UNIVERSITETI I PRISHTINËS "HASAN PRISHTINA"

FAKULTETI I SHKENCAVE MATEMATIKO-NATYRORE

DEPARTAMENTI I MATEMATIKËS

PROGRAMI: SHKENCA KOMPJUTERIKE

PUNIM SEMINARIK

Lënda: Intelegjencë Artificiale

TEMA: Dijkstra's algorithm

Punuar nga: Auron Ismajli

Florida Kurtaj

Laureta Durguti

Prill 2023

Permbajtja

Hyrja

Algoritmi i Dijkstra-s na lejon të gjejmë shtegun(rrugen) më të shkurtër midis çdo dy kulmesh të një grafi [1].

Algoritmi i Dijkstra-s është një algoritëm popullor për zgjidhjen e shumë problemeve të rrugës më të shkurtër me një burim të vetëm që kanë peshë skajore jo negative në grafikë, d.m.th., është të gjesh distancën më të shkurtër midis dy kulmeve në një grafik. Ai u konceptua nga shkencëtari holandez i kompjuterave *Edsger W. Dijkstra në 1956* [2].

Algoritmi mban një grup kulmesh të vizituara dhe një grup kulmesh të pavizituara. Fillon në kulmin e burimit dhe në mënyrë të përsëritur zgjedh kulmin e pavizituar me distancën më të vogël tentative nga burimi. Më pas viziton fqinjët e këtij kulmi dhe përditëson distancat e tyre tentative nëse gjendet një shteg më i shkurtër. Ky proces vazhdon derisa të arrihet kulmi i destinacionit ose të jenë vizituar të gjitha kulmet e arritshme [2].

Nevoja për algoritmin e Dijkstra lind në shumë aplikacione ku gjetja e shtegut më të shkurtër midis dy pikave është thelbësore.

Aplikimet e Algoritmit të Dijkstra:

- Për të gjetur rrugën më të shkurtër
- Në aplikacionet e rrjeteve sociale
- Në një rrjet telefonik
- Për të gjetur vendndodhjet në hartë [2].

Mënyrat për të zbatuar algoritmin e Dijkstra:

Ka disa mënyra për të zbatuar algoritmin e Dijkstra, por ato më të zakonshmet janë:

- Përdorimi i radhës me përparësi për të mbajtur gjurmët e të gjitha kulmeve.
- Përdorimi i një grupi për të mbajtur gjurmët e Distancave.
- Përdorimi i një grupi për të mbajtur gjurmët e kulmeve të vizituara [2].

Kompleksiteti i Algoritmit të Dijkstra:

Kompleksiteti i kohës: *O (E log V)* ku, E është numri i skajeve dhe V është numri i kulmeve.

Kompleksiteti i hapësirës: *O(V)* [2].

Si funksionon Algoritmi i Dijkstra

Më poshtë janë hapat bazë se si funksionon algoritmi i Dijkstra:

Kërkesat themelore për zbatimin e algoritmit të Dijkstra

Grafiku: Algoritmi i Dijkstra-s mund të zbatohet në çdo grafik, por funksionon më së miri me një grafik të drejtuar të ponderuar me pesha të skajeve jo negative dhe grafiku duhet të përfaqësohet si një grup kulmesh dhe skajesh [2].

Kulmi i burimit: Algoritmi i Dijkstra kërkon një nyje burimi e cila është pika e fillimit për kërkimin dhe më pas analizon gjendjen e grafikut dhe shtigjet e tij për të gjetur distancën më të shkurtër optimale midis nyjes së dhënë dhe të gjitha nyjeve të tjera në grafik [2].

