Data Structure Project
Project #3

담당교수 : 이기훈 교수님

제출일 : 2018. 12. 12.

학과 : 컴퓨터공학과

학번 : 2012722088

이름 : 정 석 호

1. Introduction

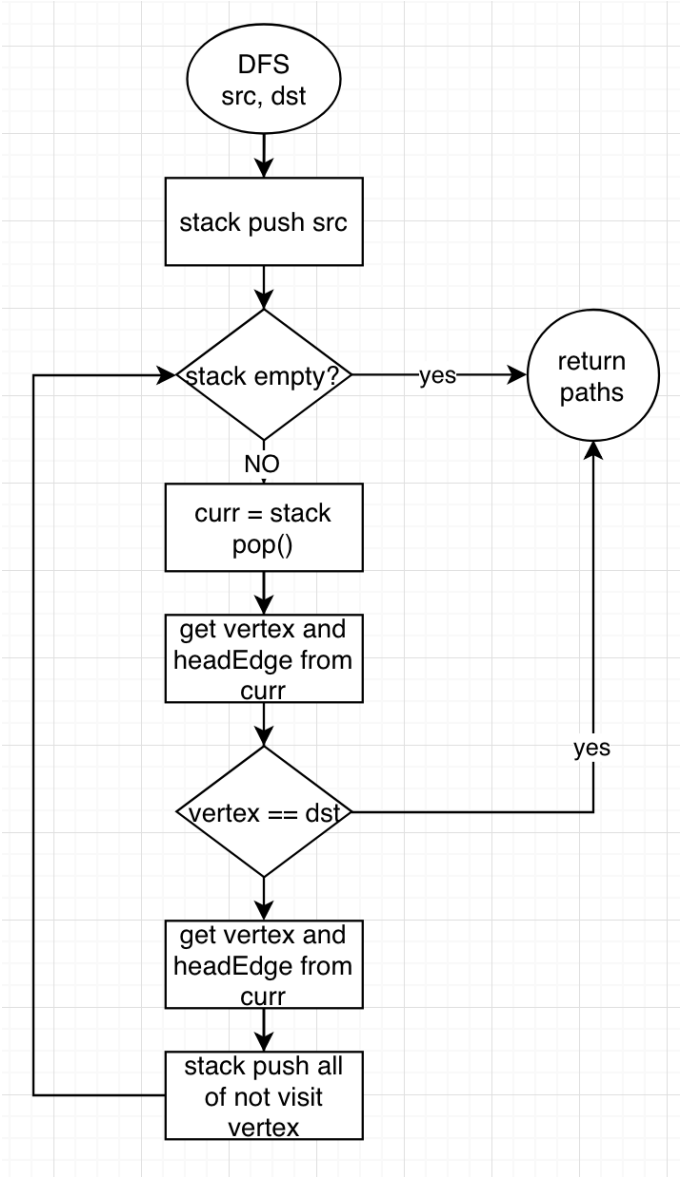
- 프로젝트 내용에 대한 설명

해당 프로젝트 최단 경로를 찾는 알고리즘에 무엇이 있고 어떤 차이가 있으며 같은 알고리즘이여도 어떤 자료구조를 사용하여 구현해야하는 것인지 배울 수 있는 프로젝트입니다.

