# 강의 16: 최단 경로 II - 다익스트라

## 강의 개요

- 복습
- DAG에서의 최단 경로
- 음의 간선이 없는 그래프에서의 최단 경로
- 다익스트라 알고리즘

## 읽을 거리

CLRS, Sections 24.2-24.3

## 복습

$d[v]$ 는 출발점 $s$ 에서의 현재까지 구한 최단 경로의 길이입니다. 완화 과정을 통해 $d[v]$ 는 최종적으로 $s$ 부터 $v$ 까지의 최단 경로의 길이인 $\delta(s,v)$ 가 됩니다. $\Pi[v]$ 는 $s$ 부터 $v$ 까지의 최단 경로에서 $v$ 의 선행자를 의미합니다.

최단 경로 계산에서의 기본 연산은 완화 연산이라고 할 수 있습니다.

$$
\begin{array}{l}
\text{RELAX}(u, v, w) \\
\quad \text{if } d[v] > d[u] + w(u, v) \\
\qquad \text{then } d[v] \leftarrow d[u] + w(u, v) \\
\qquad\qquad \Pi[v] \leftarrow u
\end{array}
$$

## 완화의 안전성

**보조 정리:** 완화 알고리즘은 불변 조건인 모든 $v \in V$ 에서 $d[v] \geq \delta(s,v)$ 임을 항상 만족한다.

**증명:** 단계의 횟수에 대한 귀납법

RELAX($u$, $v$, $w$)를 생각해보자. 귀납 가정에 의해 $d[u] \geq \delta(s,u)$ 이고, 삼각 부등식에 의해 $\delta(s,v) \leq \delta(s,u) + \delta(u,v)$ 이다. $d[u] \geq \delta(s,u)$ 이고 $w(u,v) \geq \delta(u,v)$ 이기 때문에 $\delta(s,v) \leq d[u] + w(u,v)$ 가 된다. 그러므로 $d[v] = d[u] + w(u,v)$ 로 할당하는

것은 안전하다.

## DAGs:

순환이 없기 때문에 음의 순환이 있을 수 없다!

1. DAG를 위상 정렬한다. $u$에서 $v$까지의 경로는 선형 순서에서 $u$가 $v$ 전에 온다는 의미이다.
2. 위상 정렬된 정점을 한 번 훑으면서 그 정점에서 나가는 간선에 대해 완화한다.

   $\Theta(V + E)$ 시간이 걸린다.

예:

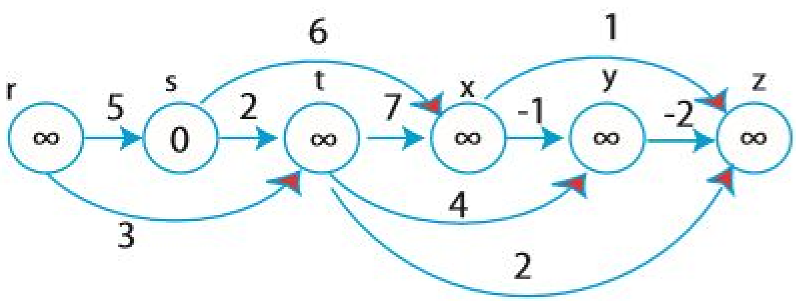



그림 1: 위상 정렬을 이용한 최단 경로 계산

정점은 왼쪽에서 오른쪽으로 위상 정렬됨

$r$의 처리: $\infty$를 유지한다. $s$의 왼쪽에 있는 모든 정점은 정의에 의해 $\infty$로 남을 것이다.

$s$의 처리: $t : \infty \to 2 \quad x : \infty \to 6$ (그림 2의 윗 부분을 보시오)

