Parametrische Suche als Algorithmen Paradigma

Verena Miller, Leo Knoll, Daniel Wagner

Universität Salzburg

27. Januar 2017

Inhalt

Algorithmenparadigma

Was ist ein Algorithmenparadigma?

Algorithmenparadigma

Ein Algorithmenparadigma ist ein allgemeiner Ansatz für den Entwurf von Algorithmen.

Beispiele:

- Teile und Herrsche
- Gierige Algorithmen
- Dynamische Programmierung

Parametrische Suche

Was ist Parametrische Suche?

Parametrische Suche

Die Parametrische Suche ist eine Technik, die genutzt wird um einen Entscheidungsalgorithmus in einen Optimierungsalgorithmus umzuwandeln.

Parametrische Suche wird oft in der Algorithmischen Geometrie angewendet.

Definition Optimierungsproblem

Optimierungsproblem

Gegeben:

- Lösungraum Ω
- Bewertungsfunktion $f : \Omega \longrightarrow \mathbb{R}$

Ziel ist es, eine Lösung $x \in \Omega$ mit einem möglichst großen f(x) zu finden oder andere Aussagen über die Werte der Lösung zu erläutern.

Definitionen

Entscheidungsalgorithmus
Hat das Optimierungsproblem eine Lösung mit einer Qualität die besser ist als ein bestimmter Schwellwert?

Optimierungsalgorithmus
Der Optimierungsalgorithmus findet eine optimale Lösung für ein Optimierungsproblem.

Parametrische Suche als Algorithmen Paradigma

Technik

Gegeben: Entscheidungsalgorithmus T mit einem numerischen Input X

- benutzt X nur um ihn mit berechneten Werten zu vergleichen oder das Vorzeichen von Funktionen niedriger Ordnungen mit X zu testen.

Gesucht: Die optimale Lösung X* des Poblems

Technik

Wir brauchen noch einen Entscheidungsalgorithmus E mit Input Y, dieser sagt aus ob

- $Y = X^*$
- $Y < X^*$
- $Y > X^*$

ist.

Diesen können wir aus T erstellen, da T sich bei Eingabe von X^* nicht stetig verhalten wird.

Technik

Gesucht ist X* und wir wissen, dass X*$\in (-\infty, \infty)$.

Wir nehmen an, dass wir X* wissen und simulieren T mit Input X*.

Jedes mal, wenn T einen Vergleich mit X* machen will, simulieren wir den Vergleich mit Hilfe von E.

Dadurch können wir direkt eine Lösung finden oder das Intervall in dem sich X* befindet soweit einschränken, dass die optimale Lösung leicht zu finden ist.

Parametrische Suche als Algorithmen Paradigma

Sequentielle Test Algorithmen

- Test-Algorithmen bzw. Entscheidungsalgorithmen sind sequentiell.

Sequentielle Test Algorithmen

- Test-Algorithmen bzw. Entscheidungsalgorithmen sind sequentiell.
- Schrittweiser Vergleich des unbekannten idealen Parameters X mit neuem Wert Y.
  $\rightarrow$ nicht möglich, daher wird Y als Parameter angenommen und mit dem nächsten Y verglichen.

Sequentielle Test Algorithmen

- Test-Algorithmen bzw. Entscheidungsalgorithmen sind sequentiell.
- Schrittweiser Vergleich des unbekannten idealen Parameters X mit neuem Wert Y.
  $\rightarrow$ nicht möglich, daher wird Y als Parameter angenommen und mit dem nächsten Y verglichen.
- Test-Algorithmus verhält sich bei optimaler Lösung nicht stetig.
  $\rightarrow$ andere Werte (größer/kleiner) können als Annäherung erkannt werden
  $\rightarrow$ optimaler Wert kann für Endoutput erkannt und gespeichert werden

Parametrische Suche als Algorithmen Paradigma

Parallele Test Algorithmen

- Der Prozess der Parametrischen Suche kann mit einem effizienten parallelen Testalgorithmus beschleunigt werden.
- Die Anzahl der Aufrufe des Entscheidungsalgorithmus kann dadurch halbiert werden.

Parallele Test Algorithmen

- Der Prozess der Parametrischen Suche kann mit einem effizienten parallelen Testalgorithmus beschleunigt werden.
- Die Anzahl der Aufrufe des Entscheidungsalgorithmus kann dadurch halbiert werden.
- Dabei wird eine PRAM mit Shared Memory verwendet.

