Data structure [A07]

김종규, PhD

2017-04-17

Reviews

- quiz review

Subjects

- Tree algorithms
  - Traversal
  - Height
- Graphs and their properties

Recall: Polish calculator

- 4 * 2 + 3 =
  - 4 2 multiply $\longrightarrow$ 8
  - 3 add $\longrightarrow$ 11

$\longrightarrow$ Reverse Polish notation

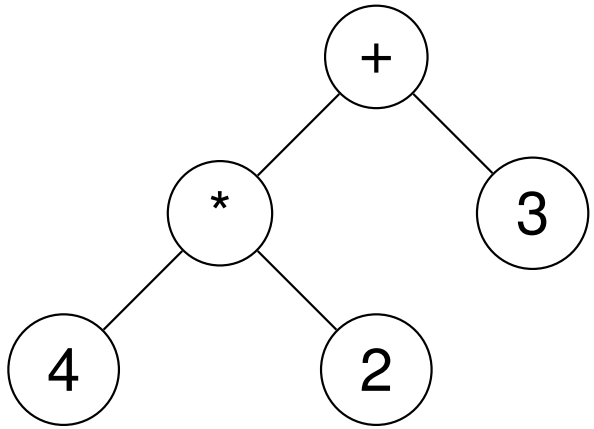
- 4 2 multiply 3 add

- 장점? Stack 을 이용하여 간단히 계산

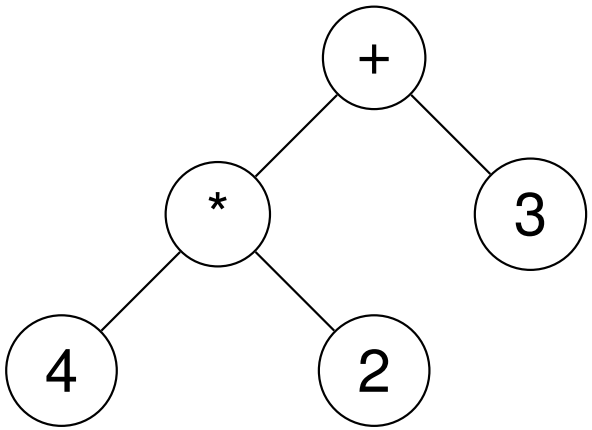
Expression tree

- 4 * 2 + 3: Operator has left and right operands

$\longrightarrow$ Binary Tree representation



Expression tree



- Print children and the operator: 4 2 * 3 +

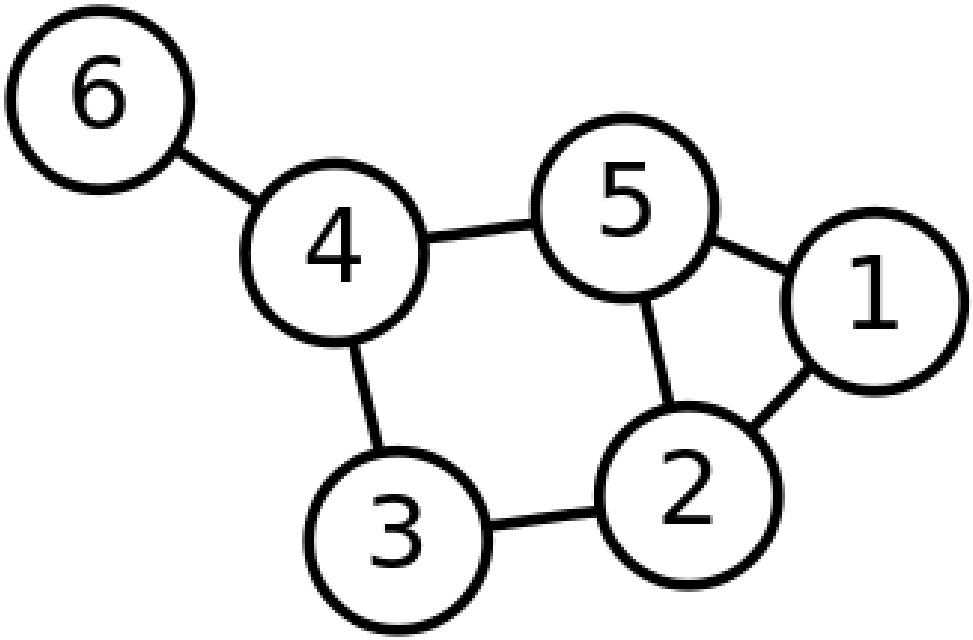
$\longrightarrow$ post-order traversal

Tree traversal

- Visiting every node in a tree
- Visiting children first $\longrightarrow$ post-order traversal
- Visiting children last $\longrightarrow$ pre-order traversal
- Visiting left side children, visit the node and then right $\longrightarrow$ in-order traversal (walk)
- Note: In-order traversal of binary search tree produces sorted list