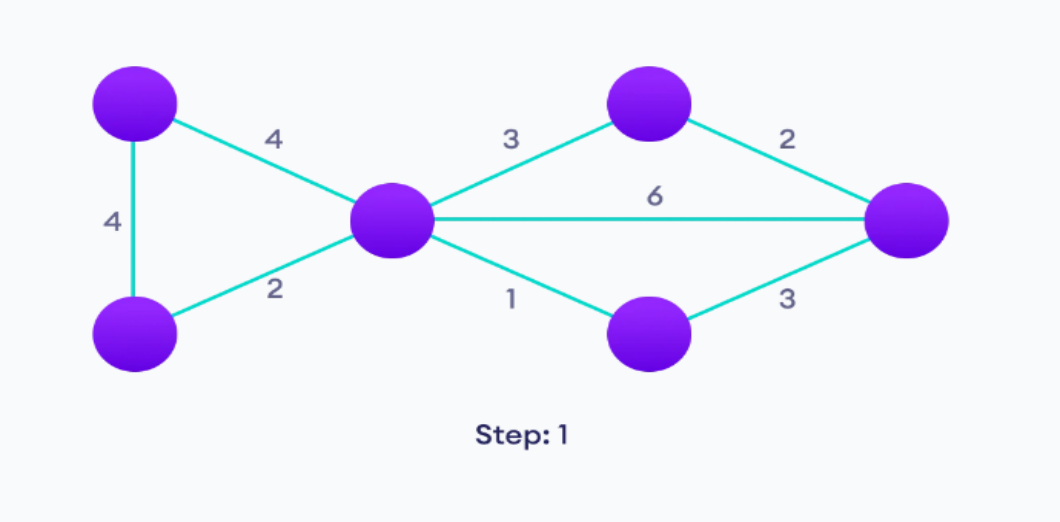
Maja e destinacionit: Algoritmi i Dijkstra mund të modifikohet për të përfunduar kërkimin sapo të arrihet një kulm i caktuar destinacioni [2].

Skajet jo negative: Algoritmi i Dijkstra-s funksionon vetëm në grafikë që kanë pesha pozitive, kjo ndodh sepse gjatë procesit duhet të shtohen peshat e skajit për të gjetur shtegun më të shkurtër. Nëse ka një peshë negative në grafik, atëherë algoritmi nuk do të funksionojë si duhet. Pasi një nyje është shënuar si e vizituar, shtegu aktual drejt asaj nyje shënohet si shtegu më i shkurtër për të arritur atë nyje [2].

Shembull i algoritmit Dijkstra

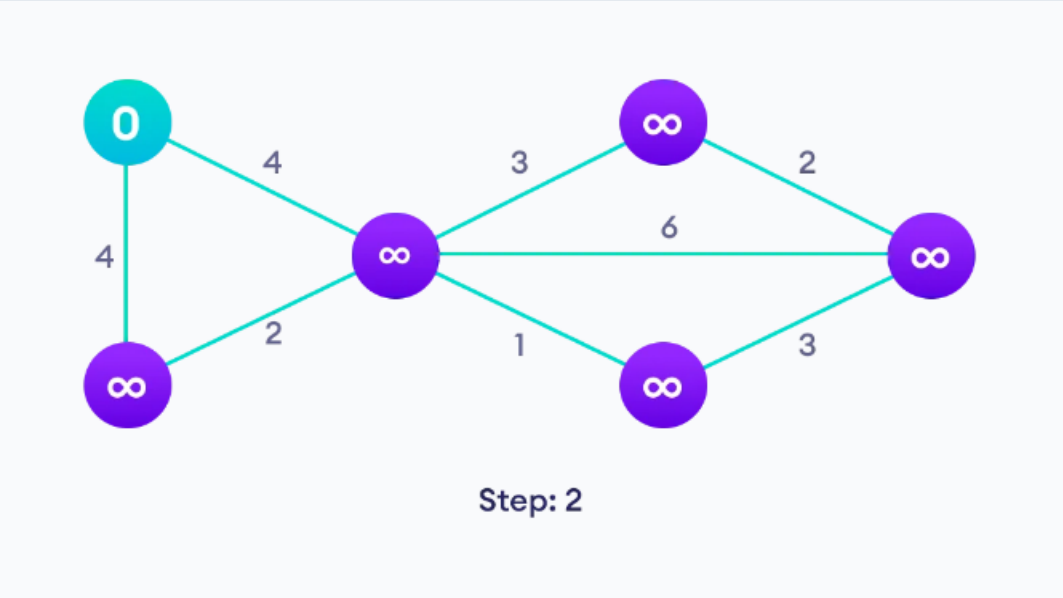
Është më e lehtë të fillosh me një shembull dhe më pas të mendosh për algoritmin [1].

