

# 1 Structures

## 1.1 Words

- alphabet  $\Sigma$
- $\Sigma^*, \Sigma^\omega$
- $\omega$ -word  $(\alpha : \mathbb{N} \rightarrow \Sigma) \in \Sigma^\omega$
- $\text{Occ}(\alpha) = \{\alpha(n) \mid n \in \mathbb{N}\}$
- $\text{Inf}(\alpha) = \bigcap_{i \in \mathbb{N}} \{\alpha(n) \mid n > i\}$

## 1.2 Finite Ranked Trees

- ranked alphabet  $\Sigma_0, \dots, \Sigma_m$
- trees  $T_\Sigma$
- trees (inductive definition)  $a \in \Sigma_i, t_1, \dots, t_i \in T_\Sigma \Rightarrow a(t_1, \dots, t_i) \in T_\Sigma$
- trees (labeled definition)
  - tree  $t = (\text{dom}_t, \text{val}_t)$
  - $\text{dom}_t \subseteq (\mathbb{N}_{>0}^*)$
  - $\text{val}_t : \text{dom}_t \rightarrow \Sigma$
  - if  $\text{val}_t(w) \in \Sigma_i$ , then  $wi \in \text{dom}_t$  and  $w(i+1) \notin \text{dom}_t$
  - if  $w = uv \in \text{dom}_t$ , then  $u \in \text{dom}_t$
  - if  $w(i+1) \in \text{dom}_t$ , then  $wi \in \text{dom}_t$

## 1.3 Finite Special Trees

- $S_\Sigma = \{t \in T_{\Sigma \cup \{\circ\}} \mid \circ \text{ occurs exactly once in } t\}$
- Notation,  $s \in S_\Sigma, t \in T_\Sigma, u \in \text{dom}(s)$ :
  - $s \cdot t$ : replace  $\circ$  in  $s$  by  $t$
  - $s|_u$ : subtree of  $s$  with root  $u$
  - $s[\circ/u]$ : replace  $u$  and its subtree by  $\circ$

## 1.4 Finite Unranked Trees

- alphabet  $\Sigma$
- trees  $T_\Sigma$
- trees (inductive definition)
  - $h \in (T_\Sigma)^*$  is a **hedge**
  - $a \in \Sigma, h \text{ hedge} \Rightarrow a(h) \in T_\Sigma$
- trees (labeled definition)
  - tree  $t = (\text{dom}_t, \text{val}_t)$
  - $\text{dom}_t \subseteq (\mathbb{N}_{>0}^*)$
  - $\text{val}_t : \text{dom}_t \rightarrow \Sigma$
  - if  $w = uv \in \text{dom}_t$ , then  $u \in \text{dom}_t$
  - if  $w(i+1) \in \text{dom}_t$ , then  $wi \in \text{dom}_t$

## 1.5 Infinite Trees

- alphabet  $\Sigma$
- trees  $T_\Sigma$
- tree  $t : \{0, 1\}^* \rightarrow \Sigma$
- path  $\pi \in \{0, 1\}^\omega$
- $t|_\pi \in \Sigma^\omega : n \mapsto t(\pi(0) \cdots \pi(n))$

## 1.6 Game Arenas

- arena / game graph  $G = (V_0, V_1, E, c)$ 
  - $V_0, V_1$  player vertices,  $V = V_0, V_1$
  - $E \subseteq V \times V$
  - colors  $C$  finite
  - colors  $c : V \rightarrow C$
- winning condition  $\text{Win} \subseteq C^\omega$
- game  $\mathcal{G} = (G, \text{Win})$
- play  $\alpha \in V^\omega$
- winner of  $\alpha = \begin{cases} 0 & \text{if } c(\alpha) \in \text{Win} \\ 1 & \text{else} \end{cases}$

