Curtin University of Technology

Theoretical Foundations of Computer Science

Lecture 7

Reducibility

Curtin University of Technology

Aims of lecture

- To explore undecidability
  - ➢ Last lecture showed
    - – Hierarchy of languages
      - » A_{TM} is undecidable
      - » The Co-language of A_{TM} is not Turing Recognised
    - – Have existence of larger languages
  - ➢ More decidable and undecidable languages
  - ➢ Determine undecidability through mapping proofs

Curtin University of Technology

Unit Learning Outcomes

- Understand recognisability and decidability, use the construction & mapping reducibility techniques to prove a problem decidable or undecidable.

Curtin University of Technology

Assessment Criteria

- **Prove** the classification of a language as Decidable or Undecidable using mapping reducibility of languages involving languages or graphs.

Curtin University of Technology

Proving Undecidablity

- To explore undecidability
  - ATM is undecidable
  - Consequences:
    - The Co-language of ATM is not Turing Recognisable
    - Have existence of larger set of languages
    - Turing Thesis: No machine will recognise them, unsolvable

- Programmers: Is your problem unsolvable?
  - Need techniques to determine undecidable languages
  - Direct techniques often difficult

Reducibility

- Transformation of problem concept
  - ➢ Used to prove other languages undecidable
  - ➢ Used in definition of NP completeness
  - ➢ To programmers:
    - Can a problem be converted to one already solved?

- Transform is a function
  - ➢ If $w \in \Sigma^*$ as input, create output $f(w) \in \Sigma^*$
  - ➢ Process to be done by a decidable TM computing f

Curtin University of Technology

Decidability of A_{REX}

- A_{REX} = { <R,w>: R is a RE that generates w}
  - Proof
    - Convert R into NFA M_1 using known construction
    - Convert M_1 into DFA M_2 using known construction
    - For any w generated by R, M_1 & M_2 both recognise w because of the equality guaranteed by the constructions
    - There is a M_3, a TM that decides M_2 recognising w since it has been shown that A_{DFA} is decidable
  - So string <R,w> => <M_1,w> => <M_2,w>
    - TM for A_{REX} is
      - TM to transform <R,w> to <M_2,w>
      - Followed by M_3

Reduction

- A way of converting one problem into another
  - A solution to the second problem can be used to solve the first problem
  - Given problems *A* and *B*, if *A* reduces to *B*, a solution to *B* can be used to solve *A*.
    - only concerned with solvability of *A* when *B*'s solution is known
  - If A reduces to B then B is more complex (harder) than A
    - A reduces to simple B, then A must be simple
    - Complex A reduces to B then B must be complex

Curtin University of Technology

MAPPING REDUCIBILITY

Concept

Computable Functions

Mapping Reducibility

Theorems concerning Reducibility

Curtin University of Technology

Mapping reducibility

- Formalizing the notion of reducibility
  - ➢ allows us to use reducibility in more refined ways
  - ➢ *e.g.*, proving that some languages are not Turing-recognizable, applications in complexity theory
- Mapping reducibility (also known as many-one reducibility)
  - ➢ Being able to reduce problem *A* problem *B* means a computable function exists that convert instances of problem *A* to instances of problem *B*.
  - ➢ Such a conversion function is called a *reduction*.

Curtin University of Technology

Computable functions

- A TM computes a function by starting with the input to the function on the tape and halting with the output of the function on the tape.
- A function f: $\Sigma^* \rightarrow \Sigma^*$ is a computable function if some TM M, on every input w, halts with just $f(w)$ on its tape.
- Example:
  - ➢ A function f takes input w and returns the description of a TM $\langle M' \rangle$ if $w = \langle M \rangle$ is an encoding of a TM M.
  - ➢ M' is a TM that recognizes the same language as M.

Curtin University of Technology

Definition of mapping reducibility

- Language A is mapping reducible to language B, written $A \leq_m B$, if there is a computable function f: $\Sigma^* \rightarrow \Sigma^*$, where for every w, $w \in A \Leftrightarrow f(w) \in B$.
- The function f is called the reduction of A to B.
- Provides a way to convert membership testing in A to membership testing in B.
  - ➢ If one problem is mapping reducible to a second previously solved problem, a solution can be obtained for the original problem.

