

# [Algorithm] 7-2. Graph 02 - BFS(Breadth-First Search, 너비우선탐색)

nroo | 2018. 4. 27. 03:27

인프런 - 부경대IT융합응용공학과 궤오흠 교수님의 '영리한 프로그래밍을 위한 알고리즘 강좌'([링크](#))와 '쉽게 배우는 알고리즘 관계 중심의 사고법 - 문병로' 참조

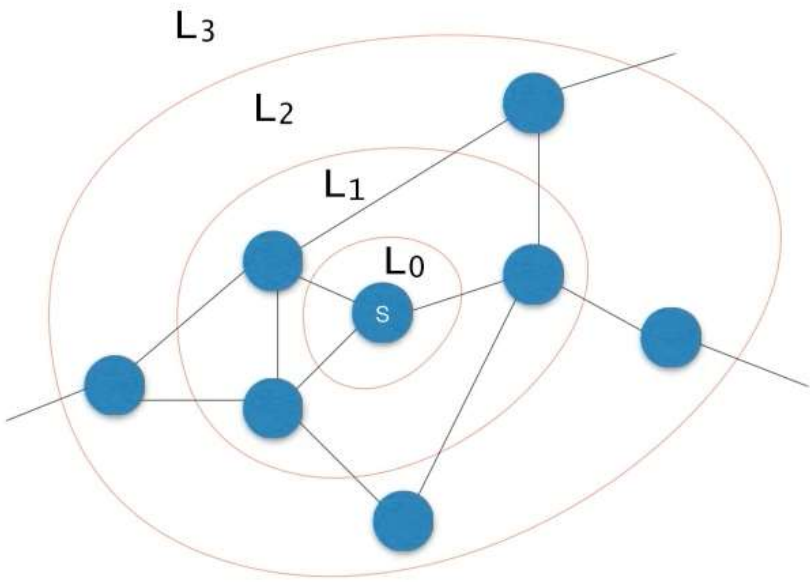
## 7-2. Graph 02 - BFS(Breadth-First Search, 너비우선탐색)

### 그래프 순회

- 순회(traversal)
  - 그래프의 모든 노드들을 방문하는 일
- 대표적 두 가지 방법
  - BFS (Breadth-First Search, 너비우선탐색)
  - DFS (Depth-First Search, 깊이우선탐색)

### 너비우선탐색(BFS)

- BFS 알고리즘은 다음 순서로 노드들을 방문
  - $L_0 = \{s\}$ , 여기서  $s$ 는 출발 노드
  - $L_1 = L_0$ 의 모든 이웃 노드들
  - $L_2 = L_1$ 의 이웃들 중  $L_0$ 에 속하지 않는 노드들
  - ...
  - $L_i = L_{i-1}$ 의 이웃들 중  $L_{i-2}$ 에 속하지 않는 노드들
  - 한마디로 그래프에서 노드들을 동심원의 형태로 순회하는 방법



동심원 형태

### 큐를 이용한 너비우선탐색

- 출발노드를 check하고 시작한다.



Namjun Kim

개발자의 기록습관 | GitHub

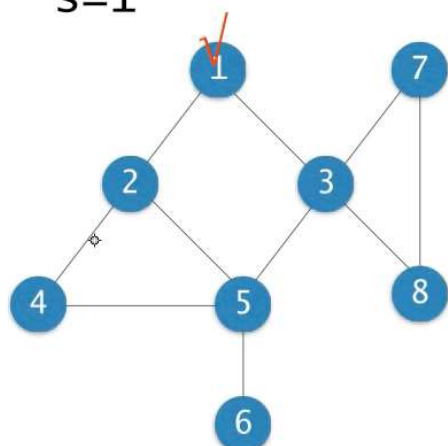


CATEGORY



체크는 이미 방문된 노드라는 표시

s=1



1. **check** the start node;
2. insert the start node into the queue;

