Recognizing Languages II

Discrete Mathematics

Evgeny Skvortsov

Previous Lecture

- Finite automata
- Transducers

Transducers

- Sometimes we want a finite automaton to output something
- A transducer is a finite automaton with two extra features
  - an output alphabet, Δ
  - the transition function is a collection of rules of the form

    $(s,a) \rightarrow (s',a')$

  where s is the current state of the automaton, $a \in \Sigma$ is an input symbol, s' is the new state, and $a' \in \Delta$ is an output symbol

- At each step, when applying a rule $(s,a) \rightarrow (s',a')$ the automaton not only changes the current state, but also outputs a', so producing a string over the output alphabet

Example

- Describe the work of the automaton:

  $S = \{ s_0, s_1, s_2 \}$, $\Sigma = \{a,b\}$, $\Delta = \{a,b\}$, s_0 is the initial state

  $v = \{ (s_0,a) \to (s_0,a),\ (s_0,b) \to (s_1,a),$
  $(s_1,a) \to (s_2,a),\ (s_1,b) \to (s_1,a),$
  $(s_2,a) \to (s_0,a),\ (s_2,b) \to (s_1,b) \}$

  on input babaab

Another Example

- Serial binary adder: an automaton that given binary expansions of two integers outputs the expansion of their sum. Assume that the least significant bits of the expansions go first.

What Finite Automata Cannot Do

- Consider the language $L = \{ a^k b^k \mid k \in \mathbf{N} \}$.
- If we try to construct a recognizing finite automaton for this language we find the most difficult thing is to synchronize the powers of a and b

Theorem

L is not accepted by any finite automaton.

Proof

By contradiction.

Suppose that L is accepted by a finite automaton

$M = (S, \Sigma, v, s_0, F)$, and $|S| = n$.

Proof (cntd)

- Run the automaton on input $a^{n+1}b^{n+1}$

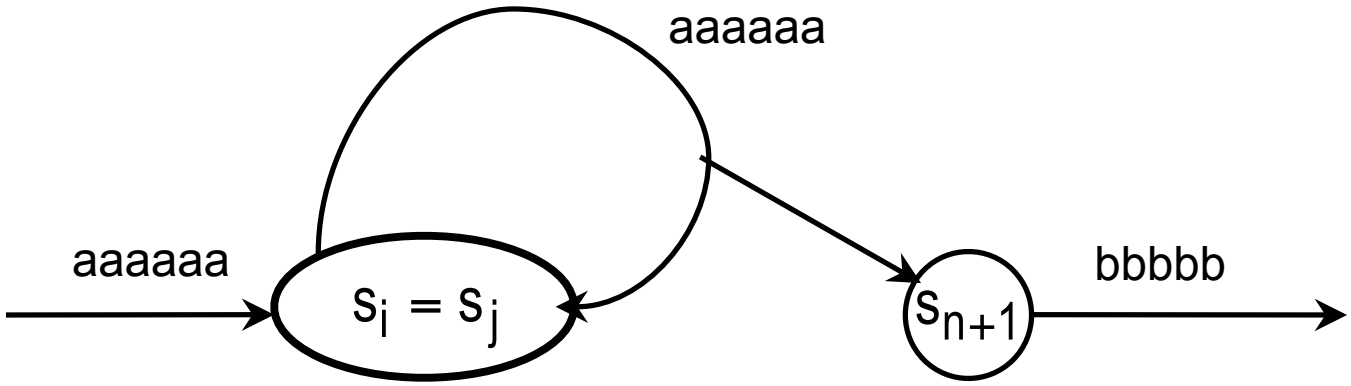
  The automaton accepts this input

  While working on the input it goes through some states. Let us denote the sequence of states by

