Goldstine Automata

Characteristics of automata:

- Sequentiality
- Non-determinism
- Monoid
- Comm. monoid

**Def. (monoid):** A monoid is a tuple \((M, \circ, e)\) where \(M\) is a set, \(\circ\) is a binary operation on \(M\), \(M \in M\), \(\circ\) is associative, and \(e\) is neutral w.r.t. \(\circ\). A monoid is called **commutative** if \(\circ\) is commutative.

**Def. (product monoid):** Let \((M_1, \circ_1, e_1), \ldots, (M_k, \circ_k, e_k)\) be monoids. The **product of** \(M_1, \ldots, M_k\), denoted by \(M_1 \times \cdots \times M_k\), is the monoid \((M_1 \times \cdots \times M_k, \circ, e)\) where \(e = (e_1, \ldots, e_k)\) and \((m_1, \ldots, m_k) \circ (m'_1, \ldots, m'_k) = (m_1 \circ_1 m'_1, \ldots, m_k \circ_k m'_k)\).

**Ex. (free monoid):** Let \(\Sigma\) be a set. The **free monoid over** \(\Sigma\) is the monoid \((\Sigma^*, \cdot, e)\).

**Ex. (free commutative monoid):** Let \(\Sigma\) be a set. The **free commutative monoid over** \(\Sigma\) is the monoid \((\Sigma \rightarrow (M_0, \oplus, 0), \oplus, 0)\) where \(\oplus\) is the point-wise addition and \(0\) is the function that always returns 0.

**Def. (Goldstine-automaton):** Let \(M_1 \times \cdots \times M_k\) be a product monoid. An \(M_1 \times \cdots \times M_k\)-**automaton** is a finite subset of \(M_1 \times \cdots \times M_k\), the elements of which are called **transitions**.

**Def. (Semantics of Goldstine-automata):** Let \(T\) be an \(M_1 \times \cdots \times M_k \times N_1 \times \cdots \times N_k\)-automaton, \(B \subseteq M_1 \times \cdots \times M_k\), and \(\varphi : N_1 \times \cdots \times N_k \rightarrow 0\) be a homomorphism into some commutative monoid \((0, +, 0)\). The \((B, \varphi)\)-**semantics** of \(T\) is \([T]_{B, \varphi} = \varphi \left( \sum^0 (m \mid (m, n) \in [T^*], m \in B, n \neq 0) \right)\).
Goldstone Automata (examples)

Ex. (finite state automaton) Let $M = (Q, \Sigma, Q_i, Q_f, T)$ be a (usual) finite state automaton. Each transition has the form $r = (q, u, q')$ where $q, q' \in Q$ and $u \in \Sigma^* \Xi^*$. We rewrite this transition as $\tilde{r} = \langle \{ (q, q') \}, u \rangle$ where $\{ (q, q') \}$ is an element of the monoid of binary relations on the set $Q$ with $\text{Id}_Q$ as the neutral element and composition as the binary, associative operation on $Q \times Q \times Q$; and $u$ is an element of the monoid $(\Sigma^*, \circ, \varepsilon)$. Furthermore, let $B = \mathcal{P}(Q_i \times Q_f) \setminus \{ \emptyset \}$, $O = (\mathcal{P}(\Sigma^*), \cup, \emptyset)$ and $\varphi : \Sigma^* \rightarrow \mathcal{P}(\Sigma^*)$ with $\varphi : u \mapsto \{ u \}$.

The $(B, \varphi)$-semantics of the $(Q \times Q) \times \Sigma^*$-automaton $\tilde{T} = \{ \tilde{r} \mid r \in T \}$ is

$$\llbracket \tilde{T} \rrbracket_{B, \varphi} = \{ (\Sigma^* \oplus (w) \mid (r, w) \in [\tilde{T}^*], r \wedge (Q_i \times Q_f) \neq \emptyset)$$

$$= \bigcup \{ \{ w \} \mid (r, w) \in [\tilde{T}^*], r \wedge (Q_i \times Q_f) \neq \emptyset \}$$

$$= \bigcup \{ \{ u_1, \ldots, u_k \} \mid \langle \{ (q_0, q_1), \ldots, (q_{k-1}, q_k) \}, u_1, \ldots, u_k \rangle \in \tilde{T},$$

$$q_0 \in Q_i, q_k \in Q_f \}$$

exactly the language of $M$.

Ex. (weighted finite-state automaton)

- $M = (Q, \Sigma, \mu_i, \mu_f, \mu_T)$, $\mu_i, \mu_f : Q \rightarrow (A, \oplus, \emptyset, \emptyset, 1)$
- $\mu_T : Q \times (\Sigma^* \Xi^*) \times \emptyset \rightarrow A$, $A$ commutative bimonoid
- $\tilde{T} = \{ \langle \{ (q, q') \}, u, a \rangle \mid (q, u, q') \in Q \times (\Sigma^* \Xi^*) \times Q, a = \mu_T((q, u, q')) \}$
- $B = \mathcal{P}(Q_i \times Q_f) \setminus \{ \emptyset \}$
- $O = (\Sigma^* \rightarrow A, \hat{\oplus}, \text{const } 0)$
- $(f \hat{\oplus} g)(w) = f(w) \oplus g(w)$
- $\varphi : \langle u, a \rangle = \{ (u, a) \} \cup \{ (u', 0) \mid u' \in \Sigma^* \setminus \{ u \} \}$
Goldstone automata (examples)

