CS 231 course note

Chenxuan Wei

May 2022

Contents

1 Fundamentals

1.1 Specificying a problem

1. Problem
   - Input specification: show type of input
   - Output specification: show how this is related to output
2. Recipe
   - Make the problem general
   - From the input
   - From the output
3. instance
   of a problem is a specifica data that satisidies the input specification
4. Solution
   to an instance of a problem satisifies the output specification

1.2 Types of data

1. Grids
   a 2 demetional sequence (start from 0 like a array)
2. Tree Terms
   - leaf: node with no child
   - internal node: node that is not a leaf
   - siblings: nodes with the same parent
   - subtree: atree within a tree consisting oa a node and all it's descendants
3. Graph
   a tree without a root
4. Data for degining problems with size
   input
   - Numbers: constant
   - String: n = length
   - Sets: n = number of element in set
   - Sequences: n = length
   - Grids: r = number of row, c = number of columns
   - Trees: n = number of nodes
   - Graphs: n = number of vertices: m = number of edges

   Output
   - Ordering of data items
   - Categorization of data items
   - Subset of data items

1.3 Type of problem

5. Optimization Problem
   Constructive: find optimal solution
   evaluation: find optimal value

6. Decision problem
   a problem that answer yes/no to a question

7. Search problem
   find a feasible solution that satisified a consition

8. Counting and enumeration problems
   counting: number of solution that satisified a condition
   emueration problem: all solution satisified a condition

1.4 Paradigms

1. Exhaustive search
    - Sketch
        - Generate all possiblities
        - Extract infomation
        - Determine the solution
    - checklist
        - Definition of set of possiblities
        - process for generating all possibilities or next possibility
        - Definition of information to extract
        - Porcess for extracting information
        - process for forming the solution from all

2 Order notation

2.1 Running time

1. Average case:
   the value of f(k) is the sum over all instance I of size k of the probability of I multiplied by the running time of the algorithm on instance I

2. Best case
   the best possible running time in terms of k

3. Worst case
   the worst possible running time

2.2 Categories

1. notation
   only have 1, logn, n, n^2, 2^n
   simple function: only have one term with no coefficient
   log is base 2
   only look at domenant term

2. Recipe
   use a simple function, remove all constant, only have dominant term

3. Different notation

   - θ: upper/lower bound
   - O: upper bound
   - ω: lower bound

2.3 Order notation

1. formal definition of $O()$
   $f(n)$ is in $O(g(n))$ if there is a real constant $c > 0$ and constant $n_0 \geq 1$ such that
   $f(n) \leq cg(n), \forall n \geq n_0$

2. formal definition for $\omega()$
   $f(n)$ is in $\omega(g(n))$ if there is a real constant $c > 0$ and constant $n_0 \geq 1$ such that
   $f(n) \geq cg(n), \forall n \geq n_0$

3. formal definition for $\theta()$
   $f(n)$ is in $\omega(g(n))$ if there is a real constant $c_1, c_2 > 0$ and constant $n_0 \geq 1$ such that
   $c_2 g(n) \geq f(n) \geq c_1 g(n), \forall n \geq n_0$

3 Algorithm analysis

3.1 Pseudocode

1. Grammer
    - Variable capitalized, function all capital
    - Simple python list operation is allowed
    - slice of [a:b] can be used
    - use append(L, 4) for use function
    - reserved words are bold
    - Assignment use $<-$
    - "for each .. in" for for loop

3.2 Analysis

1. Constant time operation
    - Assignment
    - use a variable
    - using an arithmetic or boolean operation or a comparison
    - Moving to another line in the program
    - Returning a value using return
2. Recipe for worst-case running time
    - Break each block into blocks
    - Determine a bound on each block individually
    - Retain all dominant costs
    - Use θ if all costs are expressed in θ or use O

3.3 Exhaustive search

1. Algorithm
    - Identify an exhausitive search
    - Create a ES algorithm
    - Analyze the run time
    - Implement it