Lecture 2

Asymptotic Notations

Big--Oh: English Definition

- Let T(n) = function on n = 1,2,3,…
  [usually, the worst--case running time of an algorithm]

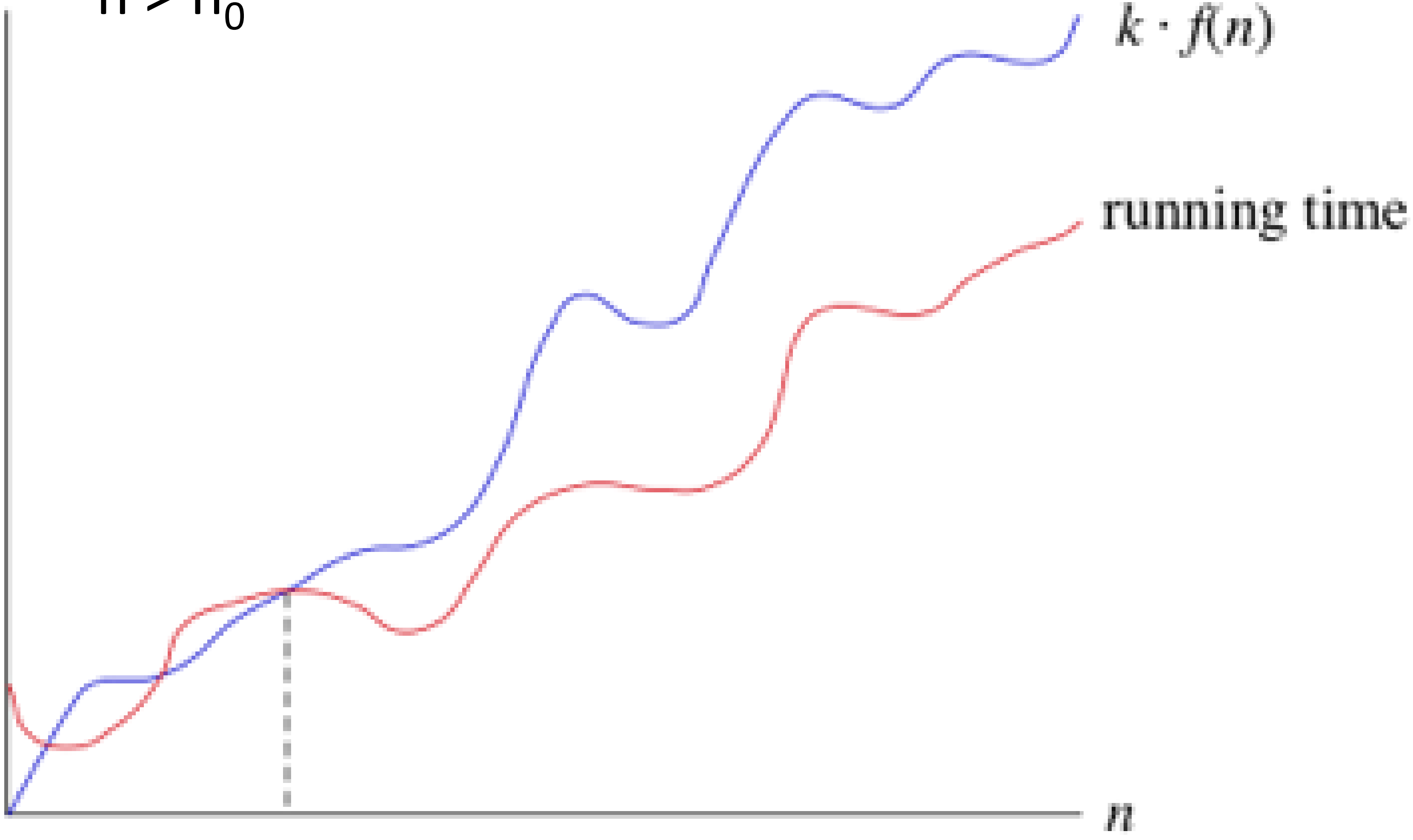
Q : When is T(n) = O(f(n)) ?
A : if eventually (for all sufficiently large n), T(n) is bounded above by a constant multiple of f(n)

Big O Notation

- The running time grows at most this much, but it could grow more slowly
- If a running time is O(f(n)), then for large enough *n* the running time is at most k·f(n) for some constant k .
- We use big-O notation for **asymptotic upper bounds**, since it bounds the growth of the running time from above for large enough input sizes.

