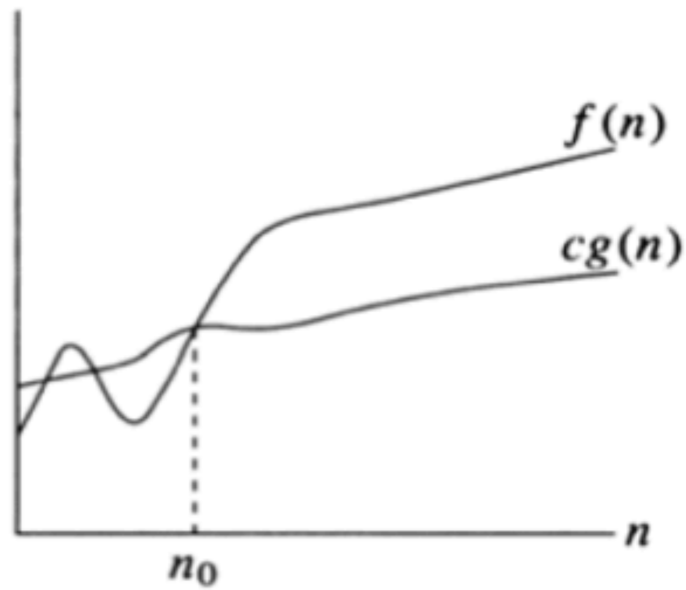
NOTASI ASIMTOTIK (2)

ANALISIS DAN DESAIN ALGORITMA 1

NOTASI OMEGA (Ω)

- **Pengucapan: 'big-omega' atau 'omega'**
  - Ω (g(n)) diucapkan 'big-omega dari g(n)'.
- **Definisi:**
  - Untuk suatu fungsi g(n), Ω (g(n)) adalah himpunan fungsi

    Ω (g(n))={f(n): terdapat konstanta positif c dan n_0 sedemikian hingga $0 \le cg(n) \le f(n)$ untuk semua $n \ge n_0$ }.
- **Jika ditulis *f*(*n*) = Ω (*g*(*n*)) berarti *f*(*n*) adalah anggota himpunan Ω (*g*(*n*)).**



- **Notasi Ω menunjukkan batas bawah asimtotik.**
- **Jika running time algoritma adalah Ω(g(n)):**
  - Tidak peduli bagaimana input berukuran n dipilih untuk setiap nilai n, running time untuk input tersebut paling sedikit g(n), untuk n cukup besar
- **Dengan kata lain: g(n) adalah best-case running time dari algoritma.**
  - Contoh: best case dari insertion sort adalah Ω(n), yang berarti bahwa best case untuk running time insertion sort adalah Ω(n).
- **Jadi: running time insertion sort berada antara Ω(n) dan $O(n^2)$.**

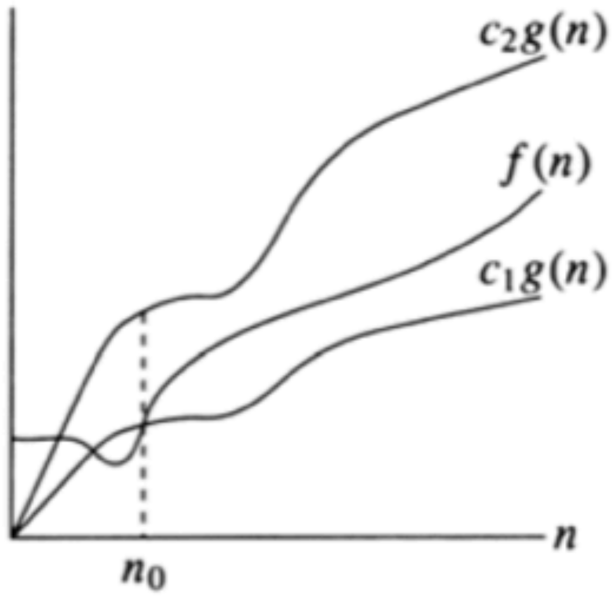
CONTOH BIG-OMEGA

- **Contoh: apakah $5n^2$ berada pada $\Omega(n)$?**
  - Dengan mengambil $c = 5$ dan $n_0 = 1$, jelas terlihat bahwa $5n \le 5n^2$ untuk $n \ge 1$. Jadi $5n^2$ berada pada $\Omega(n)$.

