Asymptotic Analysis

การวิเคราะห์เชิงเส้นกำกับ

เวลาการทำงาน (Running time)

- โปรแกรมใช้เวลาทำงานเท่าไร
  - คอมพิวเตอร์
  - ภาษาโปรแกรมและไลบรารี่
  - ตัวแปลภาษา
  - ปริมาณข้อมูล
- จะสนใจปริมาณข้อมูลเท่านั้น
  - วัดเวลาการทำงาน (running time) เป็นฟังก์ชันของปริมาณของข้อมูลเข้า

Counting Instructions

การนับจำนวนคำสั่ง

คำสั่งพื้นฐาน (Elementary operations)

- คำสั่งที่ใช้เวลาทำงานคงที่ ไม่ขึ้นกับปริมาณข้อมูลที่เกี่ยวข้อง

```
public class ArrayCollection implements Collection {
  ...
  private int indexOf(Object e) {
    for (int i=0; i<size; i++)
      if (elementData[i].equals(e)) return i;
    return -1;
  }
  ...
}
```

การนับจำนวนคำสั่ง

```java
public class ArrayCollection implements Collection {
  ...
  public void remove(Object e) {
    int i = indexOf(e);
    if (i != -1) {
      elementData[i] = elementData[--size];
      elementData[size] = null;
    }
  }
  ...
  private int indexOf(Object e) {
    for (int i=0; i<size; i++)
      if (elementData[i].equals(e)) return i;
    return -1;
  }
  ...
}
```

1 ครั้ง

1 ครั้ง

Size+1 ครั้ง

size ครั้ง

การนับจำนวนคำสั่ง

```
public void dummy1 {
  int c = 0;
  for (int i=0; i<n; i++)
    for (int j=0; j<m; j++)
      c = c+1;
}
```

m ครั้ง

n ครั้ง

mn ครั้ง

```
public void dummy2 {
  int c = 0;
  for (int i=0; i<n; i++)
    for (int j=0; j<i; j++)
      c = c+1;
}
```

i ครั้ง

n ครั้ง

0 + 1 + 2 +… + (n-1) ครั้ง

$= \sum_{i=1}^{n-1} i$

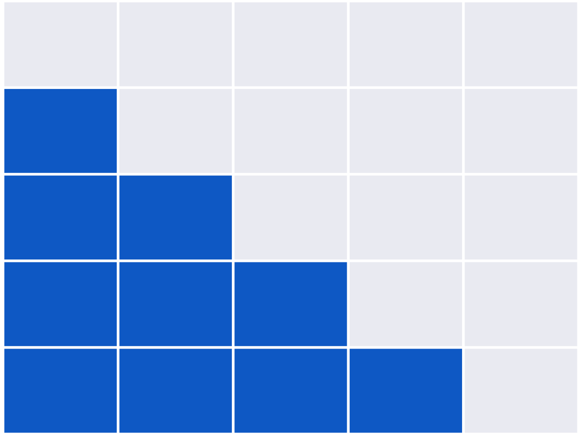
= n(n-1)/2 ครั้ง

การนับจำนวนคำสั่ง

```
public static int dummy (int[][] data) {
   int sum = 0;
   for (int[] data1 : data)
      for (int d : data1)
         sum += d;
   return sum;
}
public static int dummy (int[][] data) {
   int sum = 0;
   for (int i=0; i<data.length; i++)
      for (int j=0; j<data[i].length; j++)
         sum += data[i][j];
   return sum;
}
```

การนับจำนวนคำสั่ง

```
public static int dummyh (int[][] data) {
   int sum = 0;
   for (int i=0; i<data.length; i++)
      for (int j=0; j<i; j++)
         sum += data[i][j];
   return sum;
}
```



สัญกรณ์เชิง
กำกับ

Asymptotic Notation

สัญกรณ์เชิงเส้นกำกับ (Asymptotic Notation)