그래프 (Graph)

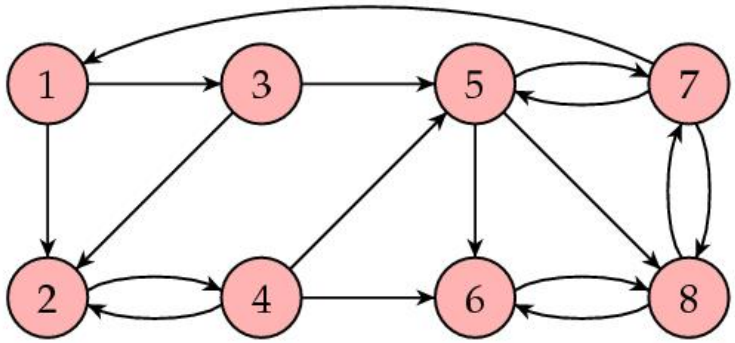
- $G = (V, E)$
  - V vertex
  - E edge



23 B

- Larry Page (Google)

Graph



$$\mathrm{H} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 \\ 1/2 & 0 & 1/2 & 1/3 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 1 & 1/3 & 0 \end{bmatrix}$$

with stationary vector

$$I = \begin{bmatrix} 0.0600 \\ 0.0675 \\ 0.0300 \\ 0.0675 \\ 0.0975 \\ 0.2025 \\ 0.1800 \\ 0.2950 \end{bmatrix}$$

- PageRank algorithm (matrix representation)

Data structure for graphs

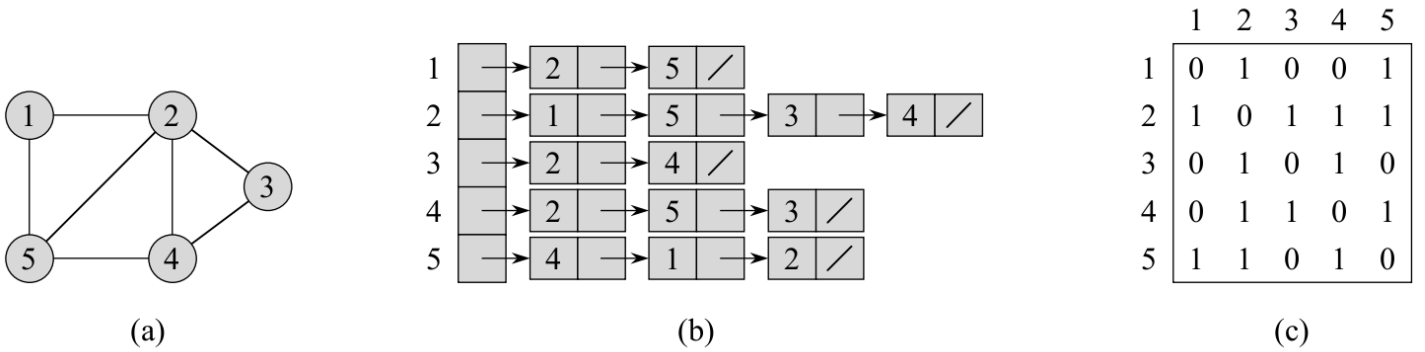


그림: Data structure for graphs

Administrivia

- 중간고사: 다음 주 월 (24일) 수업시간
  - 시험범위: 그래프와 그래프의 성질
  - 중간고사 기간 수업 없음
- Quiz 성적: 가장 낮은 성적 2 개는 제외
- 향후 수요일 수업시간 Notebook 지참 권장