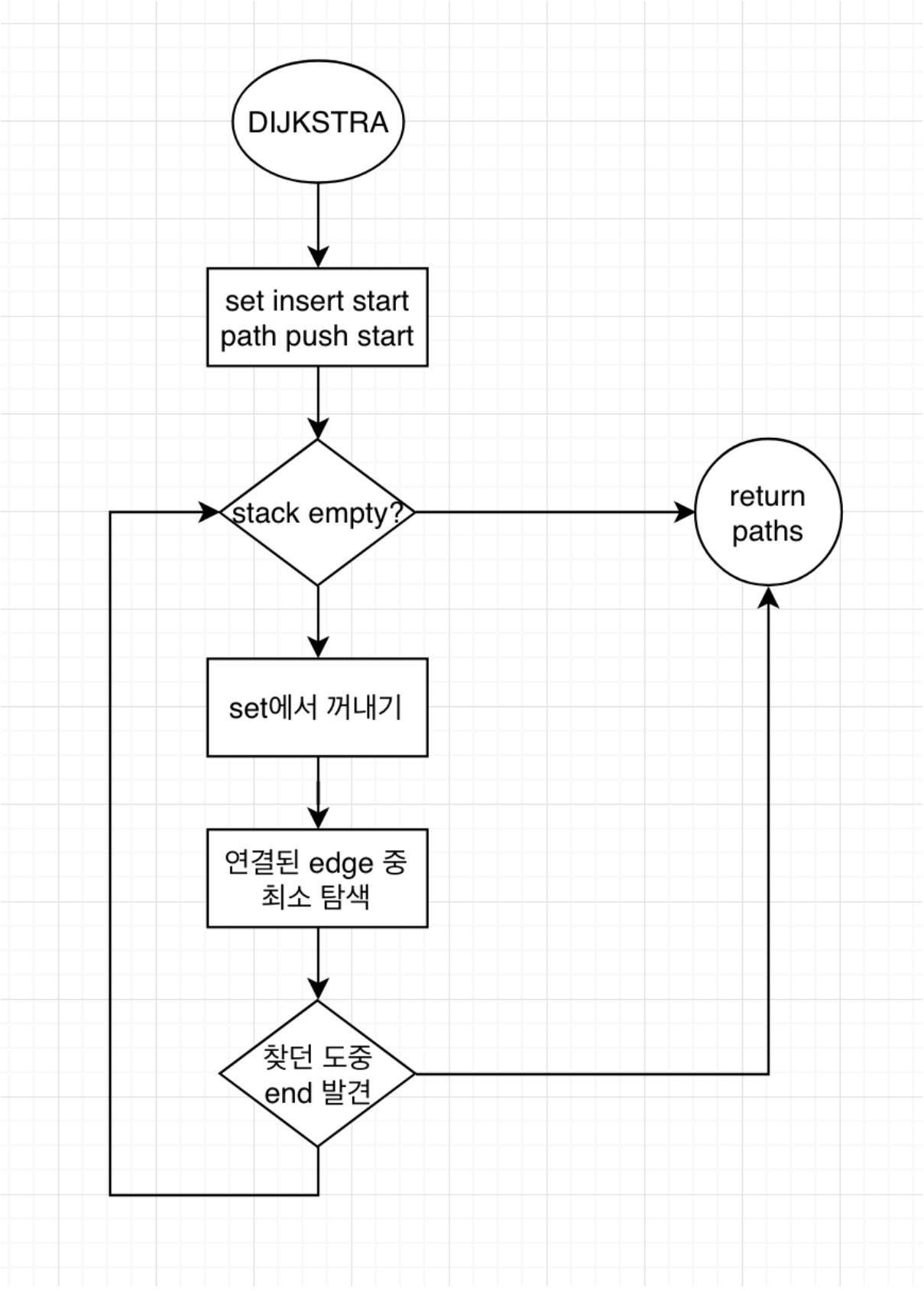
2. Flow Chart

- 설계한 프로젝트의 플로우 차트를 그리고 설명

- DST



- DIJKSTRA


DIJKSTRA
set insert start
path push start
stack empty?
return
paths
set에서 꺼내기
연결된 edge 중
최소 탐색
찾던 도중
end 발견

3. Algorithm

- 프로젝트에서 사용한 알고리즘의 동작을 설명

- DST

1. 방문배열 선언 및 false로 초기화
2. 출발지 vertex를 stack에 푸시
3. stack이 빌 때 까지 while 돌입
4. stack에서 pop한 vertex를 저장합니다.
5. 만일 pop한 vertex가 end라면 while을 종료합니다.
6. 기준으로 연결된 모든 edge 중 방문하지 않은 edge는 stack에 그대로 푸시 합니다.
7. 4번 부터 반복

