Finite automata and formal languages (DIT322, TMV028)

Nils Anders Danielsson

2020-03-05

Today

- Pushdown automata.
- Turing machines.

Pushdown automata

Pushdown automata

- The class of regular languages can be defined using regular expressions or different kinds of automata.
- Is there a class of automata that defines the context-free languages?

Pushdown automata

A pushdown automaton (PDA) can be given as a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$:

- A finite set of states (Q).
- An alphabet (Σ with $\varepsilon \notin \Sigma$).
- A stack alphabet (Γ).
- A transition function ($\delta \in Q \times (\{\, \varepsilon \,\} \cup \Sigma^1) \times \Gamma \to \wp(Q \times \Gamma^*)$).
- A start state ($q_0 \in Q$).
- A start symbol ($Z_0 \in \Gamma$).
- A set of accepting states ($F \subseteq Q$).

Pushdown automata

An *instantaneous description* (ID) for a given PDA is a triple (q, w, γ):

- The current state $(q \in Q)$.
- The remainder of the input string $(w \in \Sigma^*)$.
- The current stack $(\gamma \in \Gamma^*)$.

Pushdown automata

The following relation between IDs defines what kinds of transitions are possible:

$$\frac{u \in \{\, \varepsilon \,\} \cup \Sigma^1 \quad (q, \alpha) \in \delta(p, u, Z)}{(p, uv, Z\gamma) \vdash (q, v, \alpha\gamma)}$$

The reflexive transitive closure of $\vdash$ can be defined inductively:

$$\frac{}{I \vdash^* I} \qquad\qquad \frac{I \vdash J \quad J \vdash^* K}{I \vdash^* K}$$

Consider the PDA
$P = (\{q\}, \{0,1\}, \{A,B\}, \delta, q, B, \{q\})$, where δ is defined in the following way:

$$\begin{aligned} \delta(q,\varepsilon,A) &= \{(q,\varepsilon)\} & \delta(q,\varepsilon,B) &= \{(q,BA)\} \\ \delta(q,0,A) &= \emptyset & \delta(q,0,B) &= \{(q,\varepsilon)\} \\ \delta(q,1,A) &= \emptyset & \delta(q,1,B) &= \{(q,AB)\} \end{aligned}$$

Which of the following propositions are true for P?

1. $(q, 01, AB) \vdash^* (q, \varepsilon, \varepsilon)$
2. $(q, 01, AB) \vdash^* (q, \varepsilon, AAA)$
3. $(q, 01, AB) \vdash^* (q, 1, \varepsilon)$
4. $(q, 01, AB) \vdash^* (q, 1, AAA)$

Pushdown automata

The language of a PDA:

$$L((Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)) =$$
$$\{\, w \in \Sigma^* \mid q \in F, \gamma \in \Gamma^*, (q_0, w, Z_0) \vdash^* (q, \varepsilon, \gamma) \,\}$$

Consider the PDA
$P = (\{q\}, \{0, 1\}, \{A, B\}, \delta, q, B, \{q\})$ again, where δ is still defined in the following way:

$$\begin{aligned} \delta(q, \varepsilon, A) &= \{(q, \varepsilon)\} & \delta(q, \varepsilon, B) &= \{(q, BA)\} \\ \delta(q, 0, A) &= \emptyset & \delta(q, 0, B) &= \{(q, \varepsilon)\} \\ \delta(q, 1, A) &= \emptyset & \delta(q, 1, B) &= \{(q, AB)\} \end{aligned}$$

Which of the following strings are members of $L(P)$?

1. 00
2. 01
3. 10
4. 11

Pushdown automata

Another way to define the language of a PDA:

$$N((Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)) = \\ \{\, w \in \Sigma^* \mid q \in Q, (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon) \,\}$$

The following property holds for every language L over Σ:

$$(\exists \text{a PDA } P.\, L(P) = L) \Leftrightarrow (\exists \text{a PDA } P.\, N(P) = L)$$

Grammars and automata

For any alphabet Σ (with $\varepsilon \notin \Sigma$) and language $L \subseteq \Sigma^*$ one can prove that the following two statements are equivalent:

- There is a context-free grammar G, with Σ as its set of terminals, satisfying $L(G) = L$.
- There is a pushdown automaton P with alphabet Σ satisfying $L(P) = L$.

Grammars and automata

Given a context-free grammar $G = (N, \Sigma, P, S)$, we can construct the PDA $Q = (\{\ q\ \}, \Sigma, N \cup \Sigma, \delta, q, S, \{\ q\ \})$, where δ is defined in the following way:

$$\begin{aligned}
\delta(q, \varepsilon, A) &= \{\ (q, \alpha) \mid A \to \alpha \in P\ \} \\
\delta(q, a, a) &= \{\ (q, \varepsilon)\ \} \\
\delta(q, _, _) &= \emptyset
\end{aligned}$$

Which of the following statements is true for $G = (\{ S \}, \{ 0, 1 \}, (S \to 0S \mid 1), S)$?

1. $L(G) = L(Q)$.
2. $L(G) = N(Q)$.

Turing machines

Turing machines

- Simple computers.
- An idealised model of what it means to "compute".

Intuitive idea

- A tape that extends arbitrarily far in both directions.
- The tape is divided into squares.
- The squares can be blank or contain symbols, chosen from a finite alphabet.
- A read/write head, positioned over one square.
- The head can move from one square to an adjacent one.
- Rules that explain what the head does.

Rules

- A finite set of states.
- When the head reads a symbol
  (blank squares correspond to a special symbol):
    - Check if the current state contains a matching rule, with:
        - A symbol to write.
        - A direction to move in.
        - A state to switch to.
    - If not, halt.