- running time เป็นฟังก์ชันที่มีพารามิเตอร์เป็นปริมาณ (ขนาด) ของข้อมูลเข้า
- เทียบระหว่างฟังก์ชันของเวลาทำงานของ 2 วิธี
- ใช้สัญกรณ์เชิงเส้นกำกับเพื่อเปรียบเทียบระหว่าง 2 วิธี
- สนใจอัตราการเติบโต (growth rate) เมื่อข้อมูลเพิ่มขึ้น
- โตช้ากว่า ดีกว่า

การเปรียบเทียบการโตของฟังก์ชัน

$f(n)$ โตช้ากว่า $g(n)$ $\quad iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$

ใช้สัญลักษณ์ $f(n) \ll g(n) \quad iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$

Example: $0.5^n \ll 1 \ll \log n \ll n \ll 10^n$

$$\lim_{n\to\infty} \frac{0.5^n}{1} = 0,\ \lim_{n\to\infty} \frac{1}{\log n} = 0,\ \lim_{n\to\infty} \frac{\log n}{n} = 0,\ \lim_{n\to\infty} \frac{n}{10^n} = 0$$

$f(n)$ โตเร็วกว่า $g(n)$ $\quad iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

$f(n)$ โตด้วยอัตราเดียวกับ $g(n)$ $iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = c$

การเปรียบเทียบการโตของฟังก์ชัน

$$\lim_{n\to\infty} \frac{\log n}{n} = \lim_{n\to\infty} \frac{1/n}{1} = 0$$

$$\lim_{n\to\infty} \frac{n}{10^n} = \lim_{n\to\infty} \frac{1}{10^n \ln 10} = 0$$

เซตของฟังก์ชันที่โตช้ากว่า $g(n)$: โอเล็ก o

$$o(g(n)) = \{ f(n) \mid \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \}$$

$0.5^n \in o(1)$ เพราะ $\lim_{n\to\infty} \frac{0.5^n}{1} = 0$

$1 \in o(\log n)$ เพราะ $\lim_{n\to\infty} \frac{1}{\log n} = 0$

$\log n \in o(n)$ เพราะ $\lim_{n\to\infty} \frac{\log n}{n} = 0$

$n \in o(10^n)$ เพราะ $\lim_{n\to\infty} \frac{n}{10^n} = 0$

เปรียบเทียบฟังก์ชัน

$25\,x^3 - 563x^2 - 45x \; - 54$ กับ

$56327x^2 + 247845x + 54121$

$$\lim_{x\to\infty} \frac{56327x^2+247845x+54121}{25\,x^3-563x^2-45x\,-54}$$

$$= \lim_{x\to\infty} \frac{2*56327x+247845}{3*25\,x^2-2*563x-45}$$

$$= \lim_{x\to\infty} \frac{2*56327}{2*3*25\,x-2*563} = 0$$

เปรียบเทียบฟังก์ชัน

$545x$

$x \log x$

$$\lim_{x\to\infty} \frac{545x}{x \log x}$$

$$= \lim_{x\to\infty} \frac{545}{\log x}$$

$$= 545 \lim_{x\to\infty} \frac{1}{\log x}$$

$$= 545 * 0 = 0$$

เปรียบเทียบฟังก์ชัน

$x \log x$
$x^2 \log x$

$$\lim_{x\to\infty} \frac{x\ \log x}{x^2\ \log x} = \lim_{x\to\infty} \frac{1}{x} = 0$$

