Monash University
Faculty of Information Technology

Lecture 16
Decidability

Slides by David Albrecht (2011), modified by Graham Farr (2013).

FIT2014 Theory of Computation

Overview

- Decision problems
- Decidable problems and languages
- Deciders
- A Decidable Logic Theory
- Closure

Decision Problems

- Input: an integer
  Question: Is it even?
- Input: a string.
  Question: Is it a palindrome?
- Input: an expression in propositional logic
  Question: Is it ever True?
- Input: a graph G, and two vertices s and t
  Question: is there a path from s to t in G ?
- Input: a Java program
  Question: is it syntactically correct?

Decision Problems

- Input: a Finite Automaton
  Question: Does it define the empty language.
- Input: two Regular Expressions
  Question: Do they define the same language?
- Input: a Finite Automaton
  Question: Does it define an infinite language?
- Input: a Context Free Grammar
  Question: Does it define the empty language?
- Input: a Context Free Grammar
  Question: Does it generate an infinite language?
- Input: a Context Free Grammar and a string *w*
  Question: Can *w* be generated by the grammar?

Decision Problems

A **decision problem** is a problem where, for each input, the answer is **yes** or **no**.

Decision problem ➔ language

- { YES-inputs}

Language ➔ decision problem

- Input: a string

  (over some alphabet, usually representing some object)

  Question: Is the string in the Language?

Decidable Problems

- A decision problem is **decidable** if there is an algorithm that solves it.
  - i.e., correctly provides a **yes** or **no** answer in a **finite** number of steps.
- A language is **decidable** if there is an algorithm that solves its corresponding decision problem
  - i.e., correctly tells **whether or not** something is in the language, in a **finite** number of steps.

Decidable: synonyms

decidable
= recursive
= solvable

= computable

... sometimes,
though “computable” has been
used with other meanings too.

Encoding of Input

The input and output for a Turing Machine is always a string.

- For any object, O, **<O>** will denote encoding of the object as a string.
- If we have several objects, $O_1,..,O_n$, **<$O_1,..,O_n$>** will denote their encoding into a single string.

Deciders

A **decider** is a Turing Machine that halts for any input over a given alphabet, and has two possible outputs - either a

<YES> or a **<NO>**.

Decidable Languages

A decidable language consists of all those inputs for a decider that halt with a **<YES>**.

Examples:

- Regular Languages
- Context Free Languages
- $a^n b^n a^n$

Testing Emptiness of Regular Languages

- Problem:

  Given a Finite Automaton, decide whether the language it defines is empty.

- Let **E** be the set of **A** such that:
  - **A** is a Finite Automaton and
  - The language defined by **A** is empty.
- **E is a decidable language**.

Proof

Let T be the Turing Machine that implements the following algorithm:

On input **<A>** where **A** is a Finite Automaton.

1. Mark the start state of **A**.
2. Repeat until no new states get marked:
   - Mark any state that has a transition coming into it from any state that is already marked.
3. If no final state is marked, output **<YES>**; otherwise output **<NO>**.

Testing Equivalence of Regular Expressions

- Problem:

  Given two regular expressions decide whether they define the same language.

- For a Regular expression **R,** let **L(R)** be the language defin**e**d by **R.**
- Let **E** be the set of **{A,B}** such that:
  - **A** and **B** are Regular expressions **and**
  - **L(A) = L(B).**
- **E is a decidable language.**

Proof

Consider the Turing Machine that implements the following algorithm:

On input **<A,B>** where **A** and **B** are Regular expressions.

1. Construct a FA, **C,** that defines the language

$$\mathbf{(L(A) \cap \overline{L(B)}\,) \cup (\,\overline{L(A)} \cap L(B))}$$

- Run the previous Turing Machine, **T**, on **C**.
- If the output **T** on **C**, is **<YES>** output **<YES>,** otherwise output **<NO>**.

Testing Emptiness of Context Free Language

- Problem:

For a given Context Free Grammar, decide whether the language it defines is empty.

- Let **E** be the set of **A** such that:
  - **A** is a Context Free Grammar and
  - The language defined by **A** is empty.
- **E is a decidable language**.

Proof

Let T be the Turing Machine that implements the following algorithm:

