차수

주요 내용

- 차수의 직관적 이해
- O, Ω, Θ 정의

시간 복잡도 함수의 차수 이해

궁극적으로 빠르다? 느리다?

아래 두 알고리즘 중에서 어떤 알고리즘을 선택해야 할까?

알고리즘	일정 시간 복잡도
A	$100n$
B	$0.01n^2$

입력 크기에 따른 선택

- $n \leq 10,000$ 일 때: 알고리즘 B 선택
- $n > 10,000$ 일 때: 알고리즘 A 선택

이유:

$$\begin{aligned} 0.01n^2 > 100n \quad &\Longleftrightarrow \quad n^2 > 10000n \\ &\Longleftrightarrow \quad n > 10000 \end{aligned}$$

알고리즘 A와 B의 시간 복잡도 비교

알고리즘	일정 시간 복잡도
A	$100n$
B	$0.01n^2$

- 알고리즘 A가 알고리즘 B 보다 궁극적으로 빠르다
- 알고리즘 B가 알고리즘 A 보다 궁극적으로 느리다

시간 복잡도 함수의 차수

- 차수: "궁극적으로 느리다/빠르다"를 판단하는 기준

- "알고리즘 A의 시간 복잡도 함수의 차수가 $\Theta(f(n))$ 이다":

> *입력 크기가 n인 임의의 입력값에 대해 알고리즘 A는* ***항상 $f(n)$ 시간 정도 보다 빠르지도 느리지도 않게*** *실행을 멈춘다.*

차수 예제

- $\Theta(1)$ (상수 복잡도)

  $1,\ 17,\ 1000000,\ \cdots$

- $\Theta(n)$ (1차 시간 복잡도)

  $100n,\ 0.001n + 100,\ \cdots$

- $\Theta(n^2)$ (2차 시간 복잡도)

  $5n^2,\ 0.1n^2 + n + 100,\ \cdots$

- $\Theta(n^3)$ (3차 시간 복잡도)

  $7n^3,\ n^3 + 5n^2 + 100n + 2,\ \cdots$

- $\Theta(\log n)$ (로그 복잡도)

  $$\log n,\ \ 2\log n,\ \ \frac{1}{2}\cdot\log n+3,\ \ \cdots$$

- $\Theta(n\ \log n)$ (엔-로그-엔, $n\log n$) 복잡도

  $$n\ \log n,\ \ 2n\ \log n,\ \ \frac{1}{2}n\ \log n+\log n+3,\ \ \cdots$$

효율성	알고리즘 복잡도
매우 효율적인 알고리즘	$\Theta(1)$, $\Theta(\log n)$, $\Theta(n)$, $\Theta(n \log n)$
경우에 따라 괜찮은 알고리즘	$\Theta(n^2)$, $\Theta(n^3)$
사실상 사용 불가 알고리즘	$\Theta(2^n)$, $\Theta(n!)$

차수의 정의

차수(Θ)를 엄밀하게 정의하려면 Big-O와 Ω(Omega, 오메가) 개념이 요구된다.

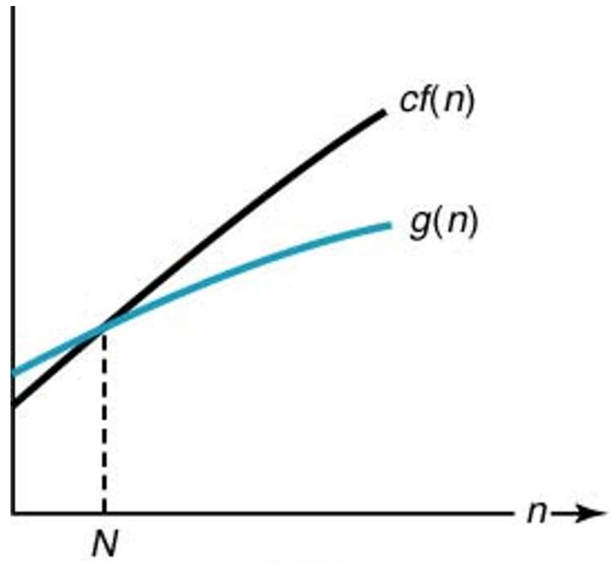
Big-O 표기법

다음 성질을 갖는 양의 실수 c 와 양의 정수 N 이 존재할 때 $g(n) \in O(f(n))$ 으로 표현한다.

$$n \geq N \text{인 임의의 정수 } n \text{에 대해 } g(n) \leq c \cdot f(n)$$

Big-O 알고리즘의 시간 복잡도 측면

- 입력 크기 n에 대해 시간 복잡도 $g(n)$의 실행시간이 궁극적으로 $f(n)$보다 나쁘지는 않음을 의미한다.
- $g(n)$의 실행시간이 최악(worst)의 경우에도 $f(n)$의 실행시간보다는 느리지 않다.



