

Quicksort

QUICKSORT(A, p, r)

Input. An array $A[1..n]$ of elements from a totally ordered set; p and r are integers with $1 \leq p \leq r \leq n$.

Output. The elements in sub-array $A[p..r]$ are rearranged in sorted order; the elements in array A outside of subarray $A[p..r]$ are unchanged.

if $p < r$ **then**

$q := \text{PARTITION}(A, p, r)$

 QUICKSORT($A, p, q - 1$)

 QUICKSORT($A, q + 1, r$)

fi

To sort the entire array $A[1..n]$ we call QUICKSORT($A, 1, n$).

PARTITION(A, p, r)

Let x be the element in $A[r]$ when PARTITION is called; x is called the *pivot element*.

PARTITION(A, p, r) returns an index q , $p \leq q \leq r$ and rearranges the elements in subarray $A[p..r]$ such that the following properties hold after execution:

- $A[q] := x$
- For $p \leq i < q$, $A[i] \leq x$
- For $q < i \leq r$, $A[i] > x$

It is easy to design an algorithm for PARTITION that runs in time linear in $m = r - p + 1$ (m is the size of the subarray $A[p..r]$). It is also possible to design this algorithm to run *in-place* in linear time, and this is given in the textbook. This is nontrivial, though we will not study this here. Please take a look at it in the textbook, and go through the proof of correctness and its carefully constructed loop invariant.

How about:

- Correctness of QUICKSORT, assuming correctness of PARTITION?
- Worst-case and ‘best-case’ running time of QUICKSORT?

Quicksort Performance

The worst-case running time of QUICKSORT is very large. However, its *expected running time*, assuming every input permutation is equally likely to occur, is pretty good.

Instead of studying the expected running time of QUICKSORT, we will consider a *randomized* version of QUICKSORT, and establish an $O(n \log n)$ bound on its expected running time. For this, we will first review some basic background on discrete probability, and then the notion of a randomized algorithm.

Basic Probability

Discrete probability deals with experiments in which each outcome has a certain likelihood of occurring.

Definition: A *Sample Space* S is a set whose elements are called *elementary events*. (For our purposes, S will always be countable.)

An *event* is a subset of S (hence it is the union of some of the elementary events).

A *probability distribution* Pr is a mapping from events to real numbers satisfying:

1. $Pr(A) \geq 0$ for all $A \subseteq S$.
2. $Pr(S) = 1$.
3. If A and B are two disjoint events, then $Pr(A \cup B) = Pr(A) + Pr(B)$.

Disjoint events are also called *mutually exclusive events*.

A simple example of a probability distribution is the *uniform distribution*, in which every event consisting of a single elementary event has the same probability.

Independent events: Two events A, B are said to be *independent* if $Pr(A \cap B) = Pr(A) \cdot Pr(B)$.

Union bound (sometimes called *Boole's Inequality*): Let $A_i, 1 \leq i \leq n$ be a set of events over a sample space with a probability distribution Pr . Then, $Pr(\cup_{i=1}^n A_i) \leq \sum_{i=1}^n Pr(A_i)$

Discrete Random Variables

Definition: A *discrete random variable* (*r.v.*) X over a sample space S is a mapping from

the sample space to the reals. X inherits the distribution of the sample space. That is, if S has a probability distribution Pr , then $Pr(X = x) = \sum_{s \in S, X(s)=x} Pr(s)$.

The function $f(x) = Pr(X = x)$ is the *probability density function* of the random variable x .

Definition: The *expected value* $E[X]$ of a random variable X is given by:

$$E[X] = \sum_{x \in T} x Pr(X = x) = \sum_{s \in S} X(s) \cdot Pr(s)$$

where T is the *range* of X .

Theorem: (*Linearity of Expectations*) Let X_1, X_2, \dots, X_r be random variables, and let $X = \sum_{i=1}^r X_i$. Then

$$E[X] = \sum_{i=1}^r E[X_i]$$

Indicator random variables. Given an event A , the *indicator random variable* (*irv*) *associated with event* A is the r.v. $I(A)$ which takes value 1 if event A occurs, and value 0 if event A does not occur. Thus,

- $Pr(I(A) = 1) = Pr(A)$,
- $Pr(I(A) = 0) = 1 - Pr(A)$,
- and $E[I(A)] = Pr(A)$.