Steffen Giersch & Maria Lüdemann

Gruppe 12

HAW Hamburg

16.12.2013

Praktikum 4

*Graphentheoretische Konzepte und Algorithmen*

Bei der Aufgabe des dritten Praktikums handelt es sich um die Implementation zweier Algorithmen zum Lösen von Tourenproblemen in unserer Graphen Implementation. Bei den implementierten Algorithmen handelt es sich um den Hierholzer Algorithmus(Eulertour) und die Einführing der dichtesten Ecke (Hamiltonkreis)

Inhaltsverzeichnis

[1. Aufgabenteilung: 2](#_Toc372014013)

[2. Quellenangaben: 2](#_Toc372014014)

[Begründung: 2](#_Toc372014015)

[3. Bearbeitungszeitraum 2](#_Toc372014016)

[4. Aktueller Stand 3](#_Toc372014017)

[5. Skizze 3](#_Toc372014018)

[6. Änderungen 4](#_Toc372014019)

[7. Zugriffe 5](#_Toc372014020)

# 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:

* Hierholzer: Diesen Algorithmus entnahmen wir direkt dem Script
* Die Einführung der dichtesten Ecke: Entnahmen wir ebenfalls direkt dem Script

## Begründung:

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

# Bearbeitungszeitraum

|  |  |  |
| --- | --- | --- |
| Datum | Dauer | Aufgabe |
| 12.12.2013 | 2 Stunden | Planung erste Implementation des Hierholzer |
| 12.12.2013 | 2 Stunden | Erweiterung des Hierholzer |
| 16.12.2013 | 3 Stunden | Implementation des zweiten Algorithmus |

# Aktueller Stand

* Fertig

# Skizze

#### Hierholzer:

**0.** Markiere jede Kante mit benutzt = 0 und erstelle eine initiale leere Kantenfolge

**1a**. Wähle einen beliebigen Knoten vi aus dem Graphen G mit einem Grad > 0,

Gehe zu 2

**1b.** Wähle einen beliebigen Knoten aus der bisherigen Kantenfolge mit unbenutzt-Grad > 0. Wenn keiner gefunden wurde, kann keine Eulertour gefunden werden (nicht zusammenhängend)

**2.** Finde einen Kreis und verwende dafür den Start und Endpunkt vi. Wenn kein Kreis gefunden wurde, kann keine Eulertour gefunden werden.

**3.** Füge den Kreis in die bestehende Kantenfolge ein und markiere jede benutzte Kante mit benutzt = 1

**4.** Wenn jede Kante mit benutzt = 1 markiert wurde, ist eine Eulertour gefunden, wenn nicht gehe zu 1b

#### Die Einführung der kürzesten Ecke:

0. Eine beliebige Ecke vi aus dem Graphen wird gewählt und der bisher gefundende Weg [vi, vi] gesetzt

1. Solange nicht alle Ecken im Weg enthalten sind:

* Die dichteste Ecke vi+1 zum bisherigen Weg wählen
* Über die bisherige Folge iterieren und berechne an welcher Stelle des Kreises vi+1 stehen muss damit es die kürzeste Kantenfolge ergibt.

# Zugriffe

Unsere Algorithmen zählen unterschiedlich der Bellmann Ford zählt jeden seiner Zugriffe auf den Graphen, der Floyd-Warshall handhabt das ein wenig anders da seine Zugriffe auf den Graphen selbst sehr gering sind. Dafür greift er aber sehr häufig auf seine eigene interne Datenstrucktur. Häufig genug damit es sinnvoll ist auch diese zu zählen. Die Zugriffe sehen wie folgt aus: