Karlsruher Institut für Technologie

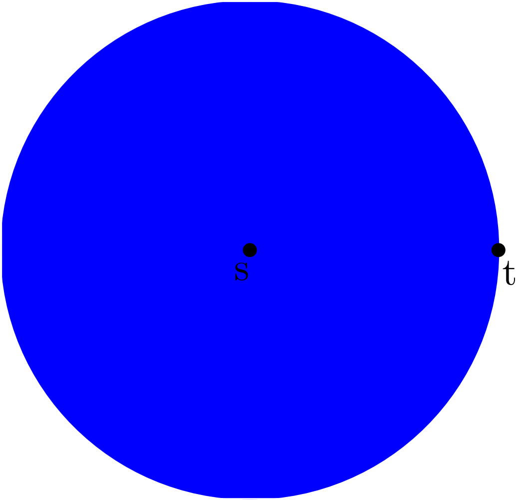
Algorithmen für Routenplanung

3. Sitzung, Sommersemester 2013

Thomas Pajor | 24.04.2013

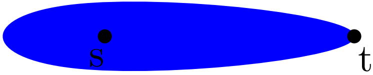
INSTITUT FÜR THEORETISCHE INFORMATIK · ALGORITHMIK · PROF. DR. DOROTHEA WAGNER

Schematischer Suchraum



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Schematischer Suchraum



Ideensammlung

Wie Suche zielgerichtet machen?

- prunen von Kanten, Knoten die in die “falsche” Richtung liegen
- Reihenfolge in der Knoten besucht werden ändern

heute letzteres

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

A*

- auch zielgerichtete Suche (goal-directed search) genannt

Idee:

- Dijkstra's Algorithmus benutzt $d[u]$ um zu entscheiden, welcher Knoten als nächstes abgearbeitet wird
- Benutze Potential $\pi_t : V \to \mathbb{R}$
- $\pi_t[v]$ ist ein Schätzwert für Entfernung $dist(v, t)$ zum Ziel t
- Benutze $d[u] + \pi_t(u)$ als Key in Q

Pseudocode A*

A*$(G = (V, E), s, t)$

$d[s] = 0$

$Q.clear(), Q.add(s, \pi_t(s))$

while $!Q.empty()$ **do**

  $u \leftarrow Q.deleteMin()$

  break if $u = t$

  forall the *edges* $e = (u, v) \in E$ **do**

    if $d[u] + \text{len}(e) < d[v]$ **then**

      $d[v] \leftarrow d[u] + \text{len}(e)$

      if $v \in Q$ **then** $Q.decreaseKey(v, d[v] + \pi_t(v))$

      else $Q.insert(v, d[v] + \pi_t(v))$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Frage

Können beliebige Potentialfunktionen benutzt werden?

- Damit könnten (fast) beliebige Abarbeitungsreihenfolgen erzeugt werden
- Dies kann offensichtlich zu falschen Ergebnissen führen.

Wie kommen wir also zu gültigen Potentialfunktionen?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

A* – Äquivalente Formulierung

Gegeben: Graph $G = (V, E, \text{len})$, Knoten $s, t \in V$, Potential π_t

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

A* – Äquivalente Formulierung

Gegeben: Graph $G = (V, E, \text{len})$, Knoten $s, t \in V$, Potential π_t

Betrachte: Graph $\overline{G} = (V, E, \overline{\text{len}})$ mit

$$\overline{\text{len}}(u, v) = \text{len}(u, v) - \pi_t(u) + \pi_t(v)$$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

A* – Äquivalente Formulierung

Gegeben: Graph $G = (V, E, \mathrm{len})$, Knoten $s, t \in V$, Potential π_t
Betrachte: Graph $\overline{G} = (V, E, \overline{\mathrm{len}})$ mit

$$\overline{\mathrm{len}}(u, v) = \mathrm{len}(u, v) - \pi_t(u) + \pi_t(v)$$

Es gilt für jeden Pfad $P = (s = v_1, \ldots, v_k)$

$$\begin{aligned}
\overline{\mathrm{len}}(P) &= \sum_{i=1}^{k} \overline{\mathrm{len}}(v_i, v_{i+1}) = \sum_{i=1}^{k} (\mathrm{len}(v_i, v_{i+1}) - \pi_t(v_i) + \pi_t(v_{i+1})) \\
&= -\pi_t(v_1) + \pi_t(v_k) + \sum_{i=1}^{k} \mathrm{len}(v_i, v_{i+1}) \\
&= -\pi_t(s) + \pi_t(v_k) + \mathrm{len}(P).
\end{aligned}$$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