Hapi 1



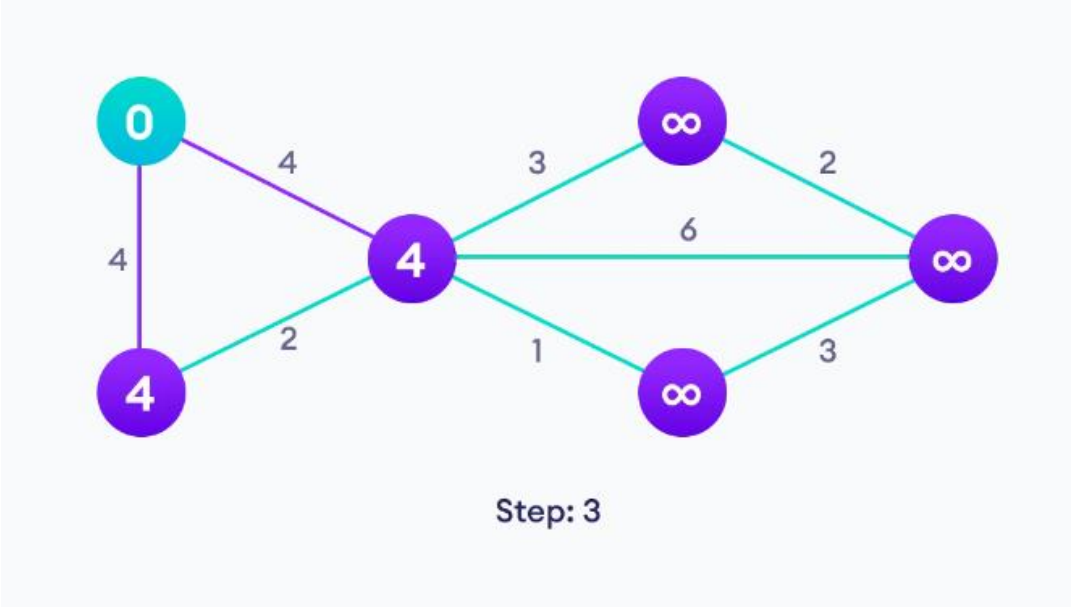
Fillon me nje graf

Hapi 2



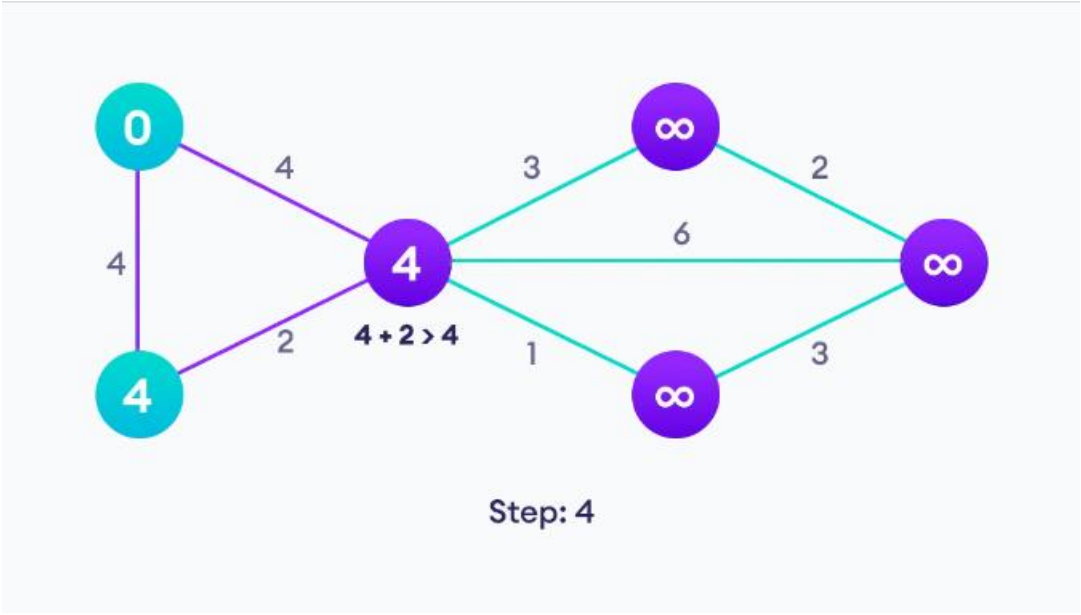
Zgjidhet një kulm fillestar dhe caktohen vlerat e rrugës së pafundësisë për të gjitha pjeset tjera.