## 2 Acceptance Conditions

$Q$  set of states/colors

$\rho \in Q^\omega$

**E**  $F \subseteq Q$ .  $\rho$  accepted iff  $F \cap \text{Occ}(\rho) \neq \emptyset$

**A**  $F \subseteq Q$ .  $\rho$  accepted iff  $\text{Occ}(\rho) \subseteq F$

**Staiger-Wagner**  $\mathcal{F} \subseteq 2^Q$ .  $\rho$  accepted iff  $\text{Occ}(\rho) \in \mathcal{F}$

**weak Parity**  $c : Q \rightarrow C$ .  $\rho$  accepted iff  $\max \text{Occ}(c(\rho))$  is even

**Büchi**  $F \subseteq Q$ .  $\rho$  accepted iff  $F \cap \text{Inf}(\rho) \neq \emptyset$

**Generalized Büchi**  $F_1, \dots, F_k \subseteq Q$ .  $\rho$  accepted iff for all  $1 \leq i \leq k$ :  $F_i \cap \text{Inf}(\rho) \neq \emptyset$

**coBüchi**  $F \subseteq Q$ .  $\rho$  accepted iff  $\text{Inf}(\rho) \subseteq F$

**Rabin**  $\Omega \subseteq 2^Q \times 2^Q$ .  $\rho$  accepted iff there is  $(E, F) \in \Omega$  s.t.  $\text{Inf}(\rho) \cap E = \emptyset$  and  $\text{Inf} \cap F \neq \emptyset$

- Rabin Chain:  $\Omega = \{(E_i, F_i) \mid 1 \leq i \leq k\}$  with  $E_k \subseteq F_k \subseteq E_{k-1} \subseteq \dots \subseteq E_1 \subseteq F_1$

**Streett**  $\Omega \subseteq 2^Q \times 2^Q$ .  $\rho$  accepted iff for all  $(E, F) \in \Omega$ :  $\text{Inf}(\rho) \cap E \neq \emptyset$  or  $\text{Inf} \cap F = \emptyset$

**Parity**  $c : Q \rightarrow C$ .  $\rho$  accepted iff  $\max \text{Inf}(c(\rho))$  is even

**Muller**  $\mathcal{F} \subseteq 2^Q$ .  $\rho$  accepted iff  $\text{Occ}(\rho) \in \mathcal{F}$

### 3 Word Automata

$(Q, \Sigma, q_0, \Delta, \text{Acc})$

$L(A) = \text{accepted words}$

**Deterministic**  $\delta : Q \times \Sigma \rightarrow Q$

**Non-Deterministic**  $\Delta \subseteq Q \times \Sigma \times Q$

**Alternating**  $\delta : Q \times \Sigma \rightarrow B_+(Q)$  where  $B_+(Q)$  are boolean formulas over  $Q \cup \{0, 1\}$  without negation

#### 3.1 Deterministic Word Automata

- DBA
- weak DBA
- coDBA
- E-automaton
- A-automaton
- Staiger-Wagner automaton
- DMA
- DRA
- DPA

#### 3.2 Nondeterministic & Alternating Word Automata

- NBA
- ABA

## 4 Finite Tree Automata

### 4.1 Ranked Top-Down Automata

$(Q, \Sigma, \Delta, F)$  where  $\Delta \subseteq \bigcup_{i=0}^m Q^i \times \Sigma_i \times Q$  or deterministic  $\delta$

- DTA
- NTA

$$\text{Size } |\mathcal{A}| = |Q| + \sum_{\tau \in \delta} |\tau|$$

### 4.2 Ranked Bottom-Up Automata

$(Q, \Sigma, Q_0, \Delta)$  where  $Q_0 \subseteq Q$  initial states,  $\Delta \subseteq \bigcup_{i=0}^m Q \times \Sigma_i \times Q^i$  or deterministic  $\delta$

- $\downarrow$ DTA ( $|Q_0| = 1$ )
- $\downarrow$ NTA

### 4.3 Unranked Bottom-Up Automata

$(Q, \Sigma, \Delta, F)$  where  $\Delta \subseteq \text{Reg}(Q) \times \Sigma \times Q$

**Normalized** For all  $(L_1, a_1, q_1), (L_2, a_2, q_2)$ : If  $a_1 = a_2$  and  $q_1 = q_2$ , then  $L_1 = L_2$

**Deterministic** For all  $(L_1, a_1, q_1), (L_2, a_2, q_2)$ :  $L_1 \cap L_2 = \emptyset$  or  $a_1 \neq a_2$  or  $q_1 \neq q_2$

