演算法 效能分析(遞迴)

- 遞迴結構與遞迴公式
- 取代方法
- 支配理論

範例 3-1 計算階乘(n!)

疊代法 (iterative) 計算階乘　$n! = 1 \times 2 \times 3 \ldots \times n$

```
int factorial(int n) {
    int i, num=1;
    if(n > 1)
        for(i = 2; i <= n; i++)
            num = num * i;
    return num;
}
```

一個迴圈, 時間複雜度為 $\Theta(n)$
空間複雜度為 $\Theta(1)$

遞迴函數

- 遞迴函數 (recursive function)
  - 函數呼叫函數本身
  - 使用堆疊：
    返回位址、輸入參數、區域變數

```
int func (int n, ...) {
    int k;
    k=func(n, ...);
    return k;
}
```

範例 3-2 計算階乘 (n!)

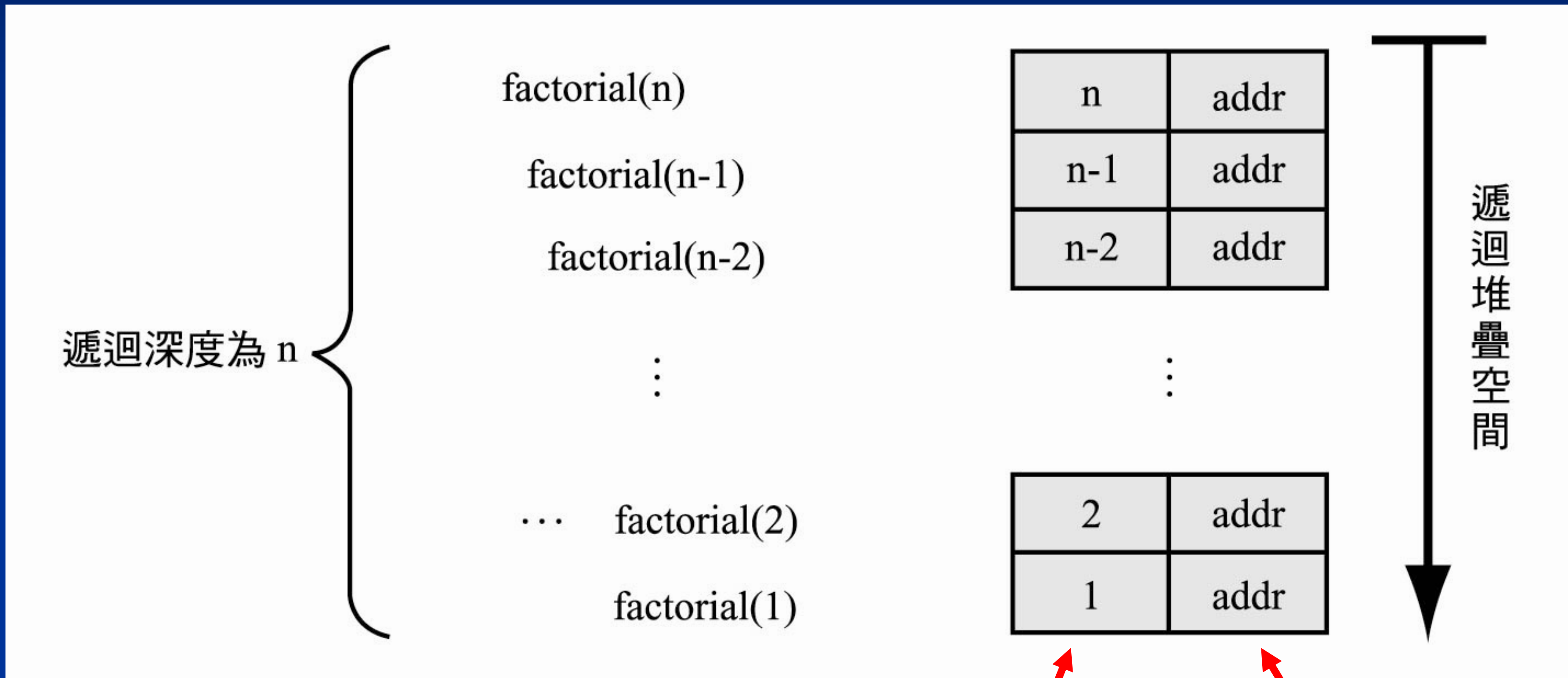
遞迴法 (recursive) 計算階乘

$$n! = \begin{cases} n(n-1)! & if \quad n \geq 1 \\ 1 & if \quad n = 0 \end{cases}$$

```
int factorial (int n) {
    if (n == 0)  return 1;
    else         return n*factorial(n-1) ;
}
```

```
ALGORITHM F(n)
        if n=0 return 1
        else return n*F(n-1)
```

遞迴函數展開之層次



執行 n 次函數
計算時間 $T(n)=Bn$
時間複雜度爲 $\Theta(n)$

輸入參數 回傳位址
使用堆疊空間爲 $S(n)=Bn$
空間複雜度爲 $\Theta(n)$

範例 3-3 最大公因數 (15.2.1)

```
int gcd(int u, int v) {
   int t;
   if (v==0) t=u;
   else t=gcd(v, u%v);
   return t;
}
```

