

# Kürzeste Pfade

*„Dijkstra Algorithmus Parallel“*

Sommersemester 2013

Philip Stewart (526571)  
Julian Vollmer (525904)

Fachbereich 4  
Master Angewandte Informatik  
Distributed Systems and Parallel Processing

**Dozent:**  
Sebastian Bauer

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>3</b>
1.1	Hintergrund . . . . .	3
1.2	Zielstellung . . . . .	3
<b>2</b>	<b>Dijkstra</b>	<b>4</b>
2.1	Wie funktioniert er . . . . .	4
<b>3</b>	<b>Ablaufplan und Eckdaten</b>	<b>5</b>
3.1	Wie läuft sein Performance Test ab . . . . .	5
3.2	Eckdaten . . . . .	5
3.3	Speedup-Berechnung . . . . .	5
<b>4</b>	<b>User Interface Desgin</b>	<b>6</b>
<b>5</b>	<b>User Interface Desgin</b>	<b>8</b>
5.1	Programm . . . . .	8
5.2	Quelltextdokumtation . . . . .	8
5.3	Zwischenstandspräsentation . . . . .	8
5.4	Kontakt . . . . .	8

# 1 Einleitung

## 1.1 Hintergrund

Motivation dieser Arbeit ist der Kurs „M31.2 Distributed Systems and Parallel Processing“ im Sommersemester 2013 welcher vom Dozenten Herr Sebastian Bauer durchgeführt wurde.

## 1.2 Zielstellung

Ziel dieser Arbeit ist es den kürzesten Pfad zwischen 2 Punkten zu bestimmen. Dieses Problem kann effektiv mit dem Dijkstra Algorithmus gelöst werden. Im Rahmen des beiliegenden Projektes wird versucht diesen Algorithmus zu parallelisieren und dadurch einen Speedup<sup>1</sup> zu erreichen.

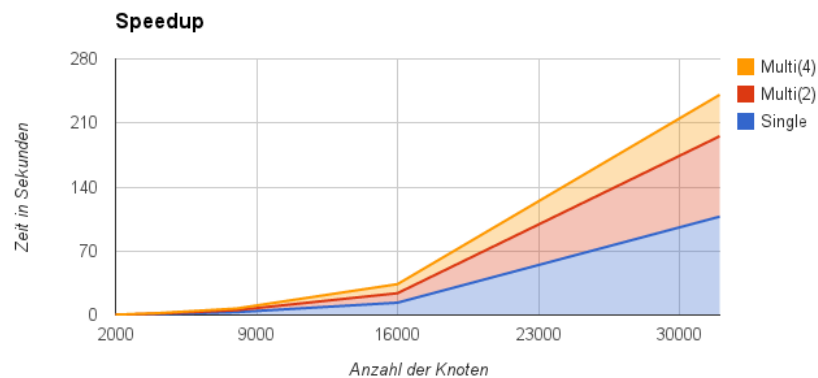


Abbildung 1.1: Speedup OpenMP

---

<sup>1</sup><http://de.wikipedia.org/wiki/Speedup>

## **2 Dijkstra**

### **2.1 Wie funktioniert er**

## 3 Ablaufplan und Eckdaten

### 3.1 Wie läuft sein Performance Test ab

- Knoten werden zufällig erzeugt
- Knotenanzahl wird um 2000 Knoten erhöht

### 3.2 Eckdaten

Es gibt zu einer Wahrscheinlichkeit von 0.1% eine Verbindung zwischen 2 Punkten.

```
void ShortPath::init_random_distances(){
    ...
    if((liefere_ganze_zufallszahl(0,1000)%1000)==0){
        it->add_distances_to_other(0);
    }
    else{
        it->add_distances_to_other(liefere_ganze_zufallszahl(1,250));
    }
    ...
}
```

### 3.3 Speedup-Berechnung

hier die Speedup-Berechnung.

## **4 User Interface Desgin**

#### 4 User Interface Desgin



```
#####  
# Kuerzeste Wegeberechnung mit Multiprocessor #  
#####  
  
(1)Datei einlesen  
(2)Kuerzte Wege (ohne Threads)  
(3)Kuerzte Wege (mit Threads)  
(4)Setze Anzahl der zu berechnenden Graphen  
(5)Anzahl der Threads fuer Multicore  
(6)Schreibe Adjazenzmatrix auf Terminal  
(7)Schreibe Wegberechnung auf Terminal  
(8)Speichere Adjazenzmatrix  
(9)Show Main Test!  
Druecken Sie 0 zum Beeneden  
Wie moechten Sie fortfahren [ 0 - 9 ] : _
```

Abbildung 4.1: User Interface

## **5 User Interface Desgin**

### **5.1 Programm**

### **5.2 Quelltextdokumtation**

### **5.3 Zwischenstandspräsentation**

### **5.4 Kontakt**



# Abbildungsverzeichnis

Abb. 1.1: Speedup OpenMP . . . . .	3
Abb. 4.1: User Interface . . . . .	7