Outline

• Review of simple data structures
• Intro to algorithm design

Fun with data structures

• Linked lists and arrays
• Stacks and queues
• Graphs
• Rooted trees

Stack

• Create(S)
• Push(x,S)
• x=Pop(S)

Stack

Implement with linked lists or arrays to get O(1) per operation:
Create(S) (create an empty stack)
Push(x,S)
x=Pop(S)

Queue

• Create(Q)
• Enqueue(x,Q)
• x=Dequeue(Q)
Queue

Implement with linked list or (circular) array to get $O(1)$ time per operation

Create(Q)
Enqueue(x,Q)
x = Dequeue(Q)

Graphs

- Is $(u,v)$ an edge of $G$?
- What are the neighbors of $v$ in $G$?
- Add vertex/edge
- Remove vertex/edge

Graph – adjacency list

Graph – adjacency matrix

Rooted Trees

- Who is the parent of $v$?
- Who are the children of $v$?
- Who is the predecessor of $v$?
- Who is the successor of $v$?
Rooted Trees

Algorithm Design Techniques

- Induction
- Divide and Conquer
- Dynamic Programming
- Greedy
- Reduction

Induced Subgraphs

- Let $G=(V,E)$ be a graph and let $W$ be a subset of $V$.
- The subgraph of $G$ induced by $W$ is the graph with
  - Vertex set: $W$
  - Edge set: $\{(x,y) \in E \text{ such that } x \text{ and } y \text{ are in } W\}$

Example

Maximal Induced Subgraph

- Input: A graph $G=(V,E)$ and an integer $k$.
- Output: A largest subgraph $G'=(V',E')$ of $G$ such that every vertex of $G'$ has degree at least $k$. 

Maximal Induced Subgraph Example

- Input: $G = K=2$
- Output: 

Maximal Induced Subgraph Example

- Input: $G = K=3$
- Output: $\emptyset$

MIS\(_n\) Algorithm

Finds MIS on graphs with \(n\) or fewer node

MIS\(_{n+1}\) Algorithm

MIS\(_n\) Algorithm

Correctness

Let \(H\) be a maximal induced subgraph of \(G\) with degree at least \(k\).
- Case 1: Every vertex of \(G\) has degree at least \(k\):
- Case 2: A vertex \(x\) of \(G\) has degree less than \(k\):

Running Time

(Assume \(G\) has \(n\) vertices and \(m\) edges)
- Adjacency Matrix: \(T(n) = T(n-1) + cn^2\)
- Adjacency List: \(T(n,m) = T(n-1,m-1) + m\)
- Other:

Polynomial Evaluation

- Input: Integers \(a_n, a_{n-1}, \ldots, a_0\) and an integer \(x\).
- Output: \(P_n(x) = \sum_{i=0}^{n} a_i x^i\)
Polynomial Evaluation

Example

• Input: 3, 1, 0, 2 and 2
• Output: 22
• Explanation: \( P(x) = 3x^3 - x^2 + 2; \ P(2) = 22 \)

Naïve Approach

• Number of multiplications:
  \[ \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \]
• Number of additions: \( n \)

PE\(_n\) Algorithm

Evaluation of polynomials of degree at most \( n \)

\[ \text{PE}(a_n, \ldots, a_0, x) \]

• If \( n=0 \) then return \( a_0 \)
• Else return \( a_0 + x \cdot \text{PE}(a_n, \ldots, a_1, x) \)

Inductive Approach

• Number of multiplications:
  \[ M(n) = 1 + M(n-1), \ M(0) = 0 \]
  Closed form: \( M(n)=n \)
• Number of additions:
  \[ M(n) = 1 + M(n-1), \ M(0) = 0 \]
  Closed form: \( M(n)=n \)

Vertex Cover

• Let \( G=(V,E) \) be a graph
• A vertex cover of \( G \) is a subset \( W \subseteq V \) such that for every edge \( e=(x,y) \) of \( G \) either \( x \) is in \( W \) or \( y \) is in \( W \)
Vertex Cover Example

Some vertex covers:
\{v_1, v_2, v_0, v_4\},
\{v_2, v_1\}

Vertex Cover in Forests

- Input: Forest \(F=(V,E)\)
- Output: A smallest vertex cover of \(F\)

A few observations and definitions

- A collection of trees is a forest
- A tree on \(n\) nodes has \(n-1\) edges
- A tree never has a cycle
- A tree is always connected
- A tree need not be rooted
- A node with 0 or 1 edges in a tree is a leaf
- A (non-empty) tree always has a leaf

VCF<sub>n</sub> Algorithm
Vertex Cover in forests with at most \(n\) nodes

Simple Case

- If \(F\) has no edges then return _________

Suppose \(F\) has an edge:
Find something to keep or throw away …
Claim

• If u is a leaf of F and v is adjacent to u then some smallest vertex cover of F includes v.
• Proof:
  – Let W be a smallest vertex cover of F.
  – If v is not in W then u must be.
  – But then W-\{u\}+\{v\} is also a smallest vertex cover of F.

Vertex Cover in Forests

If F has no edges return \(\emptyset\)
Else {
  Let u be a leaf with an edge in F
  Let v be the node adjacent to u in F
  Let F’ be the subgraph of F induced by V-\{u,v\}
  Return \{v\}+\text{VCT}(F’)
}