그래프와 관련된 정의

- 인접성 (adjacency): $(u, v) \in E$: u 와 v 가 연결되어 있다
- 방문 (walk): $(v_0, v_1, \cdots, v_{n-1})$
  - $i \in \{1, \cdots, n-1\}, \exists (v_{i-1}, v_i) \in E$
- 경로 (path): $(v_0, v_1, \cdots, v_{n-1})$
  - $(v_0, v_1, \cdots, v_{n-1})$ is a walk (traverse)
  - $\forall_{i,j} i \neq j \Rightarrow v_i \neq v_j$

  $\longrightarrow$ connected: path 가 존재함
- Cycle: $(v_0, v_1, \cdots, v_0)$
  - 시작노드와 종료노드가 같은 경우

Adjacency matrix

- mapping: $\{v_0, \cdots, v_{n-1}\} \rightarrow \{0, 1, \cdots, n-1\}$
- Matrix $E = [a_{ij}]$
- $a_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$

Graph coloring problem

- 그래프에 색깔을 할당했을 때 인접한 두 node 의 색이 같지 않도록 하는 문제
- 주어진 그래프에 할당할 수 있는 최소 color 수는?

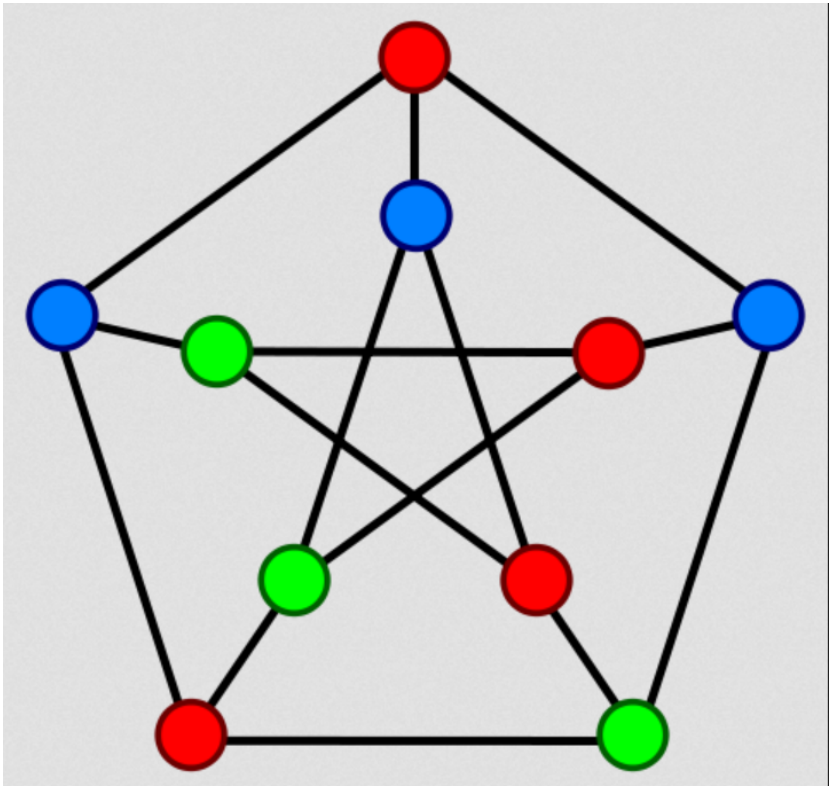
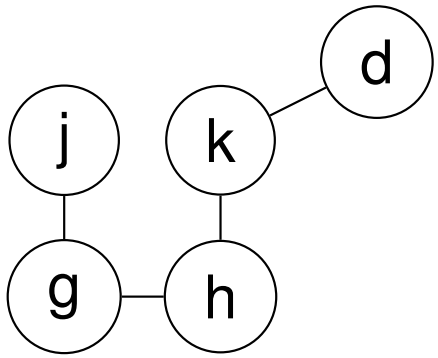


그림: Graph coloring problem

Tree

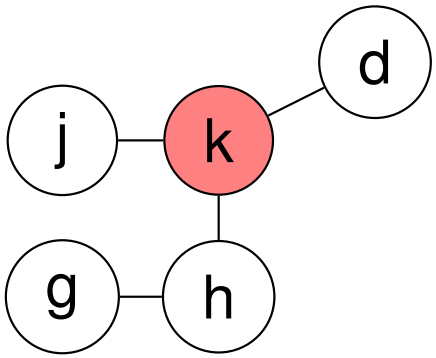
- 특별한 종류의 그래프
  - connected, no cycle (acyclic)
  - cycle is formed if any edge is added
  - not connected if any edge is removed
  - any two vertex is connected by a unique simple path