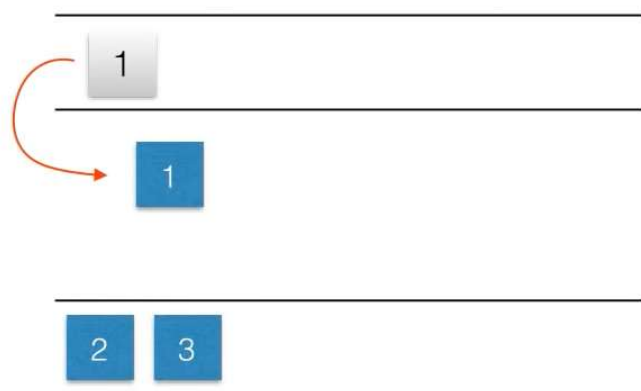
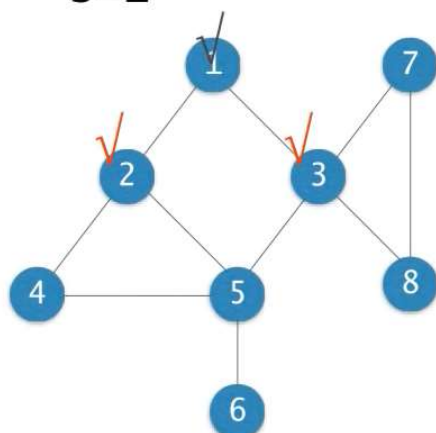
queue

1

- while문을 돌면서 큐가 비어있을 때까지 반복한다.
  - 큐에서 노드(v)를 하나 꺼내고
  - 꺼낸 노드의 인접노드 중, 아직 방문되지 않은(unchecked) 노드들(w)을 체크하고 큐에 넣는다.
  - 이 때, 큐에 넣는 순서는 중요하지 않다.

```
while the queue is not empty do
  remove a node v from queue;
  for each unchecked neighbour w of v do
    check and insert w into the queue;
```

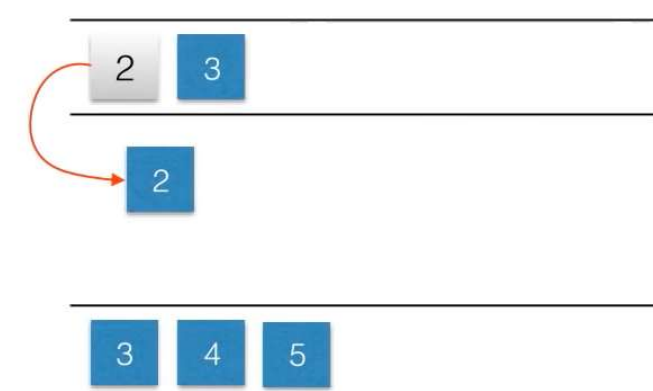
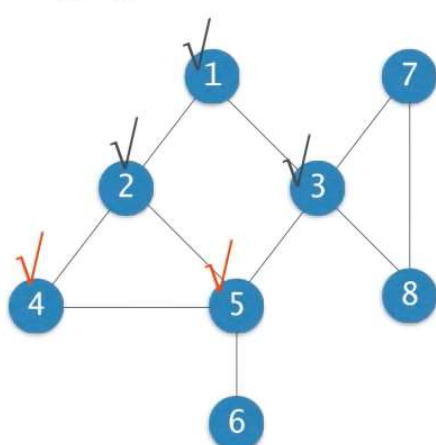
s=1



- 다시 큐에서 노드를 하나 꺼내고(2번 노드)
- 2번 노드의 체크되지 않은 인접 노드들(4, 5번 노드)을 체크상태로 변경하고 큐에 넣는다.

```
while the queue is not empty do
  remove a node v from queue;
  for each unchecked neighbour w of v do
    check and insert w into the queue;
```

s=1



- 이런 방법으로 큐가 비어있는 상태가 될때까지 반복한다.

