Linear Bounded Automata

Indu John

Department of Computer Science and Automation
Indian Institute of Science, Bangalore

December 1, 2011

Overview

Overview

- Definition

Overview

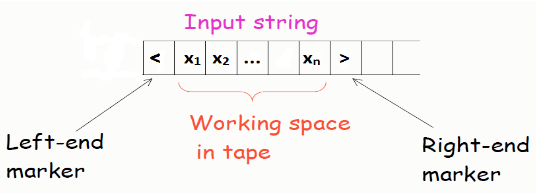
- Definition
- Results about LBAs

Overview

- Definition
- Results about LBAs
- CSLs and LBAs

Definition

A Turing machine that uses only the tape space occupied by the input is called a linear-bounded automaton (LBA).



- A linear bounded automaton is a nondeterministic Turing machine $M = (Q, \Sigma, \Gamma, \delta, s, t, r)$ such that:
  - There are two special tape symbols $<$ and $>$(the left end marker and right end marker).
  - The TM begins in the configuration $(s, < x >, 0)$.
  - The TM cannot replace $<$ or $>$ with anything else, nor move the tape head left of $<$ or right of $>$.

LBA

- An equivalent definition of an LBA is that it uses only k times the amount of space occupied by the input string, where k is a constant fixed for the particular machine.

LBA

- An equivalent definition of an LBA is that it uses only k times the amount of space occupied by the input string, where k is a constant fixed for the particular machine.
- Possible to simulate k tape cells with a single tape cell, by increasing the size of the tape alphabet

LBA

- An equivalent definition of an LBA is that it uses only k times the amount of space occupied by the input string, where k is a constant fixed for the particular machine.
- Possible to simulate k tape cells with a single tape cell, by increasing the size of the tape alphabet
- **Examples**: $\{a^n b^n c^n | n \geq 0\}$; counting number of a's

LBA

- An equivalent definition of an LBA is that it uses only k times the amount of space occupied by the input string, where k is a constant fixed for the particular machine.
- Possible to simulate k tape cells with a single tape cell, by increasing the size of the tape alphabet
- **Examples**: $\{a^n b^n c^n | n \geq 0\}$; counting number of a's
- This limitation makes the LBA a somewhat more accurate model of computers that actually exist than a Turing machine, whose definition assumes unlimited tape.

History

History

- In 1960, Myhill introduced an automaton model today known as deterministic linear bounded automaton.

History

- In 1960, Myhill introduced an automaton model today known as deterministic linear bounded automaton.
- Shortly thereafter, Landweber proved that the languages accepted by a deterministic LBA are always context-sensitive.

History

- In 1960, Myhill introduced an automaton model today known as deterministic linear bounded automaton.
- Shortly thereafter, Landweber proved that the languages accepted by a deterministic LBA are always context-sensitive.
- In 1964, Kuroda introduced the more general model of (nondeterministic) linear bounded automata, and showed that the languages accepted by them are precisely the context-sensitive languages.

Number of configurations

- Suppose that a given LBA M has
  - q states,
  - m characters in the tape alphabet ,
  - and the input length is n.
- Then M can be in at most

$$\alpha(n) = \overbrace{m^n}^{Tapecontents} * \overbrace{n}^{Headposition} * \overbrace{q}^{State}$$

configurations.

Results about LBA

Halting Problem

The halting problem is solvable for linear bounded automata.

- $Halt_{LBA} = \{< M, w > |$M is an LBA and M halts on w$\}$ is decidable.
- An LBA that stops on input w must stop in at most $\alpha(|w|)$ steps

Results about LBA

Halting Problem

The halting problem is solvable for linear bounded automata.

- $Halt_{LBA} = \{< M, w > |$M is an LBA and M halts on w$\}$ is decidable.
- An LBA that stops on input w must stop in at most $\alpha(|w|)$ steps

Membership problem

The membership problem is solvable for linear bounded automata.

- $A_{LBA} = \{< M, w > |$M is an LBA and M accepts w$\}$ is decidable.

Results about LBA

Emptiness Problem

The emptiness problem is unsolvable for linear bounded automata.

- For every Turing machine there is a linear bounded automaton which accepts the set of strings which are valid halting computations for the Turing machine.

LBA Problems

LBA Problems

1. Is the class of languages accepted by LBA equal to the class of languages accepted by deterministic LBA?

LBA Problems

1. Is the class of languages accepted by LBA equal to the class of languages accepted by deterministic LBA?
    - Open Problem!

LBA Problems

1. Is the class of languages accepted by LBA equal to the class of languages accepted by deterministic LBA?
   - Open Problem!
2. Is the class of languages accepted by LBA closed under complement?

LBA Problems

1. Is the class of languages accepted by LBA equal to the class of languages accepted by deterministic LBA?
   - Open Problem!
2. Is the class of languages accepted by LBA closed under complement?
   - Yes. (*Immerman Szelepcsenyi* Theorem)

LBAs and CSLs

Theorem(Landweber-Kuroda)

A language is accepted by an LBA iff it is context sensitive.

Proof

$\Leftarrow$

If L is a CSL, then L is accepted by some LBA.

- Let $G = (N, \Sigma, S, P)$ be the given grammar such that $L(G) = L$.
- Construct LBA M with tape alphabet $\Sigma \times \{N \cup \Sigma\}$(2- track machine)
- First track holds input string w
- Second track holds a sentential form α of G, initialized to S.

Proof...

- If $w = \epsilon$, M halts without accepting.
- Repeat :
  1. Non-deterministically select a position i in α.
  2. Non-deterministically select a production $\beta \rightarrow \gamma$ of G.
  3. If β appears beginning in position i of α, replace β by γ there.
     - If the sentential form is longer than w, LBA halts without accepting.
  4. Compare the resulting sentential form with w on track 1. If they match, accept. If not go to step 1.

Proof...

$\Rightarrow$

If there is a linear bounded automaton M accepting the language L, then there is a context sensitive grammar generating $L - \{\epsilon\}$.

Proof...

$\Rightarrow$

If there is a linear bounded automaton M accepting the language L, then there is a context sensitive grammar generating $L - \{\epsilon\}$.

- *Sketch of proof*:
- Derivation simulates moves of LBA

Proof...

$\Rightarrow$

If there is a linear bounded automaton M accepting the language L, then there is a context sensitive grammar generating $L - \{\epsilon\}$.

- *Sketch of proof*:
- Derivation simulates moves of LBA
- Three types of productions
  1. Productions that can generate two copies of a string in Σ^*, along with some symbols that act as markers to keep the two copies separate.
  2. Productions that can simulate a sequence of moves of M. During this portion of a derivation, one of the two copies of the original string is left unchanged; the other, representing the input tape to M, is modified accordingly.
  3. Productions that can erase everything but the unmodified copy of the string, provided that the simulated moves of M applied to the other copy cause M to accept.

References

John Martin, "*Introduction to Languages and the Theory of Computation*" , Tata McGraw-Hill, Third Edition.

John Hopcroft, Jeffery Ullman, "*Introduction to Automata Theory, Languages and Computation*" .

http://www.cs.uky.edu/~lewis/texts/theory/automata/lb-auto.pdf

http://en.wikipedia.org/wiki/Linear_bounded_automaton

http://www.fit.vutbr.cz/~janousek/vyuka/vsl2/newer/class19.pdf