Der Algorithmus von Blum, Floyd, Pratt, Rivest, Tarjan

$\textsc{Select}(S, k)$

(1) Teile Elemente aus S in $\lceil \frac{n}{5} \rceil$ Gruppen auf, $\lfloor \frac{n}{5} \rfloor$ davon mit jeweils genau 5 Elementen;

(2) Bestimme den Median jeder Gruppe;

(3) Bestimme rekursiv den Median dieser Mediane, nenne ihn p;

(4) Partitioniere S:

$$S_0 := \{s \in S \mid s \leq p\},$$

$$S_1 := \{s \in S \mid s > p\},$$

(5) **if** $k < |S_0|$ **then**

$\textsc{Select}(S_0, k)$

else if $k = |S_0|$ **then**

return p

else

$\textsc{Select}(S_1, k - |S_0|);$

Untere Schranke für Medianbestimmung

Dor, Zwick 1996: Es gibt ein $\epsilon > 0$, so dass jeder vergleichbasierte Algorithmus zur Bestimmung des Medians einer n-elementigen Menge mindestens $(2 + \epsilon)n - o(n)$ Vergleiche benötigt.

Wir zeigen:

Satz: Jeder vergleichbasierte Algorithmus zur Bestimmung des Medians einer n-elementigen Menge benötigt mindestens $\frac{3}{2}n - 2$ Vergleiche.

Beweisidee: Gegenspieler-Argument

Strategie des Gegenspielers: Partitioniere die Menge S aller Elemente in drei Teilmengen:

T	große Elemente
C	mittlere Elemente: Kandidaten für den Median
B	kleine Elemente

Genauer gilt: $x > y > z \;\; \forall x \in T, y \in C, z \in B$

Initialisierung: $B = T = \emptyset, \quad C = S$

Erste Phase: (solange $|B| < \lceil \frac{n}{2} \rceil - 1, |T| < \lceil \frac{n}{2} \rceil - 1$)

1. Fall: $(x \in B$ und $y \in B)$ oder $(x \in T$ und $y \in T)$

Antwort beliebig, aber konsistent.

2. Fall: $(x \in B$ und $y \in T)$ oder $(x \in T$ und $y \in B)$

Antwort beliebig, aber konsistent.

3. Fall: $(x \in B$ und $y \in C)$ oder $(x \in C$ und $y \in B)$ oder
   $(x \in C$ und $y \in T)$ oder $(x \in T$ und $y \in C)$

Antwort beliebig, aber konsistent.

4. Fall: $(x \in C$ und $y \in C)$

Falls weder x noch y schon mit einem anderen Element in C verglichen wurde, so antworte beliebig. Ansonsten verschiebe x bzw. y nach B bzw. T und antworte entsprechend.

Beobachtung: Durch diese Strategie erreichen wir, dass zu jedem Zeitpunkt gilt:

$$\#\,\text{Vergleiche} \geq 2\,\underbrace{(|B| + |T|)}_{n-|C|} + \#\,\text{Vergleiche innerhalb von } C$$

Zweite Phase:

1. Fall: C enthält keine unverglichenen Elemente

entweder $|C| = 2$ und Anzahl der Vergleiche innerhalb von $C = 1$

$\Longrightarrow$ Anzahl der Vergleiche $\geq 2(n-2)+1 \geq \frac{3}{2}n - 2$ für alle $n \geq 2$

oder $|C| = 1$ $\Longrightarrow$ Anzahl der Vergleiche $\geq 2(n-1)$

2.Fall: B oder T zu groß. O.B.d.A. sei B zu groß, also $|B| = \lceil \frac{n}{2} \rceil - 1$

