Decision problems, formal languages, and compuation

Arjun Chandrasekhar

Function Problem

Def: A **function problem** asks us to compute mathematical function

- Receives an input, produces an output
- Examples:
  - Given an integer n, output its prime factors
  - Given a graph G and vertices u, v, output the length of the shortest path from u to v

Can we design a machine to produce the correct output every time?

Decision Problem

Def: A **decision problem** asks us to compute a mathematical predicate

- Receives an input only outputs ACCEPT or REJECT
  - Does integer n have a prime factor $p \leq k$?
  - Does G have a path from u to v of length $\leq k$?

Are decision problems and function problems the equivalent?

Decision vs Function problem

Consider the shortest path decision problem

- **Input:** G, u, v, k
- **Output:** ACCEPT if G has a u-v path of length $\leq k$, and REJECT otherwise
- Suppose we have a magic crystal ball to find shortest path length from u to v.
- Can we determine if the shortest path length is $\leq k$?
  - Yes!
  - Find the shortest path length, check if it is $\leq k$

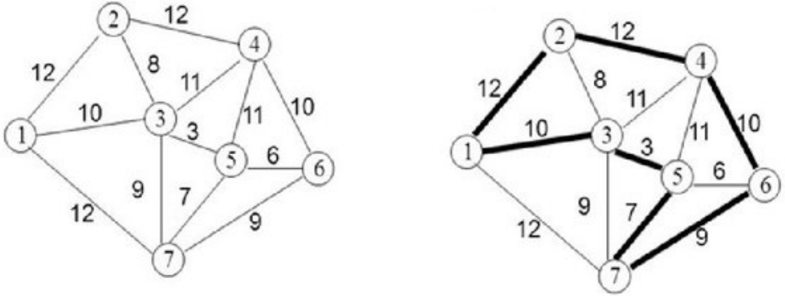
Decision vs Function problem

Consider the shortest path function problem

- **Input:** G, u, v
- **Output:** The length of the shortest path from u to v
- Suppose we have a magic crystal ball to check if the shortest path length from u to v is $\leq k$.
- Can we find the shortest path length?
  - Yes!
  - Let n be the total number of vertices in G
  - For $k = 1, 2, \ldots n$: check if the shortest path length is $\leq k$
  - Stop the first time we receive an output of ACCEPT

Travelling Salesman Problem

Travelling Salesman Problem: try find a path that visits all nodes in a graph using the lowest possible edge weights



a) b)

Travelling Salesman Problem

Travelling Salesman Problem: try find a path that visits all nodes in a graph using the lowest possible edge weights

1. What is the function problem associated with travelling salesman?
2. What is the decision problem associated with travelling salesman?
3. If we can solve the function problem, how can we solve the decision problem?
4. If we can solve the decision problem, how can we solve the function problem?

Travelling Salesman Problem

What is the function problem associated with travelling salesman?

- **Input:** A graph G
- **Output:** The length of (one of) the shortest path(s) that includes every vertex

Travelling Salesman Problem

What is the decision problem associated with travelling salesman?

- **Input:** A graph G and an integer k
- **Output:**
  - Output ACCEPT if there is a path that A) includes every vertex B) has length $\leq k$
  - Output REJECT otherwise

Travelling Salesman Problem

If we can solve the function problem, how can we solve the decision problem?

- **Input:** A graph G and a number k
- Suppose we had a crystal ball that could find the length of the shortest path that touches each vertex
- Find the shortest path, and check if its total length is $\leq k$
  - If yes, output ACCEPT
  - Otherwise, output REJECT

Travelling Salesman Problem

If we can solve the decision problem, how can we solve the function problem?

- **Input:** A graph G
- Suppose we had a crystal ball that could tell us if there is a path of length $\leq k$ that includes each vertex
- Let n total length of all edges in the graph
- For $k = 1, 2, \ldots, n-1, n$:
  - Check if there is a path of length $\leq k$ that includes each vertex
  - If yes, then output k. Otherwise, keep searching

Strings and Alphabets

- **Def:** An **alphabet** Σ is a collection of symbols
  - $\Sigma_1 = \{a, b\}$
  - $\Sigma_2 = \{0, 1\}$
  - $\Sigma_3 = \{0, 1, \ldots, 9\}$
- **Def:** A **string** is a sequence of symbols from some alphabet
  - *aaabbbbba*
  - 100101010
  - 974093273
- Let Σ be an alphabet. Then Σ^* is the set of all possible strings on that alphabet

Strings and Alphabets

Let $\Sigma = \{A, C, G, T\}$. What is in Σ^*?

$\Sigma^* = \{\epsilon$, A, C, G, T, AA, AC, CA, AG, GA, AT, TA, CC, CG, GC, CT, TC, GG, GT, TG, TT, AAA, $\ldots\}$

Note: ϵ refers to the empty string

Formal Languages

- **Def:** A **(formal) language** $L \subseteq \Sigma^*$ is a collection of strings on an alphabet
  - $L_1 = \{\text{aa, ab, ba, bb}\} \subseteq \{a, b\}^*$
  - $L_2 = \{\text{w} \mid \text{w contains 001 as a substring}\} \subseteq \{0, 1\}^*$
  - $L_3 = \{\text{w} \mid \text{w is even}\} \subseteq \{0, 1, 2, \ldots, 9\}^*$

Decision Problems and Formal Languages

Each formal language has an associated decision problem

- Input: a string from the alphabet
- Output:
  - ACCEPT if the string is in the language
  - REJECT if the string is not in the language

Can we design computers to solve these decision problems?

Some languages of interest

- $L_1 = \{\text{w} \mid \text{w is a valid Java identifier}\}$
- $L_2 = \{\text{w} \mid \text{w is a valid Java math expression}\}$
- $L_3 = \{\text{w} \mid \text{w is a valid Java program}\}$
- $L_4 = \{\text{w} \mid \text{w contains "review" as a substring}\}$
- $L_5 = \{\text{w} \mid$ w is a valid encoding for the shortest path decision problem on a graph for which the answer is yes$\}$

What is computation?

- We feed our input to a machine
  - We will define what a "machine" is later
- The machine performs some mechanical procedure
- The machine does one of three things:
  - Output ACCEPT
  - Output REJECT
  - Infinite Loop

Deciding/Recognizing a language

- A machine **recognizes** a language if it accepts all strings in the language
  - It may not accept any other strings
  - It may reject or loop on other strings
- A machine **decides** a language if it accepts all strings in the language, and rejects all strings not in the language
  - It may not loop