[Algorithm] 31강 : 플로이드 워셜 알고리즘의 정의와 구현 — 나무늘보의 개발 블로그

노트북: 첫 번째 노트북
만든 날짜: 2020-11-19 오전 9:07
URL: https://continuous-development.tistory.com/196

Algorithm

[Algorithm] 31강 : 플로이드 워셜 알고리즘의 정의와 구현

2020. 11. 19. 09:02 수정 삭제 공개

플로이드 워셜 알고리즘

1.1 플로이드 워셜 알고리즘이란?

- 모든 노드에서 다른 모든 노드까지의 최단 경로를 모두 계산
- 플로이드 워셜 알고리즘은 다익스트라 알고리즘과 마찬가지로 단계별로 겨처 가는 노드를 기준으로 알고리즘을 수행
    - 다만 매 단계마다 방문하지 않은 노드 중에 최단 거리를 갖는 노드를 찾는 과정이 필요하지 않는다
- 플로이드 워셜은 2차원 테이블에 최단 거리 정보를 저장
- 다이나믹 프로그래밍 유형에 속한다

1.2플로이드 워셜 알고리즘 점화식

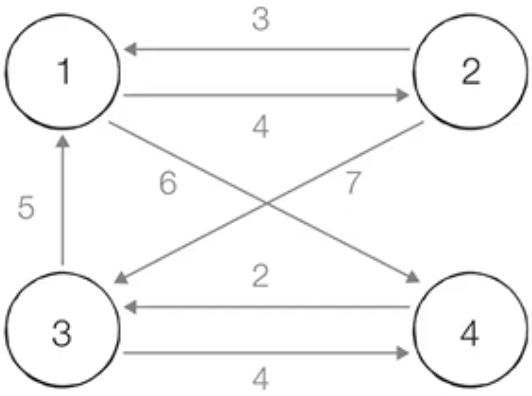
- 각 단계마다 특정한 노드 k를 거쳐 가는 경우를 확인

- a에서 b로 가는 최단 거리보다 a에서 k를 거쳐 b로 가는 거리가 더 짧은 지 검사

$$D_{ab} = min(D_{ab}, D_{ak} + D_{kb})$$

1.3 동작과정

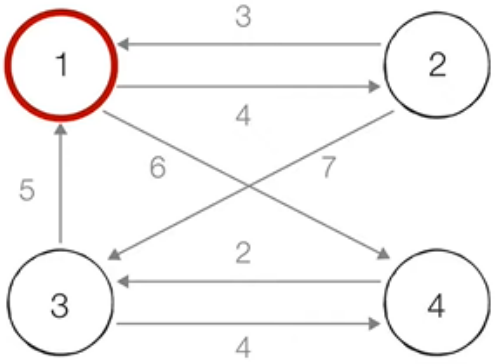
- **[초기 상태]** 그래프를 준비하고 최단 거리 테이블을 초기화합니다.
  - **기본 점화식**: $D_{ab} = \min(D_{ab}, D_{ak} + D_{kb})$



출발 \ 도착	1번	2번	3번	4번
1번	0	4	무한	6
2번	3	0	7	무한
3번	5	무한	0	4
4번	무한	무한	2	0

초기 상태로 해당 테이블이 만들어지고 경로에 대해 만들어진다.

- [Step 1] 1번 노드를 거쳐 가는 경우를 고려하여 테이블을 갱신합니다.
  - 점화식: $D_{ab} = \min(D_{ab}, D_{a1} + D_{1b})$



$$D_{23} = \min(D_{23}, D_{21} + D_{13})$$
$$D_{24} = \min(D_{24}, D_{21} + D_{14})$$
$$D_{32} = \min(D_{32}, D_{31} + D_{12})$$
$$D_{34} = \min(D_{34}, D_{31} + D_{14})$$
$$D_{42} = \min(D_{42}, D_{41} + D_{12})$$
$$D_{43} = \min(D_{43}, D_{41} + D_{13})$$

갱신된 최단 거리 테이블

0	4	무한	6
3	0	7	9
5	9	0	4
무한	무한	2	0

이 과정을 모든 노드에 적용하여 한다.

1.4 알고리즘 구현

```
INF = int(1e9) #무한을 의미하는 값으로 10억을 설정

#노드의 개수 및 간선의 개수를 입력 받기

n = int(input())
m = int(input())

#2차원 리스트(그래프 표현)를 만들고, 무한 으로 초기화
graph = [[INF] * (n+1) for _ in range(n+1)]

#자기 자신에서 자기 자신으로 가는 비용은 0 으로 초기화
for a in range(1,n+1):
    if b in range(1,n+1):
        if a==b:
            graph[a][b] = 0

#각 간선에 대한 정보를 입력 받아, 그 값으로 초기화
for _in range(m):
    # A 에서 B로 가는 비용은 C라고 설정
    a,b,c = map(int,input().split())
    graph[a][b] = c

# 점화식에 따라 플로이드 워셜 알고리즘을 수행
for k in range(1,n+1):
    for a in range(1, n+1):
        for b in range(1,n+1):
            graph[a][b] = min(graph[a][b], graph[a][k],graph[k][b])
```

```
# 수행된 결과를 출력
for a in range(1,n+1):
  for b in range(1, n+1):
    @ 도달할 수 없는 경우, 무한이라고 출력
    if graph[a][b] ==INF:
      print("INFINITY",end=" ")
  print()
```

1.5 플로이드 워셜 알고리즘 성능 분석

- 노드의 개수가 N개일 때 알고리즘상으로 N번의 단계를 수행
  - 각 단계마다 O(N제곱)의 연산을 통해 현재 노드를 거쳐가는 모든 경로를 고려
- 따라서 플로이드 워셔 ㅏ ㄹ 알고리즘의 총 시간 복잡도는 O(N세제곱)이다.

이 자료는 동빈 나 님의 **이코 테** 유튜브 영상을 보고 정리한 자료입니다.
참고 : www.youtube.com/watch?v=m-9pAwq1o3w&list=PLRx0vPvlEmdAghTr5mXQxGpHjWqSz0dgC

'Algorithm' 카테고리의 다른 글□

[Algorithm] 31강 : 플로이드 워셜 알고리즘의 정의와 구현□

[Algorithm] 30강 : 다익스트라 최단 경로 알고리즘의 정의와 구현□

[Algorithm] 29강 : 다이나믹 프로그래밍의 기초 문제 풀이□

[Algorithm] 28강 : 다이나믹 프로그래밍의 정의와 구현□

[Algorithm] 27강 : 이진 탐색 기초 문제 풀이□

[Algorithm] 26강 : 이진 탐색 알고리즘 정의와 구현□

파이썬 플로이드 워셜 알고리즘 플로이드 워셜 경로 알고리즘 플로이드 워셜 알고리즘

나아무늘보

혼자 끄적끄적하는 블로그 입니다.