{Nat}, \Sigma)$ . For all  $w \in \Sigma^*$ , we have  $\llbracket \varphi \rrbracket(w) = |w|_a$ , i.e.,  $\varphi$  “counts” the number of occurrences of  $a$  in  $w$ .

EXAMPLE 4.9. Consider the alphabet  $\Sigma = \{a_1, \dots, a_k\}$  and the *reliability* semiring  $([0, 1], \max, \cdot, 0, 1)$ . Assume that every letter  $a_i$  comes with a reliability  $p_i \in [0, 1]$ . Let  $\varphi = \forall x.\bigvee_{i=1}^k (P_{a_i}(x) \wedge p_i)$ . Then,  $\llbracket \varphi \rrbracket(w)$  can be interpreted as the reliability of the word  $w \in \Sigma^*$ .

EXAMPLE 4.10. Consider the sentences  $\varphi_1 = \forall x.2$  and  $\varphi_2 = \forall y.\forall x.2$  from  $\text{wMSO}(\mathbb{Nat}, \Sigma)$ . For all  $w \in \Sigma^*$ , we have  $\llbracket \varphi_1 \rrbracket(w) = 2^{|w|}$  and  $\llbracket \varphi_2 \rrbracket(w) = (2^{|w|})^{|w|} = 2^{|w|^2}$ . While  $\llbracket \varphi_1 \rrbracket$  is recognizable (i.e., there is a weighted automaton whose semantics is  $\llbracket \varphi_1 \rrbracket$ ),  $\llbracket \varphi_2 \rrbracket$  is not: Suppose there is a weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\mathbb{Nat}$  such that  $\|\mathfrak{A}\| = \llbracket \varphi_2 \rrbracket$ . Let  $m = \max\{\lambda(p), \gamma(p), \mu(p, a, q) \mid p, q \in Q \text{ and } a \in \Sigma\}$ . Then, for all  $w \in \Sigma^*$ ,

$$\begin{aligned}
\llbracket s \rrbracket_{\mathcal{V}}(w, \sigma) &= s \\
\llbracket P_a(x) \rrbracket_{\mathcal{V}}(w, \sigma) &= \begin{cases} 1 & \text{if } a_{\sigma(x)} = a \\ 0 & \text{otherwise} \end{cases} \\
\llbracket x \leq y \rrbracket_{\mathcal{V}}(w, \sigma) &= \begin{cases} 1 & \text{if } \sigma(x) \leq \sigma(y) \\ 0 & \text{otherwise} \end{cases} \\
\llbracket x \in X \rrbracket_{\mathcal{V}}(w, \sigma) &= \begin{cases} 1 & \text{if } \sigma(x) \in \sigma(X) \\ 0 & \text{otherwise} \end{cases} \\
\llbracket \neg\varphi \rrbracket_{\mathcal{V}}(w, \sigma) &= \begin{cases} 1 & \text{if } \llbracket \varphi \rrbracket_{\mathcal{V}}(w, \sigma) = 0 \\ 0 & \text{if } \llbracket \varphi \rrbracket_{\mathcal{V}}(w, \sigma) = 1 \end{cases}
\end{aligned}$$

$$\begin{aligned}
\llbracket \varphi_1 \vee \varphi_2 \rrbracket_{\mathcal{V}}(w, \sigma) &= \llbracket \varphi_1 \rrbracket_{\mathcal{V}}(w, \sigma) \oplus \llbracket \varphi_2 \rrbracket_{\mathcal{V}}(w, \sigma) \\
\llbracket \varphi_1 \wedge \varphi_2 \rrbracket_{\mathcal{V}}(w, \sigma) &= \llbracket \varphi_1 \rrbracket_{\mathcal{V}}(w, \sigma) \odot \llbracket \varphi_2 \rrbracket_{\mathcal{V}}(w, \sigma)
\end{aligned}$$