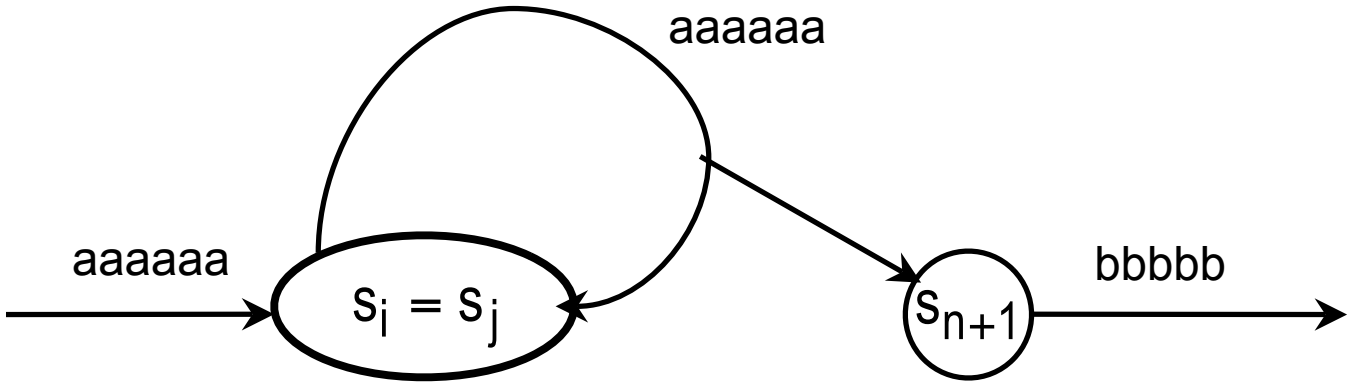
  input a a a a a … a b …

  states $s_1\ s_2\ s_3\ s_4\ s_5\ \dots\ s_{n+1}\ \dots$

  Since $|S| = n$, by the pigeonhole principle $s_i = s_j$ for some $i \neq j$



Proof (cntd)



- The loop in the picture contains $j - i$ transitions
  Consider the automaton on input $a^{n+1+(j-i)}b^{n+1}$
  This time it goes around the loop one time more, but still arrives to the same state s_{n+1} with the same input
  Thus the automaton accepts.

Kleene Theorem

Theorem

A language is accepted by a finite automaton if and only if it can be represented by a regular expression

Proof (small part)

We only partially prove that any language that can be represented by a regular expression is accepted by a finite automata

Definition of regular expressions:

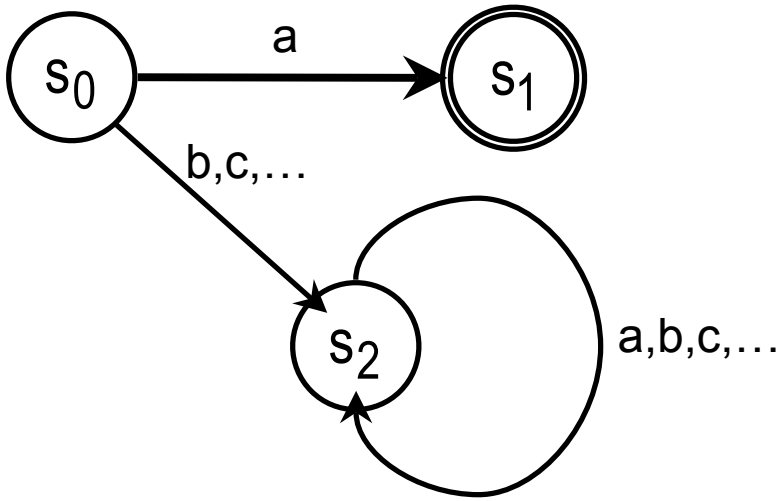
B.S. Atomic languages are regular expressions

I.S. If E and F are regular expressions then $\overline{E}$, $E \cap F$, $E \cup F$, EF (concatenation), and E^* are regular expressions