เปรียบเทียบฟังก์ชัน

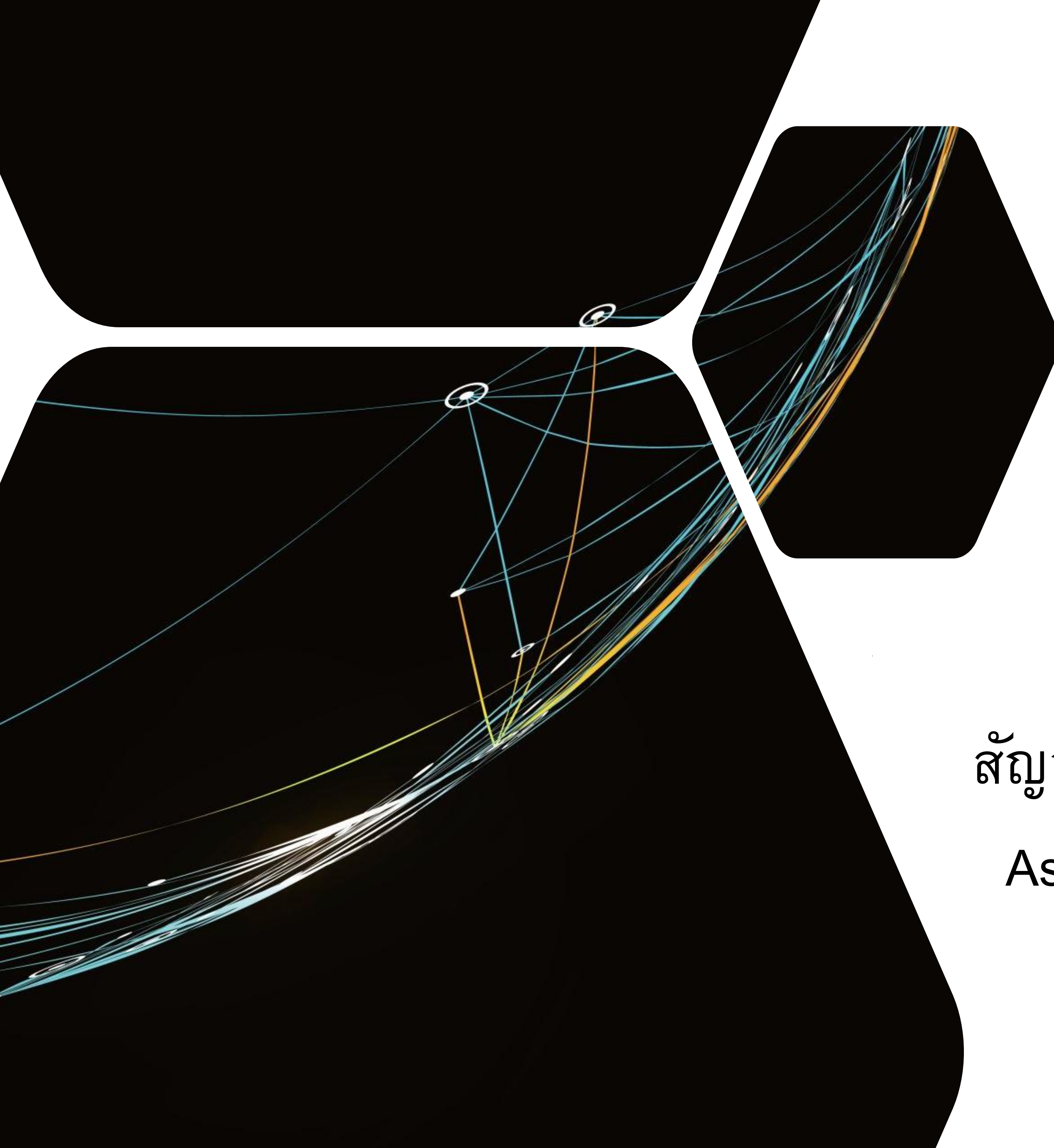
$0.5^n \ll 1 \ll \log n \ll n \ll n \log n \ll 10^n$

$\ldots \ll 1/n^3 \ll 1/n^2 \ll 1/n \ll n \ll n^2 \ll n^3 \ll \ldots$

เปรียบเทียบฟังก์ชัน

$$56327x^2 + 247845x + 54121$$
$$563x^2 - 45x + \log x$$

$$\lim_{x\to\infty} \frac{563x^2 - 45x + \log x}{56327\, x^2 + 247845x + 54121}$$
$$= \lim_{x\to\infty} \frac{2*563x - 45 + 1/x}{2*56327x + 247845}$$
$$= \lim_{x\to\infty} \frac{2*563x^2 - 45x + 1}{2*56327x^2 + 247845}$$
$$= \lim_{x\to\infty} \frac{2*2*563}{2*2*56327} = c$$



