Three Kinds of Tasks

CMPT 145

Algorithms

- An algorithm is a sequence of instructions that accomplish a stated task.
- Example tasks:
  - Calculate the average of a collection of numbers
  - Calculate the square root of a number
  - Check if a binary tree is ordered.

How do you design an algorithm
if you do not already know
how the algorithm should work?

Study algorithms designed by someone else.

Algorithms Unit Overview

1. Tasks: What kinds of tasks do we write algorithms for?
2. Algorithm Styles: What kinds of algorithms are there?
3. Examples: We study example algorithms for a variety of tasks.

Learning Objectives: Tasks

- To distinguish between search tasks, decision tasks, and optimization tasks.
- To give examples of search tasks, decision tasks, and optimization tasks.

Example Tasks

- Sort a list of numbers
- Collect records for all employees hired after 2015
- Target the closest enemy unit
- Plan an itinerary from Saskatoon to Hong Kong
- Find a time that all project members can meet
- Find values for x, y, z that satisfy a system of equations

Three kinds of tasks

- Search tasks
  - Find a value that satisfies given requirements
- Decision tasks
  - Determine if there is a value that satisfies given requirements
- Optimization tasks
  - Of all the values that satisfy given requirements, find the best one.

Search Tasks

- Find a value that satisfies given requirements
- When you find a value, you can verify that it satisfies the requirements (or not)
- Examples:
  - Sort a list of numbers
  - Plan an itinerary from Saskatoon to Hong Kong
  - Find a time that all project members can meet

Decision Tasks

- Determine if there is a value that satisfies given requirements.
- Decision tasks are Search tasks turned into yes-no questions.
- Examples:
  - Is this given list sortable?
  - Can I get from Saskatoon to Hong Kong?
  - Is there a meeting time that everyone can attend?
- If you can solve the associated Search task, the answer is yes.

Optimization Tasks

- Of all the values that satisfy given requirements, find the best one.
- An optimization problem is a search problem.
- When you find a value, you have 2 verifications:
  1. The value satisfies the given requirements
  2. No other value is better.
- Examples:
  - Find the closest enemy unit.
  - Find the cheapest/fastest/shortest itinerary.
  - Find the earliest time for a meeting
  - Find a meeting time where the most group members can attend

Exercise: Search, decision, or optimization?

- Evaluate an arithmetic expression
- Find a path through a maze
- Find the shortest path through a maze
- Determine if a binary tree is ordered
- Check if a square is magic
- Arrange numbers 1-9 into a magic square
- Is there a magic square using odd numbers 1, 3, …, 19?
- Calculate the nth Fibonacci number
- Substitute data value t with v in a node-chain
- Determine the length of a node-chain

Example Problems

To learn about algorithms, we will study these problems:

- Subset Sum
- Maximum Slice
- Making Change
- Maximum Tree Path
- Leap Line

They have interesting algorithms!

Subset Sum

- Given:
  - List of positive numbers, L
  - Target value T
- Find a list of numbers M, taken from L, whose sum is exactly T.

- Example:
  - $L = [1, 3, 5, 7]$
  - $T = 8$
- Solution: $M = [1, 7]$.

- Search problem: we can check any M comparing sum to T.
- The solution, M, is a list, and we can construct a solution by inserting numbers from L.

Maximum Slice

- Given a list of numbers, L
- Find the slice from index a to index b that has the largest sum of all possible slices of L.

- Example: $L = [1, -2, 3, 4, -5]$
- Solution: $L[2:4]$

- Optimization problem: We have to know that no other slice has a bigger sum.
- The solution $L[a:b]$ can be constructed by exploring different indices a and b.

Making Change version 1

- Given:
  - Positive integer D
  - A list L of coin values
- Find a list of integers C, indicating how many of each coin are needed to have the value of D exactly.

- Example:
  - $D = 37$
  - $L = [1, 5, 10, 25]$
- Solution: $C = [2, 2, 0, 1]$.

- Search task: C can be checked if its value is D.
- The solution is a list, C, and we can build this list by adding coins.

Making Change version 2

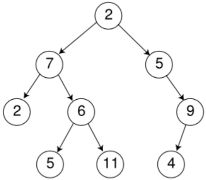
- Given:
  - Positive integer D
  - A list L of coin values
- Find the of integers C, whose value is D, but which has the fewest coins.

- Example:
  - $D = 37$
  - $L = [1, 5, 10, 25]$
- Solution: $C = [2, 0, 1, 1]$.

- Optimization task: C's value can be checked, but the claim that C has the smallest number of coins cannot be checked by looking only at C.
- The potential solutions are lists, and we can build them by adding coins.

Maximum Tree Path version 1

- Given:
  - Binary tree T
  - Integer value V.
- Find a path from the root to any leaf with a sum at least V.

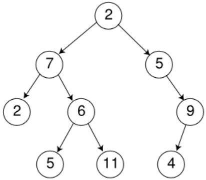


$V = 25$
Solution: $2 \Rightarrow 7 \Rightarrow 6 \Rightarrow 11$

- Search task: The path sum can be verified.
- The path can be constructed by choosing to go left or right at any node.

Maximum Tree Path version 2

- Given:
  - Binary tree T
  - Integer value V.
- Is there a path from the root to any leaf with a sum at least V?

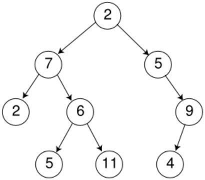


$V = 25$
Solution: $2 \Rightarrow 7 \Rightarrow 6 \Rightarrow 11$

- Decision task: The path sum can be verified.
- The path can be constructed by choosing to go left or right at any node.

Maximum Tree Path version 3

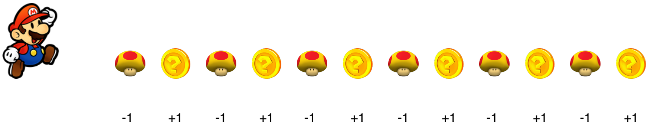
- Given:
  - Binary tree T
- Find the path from the root to any leaf with the maximum sum of any path.



Solution: $2 \Rightarrow 7 \Rightarrow 6 \Rightarrow 11$

- Optimization task: The path sum can be verified, but we need to look beyond the path to verify it's the maximum path.
- Each path can be constructed by choosing to go left or right at any node.

Leap Line



- Mario moves left to right only.
- Mario can step on, or jump over, any of the items.
- Stepping or landing on a coin gives Mario 1 point.
- Stepping or landing on a mushroom deducts 1 point.
- Jumping over any item gives 0 points.
- Given a sequence of coins/mushrooms, what's Mario's highest point total?
- Optimization: Find the best sequence of steps/jumps.
- Each sequence of steps/jumps can be constructed.

How do you design an algorithm
if you do not already know
how the algorithm should work?