<Discrete Mathematics>

작성자_2018320161_송대선

작성일_06_07

Asymptohic notations

O, Ω, Θ, o, w (Big Oh, Big Omega, Theta, little Oh, little Omega)

->relative grow rates of functions

ex) f(n)=n^2+1

g(n) = 2n+5

->which one grows faster?

$\lim_{n\to\infty}$이걸 쓰면 안다.

5 kinds of notations

the behavior of an algorithm

(running time of an algorithm)

always -> a function of input size이걸 기준 삼는다.

어떤 problem에 대해서 그 problem을 푸는 Alg1(input size=n), Alg2(input size=n)가 있다고 하자.

Alg1의 runtime: n^3+n+1

Alg2의 runtime: $2n^5+1$

$\lim_{n\to\infty}$으로가면 worst case가 나온다.

그래서 Alg1가 더 적게 늘어남으로 이게 더 낫다.

O, Ω, Θ, o, w (Big Oh, Big Omega, Theta, little Oh, little Omega):

where these are used? n>=0으로 가정한다.

(input size가 음수가 될 수는 없다. -> not negative integer)

Definition) Given a function f(n) (=runtime of an algorithm)

Definition) O(f(n)) is the set of functions that grow slower than f(n),

OR grow at the same rate as f(n) as $n\to\infty$

ex) f(n)= n^2+n+1

(1) h(n) = 100n+10 ∈ O(f(n))

(2) g(n) = $2n^2+5$ ∈ O(f(n)) (coefficient does not matter)

$\not\subset$ o(f(n)) little Oh는 같은 것을 포함하지 않는다.

(3) I(n) = $n^3 + 1$ $\not\subset$ O(f(n))

Ω(f(n)): is the set of functions that grow faster than f(n),
OR grow at the same rate as f(n) as $n \to \infty$

Θ(f(n)): is the set of functions that grow at the same rate as f(n) as $n \to \infty$

o(f(n)): is the set of functions that grow slower than f(n) as $n \to \infty$

w(f(n)): is the set of functions that grow faster than f(n) as $n \to \infty$

-insertion sort (input array: A)-
A has n distinct integers
-> insertion is done again again and again (iteratively perform insertion)
-> ascending, descending한 order로 정렬 가능
그럼
(1) What needs to do inserted?
(2) Where to insert?

ex) “A” with 5 numbers
5!의 경우의 수의 input -> 1,2,3,4,5 or 5,4,3,2,1으로 정렬

input: 3, 2, 1, 4, 5

->“3”에만 신경 써보자
3은 window of size 1이다.
하나만 있으면 이미 정렬이 끝났다.

(1) window 2:
3, 2-> 2, 3

(2) window 3: 2, 3, 1 -> 1, 2, 3

(3) window 4: 1, 2, 3, 4

(4) window 5: 1, 2, 3, 4, 5

analysis of insertion sort:
size of input: n
run time of insertion sort?

choose the worst situation
가장 짧게 걸린 것을 선택: Best case analysis
가장 길게 걸린 것을 선택: Worst case analysis

5!=120 possible of inputs
insertion sort의 관점으로 보았을 때
input: 1, 2, 3, 4, 5이면
1. 1, 2 (only one compare is needed)
2. 1, 2, 3 (only one compare is needed)
3. 1, 2, 3, 4 (only one compare is needed)
4. 1, 2, 3, 4, 5 (only one compare is needed)
-> n-1 comparision needed (Best case analysis)

input: 5, 4, 3, 2, 1
1. 5, 4 -> 4, 5 (one compare is needed):1
2. 4, 5, 3 -> (2 compare is needed):3
3. (3 compare is needed):6
4. (4 compare is needed):10
-> 10 compare is needed
-> $\frac{n(n-1)}{2}$ compare is needed
일반적으로 insertion sort의 runtime function은 $\frac{n(n-1)}{2}$=O(n^2)이라고 한다.
even with a same algorithm and same input size,
algorithm can work more or less.

<ch_8>
a partial order
an equivalence relation
-> these two are special kinds of a binary relation

Definition) an equivalence relation:
a binary relation R is called an equivalence relation
if
(1) R is reflexive
(2) R is symmetric
(3) R is transitive

ex) A={1, 2, 3, 4}
a subset of AxA
n(AxA)=16 -> R has 2^{16} possible cases