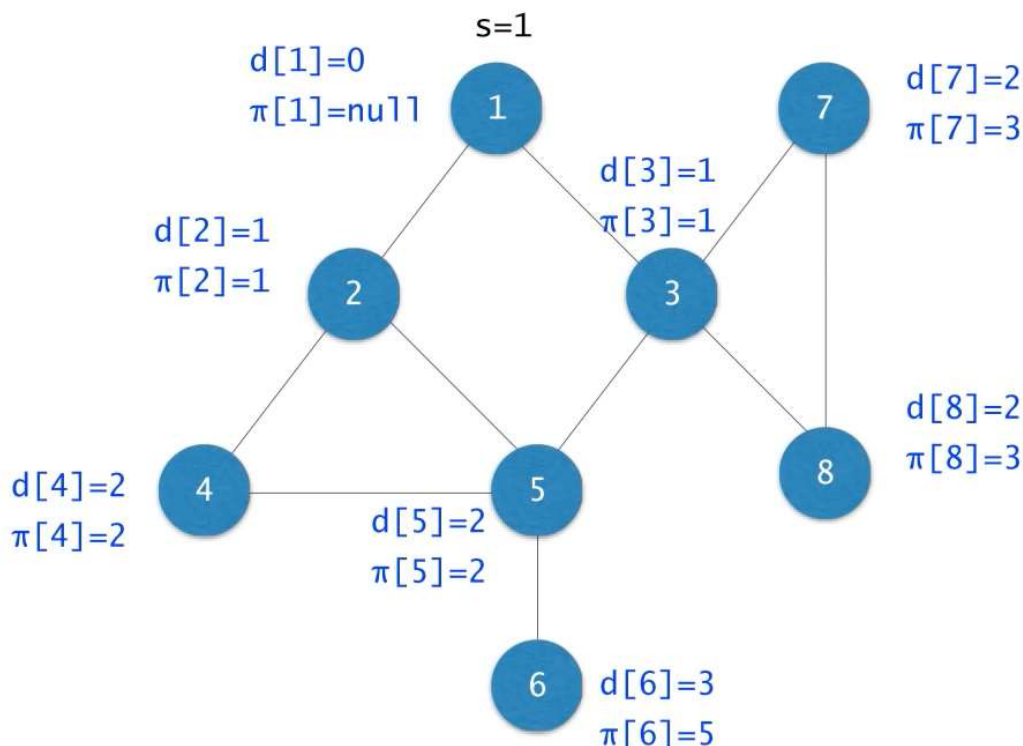




- 10라인의 for문은 u의 degree() 만큼 돈다. 리스트를 인접 행렬로 구현하느냐, 인접리스트로 구현하느냐에 따라 for문의 시간복잡도가 달라진다.
- degree(v)는 어떤 한 노드 v에 실제로 인접한 노드의 수