$$\begin{aligned}
\llbracket \exists x.\varphi \rrbracket_{\mathcal{V}}(w, \sigma) &= \bigoplus_{i \in \{1, \dots, n\}} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{x\}}(w, \sigma[x \rightarrow i]) \\
\llbracket \forall x.\varphi \rrbracket_{\mathcal{V}}(w, \sigma) &= \bigodot_{i \in \{1, \dots, n\}} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{x\}}(w, \sigma[x \rightarrow i]) \\
\llbracket \exists X.\varphi \rrbracket_{\mathcal{V}}(w, \sigma) &= \bigoplus_{I \subseteq \{1, \dots, n\}} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{X\}}(w, \sigma[X \rightarrow I]) \\
\llbracket \forall X.\varphi \rrbracket_{\mathcal{V}}(w, \sigma) &= \bigodot_{I \subseteq \{1, \dots, n\}} \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{X\}}(w, \sigma[X \rightarrow I])
\end{aligned}$$

TABLE 1. The semantics of  $\text{wMSO}(\mathbb{S}, \Sigma)$ -formulas

$\|\mathfrak{A}\|(w) \leq |Q|^{|w|+1} \cdot m^{|w|+2}$ , which is a contradiction to  $\|\mathfrak{A}\|(w) = 2^{|w|^2}$ . Now let  $\varphi_3 = \forall X. 2 \in \text{MSO}(\text{Nat}, \Sigma)$ . Then, for all  $w \in \Sigma^*$ ,  $\llbracket \varphi_3 \rrbracket(w) = 2^{2^{|w|}}$ . Thus,  $\llbracket \varphi_3 \rrbracket$  is not recognizable either.

The last examples show that we need to restrict universal quantification in formulas to obtain a logical characterization of weighted automata. So let us introduce the notion of a restricted formula:

DEFINITION 4.11. A formula  $\varphi \in \text{wMSO}(\mathbb{S}, \Sigma)$  is called *restricted* if

- it does not contain universal set quantification, and
- for every subformula of  $\varphi$  of the form  $\forall x.\psi$ , the series  $\llbracket \psi \rrbracket$  is a recognizable step function.

The set of restricted MSO formulas is denoted by  $\text{wRMSO}(\mathbb{S}, \Sigma)$ . Def.

Here, a series  $f \in \mathbb{S}\langle\langle \Sigma_{\mathcal{V}}^* \rangle\rangle$  (for a finite set  $\mathcal{V}$ ) is a *recognizable step function* if there are  $k \in \mathbb{N}$ ,  $s_1, \dots, s_k \in S$ , and regular languages  $L_1, \dots, L_k \subseteq \Sigma_{\mathcal{V}}^*$  such that

$$f = \bigoplus_{j=1}^k s_j \odot \mathbb{1}_{L_j}$$

where  $\mathbb{1}_{L_i}$  is the characteristic function of  $L_i$ .

Let  $f \in \mathbb{S}\langle\langle \Sigma^* \rangle\rangle$ . We say that  $f$  is *recognizable* if there is a weighted automaton  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  over  $\mathbb{S}$  such that  $\|\mathfrak{A}\| = f$ . For a formula class  $\mathcal{C} \subseteq \text{MSO}(\mathbb{S}, \Sigma)$ , we say that  $f$  is  $\mathcal{C}$ -*definable* if there exists a sentence  $\varphi \in \mathcal{C}$  such that  $\llbracket \varphi \rrbracket = f$ .

The rest of this chapter will be devoted to the proof of the following theorem, which is a proper generalization of Theorem 4.4 to weighted automata and logic.

**THEOREM 4.12** (Droste & Gastin [2, 3]). *Let  $f \in \mathbb{S}\langle\langle \Sigma^* \rangle\rangle$ . The following statements are equivalent:*

- (1)  $f$  is  $\text{wRMSO}(\mathbb{S}, \Sigma)$ -definable.
- (2)  $f$  is recognizable.

