Algorithms I

- Informally, an algorithm is a collection of instructions
- Formal definition was not done until 20th century

Hilbert's problems I

- In 1900, Hilbert in an address at the International Congress of Mathematicians identified 23 mathematical problems for the coming century.
- The 10th asks for an algorithm to test if a polynomial has integer root or not

$$6x^3yz^2 + 3xy^2 - x^3 - 10 = 0$$
$$x = 5, y = 3, z = 0$$

- In Hilbert's description, the word "algorithm" was not used

Hilbert's problems II

- Roughly he said a process of a finite # of operations
- However, Hilbert explicitly asked the algorithm be "devised"
- Thus we need a definition of algorithms
- In the end this problem is algorithmically unsolvable

Church-Turing thesis I

- Proposed in 1936
- Intuitive algorithms $\equiv$ TM algorithms
- Note that this is a definition but not a theorem

Hilbert's 10th problem I

- Using our terms

$$D = \{P \mid P: \text{ polynomial with integer roots}\}$$

  D: decidable or not?

- A simpler problem of a single variable

$$D_1 = \{P \mid P: \text{ polynomial of } x \text{ with integer roots}\}$$

- Example:

$$4x^3 - 2x^2 + x - 7$$

Hilbert's 10th problem II

- We can use a TM to evaluate x at

$$0, 1, -1, 2, -2, \ldots$$

  If 0, accept
- If P has no integer root $\Rightarrow$ this evaluation runs forever
- Thus we have a recognizer, but not a decider

Hilbert's 10th problem III

- It can be proved that roots of a 1-variable polynomial is within the range

$$\pm k \frac{c_{\max}}{c_1}$$

k: # terms, $c_{\max}$: max(abs(coefficients))
c_1: coefficient of the highest order

Hilbert's 10th problem IV

- For the example

$$4x^3 - 2x^2 + x - 7$$

  we have

$$\pm 4 \times \frac{7}{4} = \pm 7$$

- The proof is easy (an exercise in the book)
- Unfortunately, the case of multiple variables is very hard
- Only until 1970: it's proved that bounds for multi-variable polynomials are not possible
- Thus this problem is undecidable

Description of Turing Machines I

- Three levels
  1. High-level: no mention how to manage tape and head

     Like how we describe algorithms
  2. Implementation-level: English to describe how head moves

Description of Turing Machines II

For example, our description of the $\{w\#w \mid w \in \{0.1\}^*\}$ language

$$\overset{\blacktriangledown}{0}\ 1\ 1\ 0\ 0\ 0\ \#\ 0\ 1\ 1\ 0\ 0\ 0\ \sqcup$$

$$\mathrm{x}\ \overset{\blacktriangledown}{1}\ 1\ 0\ 0\ 0\ \#\ 0\ 1\ 1\ 0\ 0\ 0\ \sqcup$$

$$\mathrm{x}\ 1\ 1\ 0\ 0\ 0\ \#\ \overset{\blacktriangledown}{\mathrm{x}}\ 1\ 1\ 0\ 0\ 0\ \sqcup$$

3. formal-level: all detailed transitions

- We will mainly use high-level descriptions later

Example 3.23 I

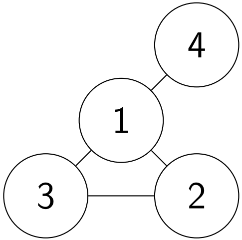
$$A = \{\langle G \rangle \mid G : \text{a connected undirected graph}\}$$

- A high-level TM
  1. Mark a node in G
  2. Repeat until no new nodes marked

     For every node G, mark it if $\exists$ an edge to a marked node
  3. If all nodes marked: accept, otherwise: reject
- Real implementation

Example 3.23 II

Figure 3.24



$$\langle G \rangle = (1,2,3,4)((1,2),(2,3),(3,1),(1,4))$$

is the input string

- Details

Example 3.23 III

- The first step is to check if the input is in the correct format
- In the first step we begin with seeing if the first part of the input $\langle G \rangle$ includes distinct numbers (as node IDs should be different)
- This is similar to an example before

$$\{\#x_1\#x_2 \cdots \#x_l \mid x_i \in \{0,1\}^*, x_i \neq x_j\}$$

- Then we can talk about how the head is moved
- Thus we have implementation-level descriptions