A* – Äquivalente Formulierung

Gegeben: Graph $G = (V, E, \text{len})$, Knoten $s, t \in V$, Potential π_t
Betrachte: Graph $\overline{G} = (V, E, \overline{\text{len}})$ mit

$$\overline{\text{len}}(u, v) = \text{len}(u, v) - \pi_t(u) + \pi_t(v)$$

Es gilt für jeden Pfad $P = (s = v_1, \ldots, v_k)$

$$\begin{aligned}
\overline{\text{len}}(P) &= \sum_{i=1}^{k} \overline{\text{len}}(v_i, v_{i+1}) = \sum_{i=1}^{k} (\text{len}(v_i, v_{i+1}) - \pi_t(v_i) + \pi_t(v_{i+1})) \\
&= -\pi_t(v_1) + \pi_t(v_k) + \sum_{i=1}^{k} \text{len}(v_i, v_{i+1}) \\
&= -\pi_t(s) + \pi_t(v_k) + \text{len}(P).
\end{aligned}$$

A* – Äquivalente Formulierung

Gegeben: Graph $G = (V, E, \text{len})$, Knoten $s, t \in V$, Potential π_t
Betrachte: Graph $\overline{G} = (V, E, \overline{\text{len}})$ mit

$$\overline{\text{len}}(u, v) = \text{len}(u, v) - \pi_t(u) + \pi_t(v)$$

Es gilt für jeden Pfad $P = (s = v_1, \ldots, v_k)$

$$\begin{aligned}
\overline{\text{len}}(P) &= \sum_{i=1}^{k} \overline{\text{len}}(v_i, v_{i+1}) = \sum_{i=1}^{k} (\text{len}(v_i, v_{i+1}) - \pi_t(v_i) + \pi_t(v_{i+1})) \\
&= -\pi_t(v_1) + \pi_t(v_k) + \sum_{i=1}^{k} \text{len}(v_i, v_{i+1}) \\
&= -\pi_t(s) + \pi_t(v_k) + \text{len}(P).
\end{aligned}$$

A* auf G ist gleich Dijkstra's Algorithmus auf $\overline{G}$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

A* – Äquivalente Formulierung

Gegeben: Graph $G = (V, E, \text{len})$, Knoten $s, t \in V$, Potential π_t
Betrachte: Graph $\overline{G} = (V, E, \overline{\text{len}})$ mit

$$\overline{\text{len}}(u, v) = \text{len}(u, v) - \pi_t(u) + \pi_t(v)$$

Es gilt für jeden Pfad $P = (s = v_1, \ldots, v_k)$

$$\begin{aligned}
\overline{\text{len}}(P) &= \sum_{i=1}^{k} \overline{\text{len}}(v_i, v_{i+1}) = \sum_{i=1}^{k} (\text{len}(v_i, v_{i+1}) - \pi_t(v_i) + \pi_t(v_{i+1})) \\
&= -\pi_t(v_1) + \pi_t(v_k) + \sum_{i=1}^{k} \text{len}(v_i, v_{i+1}) \\
&= -\pi_t(s) + \pi_t(v_k) + \text{len}(P).
\end{aligned}$$

Wie bekommen wir dann gültige Potentiale?