Hapi 3



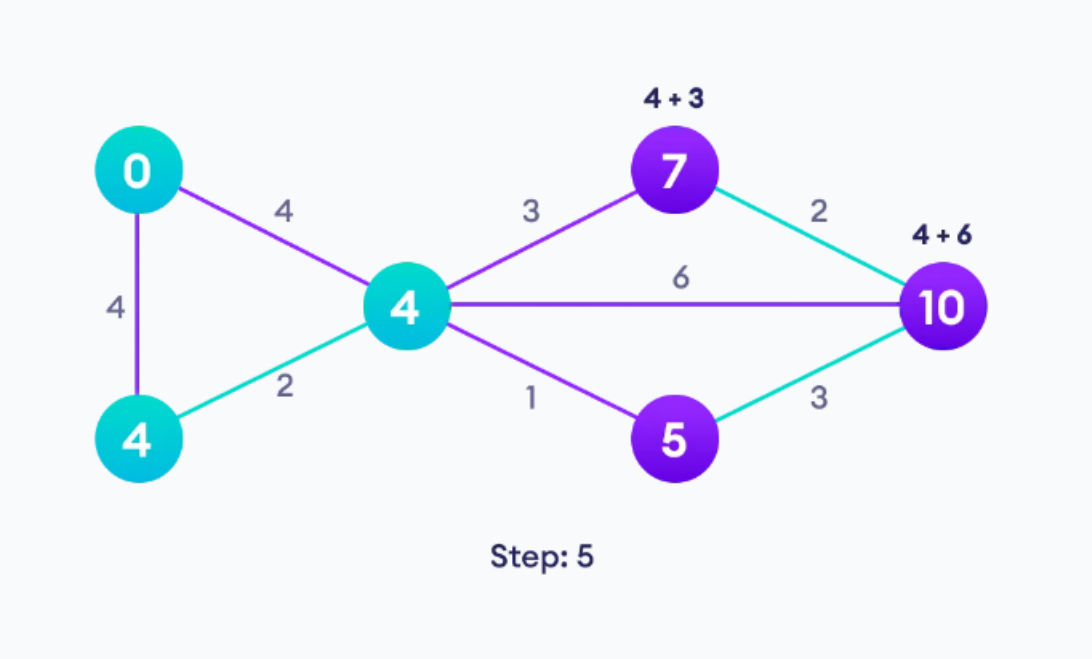
Shkon tek çdo kulm dhe përditëson gjatësinë e shtegut(rruges) të tij.

Hapi 4



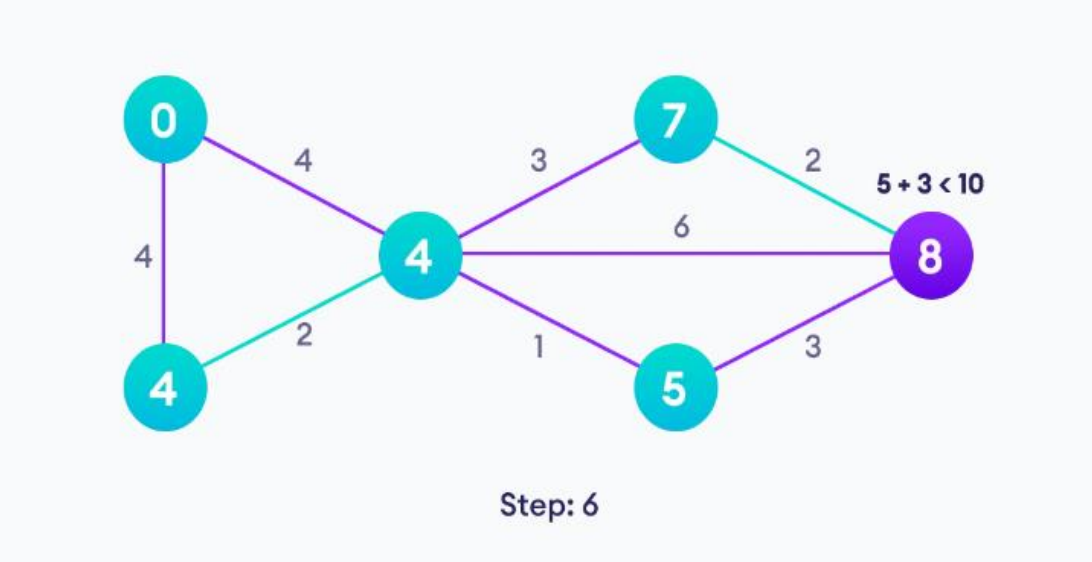
Nëse gjatësia e shtegut të kulmit ngjitur është më e vogël se gjatësia e rrugës së re,nuk e merr parasysh atë.