### 4.3. From Logic to Automata

For proving the direction (1)  $\implies$  (2) of Theorem 4.12, we proceed inductively. Two lemmas will provide us with the required translations.

**LEMMA 4.13.** *Let  $\varphi, \psi \in \text{wMSO}(\mathbb{S}, \Sigma)$ .*

- (a) *If  $\varphi$  is atomic, then  $\llbracket \varphi \rrbracket$  is recognizable.*
- (b) *If  $\llbracket \varphi \rrbracket$  and  $\llbracket \psi \rrbracket$  are recognizable, then so are  $\llbracket \varphi \vee \psi \rrbracket$  and  $\llbracket \varphi \wedge \psi \rrbracket$ .*
- (c) *If  $\llbracket \varphi \rrbracket$  is recognizable, then  $\llbracket \exists x.\varphi \rrbracket$  and  $\llbracket \exists X.\varphi \rrbracket$  are recognizable.*

The most difficult case, however, arises when we are facing universal quantification.

**LEMMA 4.14.** *Let  $\varphi \in \text{wMSO}(\mathbb{S}, \Sigma)$  such that  $\llbracket \varphi \rrbracket$  is a recognizable step function. Then,  $\llbracket \forall x.\varphi \rrbracket$  is recognizable.*

**PROOF.** Let  $\varphi \in \text{wMSO}(\mathbb{S}, \Sigma)$  such that  $\llbracket \varphi \rrbracket$  is a recognizable step function. Let  $\mathcal{W} = \text{Free}(\varphi)$  and  $\mathcal{V} = \text{Free}(\forall x.\varphi) = \mathcal{W} \setminus \{x\}$ . There are  $k \in \mathbb{N}$ ,  $s_1, \dots, s_k \in S$ , and regular languages  $L_1, \dots, L_k \subseteq \Sigma_{\mathcal{W}}^*$  such that, for all  $(w, \sigma) \in \Sigma_{\mathcal{W}}^*$ , we have  $\llbracket \varphi \rrbracket(w, \sigma) = \bigoplus_{j=1}^k s_j \odot \mathbb{1}_{L_j}$ . Without loss of generality, we assume that the sets  $L_j$  form a partition of  $\Sigma_{\mathcal{W}}^*$ .

Case 1: Suppose  $x \in \mathcal{W}$ . Consider the alphabet  $\tilde{\Sigma} = \Sigma \times \{1, \dots, k\}$ . A word from  $(\tilde{\Sigma}_{\mathcal{V}})^*$  will be written as the triple  $(w, \nu, \sigma)$  where  $(w, \sigma)$  is its projection onto  $\Sigma_{\mathcal{V}}$  and  $\nu \in \{1, \dots, k\}^{|w|}$ . In the obvious manner, we can consider  $\nu$

to be a mapping  $\nu : \{1, \dots, |w|\} \rightarrow \{1, \dots, k\}$ . Let  $\tilde{L}$  be the set of words  $(w, \nu, \sigma)$  over  $\tilde{\Sigma}_{\mathcal{V}}$  such that, for all  $i \in \{1, \dots, |w|\}$  and  $j \in \{1, \dots, k\}$ ,

$$\nu(i) = j \text{ implies } (w, \sigma[x \rightarrow i]) \in L_j.$$

We will show that

$$(*) \quad \tilde{L} \text{ is regular.}$$

Thus, there is a deterministic finite automaton  $\tilde{\mathfrak{A}}$  over  $\tilde{\Sigma}_{\mathcal{V}}$  such that  $L(\tilde{\mathfrak{A}}) = \tilde{L}$ . In turn,  $\tilde{\mathfrak{A}}$  can be transformed into a weighted automaton  $\mathfrak{A}$  with input alphabet  $\tilde{\Sigma}_{\mathcal{V}}$  as follows:

- A transition of the form  $(p, (a, j, \tau), q)$  is replaced with a transition  $(p, (a, j, \tau), q)$  of weight  $s_j$  (i.e.,  $\mu(p, (a, j, \tau), q) = s_j$ ). All other transitions have weight 0.
- The initial state of  $\mathfrak{A}$  gets the initial weight 1, the other states get the initial weight 0.
- Each final state of  $\tilde{\mathfrak{A}}$  gets the final weight 1, the other states get the final weight 0.