Big O Notation

$T(n) = O(f(n))$ if an only if there exist constants k and n_0 such that $T(n) < k.f(n)$ for all $n > n_0$



Definition of big-Oh

Function $f(n) = O(g(n))$ ("f is big oh of g") iff

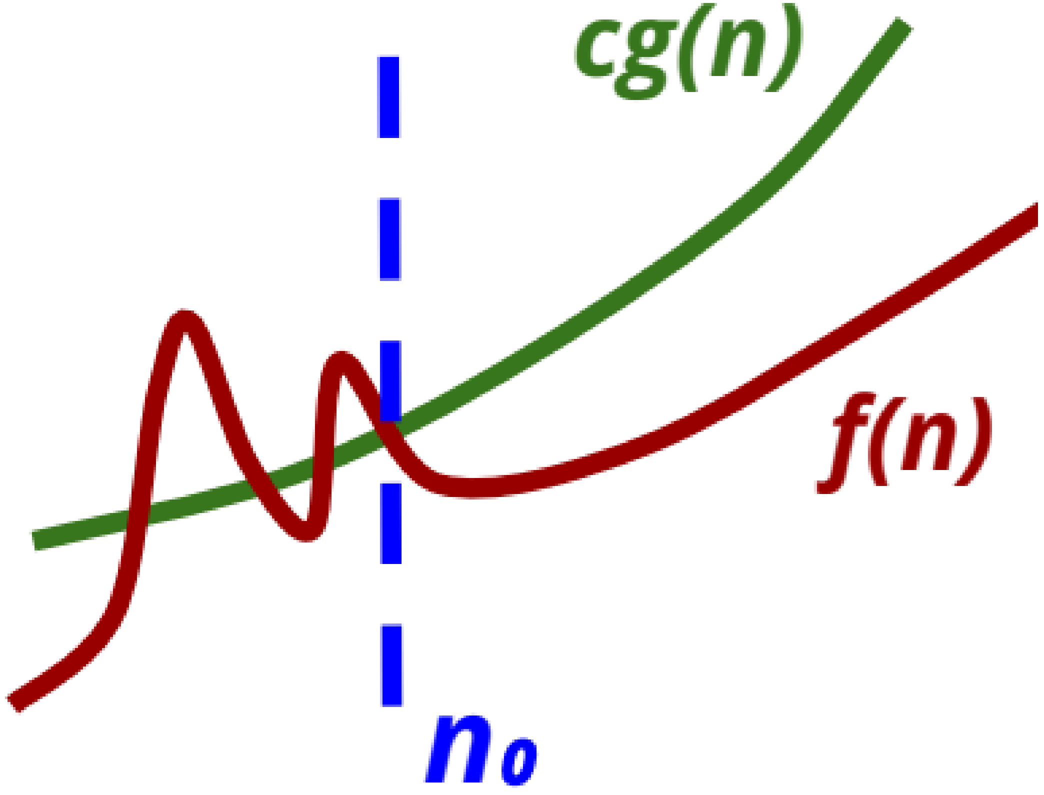
(i) There is some positive $n_0 \in N$

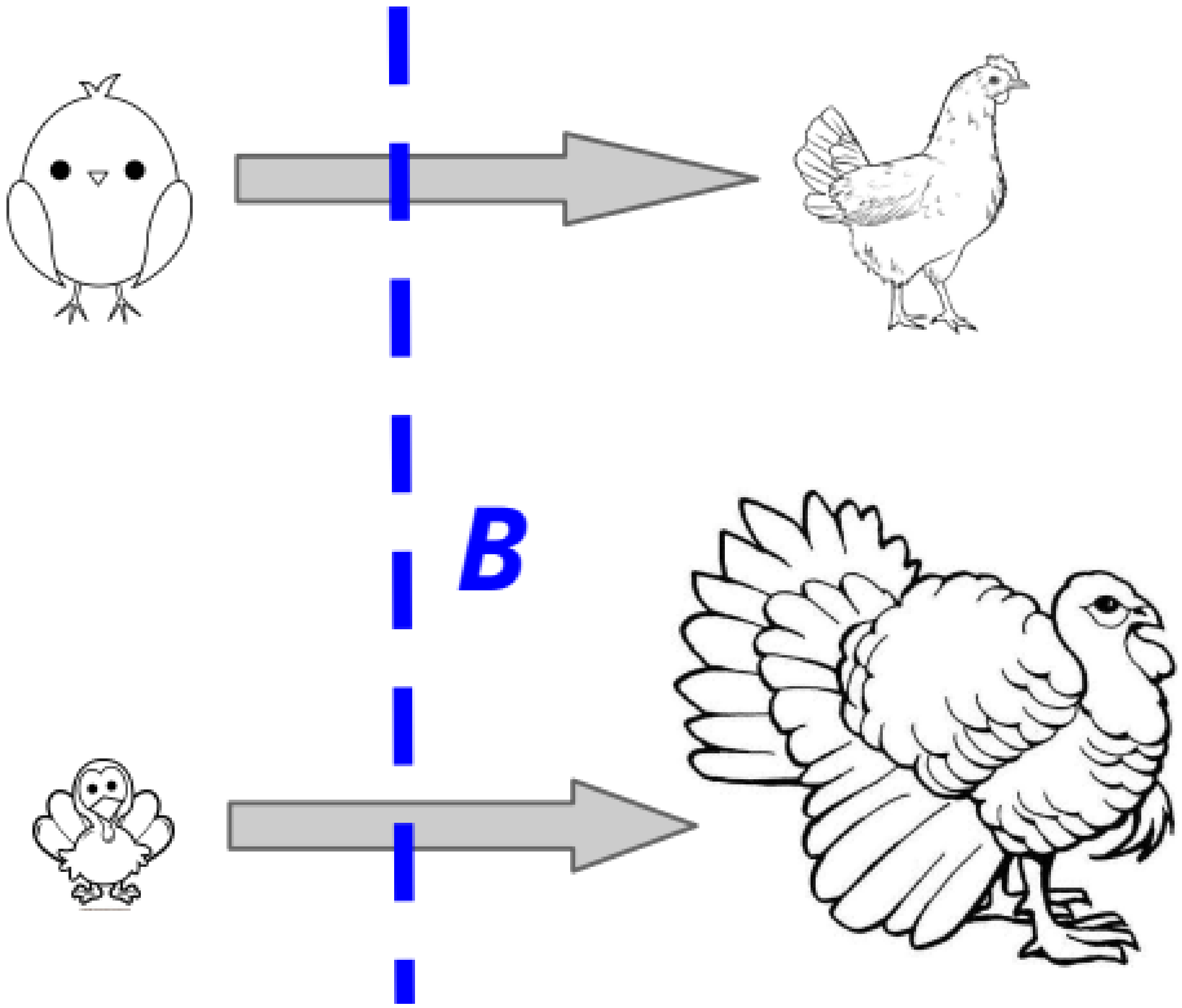
(ii) There is some positive $c \in R$

such that

$\forall n \geq n_0, f(n) \leq cg(n)$

*Beyond the **breakpoint n_0**, f(n) is **upper-bounded** by **cg(n)**, where **c** is some wisely chosen constant multiplier.*





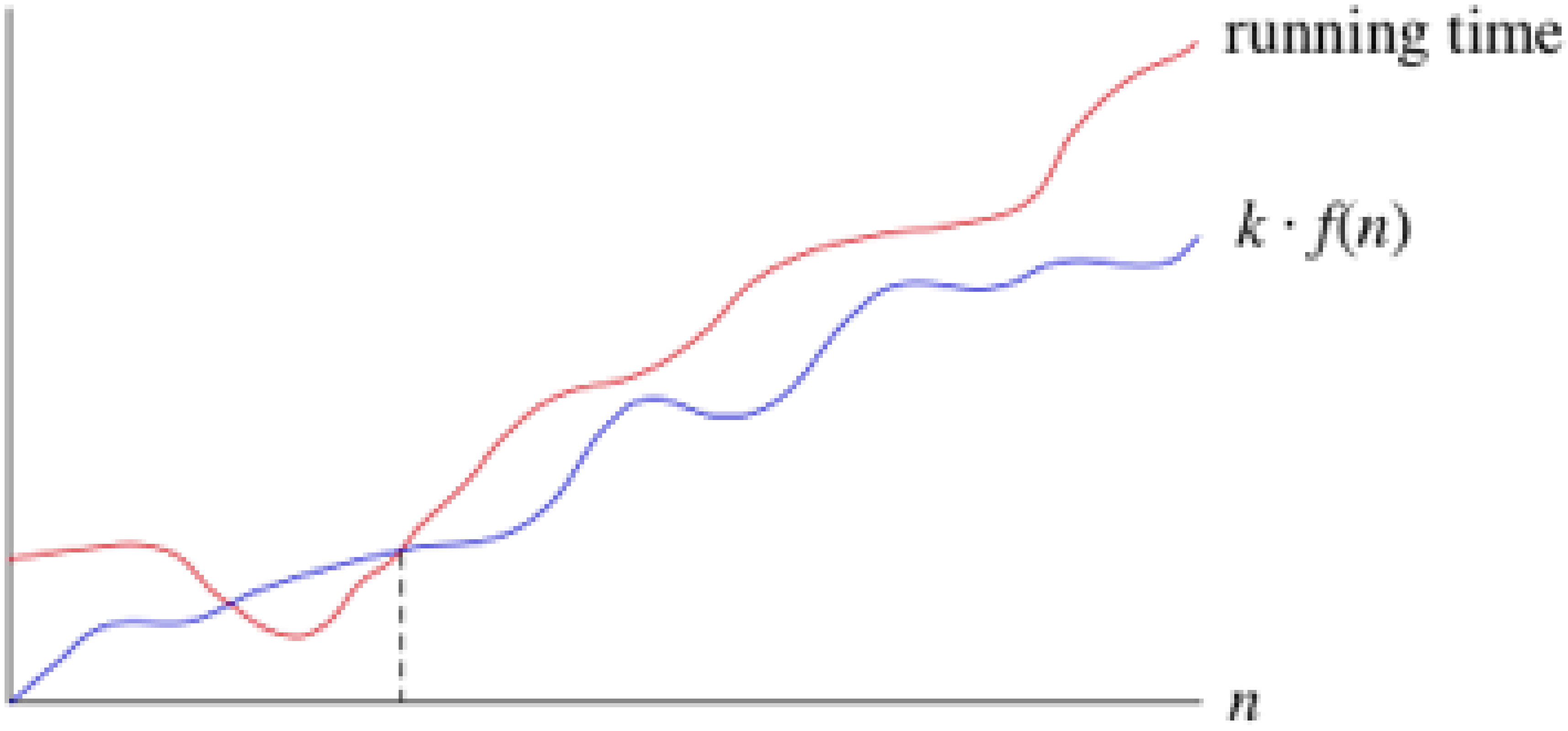
- Chicken **grows slower** than turkey, or **chicken size** is in **O(turkey size).** What it really means:
- Baby chicken might be larger than baby turkey at the beginning.
- But after certain “**breakpoint**”, the chicken size will be **surpassed** by the turkey size.
- From the **breakpoint on**, the chicken size will **always** be smaller than the turkey size.