Idee: Der Gegenspieler zwingt den Algorithmus ab jetzt das kleinste Element in C zu finden. Dazu sind insgesamt (also mit den bereits in der ersten Phase durchgeführten Vergleichen) mindestens $|C| - 1$ Vergleiche nötig. Insgesamt also:

$$\begin{aligned}
\text{Anz. d. Vgl.} &= 2(|B| + |T|) + |C| - 1 \\
&\overset{|B|+|T|=n-|C|}{=} |B| + |T| + n - |C| + |C| - 1 \\
&= \underbrace{|B|}_{=\lceil \frac{n}{2} \rceil - 1} + \underbrace{|T|}_{\geq 0} + n - 1 \geq \frac{3}{2}n - 2
\end{aligned}$$

RandomizedMedian – Ideen

- Wähle zufällig (unabhängig und gleichverteilt) m Elemente von $a[1], \dots, a[n]$ aus. (Manche Elemente werden dabei unter Umständen mehrfach ausgewählt, aber da dies nur selten passiert, wird das nicht weiter stören.)
- Die ausgewählten Elemente werden in ein Array $b[1], \dots, b[m]$ eingetragen und sortiert.
- Wegen der unabhängigen Auswahl der Elemente erwarten wir, dass der Median von $b[.]$ (also das Element $b[\lceil \frac{1}{2}m \rceil]$ nach dem Sortieren) „nahe beim" Median von $a[.]$ liegt.
- Dieses Wissen verwenden wir, um eine kleine Menge von Kandidaten für den Median von $a[.]$ zu erhalten. Dazu betrachten wir zwei Pivotelemente

$$p_\ell = b[\lfloor \tfrac{1}{2}m - \sqrt{n} \rfloor], \quad p_r = b[\lceil \tfrac{1}{2}m + \sqrt{n} \rceil].$$

- Wir werden zeigen, dass die Menge $T := \{x \in S \mid p_\ell \leq x \leq p_r\}$ aller Elemente von $a[.]$, die zwischen p_ℓ und p_r liegen, mit hoher Wahrscheinlichkeit die folgenden Eigenschaften besitzt:
  1. $|T|$ ist "klein".
  2. Der Median von $a[.]$ liegt in T.
- Um den Median zu bestimmen, genügt es daher, die Menge T zu sortieren.

$\textsc{RandomizedMedian}(a, n)$

(1) Wähle m Elemente zufällig aus:

for $i = 1$ **to** m **do** $\quad b[i] := a[\textsc{Random}(1, n)];$

(2) Sortiere die ausgewählten Elemente:

$\textsc{MergeSort}(b, m);$

(3) Bestimme zwei Pivotelemente und partitioniere Elemente:

$p_1 := b[\max\{\frac{1}{2}m - \xi, 1\}];$

$p_2 := b[\min\{\frac{1}{2}m + \xi, m\}];$

$i_0 := 0; i_1 := 0;$

for $j = 1$ **to** n **do**

  if $p_1 \leq a[j] \leq p_2$ **then** $i_1 := i_1 + 1; b[i_1] = a[j]$

  else if $a[j] < p_1$ **then** $i_0 := i_0 + 1;$

(4) Bestimme Median, bzw. iteriere:

if ($i_1 > m'$ oder $i_0 \geq \lfloor \frac{n}{2} \rfloor$ oder $i_0 + i_1 < \lfloor \frac{n}{2} \rfloor$)

  then Gehe zurück zu Schritt (1)

  else $\textsc{MergeSort}(b, i_1)$; **return** $b[\lfloor \frac{n}{2} \rfloor - i_0]$.

Frage: Wie soll man m, ξ und m' wählen???

Analyse von RandomizedMedian

Setze

$$S_k := \text{Menge der } k \text{ kleinsten Elemente von } a[.].$$

und definiere eine Zufallsvariable X_k wie folgt:

$X_k :=$ Anzahl Elemente in $b[.]$, die zur Menge S_k gehören.