Then, for all  $(w, \nu, \sigma) \in (\tilde{\Sigma}_{\mathcal{V}})^*$ , we have

$$\|\mathfrak{A}\|(w, \nu, \sigma) = \begin{cases} \bigodot_{j=1}^k s_j^{|\nu^{-1}(j)|} & \text{if } (w, \nu, \tau) \in \tilde{L} \\ 0 & \text{if } (w, \nu, \tau) \notin \tilde{L} \end{cases}$$

Note that, for each  $(w, \sigma) \in \Sigma_{\mathcal{V}}^*$ , there is a unique extension  $\nu$  such that  $(w, \nu, \sigma) \in \tilde{L}$ . If  $\mathfrak{A}'$  is the canonical projection of  $\mathfrak{A}$  onto the alphabet  $\Sigma_{\mathcal{V}}$ , we therefore have  $\|\mathfrak{A}'\|(w, \sigma) = \bigodot_{j=1}^k s_j^{|\nu^{-1}(j)|}$ . As

$$\llbracket \forall x. \varphi \rrbracket(w, \sigma) = \bigodot_{i=1}^{|w|} \llbracket \varphi \rrbracket(w, \sigma[x \rightarrow i]) = \bigodot_{j=1}^k s_j^{|\nu^{-1}(j)|},$$

we are done.

Let us show (\*). Actually, it is sufficient, to construct, for every  $j \in \{1, \dots, k\}$ , a finite automaton  $\tilde{\mathfrak{A}}_j$  over  $\tilde{\Sigma}_{\mathcal{V}}$  recognizing the set  $\tilde{L}_j$  of words  $(w, \nu, \sigma)$  over  $\tilde{\Sigma}_{\mathcal{V}}$  such that, for all  $i \in \{1, \dots, |w|\}$ ,

$$\nu(i) = j \text{ implies } (w, \sigma[x \rightarrow i]) \in L_j.$$

The reason is that  $\tilde{L} = \bigcap_{j \in \{1, \dots, k\}} \tilde{L}_j$ .

So let us fix  $j \in \{1, \dots, k\}$  and let  $\mathfrak{A}_j = (Q, \Sigma_{\mathcal{W}}, \delta, q_0, F)$  be a deterministic finite automaton such that  $L(\mathfrak{A}_j) = L_j$ . We specify  $\tilde{\mathfrak{A}}_j = (\tilde{Q}, \tilde{\Sigma}_{\mathcal{V}}, \tilde{\delta}, \tilde{q}_0, \tilde{F})$  with  $L(\tilde{\mathfrak{A}}_j) = \tilde{L}_j$  as follows:

- $\tilde{Q} = Q \times 2^Q$
- $\tilde{q}_0 = (q_0, \emptyset)$

- $\tilde{F} = Q \times 2^F$
- $\tilde{\delta}((p, P), (a, \ell, \tau)) = (\delta(p, (a, \tau[x \rightarrow 0])), P')$

where

$$P' = \begin{cases} \{\delta(q, (a, \tau[x \rightarrow 0])) \mid q \in P\} & \text{if } \ell \neq j \\ \{\delta(q, (a, \tau[x \rightarrow 0])) \mid q \in P\} \cup \{\delta(p, (a, \tau[x \rightarrow 1]))\} & \text{if } \ell = j \end{cases}$$