u=382 v=328 u%v=54
u=328 v=54 u%v=4
u=54 v=4 u%v=2
u=4 v=2 u%v=0
u=2 v=0

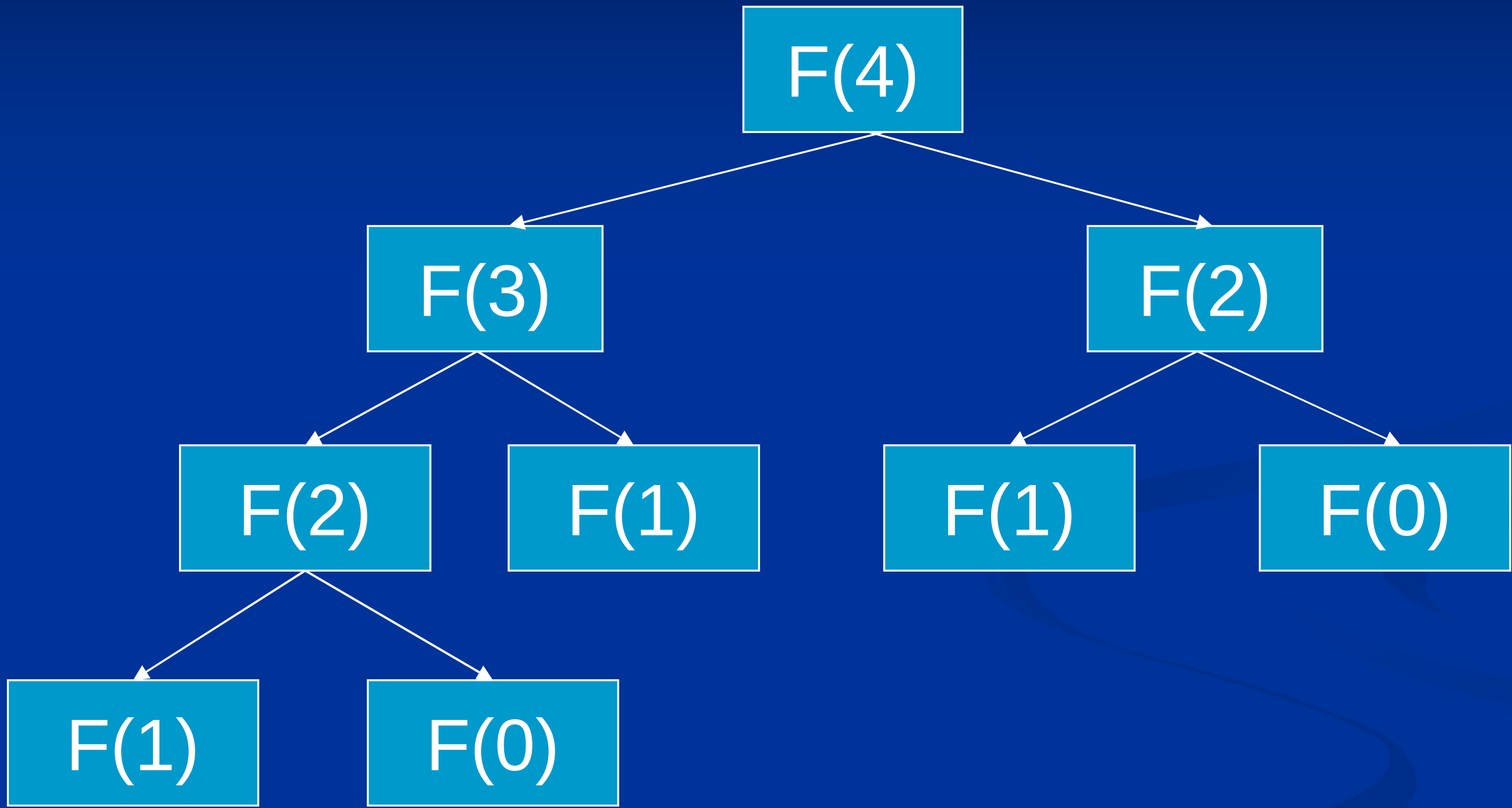
範例 3-4 Fibonacci number

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89

$$F(n) = \begin{cases} 0 & & n = 0 \\ 1 & if & n = 1 \\ F(n-1) + F(n-2) & & n > 1 \end{cases}$$

```
int fib(n){
        if (n==0) return 0;
        else if (n==1) return 1;
        else return fib(n-1)+fib(n-2);
}
```

Fibonacci number



看起來約需要 2^n 次遞迴?

Fibonacci number

- 計算時間是遞迴關係

  $T(n)=T(n-1)+T(n-2)$ $T(n)>0$

  猜測 $T(n)=Ka^n$ 則 $a^n=a^{n-1}+a^{n-2}$

  $a^2=a+1$

$$a=\frac{1+\sqrt{5}}{2}\approx 1.618$$

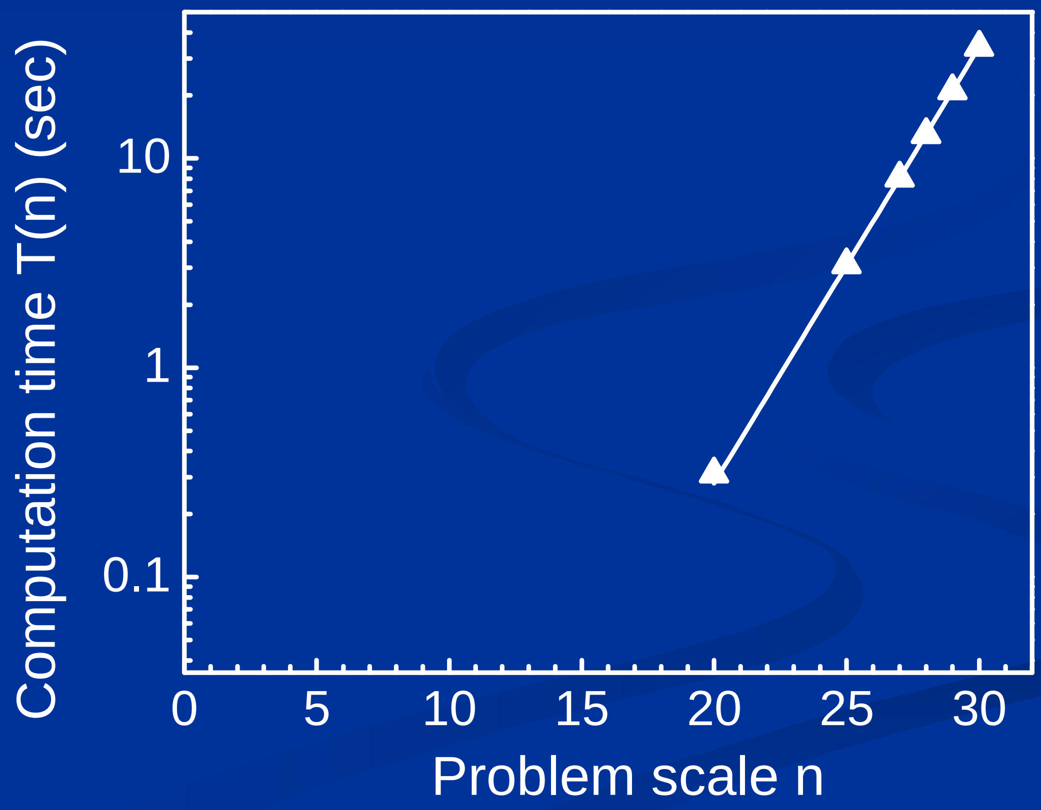
- 時間複雜度 $\Theta(a^n)$ 空間複雜度 $\Theta(a^n)$

