2020년 가을학기 알고리즘

중간고사 리뷰 / GraphBasic

데이터네트워크연구실
문현수, 이영석
munhyunsu@cs-cnu.org

중간고사 Feedback

- 점수 공개
- 평균: 25.86

이번주 실습 목표

중간고사 리뷰

그래프의 표현

- 노드
- 엣지
- 각 언어에서의 구현 (저장)

- 실습1: Neighbor (1점)
- 실습2: Breath First Search (2점)
- 실습3: Depth First Search (2점)

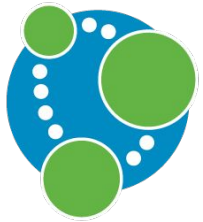
그래프 표현

for 코딩테스트 (구현이 빠른 버전, 문제를 풀어야 할 때)

- Dict/Map 을 사용하면 매우 편리함!
- 노드: Key
- 에지: Value
  - 가중치가 없을 경우: Set
  - 가중치가 필요할 경우: Dict/Map

```
GRAPH = {
    'a': {'b': 8, 'd': 9, 'f': 11},
    'b': {'c': 10},
    'c': {'e': 2},
    'd': {'c': 1, 'f': 3},
    'e': {'g': 4},
    'f': {'g': 8, 'h': 8},
    'g': {'h': 7},
    'h': {'d': 12, 'e': 5}
    }
```

Neo4j



- 응용에서 그래프를 활용할 때에는 단순히 key-value 만으로는 부족함!
- 노드 속성
  - 이름, 가중치, 특징...
- 에지 속성
  - 관계, 가중치, 특징...

for 응용 (구현이 느린 버전, 응용에서 활용하기 좋음)

```
class Graph(object):$
    def __init__(self):$
        self.vertices = dict()$
        self.edges = list()$

class Node(object):$
    def __init__(self, name, weight=0)
        self.name = name$
        self.weight = weight$
        self.edges = dict()$

class Edge(object):$
    def __init__(self, src, dst, weight=0):
        self.src = src$
        self.dst = dst$
        self.weight = weight$
```

```
public class Graph {$
    Map<String, Node> vertices;$
    List<Edge> edges;$
    $
    Graph() {$
        this.vertices = new HashMap<>();$
        this.edges = new LinkedList<>();$
    }$
```

```
public class Node {$
    String name;$
    int weight;$
    Map<Node, Edge> edges;$
$
    Node(String name) {$
        this.name = name;$
        this.weight = 0;$
        this.edges = new HashMap<>();
    }$
```

```
public class Edge {$
    Node src;$
    Node dst;$
    int weight;$
$
    Edge(Node src, Node dst) {$
        this.src = src;$
        this.dst = dst;$
        this.weight = 0;$
    }$
```

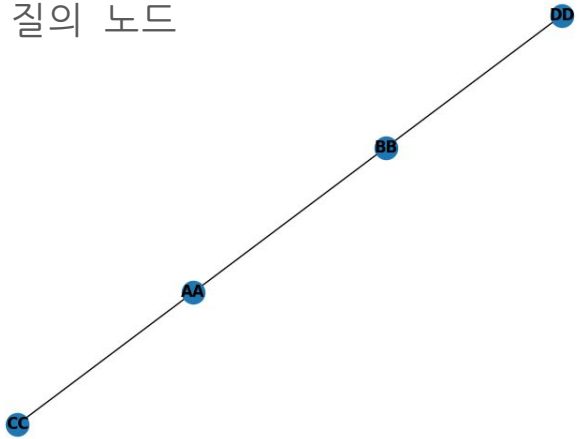
실습1. Neighbor (1)

이웃 - 그래프 표현 연습

- 표준 입력으로 노드와 에지가 주어졌을 때, 이를 무방향 그래프 구조로 저장하고 질의 노드의 이웃 노드 수를 출력하시오
  - 입력) 노드 N개, 에지 M개
    질의 노드 1개
  - 출력) 질의 노드의 이웃 노드 수

```
[Input]
4 4           # N M
AA BB CC DD   # 노드의 이름 N 개
AA CC         # 에지 M개
BB DD
AA BB
BB AA
AA            # 질의 노드

[Output]
2
```

