


Fritz Schubert
5CS21-2
Dijkstra Algorithmus

Schwarzenberg, 26.03.2023, F.S. 

Ort, Datum, Unterschrift

Inhaltsverzeichnis

1	Aufgabenstellung	3
2	Aufgabenlösung	3
2.1	Beschreibung des Algorithmus	3
2.2	Eigenschaften und Voraussetzungen	3
2.3	Programmcode und Erläuterung	4
2.4	Kürzeste Wege 0 bis xKnoten	5
3	Verzeichnisse	7
3.1	Quellverzeichnis	7

1 Aufgabenstellung

- Beschreiben Sie den Algorithmus Dijkstra
- Diskutieren Sie wesentliche Eigenschaften und Voraussetzungen zur Anwendung
- Schreiben Sie einen Programmcode, welcher den Algorithmus implementiert
- Als Graphen verwenden Sie die Dateien Graph_Dijk2_1.txt und Graph_Dijk2_2.txt. Jede Zeile ist gespeichert in der Form: Knoten1 Knoten2 Gewicht12
- Berechnen Sie die kürzesten Wege vom Startknoten 0 aus; zeigen Sie, dass der Dijkstra Algorithmus mit negativen Gewichten i. allg. keine korrekten Ergebnisse liefert.

2 Aufgabenlösung

2.1 Beschreibung des Algorithmus

Der Dijkstra-Algorithmus ist ein Algorithmus zur Bestimmung des kürzesten Weges in einem gewichteten Graphen. Er wurde 1956 von dem niederländischen Informatiker Edsger W. Dijkstra entwickelt und ist einer der bekanntesten Algorithmen in der Graphentheorie. [1,2]

Der Algorithmus arbeitet iterativ und berechnet für jeden Knoten des Graphen den kürzesten Weg zum Startknoten. Dabei nutzt er eine Prioritätswarteschlange, um die Knoten nach ihrem geschätzten Abstand zum Startknoten zu sortieren und in der Reihenfolge ihres Abstands zu besuchen. [2]

Zu Beginn wird der Startknoten in die Warteschlange eingefügt und sein Abstand zu sich selbst auf 0 gesetzt. Dann wird so lange der Knoten mit dem kleinsten Abstand aus der Warteschlange genommen, bis alle Knoten besucht wurden. Für jeden besuchten Knoten werden die Abstände seiner unbesuchten Nachbarn überprüft und aktualisiert, wenn ein kürzerer Weg gefunden wurde. [3]

Der Algorithmus endet, wenn alle Knoten besucht wurden oder der Zielknoten erreicht wurde. Wenn der Zielknoten erreicht wurde, kann der kürzeste Weg durch Rückverfolgung der Vorgänger-Knoten vom Zielknoten bis zum Startknoten bestimmt werden. [3]

2.2 Eigenschaften und Voraussetzungen

Der Dijkstra-Algorithmus kann nicht mit negativen Kantengewichten umgehen. Für negative Kantengewichte liefert der Algorithmus keine korrekten Ergebnisse und es können Endlosschleifen entstehen (siehe Aufgabe 5). [4]

Eine weitere Voraussetzung ist, dass der Graph zusammenhängend sein muss, da sonst keine kürzesten Wege zwischen allen Knoten existieren. [4]

Der Dijkstra-Algorithmus ist ein optimaler Algorithmus, d.h. er findet immer den kürzesten Weg zwischen zwei Knoten, wenn er korrekt implementiert und für den gegebenen Graphen anwendbar ist. [4]

Im schlimmsten Fall hat der Dijkstra-Algorithmus eine polynomielle Komplexität. Wenn n die Anzahl der Eckpunkte und a die Anzahl der Bögen ist, beträgt diese Komplexität $O(a + n \cdot \log(n))$. [4]

2.3 Programmcode und Erläuterung

- Siehe Datei „main.py“, Kommentare und folgende Erläuterung

Der Programm-Code wurde in Python geschrieben, da sich python's dictionaries und lists sich gut für dieses Problem verwenden lassen. Es wurde der Code-Editor Visual Studio Code benutzt.

Zunächst wird eine der beiden Dateien, je nach Benutzerentscheidung, eingelesen und jede Zeile der Datei wird in Knoten1, Knoten2 und Gewicht aufgeteilt und in ein Set eingefügt. Dabei werden Knoten1 und Knoten2 in die Knotenmenge aufgenommen und wenn sie noch nicht im Graphen-Dictionary enthalten sind, wird eine leere Liste als Wert für den Knoten1-Eintrag im Dictionary hinzugefügt. Wenn Knoten1 bereits im Dictionary vorhanden ist, wird der Eintrag für Knoten2 im Dictionary aktualisiert.

