Computational Geometry - Abgabe 4

1st Bartolovic Eduard
Hochschule München
München, Deutschland
eduard.bartolovic0@hm.edu

I. Konvexe Hülle

Für die Berechnung der Konvexe Hülle wurde das Programm *qhull* [1] verwendet. *qhull* verwendet den Quickhull Algorithmus um die Konvexe Hülle zu berechnen. *qhull* lässt sich per Kommandozeile ansprechen. Ein Beispiels für einen Aufruf:

```
rbox 3 D2 | qconvex s o TO result
```

Der Befehl *rbox* generiert eine gewisse Anzahl an Punkten in einer definierten Dimension. Der Befehl *qconvex* berechnet die Konvexe Hülle. Mit einer Pipe können die Punkte direkt *qconvex* übergeben werden.
Es wurden verschiedene Untersuchungen durchgeführt.

++++++++++++ 1 Punkt mehr als Dimensionen nötig +++++++

Laufzeit von *qhull* Es wurde die Laufzeit von *qhull* untersucht. Hierbei wurden verschiedene Konfiguration bezüglich Anzahl und Dimension getestet. Um den Einfluss von Ausreißern zu reduzieren wurde das mittel aus fünf Durchläufen gebildet.
Die Ergebnisse sind in den beiden Abbildungen 1 und 2 zu sehen.

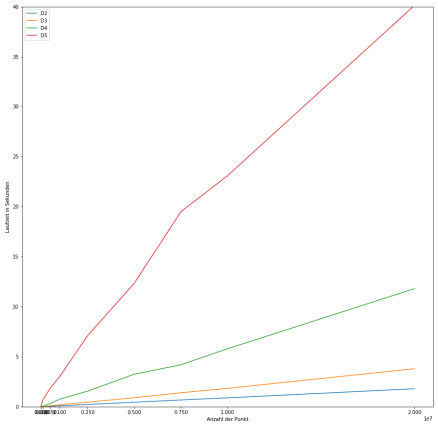

Abbildung 1. Messung der Laufzeit abhängig zur Anzahl der Punkte und Dimensionen

Wie gut in den Messungen zu sehen ist steigt der Aufwand mit Anzahl der Punkte und Dimensionen. Die gemessenen

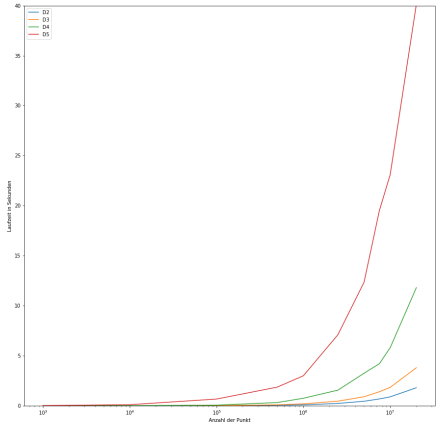

Abbildung 2. Messung der Laufzeit abhängig zur Anzahl der Punkte und Dimensionen (X-Achse ist Log Skalierung)

Laufzeitsteigerungen entsprechen grob der Komplexität $\mathcal{O}(n * \log(n))$???????.
Dies entspricht auch dem Best Case Szenario von Quickhull. Quickhull nutzt einen Divide and Conquer Ansatz um das Problem zu lösen. Im Zwei- und Dreidimensionalen besitzt das Verfahren eine Komplexität von $\mathcal{O}(n * \log(r))$ wobei n die Anzahl der Input Punkten entspricht und r die Anzahl der betrachteten Punkte. Im Zweidimensionalen hat Quickhull im Durchschnitt eine Komplexität bei zufälligen Punkten von $\mathcal{O}(n * \log(n))$.

+++++ N*logN Quickhull mit Worstcase n^2
Höher:
Eine symmetrische Anordnung der Punkte besitzt jedoch eine höhere Wahrscheinlichkeit die Best Case (bester Fall) Laufzeitschranke von $\mathcal{O}(n * \log(n))$ zu verlassen und deutlich langsamer zu sein.

Literatur

[1] http://www.qhull.org/html/qconvex.htm

II. Anhang

Batchfile zum Testen der Laufzeit von Q-Hull:

```
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
```

```
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
rbox 5000000 D2 | qconvex s o TO result
```