NOTASI THETA (Θ)

- **Pengucapan: 'theta'**
  - Θ (g(n)) diucapkan 'theta dari g(n)'.
- **Definisi:**
  - Untuk suatu fungsi g(n), Θ (g(n)) adalah himpunan fungsi

    Θ (g(n))={f(n): terdapat konstanta positif c_1, c_2, dan n_0 sedemikian hingga $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ untuk semua $n \ge n_0$ }.
- **g(n) disebut sebagai batas ketat asimtotik (asymptotically tight bound) untuk f(n).**



- **Theta mendeskripsikan batas bawah dan atas asimtotik.**

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

- **Karena $\Theta(g(n))$ adalah himpunan, kita dapat menulis "$f(n) \in \Theta(g(n))$".**
  - Biasanya ditulis: "$f(n) = \Theta(g(n))$"
- **Setiap anggota $f(n) \in \Theta(g(n))$ harus non-negatif secara asimtotis. Akibatnya, g(n) harus non-negatif secara asimtotis.**

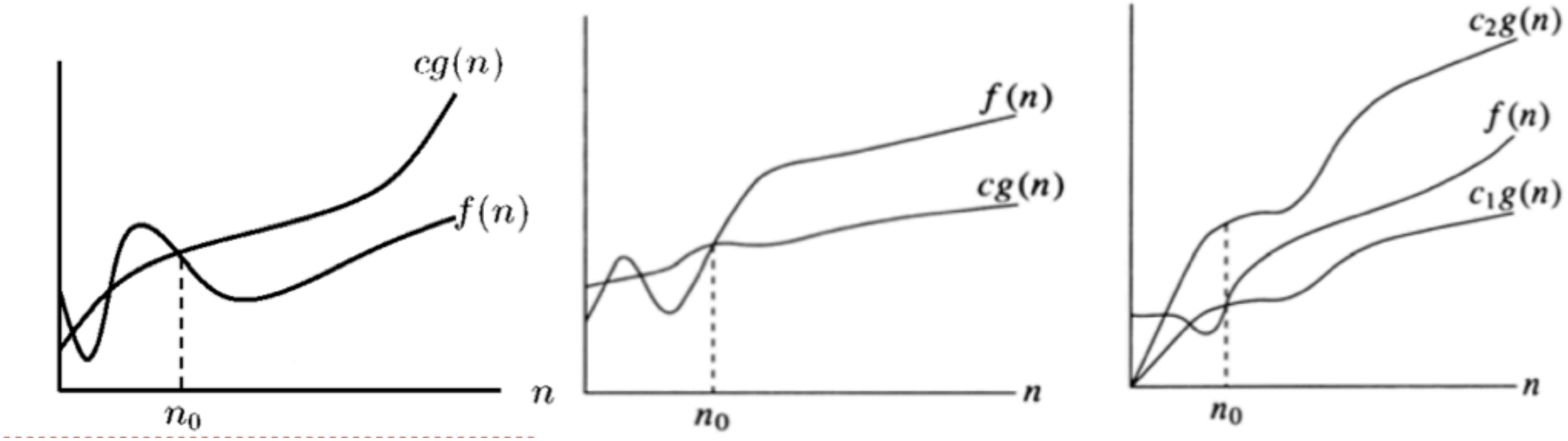
CONTOH THETA

- **Contoh: akan ditunjukkan bahwa ½ n^2 – 3n = $\Theta(n^2)$.**
  - Tentukan konstanta positif c_1, c_2, dan n_0 sedemikian hingga

    $c_1 n^2 \leq$ ½ $n^2 - 3n \leq c_2 n^2$ untuk semua $n \geq n_0$.
  - Menentukan c_1, c_2, and n_0
    - $c_1 n^2 \leq n^2/2 - 3n$
      - c_1 harus lebih kecil dari ½. Misalkan dipilih c_1 = ¼.