สัญกรณ์เชิงเส้นกำกับ

Asymptotic Notation

2301263

เซตของฟังก์ชันที่โตเร็วกว่า $g(n)$: โอเมกาเล็ก ω

$$\omega(g(n)) = \left\{ f(n) \;\middle|\; \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \right\}$$

$f(n) \in \omega\big(g(n)\big)$ ก็ต่อเมื่อ $g(n) \in o\big(f(n)\big)$

$1 \in \omega(0.5^n)$ เพราะ $0.5^n \in o(1)$

$\log n \in \omega(1)$ เพราะ $1 \in o(\log n)$

$n \in \omega(\log n)$ เพราะ $\log n \in o(n)$

$10^n \in \omega(n)$ เพราะ $n \in o(10^n)$

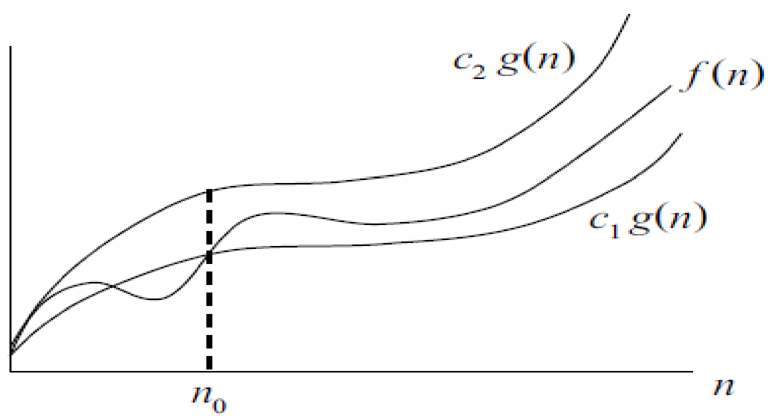
เซตของฟังก์ชันที่โตด้วยอัตราเดียวกับ $g(n)$:ทีตาใหญ่ Θ

$$\Theta(g(n)\,) = \left\{ f(n) \,\middle|\, \lim_{n\to\infty} \frac{f(n)}{g(n)} = c, c \neq 0, c \neq \infty \right\}$$

หรือ

$f(n)$ ซึ่งมีจำนวนเต็มบวก c_1, c_2 และ n_0 ที่ทำให้

$c_1\, g(n) \leq f(n) \leq c_2\, g(n)$ เมื่อ $n \geq n_0$

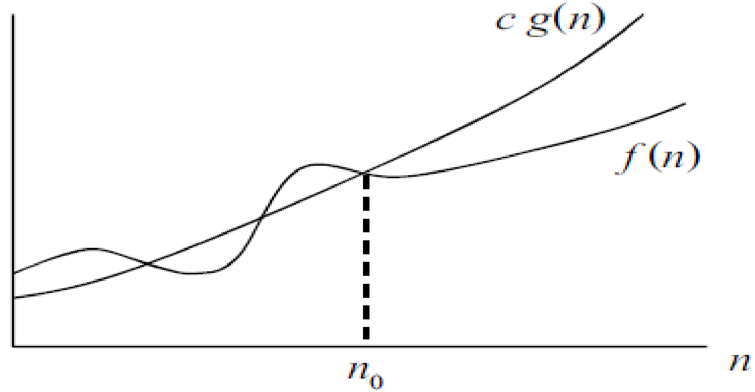


รูปที่ 3–1 $f(n) \in \Theta(g(n))$

เซตของฟังก์ชันที่โตไม่เร็วกว่า $g(n)$:โอใหญ่ O

$O(g(n)) =$

$\{\ f(n)$ | มีจำนวนเต็มบวก c และ n_0 ที่ทำให้ $f(n) \le c\ g(n)$ เมื่อ $n \ge n_0\ \}$