Nachdem alle Zeilen gelesen wurden, wird das Set in eine Liste umgewandelt und sortiert, um datentypbezogene Operationen zu ermöglichen.

Anschließend wird eine Klasse "Graph" definiert, die ein Dictionary "graph" enthält, welches die Kanten des Graphen repräsentiert. Der Graph wird als Adjazenzliste gespeichert, wobei die Schlüssel des äußeren dictionaries die Knoten sind und die dazugehörigen Werte ein weiteres, inneres dictionary. Der Schlüssel des inneren dictionary gibt Knoten an und der dazugehörige Wert, die Kantengewichtung.

Beispiel: `{'1': {'2': 3, '0': 2, '4': 4}}`

Von Knoten 1 führt ein Weg zum Knoten 2 mit der Kantengewichtung 3, zu Knoten 0 ein Weg mit der Gewichtung 2 und zu Knoten 4 ein Weg mit der Gewichtung 4.

Die Funktion "dijkstra" ist der Kern des Algorithmus. Der Algorithmus berechnet den kürzesten Pfad von einem Startknoten zu allen anderen Knoten im Graphen. Es wird eine Liste unbesuchter Knoten initialisiert, die Gewichtung aller Knoten auf einen sehr großen Wert gesetzt und die Gewichtung des Startknotens auf 0 gesetzt.

In einer Schleife wird dann so lange das Minimum der Knoten aus der Liste unbesuchter Knoten gewählt, bis alle Knoten besucht wurden. Für den ausgewählten Knoten werden die Gewichtungen seiner unbesuchten Nachbarn aktualisiert und der beste Pfad zu jedem Knoten wird gespeichert. Der aktuell besuchte Knoten wird aus der Liste unbesuchter Knoten entfernt.

Die Klasse enthält auch Methoden zur Rückgabe der Knoten, der Nachbarn eines Knotens und des Gewichts einer Kante zwischen zwei Knoten. Die Funktion "print_ergebnis" gibt den kürzesten Pfad von einem Startknoten zu einem Zielknoten auf der Konsole aus.

2.4 Kürzeste Wege 0 bis xKnoten

Für die Datei "Graph_Dijk2_1.txt" liefert der Dijkstra-Algorithmus die korrekten Ergebnisse für die kürzesten Wege vom Startknoten 0 aus. Die kürzesten Wege sind wie folgt:

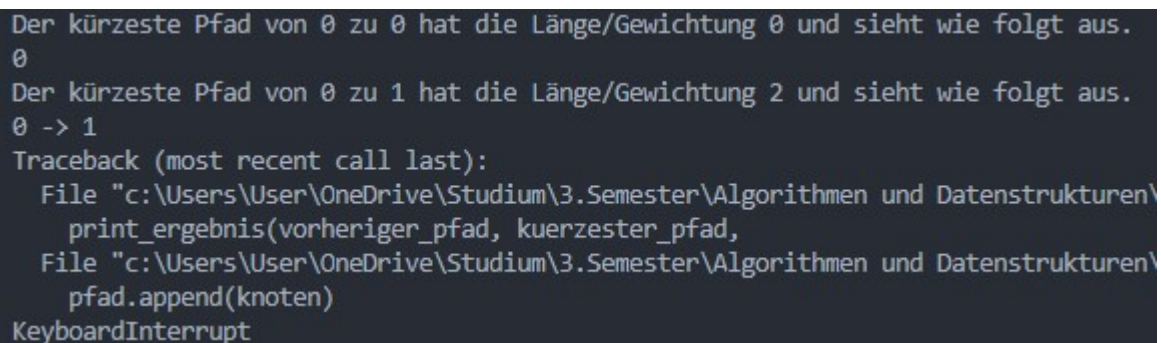
```
Der kürzeste Pfad von 0 zu 0 hat die Länge/Gewichtung 0 und sieht wie folgt aus.  
0  
Der kürzeste Pfad von 0 zu 1 hat die Länge/Gewichtung 2 und sieht wie folgt aus.  
0 -> 1  
Der kürzeste Pfad von 0 zu 2 hat die Länge/Gewichtung 5 und sieht wie folgt aus.  
0 -> 1 -> 2  
Der kürzeste Pfad von 0 zu 3 hat die Länge/Gewichtung 7 und sieht wie folgt aus.  
0 -> 1 -> 2 -> 3  
Der kürzeste Pfad von 0 zu 4 hat die Länge/Gewichtung 6 und sieht wie folgt aus.  
0 -> 1 -> 4  
Der kürzeste Pfad von 0 zu 5 hat die Länge/Gewichtung 7 und sieht wie folgt aus.  
0 -> 1 -> 2 -> 5  
Der kürzeste Pfad von 0 zu 6 hat die Länge/Gewichtung 9 und sieht wie folgt aus.  
0 -> 1 -> 2 -> 3 -> 6  
Der kürzeste Pfad von 0 zu 7 hat die Länge/Gewichtung 12 und sieht wie folgt aus.  
0 -> 1 -> 2 -> 3 -> 6 -> 7  
Der kürzeste Pfad von 0 zu 8 hat die Länge/Gewichtung 7 und sieht wie folgt aus.  
0 -> 1 -> 4 -> 8  
Der kürzeste Pfad von 0 zu 9 hat die Länge/Gewichtung 11 und sieht wie folgt aus.  
0 -> 1 -> 2 -> 3 -> 6 -> 9
```

