Steffen Giersch & Maria Lüdemann

Gruppe 12

HAW Hamburg

08.11.2013

Praktikum 2

*Graphentheoretische Konzepte und Algorithmen*

Bei der Aufgabe des zweiten Praktikums handelt es sich um die Implementation zweier Algorithmen zur Findung optimaler Graphen in unserer Graphen Implementation. Bei den implementierten Algorithmen handelt es sich um den Bellmann-Ford und den Floyd-Warshall.

Inhaltsverzeichnis

[1. Aufgabenteilung: 2](#_Toc371791603)

[2. Quellenangaben: 2](#_Toc371791604)

[Begründung: 2](#_Toc371791605)

[3. Bearbeitungszeitraum 2](#_Toc371791606)

[4. Aktueller Stand 3](#_Toc371791607)

[5. Skizze 3](#_Toc371791608)

# Aufgabenteilung:

|  |  |
| --- | --- |
| Student | Aufgabe |
| Steffen Giersch | Entwurf, Implementation, Test |
| Maria Lüdemann | Entwurf, Implementation, Test |

Da wir uns beim Programmieren und Planen immer zusammen setzten haben wir jeden Teil gemeinsam bearbeitet.

# Quellenangaben:

* Floyd-Warshall: Graphentheorie für Studierende der Informatik, Christoph Klauck & Christoph Maas,4. Auflage 2011 diente als Vorlage für den Pseudocode
* Belmann-Ford: Als Vorlage wurde der Pseudocode von Wikipedia genutzt

## Begründung:

Wir übernahmen für diesen Aufgabenteil keinen Fremdcode doch zogen wir sehr anschauliche Algorithmen Beschreibungen zu rate

# Bearbeitungszeitraum

|  |  |  |
| --- | --- | --- |
| Datum | Dauer | Aufgabe |
| 22.10.2013 | 1 Stunde | Planung erste Implementation des Bellmann Ford |
| 23.10.2013 | 1 Stunden | Implementation des Bellman Ford |
| 29.10.2013 | 1 Stunden | Implementation des Bellman Ford |
| 7.11.2013 | 1 Stunden | Planung und erste Implementation des Floyd Warshall |
| 8.11.2013 | 2 Stunden | Implementation des Floyd Warshall |
| 10.11. 2013 | 3 Stunden | Tests |

# Aktueller Stand

* Überwiegend fertig. Ausreichende Tests fehlen noch

# Skizze

#### Floyd Warshall:

floydWarshall(Graph graph){

tranistionsmatrix;

distanzmatrix;

for(i = 0; i < |vertexList|; i +1){

tranistionsmatrix neue Zeile einfügen;

distanzmatrix neue Zeile einfügen;

for(j = 0; j < |vertexList|; j +1){

tranistionsmatrix(i,j) = -1;

if(i == j){

distanzmatrix(i,j) = 0;

}else

distanzmatrix(i,j) = unendlich;

}

for( edgeID : edgeList){

distanzmatrix(source,targed) = gewicht(source, target);

}

for( j = 0; j < |vertexList|; j +1){

for(i = 0; i < |vertexList|; i +1){

if(i != j){

for(k = 0; k < |vertexList|; k +1){

if(dik > dij + djk){

dik = min{dik, dij + djk};

tik := j;

}

if( dii < 0){

Abbruch;

}

}

}

}

}

}

}

#### Bellmann Ford:

bellmanFord(Graph graph, VertexID v) {

for (vertexID vid: vertexList){

distanz(vid) = unendlich;

vorgaenger(vid) = -1;

}

distanz(v) = 0;

for( i = 0; i < |vertexList|; i +1){

for(EdgesID eid: edgeList){

if(distanz(eid) nicht unendlich und distanz(eid) + gewicht(eid,v) < distanz(v)){

distanz(v) =distanz(eid) + gewicht(eid,v);

vorgaenger(v) = eid;

}

}

}

for(EdgesID eid: edgeList){

if(distanz(eid) nicht unendlich und distanz(eid) + gewicht(eid,v) < distanz(v)){

Abbruch;

Ausgabe(„Es gibt einen negativen Kreis“);

}

}

Ausgabe distanz;

}