Since regular expressions are defined inductively so will be our proof

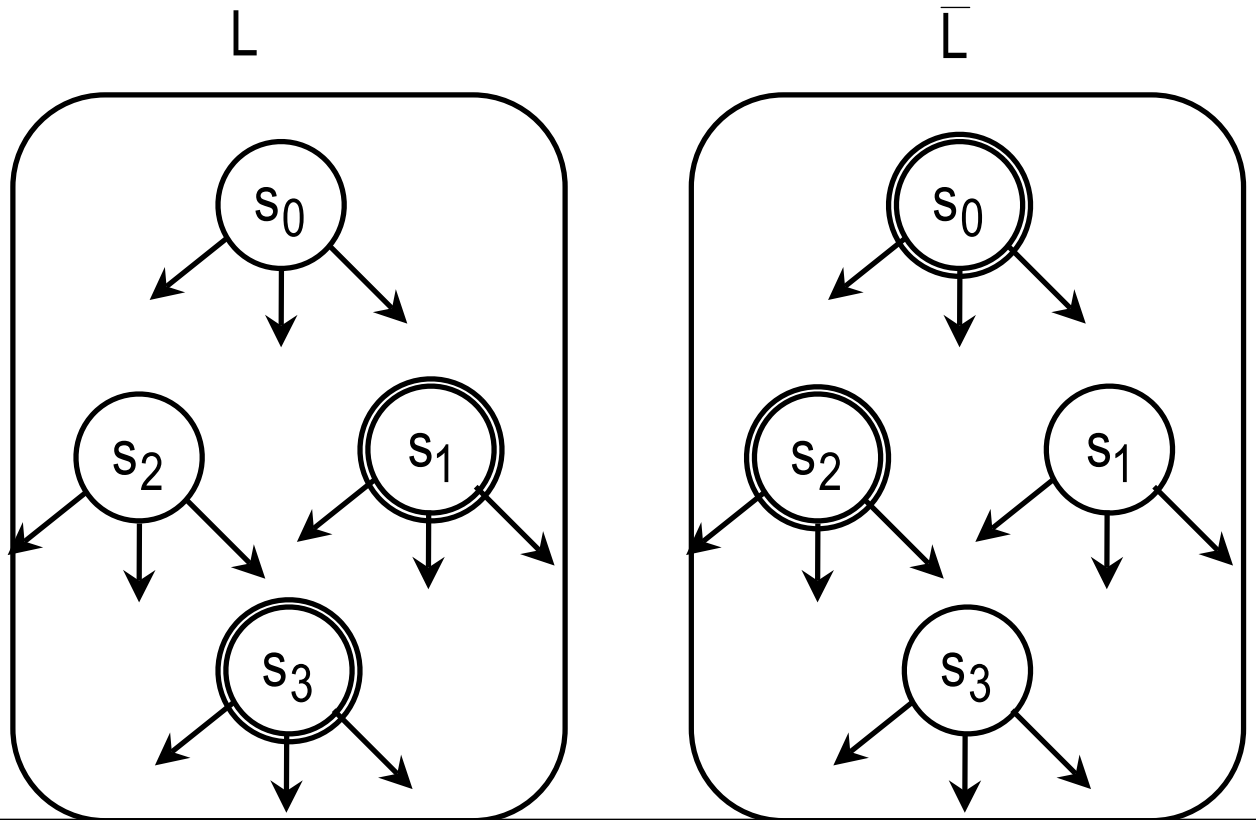
Kleene Theorem: Atomic Languages

- Let L = {a} be an atomic language over an alphabet Σ = {a,b,c,…}
  The following automaton accepts L



Kleene Theorem: Complement

- Suppose that language L is represented by regular expression E and accepted by a finite automaton M
- Clearly, $\overline{E}$ represents $\overline{L}$. What is an automaton accepting $\overline{L}$?