        ¼ $n^2 \leq$ ½ $n^2 - 3n$ → $3n \leq$ ¼ n^2 → $n \geq 12$ (n_0=12)
    - $n^2/2 - 3n \leq n^2$ for $n \geq 1$ ($c_2 = 1$, n_0=1)
    - Nilai n_0 diambil dari irisan kedua n_0 di atas.
- **Jadi dengan memilih c_1 = ¼ , c_2 =1, and n_0=12, terbukti bahwa ½ n^2 – 3n = $\Theta(n^2)$.**

PERBANDINGAN GRAFIK



Cormen et.al. (2009)

HUBUNGAN ANTARA Θ, O, DAN Ω

- **Teorema:**
  - Untuk sembarang 2 fungsi $f(n)$ dan $g(n)$, maka $f(n) = \Theta(g(n))$ jika dan hanya jika $f(n) = O(g(n))$ dan $f(n) = \Omega(g(n))$.
- **Contoh:**
  - $n^2/2-3n=\Theta(n^2) \rightarrow n^2/2-3n=O(n^2)$ dan $n^2/2-3n=\Omega(n^2)$
  - $n^2/2-3n=O(n^2)$ dan $n^2/2-3n=\Omega(n^2) \rightarrow n^2/2-3n=\Theta(n^2)$

Contoh: Tentukan notasi Θ untuk $T(n) = 2n^2 + 6n + 1$.

Jawab:

Karena $2n^2 \le 2n^2 + 6n + 1$ untuk $n \ge 1$,

maka dengan $c_1 = 2$ kita memperoleh

$$2n^2 + 6n + 1 = \Omega(n^2)$$

Karena $2n^2 + 6n + 1 \le 2n^2 + 6n^2 + n^2 = 9n^2$ untuk semua $n \ge 1$ (c_2 =9 dan $n_0 = 1$)

maka $2n^2 + 6n + 1 = O(n^2)$

Karena $2n^2 + 5n + 1 = O(n^2)$ dan $2n^2 + 6n + 1 = \Omega(n^2)$,

maka $2n^2 + 6n + 1 = \Theta(n^2)$ untuk n $\ge$ 1.

Contoh: Tentukan notasi-notasi O, Ω dan Θ untuk $T(n) = 5n^3 + 6n^2 \log n$.

Jawab:

Karena $6n^2 \log n \le 6n^3$, maka $5n^3 + 6n^2 \log n \le 11n^3$ untuk $n \ge 1$. Dengan mengambil $c_1 = 11$, maka

$$5n^3 + 6n^2 \log n = O(n^3)$$

Karena $5n^3 + 6n^2 \log n \ge 5n^3$ untuk $n \ge 1$, maka maka dengan mengambil $c_2 = 5$ kita memperoleh

$$5n^3 + 6n^2 \log n = \Omega(n^3)$$

Karena $5n^3 + 6n^2 \log n = O(n^3)$ dan $5n^3 + 6n^2 \log n = \Omega(n^3)$, maka $5n^3 + 6n^2 \log n = \Theta(n^3)$ untuk $n \ge 1$.

CONTOH

- Tentukan kompleksitas waktu dari algoritma di bawah ini dari jumlah operasi a←a+1

```
for i ← 1 to n do
  for j ← 1 to i do
      for k ← j to n do
          a ← a + 1
     endfor
  endfor
endfor
```

- Tentukan pula nilai O-besar, Ω-besar, dan Θ-besar dari algoritma di atas (dengan penjelasan).