- DUTA
- NUTA

### 4.4 Ranked Tree Walking Automata

Types =  $\{\text{root}, 1, \dots, m\}$

Dir =  $\{\uparrow, 0, 1, \dots, m\}$

$$\text{type}_t(u) = \begin{cases} \text{root} & \text{if } u = \epsilon \\ i & \text{if } u = vi \end{cases}$$

$(Q, \Sigma, q_0, \Delta, F)$  where  $\Delta \subseteq Q \times \text{Types} \times \Sigma \times Q \times \text{Dir}$

(current state, current type, current symbol, new state, movement)  $\in \Delta$

- TWA
- DTWA

## 4.5 Tree Transducers

- Bottom-up Tree Transducer (BTT)
- Top Down Transducer (TDT)

### BTT

$(Q, \Sigma, \Gamma, \Delta, F)$

$\Sigma, \Gamma$  input / output alphabet

$\Delta$ : tree transition &  $\varepsilon$  transition

$F \subseteq Q$

Tree with variables  $X$  at leafs:  $T_\Sigma(X)$

Tree transitions:  $f(q_1(x_1), \dots, q_n(x_n)) \rightarrow q(u)$

for some  $f \in \Sigma_n$ ,  $q, q_1, \dots, q_n \in Q$ ,  $x_1, \dots, x_n \in X$ ,  $u \in T_\Gamma(\{x_1, \dots, x_n\})$

$\varepsilon$  transitions:  $q(x) \rightarrow q'(u)$

for some  $q, q' \in Q$ ,  $x \in X$ ,  $u \in T_\Gamma(\{x_1\})$

Configurations are trees over  $\Sigma \cup \Gamma \cup Q$ .

BTTs define relations  $R(\mathcal{A}) = \{(t, t') \mid \exists q \in F : t \rightarrow_{\mathcal{A}}^* q(t')\}$ .

Transition relation:  $s \cdot f(q_1(t_1), \dots, q_n(t_n)) \rightarrow_{\mathcal{A}} s \cdot q(u[x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n])$

### TDT

Similar properties as BTT but incomparable class of relations

## 5 Infinite Tree Automata

$(Q, \Sigma, q_0, \Delta, \text{Acc})$

$\Delta \subseteq Q \times \Sigma \times Q \times Q$

$\text{Acc} \subseteq Q^\omega$

Run  $\rho : \{0, 1\}^* \rightarrow Q$  accepting iff  $\rho|_\pi \in \text{Acc}$  for all paths  $\pi$

$T(A) = \text{accepted trees}$

### 5.1 Nondeterministic

- BTA
- MTA
- PTA

### 5.2 Deterministic

$|\{(q, a, q_1, q_2) \mid q_1, q_2\}| = 1$  for all  $q, a$

- DTBA

### 5.3 Regular Tree Automata

$(Q_{\mathcal{B}}, \{0, 1\}, q_0^{\mathcal{B}}, \delta_{\mathcal{B}}, f_{\mathcal{B}})$  reads  $\{0, 1\}$ -words

$\delta_{\mathcal{B}} : Q \times \{0, 1\} \rightarrow Q$

$f_{\mathcal{B}} : Q_{\mathcal{B}} \rightarrow \Sigma$

Defines tree  $t_{\mathcal{B}} : u \mapsto f_{\mathcal{B}}(\delta_{\mathcal{B}}^*(u))$

## 6 Infinite Games

- Reachability (E-condition)
- Safety (A-condition)
- Staiger-Wagner
- weak Parity (assuming  $C \subset \mathbb{N}$ )
- Büchi
- Parity (assuming  $C \subset \mathbb{N}$ )
- Muller
- Rabin
- Streett