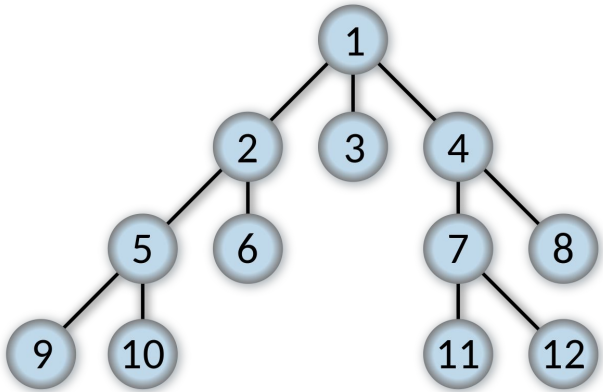


실습2. Breadth First Search (2)
실습3. Depth First Search (2)

Breadth first search: Queue!

- 너비 우선 탐색: 길찾기, Garbage collection 등에서 활용

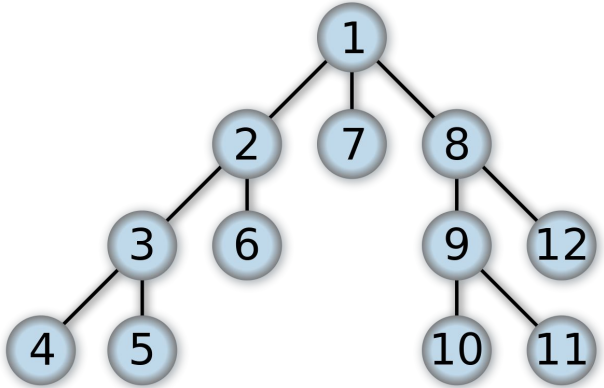
```
procedure BFS(G, root) is
    let Q be a queue
    label root as discovered
    Q.enqueue(root)
    while Q is not empty do
        v := Q.dequeue()
        if v is the goal then
            return v
        for all edges from v to w in G.adjacentEdges(v) do
            if w is not labeled as discovered then
                label w as discovered
                Q.enqueue(w)
```



Depth first search: Stack!

- 깊이 우선 탐색: 미로 탈출, 미로 생성, 길찾기 등에서 활용

```
1  procedure DFS-iterative(G,v):
2      let S be a stack
3      S.push(v)
4      while S is not empty
5          v = S.pop()
6          if v is not labeled as discovered:
7              label v as discovered
8              for all edges from v to w in G.adjacentEdges(v) do
9                  S.push(w)
```



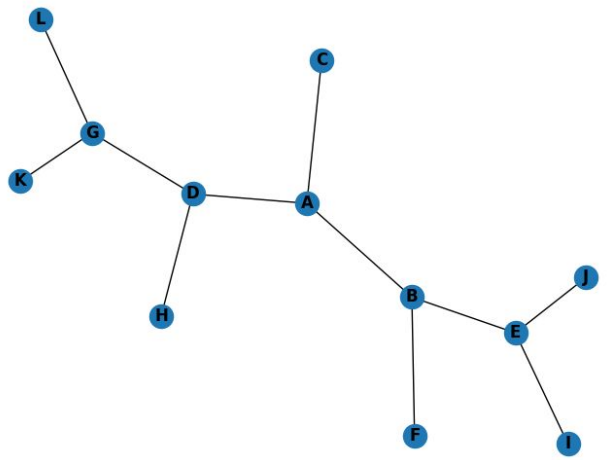
BFS / DFS - 그래프 탐색 연습

- 표준 입력으로 무방향 그래프가 주어지고, 출발 노드가 입력되었을 때 BFS (실습2), DFS (실습3) 로 그래프를 탐색하고, 방문한 노드 순서를 출력하시오.
  - 입력: 이웃 - 그래프 표현과 같음
  - 출력: 방문한 노드
    - 같은 깊이 / 너비일 경우 노드 이름으로 정렬하여 방문

```
[Input]
12 11
A B C D E F G H I J K L
A B
A C
A D
B E
B F
D G
D H
E I
E J
G K
G L
A

[BFS-Output]
A B C D E F G H I J K L

[DFS-Output]
A B E I J F C D G K L H
```