Ex. (string transducers)

\[ \mathcal{M} = (Q, \Sigma, \Delta, Q_{i}, Q_{f}, T) \]

\[ T \subseteq \Sigma \times (\Sigma \cup \{\varepsilon\}) \times \Delta^{*} \times \Sigma \text{ finite} \]

\[ \tilde{T} = \{ (\langle q, q' \rangle, i, u, v) \mid (q, i, u, v, q') \in T \} \]

\[ B = \mathcal{P}(Q_{i} \times Q_{f}) \setminus \{\emptyset\} \]

\[ O = (\Sigma^{*} \rightarrow \mathcal{P}(\Delta^{*}), \omega, \text{const } \emptyset) \text{ where} \]

\[ (f \circ g)(\omega) = f(\omega) \cup g(\omega) \quad \forall \omega \in \Sigma^{*} \]

\[ \varphi(\langle u, v \rangle) = \{ u \} \]

\[ \mathcal{L} = \langle \Sigma^{+}, (\langle u, v \rangle) \rangle \quad \langle r, u, v \rangle \in \tilde{T} \star \cup \{ r \} \]

\[ = \{ (u, v) \mid \langle u, v \rangle \in \tilde{T} \star \cup \{ r \}, u \in \Sigma^{*} \} \]

\[ = \{ (u, v) \mid \langle u, v \rangle \in \tilde{T} \star \cup \{ r \}, v \in \Sigma^{*} \} \]

\[ = \mathcal{L} \]

Ex. (automata with storage)

\[ \mathcal{M} = (Q, \Sigma, \Delta, Q_{i}, Q_{f}, T) \]

\[ DS = (C, I, C_{i}, C_{f}) \]

\[ T \subseteq Q \times I \times (\Sigma \cup \{\varepsilon\}) \times Q \text{ finite} \]

\[ \tilde{T} = \{ (\langle q, q' \rangle, i, u) \mid (q, i, u, q') \in T \} \]

\[ B = \mathcal{P}(Q_{i} \times Q_{f}) \setminus \{\emptyset\} \times \mathcal{P}(C_{i} \times C_{f}) \setminus \{\emptyset\} \]

\[ O = (\Sigma^{*}, \omega, \emptyset) \]

\[ \varphi(\langle u \rangle) = \{ u \} \]
Goldstine automata (genealogy)

Parameters I: internal behaviour

- automaton exhibits state behaviour (over finite set \(Q\))
  \[ \rightarrow \text{state transition } q \rightarrow q' \text{ becomes a singleton} \]
  \[ \text{binary relation } \{(q, q')\} \subseteq Q \times Q \]

- automaton exhibits storage behaviour (over set \(C\))
  \[ \rightarrow \text{storage instructions can be directly taken over} \]
  \[ \rightarrow \text{storage predicates become partial identities} \]

Parameters II: output behaviour

- automaton models a language \((\in \Sigma^*)\)
  \[ \rightarrow O = (\mathcal{P}(\Sigma^*), \nu, \emptyset) \]
  \[ \rightarrow \nu(\langle u \rangle) = \{ u \} \]

- automaton models a transduction \((\in \Sigma^* \times \Delta^*)\)
  \[ \rightarrow O = (\mathcal{P}(\Sigma^* \times \Delta^*), \nu, \emptyset) \]
  \[ \rightarrow \nu(\langle u, v \rangle) = \{ (u, v) \} \]

- automaton models a weighted language \((\in \Sigma^* \rightarrow A)\)
  \[ \rightarrow O = (\Sigma^* \rightarrow A, \Theta, \text{const } \emptyset) \]
  \[ (s_1 \oplus s_2)(w) = s_1(w) + s_2(w) \]
  \[ \rightarrow \nu(\langle u, a \rangle) = (\text{const } \emptyset)[u/a] \]

- automaton models a weighted transduction \((\in \Sigma^* \times \Delta^* \rightarrow A)\)
  \[ \rightarrow O = (\Sigma^* \times \Delta^* \rightarrow A, \Theta, \text{const } \emptyset) \]
  \[ (s_1 \oplus s_2)(u, v) = s_1(u, v) + s_2(u, v) \]
  \[ \rightarrow \nu(\langle u, v, a \rangle) = (\text{const } \emptyset)[(u, v)/a] \]

Transitions:

\[ \Gamma \subseteq \mathcal{M}_I \times \ldots \times \mathcal{M}_k \times \mathcal{N}_i \times \mathcal{N} \text{out} \times \mathcal{N} \text{out} \]

- internal behaviour (present if relevant)
- demonstrate usefulness (by exhibiting proofs)
  - proofs of “algebraic” properties should benefit
  - usual proof-techniques could be translated
  - hopefully find new proof-techniques

- extend to the tree case
  - only bottom-up?
  - weighted/weighted tree automata/transducers
  - with/without storage

characteristics: branching and non-determinism

- extend to the unranked tree case

characteristics: branching and non-determinism

Problems/questions (from the discussion)

- $M \times N$ enough for syntax of Goldstine automata?
- $\{ T^* \} = \langle T, 0, 1 \rangle$ (submonoid generated by $T$)
- maybe even one monoid is enough.
- is this more than group theory?