(a) $g(n) \in O(f(n))$

예제 1

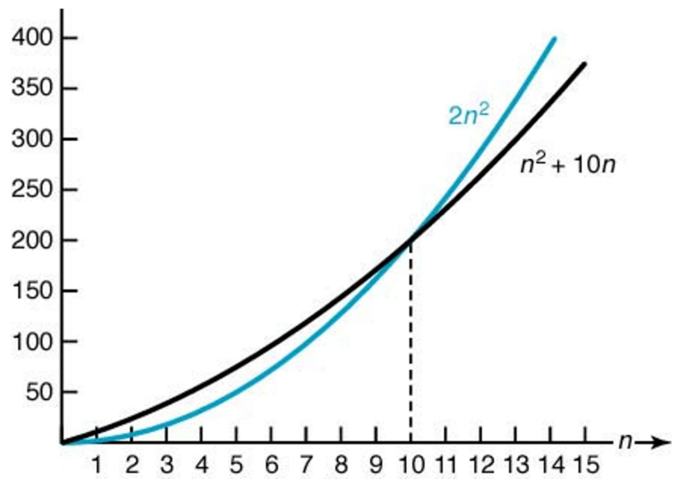
$n \geq 10$인 경우 다음 부등식이 참이다.

$$n^2 + 10n \leq 2n^2$$

따라서 $c = 2$와 $N = 10$을 선택하면 Big-O의 정의에 의해 다음이 성립한다.

$$n^2 + 10n \in O(n^2)$$

아래 그래프가 이를 잘 설명한다.



예제 2

모든 정수 n에 대해 다음 부등식이 참이다

$$5n^2 \leq 5n^2$$

따라서 $c = 5$와 $N = 1$을 선택하면 Big-O의 정의에 의해 다음이 성립한다.

$$5n^2 \in O(n^2)$$

예제 3

2보다 같거나 큰 임의의 양의 정수 n에 대해 다음 부등식이 참이다.

$$2\,n\ \log(n) \leq n^2$$

따라서 $c = 1$과 $N = 2$을 선택하면 Big-O의 정의에 의해 다음이 성립한다.

$$2\,n\ \log(n) \in O(n^2)$$

예제 4

임의의 양의 정수 n에 대해 다음 부등식이 참이다.

$$n \leq n^2$$

따라서 $c = 1$과 $N = 1$을 선택하면 Big-O의 정의에 의해 다음이 성립한다.

$$n \in O(n^2)$$

예제 5

c와 N을 아무리 크게 잡더라도 n이 c보다 크면 다음 부등식이 참이다.

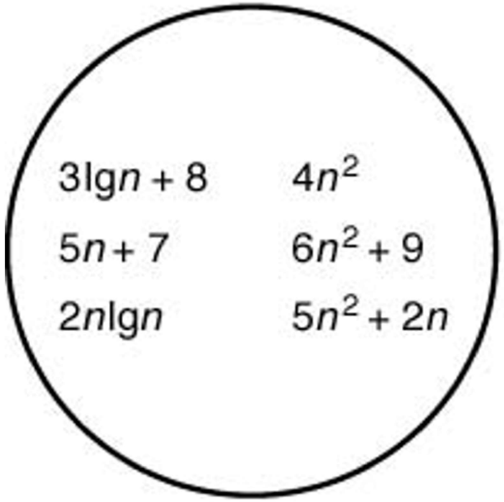
$$n^3 > c \cdot n^2$$

따라서 다음이 성립한다.

$$n^3 \notin O(n^2)$$

$O(f(n))$

$O(f(n))$ 에 속하는 시간 복잡도 함수는 기본적으로 $f(n)$ 과 비슷하거나 보다 느리게 증가하는 그래프를 갖는다.