### 6.1 Strategy Automata

$(M, C, m_0, \sigma^u, \sigma^n)$  strategy for player 0

Memory states  $M$

Input colors  $C$

Transition / memory update function  $\sigma^u : M \times C \rightarrow M$

Next move function  $\sigma^n : M \times V_0 \rightarrow V$

Strategy  $\sigma_{\mathcal{A}}(v_0 \dots v_n) = \sigma^n((\sigma^u)^*(v_0 \dots v_{n-1}), v_n)$



## 7 Logics

### 7.1 LTL

$\varphi ::= p_i \mid \neg\varphi \mid \varphi \vee \varphi \mid X\varphi \mid F\varphi \mid G\varphi \mid \varphi U \varphi$   
 $(\alpha, i) \models \varphi$

- $\omega$ -languages
- Game winning conditions

### 7.2 FO

$\varphi ::= t_1 = t_2 \mid R\bar{t} \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\varphi \mid \forall x\varphi$

- Finite tree languages

#### Finite Tree Structure

Ranked  $\underline{t} = (\text{dom}_t, (S_i^t)_{1 \leq i \leq m}, \sqsubseteq^t, (P_a^t)_{a \in \Sigma})$   
Unranked  $\underline{t} = (\text{dom}_t, S^t, \sqsubseteq^t, <^t, (P_a^t)_{a \in \Sigma})$   
 $\sqsubseteq$  prefix order;  $<$  sibling order

### 7.3 MSO

$\varphi ::= \text{FO} + (\exists X \mid \forall X)$   
set quantification

- Finite tree languages
- S1S, S2S

### 7.4 wMSO

MSO with only finite set quantification

- $\omega$ -tree languages

### 7.5 S1S

MSO over  $(\mathbb{N}, +1, <, 0)$

- $\omega$ -languages
- Game winning conditions

#### S1S<sub>0</sub>

S1S - element variables +  $(X \subseteq Y \mid \text{Sing}(X) \mid \text{Succ}(X))$

## 7.6 S2S

MSO over  $\underline{T}_2 = (\{0, 1\}^*, \varepsilon, S_0, S_1)$

- $\omega$ -tree languages

### S2S<sub>0</sub>

S2S - element variables +  $(X \subseteq Y \mid \text{Sing}(X) \mid \text{Succ}_0(X) \mid \text{Succ}_1(X))$

## 8 Grammars

### 8.1 DTD

Document Type Definitions  $D = (\Sigma, P, S)$  with  $P \subseteq \Sigma \times \text{Reg}(\Sigma)$ .  
 $T(D)$  is the set of derivation trees starting from  $S$  with rules in  $P$ .

### 8.2 EDTD

$E = (\Sigma, P, S)$  with  $P \subseteq \Sigma' \times \text{Reg}(\Sigma')$  and  $\Sigma' = \{a^{(n)} \mid a \in \Sigma, n \in \mathbb{N}\}$ .  
 $T(E)$  is the set of derivation trees starting from  $S$  with rules in  $P$ , and replacing every  $a^{(n)}$  by  $a$ .

*single-type EDTD*: no two  $a^{(i)}$  and  $a^{(j)}$  with  $i \neq j$  occur in the same regular expression of a rule.

*deterministic EDTD*: every regular expression in a rule can be transformed to a DFA in polynomial time.

### 8.3 Regular Tree Grammar

$G = (N, \Sigma, S, P)$  with

- $N \cap \Sigma = \emptyset$
- $S \in N$
- $P \subseteq N \times T_{\Gamma}$
- $\Gamma_i = \begin{cases} \Sigma_i & \text{if } i > 0 \\ \Sigma_i \cup N & \text{else} \end{cases}$

### 8.4 Regular Tree Expressions

Inductive definition:

- Every  $t \in T_{\Sigma \cup C}$  is a regular expression.
- If  $r$  and  $s$  are regular expressions, then  $r + s$  is a regular expression. (union)
- If  $r$  and  $s$  are regular expressions and  $c \in C$ , then  $r \cdot^c s$  is a regular expression. (replace  $c$ -leafs in  $T(r)$  by trees in  $T(s)$ )
- If  $r$  is a regular expression and  $c \in C$ , then  $r^{*c}$  is a regular expression. (iteration)