Big O Example

$$T(n) = \frac{1}{2}n^2 + 3n$$

Big-Ω (Big-Omega) notation

$T(n) = \Omega(f(n))$ if an only if there exist constants k and n_0 such that $T(n) \geq k.f(n)$ for all $n > n_0$



Big-Ω Example

$$T(n) = \frac{1}{2}n^2 + 3n$$

Big-Ω Example

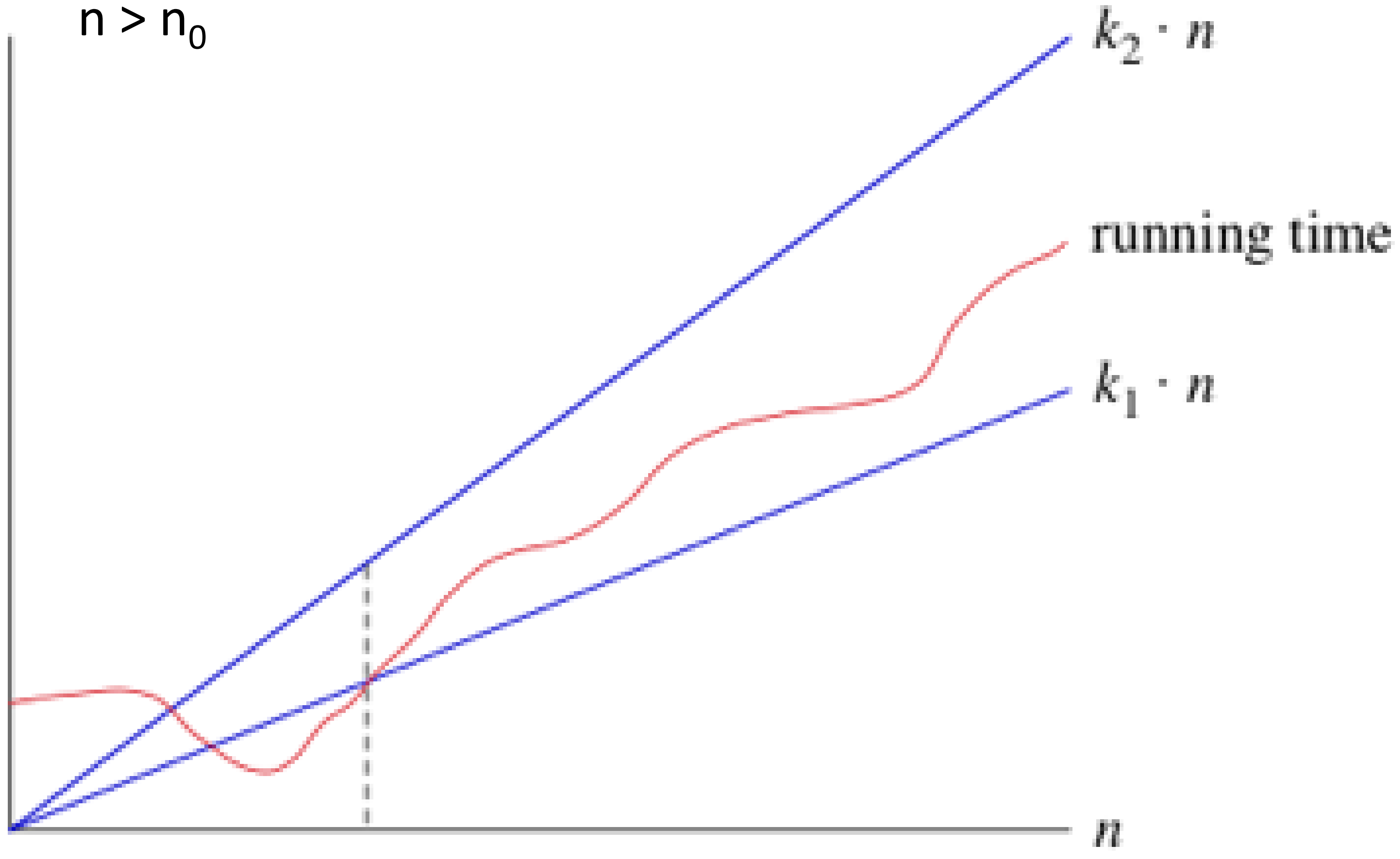
$$T(n) = \frac{1}{2}n^2 + 3n$$

Theta Notation Θ(n)