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Ein Beispiel - Euklidische Ebene

- Knoten sind Punkte in der (euklidischen) Ebene
- Kantenlängen sind euklidische Abstände (d.h. $\text{len}(u, v) = \|u - v\|_2$).
- $\pi_t(v) = \|v - t\|_2$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Ein Beispiel - Euklidische Ebene

- Knoten sind Punkte in der (euklidischen) Ebene
- Kantenlängen sind euklidische Abstände (d.h. $\text{len}(u, v) = \|u - v\|_2$).
- $\pi_t(v) = \|v - t\|_2$

π ist zulässiges Potential, denn

$$\text{len}(u, v) - \pi_t(u) + \pi_t(v) = \|u - v\|_2 - \|u - t\|_2 + \|v - t\|_2 \geq 0$$

wegen Dreiecksungleichung ($\triangle$-UGL)

$$\|u - v\|_2 + \|v - t\|_2 \geq \|u - t\|_2$$

Ein Beispiel - Daten mit geographischer Herkunft

Karlsruher Institut für Technologie

Idee:

- Knoten besitzen Geokoordinaten
- Kanten besitzen Fahrtgeschwindigkeiten
- Kantenmetrik: Fahrtzeit
- es gibt eine Maximalgeschwindigtkeit $v_{\max}$ in G
- nimm $\|u - t\|/v_{\max}$ als Potential

Ist gültiges Potential, denn

$$\text{len}(u, v) - \pi_t(u) + \pi_t(v) = \frac{\|u - v\|_2}{v_{(u,v)}} - \frac{\|u - t\|_2}{v_{\max}} + \frac{\|v - t\|_2}{v_{\max}} \geq 0$$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Ein Beispiel - Daten mit geographischer Herkunft

Idee:

- Knoten besitzen Geokoordinaten
- Kanten besitzen Fahrtgeschwindigkeiten
- Kantenmetrik: Fahrtzeit
- es gibt eine Maximalgeschwindigtkeit $v_{\max}$ in G
- nimm $\|u - t\| / v_{\max}$ als Potential

Probleme (bei Straßengraphen):

- Overhead zur Berechnung des Potentials $\|u - t\| / v_{\max}$ groß
- Abschätzung eher schlecht
- deswegen praktisch keine Beschleunigung in Transportnetzen

Zulässige Potentiale – Eigenschaften

- Intuition: $\pi_t(v)$ ist ein Schätzwert für $\text{dist}(v, t)$
- Falls $\pi_t(t) = 0$, so ist das Potential $p_t(v)$ eine untere Schranke für $\text{dist}(v, t)$.
- **Faustregel:**
  Bessere untere Schranken ergeben kleinere Suchräume
- Ist $\pi_t(v) = \text{dist}(v, t)$ für alle $v \in V$, so werden nur Knoten auf kürzesten s-t-Wegen abgearbeitet.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kombinierbarkeit von Potentialen

Kombinierbarkeit von Potentialen

Seien π_1 und π_2 zulässige Potentiale. Dann ist $p = \max\{\pi_1, \pi_2\}$ (komponentenweises Maximum) auch ein gültiges Potential.

Herleitung

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Kombinierbarkeit von Potentialen

Kombinierbarkeit von Potentialen

Seien π_1 und π_2 zulässige Potentiale. Dann ist $p = \max\{\pi_1, \pi_2\}$ (komponentenweises Maximum) auch ein gültiges Potential.

Herleitung

$$\begin{aligned}
\mathrm{len}(u,v) - \pi_t^1(u) + \pi_t^1(v) &\geq 0 \\
\mathrm{len}(u,v) - \pi_t^2(u) + \pi_t^2(v) &\geq 0
\end{aligned}$$

$$\begin{aligned}
\mathrm{len}(u,v) - \pi_t^1(u) + \max\{\pi_t^1(v), \pi_t^2(v)\} &\geq 0 \\
\mathrm{len}(u,v) - \pi_t^2(u) + \max\{\pi_t^1(v), \pi_t^2(v)\} &\geq 0
\end{aligned}$$

$$\mathrm{len}(u,v) - \max\{\pi_t^1(u), \pi_t^2(u)\} + \max\{\pi_t^1(v), \pi_t^2(v)\} \geq 0$$

Die Dreiecksungleichung für Graphen

Für alle Knoten $s, u, t \in V$ gilt

$$\mathrm{dist}(s, u) + \mathrm{dist}(u, t) \geq \mathrm{dist}(s, t)$$

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Der ALT-Algorithmus

- Der ALT-Algorithmus ist A* mit einer speziellen vorberechneten Potentialfunktion
- ALT steht für A*, landmarks, triangle-inequality

