Algorithms for Big Data Date:Sep.18,2017

Lecture L1–Median and Ranking

Prof. Xiaotie Deng *Scribe: Yiren Li*

1 From Big Computer Algorithms to Small Computer Algorithm

1.1 Strategy

Size Reduction + Recursion + Terminate when brute force size is sufficiently small

1.2 Simple idea

Choose a split, divide the set into two part and cost time to work on the part contains median

2 Find k-th Smallest Element in Big Computer

Question:
Ranking sequence A(n,k) (1≤k≤n), find k-th smallest element in A of n elements

2.1 Algorithms

1. **randomly** pick a element of A;
2. use s as the splitter to divide A into B =$A_{smaller}$(s), C =A_{bigger}(s)
3. suppose b=$|B|$≤k
   find C(n-b, k-b)
   else $b > k$,
   find B(b, k)

Emphasizing:RANDAMLY!

2.2 Time

consider expected time:

$$T(n) = (\sum_{b=1}^{k-1} T(n-b) + \sum_{b=k+1}^{n-1} T(b)) \times \frac{1}{n-1} + n$$

solution:

$$T(n) = O(n)$$

2.3 Proof

Decide what is d for the proof to hold:
Assume

$$T(m) \leqslant d \times m$$

For all $T(m) \leq d \times m$

$$T(n) = (\sum_{b=1}^{k-1} T(n-b) + \sum_{b=k+1}^{n-1} T(b)) \times \frac{1}{n-1} \leqslant d \times (\sum_{b=1}^{k-1} T(b)) \times \frac{1}{n-1} + 1 \leqslant d \times n$$

3 Find Median in Small Computer

Questions:
How to find median of A?(no enough space for every elements,one pass algorithm, which means every element goes into cache once)
Let the size of cache be F,how can we find the median?

3.1 Method

Throw away max or min or both with a high probability

- If the new number is larger than max(S) or smaller than min(S) remove it to place in H or L accordingly
- If the new number is in (min(S),max(S)), then keep it and remove max(S) or min(S) to make L or H more balanced

3.2 Algorithms

1. take $|F|$ elements median in randomly (every element in is chosen at **random**)
2. throw away an item(min or max) (median,though not know which is kept with high probability compete it)
3. redo step 2,until every item in F, L or H
4. sort (F), return the median

`BEST situation when the size of F is` $\sqrt{\text{n}}$