To do an average case analysis of quickselect one has to consider how likely it is that two elements are compared during the algorithm *for every pair of elements* and assuming a *random pivoting*. From this we can derive the average number of comparisons. Unfortunately the analysis I will show requires some longer calculations but it is a clean average case analysis as opposed to the current answers.

Let's assume the field we want to select the k-th smallest element from is a random permutation of [1,...,n]. The pivot elements we choose during the course of the algorithm can also be seen as a given random permutation. During the algorithm we then always pick the next feasible pivot from this permutation therefore they are chosen uniform at random as every element has the same probability of occurring as the next feasible element in the random permutation.

There is one simple, yet very important, observation: We only compare two elements i and j (with i<j) if and only if one of them is chosen as first pivot element from the range [min(k,i), max(k,j)]. If another element from this range is chosen first then they will never be compared because we continue searching in a sub-field where at least one of the elements i,j is not contained in.

Because of the above observation and the fact that the pivots are chosen uniform at random the probability of a comparison between i and j is:

2/(max(k,j) - min(k,i) + 1)

(Two events out of max(k,j) - min(k,i) + 1 possibilities.)

We split the analysis in three parts:

1. max(k,j) = k, therefore i < j <= k
2. min(k,i) = k, therefore k <= i < j
3. min(k,i) = i and max(k,j) = j, therefore i < k < j

In the third case the less-equal signs are omitted because we already consider those cases in the first two cases.

Now let's get our hands a little dirty on calculations. We just sum up all the probabilities as this gives the expected number of comparisons.

Case 1 (N=k-i replace this at the proof of quick sort at slide)

[https://i.stack.imgur.com/f8WVd.gif](https://i.stack.imgur.com/f8WVd.gif)

[https://i.stack.imgur.com/tXoBr.gif](https://i.stack.imgur.com/tXoBr.gif)

Case 2 (N=j-k replace at the formula at slide of quick sort)

Similar to case 1 so this remains as an exercise. ;)

Case 3 (N=j-i)

https://i.stack.imgur.com/bpASF.gif

https://i.stack.imgur.com/nWWtP.gif

https://i.stack.imgur.com/nV5ld.gif

We use H\_r for the r-th harmonic number which grows approximately like ln(r).

Conclusion

All three cases need a linear number of expected comparisons. This shows that quickselect indeed has an expected runtime in O(n). Note that - as already mentioned - the worst case is in O(n^2).

Note: The idea of this proof is not mine. I think that's roughly the standard average case analysis of quickselect.

If there are any errors please let me know.