Vorberechnung

- Dazu wird eine kleine Menge L (≈ 16) an Knoten (sog. Landmarken) ausgewählt
- Für jede Landmarke l und jeden Knoten $v \in V$ werden die Distanzen $d(v, l)$ und $d(l, v)$ vorberechnet

Potentiale durch Landmarken

$\triangle$-UGL

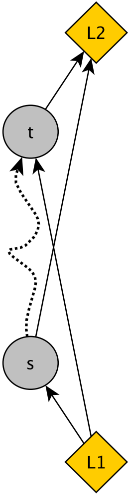
$$\mathrm{dist}(s,u) + \mathrm{dist}(u,t) \geq \mathrm{dist}(s,t)$$

also gilt für alle Knoten $u, t, l \in V$

$$\begin{aligned} d(u,t) &\geq d(l,t) - d(l,u) \\ d(u,t) &\geq d(u,l) - d(t,l) \end{aligned}$$

Benutze die Potentiale

$$\begin{aligned} \pi_t^+(u) &= d(l,t) - d(l,u) \\ \pi_t^-(u) &= d(u,l) - d(t,l) \end{aligned}$$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Potentiale durch Landmarken

Satz

Die Potentiale

$$\pi_t^{l+}(u) = d(l,t) - d(l,u)$$
$$\pi_t^{l-}(u) = d(u,l) - d(t,l)$$

sind zulässig für jede Landmarke $l \in L$.

Beweis: Mit $\triangle$-UGL für Graphen.

Korollar

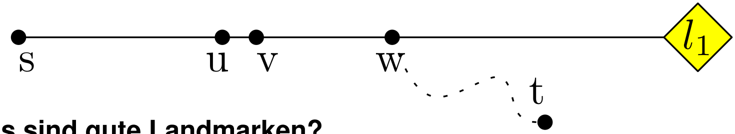
Für eine Menge L von Landmarken ist das Potential

$$\pi_t^L(u) = \max_{l \in L}\{\pi_t^{l+}(u), \pi_t^{l-}(u)\}$$

zulässig.

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Eigenschaften Landmarken



Was sind gute Landmarken?

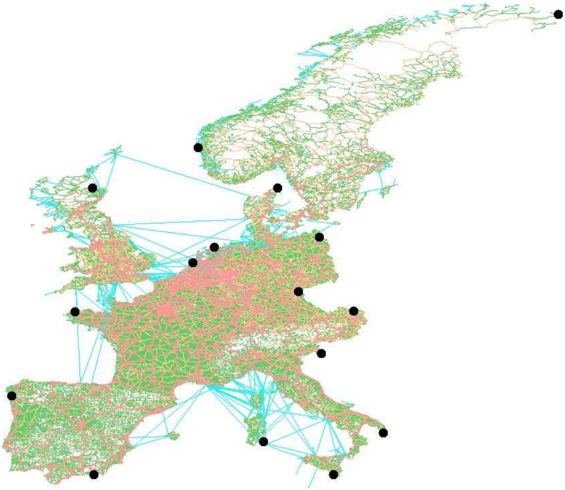
- $\pi_t^{l_1+}(u) = d(u, l_1) - d(t, l_1)$, $\pi_t^{l_1+}(v) = d(v, l_1) - d(t, l_1)$
- also $\text{len}_\pi(u, v) = \text{len}(u, v) - d(u, l_1) + d(v, l_1) = 0$
- gemeinsame Kanten (kürzester Weg und Weg zur Landmarke) haben reduzierte Kosten von 0

Also:

- “gute” Landmarken überdeckten viele Kanten für viele Paare
- trifft unter anderem zu, wenn “hinter” vielen Knoten
- Rand des Graphen

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Beispiel gute Landmarken



Berechnung Landmarken

mehrere Ansätze:

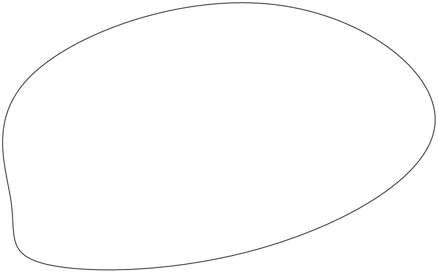
- brute force: $\mathcal{O}(n^{|L|} \cdot \underbrace{n(m + n \log n)}_{\text{all pair shortest path}})$
  - \+ höchste Beschleunigung
  - – zu lange Vorberechnung
- wähle zufällig
  - \+ schnellste Vorberechnung
  - – schlechte Beschleunigung
- mehrere Heuristiken, die versuchen den Rand zu finden
  - planar
  - farthest
  - avoid
  - lokale Optimierung (maxCover)

