ITS64304 Theory of Computation

School of Computer Science
Taylor's University Lakeside Campus

Lecture 6: Un-Decidability

Dr Raja..

Learning outcomes

At the end of this topic you should be able to:

- Identify there are problems that are undecidable*

* Aligns to Course Learning Outcome 3

Computability

What can be computed?

- What are the limits of computation? Or power of computers?
- Are there problems for which no algorithms can ever exist?
  - How can we prove this?

Computability

- Some problems are obviously too vague to be solved by an algorithm
  - e.g. How big is the universe?

- Some practical problems are hard, although easy to define precisely.
  - *can a compiler determine if a program contains the possibility of an infinite loop*?
  - *Or any uninitialized variables*?

- Intuitively any statement should either be *true* or *false* (provided we have all the facts)

Computability

- Some problems are called *undecidable;* do not have any algorithmic solution!

- Some problems are *semi-decidable*; we can determine when a property is satisfied, but not when it is not
  - e.g. is there an odd perfect number? (a perfect number equals the sum of its proper divisors, e.g. 6 = 1+2+3)

  - Is 1 perfect?
  - Is 3 perfect?
  - Is 5 perfect?

Standard and Complete TMs

Church-Turing thesis

- (Standard) Turing Machines are mechanistic and deterministic.
- A Complete Turing Machine M halts on all inputs
- Hence, Turing machines which always halt correspond to the intuitions above

Standard TMs

All the following are equivalent to standard Turing Machines:

- multiple tapes
- two-way infinite tapes
- multiple heads
- two-dimensional tapes
- random access
- nondeterministic machines
- unrestricted grammars
- + every formal model yet found!

Church-Turing Thesis

Church-Turing thesis:

- All *possible* models of computation are equivalent to a standard Turing machine

- Any *future* models of computation will also be no more powerful than a standard Turing machine!

  - Cannot be proved; it is not a theorem!
  - Can only be experimentally verified
  - Will quantum computing disprove this? Too early to tell!

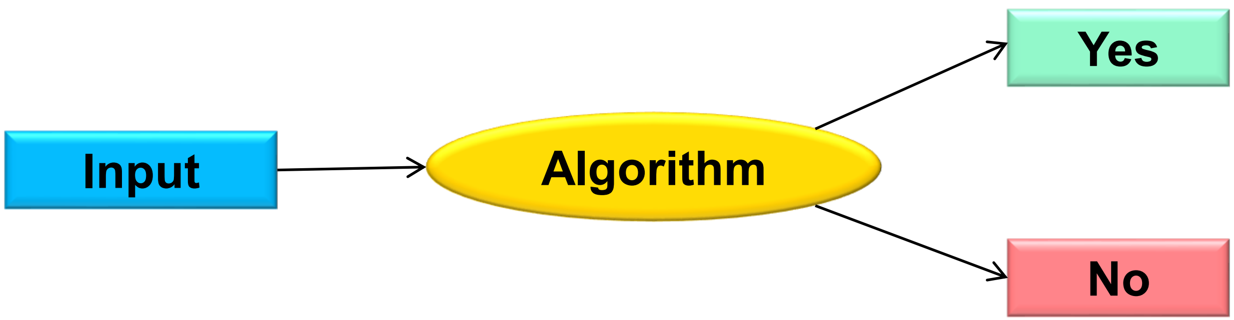
Universal Turing Machine

The Universal Turing machine

- Takes another Turing machine M as input
- Simulates the actions of M
  - Model for “real” computers
    - i.e. input is both data and instructions

Decision Problem

- is a question in some formal system with a ‘yes’ or ‘no’ answer, depending on the values of some input



- E.g.
  - Given a number x, is x a prime number?
  - Given a 3-SAT formula is it satisfiable?
  - Halting Problem

Decision Problem

- A decision problem is *decidable* if there a TM exists to solve it
- A decision problem is *un-decidable* if no TM exists to solve it

Paradoxes

A ruler decrees:

1. All men who do not shave themselves must be shaved by the barber
2. All men who do shave themselves must not be shaved by the barber

So who shaves the barber?

- Barber shaves himself $\Rightarrow$ not shaved by Barber $\Rightarrow$ doesn't shave himself
- Barber doesn't shave himself $\Rightarrow$ shaved by Barber $\Rightarrow$ shaves himself ????

Paradoxes

Resolutions:

1. Don’t shave!
2. Exempt the barber from the rules
3. The barber is a woman ...

Self-Reference

Self-Reference

- The problem with many paradoxes is that they are *self-referential*
- If we avoid self-referential statements logic and mathematics won’t have paradoxes
- However in computation it is sometimes hard to avoid self-referential statements

Halting Problem

An algorithm to determine if a given Turing Machine halts on a given input" is self-referential.

- Given an arbitrary Turing machine M, does the computation of M with input w halt?
  - Outputs 'yes' if M halts on w
  - Outputs 'no' if M does not halt on w
- This is undecidable. No algorithm for this problem.
- **Halting Problem is undecidable**
- It is possible for a TM to solve problem for particular cases, but not in general

Read your lesson materials

Look at Tutorial 9 and prepare for it…..