範例 3-5 Fibonacci number

- 遞迴計算效能

$T(n)=Ka^n$

n=20　0.311s
n=25　3.108s
n=27　8.061s
n=28　13.03 s
n=29　21.06 s
n=30　34.03 s

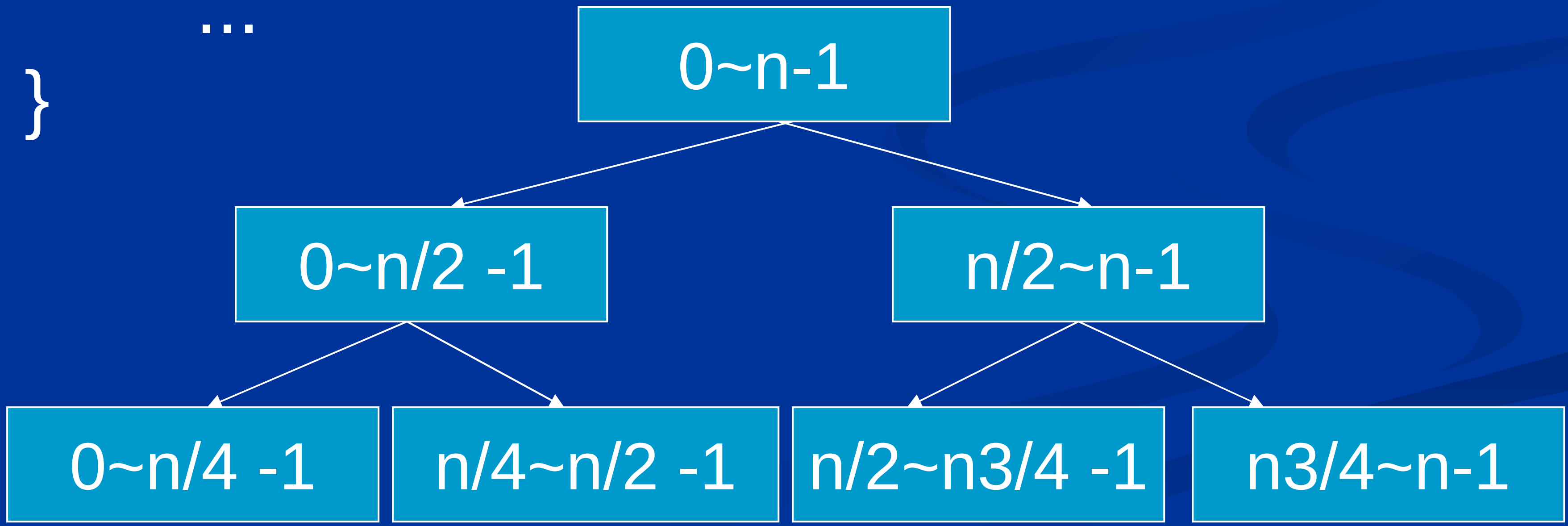


遞迴與 Divide and Conquer

- 每一層遞迴中包含的步驟
  - 將問題分割 (divide) 成數個子問題
  - 重複處理 (conquer) 這些子問題，如果子問題夠小，就用直接計算的方式來解決子問題
  - 將子問題的解答合併 (combine) 成爲原來問題的解答