Case 2: Suppose  $x \notin \mathcal{W}$ , which implies  $\mathcal{W} = \mathcal{V}$ . Consider the formula  $\varphi' = \varphi \wedge (x \leq x)$ . Then,  $\llbracket \varphi' \rrbracket$  is recognizable due to Lemma 4.13(a) and (b). Moreover, it is a recognizable step function. Clearly,  $\llbracket \varphi' \rrbracket_{\mathcal{V} \cup \{x\}} = \llbracket \varphi \rrbracket_{\mathcal{V} \cup \{x\}}$  and  $\llbracket \forall x. \varphi' \rrbracket_{\mathcal{V}} = \llbracket \forall x. \varphi \rrbracket_{\mathcal{V}}$ , which is recognizable due to Case 1.  $\square$

From Lemmas 4.13 and 4.14, we can indeed deduce the direction (1)  $\implies$  (2) of Theorem 4.12. If  $\mathbb{S}$  is a computable field, then that direction is effective and one can decide if a formula is restricted. Thus, decidable decision problems for weighted automata such as emptiness and equivalence can be extended to wRMSO-formulas [2, 3].

#### 4.4. From Automata to Logic

Let us prove the direction (2)  $\implies$  (1) of Theorem 4.12.

Let  $\varphi \in \text{MSO}(\Sigma)$  be an unweighted MSO formula without set quantifier. We define formulas  $\varphi^+, \varphi^- \in \text{wMSO}(\mathbb{S}, \Sigma)$  such that  $\llbracket \varphi^+ \rrbracket = \mathbb{1}_{L(\varphi)}$  and  $\llbracket \varphi^- \rrbracket = \mathbb{1}_{L(\neg\varphi)}$  inductively as follows: We assume that, in  $\varphi$ , negation is pushed inwards so that it is applied to (positive) atomic formulas only (i.e., our syntax makes use of universal quantification and conjunction). If  $\varphi$  is atomic, then we set  $\varphi^+ = \varphi$  and  $\varphi^- = \neg\varphi$  (where  $\neg\neg\psi$  is reduced to  $\psi$ ). Moreover,

$$\begin{aligned} (\varphi \vee \psi)^+ &= \varphi^+ \vee (\varphi^- \wedge \psi^+) \\ (\varphi \vee \psi)^- &= \varphi^- \wedge \psi^- \\ (\varphi \wedge \psi)^- &= \varphi^- \vee (\varphi^+ \wedge \psi^-) \\ (\varphi \wedge \psi)^+ &= \varphi^+ \wedge \psi^+ \\ (\exists x. \varphi)^+ &= \exists x. (\varphi^+(x) \wedge \forall y. ((x \leq y) \vee (\neg(x \leq y) \wedge \varphi^-(y)))) \\ (\exists x. \varphi)^- &= \forall x. \varphi^- \\ (\forall x. \varphi)^- &= \exists x. (\varphi^-(x) \wedge \forall y. ((x \leq y) \vee (\neg(x \leq y) \wedge \varphi^+(y)))) \\ (\forall x. \varphi)^+ &= \forall x. \varphi^+ \end{aligned}$$

Now let  $\mathfrak{A} = (Q, \Sigma, \lambda, \mu, \gamma)$  be a weighted automaton over  $\mathbb{S}$ . We show that there is a sentence  $\varphi \in \text{wRMSO}(\mathbb{S}, \Sigma)$  such that  $\llbracket \varphi \rrbracket = \|\mathfrak{A}\|$ .

In the following,  $t$  and  $t'$  will range over  $Q \times \Sigma \times Q$ . Let  $\overline{X}$  be the collection  $(X_t)_t$  of second-order variables. The construction of a wMSO sentence from a weighted automaton follows the standard procedure of transforming a finite

automaton into an MSO sentence: an interpretation of second-order variables reflects an assignment of positions in a word to transitions. We first provide some building blocks of the desired wRMSO formula.

