

알고리즘

김정훈교수님

소프트웨어학과

2학년

2020039058

김동우

2023.11.26.

1. Kruskal’s Algorithm

Kruskal의 시간 복잡도는 Kruskal의 시간 복잡도는 주로 간선을 정렬하는 시간과 간선을 검사하며 사이클을 확인하고 추가하는 과정의 시간에 의해 결정됩니다.

간선 정렬: O(E log E), 여기서 E는 간선의 수입니다.

Union-Find 연산: O(E log V), 여기서 V는 정점의 수입니다.

간선의 수 E만큼 Union-Find 연산을 수행하면서 사이클을 확인합니다.

따라서 Kruskal 알고리즘의 총 시간 복잡도는 O(E log E + E log V)입니다.

1. Prim's algorithm

Prim의 시간 복잡도는 주로 우선순위 큐를 사용하여 가장 작은 가중치의 간선을 찾는 시간에 의해 결정됩니다.

우선순위 큐 연산:

O((E + V) log V), 여기서 E는 간선의 수, V는 정점의 수입니다.

간선의 수와 정점의 수만큼 우선순위 큐 연산을 수행합니다.

따라서 Prim 알고리즘의 총 시간 복잡도는 O((E + V) log V)입니다.

이러한 시간 복잡도는 최악의 경우를 나타내며, 각 알고리즘의 성능은 사용된 자료구조에 따라 다를 수 있습니다. Union-Find 연산이 효율적으로 구현되면 Kruskal이 더 빠를 수 있고, 우선순위 큐의 효율적인 구현에 따라서 Prim이 더 빠를 수 있습니다.

1. 고찰

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Kruskal은 간선을 가중치별로 정렬한 후에 그 순서대로 선택합니다. 따라서 그래프의 간선이 상대적으로 많이 있을 때, 즉 edges 값이 큰 경우 Kruskal이 특히 빠르게 실행될 수 있습니다. Kruskal 알고리즘에서는 Union-Find 자료구조를 사용하여 사이클을 확인하고 간선을 선택합니다. Union-Find의 효율성은 그래프의 크기와 구조에 따라 다르며, 이 효율성이 높을수록 Kruskal의 성능이 개선될 수 있습니다. 현재 코드에서는 간선의 가중치가 1부터 100까지의 랜덤 한 값으로 설정됩니다. 이 범위 내에서 Kruskal이 더 효율적으로 동작할 수 있습니다. 따라서 이 코드에서는 Kruskal 알고리즘이 더 효율적으로 동작하였음을 알 수 있습니다.