Recurrence relations Lab 6
Lizzie Siegle

Recurrence relation: insertion sort

Base Case: P(1) **=** the time to sort an array containing 1 element == constant:

P(1) = O(1) = 1

P(N) = P(N-1) + N-1

P(N-1) = P(N-2) + N-2

P(N-2) = P(N-3) + N-3

P(2) = P(1) + 1

P(N) + P(N-1) + P(N-2) + P(N-3) + ….+ P(3) + P(2) =

P(N-1) + P(N-2) + P(N-3) + …. +P(3) + P(2) + P(1) + (N-1) + (N-2) + (N-3) +…+3+2+1

P(N) = P(1) + N*(N-1)/2 1 + N*(N - 1)/2

P(N) = 1 + N*(N - 1)/2

Therefore….P(N) = O(1 + N(N - 1)/2) = O(N^2)*

Recurrence relation: quick sort

P(N) = P(N-1) + cN, where N>= 2

P(N-1) - P(N-2) + c(N-1) =

P(N-2) = P(N-3) +c (N-2) =

P(N-3) = P(N-4) + c(N-3)

P(2) = P(1) + c*2

P(N) + P(N-1) + P(N-2) +…+P(2) =

P(N-1) + P(N-2) +…+P(2) + P(1) + c(N) + c(N-1) + c(N-2) =

P(N) = P(1) + c(2 + 3…+N) =

P(N) = 1 + c(N(N+1)/2 -1) ==

P(N) = O(N^2)

Recurrence relation: merge sort

P(N) = 2*P(N/2) + N =

2*(2P(N/4) + N/2) + N =

4P(N/4) + N + N =

4(2P(N/8) + N/4) + N + N =

8*P(N/8) + N + N + N =

N*P(N/N) + N +…+N+N+N =

N +N +…+N+N+N =

P(N)=O(NlgN)