Dann gilt:

$$X_k \sim Binomial(m, p) \text{ mit } p = k/n$$

und somit:

$$\mathbb{E}[X_k] = k\frac{m}{n}$$

und

$$\text{Var}[X_k] = mp(1-p) \leq \tfrac{1}{4}m.$$

Mit Hilfe der Chebyshev-Ungleichung erhalten wir

$$\text{Pr}[|X_k - k\frac{m}{n}| \geq \xi] \leq \frac{\text{Var}[X_k]}{\xi^2} \leq \frac{1}{4}\frac{m}{\xi^2}.$$

Wir wählen:

$$m = \lceil n^{3/4} \rceil, \quad \xi = \sqrt{n}$$

Die Chebyshev-Ungleichung besagt dann

$$\text{Pr}[|X_k - kn^{-1/4}| \geq \sqrt{n}] \leq \frac{\text{Var}[X_k]}{n} \leq \tfrac{1}{4}n^{-1/4}. \quad (1)$$

Zum Beweis des Satzes zeigen wir, dass höchstens mit Wahrscheinlichkeit $n^{-1/4}$ eines der folgenden Ereignisse eintritt:

$$\begin{aligned} A &:= \text{„}i_0 \geq n/2\text{“}, \\ B &:= \text{„}i_0 + i_1 < n/2\text{“} \\ C &:= \text{„}i_0 < n/2 - 2n^{3/4}\text{“}, \\ D &:= \text{„}i_0 + i_1 \geq n/2 + 2n^{3/4}\text{“}. \end{aligned}$$

Wenn keines dieser Ereignisse eintritt, so terminiert der Algorithmus in der ersten Iteration. Somit genügt es zu zeigen, dass

$$\text{Pr}[A], \text{Pr}[B], \text{Pr}[C], \text{Pr}[D] \leq \tfrac{1}{4} n^{-1/4}$$

Wenn das Ereignis A eintritt, so enthält $b[.]$ höchstens $\frac{1}{2}n^{3/4} - \sqrt{n}$ Elemente aus $S_{\frac{n}{2}}$. Daraus folgt mit (1):

$$\begin{aligned} \text{Pr}[A] &\leq \text{Pr}[X_{\frac{n}{2}} \leq \frac{1}{2} n^{3/4} - \sqrt{n}] \\ &\leq \text{Pr}[|X_{\frac{n}{2}} - \frac{n}{2} \cdot n^{-1/4}| \geq \sqrt{n}] \\ &\leq \frac{1}{4} n^{-1/4}. \end{aligned}$$

Analog behandelt man die Ereignisse B, C und D.

$\textsc{RandomizedMedian}(a, n)$

(1) Wähle $n^{3/4}$ Elemente zufällig aus:

for $i = 1$ **to** $\lceil n^{3/4} \rceil$ **do** $\quad b[i] := a[\textsc{Random}(1, n)];$

(2) Sortiere die ausgewählten Elemente:

$\textsc{MergeSort}(b, \lceil n^{3/4} \rceil);$

(3) Bestimme zwei Pivotelemente und partitioniere Elemente:

$p_1 := b[\max\{\lfloor \frac{1}{2} n^{3/4} - \sqrt{n} \rfloor, 1\}];$

$p_2 := b[\min\{\lceil \frac{1}{2} n^{3/4} + \sqrt{n} \rceil, \lceil n^{3/4} \rceil\}];$

$i_0 := 0; i_1 := 0;$

for $j = 1$ **to** n **do**

  if $p_1 \leq a[j] \leq p_2$ **then** $i_1 := i_1 + 1; b[i_1] = a[j]$

  else if $a[j] < p_1$ **then** $i_0 := i_0 + 1;$

(4) Bestimme Median, bzw. iteriere:

if ($i_1 > 4n^{3/4}$ oder $i_0 \geq \lfloor \frac{n}{2} \rfloor$ oder $i_0 + i_1 < \lfloor \frac{n}{2} \rfloor$)

  then Gehe zurück zu Schritt (1)

  else $\textsc{MergeSort}(b, i_1)$; **return** $b[\lfloor \frac{n}{2} \rfloor - i_0]$.