Curtin
University of Technology

Theorems of reducibility

- Classifying problems by decidability:
  - If *A* is reducible to *B* and *B* is decidable, then *A* is also decidable.
  - Similarly, if *A* is undecidable and reducible to *B*, then *B* is also undecidable.

DECIDABILITY & MAPPING REDUCIBILITY

Approach

Used for A_{REX}

Approach

- Aim
  - ➢ Use Mapping Reducibility to prove Decidability
- Reason
  - ➢ Direct proof involves deep insight into the process
  - ➢ Mapping Reducibility
    - – Ignores the machine; purely language
    - – Requires only matching a problem to another problem
- Technique
  - ➢ A, find decidable B, show $A \leq_m B$; A decidable

Curtin University of Technology

Fact for Solving A_{REX}

- Need known facts
  - ➢ A_{DFA} {<M,w>: M is a DFA and M accepts w}
    - A_{DFA} is decidable
  - ➢ R a regular expression
    - Can construct NFA M_1 so R generates w $\Leftrightarrow$ M_1 accepts w
    - Algorithm has a TM that finishes in finite steps
  - ➢ M_1 describes an NFA
    - Can construct DFA M_2 so M_1 accepts w $\Leftrightarrow$ M_2 accepts w
    - Algorithm has a TM that finishes in finite steps
  - ➢ A_{REX} = { <R,w>: R is a RE that generates w}

Curtin University of Technology

Proof

- Currently A_{REX} is unknown A_{DFA} is decidable
  - So want $A_{REX} \leq_m A_{DFA}$
- Start with a string from A_{REX}
  - Let $<R,w> \in AREX$
- Need a function into ADFA so of form <M,w>
  - Let $f_1 : \Sigma^* \rightarrow \Sigma^*$, mapping for RE to NFA
  - So $f_1(<R,w>) = < M_1,w>$
  - $<R,w> \in A_{REX}$ means R generates w
  - f_1 is such that R generates w $\Leftrightarrow$ M_1 accepts w
  - So $f_1(<R,w>) \in A_{NFA}$.

Curtin University of Technology

Proof (continued)

- Note the equivalence, so argument is both ways
  - So $\langle R,w\rangle \in A_{REX} \Leftrightarrow f_1(\langle R,w\rangle) \in A_{NFA}$
  - f_1 is a reduction

- f_1 is a reduction, but we have not proved A_{NFA} to be undecidable, hence we are not finished
  - Using similar reasoning, there is a function $f_2: \Sigma^* \rightarrow \Sigma^*$ such that:
    - $\langle M_1,w\rangle \in A_{NFA} \Leftrightarrow f_2(\langle M_1,w\rangle) \in A_{DFA}$

- Hence $\langle R,w\rangle \in A_{REX} \Leftrightarrow f_2(f_1(\langle R,w\rangle)) \in A_{DFA}$

Curtin University of Technology

Conclusion of Proof

- So we have the reduction
  - Let $f: \Sigma^* \to \Sigma^*$: such that $f(\langle R,w \rangle) = f_2(f_1(\langle R,w \rangle))$
  - f is a reduction, $A_{REX} \leq_m A_{DFA}$
  - We know that is A_{DFA} decidable so A_{REX} is also decidable.

Curtin University of Technology

REDUCTION

Principle
The Real Halting Problem
Two More Examples

Reduction

- A way of converting one problem into another
  - ➢ so that a solution to the second problem can be used to solve the first problem
  - ➢ Given problems A and B, if A reduces to B, a solution to B can be used to solve A.
    - – only concerned with solvability of A when B's solution is known
- Examples from everyday life
  - ➢ Finding your way around a new city can be reduced to getting a map of the city
  - ➢ Problem of travelling from Perth to Sydney is reduced to buying a plane ticket for the journey

Curtin University of Technology

Role of reducibility

- Examples from Maths:
  - ➢ Measuring the area of a triangle can be solved if its height and width can be measured
  - ➢ A system of linear equations can be solved if matrix inversion can be done
- Classifying problems by decidability
  - ➢ If A is reducible to B and B is decidable, then A is also decidable.
  - ➢ Similarly, if A is undecidable and reducible to B, then B is also undecidable.