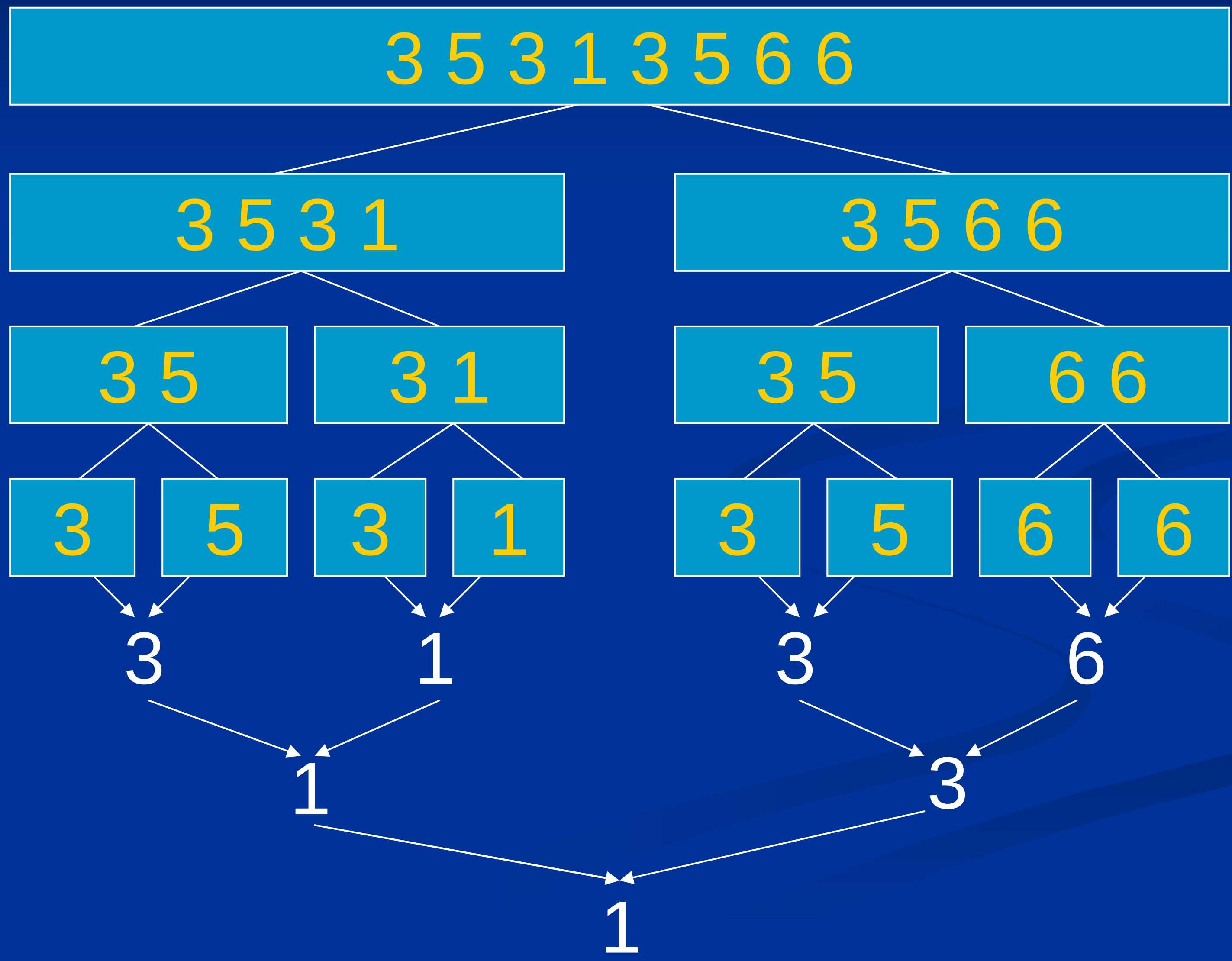
Divide and Conquer

```
void func(int A[], int p, int r){
        q=(p+r)/2
        func(A, p, q);
        func(A, q+1, r);
        ...
}
```



範例 3-6 遞迴尋找最小值

n=8



遞迴尋找最小值

```
int min( int A[], int p, int r){
        int a, b, q;
        if (p==r)
          return A[p];
        else {
                q=(p+r)/2;
                a=min(A, p, q);
                b=min(A, q+1, r);
                return (a<b)?a:b;
        }
}
```

時間複雜度

- 計算時間 T(n)=2T(n/2)+K　T(1)=C

  K 配置堆疊, 比較, 回傳　C 配置堆疊, 回傳

  T(n)= 2(2 T(n/4) + K) + K

  = 2(2 (2 T(n/8) + K) + K) + K

$$= 2^m C + K(1 + 2 + \ldots + 2^{m-1})$$

$n/2^m=1$

$n=2^m$

$m=\log_2 n$

$$= nC + K\frac{2^m - 1}{2-1}$$

$$= nC + K(n-1)$$

$$T(n) \in \Theta(n)$$

效能

- 效能測試

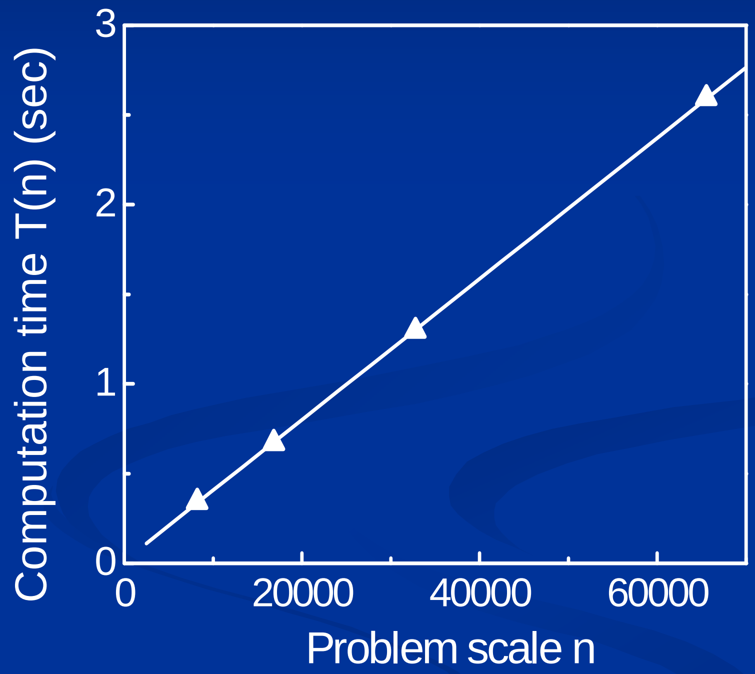
(執行1000次演算法)

n=8192 0.343 s

n=16384 0.671 s

n=32768 1.296 s

n=65536 2.593 s



約為 T(n)=Bn 時間複雜度為 Θ(n)