Tree

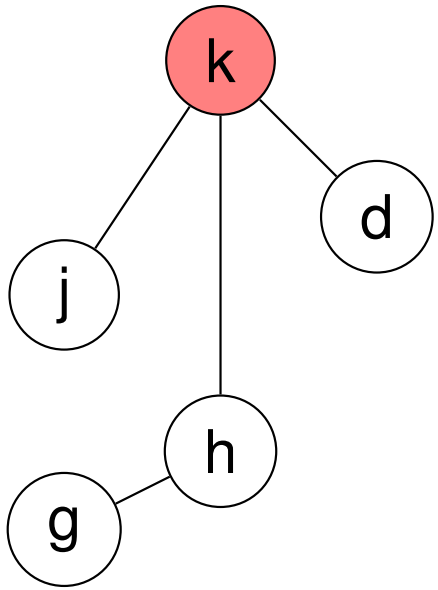


Rooted tree

- 임의의 한 노드를 root 로 지정한 것



Rooted tree



Rooted tree

- Parent: root 방향의 인접한 노드
- Child: root 방향과 반대방향으로 인접한 노드
- Sibling: Parent 가 같은 노드들의 집합
- Descendent: root 의 반대 방향에 있는 한 노드
- Ancestor: root 까지의 경로상에 있는 한 노드
- Sub-tree: 어떤 노드와 모든 descendent 를 포함하는 노드
- Terminal/leaf: child 가 없는 node. deg(n) = 1
- Internal node: leaf 를 제외한 모든 노드
- Height: Root 에서 leaf 까지 도달하는 가장 긴 path 의 길이
- Level: Root 에서 거리가 같은 노드들의 모임

