

■ Warshall-Algorithmus (Floyd-Warshall)

1 Grundidee

Der **Warshall-Algorithmus** (häufig als **Floyd-Warshall** bezeichnet) berechnet die **kürzesten Wege zwischen allen Knotenpaaren** eines Graphen.

- Dynamic-Programming-Ansatz
 - Betrachtet schrittweise alle Knoten als mögliche Zwischenknoten
 - Funktioniert für **gewichtete Graphen**
 - **Negative Kantengewichte erlaubt**, aber **keine negativen Zyklen**
 - ist **nicht** greedy wie Dijkstra
-

2 Voraussetzungen

- Gewichteter Graph
 - Darstellung als **Adjazenzmatrix**
 - Keine negativen Zyklen
-

3 Laufzeiten & Eigenschaften

Eigenschaft	Wert
Laufzeit	$O(V^3)$
Speicherbedarf	$O(V^2)$
Kürzeste Wege	alle Paare
Negative Kanten	✓ erlaubt
Negative Zyklen	✗ nicht erlaubt

Hinweis: Der Algorithmus ist unabhängig von der Anzahl der Kanten – nur die Anzahl der Knoten zählt.

4 Idee des Algorithmus

Wir prüfen für jedes Knotenpaar (i, j) , ob der Weg über einen Zwischenknoten k kürzer ist:

$$\text{dist}[i][j] = \min(\text{dist}[i][j], \text{dist}[i][k] + \text{dist}[k][j])$$

Dies wird für alle k von 1 bis V durchgeführt.

5 Schritt-für-Schritt-Beispiel

Graph (Adjazenzmatrix):

	A	B	C
A	0	3	∞
B	∞	0	1
C	2	∞	0

Zwischenknoten B

- Prüfe $A \rightarrow B \rightarrow C$: $3 + 1 = 4 < \infty \rightarrow \text{Update}$

Ergebnis:

	A	B	C
A	0	3	4
B	∞	0	1
C	2	∞	0

Warshall-Algorithmus – Algorithmus in Worten (Grundidee)

Der Warshall-Algorithmus ist ein Graphalgorithmus.

Er bestimmt für alle Knotenpaare,
ob ein Pfad zwischen ihnen existiert.

Das Ergebnis ist die transitive Hülle eines Graphen.

Der Algorithmus arbeitet auf einer Adjazenzmatrix.

Dabei werden schrittweise indirekte Verbindungen berücksichtigt.

Warshall-Algorithmus – Algorithmus in Worten (Ablauf)

Der Algorithmus startet mit der Adjazenzmatrix des Graphen.

Diese Matrix beschreibt, welche Knoten direkt miteinander verbunden sind.

Nun wird jeder Knoten der Reihe nach als Zwischenknoten betrachtet.

Für jedes Knotenpaar wird geprüft,

ob ein Weg über diesen Zwischenknoten existiert.

Existiert ein Weg von Knoten A zu Knoten C

und von Knoten C zu Knoten B,

dann existiert auch ein Weg von A zu B.

In diesem Fall wird der Eintrag in der Matrix entsprechend gesetzt.

Dieser Vorgang wird für alle Zwischenknoten wiederholt.

Am Ende zeigt die Matrix,
welche Knotenpaare durch irgendeinen Pfad verbunden sind.

6 Besonderheiten / Prüfungsrelevante Hinweise

- Warshall berechnet **alle kürzesten Wege**
 - Sehr häufige Prüfungsfrage:
 - *Unterschied Dijkstra vs. Warshall?*
 - Negative Kantengewichte erlaubt
 - Negative Zyklen müssen ausgeschlossen sein
-

7 Vor- und Nachteile

Vorteile

- alle kürzesten Wege auf einmal
- einfacher, kompakter Algorithmus
- negative Kanten erlaubt

Nachteile

- hohe Laufzeit $O(V^3)$
 - ungeeignet für große Graphen
-

Merksatz für die Prüfung

Der Warshall-Algorithmus berechnet alle kürzesten Wege mit dynamischer Programmierung und erlaubt negative Kantengewichte ohne negative Zyklen.

8 Python-Implementierung

```
In [1]: def floyd_warshall(dist):
        n = len(dist)

        for k in range(n):
            for i in range(n):
                for j in range(n):
                    if dist[i][k] + dist[k][j] < dist[i][j]:
                        dist[i][j] = dist[i][k] + dist[k][j]

        return dist
```

```
# Beispiel
INF = float("inf")

dist = [
    [0, 3, INF],
    [INF, 0, 1],
    [2, INF, 0]
]

result = floyd_warshall(dist)

for row in result:
    print(row)
```

```
[0, 3, 4]
[3, 0, 1]
[2, 5, 0]
```