PRAM: Parallel Random Access Machine

PRAM ist eine Registermaschine zur Analyse von parallelen Algorithmen

Parallele Test Algorithmen

- Der Prozess der Parametrischen Suche kann mit einem effizienten parallelen Testalgorithmus beschleunigt werden.
- Die Anzahl der Aufrufe des Entscheidungsalgorithmus kann dadurch halbiert werden.
- Dabei wird eine PRAM mit Shared Memory verwendet.

PRAM: Parallel Random Access Machine

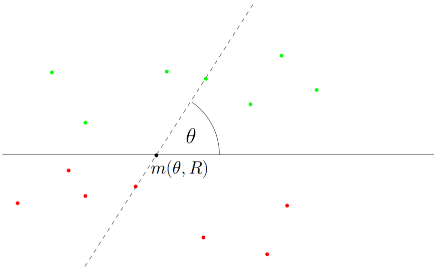
PRAM ist eine Registermaschine zur Analyse von parallelen Algorithmen

Shared Memory

Mehrere Prozesse teilen sich einen gemeinsamen Speicher.

root-finding Problem

- Wir haben 2 Punktmengen, $G = \{g_1, g_2, ..., g_n\}$ und $R = \{r_1, r_2, ..., r_n\}$, wobei gilt: $\forall g_i \in G : y(g_i) < 0$ und $\forall r_i \in R : y(r_i) > 0$.
- Es gilt auch:
  - Keine drei Punkte können auf einer Geraden liegen
  - Keine zwei Punkte besitzen die selbe y-Koordinate
  - n ist ungerade



Quelle

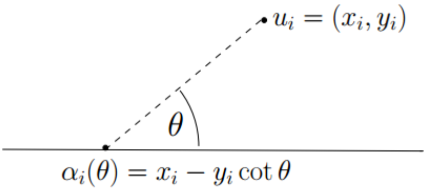
root-finding Problem

- Ziel ist es, den „planar ham sandwich cut" zu finden, der beide Mengen in der Mitte teilt.
- $m(\Theta, G)$ und $m(\Theta, R)$ sind die Schnittpunkte mit der x-Achse durch den Winkel Θ.
- Da n ungerade ist, muss der „planar ham sandwich cut" durch einen Punkt der Menge G und durch einen Punkt der Menge R hindurchgehen.

root-finding Problem

- Ziel ist es, den „planar ham sandwich cut“ zu finden, der beide Mengen in der Mitte teilt.
- $m(\Theta, G)$ und $m(\Theta, R)$ sind die Schnittpunkte mit der x-Achse durch den Winkel Θ.
- Da n ungerade ist, muss der „planar ham sandwich cut“ durch einen Punkt der Menge G und durch einen Punkt der Menge R hindurchgehen.
- X^* ist eine monoton wachsende Funktion F, welche die Bedingung $F(\Theta) = 0$ erfüllt.
- Θ ist der Winkel zwischen der x-Achse und dem „planar ham sandwich cut“.

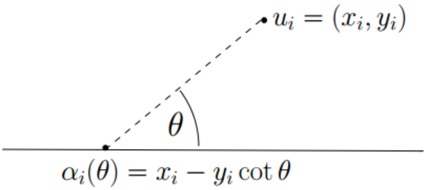
root-finding Problem



Quelle

- Es werden nun alle Punkte mit dem Winkel Θ auf die x-Achse projiziert.

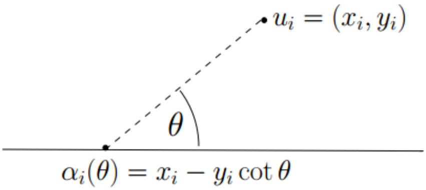
root-finding Problem



Quelle

- Es werden nun alle Punkte mit dem Winkel Θ auf die x-Achse projiziert.
- $\alpha_i = x_i - y_i * tan(\Theta)$, ist ein projizierter Punkt aus den Mengen G und R.

root-finding Problem



Quelle

- Es werden nun alle Punkte mit dem Winkel Θ auf die x-Achse projiziert.
- $\alpha_i = x_i - y_i * tan(\Theta)$, ist ein projizierter Punkt aus den Mengen G und R.
- Der Median entspricht somit $m(\Theta, G)$ und $m(\Theta, R)$.
- Die Entscheidungsfunktion $g_s(\Theta)$ wird mit $g_s(\Theta) = \alpha_i - \alpha_j$ definiert.

root-finding Problem

- Auf der x-Achse gilt $g_s(\Theta) = 0$, da dort die beiden Projektionen auf einen Punkt fallen.