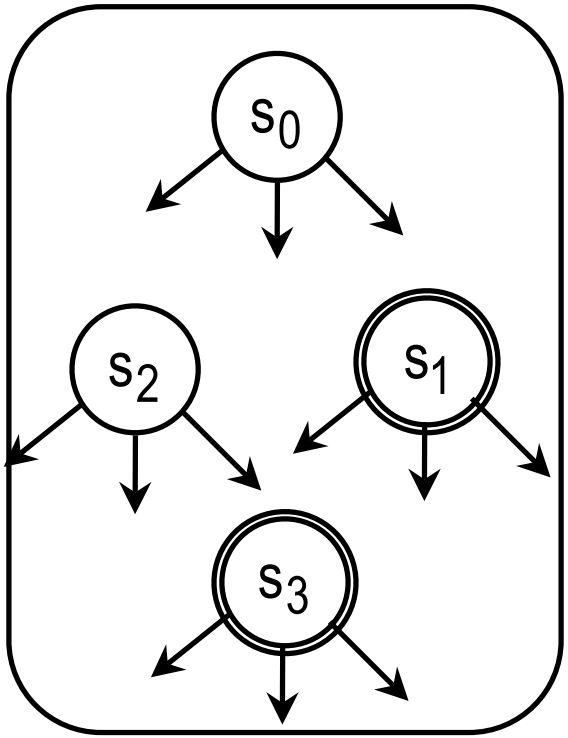
Kleene Theorem: Complement (cntd)

- Formally, if $M = (S, \Sigma, v, s_0, F)$, then the automaton accepting $\overline{L}$ is $(S, \Sigma, v, s_0, S - F)$,

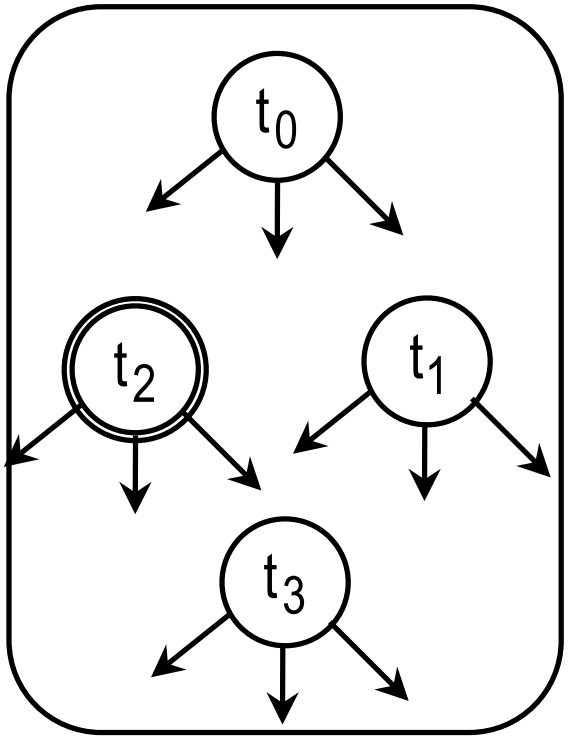
Kleene Theorem: Intersection

- Let E and F be regular expressions representing languages L and L'. The expression $E \cap F$ represents $L \cap L'$.
- If L and L' are accepted by automata M and M', then what is the automaton accepting $L \cap L'$?

M:

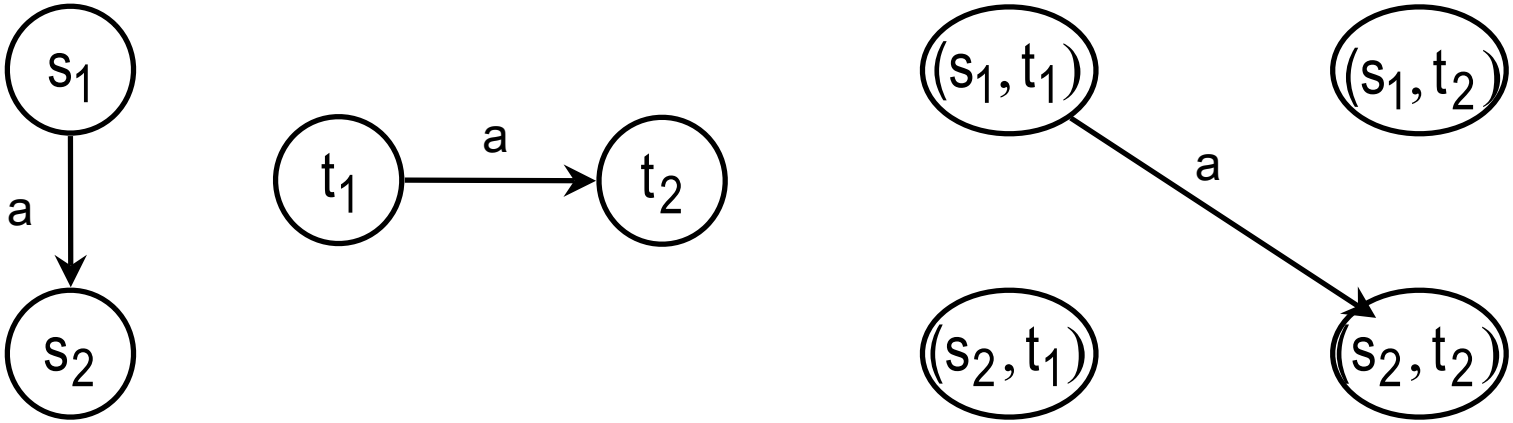


M':



Kleene Theorem: Product of Automata

- Let $M = (S, \Sigma, v, s_0, F)$, and $M' = (T, \Sigma, v', t_0, F')$ be finite automata.
- The product of M and M' is the finite automaton with
  - $S \times T$ the set of internal states
  - Σ the input alphabet
  - $v \times v'$ the transition function
  - (s_0, t_0) the initial state



Kleene Theorem: Product of Automata (cntd)

- Formally, $v \times v'$ consists of rules of the form

  $$((s,t),a) \rightarrow (s',t')$$

  where $(s,a) \rightarrow s'$ is a rule of M, and $(t,a) \rightarrow t'$ is a rule of M'

Lemma

If after reading a string w the automaton M ends up in the state s, and the automaton M' in the state t, then $M \times M'$ ends up in the state (s,t).

Proof.

Induction on the length of w.

B.S.: if $w = \lambda$ then M, M', and $M \times M'$ end up in the states $s_0, t_0,$ and (s_0, t_0), respectively

Kleene Theorem: Product of Automata (cntd)

I.S.: Take a string w and let w' be the prefix of w containing all its symbols except for the last one.

By the induction hypothesis if M and M' end up working on w' in the states s and t, respectively, then $M \times M'$ ends up in the state (s,t)

Suppose that the last symbol of w is a, and v and v' contain rules $(s,a) \rightarrow s'$ and $(t,a) \rightarrow t'$, respectively

Then $M \times M'$ contains the rule $((s,t),a) \rightarrow (s',t')$.

Thus M and M' end up working on w in the states s' and t', respectively, while $M \times M'$ ends up in the state (s',t')

Q. E. D.

Kleene Theorem: Intersection and Union

- For the intersection: a string is accepted if and only if both automata accept it.

  So, the set of accepting states of $M \times M'$ should be

  $\{ (s,t) \mid s \in F \text{ and } t \in F' \}$,

  that is s is an accepting state of M, and t is an accepting state of M'

- For the union: a string is accepted if and only if one of the automata accepts it.

  So, the set of accepting states of $M \times M'$ should be

  $\{ (s,t) \mid s \in F \text{ or } t \in F' \}$,

  that is s is an accepting state of M, or t is an accepting state of M'

Homework

Exercises from the Book:

No. 1, 3, 5 (page 324 – 325)