(a) $O(n^2)$

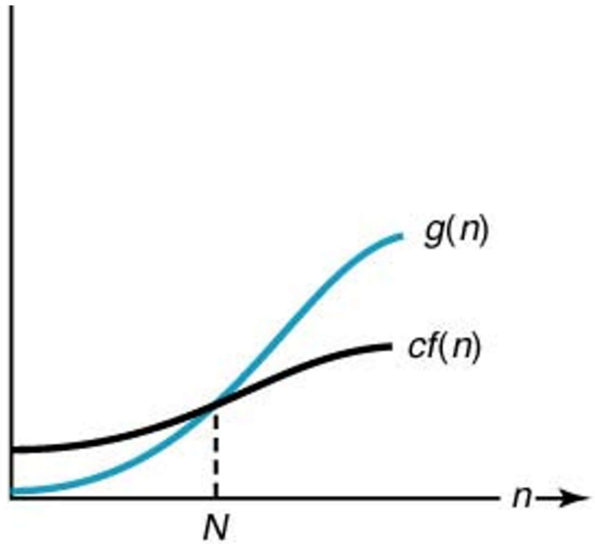
Ω 표기법

다음 성질을 갖는 양의 실수 c 와 양의 정수 N 이 존재할 때 $g(n) \in \Omega(f(n))$ 으로 표현한다.

$$n \geq N\text{인 임의의 정수 } n\text{에 대해 } g(n) \geq c \cdot f(n)$$

알고리즘의 시간 복잡도 측면

- 입력 크기 n에 대해 시간 복잡도 $g(n)$의 수행시간은 궁극적으로 $f(n)$보다 효율적이지 못함을 의미한다.
- $g(n)$의 수행시간이 최선(best)의 경우에도 $f(n)$의 수행시간보다는 빠르지 않다.



(b) $g(n) \in \Omega(f(n))$

예제 6

2보다 같거나 큰 임의의 양의 정수 n에 대해 다음 부등식이 참이다.

$$\frac{n(n-1)}{2} \geq \frac{n^2}{4}$$

따라서 $c = \frac{1}{4}$과 $N = 2$을 선택하면 Ω의 정의에 의해 다음이 성립한다.

$$\frac{n(n-1)}{2} \in \Omega(n^2)$$

예제 7

임의의 양의 정수 n에 대해 다음 부등식이 참이다.

$$n^3 \geq n^2$$

따라서 $c = 1$과 $N = 1$을 선택하면 Ω의 정의에 의해 다음이 성립한다.

$$n^3 \in \Omega(n^2)$$

예제 8

임의의 정수 n에 대해 다음 부등식이 참이다.

$$6n^6 + n^4 \geq n^2$$

따라서 $c = 1$과 $N = 1$을 선택하면 Ω의 정의에 의해 다음이 성립한다.

$$6n^6 + n^4 \in \Omega(n^2)$$

예제 9

임의의 양의 정수 n에 대해 다음 부등식이 참이다.

$$2^n + 4n \geq n^2$$

따라서 $c = 1$과 $N = 1$을 선택하면 Ω의 정의에 의해 다음이 성립한다.

$$2^n + 4n \in \Omega(n^2)$$

예제 10

양의 실수 c를 아무리 작게 잡더라도 n 이 $\frac{1}{c}$ 보다 크면 다음 부등식이 참이다.

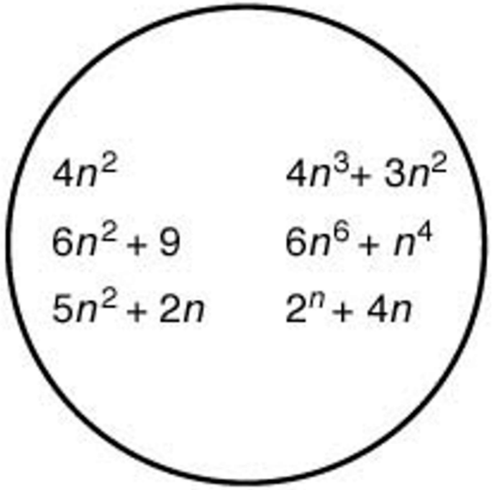
$$n \le cn^2$$

따라서 다음이 성립한다.

$$n \notin \Omega(n^2)$$

$\Omega(f(n))$

$\Omega(f(n))$ 에 속하는 시간 복잡도 함수는 기본적으로 $f(n)$ 과 비슷하거나 보다 빠르게 증가하는 그래프를 갖는다.