Curtin University of Technology

Undecidability

- Have that A_{TM} is undecidable
  - ➢ This is really the Acceptance Problem
  - ➢ Will use Reductions to prove undecidability
  - ➢ Note finiteness
    - If we can prove TM will always stop in a finite number of moves then language is decidable
    - Undecidability depends on proof by contradiction
  - ➢ Will look at the true Halting Problem

Curtin University of Technology

Approach

- Use Reducibility
  - ➢ Aim: So A is undecidable
  - ➢ Proof
    - – Assume A is decidable
    - – Find an undecidable B — the key insight
    - – Show $B \leq_m A$ — often have to guess appropriate B
    - – But implies B is decidable
    - – Contradiction so A is undecidable
  - ➢ Or Reduce selected B to A

Curtin University of Technology

Undecidable problems from language theory

- Determining whether a TM halts on a given input
  - ➢ $HALT_{TM}$= {<M, w>|M is a TM and M halts on input w}
- Theorem: $HALT_{TM}$ is undecidable
- Proof idea:
  - ➢ Assume that R is a TM that decides $HALT_{TM}$.
  - ➢ Using R, we can test whether M halts on w.
    - – If R indicates that M doesn't halt on w, <M,w> is not in A_{TM}.
    - – If R indicates that M does halt on w, we can simulate without danger of looping.
  - ➢ If R exists, we can decide A_{TM}, contradicting an earlier theorem.

Curtin University of Technology

Proof

- Assume that TM R decides $HALT_{TM}$.
  - ➢ Construct TM S to decide A_{TM}
  - ➢ S = "On input <M, w>:
    - 1. Run TM R on input <M, w>.
    - 2. If R rejects, *reject*.
    - 3. If R accepts, simulate M on w until it halts.
    - 4. If M has accepted, *accept*; if M has rejected, *reject*."
  - ➢ If R decides $HALT_{TM}$, then S decides A_{TM}.
    - Because A_{TM} is undecidable, $HALT_{TM}$ also must be undecidable.

Curtin University of Technology

Halting problem

- $HALT_{TM}$ is the real halting problem
  - $A_{TM} = \{<M, w> | M$ is a TM and M accepts input $w\}$ is the acceptance problem
- Proof of $HALT_{TM}$ illustrates the strategy for proving that a problem is undecidable
  - Common strategy for most proofs of undecidability
  - A_{TM} itself is directly proved (via diagonalisation)

Curtin University of Technology

Further examples of reducibility

- $E_{TM} = \{<M>|\ M \text{ is a TM and } L(M) = \phi\}$
- Theorem: E_{TM} is undecidable.
- Proof idea:
  - Assume that E_{TM} is decidable.
  - Then show that A_{TM} is decidable, a contradiction.

Curtin University of Technology

Further examples of reducibility

- $REGULAR_{TM} = \{\langle M \rangle \mid M$ is a TM and $L(M)$ is a regular language$\}$
- Theorem: $REGULAR_{TM}$ is undecidable.
- Proof idea:
  - ➢ Assume that $REGULAR_{TM}$ is decidable by a TM R.
  - ➢ Use R to construct a TM S that decides A_{TM}.

Curtin University of Technology

Post Correspondence Problem

- An undecidable problem concerning simple manipulations of strings
- Two collections of dominos

$$\left\{\left[\frac{b}{ca}\right],\left[\frac{a}{ab}\right],\left[\frac{ca}{a}\right],\left[\frac{abc}{c}\right]\right\}$$

$$\left\{\left[\frac{a}{ab}\right],\left[\frac{b}{ca}\right],\left[\frac{ca}{a}\right],\left[\frac{a}{ab}\right],\left[\frac{abc}{c}\right]\right\}$$

- When the string of symbols at the top and the bottom of a collection are the same, there is a match.
- PCP is to determine whether a collection of dominos has a match. Repetitions of dominos is allowed.
- The general form of this problem is unsolvable by algorithms.

Curtin University of Technology

Summary

- Role of reducibility
- Reductions via computation histories
- Mapping reducibility