Contents

2015-05-19 14:43:46

Uebung zum 3. Blatt

1

geg. “Nachfolger” gegen Uhrzeigersinn, Theta-Funktionen, etc wie in Angabe.

- OGDF: C++ bib zum ausprobieren, Enthält Graphstrukturen
- yEd: Grapheditor

Von beliebigem Knoten alle Kreise entlanggehen und füllen - geht nicht wegen Doppelkanten z.B. bei Knoten mit Grad 1

Siehe mehr oder weniger http://i11www.iti.uni-karlsruhe.de/_media/teaching/winter2006/algorithmengineering/triangulierung.pdf

- Für alle $v \in V$
  - Für alle Kanten e indizent zu v
    - $e' = \Theta(e)$
    - trianguliere Facette die (e,e') enthält
    - $e'' = \Theta^*(e')$
    - wenn $\overline{e} = \Theta^*(e'')$ dann
      - fertig
    - wenn $\{v, t(e'')\} \in E$ dann
      - füge $\{s(e''), t(\Theta^*(e''))\}$ ein
      - trianguliere(e,e')
    - sonst

        - füge $e_{neu} = \{v, t(e'')\}$ ein
        - trianguliere(e, e_{neu})

Damit in Linearzeit ist, um den Algorithmus oben außenrum:

- N[] = Array der Größe n (initialisiert mit 0)
- Für alle $v \in V$
  - Für alle Nachbarn u von v (O(deg(v)), |V| mal => O(Anzahl kanten))
    - N[u] = 1
  - Für alle Kanten indizent zu v
    - … siehe oben
  - Für alle Nachbarn u von v
    - N[u] = 0

2

unwichtig

3

3.1

Nein, Beispiel:

- Einfacher Pfad mit n Knoten: $ h ≥n/2 $
- "Nested-Triangles"-Graph: $n = 3j + 1, h \geq j/2$

4

unwichtig

5

(a) BFS O(m)

(Aufgabenstellung falsch, sollte sein "oder entscheidet, dass v nicht auf einem Kreis in G liegt")

(b) BFS von jedem Knoten aus O(n*m)

(c) Planar Separator Theorem verwenden:

- G_1, G_2 mit $n/3 \leq k \leq 2n/3$ Knoten
- S mit $4\sqrt{n}$ Knoten

$$T(n) = T(\alpha_1 n) + T(\alpha_2 n) + O(n\sqrt{n}) \in \Theta(n^k) = \Theta(n\sqrt{n})$$

(Mastertheorem anwenden)