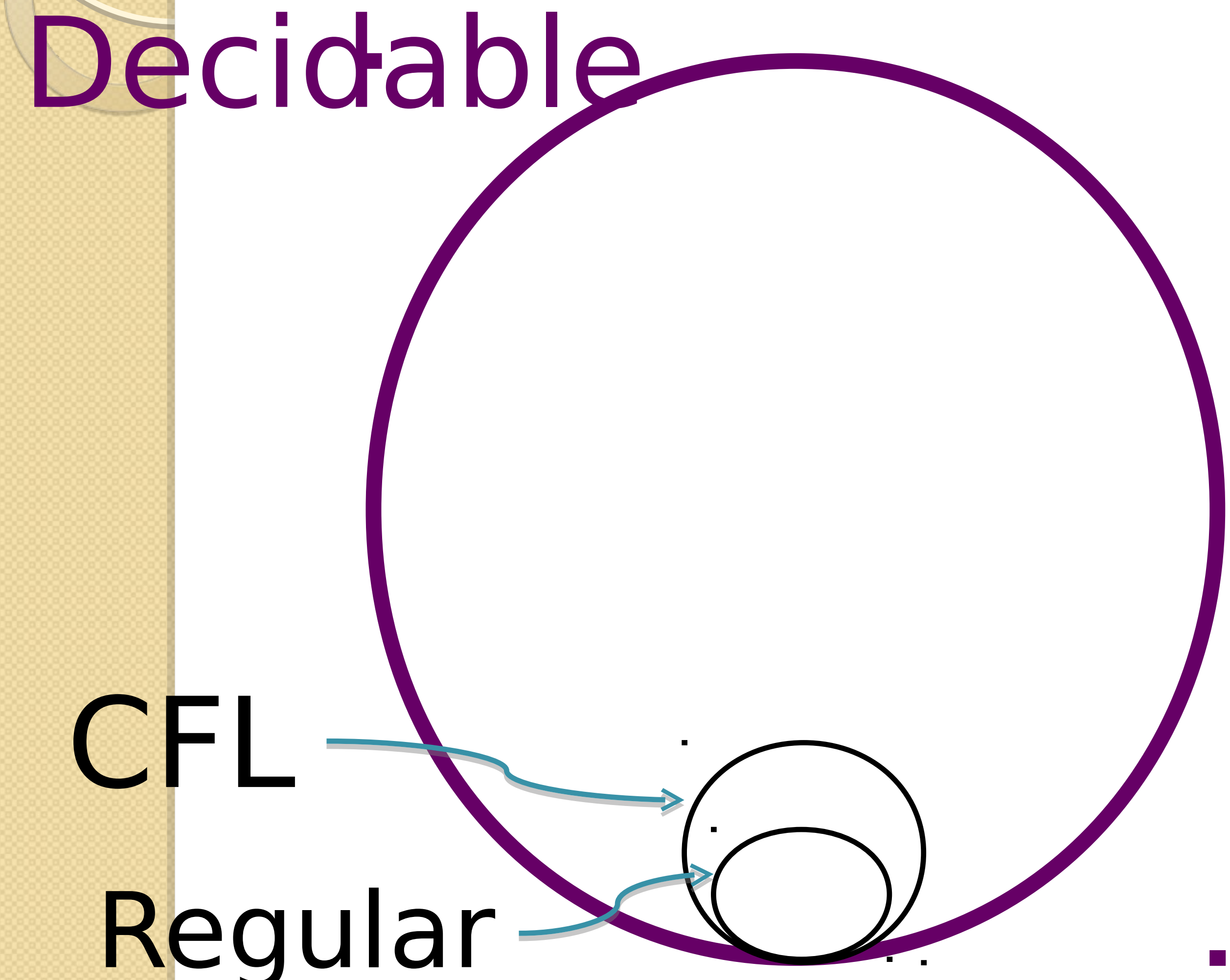
On input **<A>** where **A** is a Context Free Grammar.

1. Mark all the terminal symbols in A.
2. Repeat until no new symbols get marked:
   - Mark any non-terminal X that has a production which has all the right-hand symbols marked.
3. If start symbol is not marked, output **<YES>**; otherwise output **<NO>**.

Some Decidable Problems

- Input: a Finite Automaton
  Question: Does it define the empty language.
- Input: two Regular Expressions
  Question: Do they define the same language?
- Input: a Finite Automaton
  Question: Does it define an infinite language?
- Input: a Context Free Grammar
  Question: Does it define the empty language?
- Input: a Context Free Grammar
  Question: Does it generate an infinite language?
- Input: a Context Free Grammar and a string *w*
  Question: Can *w* be generated by the grammar?

Language classes



Simple Logic Sentences

$$S \rightarrow \forall X\ S \mid \exists X\ S$$

$$S \rightarrow \neg S \mid (S \vee S) \mid (S \wedge S)$$

$$S \rightarrow (T = T)$$

$$T \rightarrow T + X \mid X$$

$$X \rightarrow \text{variable}$$

Universe of Natural Numbers

Consider the following sentences for natural numbers:

- $\forall x\ \exists y\ (x + x = y)$
- $\exists y\ \forall x\ (x + x = y)$
- $\forall x\ \exists y\ (y + y = x)$

Simple Logic Theory

The set of sentences which are generated by the following grammar that are true for Natural numbers is decidable.

$$\mathbf{S \rightarrow \forall X\ S \mid \exists X\ S}$$

$$\mathbf{S \rightarrow \neg S \mid (S \vee S) \mid (S \wedge S)}$$

$$\mathbf{S \rightarrow (T = T)}$$

$$\mathbf{T \rightarrow T + X \mid X}$$

$$\mathbf{X \rightarrow \text{variable}}$$

Closure properties

If $\boldsymbol{L}$ is decidable, then $\overline{\boldsymbol{L}}$ is decidable.

If $\boldsymbol{L_1}$ and $\boldsymbol{L_2}$ are decidable, then so are

- $\boldsymbol{L_1} \cup \boldsymbol{L_2}$
- $\boldsymbol{L_1} \cap L_2$
- $\boldsymbol{L_1 L_2}$
- ….

Exercise:

Formulate and prove more closure results.

Revision

- Decision problems, relationship with languages
- What is a Decidable Problem?
- What is a Decidable Language?
- The connection between decidable problems and decidable languages.
- Examples of Decidable Problems.
- Closure properties

Reading:

- Sipser, Section 4.1, pp. 190-201.

Preparation:

- Sipser, Section 4.2, pp. 201-213, especially pp. 207-209.