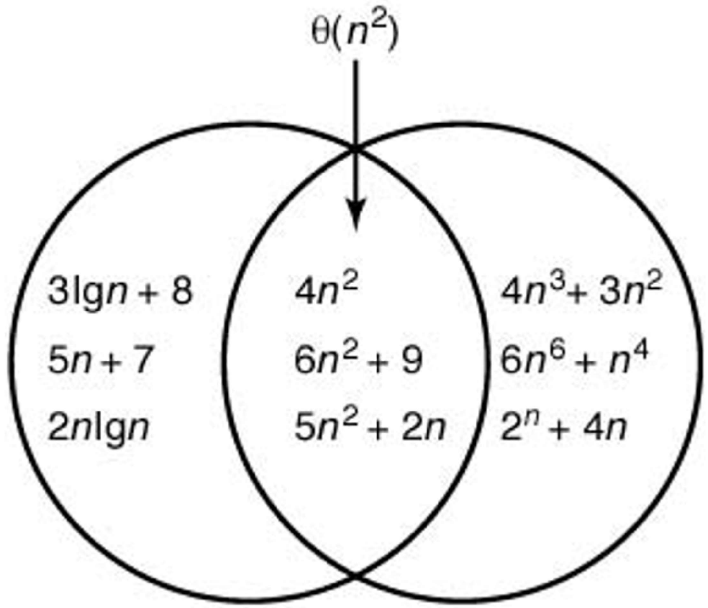


(b) $\Omega(n^2)$

차수(Θ) 표기법

$$\Theta(f(n)) = O(f(n)) \cap \Omega(f(n))$$

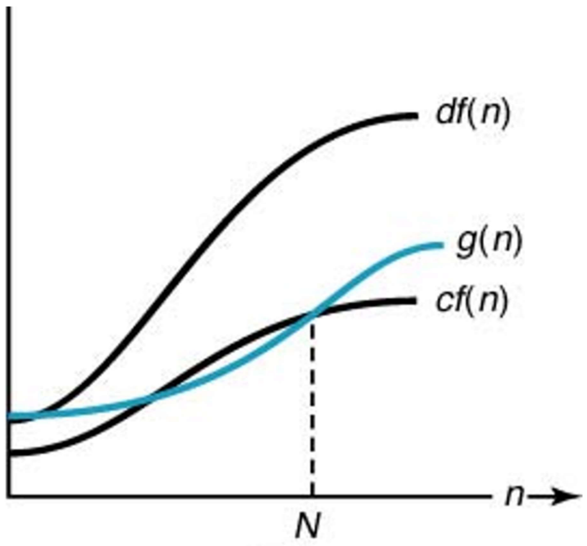
$\Theta(f(n))$ 에 속하는 시간 복잡도 함수는 기본적으로 $f(n)$ 과 비슷하게 증가하는 그래프를 갖는다.



(c) $\theta(n^2) = O(n^2) \cap \Omega(n^2)$

알고리즘의 시간 복잡도 측면

- $\Theta(f(n))$의 차수를 갖는 알고리즘은 입력 크기 n에 대해 최악의 경우와 최선의 경우 모두 시간 복잡도 $f(n)$의 수행시간보다 느리지도 않고 빠르지도 않다.
- 기본적으로 $f(n)$의 시간 내에 실행이 종료되며, 아래 그림이 이를 잘 반영한다.



(c) $g(n) \in \theta(f(n))$

차수의 특성

1. $g(n) \in O(f(n))$ 이 성립하면 $f(n) \in \Omega(g(n))$ 도 성립한다.
2. $f(n) \in \Omega(g(n))$ 이 성립하면 $g(n) \in O(f(n))$ 도 성립한다.
3. $g(n) \in \Theta(f(n))$ 이 성립하면 $f(n) \in \Theta(g(n))$ 도 성립한다.
4. 로그 함수는 모두 동일한 복잡도 카테고리에 속한다.

$$\log_a n \in \Theta(\log_b n)$$

$2 < j < k$ 와 $1 < a < b$ 가 성립한다고 가정했을 때 다음이 성립한다.

$$\Theta(\lg n),\quad \Theta(n),\quad \Theta(n\,\lg n),\quad \Theta(n^2),\quad \Theta(n^j),\quad \Theta(n^k),\quad \Theta(a^n),\quad \Theta(b^n),\quad \Theta(n!)$$