Aufgabe 2 „Dreiecksbeziehungen“-Dokumentation

37. Bundeswettbewerb Informatik 2018/19 - 2. Runde

Lukas Rost

Teilnahme-ID: 48125

29. April 2019

Inhaltsverzeichnis

1 Lösungsidee

1.1 Mathematische Präzisierung der Aufgabenstellung

Bei der Eingabe handelt es sich um eine Menge $D = \{d_1, ..., d_n\}$ von Dreiecken d_i. Jedes Dreieck ist dabei durch seine drei Eckpunkte vollständig definiert ($d_i = \{p_1, p_2, p_3\}$). Ein Eckpunkt ist dabei wiederum ein Punkt $p_i = (x_i, y_i)$ des $\mathbb{R}^2$.

Die Aufgabenstellung fordert nun, dass eine Abbildung $D' = f(D)$ gefunden werden soll. Diese ordnet der Menge D eine Bildmenge D' zu. Für diese müssen bestimmte Bedingungen gelten:

- Für jedes $d \in D'$ gilt:
$$\forall\ (x, y) \in d : y \geq 0 \wedge x \geq 0 \tag{1}$$
Alle Punkte müssen also über oder auf der x-Achse sowie rechts oder auf der y-Achse liegen.

- Für jedes $d \in D'$ gilt:
$$\exists\ (x, y) \in d : y = 0 \tag{2}$$
Es muss also in jedem Dreieck mindestens einen Punkt geben, der auf der x-Achse liegt. Die Menge aller solchen Punkte eines Dreiecks sei N_i (anschaulich die Menge der Straßenecken).

- Für jedes $d \in D'$ und jedes $e \in D'$ gilt:
$$d \cap e = \emptyset \tag{3}$$
$d \cap e$ stellt dabei die Schnittfläche der beiden Dreiecke dar. Es dürfen sich also keine zwei Dreiecke überlappen.

  Eine Dreiecksanordnung wird als **erlaubt** bezeichnet, wenn sie diese Bedingungen erfüllt. Die Menge der erlaubten Dreiecksanordnungen sei dabei E.

Nun ist eine Dreiecksanordnung D' gesucht, die **optimal** ist. Eine optimale Dreiecksanordnung sei dabei folgendermaßen definiert:

- D' minimiert den folgenden Wert über alle erlaubten Dreiecksanordnungen E:
$$\max_{d_i \in D'\ d_j \in D'} \min_{n \in N_i\ m \in N_j} |n.x - m.x| \tag{4}$$
Der Minimums-Term bildet dabei den Abstand zwischen zwei Dreiecken als minimalen Abstand der Straßenecken, während der Maximums-Term den maximalen solchen Abstand berechnet.

Die optimale Dreiecksanordnung D' bildet die Ausgabe des Algorithmus, der $f(D)$ möglichst effizient berechnen soll.

1.2 Intuitive Beschreibung der Lösungsidee

1.3 Mathematische Präzisierung des Algorithmus

1.4 Laufzeitbetrachtung und NP-Vollständigkeit

1.5 Erweiterungen

2 Umsetzung

2.1 Allgemeine Hinweise zur Benutzung

2.2 Struktur des Programms und Implementierung der Algorithmen

3 Beispiele

3.1 Beispiel 1

3.2 Beispiel 2

3.3 Beispiel 3

3.4 Beispiel 4

3.5 Beispiel 5

3.6 Eigene Beispiele

4 Quellcode