有關公式

$$\frac{1}{1-x} = 1 + x + x^2 + \ldots \quad \textit{if } 0 \le x < 1$$

$$\log_b n = \frac{\log n}{\log b} \qquad \log_2 n \in \Theta(\log n)$$

$$c^{\log_b n} = n^{\log_b c}$$

遞迴公式

- 遞迴公式一個方程式或是不等式，可用來描述一個函數與其數值之間的關係

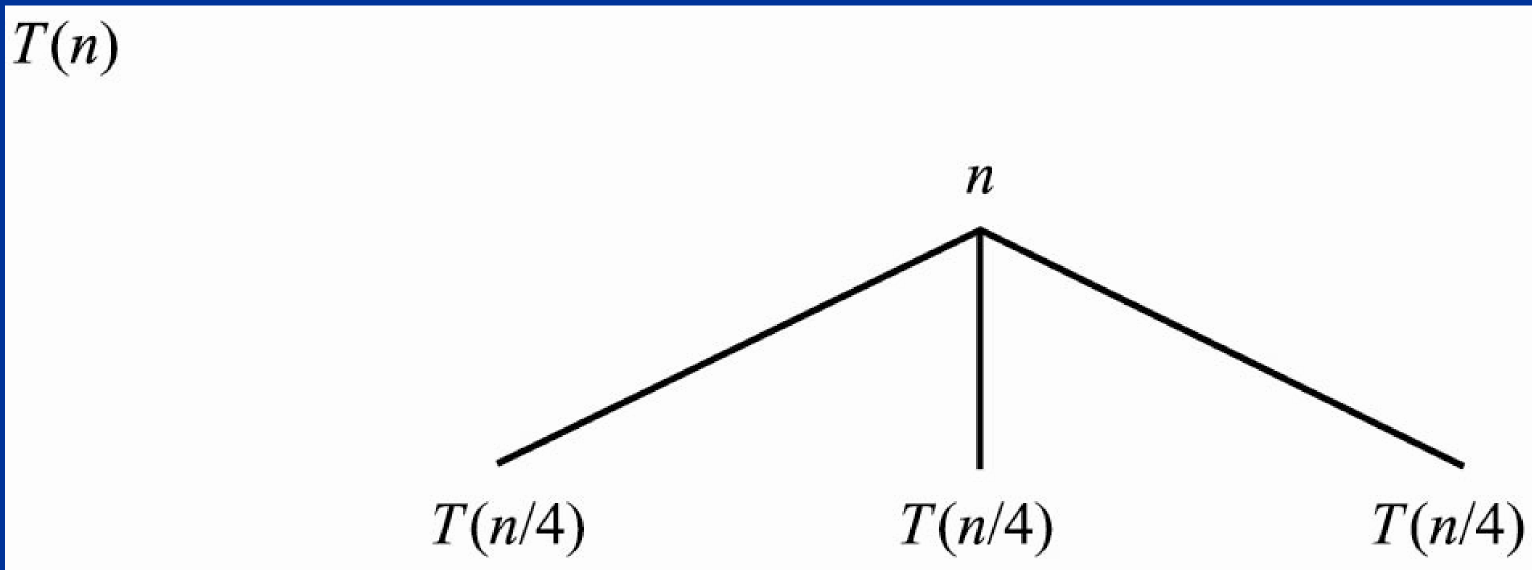
範例

$T(n) = aT(n/b) + f(n)$ ，$a \geqq 1$、$b>1$

$f(n)$ 是指定函數，如常數 、$\log n$、n 、n^2等