참고

Standard Library

- Python 3: https://docs.python.org/3/library/
- JAVA 11: https://cr.openjdk.java.net/~iris/se/11/latestSpec/api/
- C: https://en.cppreference.com/w/c
- CPP: https://en.cppreference.com/w/cpp

ID/ext	external ID	name
adb	ada	Ada
awk	awk	AWK
bash	bash	Bash shell
c	c	C
csharp	csharp	C#
cpp	cpp	C++
f95	f95	Fortran
hs	haskell	Haskell
java	java	Java
js	javascript	JavaScript
kt	kotlin	Kotlin
lua	lua	Lua
pas	pascal	Pascal
pl	pl	Perl
sh	sh	POSIX shell
plg	prolog	Prolog
py2	python2	Python 2
py3	python3	Python 3
r	r	R
rb	ruby	Ruby
scala	scala	Scala
swift	swift	Swift

주요 링크

- 코딩테스트: http://coding.cnu.ac.kr:8080/domjudge/public
- FAQ : https://docs.google.com/document/d/1ntR6GS1SI7dRbYlw-pu8uT8U65Wc-RtMj10IEpiapfU/
- 질문: 메일!
- 문의 사항: munhyunsu@cs-cnu.org
  - ID / PASSWORD 변경
  - 실습 시간외 질문: [AL]질문제목
- 설문조사: https://docs.google.com/forms/d/e/1FAIpQLSegxN0CaEPiostkOBS-na4xpsl3QBwbrgbAMC_KKNvNb2vIXw/viewform?usp=sf_link

잊지 않아야 할 것) 소스코드 및 보고서

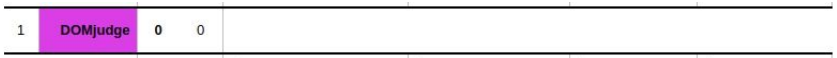
- 사이버 캠퍼스에 목요일까지 제출
  - 추가 시간 필요한 학생들도 목요일까지 제출. 추가 시간 문제 해결은 메일로도 제출!
- 보고서(.pdf 파일), 소스코드(.java, .py 등) zip 파일 압축
  - AL_학번_이름_06.zip (메일 추가 제출) or AL19_06.zip (사이버캠퍼스)
- 시간/공간 복잡도 해석(STL 고려), 자신의 생각, 질문, 느낀점, 공유하고 싶은 문제
  - 문제 해결을 위해 어떤 접근법을 사용하였는지, 무엇을 배웠고 느꼈는가?

어디에 제출해야하는가? 헷갈린다.

- 제한 시간 내에 모두 해결했는가?
  - 네 - 사이버캠퍼스에 소스코드와 보고서 압축해서 제출
  - 아니요 -
    - 추가 시간 내에 해결한 문제가 있는가?
      - 네 - 사이버캠퍼스에 소스코드와 보고서 압축해서 제출하고, 메일로도 제출
      - 아니요 - 사이버캠퍼스에 소스코드와 보고서 압축해서 제출

보고서 템플릿: .pdf 로 제출!

- 알고리즘-x주차-주제
  학번 이름
- 코드 테스팅 결과 (점수표)



- 각 문제별 내용
  a. 문제 / 목표
  b. 해결 방법 (주요 소스코드 첨부. 스크린샷 or 코드 CV)
  c. 결과 (입력, 출력 결과)
- 느낀점: 과제를 하며 느낀 점 / 공유하고싶은 문제 / 난이도 / 부탁 / 조교에게... 등