Hapi 5



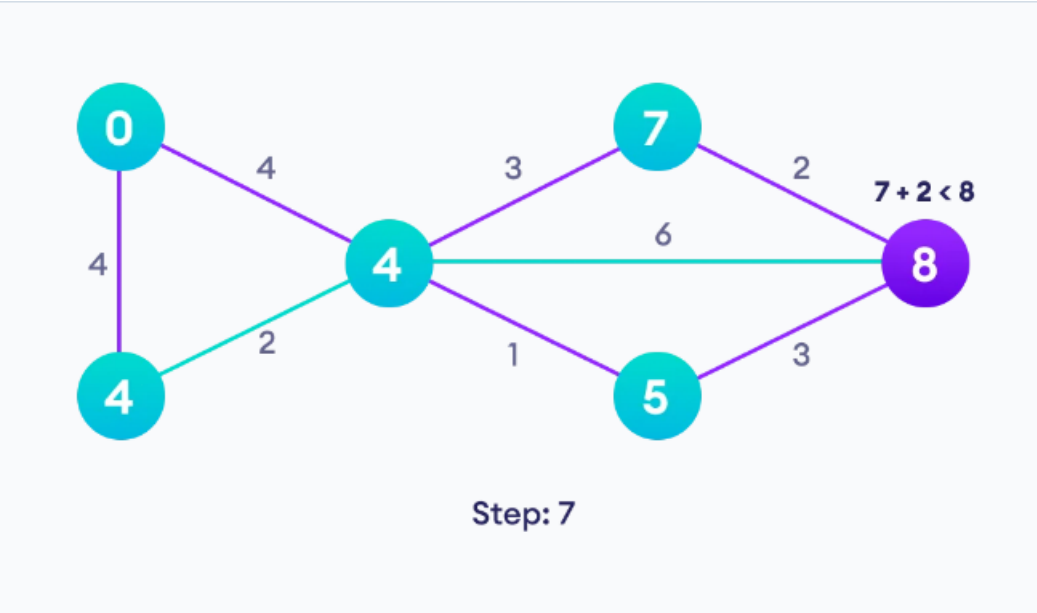
Shmangni përditësimin e gjatësisë së shtigjeve të kulmeve tashmë të vizituara.

Hapi 6



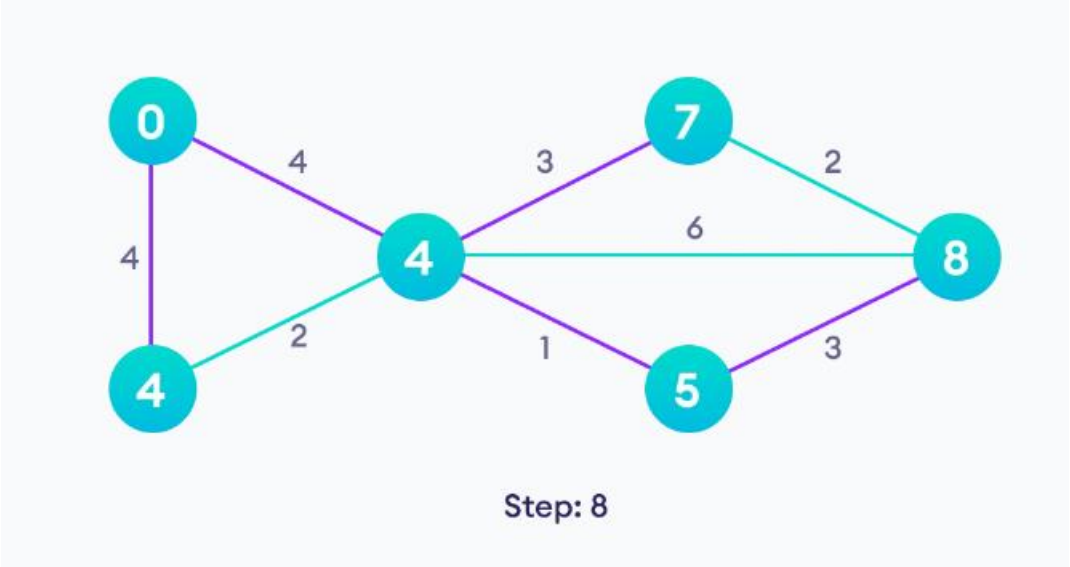
Pas çdo përsëritjeje, zgjedhim kulmin e pavizituar me gjatësinë më të vogël të rrugës. Pra zgjedhim 5 para 7.