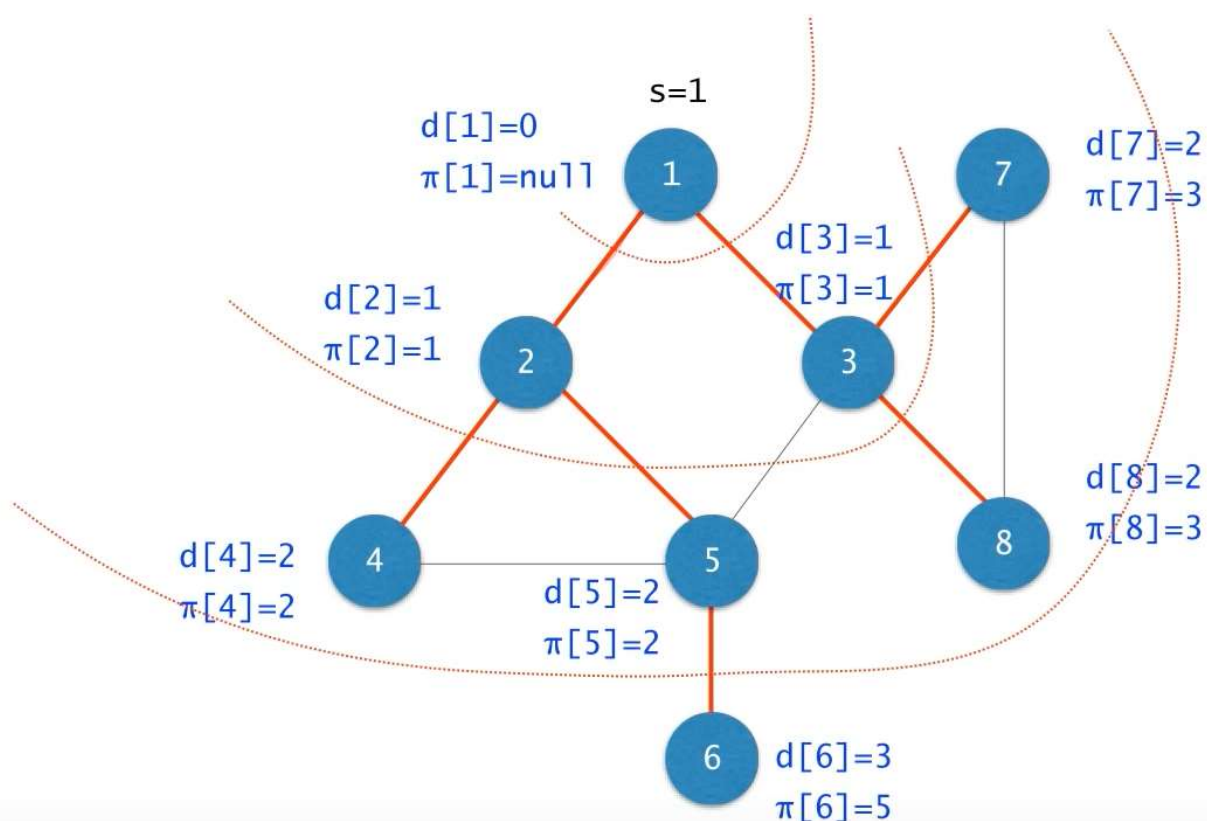
- 그래프를 인접 행렬로 구현할 경우 degree(v)를 찾으려면  $O(n)$ 이 든다. 따라서 인접행렬로 구현했을 때의 while 문의 시간복잡도는  $O(n^2)$ 이 된다.
- 인접 리스트로 구현할 경우 전체 그래프에서 보면, for 문은 결국 모든 노드들의 degree() 만큼 돌 것이다. 인접리스트에서 그것은  $2m$ 이다.(무방향 그래프에서 총 엣지의 수) 시간복잡도는  $O(m)$ . 따라서, while문의 시간복잡도는  $O(n + m)$ 이 된다.
- 결과적으로 인접 리스트의 최악의 경우  $m$ 이  $n$ 이 되므로 최악의 경우가 아닌 이상 인접 리스트로 구현하는 것이 좀 더 효율적이다.

## BFS로 구현한 d[]와 π[] 예시



## BFS 트리

- 각 노드 v와 π[v]를 연결하는 엣지들로 구성되는 트리
- BFS 트리에서 s에서 v까지의 경로는 s에서 v까지 가는 최단 경로
- 어떤 엣지도 동심원의 2개의 layer(L0에서 L2로 가지 않는다)를 건너가지 않는다.(동일 레이어의 노드를 연결하거나, 혹은 1개의 layer를 건너간다.)



## 너비우선탐색: 최단 경로 출력하기

- 출발점 s에서 노드 v까지의 경로 출력하기
  - resursion으로 해결한다.
  - s에서 v까지 가는 최단 경로는 먼저 s에서 π[v]까지 가는 경로를 출력하고, v를 추가로 출력하면 된다.