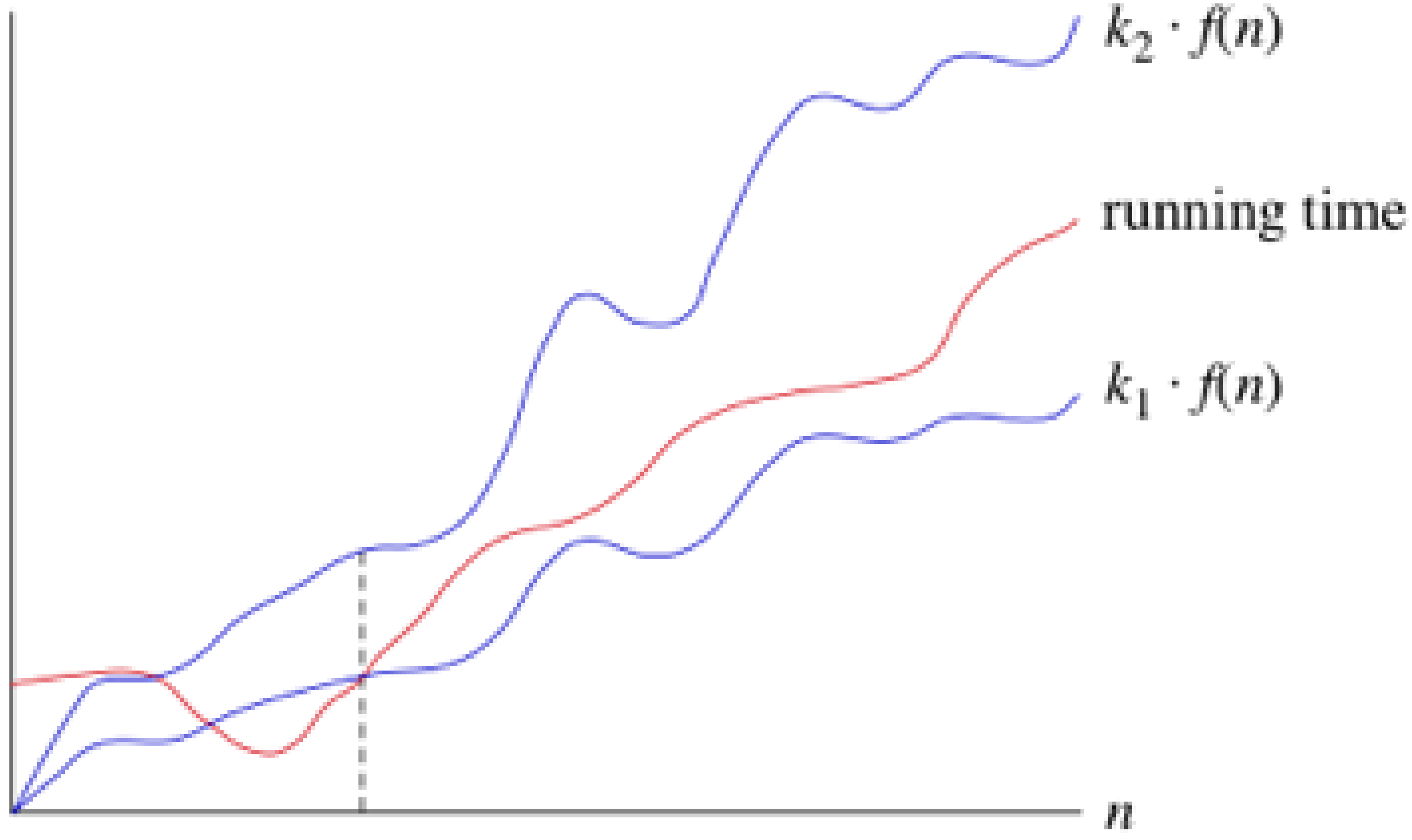
- Once n gets large enough, the running time is at least $k_1.n$ and at most $k_2 \cdot n$ for some constants k_1 and k_2