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Planar-Landmarken

Vorgehen:

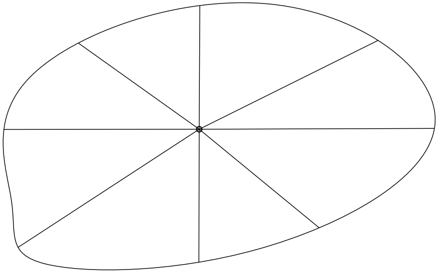
- suche Mittelpunkt *c* des Graphen
- teile Graphen in *k* Teile
- in jedem Teil wähle Knoten mit maximalen Abstand zu *c* als Landmarke



Planar-Landmarken

Vorgehen:

- suche Mittelpunkt *c* des Graphen
- teile Graphen in *k* Teile
- in jedem Teil wähle Knoten mit maximalen Abstand zu *c* als Landmarke

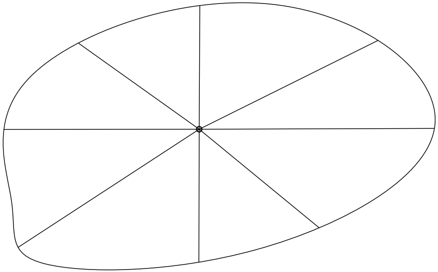


Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Planar-Landmarken

Vorgehen:

- suche Mittelpunkt *c* des Graphen
- teile Graphen in *k* Teile
- in jedem Teil wähle Knoten mit maximalen Abstand zu *c* als Landmarke



Anmerkungen:

- (benötigt planare Einbettung)
- liefert erstaunlich schlechte Ergebnisse

Farthest-Landmarken

Farthest-Landmarks(G,k)

$L \leftarrow \emptyset$
while $|L| < k$ **do**
  if $|L| = 0$ **then** DIJKSTRA(G, RANDOMNODE)
  else DIJKSTRA(G, L)
  $u \leftarrow$ last settled node
  $L \leftarrow L \cup \{u\}$

Anmerkungen:

- Multi-Startknoten Dijkstra
- schlecht für kleine k
- erste Landmarke schlecht
- weitere Landmarken massiv abhängig von erster

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Avoid-Landmarken

Vorgehen:

- berechne kürzeste Wege Baum T_r von einem Knoten r
- $weight(u) = d(r, u) - \underline{d(r, u)}$ (bzgl. L)
- $size(u)$ Summe der Gewichte (weight) seiner Nachfolger in T_r
- $size(u) = 0$ wenn mindestens ein Nachfolger in T_r eine Landmarke ist
- w sei der Knoten mit maximaler Größe (size)
- traversiere T_r startend von w, folge immer dem Knoten mit maximaler Größe
- das erreichte Blatt wird zu L hinzugefügt

Anmerkungen:

- Verfeinerung von Farthest Strategie

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Optimierung von Landmarken

Problem:

- konstruktive Heuristik
- anfangs gewähle Landmarken eventuell suboptimal

Idee:

- lokale Optimierung
- berechne mehr landmarken als nötig ($\approx 4k$)
- wähle beste durch Optimierungsfunktion, z.B.
  - maximiere Anzahl überdeckter Kanten, d.h. $\text{len}(u, v) - \pi(u) + \pi(v) = 0$
  - maximiere $\pi(s)$ für 1 Mio. s-t Paare (simuliert Anfragen)
- avoid + Funktion I wird maxCover genannt

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Literatur

Karlsruher Institut für Technologie

Bidirektionale Suche, A* und ALT:

- Andrew Goldberg, Georg Harrelson, SODA 2005
- Andrew Goldberg, Renato Werneck, ALENEX 2005

Anmerkung:

- wird auf der Homepage verlinkt

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Nächste Termine

Montag, 29.04.2013

Institut für Theoretische Informatik
Lehrstuhl Algorithmik