Namjun Kim  
개발자의 기록습관



CATEGORY



```
02 print s;  
03 else if  $\pi[v]$  = null then // 실제로 s에서 v까지 가는 경로가 없을 경우(최단경로도 없음)  
04     print "no path from s to v exists";  
05 else  
06     PRINT-PATH(G, s,  $\pi[v]$ );  
07     print v;
```

## 너비우선탐색(BFS) 정리

- 그래프가 connected 라면 모든 노드를 방문하게 된다. 하지만, 그래프가 **disconnected** 이거나 혹은 방향 그래프 라면 BFS에 의해서 모든 노드가 방문되지 않을 수도 있다.
- disconnected 그래프의 모든 노드를 방문하려면 BFS를 반복하여 모든 노드 방문
  - 전체 노드중 unvisited 노드가 없을 때까지 BFS를 반복한다.

```
BFS-ALL(G)  
while there exists unvisited node v  
    BFS(G, V)
```



공감



구독하기



'ICT Eng > Algorithm' 카테고리의 다른 글

[Algorithm] 7-3. Graph 03 - DFS(Depth-First ... (0)

[Algorithm] 7-1. Graph 01 - 개념과 표현 (0)

[Algorithm] 6-1. Hashing 개요 - Chaining, Op... (0)

[Algorithm] 7-2. Graph 02 - BFS(Breadth-... (2)

[Algorithm] 6-2. Hash 함수, Hashing in Java (0)

[Algorithm] 5-3. Red Black Tree 03 - DELETE,... (3)

TAG BFS, traversal, 그래프, 너비우선회, 시간복잡도, 알고리즘, 인접그래프, 인접행렬, 최단경로

댓글 2 ^



sunjunjun

안녕하세요 우연히 들르다가 늘 궁금했던 점 질문하려 합니다.  
이러한 전문적 지식들은 어디에 쓰이는 건가요? 웹개발에는 쓰이지 않겠죠?

2018.07.19 01:09




nroo PROFILE


안녕하세요, 알고리즘은 개발자가 갖춰야할 필수지식이라고 생각합니다. 알고리즘, 자료구조, 운영체제, 네트워크, 프로그래밍언어 등 기본기가 잘 다져진 개발자가 그렇지 않은 개발자에 비해 우위에 있을것이고, 현실적으로 보면 요즘 대부분의 서비스 개발 기업들의 채용과정에 알고리즘 코딩테스트 역량과 깊이 있는 기술 면접은 필수 단계에 포함됩니다. 더 궁금한 점 있으시면 얘기해주세요:)

2018.07.19 01:20 신고

Namjun Kim

개발자의 기록습관 |  GitHub

🔍

CATEGORY 

이름

비밀번호

비밀글

입력

<

1

...

51

52

53

54

55

56

57

58

59

...

129

>

공지사항

블로그명 변경

최근에 올라온 글

[Spring]Jackson json des...

[AWS] EC2 서버 생성, 접...

NHN FORWARD 2019 후기

우아한 Redis 세미나 후기

최근에 달린 댓글

혹시 로컬이 아닌 gcp에 원격...

쓰지말아야 할 이유 요약하...

좋은 글 잘 보고 갑니다 감...

자동으로 생성하게 하지 말고...

Total

1,021,176

Today414

Yesterday797

링크

kakao 기술 블로그

우아한형제들 기술 블로그

NAVER D2 개발 블로그

라인 기술블로그

Meetup : NHN TOAST

줌인터넷 기술블로그

티몬의 개발이야기

jojoldu

Carrey`s님의 기술블로그

조대협's 블로그

beyondJ2EE님의 블로그

조인석의 브런치

JBee 블로그

소용환의 생각저장소

권용근님의 블로그

Wisoft Lab.

ngelmaum notes

폴라리언트 장 혁의 브런치

자피킨치블로그

TAG

Algorithm

springboot

JPA

알고리즘

AWS

무선통신소...

Wisoft

시간복잡도

Spring

Vue.js

인프런

라즈베리파이

vuex

젠킨스

한밭이글스

스프링부트

Recursion

레드블랙트리

Raspberry Pi

vuejs

RBT

Spring Boot

IT융합인력...

github

ORM

자바

정렬

한밭대학교

순환

Java

more

« 2022/06 »

일

월

화

수

목

금

토

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

글 보관함

2020/06 (1)

2019/12 (1)

2019/11 (2)

2019/09 (2)