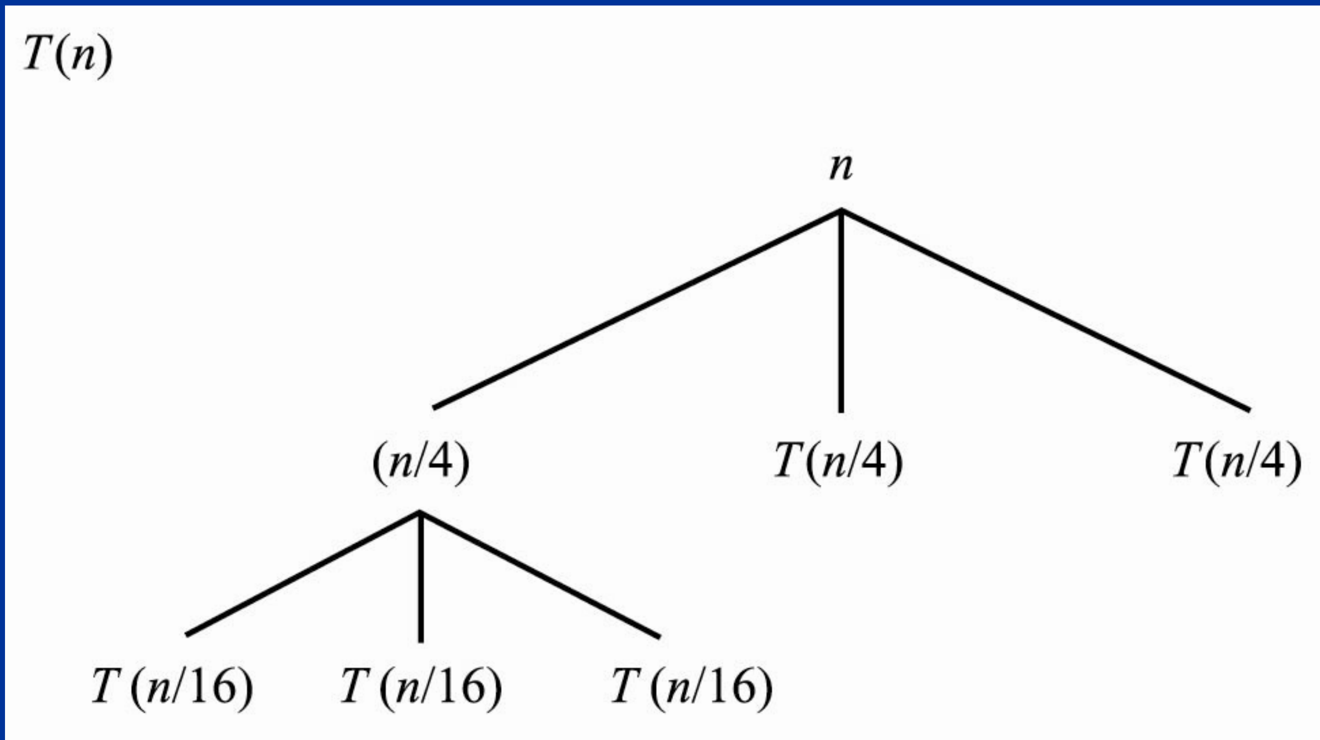
第一次遞迴

$$T(n) = \begin{cases} 3T(n/4) + n & n \geq 2 \\ 1 & n = 1 \end{cases}$$

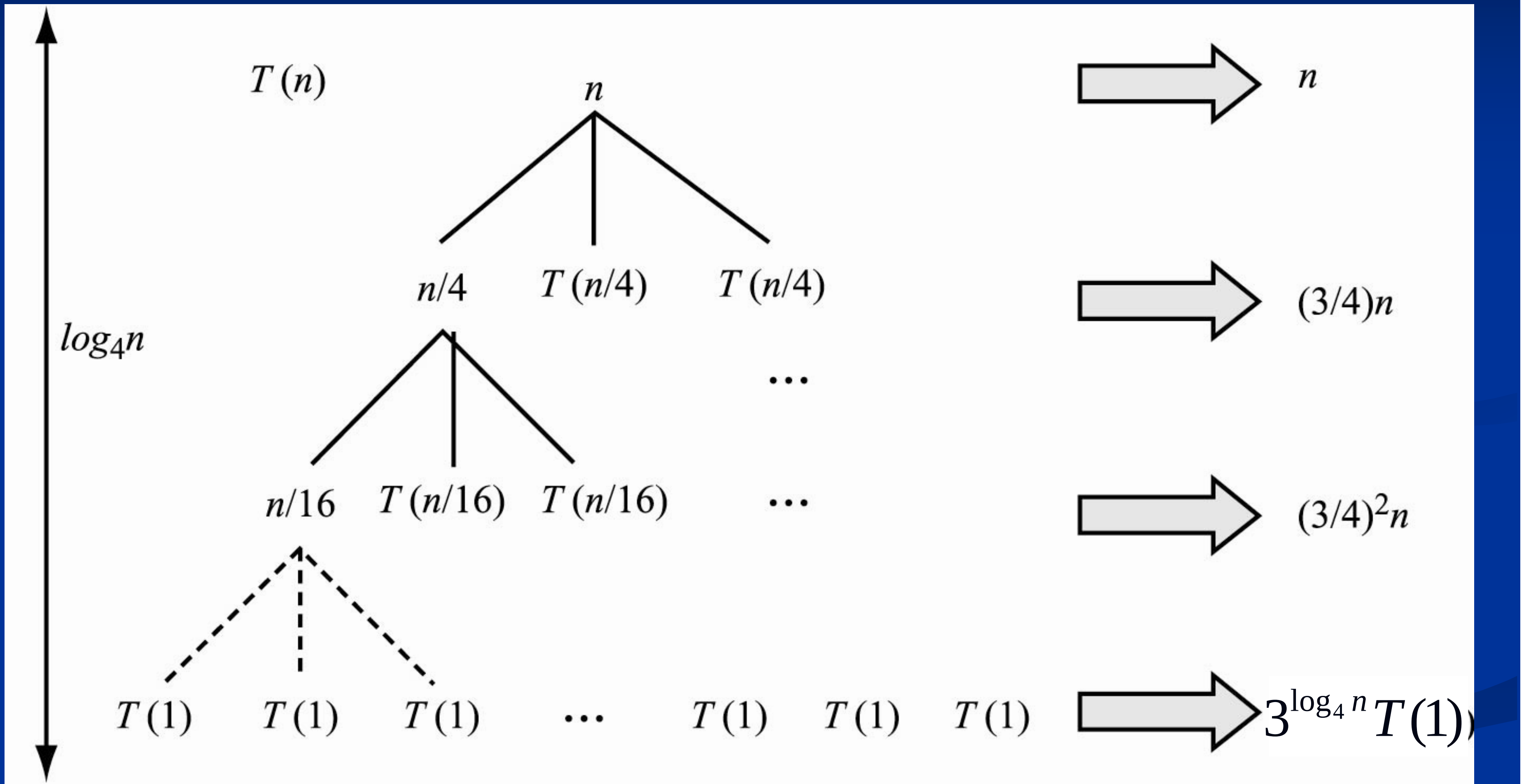


第二次遞迴

$$T(n) = \begin{cases} 3T(n/4) + n & n \geq 2 \\ 1 & n = 1 \end{cases}$$



取代方法



T(n)
n
n/4
T(n/4)
T(n/4)
log4n
…
n/16
T(n/16)
T(n/16)
…
T(1)
T(1)
T(1)
…
T(1)
T(1)
T(1)
n
(3/4)n
(3/4)2n
3log4 n T(1)