- DIJKSTRA

1. edge의 key와 weight를 저장할 수 있는 set(std)를 선언
2. sets을 start 키와 가중치 0으로 된 pari를 삽입하여 초기화
3. 반환할 결과 path vector에 start 키 push
4. stack이 빌 때 까지 while 돌입
5. sets에서 첫번째 요소를 꺼내고 기존의 sets에서는 삭제
6. 꺼낸 요소를 current라 할 때 current의 key를 저장
7. 해당 key를 갖는 vertex를 찾습니다.
8. 찾은 vertex와 연결된 edge를 모두 탐색합니다.
9. 연결된 edge의 키와 가중치 중 최소값을 탐색합니다.
10. 찾는데 도중에 end와 동일한 키를 갖을 때 while문을 종료합니다.
11. 끝까지 돌았는데 못찾으면 예외를 발생시킵니다.

4. Result Screen

- 모든 명령어에 대해 결과화면을 캡처하고 동작을 설명
- make 실행 후 ./run 실행

```
g++ -std=c++11 -g -o run Edge.cpp Vertex.cpp Manager.cpp Graph.cpp main.cpp Result.h Vertex.h Manager.h Graph.h Edge.h MinHeap.h Stack.h
```

- command.txt의 내용

```
ubuntu@ip-172-31-20-57:~/git/KW_DS_Project/design/Project3/project/test$ cat command.txt
LOAD mapdata.txt
PRINT
DFS 0 3
DIJKSTRA 0 3
DFS 1 4
BELLMANFORD 1 4
DIJKSTRA -1 10
BELLMANFORD
ASTAR 1 4ubuntu@ip-172-31-20-57:~/git/KW_DS_Project/design/Project3/project/test$ cat
```

- 각 커맨드 의 결과

```
====== PRINT ======
0 6 13 0 0
0 0 5 6 0
2 0 0 7 4
0 6 0 0 3
0 0 5 2 0
==================
===== DFS =====
shortest path: 0 2 4 3
sorted nodes: 0 2 3 4
path length: 19
===============
===== DIJKSTRA =====
shortest path: 0 1 3
sorted nodes: 0 1 3
path length: 12
===============
===== DFS =====
shortest path: 1 3 4
sorted nodes: 1 3 4
path length: 9
===============
==================
Error code: 300
==================
==================
Error code: 300
==================
```

```
====================
Error code: 300
====================
===== DIJKSTRA =====
shortest path: -1
sorted nodes: -1
path length: 0
===============
====================
Error code: 300
====================
====================
Error code: 300
====================
====================
Error code: 300
====================
====================
Error code: 300
====================
```

구현한 LOAD, PRINT, DFS, DIJKSTRA 에 대해서 올바른 결과를 보여주고 있습니다.

5. Consideration

알고리즘 대회 및 코딩 테스트에서 항상 나오던 최단 거리 문제를 프로젝트로 진행하여 문제 해결에 많은 도움을 느꼈습니다. MinHeap을 먼저구현하였으나 시간적 여유가 없어 Dijkstramin을 구현하지 못한 점이 많이 아쉽습니다.

이번 기회를 통해 최단거리에 대한 문제는 어렵지 않게 풀어낼 경험을 얻었습니다.

DFS의 경우 key와 weight 중 어느 것을 이용하여 구현해야 하는지 처음에 헷갈렸으나 '조만재' 조교님이 올려준 DFS 정리 자료를 보고 바로 이해할 수 있었습니다.

현재 진행했던 코드 작성들도 github를 이용하여 계속해서 코드 향상시킬 계획입니다.

4학년 마지막이라 취업준비 및 대외활동으로 인해 프로젝트 진행에 있어서 만족스럽지 못했던 부분이 너무 아쉽습니다. 하지만 데이터구조를 수강함에 따라 더 이상 자료구조에 겁먹지 않고 도전할 수 있는 용기를 얻었습니다.

제 경우는 재수강이 아닌 첫 수강인 이유가 있었는데 진작 들었더라면 좀 더 높은 꿈을 이루지 않았을까 라는 후회가 듭니다.

감사합니다.