Hapi 7



Vini re se si kulmi më i djathtë azhurnohet dy herë gjatësinë e rrugës.

Hapi 8



Përsëriteni derisa të vizitohen të gjitha kulmet.

Kodi ne Java

```
import java.util.*;
public class Djikstra {
private static final int INF = Integer.MAX_VALUE;
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println(&amp;quot;Write the number n of vertices&amp;quot;);
int n = scanner.nextInt();
System.out.println(&amp;quot;Write the number of edges&amp;quot;);
int m = scanner.nextInt();
List&amp;lt;List&amp;lt;Edge&amp;gt;&amp;gt; adj = new ArrayList&amp;lt;&amp;gt;();
for (int i = 0; i &amp;lt; n; i++) {
adj.add(new ArrayList&amp;lt;&amp;gt;());
}
for (int i = 0; i &amp;lt; m; i++) {
System.out.println(&amp;quot;u =&amp;quot;);
int u = scanner.nextInt();
System.out.println(&amp;quot;v = &amp;quot;);
int v = scanner.nextInt();
System.out.println(&amp;quot;w = &amp;quot;);
int w = scanner.nextInt();
adj.get(u).add(new Edge(v, w));
adj.get(v).add(new Edge(u, w));
}
System.out.println(&amp;quot;Give the value of source vertex&amp;quot;);
int src = scanner.nextInt();
int[] dist = dijkstra(adj, src);
for (int i = 0; i &amp;lt; n; i++) {
if (dist[i] == INF) {
System.out.println(&amp;quot;INF&amp;quot;);
```

```
} else {
System.out.print(&quot;The shortest distances from the source vertex to
all other vertices is &quot;
+ dist[i]);
}
}
}
public static int[] dijkstra(List&lt;List&lt;Edge&gt;&gt; adj, int src) {
int n = adj.size();
int[] dist = new int[n];
Arrays.fill(dist, INF);
dist[src] = 0;
PriorityQueue&lt;Node&gt; pq = new PriorityQueue&lt;&gt;((a, b) -&gt; a.dist - b.dist);
pq.offer(new Node(src, 0));
while (!pq.isEmpty()) {
Node node = pq.poll();
int u = node.vertex;
int d = node.dist;
if (d &gt; dist[u]) {
continue;
}
for (Edge edge : adj.get(u)) {
int v = edge.to;

int w = edge.weight;
if (dist[u] + w &lt; dist[v]) {
dist[v] = dist[u] + w;
pq.offer(new Node(v, dist[v]));
}
}
}
```

```
return dist;
}
}
class Edge {
public int to;
public int weight;
public Edge(int to, int weight) {
this.to = to;
this.weight = weight;
}
}
class Node {
public int vertex;
public int dist;
public Node(int vertex, int dist) {
this.vertex = vertex;
this.dist = dist;
```

Referencat

[1] "Programiz-Dijkstra's Algorithm," [Online]. Available: https://www.programiz.com/dsa/dijkstra-algorithm.

[2] "Geeksforgeeks-What is Dijkstra’s Algorithm? | Introduction to Dijkstra’s Shortest Path Algorithm," [Online]. Available: https://www.geeksforgeeks.org/introduction-to-dijkstras-shortest-path-algorithm/?ref=rp.