取代方法

$$T(n)=\begin{cases} 3T(n/4)+n & n\ge 2 \\ 1 & n=1 \end{cases}$$

- 假設$n=4^k$，k是一個整數，展開遞迴式：

$$\begin{aligned}T(n) &= n+3T(n/4)\\ &= n+3(n/4+3T(n/16))\\ &= n+3(n/4+3(n/16+3T(n/64)))\\ &= n+3n/4+9n/16+27n/64+\cdots+\underline{3^{\log_4 n}T(1)}\\ &= 4n+\underline{n^{\log_4 3}}\end{aligned}$$

$$T(n)\in\Theta(n)$$

取代方法

$$T(n)=\begin{cases} 4T(n/4)+n & n\geq 2 \\ 1 & n=1 \end{cases}$$

- 假設$n=4^k$，k是一個整數，展開遞迴式：

$$\begin{aligned} T(n) &= n+4T(n/4) \\ &= n+4(n/4+4T(n/16)) \\ &= n+4(n/4+4(n/16+4T(n/64))) \\ &= n+4n/4+16n/16+64n/64+\cdots+\underline{4^{\log_4 n}T(1)} \\ &= n\log_4 n+\underline{n} \end{aligned}$$

$$T(n)\in\Theta(n\log n)$$

取代方法

$$T(n) = \begin{cases} 5T(n/4) + n & n \geq 2 \\ 1 & n = 1 \end{cases}$$

- 假設n=4^k，k是一個整數，展開遞迴式：

$$\begin{aligned} T(n) &= n + 5T(n/4) \\ &= n + 5(n/4 + 5T(n/16)) \\ &= n + 5(n/4 + 5(n/16 + 5T(n/64))) \\ &= n + 5n/4 + 25n/16 + 125n/64 + \cdots + \underline{5^{\log_4 n} T(1)} \\ &= Bn + \underline{n^{\log_4 5}} \end{aligned}$$

$$T(n) \in \Theta(n^{\log_4 5})$$

General divide-and-conquer recurrence

$$T(n) = \begin{cases} aT(n/b) + f(n) & n \geq 2 \\ C & n = 1 \end{cases}$$

- 假設 $n=b^k$，其中k是一個整數

$$T(n) = \underline{Cn^{\log_b a}} + \sum_{j=0}^{(\log_b n)-1} a^j f(n/b^j)$$

所有最後遞迴的計算時間　　　　所有遞迴的額外計算時間

支配理論 Master theorem

如果 $f(n) \in \Theta(n^d) \quad d \geq 0$

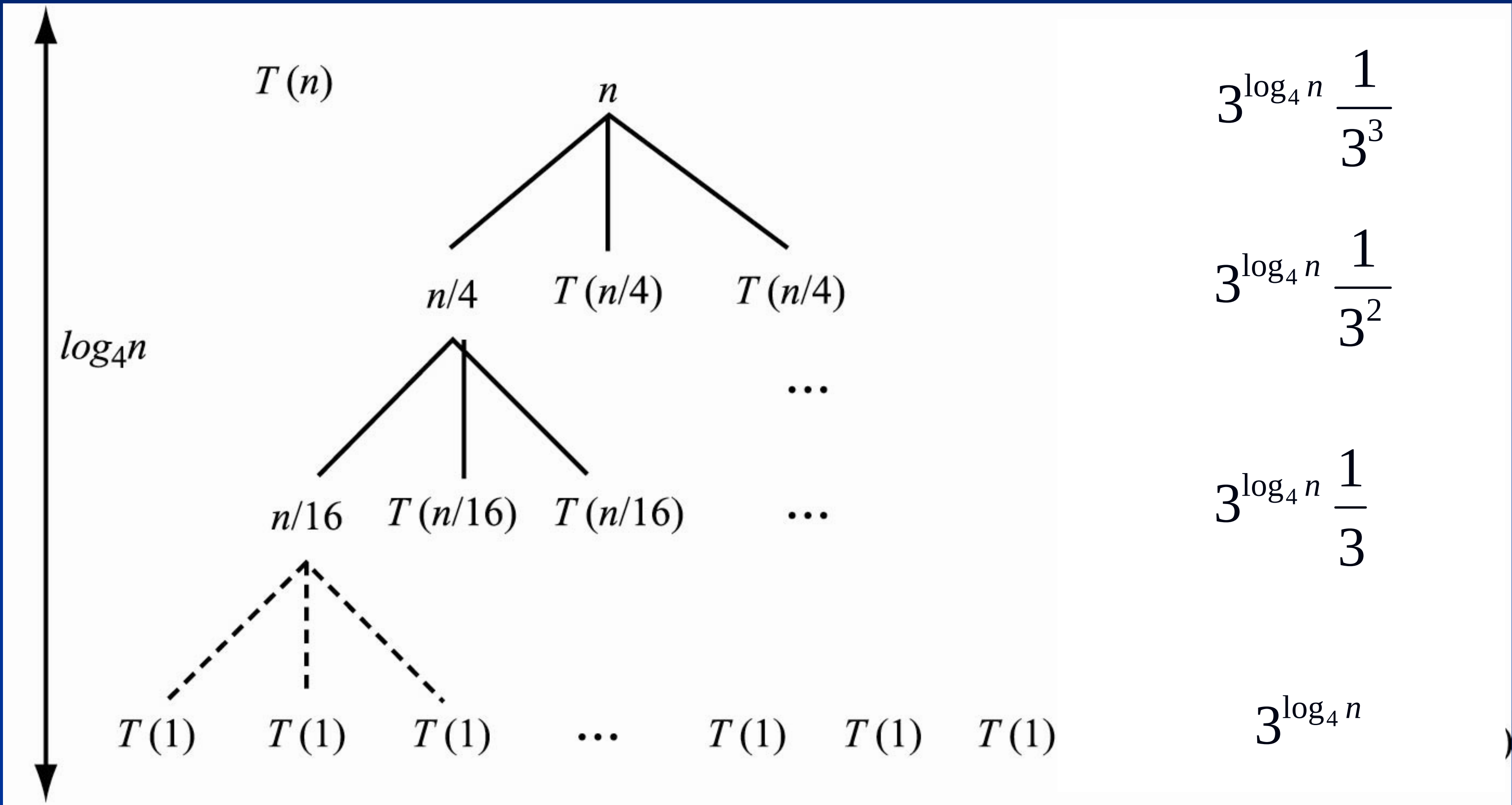
$$T(n) \in \begin{cases} \Theta(n^d) & a < b^d \\ \Theta(n^d \log n) & if\ a = b^d \\ \Theta(n^{\log_b a}) & a > b^d \end{cases}$$

