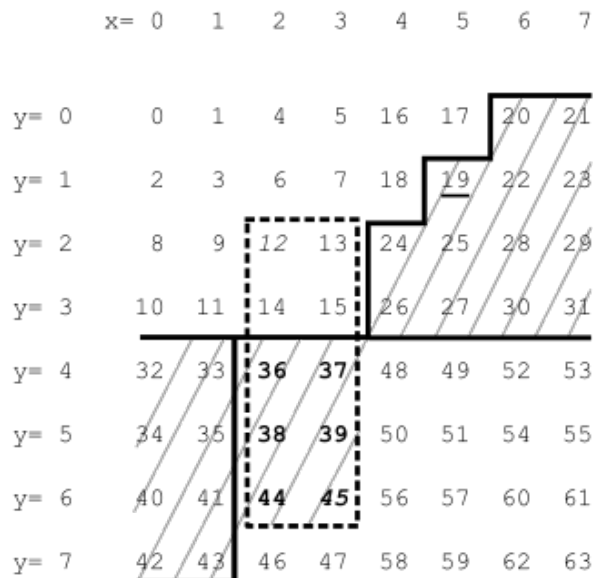


Die folgenden Details findet man auch bei wikipedia „z-kurve“, Thema „Anwendungen in der Informatik“

bzw. im englischen Wikipedia „Z-order curve“ content “Use with one-dimensional data structures for range searching”

Durch Bitverschränken werden die Datenbankeinträge in eine (möglicherweise sehr lange) Folge von Bits umgewandelt. Die Bitfolgen werden als Binärzahlen interpretiert, und die Datenbankeinträge werden nach den Binärwerten sortiert oder indiziert, wobei eine beliebige eindimensionale Datenstruktur verwendet wird. Jedoch ist bei der Abfrage eines mehrdimensionalen Suchbereichs in diesen Daten eine binäre Suche nicht wirklich effizient. Trotz der guten Nachbarschaftserhaltung ist für die mehrdimensionale Bereichssuche ein Algorithmus erforderlich, um, ausgehend von einem in der Datenstruktur außerhalb des Suchbereichs angetroffenen Punkt, den nächstmöglichen Z-Wert zu bestimmen, dessen Koordinaten im Suchbereich liegen.



Im Beispiel der oben stehenden Abbildung ist der Suchbereich ($x=2..3$, $y=2..6$), ein 2D-Intervall, als gestricheltes Rechteck angezeigt. Der höchste Z-Wert darin ist $MAX=45$. Angenommen, im Laufe der Suche wird der Wert $F=19$ angetroffen, bei Suche nach steigenden Werten. Das 1D-Intervall zwischen F und MAX (schraffiertes Gebiet) ist Obermenge des noch zu durchsuchenden Teils des Rechtecks. Um die Suche zu beschleunigen, wird der nächstmögliche Z-Wert im Suchbereich berechnet, im Folgenden **BIGMIN** genannt (36 im Beispiel). Dann muss nur das Intervall zwischen **BIGMIN** und MAX durchsucht werden (fett gezeichnete Werte), dadurch wird der Großteil des schraffierten Gebiets übersprungen. Die Suche nach fallenden Werten ist analog dazu, mit **LITMAX**, dem größten Z-Wert im Suchbereich, der kleiner ist als F (15 im Beispiel).

Das Problem und seine Lösung wurde zuerst im Jahr 1981 von Tropf und Herzog beschrieben (<https://hermanntropf.de/media/multidimensionalrangequery.pdf>). Details findet man in

chapter 4, mit Hinweis auf „Range search“ und „computing Litmax“ und die Tabellen “Litmax decision table” und “Bigmin Decision Table”.

Eine ausführliche Erläuterung des LITMAX/BIGMIN-Berechnungsalgorithmus, zusammen mit Pascal-Quellcode (3D, leicht an nD anzupassen) und Hinweisen zum Umgang mit Fließkommatdaten und möglicherweise negativen Daten, wird bereitgestellt 2021 von Tropf (https://hermanntropf.de/media/DBCode_mit_Erlaeuterung.txt).