The formula

$$\text{partition}(\overline{X}) := \forall x. \bigvee_t ((x \in X_t) \wedge \bigwedge_{t' \neq t} \neg(x \in X_{t'}))$$

claims that  $\overline{X}$  actually represents a run, i.e., an assignment of vertices to transitions. Now let

$$\begin{aligned} \psi(\overline{X}) := & \text{partition}(\overline{X}) \wedge \bigwedge_{p,a,q} \forall x. ((x \in X_{p,a,q}) \rightarrow P_a(x))^+ \\ & \wedge \forall x. \forall y. (\text{succ}(x, y) \rightarrow \bigvee_{\substack{p,q,r \in Q \\ a,b \in \Sigma}} (x \in X_{p,a,q} \wedge (y \in X_{q,b,r})))^+ \end{aligned}$$

where

$$\text{succ}(x, y) := (x \leq y) \wedge \neg(y \leq x) \wedge \forall z. (z \leq x \vee y \leq z)$$

and the implications are considered to be unweighted (i.e., boolean) formula.

We set

$$\begin{aligned} \varphi(\overline{X}) := & \psi(\overline{X}) \wedge \bigwedge_{p,a,q} \forall x. ((x \in X_{p,a,q}) \rightarrow \mu(p, a, q)) \\ & \wedge \exists y. (\min(y) \wedge \bigvee_{p,a,q} (y \in X_{p,a,q} \wedge \lambda(p))) \\ & \wedge \exists z. (\max(z) \wedge \bigvee_{p,a,q} (z \in X_{p,a,q} \wedge \gamma(q))) \end{aligned}$$

where  $(x \in X) \rightarrow s := \neg(x \in X) \vee ((x \in X) \wedge s)$ ,  $\min(y) := \forall x. y \leq x$ , and  $\max(z) := \forall x. x \leq z$ . For

$$\xi := \exists \overline{X}. \varphi(\overline{X})$$

$$\zeta := (\lambda \odot \gamma) \wedge \forall x. \emptyset$$

we have

$$\|\mathfrak{A}\| = \llbracket \zeta \vee \xi \rrbracket.$$

As  $\zeta \vee \xi \in \text{wRMSO}(\mathbb{S}, \Sigma)$ , this concludes the proof of (2)  $\implies$  (1) in Theorem 4.12.

## Exercises for Chapter 4

**EXERCISE 4.15.** Let  $\Sigma = \{a, b\}$ . Determine semirings  $\mathbb{S}_i$  and sentences  $\varphi_i \in \text{wMSO}(\mathbb{S}_i, \Sigma)$ ,  $i = 1, 2$ , such that, for all  $w = a_1 \dots a_n \in \Sigma^*$ ,

- $\llbracket \varphi_1 \rrbracket(w) = |\{i \in \{1, \dots, n-1\} \mid a_i a_{i+1} = ab\}|$ .
- $\llbracket \varphi_2 \rrbracket(w) = \{u \in \Sigma^* \mid u \text{ is an infix of } w\}$ .

*Hint:* Note that  $\mathbb{S}_2$  is not commutative. For the universal quantification, the product over the positions  $\{1, \dots, n\}$  of  $w$  should follow the natural ordering  $1 \leq \dots \leq n$ . Ex.

EXERCISE 4.16. Let  $\Sigma = \{a, b\}$ . Determine a sentence  $\varphi \in \text{wRMSO}(\mathbb{N}\text{at}, \Sigma)$  such that  $\llbracket \varphi \rrbracket$  is not definable by a sentence without existential set quantification. Ex.

### Further Reading

- [1] J. R. Büchi. Weak second-order arithmetic and finite automata. *Z. Math. Logik Grundlagen Math.*, 6:66–92, 1960.
- [2] M. Droste and P. Gastin. Weighted automata and weighted logics. *Theoretical Computer Science*, 380(1-2):69–86, 2007. Special issue of ICALP'05.
- [3] M. Droste and P. Gastin. Weighted automata and weighted logics. In W. Kuich, H. Vogler, and M. Droste, editors, *Handbook of Weighted Automata*, EATCS Monographs in Theoretical Computer Science. Springer, 2009.
- [4] C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc.*, 98:21–52, 1961.