$$T(n) \in \Theta(n^d) \quad \Rightarrow \quad T(n) \in O(n^d)$$

支配理論

- T(n) = aT(n/b) + cn
  - T(n) ∈ Θ(n) , if a < b
    分配了 b 份，遞迴處理少於 b 次
  - T(n) ∈ Θ(n*log*n) , if a = b
    分配了 b 份，遞迴處理等於 b 次
  - T(n) ∈ $\Theta\left(n^{\log_b a}\right)$, if a > b
    分配了 b 份，遞迴處理多於 b 次

遞迴堆疊空間



T (n)
n
n/4
T (n/4)
T (n/4)
log₄n
n/16
T (n/16)
T (n/16)
T (1)
T (1)
T (1)
T (1)
T (1)
T (1)
$3^{\log_4 n}\frac{1}{3^3}$
$3^{\log_4 n}\frac{1}{3^2}$
$3^{\log_4 n}\frac{1}{3}$
$3^{\log_4 n}$

支配理論

- $T(n) = aT(n/b) + cn^2$
  - $T(n) \in \Theta(n^2)$, if $a < b^2$

    分配了 b 份，遞迴處理少於 b^2 次

  - $T(n) \in \Theta(n^2 \log n)$, if $a = b^2$

    分配了 b 份，遞迴處理等於 b^2 次

  - $T(n) \in \Theta\left(n^{\log_b a}\right)$, if $a > b^2$

    分配了 b 份，遞迴處理多於 b^2 次

取代方法

15<4²

$$T(n)=\begin{cases} 15T(n/4)+n^2 & n\geq 2 \\ 1 & n=1 \end{cases}$$

- 假設n=4^k，k是一個整數，展開遞迴式：

$$\begin{aligned} T(n) &= n^2+15T(n/4) \\ &= n^2+15((n/4)^2+15T(n/16)) \\ &= n^2+15((n/4)^2+15((n/16)^2+15T(n/64))) \\ &= n^2+15n^2/4^2+15^2n^2/16^2+\cdots+\underline{15^{\log_4 n}T(1)} \\ &= 16n^2+\underline{n^{\log_4 15}} \end{aligned}$$

$$T(n)\in\Theta(n^2)$$

取代方法

16=4²

$$T(n)=\begin{cases}16T(n/4)+n^2 & n\geq 2\\ 1 & n=1\end{cases}$$

- 假設$n=4^k$，k是一個整數，展開遞迴式：

$$\begin{aligned}T(n)&=n^2+16T(n/4)\\&=n^2+16((n/4)^2+16T(n/16))\\&=n^2+16((n/4)^2+16((n/16)^2+16T(n/64)))\\&=n^2+16n^2/4^2+16^2n^2/16^2+\cdots+\underline{16^{\log_4 n}T(1)}\\&=n^2\log_4 n+\underline{2n^2}\end{aligned}$$

$$T(n)\in\Theta(n^2\log n)$$

取代方法

17>4^2

$$T(n) = \begin{cases} 17T(n/4) + n^2 & n \geq 2 \\ 1 & n = 1 \end{cases}$$

- 假設$n=4^k$，k是一個整數，展開遞迴式：

$$\begin{aligned} T(n) &= n^2 + 17T(n/4) \\ &= n^2 + 17((n/4)^2 + 17T(n/16)) \\ &= n^2 + 17((n/4)^2 + 17((n/16)^2 + 17T(n/64))) \\ &= n^2 + 17n^2/4^2 + 17^2 n^2/16^2 + \cdots + \underline{17^{\log_4 n} T(1)} \\ &= Bn + \underline{n^{\log_4 17}} \end{aligned}$$

$$T(n) \in \Theta(n^{\log_4 17})$$

空間複雜度

- 堆疊空間 S(n)=2T(n/2)+K T(1)=K

  K 配置堆疊

  S(n)= 2(2 T(n/4) + K) + K

  = 2(2 (2 T(n/8) + K) + K) + K

$$= 2^m K + K(1+2+\ldots+2^{m-1})$$

n/2^m=1

n=2^m

m=log$_2$n

$$= nK + K\frac{2^m - 1}{2-1}$$

$$= nK + K(n-1)$$
$$= (2n-1)K$$

$$S(n) \in \Theta(n)$$

遞迴堆疊空間

- 使用 Divide and conquer 時, 假設每次遞迴消耗堆疊空間是固定的, 則S(n)正比於遞迴總數.

a<b

$$S(n) = 3^{\log_4 n}\left(1+\frac{1}{3}+\frac{1}{3^2}+\ldots\right)C$$

$$\approx Cn^{\log_4 3}\frac{1}{1-\frac{1}{3}} = \frac{3}{2}Cn^{\log_4 3}$$

$$\le \frac{3}{2}Cn$$

$$S(n) \in O(n)$$

遞迴堆疊空間

a=b

$$S(n) = 4^{\log_4 n}\left(1 + \frac{1}{4} + \frac{1}{4^2} + \ldots\right)C$$

$$\approx Cn\frac{1}{1 - \frac{1}{4}} = \frac{4}{3}Cn$$

$$S(n) \in \Theta(n)$$