Quick Sort Laufzeit Analyse

Satz: Quick Sort arbeitet korrekt. Die Laufzeit ist in $\Theta(n^2)$ für $n = |S|$.

Beweis:

1) Betrachte Rekursionsbaum. Knoten eines Levels repräsentieren disjunkte Teilmenge von S. da S_1, S_2 in Schritt [1] disjunkt sind.

[1]:
```
S1 = quickSort([s for s in A[1:] if s<=x])
S2 = quickSort([s for s in A[1:] if not s<=x])
```

用中文讲，同一层的子问题两两不相交，递归树同一层的所有结点对应的子数组是互不重叠的，把它们的大小加起来就是n.

2) Aufspalten (Divide) einer Teilmenge S' in zwei disjunkte Teilmengen benötigt $|S'|-1$ Vergleiche und Laufzeit $O(|S'|)$

即一次分割的代价是线性的. 把某个子数组 S' 按 pivot 分成两块时，需要把每个元素与 pivot 比较一次，因此比较/时间 $\in O(|S'|)$，更精确是 $|S'|-1$ 次比较.

3) Alle auf einem Level durchgeführten Operationen benötigen Laufzeit $O(n)$

因为，假设这一层有若干个子数组，大小分别是 $k_1, k_2, \dots, k_m$，它们互不重叠，且 $\sum k_i = n$.

那么，这一层做分割的比较次数是

$$\sum_{i=1}^{m} (k_i - 1) = \sum_{i=1}^{m} k_i - m = n - m < n \Rightarrow O(n)$$

所以同一层的代价是 $O(n)$

递归深度的最坏上界是 n.

4) Rekursive Aufrufe erfolgen für echt kleinere Teilmengen, ist Anzahl der Level (Rekursionstiefe) höchstens n. $\Rightarrow$ Laufzeitfunktion in $O(n^2)$

因为每次分割都会让某个子问题的规模严格变小，在最坏情况（pivot总是最小或最大），规模序列是 $n \to n-1 \to n-2 \to \dots \to 1$，层数最多 n

于是总时间 $\leq$ 每层 $O(n) \times$ (层数 $\leq n$) $= O(n^2)$ $\Rightarrow$

5) 如果每次 pivot 都恰好是最小或最大的元素，那么第一次分割要做 $n-1$ 次比较，第二次 $n-2$ 次，…，最后1次。总比较次数：$\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2} \in \Omega(n^2)$

结合上界 $O(n^2)$，得到最坏情况 $\Theta(n^2)$

□

$\boxed{\text{Satz}}$: wird in Quick-Sort als Pivot-Element der Median von S gewäht, so ist die Rekursionstiefe nur $O(\log n)$ für $n = |S|$.

Beweis: Größe der Teilmengen halbiert sich mit jedem Level des Rekursionsbaums. Daher nur $O(\log n)$ Level.

$\boxed{\text{Satz}}$ Wählt man das Pivot-Element zufällig und gleichverteilt aus S, so ist die erwartete Laufzeit von Quick-Sort in $O(n\log n)$ für $n = |S|$

Beweis K8课件中有证明，这里对证明步骤进行补充.

$$f(n) \le c \cdot n + \frac{1}{n}\sum_{i=1}^{n}\left(f(i-1) + f(n-i)\right) = c \cdot n + \frac{2}{n} \cdot \sum_{i=1}^{n-1} f(i)$$

为什么 $\frac{1}{n} \cdot \sum_{i=1}^{n}\left(f(i-1) + f(n-i)\right)$?

因为 pivot 是随机，等概率地从 n 个元素中选出来的.

设当前数组大小为 n

pivot 可能是第 i 小的元素 $(i = 1, 2, \ldots, n)$

每个 i 出现的概率都是 $\frac{1}{n}$ (均匀分布)

如果 pivot = 第 i 小:

- 左边子数组大小 $= i-1$，代价 $= f(i-1)$
- 右边子数组大小 $= n-i$，代价 $= f(n-i)$

$\Rightarrow$ 递归代价 $= f(i-1) + f(n-i)$

$E[X] = \sum$ 事件概率 $\times$ 事件代价

事件 = pivot 选到第 i 小元素

概率 $= \frac{1}{n}$

代价 $= f(i-1) + f(n-i)$

$\Rightarrow$ $E[\text{递归代价}] = \frac{1}{n}\sum_{i=1}^{n}\left(f(i-1) + f(n-i)\right)$

再加上 partition 划分的代价 $c \cdot n$

$$\Rightarrow f(n) \le c \cdot n + \frac{1}{n}\sum_{i=1}^{n}\left(f(i-1) + f(n-i)\right)$$

为什么 $\frac{1}{n}\sum_{i=1}^{n}(f(i-1)+f(n-i)) = \frac{2}{n}\sum_{i=1}^{n-1}f(i)$

因为对称性

$j=i-1$, $\sum_{i=1}^{n}f(i-1) = \sum_{j=0}^{n-1}f(j)$

$k=n-i$, $\sum_{i=1}^{n}f(n-i) = (n-1)+(n-2)+\cdots+0 = \sum_{k=0}^{n-1}f(k)$

$\Rightarrow \sum_{i=1}^{n}(f(i-1)+f(n-i)) = \sum_{j=0}^{n-1}f(j) + \sum_{j=0}^{n-1}f(j) = 2\cdot\sum_{j=0}^{n-1}f(j)$

课件中的 Induktionsschluss 是如何推导出来的？

Z.Z. $f(n) \le 2c\cdot n\cdot(1+\ln n)$, $\forall n\in N_+$ ①

IA. $n=1$, $f(1)=c\le 2c\cdot 1\cdot(1+\ln 1)=2c$

* 之前已证 $f(n)\le c\cdot n + \frac{2}{n}\sum_{i=1}^{n-1}f(i)$ ②

IV. ...

IS. 把 ① 代入 ②

$$f(n)\le c\cdot n + \frac{2}{n}\cdot\sum_{i=1}^{n-1}(2cn\cdot(1+\ln n)) = cn + \frac{4c}{n}\cdot\sum_{i=1}^{n-1}(1+\ln i)\cdot i$$

用积分近似求和.

$$\sum_{i=1}^{n-1} i\cdot(1+\ln i) \simeq \int_1^n x(1+\ln x)\,dx = \frac{n^2}{2}(1+\ln n) - \frac{1}{4}n^2 = \frac{n^2}{2}(1+\ln x)$$

$$\Rightarrow f(n)\le c\cdot n + \frac{4c}{n}\cdot\left[\frac{n^2}{2}(1+\ln x)\right] = \cancel{2cn(1+\ln n)}\ cn + 2cn(1+\ln x)$$

$$\le 2cn(1+\ln x)$$

Integral schon vergessen.