Computability and Complexity

Lecture 2

Multitape Turing machines
Nondeterministic Turing machines
Enumerators
Church-Turing Thesis

given by Jiri Srba

Variants of Turing Machines

Question

What happens if we modify the definition of a Turing machine? Can we then possibly recognize more languages?

Example: $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$ where S means "stay".

Variants of Turing Machines

Question

What happens if we modify the definition of a Turing machine? Can we then possibly recognize more languages?

Example: $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$ where S means "stay".

Answer

No, the notion of a TM is robust. Hence no reasonable extension of a TM increases its power.

Example: "Stay" can be simulated in ordinary TM by two head movements (move right, move left).

Multitape Turing Machine

Definition (Multitape Turing Machine)

A *k*-tape TM is a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ where

$$\delta : Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R\}^k$$

and the rest is the same as before.

Remark: 1-tape TM is exactly our original definition of TM.

- Configuration: a control state, plus the content of all k tapes together with the position of k heads.
- Initial configuration: input string written on the first tape, all other tapes are empty (contain the blank symbols).
- Computational step: all heads can move independently.

Equivalence of 1-tape and k-tape TM

Theorem

Every k-tape TM is equivalent to 1-tape TM.

Equivalent means that they recognize the same language.

Equivalence of 1-tape and k-tape TM

Theorem

Every k-tape TM is equivalent to 1-tape TM.

Equivalent means that they recognize the same language.

Corollary

A language is recognizable iff it is recognized by some multitape Turing machine.

Corollary

A language is decidable iff it is recognized by some multitape Turing machine which is a decider.

Nondeterministic Turing Machine

Definition (Nondeterministic Turing Machine)

A nondeterministic TM is a 7-tuple
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ where

$$\delta : Q \times \Gamma \to 2^{Q \times \Gamma \times \{L,R\}}$$

and the rest is the same as before.

Remark: Every deterministic TM is also a nondeterministic TM.

Nondeterministic Turing Machine

Definition (Nondeterministic Turing Machine)

A nondeterministic TM is a 7-tuple
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ where

$$\delta : Q \times \Gamma \to 2^{Q \times \Gamma \times \{L,R\}}$$

and the rest is the same as before.

Remark: Every deterministic TM is also a nondeterministic TM.

- Configurations and Initial configuration as before.
- Computation tree: a tree of all configurations reachable from the initial one. The tree can have infinite branches!

Nondeterministic Turing Machine

Definition (Nondeterministic Turing Machine)

A nondeterministic TM is a 7-tuple
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ where

$$\delta : Q \times \Gamma \to 2^{Q \times \Gamma \times \{L,R\}}$$

and the rest is the same as before.

Remark: Every deterministic TM is also a nondeterministic TM.

- Configurations and Initial configuration as before.
- Computation tree: a tree of all configurations reachable from the initial one. The tree can have infinite branches!

Acceptance Condition of a Nondeterm. TM

A nondeterministic TM M accepts a string w if the computation tree for M and w contains at least one accepting configuration.

Equivalence of Deterministic and Nondeterministic TM

Theorem

Every nondeterministic TM is equivalent to some deterministic TM.

Equivalent means that they recognize the same language.

Equivalence of Deterministic and Nondeterministic TM

Theorem

Every nondeterministic TM is equivalent to some deterministic TM.

Equivalent means that they recognize the same language.

Corollary

A language is recognizable iff it is recognized by some nondeterministic Turing machine.

Equivalence of Deterministic and Nondeterministic TM

Theorem

Every nondeterministic TM is equivalent to some deterministic TM.

Equivalent means that they recognize the same language.

Corollary

A language is recognizable iff it is recognized by some nondeterministic Turing machine.

Corollary

A language is decidable iff it is recognized by some nondeterministic Turing machine which is a decider.

A nondeterministic TM is a decider if for any given input every branch in the computation tree is finite (accepts or rejects).

Enumerators

Another terminology for recognizable languages is the term
Recursively Enumerable languages. Why?

Enumerators

Another terminology for recognizable languages is the term
Recursively Enumerable languages. Why?

Definition (Enumerator)

An enumerator is a 2-tape Turing machine with a special control state called q_{print}.

Definition (Language Generated by Enumerator)

Let E be an enumerator. We run E on the empty string as input. The language of E, denoted by $L(E)$, is the collection of all strings that are on the second tape whenever E is in the state q_{print}.

Remark: If E does not terminate then $L(E)$ may be infinite.
Strings in $L(E)$ may repeat and may be printed in arbitrary order.

Theorem about Enumerators

Theorem

A language L is recognizable if and only if there exists an enumerator E that enumerates L, i.e. $L(E) = L$.

Proof (Enumerable $\implies$ Recognizable)

Every enumerable language L is recognizable:

Let E be an enumerator for L. We construct a recognizer M for L.

$M =$ " On input w:

1. Run E.
2. If w gets ever printed then M accepts,
   otherwise continue running E in step 1."

Proof (Recognizable $\implies$ Enumerable)

Every recognizable language L is enumerable:

Let M be a recognizer for L. We construct an enumerator E for L.

Let $s_1, s_2, s_3, \ldots$ be all possible strings from Σ^*.

Proof (Recognizable $\implies$ Enumerable)

Every recognizable language L is enumerable:

Let M be a recognizer for L. We construct an enumerator E for L.

Let $s_1, s_2, s_3, \ldots$ be all possible strings from Σ^*.

