

GEM 1501 Problem Solving With Computers

Lecture 4:

Algorithmic Methods

Martin Henz

Summary of Previous Lecture

- Programming Languages
- Data Types in Oz

Overview of Today's Lecture

- **Searches and traversals**
- Divide and conquer
- Greedy algorithms
- Dynamic programming

Searches and Traversals

- Data structures provide complex information
- Structures have information implicit
- Searches and traversals make information explicit

Examples

- Height of trees
- Number of nodes
- Maximal polygonal distance

Overview of Today's Lecture

- Searches and traversals
- **Divide and conquer**
- Greedy algorithms
- Dynamic programming

Divide-and-Conquer

- If the problem is small enough, the solution is immediate
- If not, split the problem into smaller ones.
- Solve the smaller problems separately.
- Combine the solutions of the smaller problems into a solution of the bigger problem.

Examples for Divide-and-Conquer

- Towers of Hanoi
- Finding minimal and maximal elements in a list
- Merge sort

Towers of Hanoi

- Move a tower of rings from A to B using C
- No larger ring can sit on top a smaller ring
- Idea: use subroutine for moving smaller tower

Subroutine: move N from X to Y using Z

1. if N is 1 then output "move X to Y"
2. otherwise
 - (a) call move $N - 1$ from X to Z using Y
 - (b) output "move X to Y"
 - (c) call move $N - 1$ from Z to Y using X

Finding Minimal and Maximal Elements

- Input: Unsorted list of integers L
- Output: Largest and smallest element of L

Divide-and-Conquer Algorithm

1. If L has one element, this element is the smallest and largest
2. If L has two elements, take the smaller and larger ones
3. Otherwise:
 - (a) Split L in two halves: L_{left} and L_{right}
 - (b) Find their extremal elements: MIN_{left} , MIN_{right} , MAX_{left} , MAX_{right} .
 - (c) The smaller one of MIN_{left} and MIN_{right} is the smallest element of L .
 - (d) The larger one of MAX_{left} and MAX_{right} is the largest element of L .

Merge-sort

- Input: unsorted list of integers
- Output: sorted list with the same elements
- Idea:
 - If the list has one element, it is sorted.
 - If not, split it in half, sort the halves and merge them together.

Overview of Today's Lecture

- Searches and traversals
- Divide and conquer
- **Greedy algorithms**
- Dynamic programming

Greedy Algorithms

- Idea: Start somewhere and construct the solution step-by-step
- Leads to very efficient algorithms
- Works for some problems, but by far not all
- Requires proof of correctness

Examples

- Navigation in a source/sink directed graph
- Railroad contractor

Navigation in a Source/Sink Directed Graph

- Input: Directed acyclic labelled graph, where A is the only node that has no incoming edge, and B is the only node that has no outgoing edge.
- Output: A path from A to B

Railroad Contractor

- Goal: Construct the cheapest railroad system for a country
- Input: Labelled graph, two nodes A and B
- Output: Subgraph with smallest sum of labels
- This problem is known as “Minimal Spanning Tree Problem”

Overview of Today's Lecture

- Searches and traversals
- Divide and conquer
- Greedy algorithms
- **Dynamic programming**

The Weary Traveller

- Input: Directed labelled graph, two nodes A and B
- Output: Path from A to B with smallest sum of labels
- This problem is known as “Shortest Path Problem”

Dynamic Programming

- Idea: On the path to a solution, accumulate useful information
- Re-use this information whenever needed
- Often surprisingly efficient algorithms
- Applied on a wide variety of problems