Tree 의 성질

- Graph coloring: 2 colors

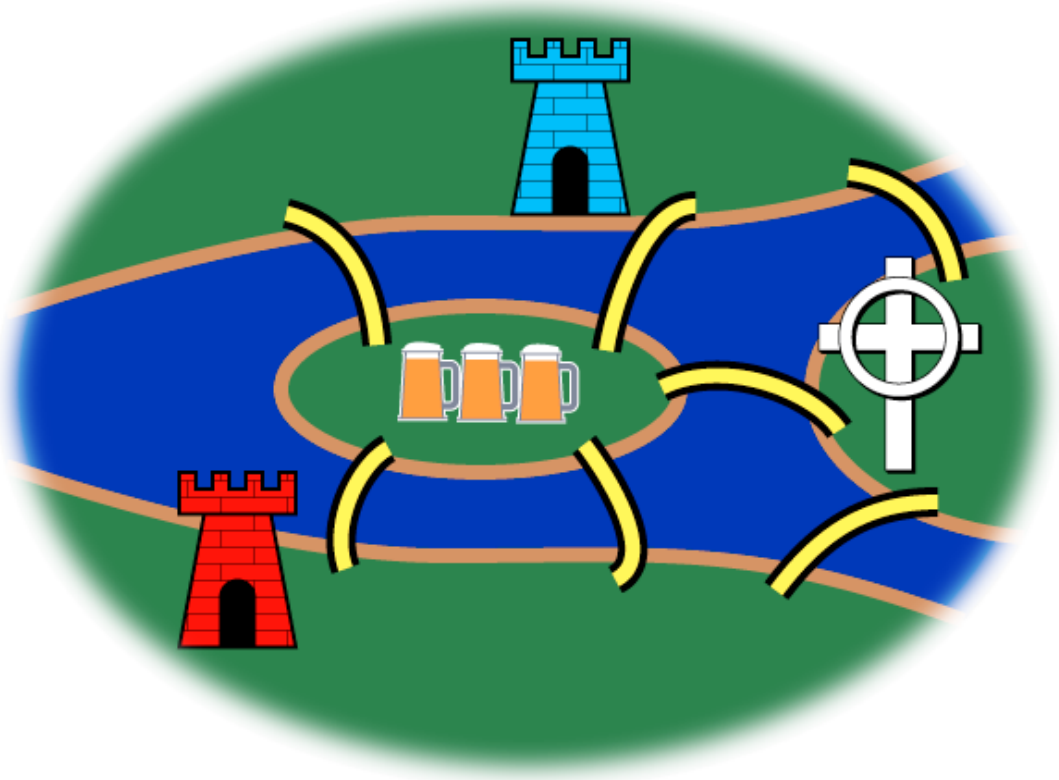
  $\longrightarrow$ Red-black tree

- (Root 노드를 제외하면) 모든 노드가 unique 한 parent 를 갖는다

Tree 의 성질

- n 개의 노드를 갖는 tree 는 $n-1$ 개의 edge 를 갖는다
  - pf by induction $P(n)$
  - basis: $n = 1$, edge 가 없다
  - induction hypothesis: P(n) 에 성립
    - $n+1$ 개의 노드에서 leaf node 를 제거하면 n 개의 노드를 갖는 tree
    - leaf node 는 parent node 와 degree 1 으로 연결되어 있으므로
    - $n+1$ 개의 노드를 갖는 tree 는 $n-1+1$ 개의 edge 를 갖게됨 *Q.E.D*

그래프의 역사: Seven bridges of Königsberg

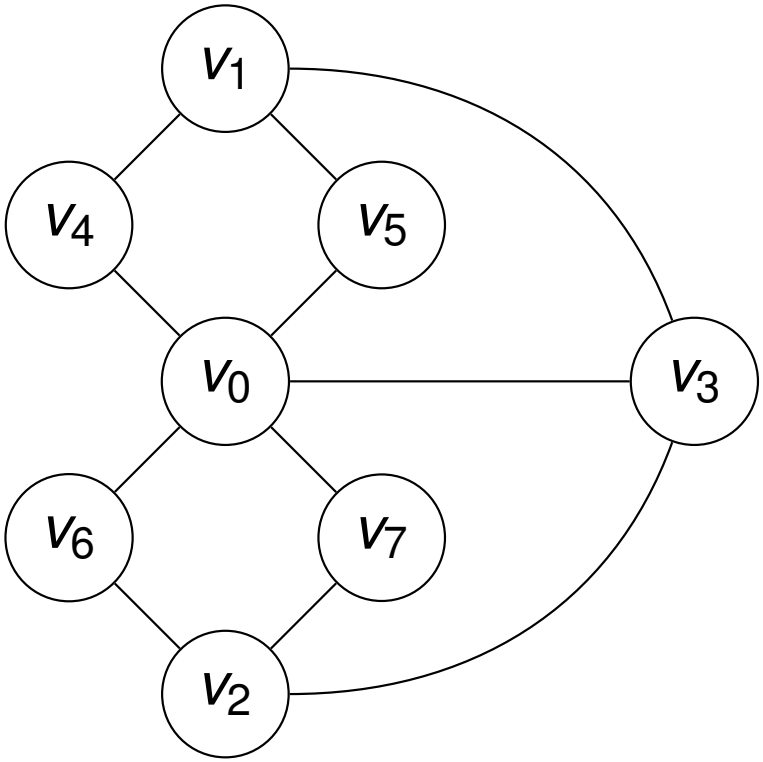


Graph Formulation

- $V = \{v_0 = \text{Guest house}, v_1 = \text{red castle}, v_2 = \text{blue casle}, v_3 = \text{church}\}$
- $E = \{(v_0, v_1), (v_0, v_1), \cdots\}$??
  - Simple graph 가 아니다!

$\longrightarrow$ 손쉽게 simple graph 로 변환 가능

Graph Formulation



- 0: 3, 4, 5, 6, 7
- 1: 3, 4, 5
- 2: 3, 6, 7
- 3: -
- 4: -
- 5: -
- 6: -
- 7: -

$\longrightarrow$ 한 붓 그리기 문제

연습: in-order traversal

- Binary search tree 의 in-order traversal (walk) 을 수행하는 알고리즘을 작성하시오.

```
INORDER-TREE-WALK(x)
1  if x ≠ NIL
2      INORDER-TREE-WALK(x.left)
3      print x.key
4      INORDER-TREE-WALK(x.right)
```

그림: in-order traversal (walk)

Wrap-up

- All nodes in a tree can be visited using recursive algorithm

  $\longrightarrow$ Tree traversal: pre-order, in-order, post-order
- Tree is a special kind of graph
  - Connected, acyclic
- There are a few terminologies to learn regarding trees and graphs