$E =$ "
1. $i := 1$;
2. Run M for i steps on each input $s_1, s_2, \ldots, s_i$.
3. If M accepted any of the strings, <u>print it</u>
4. $i := i + 1$; goto step 2."

Every recognizable language L is enumerable:
Let M be a recognizer for L. We construct an enumerator E for L.

Let $s_1, s_2, s_3, \ldots$ be all possible strings from Σ^*.

$E =$ "
1. $i := 1$;
2. Run M for i steps on each input $s_1, s_2, \ldots, s_i$.
3. If M accepted any of the strings, print it
4. $i := i + 1$; goto step 2."

This technique is called Dovetailing.

Hilbert's Tenth Problem

- In 1900 David Hilbert asked to find a mechanical way to check whether a polynomial (over several variables) has an integral root.

  Example:
  $$6x^3yz^2 + 3xy^2 - x^3 - 10$$
  has an integral root at $x = 5$, $y = 3$ and $z = 0$.

Hilbert's Tenth Problem

- In 1900 David Hilbert asked to find a mechanical way to check whether a polynomial (over several variables) has an integral root.

  Example:
  $$6x^3yz^2 + 3xy^2 - x^3 - 10$$
  has an integral root at $x = 5$, $y = 3$ and $z = 0$.

Answer

Nobody has found such an algorithm yet ...

Hilbert's Tenth Problem

- In 1900 David Hilbert asked to find a mechanical way to check whether a polynomial (over several variables) has an integral root.

  Example:

  $$6x^3yz^2 + 3xy^2 - x^3 - 10$$

  has an integral root at $x = 5$, $y = 3$ and $z = 0$.

Answer

Nobody has found such an algorithm yet ... in fact, we know that this is impossible, and we can prove this!!! (Yuri Matijasevic'1970).

We need a model of an algorithm to demonstrate such a proof.

Models of an algorithm:

- 1936: Alan Turing came with Turing machine and Alonzo Church with λ-calculus.

Church-Turing Thesis

Models of an algorithm:

- 1936: Alan Turing came with Turing machine and Alonzo Church with λ-calculus.
- Turing machines and λ-calculus were shown equivalent.
- Many more models suggested: Kleene's μ-recursive functions, Post-systems, Minsky machine ... all shown equivalent!

Church-Turing Thesis

Models of an algorithm:

- 1936: Alan Turing came with Turing machine and Alonzo Church with λ-calculus.
- Turing machines and λ-calculus were shown equivalent.
- Many more models suggested: Kleene's μ-recursive functions, Post-systems, Minsky machine ... all shown equivalent!

Church-Turing Thesis

Algorithms	$=$	Turing machines
(informal notion)		(formal, mathematical, concept)

Facts:

- Church-Turing Thesis cannot be proved, BUT ...
- Java/Python/C++/C# programs can be run on a TM, and
- nothing more powerful than a TM has been found so far.

Algorithmic Problems vs. Decidable Languages

How do algorithmic problems correspond to languages?

Algorithmic Problems vs. Decidable Languages

How do algorithmic problems correspond to languages?

Example:

Hilbert's Tenth Problem

Can the problem whether a given polynomial has an integral root be algorithmically solved?

Algorithmic Problems vs. Decidable Languages

How do algorithmic problems correspond to languages?

Example:

Hilbert's Tenth Problem

Can the problem whether a given polynomial has an integral root be algorithmically solved?

Language Formulation of Hilbert's Tenth Problem

$$D \stackrel{\mathrm{def}}{=} \{\langle p\rangle \mid p \text{ is a polynomial with integral root } \}$$

where $\langle p\rangle$ is the textual (string) encoding of the polynomial p.

Is D a decidable language?

Algorithmically solvable problems $\equiv$ decidable languages.

Algorithmic Problems vs. Decidable Languages: Examples

Graph Connectivity

"Is a given graph G connected?" corresponds to:

Does $\langle G \rangle$ (encoding of G) belong to the language

$$L_{connected} \stackrel{\text{def}}{=} \{\langle G' \rangle \mid \; G' \text{ is a graph and } G' \text{ is connected } \} \,?$$

Acceptance Problem of a TM

"Does a given TM M accept a string w?" corresponds to:

Does $\langle M, w \rangle$ belong to the language

$$A_{TM} \stackrel{\text{def}}{=} \{\langle M', w' \rangle \mid \; M' \text{ is a TM and } M' \text{ accepts } w' \;\} \,?$$

Algorithmic Problems vs. Decidable Languages: Examples

Graph Connectivity

"Is a given graph G connected?" corresponds to:

Does $\langle G \rangle$ (encoding of G) belong to the language

$$L_{connected} \stackrel{\text{def}}{=} \{ \langle G' \rangle \mid G' \text{ is a graph and } G' \text{ is connected } \} \ ?$$

Acceptance Problem of a TM

"Does a given TM M accept a string w?" corresponds to:

Does $\langle M, w \rangle$ belong to the language

$$A_{TM} \stackrel{\text{def}}{=} \{ \langle M', w' \rangle \mid M' \text{ is a TM and } M' \text{ accepts } w' \ \} \ ?$$

Facts: $L_{connected}$ is decidable, while A_{TM} is undecidable!

Exam Questions

- Equivalence of k-tape TM with 1-tape TM.
- Nondeterministic TM, definition, acceptance of a string, equivalence with ordinary TM.
- Enumerators, definition, equivalence with ordinary TM.
- Dovetailing technique.
- Church-Turing Thesis.