Theta Notation Θ(n)

$T(n) = \Theta(f(n))$ if an only if there exist constants k_1, k_2 and n_0 such that $k_{1.}f(n) \leq T(n) \leq k_2.f(n)$ for all $n > n_0$



Theta Notation Θ(n)



Θ (Theta) Example

$$T(n) = \frac{1}{2}n^2 + 3n$$

Theta Notation Θ(n)

- **asymptotically tight bound** on the running time.
- "Asymptotically" because it matters for only large values of n
- "Tight bound" because we've nailed the running time to within a constant factor above and below.

Asymptotic Notations

- Big-Oh, Big-Omega, Big-Theta4
- O(f(n)): The set of functions that grows no faster than f(n)
  - asymptotic upper-bound on growth rate
- Ω(f(n)): The set of functions that grows no slower than f(n)
  - asymptotic lower-bound on growth rate
- Θ(f(n)): The set of functions that grows no faster and no slower than f(n)
  - asymptotic tight-bound on growth rate

Asymptotic Notations

- All three (Omega, O, Theta) give only *asymptotic information* ("for large input"):
  - Big O gives upper bound
  - Big Omega gives lower bound and
  - Big Theta gives both lower and upper bounds
- Note that this notation is **not** related to the best, worst and average cases analysis of algorithms. Each one of these can be applied to each analysis.

Knowing the definition,
now we can write proofs for big-Oh.

The key is finding n_0 and c

Asymptotic Analysis
Big--Oh: Basic Examples

Example 1

Let $T(n) = \frac{1}{2}n^2 + 3n$ Which of the following statements are true?

Check all that apply.

a) T(n) = O(n)
b) T(n) = Ω (n)
c) T(n) = Θ (n^2)
d) T(n) = O (n^3)

Example 1

Let $T(n) = \frac{1}{2}n^2 + 3n$. Which of the following statements are true?

Check all that apply.

a) $T(n) = O(n)$

b) $T(n) = \Omega(n)$, [$n_0 = 1$, $c = 1/2$]

c) $T(n) = \Theta(n^2)$, [$n_0 = 1$, $c_1 = 1/2$, $c_2 = 4$]

d) $T(n) = O(n^3)$, [$n_0 = 1$, $c = 4$]

b, c, d are correct

Example 1

Let $T(n) = \frac{1}{2}n^2 + 3n$. Which of the following statements are true?

(a) T(n) = O(n)

$$\frac{1}{2}n^2 + 3n \ \le c.n$$

$\frac{1}{2}n + 3 \ \le c$ which is not possible because input n cannot be bounded above by a constant (input can be of any length)

So (a) is not correct

Example 1

Let $T(n) = \frac{1}{2}n^2 + 3n$. Which of the following statements are true?

(b) T(n) = Ω (n)

$$\frac{1}{2}n^2 + 3n \geq c.n$$

$\frac{1}{2}n + 3 \geq c$ This holds for c =1/2 and $n \geq n_0$, $n_0 = 1$

Example 1

Let $T(n) = \frac{1}{2}n^2 + 3n$. Which of the following statements are true?