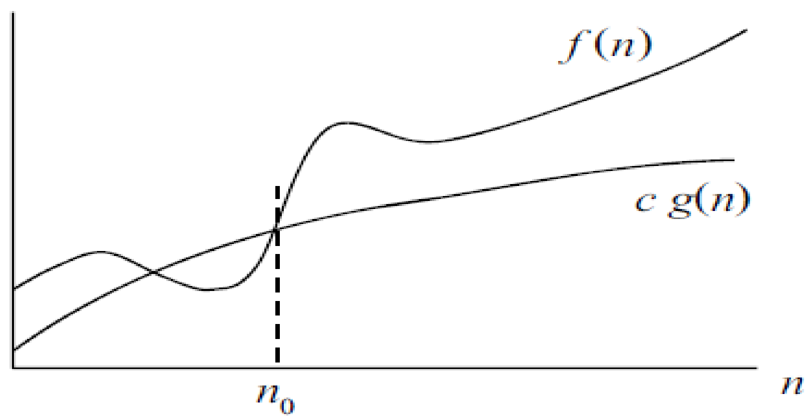
รูปที่ 3–2 $f(n) \in O(g(n))$

$O(g(n)) = o(g(n)) \cup \Theta(g(n))$

เซตของฟังก์ชันที่โตไม่ช้ากว่า $g(n)$:โอเมกาใหญ่ Ω

$\Omega(g(n))$ =

{ $f(n)$ | มีจำนวนเต็มบวก c และ n_0 ที่ทำให้ $c\,g(n) \le f(n)$ เมื่อ $n \ge n_0$ }



รูปที่ 3–3 $f(n) \in \Omega(g(n))$

$$\Omega(g(n)) = \omega(g(n)) \cup \Theta(g(n))$$

Analysis

การวิเคราะห์

การวิเคราะห์เชิงเส้นกำกับ

O(C*size)

```
public class ArrayCollection implements Collection
  ...
  public void remove(Object e) {
    int i = indexOf(e);
    if (i != -1) {
      elementData[i] = elementData[--size];
      elementData[size] = null;
    }
  }
  ...
  private int indexOf(Object e) {
    for (int i=0; i<size; i++)
      if (elementData[i].equals(e)) return i;
    return -1;
  }
  ...
}
```

1 ครั้ง

≤ C*size ครั้ง

การวิเคราะห์เชิงเส้นกำกับ

```
public void dummy1 {
  int c = 0;
  for (int i=1; i<n; i++)
    for (int j=0; j<m; j++)
      c = c+1;
}
```

m ครั้ง

n ครั้ง

mn ครั้ง, O(mn)

```
public void dummy2 {
  int c = 0;
  for (int i=1; i<n; i++)
    for (int j=0; j<i; j++)
      c = c+1;
}
```

i ครั้ง

n-1 ครั้ง

$n(n-1)/2 = n^2/2 - n/2$ ครั้ง

$O(n^2)$

การวิเคราะห์เชิงเส้นกำกับ

```
static int binarySearch (int[] data, int e) {
  int L = 0, R = data.length -1;
  while (L <= R) {
    int M = (L + R)/2;
    if (e == data[M]) return M;
    if (e > data[M])
      L = M+1;
    else
      R = M -1
  }
  return -1;
}
...
binarySearch(data,22);
```

index	0	1	2	3	4	5	6	7	8	9	10	11
data	-7	-4	0	3	4	10	21	23	24	27	30	31
	L					M						R
							L		M			R
							L M	R				
						R	L					

การวิเคราะห์เชิงเส้นกำกับ

```
static int binarySearch (int[] data, int e) {
  int L = 0, R = data.length -1;
  while (L <= R) {
    int M = (L + R)/2;
    if (e == data[M]) return M;
    if (e > data[M])
      L = M+1;
    else
      R = M -1
  }
  return -1;
}
```

data เป็น array ขนาด n, ให้ $n = 2^k$

while loop หยุดวนเมื่อเหลือข้อมูล 0 ตัว

คือ $k+2$ รอบ = $\log_2 n + 2$

$O(\log n)$

round	R-L+1
1	n
2	$n/2$
3	$n/2/2$
4	$n/2/2/2$
…	…
k	$n/2^{k-1}$
$k+1$	$n/2^k$
$k+2$	$n/2^{k+1}$