O(n^d)

K number of subproblem and with the size of l

Then the number of sub problems are O(n^d)\*k

Eg

T(n)=n^0.5\*T(n^0.5)+n

O(l^d)k

N^(1/2^t)<=1

Take the log

1/(2^t)\*log(n) less than 0

False

Set to 2

N^(1/2^t) <=2

½^t

If we want to find the T(n)=T(n/2)+T(n/3)+n

We just look at the work done at each level and to find the more work done at this leyer.

The next layer is 5n/6

Why we can not bucket sort, because we can only get bit, which takes log(n) to find the total number

Worst case runtime is wort randomness

Expectation worst input normal randomness

Induction on the iteration of index of the iteration.

Recursion based algorithm based on the size of the problem.

IH: pickselect works for length <=n

IS: assume IH(k) want to prove IH(k+1)

Say k >=50

Case one

If len(L)=k-1

We correctly return the kth smallest element

Case two

If len(K)>k-1

Kth smallest element in L is the k th smallest in A

Len(L)<=3len(A)/4<k+1

Thus,pickyselectt（l,k） correctly return the kth smallest element in A

Case three

k-len(L)-1 th smallest

len® ® (r )<k+1

LOOK AT THE THREE DIFFERENCT CASES

I for the k

The

The format

Assume the statement holds at the begin of the loop

Tell what the algorithm does

Do the

Recursion

Do the recursion call

Show that recursion