root-finding Problem

- Auf der x-Achse gilt $g_s(\Theta) = 0$, da dort die beiden Projektionen auf einen Punkt fallen.
- Durch Umformungen kommt man zu dem Ergebnis, dass $\Theta_s = arctan(\frac{y_j - y_i}{x_j - x_i})$ ist.

root-finding Problem

- Auf der x-Achse gilt $g_s(\Theta) = 0$, da dort die beiden Projektionen auf einen Punkt fallen.
- Durch Umformungen kommt man zu dem Ergebnis, dass $\Theta_s = arctan(\frac{y_j - y_i}{x_j - x_i})$ ist.
- Die Steigung des „planar ham sandwich cut“ ist Θ^*.

root-finding Problem

- Auf der x-Achse gilt $g_s(\Theta) = 0$, da dort die beiden Projektionen auf einen Punkt fallen.
- Durch Umformungen kommt man zu dem Ergebnis, dass $\Theta_s = arctan(\frac{y_j - y_i}{x_j - x_i})$ ist.
- Die Steigung des „planar ham sandwich cut“ ist Θ^*.
- Nun soll Θ^* berechnet werden, dazu muss $F(\Theta^*) = 0$ ausgewertet werden.
- Da Θ^* noch unbekannt ist, kann $g_s(\Theta^*)$ nicht berechnet werden.

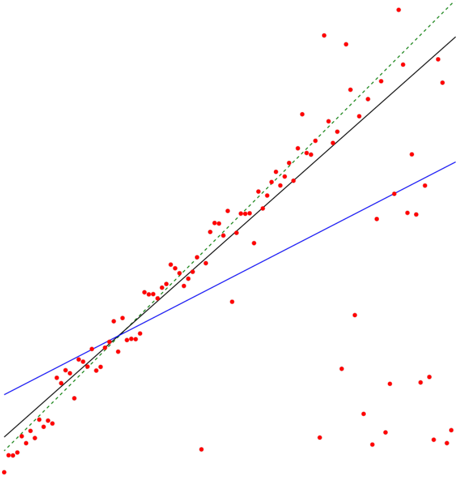
root-finding Problem

- Auf der x-Achse gilt $g_s(\Theta) = 0$, da dort die beiden Projektionen auf einen Punkt fallen.
- Durch Umformungen kommt man zu dem Ergebnis, dass $\Theta_s = arctan(\frac{y_j - y_i}{x_j - x_i})$ ist.
- Die Steigung des „planar ham sandwich cut“ ist Θ^*.
- Nun soll Θ^* berechnet werden, dazu muss $F(\Theta^*) = 0$ ausgewertet werden.
- Da Θ^* noch unbekannt ist, kann $g_s(\Theta^*)$ nicht berechnet werden.
- Dazu wird das Vorzeichen von $F(\Theta_s)$ berechnet, da gilt: $F(\Theta_s) < 0 => \Theta_s < \Theta^* \wedge F(\Theta_s) > 0 => \Theta_s > \Theta^*$.
- Wenn $F(\Theta_s) = 0 => \Theta^* = \Theta_s$ und das Problem wurde gelöst.

Anwendungen

Theil-Sen Estimator

- Anwendung in einfacher linearer Regression: robustes Verfahren zum Anpassen einer Linie an einen Satz von Punkten
- bildet den Median der Neigungen aller Linien durch Punktepaare
- Unempfindlichkeit gegenüber Ausreißern
- genauer als nicht-robuste einfache lineare Regression

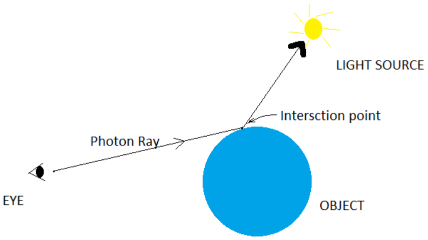


Quelle

Ray Shooting

- Problem in der Computergrafik: Erstes Objekt finden, das von einem Lichtstrahl getroffen wird.

- Durch Parametrische Suche kann herausgefunden werden, ob irgendeine Menge von Hindernissen mit einem Lichtstrahl kreuzt.



Quelle

Quellen

- Cole's Parametric Search Technique Made Practical
- Megiddo, Nimrod (1983), Applying parallel computation algorithms in the design of serial algorithms
- van Oostrum, René; Veltkamp, Remco C. (2002), Parametric search made practical
- Cole, Richard (1987), Slowing down sorting networks to obtain faster sorting algorithms
- Parametrische Suche
- Bernhard Keller: Parametric Search

Vielen Dank für die Aufmerksamkeit