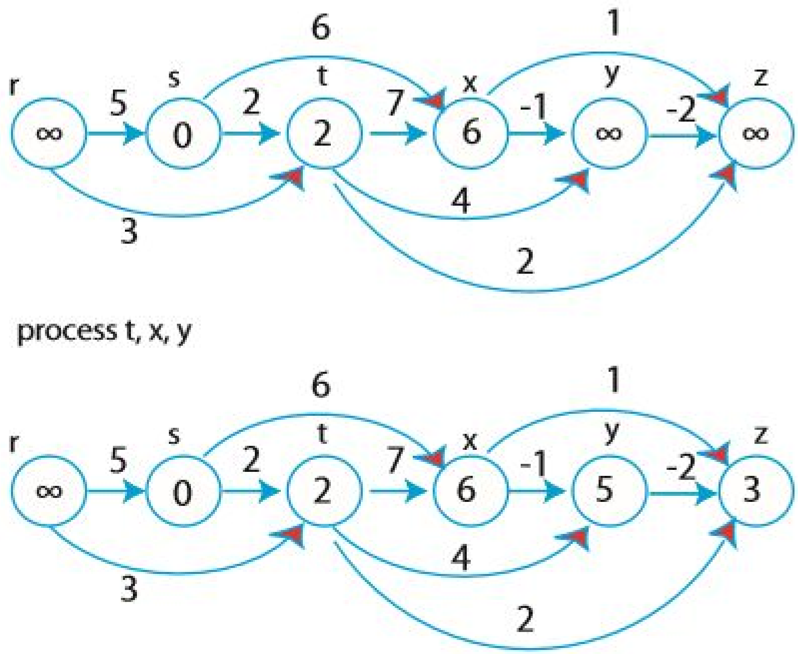



그림 2: 동적 계획법 미리보기

## 다익스트라 시연

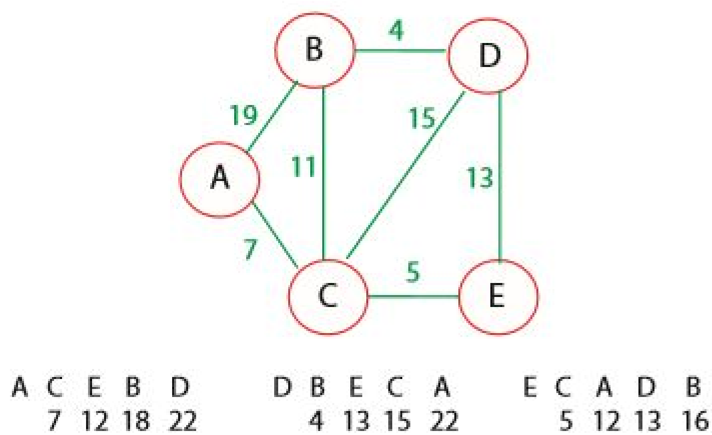



그림 3: 공과 실을 이용한 다익스트라 시연

## 다익스트라 알고리즘

각 간선 $(u,v) \in E$ 에 대하여, $w(u,v) \geq 0$ 을 가정하고, 정점의 집합 $S$ 에 최단 경로의 값이 결정된 정점만을 유지한다. 최소인 최단 경로 어림을 가지는 $u$ 를 $V - S$ 집합에서 반복적으로 선택하여, $u$ 를 $S$ 에 넣고, $u$ 로부터 나가는 모든 간선에 대해 완화한다.

### 의사 코드

Dijkstra $(G, W, s)$ //uses priority queue Q

Initialize $(G, s)$

$S \leftarrow \phi$

$Q \leftarrow V[G]$ //Insert into $Q$

while $Q \neq \phi$

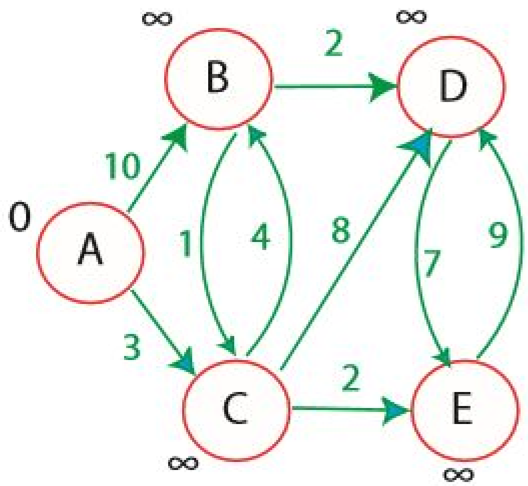
do $u \leftarrow$ EXTRACT-MIN($Q$) //deletes $u$ from $Q$

$S = S \cup \{u\}$

for each vertex $v \epsilon$ Adj[$u$]

do RELAX $(u, v, w)$ $\leftarrow$ this is an implicit DECREASE_KEY operation

## 예




| S={ } | { A | B | C | D | E } | = Q |
|---|---|---|---|---|---|---|
| S={A} | 0 | ∞ | ∞ | ∞ | ∞ | |
| S={A, C} | 0 | 10 | 3 | ∞ | ∞ | ← after relaxing edges from A |
| S={A, C} | 0 | 7 | 3 | 11 | 5 | ← after relaxing edges from C |
| S={A, C, E} | 0 | 7 | 3 | 11 | 5 | |
| S={A, C, E, B} | 0 | 7 | 3 | 9 | 5 | ← after relaxing edges from B |

그림 4: 다익스트라의 실행

전략: 다익스트라는 탐욕 알고리즘이다. $V - S$ 에서 가장 가까운 정점을 선택하여 집합 $S$ 에 넣는다.

타당성: 완화가 안전하다는 것은 안다. 중요 관찰점은 집합 $S$ 에 정점 $u$ 가 추가될 때마다 $d[u] = \delta(s, u)$ 가 된다는 것이다.

## 다익스트라의 복잡도

$\Theta(V)$ 번의 우선순위 큐 삽입 연산
$\Theta(V)$ 번의 EXTRACT-MIN 연산
$\Theta(E)$ 번의 DECREASE-KEY 연산

배열을 이용한 구현:

$\Theta(V)$ : 최소의 원소 추출
$\Theta(1)$ : 키 감소 연산
계: $\Theta(V \cdot V + E \cdot 1) = \Theta(V^2 + E) = \Theta(V^2)$

이진 최소 힙을 이용한 구현:

$\Theta(lgV)$ : 최소의 원소 추출
$\Theta(lgV)$ : 키 감소 연산
계: $\Theta(V\, lgV + ElgV)$

피보나치 힙 (6.006의 범위 아님):

$\Theta(lgV)$ : 최소의 원소 추출
$\Theta(1)$ : 키 감소 연산
분할상환 비용
계: $\Theta(V\, lgV + E)$