The Church-Turing thesis

- Turing motivated his design partly by reference to what a human computer does.
- The Church-Turing thesis:
  Every effectively calculable function
  on the positive integers can be computed
  using a Turing machine.
- "Effectively calculable function" is not a well-defined concept, so this is not a theorem.

Syntax

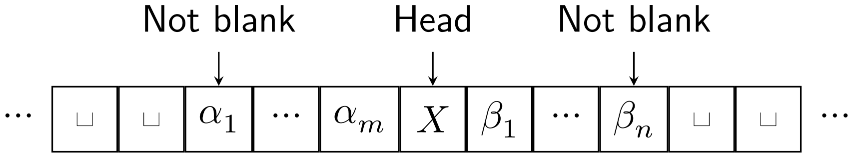
A Turing machine (TM) can be given as a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F)$:

- A finite set of states (Q).
- An input alphabet (Σ).
- A tape alphabet (Γ with $\Sigma \subseteq \Gamma$).
- A (partial) transition function ($\delta \in Q \times \Gamma \rightharpoonup Q \times \Gamma \times \{ \mathsf{L}, \mathsf{R} \}$).
- A start state ($q_0 \in Q$).
- A blank symbol ($\sqcup \in \Gamma \setminus \Sigma$).
- A set of accepting states ($F \subseteq Q$).

Instantaneous descriptions

An *instantaneous description* (ID) for a given TM is a 4-tuple (α, q, X, β), often written $\alpha q X \beta$:

- The current state ($q \in Q$).
- The non-blank portion of the tape ($X \in \Gamma$, $\alpha, \beta \in \Gamma^*$).



Transition relation

The following relation between IDs defines what kinds of transitions are possible:

$$\frac{\delta(p, X) = (q, Y, \mathsf{R})}{(\alpha, p, X, Z\beta) \vdash (l(\alpha Y), q, Z, \beta)}$$

$$\frac{\delta(p, X) = (q, Y, \mathsf{R})}{(\alpha, p, X, \varepsilon) \vdash (l(\alpha Y), q, \sqcup, \varepsilon)}$$

The function l removes leading blanks.

Transition relation

$$\frac{\delta(p, X) = (q, Y, \mathsf{L})}{(\alpha Z, p, X, \beta) \vdash (\alpha, q, Z, r(Y\beta))}$$

$$\frac{\delta(p, X) = (q, Y, \mathsf{L})}{(\varepsilon, p, X, \beta) \vdash (\varepsilon, q, \sqcup, r(Y\beta))}$$

The function r removes trailing blanks.

Transition relation

The reflexive transitive closure of $\vdash$ can be defined inductively:

$$\frac{}{I \vdash^* I} \qquad\qquad \frac{I \vdash J \quad J \vdash^* K}{I \vdash^* K}$$

Consider the TM
$M = (\{ p, q \}, \{ 0, 1 \}, \{ 0, 1, \sqcup \}, \delta, p, \sqcup, \emptyset)$, where
δ is defined in the following way:

$$\delta(p, \sqcup) = (q, \sqcup, \mathsf{L})$$

$$\delta(p, 0) = (p, 1, \mathsf{R}) \qquad \delta(q, 0) = (q, 0, \mathsf{L})$$

$$\delta(p, 1) = (p, 0, \mathsf{R}) \qquad \delta(q, 1) = (q, 1, \mathsf{L})$$

Which of the following statements are true for M?

1. $p01 \vdash^* 10p\sqcup$
2. $p01 \vdash^* q\sqcup 10$
3. $p01 \vdash^* q\sqcup\sqcup 10$
4. $p111 \vdash^* 00p1$
5. $p111 \vdash^* 00q1$
6. $p111 \vdash^* 0q00$

Language

The language of a TM:

$$L((Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F)) = \left\{ w \in \Sigma^* \,\middle|\, \begin{array}{l} q \in F, X \in \Gamma, \alpha, \beta \in \Gamma^*, \\ q_0 w \vdash^* \alpha q X \beta \end{array} \right\}$$

(Here $q_0\varepsilon$ means $q_0\sqcup$.)

Halting

- Turing machines can fail to halt ($I_0 \vdash I_1 \vdash \ldots$).
- A language is called *recursively enumerable* if it is the language of some Turing machine.
- A language is called *recursive* if it is the language of some Turing machine that always halts.
- There are languages that are recursively enumerable but not recursive.
- An example: The language of (strings representing) Turing machines that halt when given the empty string as input.

A hierarchy

A hierarchy of languages over the alphabet Σ (if $|\Sigma| \geq 2$):

$$
\begin{array}{ll}
\text{Finite} & \subsetneq \\
\text{Regular} & \subsetneq \\
\text{Context-free} & \subsetneq \\
\text{Recursive} & \subsetneq \\
\text{Recursively enumerable} & \subsetneq \\
\wp(\Sigma^*) &
\end{array}
$$

Some undecidable problems

The following things cannot, in general, be determined (using, say, a Turing machine that always halts):

- If a Turing machine halts for a given input.
- If two Turing machines accept the same language.
- ...

Consider the TM
$M = (\{ p, q, r \}, \{ 1 \}, \{ 1, \sqcup \}, \delta, p, \sqcup, \{ r \})$,
where δ is defined in the following way:

$$\delta(p, \sqcup) = (r, \sqcup, \mathsf{R})$$
$$\delta(p, 1) = (q, \sqcup, \mathsf{R})$$
$$\delta(q, 1) = (p, \sqcup, \mathsf{R})$$

Which of the following strings are members of $L(M)$? Does M always halt?

1. ε
2. 1
3. 11
4. 111
5. 1111
6. It always halts

Today

- Pushdown automata.
- Turing machines.

Next lecture

- A summary of the course.