JAWABAN

Untuk $i = 1$,

Untuk $j = 1$, jumlah perhitungan = n kali

Untuk $i = 2$,

Untuk $j = 1$, jumlah perhitungan = n kali

Untuk $j = 2$, jumlah perhitungan = $n - 1$ kali

...

Untuk $i = n$,

Untuk $j = 1$, jumlah perhitungan = n kali

Untuk $j = 2$, jumlah perhitungan = $n - 1$ kali

...

Untuk $j = n$, jumlah perhitungan = 1 kali.

Jadi jumlah perhitungan = $T(n) = n^2 + (n - 1)^2 + (n - 2)^2 + ... + 1$

- **Salah satu cara penjelasan:**

  $T(n) = n^2 + (n - 1)^2 + (n - 2)^2 + \dots + 1$

  $= n(n + 1)(2n + 1)/6$

  $= (2n^3 + 3n^2 + 1)/6$

- **Diperoleh $T(n) \leq n^3$ untuk $n \geq 1$ dan**

  $T(n) \geq n^3/6$ untuk $n \geq 1$.

Maka: $T(n) = O(n^3) = \Omega(n^3) = \Theta(n^3)$.

NOTASI LITTLE-OH (o)

- **Untuk menotasikan batas atas yang tidak ketat secara asimtotik.**
- **Definisi:**
  - Untuk suatu fungsi g(n), o(g(n)) adalah himpunan fungsi o(g(n))={f(n): untuk **sembarang** konstanta positif $c > 0$, terdapat konstanta $n_0 > 0$ sedemikian hingga $0 \leq f(n) < cg(n)$ untuk semua $n \geq n_0$ }.
- **Contoh: $2n = o(n^2)$, tetapi $2n^2 \neq o(n^2)$.**
- **Perbedaan dengan O(g(n)):**
  - Pada $f(n) = O(g(n))$: batas $0 \leq f(n) \leq cg(n)$ berlaku untuk **suatu** konstanta $c > 0$.
  - Pada $f(n) = o(g(n))$: batas $0 \leq f(n) < cg(n)$ berlaku untuk **semua** konstanta $c > 0$.

NOTASI LITTLE-OMEGA (ω)

- **Untuk menotasikan batas bawah yang tidak ketat secara asimtotik.**
- **Definisi:**
  - Untuk suatu fungsi g(n), ω(g(n)) adalah himpunan fungsi ω(g(n))={f(n): untuk sembarang konstanta positif c > 0, terdapat konstanta n_0 > 0 sedemikian hingga $0 \leq cg(n) < f(n)$ untuk semua $n \geq n_0$ }.
- **Contoh: $n^2/2 = \omega(n)$, tetapi $n^2/2 \neq \omega(n^2)$.**
- **Perbedaan dengan Ω(g(n)):**
  - Pada $f(n) = \Omega(g(n))$: batas $0 \leq cg(n) < f(n)$ berlaku untuk **suatu** konstanta c > 0.
  - Pada $f(n) = \omega(g(n))$: batas $0 \leq cg(n) < f(n)$ berlaku untuk **semua** konstanta c > 0.

