Probabilistic (or Randomized) Algorithms

Primality: given an n-bit integer x, determine whether x is prime or composite.

(Related to the difficult problem of factoring x.)

1977: Solovay & Strassen showed how to solve primality in time polynomial in n, using a random number generator.

1959: Hoare discovered Quicksort, using a random number generator to sort quickly.

To sort $a_1, a_2, \ldots, a_n$: If $n > 1$,

1. Choose $p \in \{1, 2, \ldots, n\}$ randomly and uniformly.
   I.e., $p \sim \text{Unif}(1, n)$.

2. Let $L = \{a_i \mid a_i < a_p\}$,
   $E = \{a_i \mid a_i = a_p\}$,
   $G = \{a_i \mid a_i > a_p\}$.

3. Recursively sort and output $L$.
   Output $E$.
   Recursively sort and output $G$.

Sorts correctly, by induction on $n$.

Because the running time is a random variable, it makes sense to compute $E(T)$.

If the pivot is always the minimum input, the time is $\Theta(n^2)$.

$E(T) = O(n \log n)$. 