Bestimmung des engsten Punktpaares

Wolfgang Mulzer

Gegeben: Eine Menge $P = \{p_1, p_2, \ldots, p_n\}$ von Punkten in der Ebene, so dass die x-Koordinaten p_i`.x` paarweise verschieden sind.

Gesucht: Ein Paar $\{p_i, p_j\}$ mit $i \neq j$, so dass der euklidische Abstand $d(p_i, p_j)$ minimal ist.

In der naiven Lösung probiert man alle $\binom{n}{2}$ Paare von verschiedenen Punkten und wählt das Paar mit minimalem Abstand. Dadurch erhält man eine Laufzeit von $O(n^2)$. Wir werden nun sehen, wie man diese Laufzeit durch Divide and Conquer auf $O(n \log n)$ verringern kann.

Vorverarbeitung: Lege zwei Listen an, `xList` und `yList`. Diese Listen enthalten die Punkte aus P, sortiert nach der x-Koordinate (`xList`) bzw. nach der y-Koordinate (`yList`). Die Vorverarbeitungszeit beträgt $O(n \log n)$. Der Pseudocode für den Algorithmus ist wie folgt:

```
// Eingabe: xList: Punkte nach x-Koordinate sortiert
//          yList: Punkte nach y-Koordinate sortiert
// Ausgabe (d, p, q): d ist der Abstand des engsten Paares {p, q}
CP(xList, yList)
  if xList.size < 4 then
    solve the problem brute force
    return
  m <- median element of xList
  // Aufteilen der Listen
  xListL <- all points r in xList with r.x <= m.x
  xListR <- all points r in xList with r.x > m.x
  yListL <- all points r in yList with r.x <= m.x
            (sorted according to the order of yList)
  yListR <- all points r in yList with r.x > m.x
            (sorted according to the order of yList)
  // Rekursiver Aufruf
  (dL, pL, qL) <- CP(xListL, yListL)
  (dR, pR, qR) <- CP(yListR, yListR)
  (d, p, q) <- (dL < dR) ? (dL, pL, qL) : (dR, pR, qR)
  // Finden eines potentiell engeren Paares
  yList' <- all points r in yList with |r.x - m.x| <= d
            (sorted according to the order in yList)
  for all points r in yList' in order
    determine all distances between r and the 9 succeeding elements in yList',
    take the minimum and change the global minimum (d, p, q) if necessary
  return (d, p, q)
```

Die Zeit für das Aufteilen der Listen und für das Finden eines potentiell engeren Paares ist $O(n)$. Somit erhält man eine Rekursionsgleichung der Form $T(n) = 2T(n/2) + O(n)$, so dass $T(n) = O(n \log n)$ ist.

Durch ein Volumenargument sieht man, dass es genügt, nur die Abstüde zu neun Nachfolgern eines Punktes r in `yList'` zu bestimmen, um ein Paar zu finden, dessen Abstand kleiner als `d` ist (wenn es existiert). Die Laufzeit $O(n \log n)$ ist optimal in einem geeigneten Entscheidungsbaummodell.