ANALOGI DENGAN BILANGAN REAL

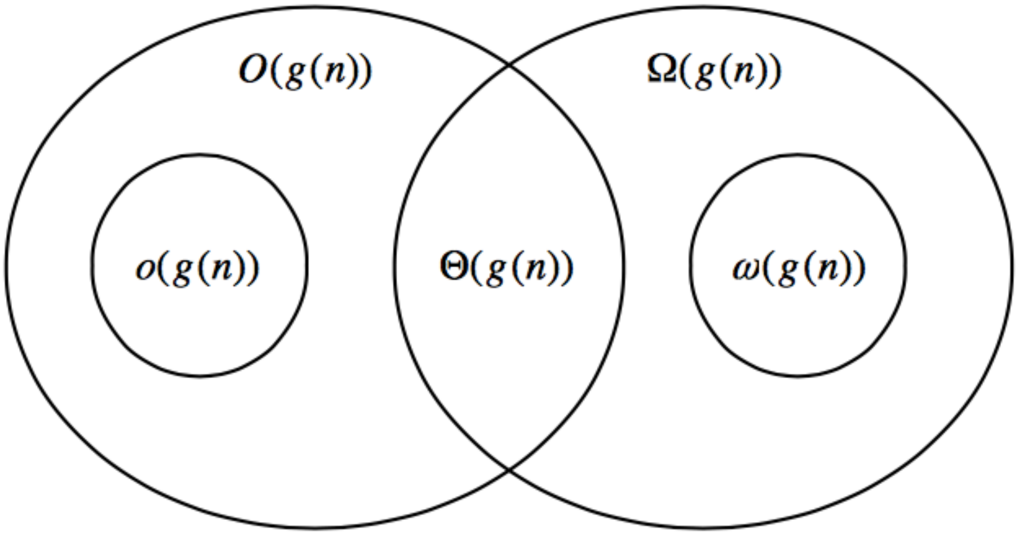
- **Jika fungsi f dan g dianalogikan dengan bilangan real a dan b:**
  - $f(n) = O(g(n)) \approx a \leq b$
  - $f(n) = \Omega(g(n)) \approx a \geq b$
  - $f(n) = \Theta(g(n)) \approx a = b$
  - $f(n) = o(g(n)) \approx a < b$
  - $f(n) = \omega(g(n)) \approx a > b$
- **Properti trichotomy untuk bilangan real tidak berlaku pada notasi asimtotik:**
  - Trichotomy: untuk sembarang 2 bilangan real a and b, salah satu dari berikut pasti berlaku: $a < b$, $a = b$, or $a > b$.
  - Tidak semua fungsi dapat dibandingkan secara asimtotik.
    - Untuk 2 fungsi $f(n)$ and $g(n)$, mungkin terjadi bahwa $f(n)=O(g(n))$ dan $f(n)=\Omega(g(n))$ tidak berlaku.
    - Contoh: fungsi n dan $n^{1+\sin n}$ tidak dapat dibandingkan, karena nilai dari $n^{1+\sin n}$ berosilasi antara 0 dan 2.

SIFAT-SIFAT NOTASI ASIMTOTIK

- **Transitif:**
  - $f(n) = \Theta(g(n))$ dan $g(n) = \Theta(h(n))$ maka $f(n) = \Theta(h(n))$
  - $f(n) = O(g(n))$ dan $g(n) = O(h(n))$ maka $f(n) = O(h(n))$
  - $f(n) = \Omega(g(n))$ dan $g(n) = \Omega(h(n))$ maka $f(n) = \Omega(h(n))$
  - $f(n) = o(g(n))$ dan $g(n) = o(h(n))$ maka $f(n) = o(h(n))$
  - $f(n) = \omega(g(n))$ dan $g(n) = \omega(h(n))$ maka $f(n) = \omega(h(n))$
- **Refleksif:**
  - $f(n) = \Theta(f(n))$
  - $f(n) = O(f(n))$
  - $f(n) = \Omega(f(n))$

- **Simetri:**
  - $f(n) = \Theta(g(n))$ jika dan hanya jika $g(n) = \Theta(f(n))$
- **Simetri transpos:**
  - $f(n) = O(g(n))$ jika dan hanya jika $g(n) = \Omega(f(n))$
  - $f(n) = o(g(n))$ jika dan hanya jika $g(n) = \omega(f(n))$

HUBUNGAN ANTAR NOTASI ASIMTOTIK



LATIHAN

- **Mengapa pernyataan “running time algoritma A paling kecil $O(n^2)$” tidak memiliki arti?**
- **Apakah $27n^2 + 2n + 12 = O(n^2)$? Buktikan.**
- **Tunjukkan $T(n) = 6*2^n + 2n^2 = O(2^n)$**
- **Tunjukkan $T(n) = 1 + 2 + .. + n = O(n^2)$**
- **Tunjukkan $T(n) = n! = O(n^n)$**