(c) $T(n) = \Theta(n^2)$

$$c_1 n \le \frac{1}{2}n^2 + 3n \le c_2 n$$

$c_1 \le \frac{1}{2}n + 3 \le c_2$ This holds for $c_1 = 1/2$, $c_2 = 4$ and $n \ge n_0$, $n_0 = 1$

Example 2

Which of the following is correct?

a) $\log_2 n$ is Θ ($\log_8 n$)
b) $\log_2 n$ is O ($\log_8 n$)
c) $\log_2 n$ is Ω ($\log_8 n$)

$\log_8 n = \log_2 n \,/\, \log_2 8 \;= \frac{\log_2 n}{3}$

Change-of-Base Formula:

$$\log_b(x) = \frac{\log_d(x)}{\log_d(b)}$$

Example 2

Which of the following is correct?

a) $\log_2 n$ is $\Theta(\log_8 n)$
b) $\log_2 n$ is $O(\log_8 n)$
c) $\log_2 n$ is $\Omega(\log_8 n)$

a, b, c are correct

Example 2

Which of the following is correct?

a) $\log_2 n$ is $\Theta(\log_8 n)$

$$c_1 \log_8 n \leq \log_2 n \leq c_2 \log_8 n$$

$$c_1 \frac{\log_2 n}{3} \leq \log_2 n \leq c_2 \frac{\log_2 n}{3}$$

$$\frac{c_1}{3} \leq 1 \leq \frac{c_2}{3}$$

It holds for $c_1 = 2$ and $c_2 = 4$

Example 3

Which of the following is correct?

a) $n^3 log_2 n$ is Θ ($3nlog_8\, n$)
b) $n^3 log_2 n$ is O ($3nlog_8\, n$)
c) $n^3 log_2 n$ is Ω ($3nlog_8\, n$)

Example 3

Which of the following is correct?

a) $n^3 log_2 n$ is $\Theta\ (3nlog_8\ n)$
b) $n^3 log_2 n$ is $O\ (3nlog_8\ n)$
c) $n^3 log_2 n$ is $\Omega\ (3nlog_8\ n)$

C is correct

Example 4

Which of the following is correct?

a) 8^n is $\Theta(4^n)$
b) 8^n is $O(4^n)$
c) 8^n is $\Omega(4^n)$

Example 4

Which of the following is correct?

a) 8^n is Θ (4^n)
b) 8^n is O (4^n)
c) 8^n is Ω (4^n)

(b) $8^n = 2^{3n} = 2^{2n+n}$

$$2^{2n+n} \leq c2^{2n}$$
$$2^{2n}2^n \leq c2^{2n}$$
$$2^n \leq c$$

Which is a contradiction

Option c is correct for constant c = 1 and n_0 =1

$2^n \geq c$

Example 5

Claim $2^{n+10} = O\,(2^n)$ Is it True or False?

Example 5

Claim $2^{n+10} = O\ (2^n)$

Proof: we need to pick constants c, n_0 such that

$$2^{n+10} \leq c.2^n \qquad n \geq n_0$$

Note $2^{n+10} = 2^n \times 2^{10} = 2^n \times 1024$

So if we choose n_0= 1, c = 1024, claim holds

Example 6

Claim $2^{10n} \neq O\ (2^n)$

Proof: By contradiction. If $2^{10n} = O\ (2^n)$ then there exist constants $c, n_0 > 0$ such that

$$2^{10n} \leq c.2^n \qquad n \geq n_0$$

But then cancelling 2^n

$$2^{9n} \leq c \qquad \forall\ n \geq n_0$$

Which is certainly false

Example 7

Prove T(n) = $2n^3 - 7n + 1 = \Omega(n^3)$

$$2n^3 - 7n + 1 \geq kn^3$$

$$2 - 7/n^2 + 1/n^3 \geq k$$

The above inequality does not hold for $n_0 = 1$ and 2 as L.H.S becomes negative and k has to be positive

Lets try $n_0 = 3$ and $k = 1$

$2 - 7/9 + 1/27 \geq 1$, *this is true*

What happens when n gets large?

When n gets large $7/n^2$ will become small and the L.H.S will become greater so the inequality holds for $n \geq n_0$, where $n_0 = 3$

Example 8

Prove T(n) = $2n^3 - 7n + 1 = O(n^3)$

$$2n^3 - 7n + 1 \le kn^3$$

Let k = 2

$$2n^3 - 7n + 1 \le 2n^3$$

$n_0 = 1, k = 2$ since 1-7n will be negative for n $\geq$ 1