Abbildung 1: kürzeste Wege von 0 zu allen anderen Knoten

Für die Datei "Graph_Dijk2_2.txt" liefert der Dijkstra-Algorithmus aufgrund der negativen Gewichte keine korrekten Ergebnisse. Der Dijkstra-Algorithmus ist für solche Graphen ungeeignet, da er nur für Graphen mit nicht-negativen Gewichten geeignet ist. In diesem Fall gibt es einen negativen Kreis, was bedeutet, dass ein Kreis im Graphen existiert, dessen Gewichte negativ sind. In einem solchen Fall ist es nicht möglich, den kürzesten Pfad zu finden, da es theoretisch möglich ist, den Pfad durch diesen Kreis unendlich oft zu durchlaufen und die Distanz immer weiter zu reduzieren. Der Dijkstra-Algorithmus kann dieses Problem nicht bewältigen.

Bei der Ausführung erkennt man es daran, dass das Programm in einen unendlichen Loop fällt.

Konsolenausgabe:



```
Der kürzeste Pfad von 0 zu 0 hat die Länge/Gewichtung 0 und sieht wie folgt aus.  
0  
Der kürzeste Pfad von 0 zu 1 hat die Länge/Gewichtung 2 und sieht wie folgt aus.  
0 -> 1  
Traceback (most recent call last):  
  File "c:\Users\User\OneDrive\Studium\3.Semester\Algorithmen und Datenstrukturen\  
    print_ergebnis(vorheriger_pfad, kuerzester_pfad,  
  File "c:\Users\User\OneDrive\Studium\3.Semester\Algorithmen und Datenstrukturen\  
    pfad.append(knoten)  
KeyboardInterrupt
```

Abbildung 2: Abbruch des Programms

Die Anwendung musste durch die Tastenkombination STRG+C gestoppt werden.

3 Verzeichnisse

3.1 Quellverzeichnis

- [1] Anonym, Operations Research 1 DIJKSTRA-ALGORITHMUS, ingenieurkurse.de, (Datum unbekannt), URL: <https://www.ingenieurkurse.de/unternehmensforschung/greedy-algorithmus/dijkstra-algorithmus.html#:~:text=Der%20Dijkstra%2DAlgorithmus%20wurde%20im,der%20Zeitschrift%20Numerische%20Mathematik%20ver%C3%B6ffentlicht>, Letzter Abruf: 06.03.2023
- [2] Edsger Wybe Dijkstra: A note on two problems in connexion with graphs, Numerische Mathematik, Erste Auflage 1959
- [3] Anonym, Der Dijkstra-Algorithmus. Ein Algorithmus der Graphentheorie zur Lösung des Kürzesten-Wege-Problems, GRIN Verlag, 2015 <https://www.grin.com/document/366441>, ISBN: 9783668451711
- [4] Anonym, Dijkstra-Algorithmus: Wie findet man den kürzesten Weg?, datascientest.com, 14.12.2022, URL: <https://datascientest.com/de/was-ist-der-dijkstra-algorithmus>, Letzter Abruf 12.03.2023

Selbstständigkeitserklärung:

Ich versichere, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder veröffentlicht noch einer anderen Prüfungsbehörde vorgelegt.

Schwarzenberg, 26.03.2023, F. Schell

Ort, Datum, Unterschrift