

K-Shortest-Path Eppsteins Algorithmus

Darian L. Tratar

7. Dezember 2023

1 Verwendung

- Erfüllen zusätzlicher Bedingungen, die schwer zu optimieren sind
- Lösung für ein NP-Schweres Problem (unterschiedlich markierte Kanten + jede Kante des Weges muss unterschiedlich markiert sein) (siehe Dreyfus und Lawler)
- Evaluieren eines Modells: k als Maß
- Vieles mehr (komplizierter)

2 Laufzeiten

m ist die Anzahl der Kanten und n ist die Anzahl der Knoten

- Für ein Paar von Punkten: $O(m + n \log n + k)$
- Für ein Punkt zu allen andern Punkten: $O(m + n \log n + kn)$
- Aufstellen des Shortest-Path-Trees: $O(n \log n)$
- Gleiche Laufzeit für alle Wege kleiner als eine gegebene Strecke
- Dijkstra implementierung von Fox: $O(m + kn \log n)$
- Mit negativen Kanten ohne negativen Zyklus: $O(mn)$ oder $O(mn^{1/2} \log N)$ mit N als größter Absolutbetrag der negativen Kanten

3 Basics des Algorithmus

- Binary-Heap für jeden Knoten, der nicht zum shortest-Path-Tree gehört und die von diesem erreicht werden können
- Heaps teilen sich gleiche Teile -> Graph mit $O(m + n \log n)$ Knoten
- Verbesserung mit Fredericksons tree decomposition techniques