

Dynamic Programming III

Michael Tsai

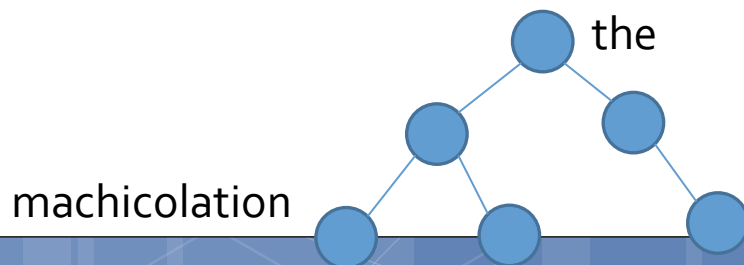
2012/10/18



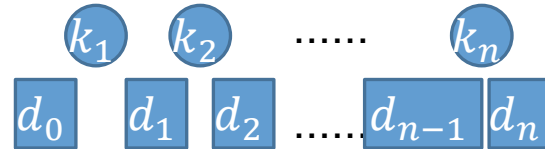
翻譯機問題



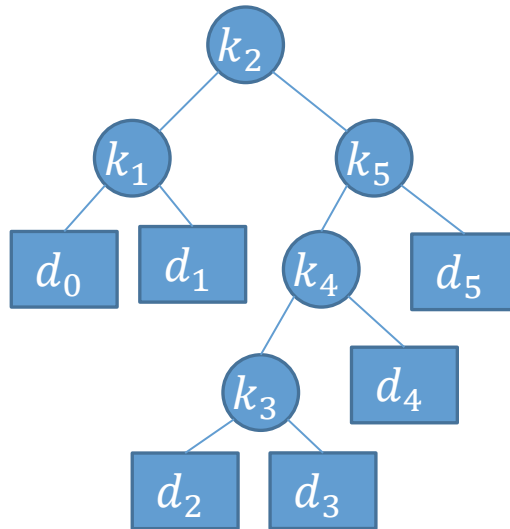
- 最笨翻譯機：
每個英文單字直接翻成法文單字
- 做法：建一棵balanced binary search tree (例如紅黑樹)，裡面用英文單字當key, 法文單字當作對應的資料
- 則每個字平均花 $O(\log n)$ 的時間
- 假設我們知道每個字出現的頻率(或機率), 可以做得更好嗎?
- 答：可以！把常用的字放離root近一點。



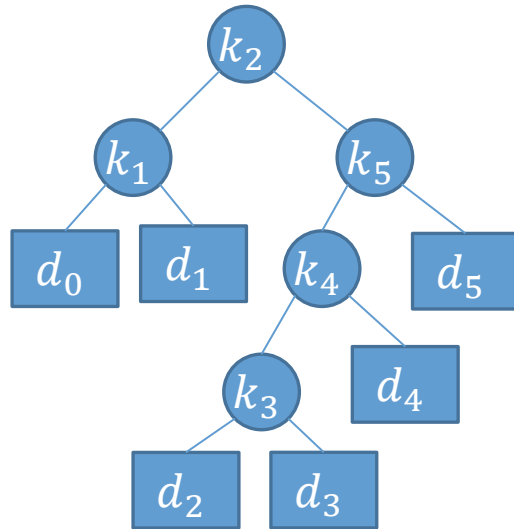
翻譯機問題 \rightarrow Optimal Binary Search Tree



- 問題:
 - 給一個序列 $K = \langle k_1, k_2, \dots, k_n \rangle$ 共 n 個排好序的 key ($k_1 < k_2 < \dots < k_n$). 我們要用這些 key 建立一棵 binary search tree.
 - k_i 出現的機率為 p_i .
 - 另外我們也有 $n+1$ 個“假key”代表沒有出現在 K 中的值, 可用 $d_0, d_1, d_2, \dots, d_n$ 來表示. d_0 代表小於 k_1 的值, d_1 代表介於 k_1 和 k_2 的值, \dots, d_n 代表大於 k_n 的值.
 - 假key d_i 出現的機率為 q_i .
 - 則目標是找出一棵 binary search tree 使得 Expected cost 最小.



- $E[\text{search cost}] =$
 $\sum_{i=1}^n (\text{depth}_T(k_i) + 1) \cdot p_i + \sum_{i=0}^n (\text{depth}_T(d_i) + 1) \cdot q_i$
 $= 1 + \sum_{i=1}^n \text{depth}_T(k_i) \cdot p_i + \sum_{i=0}^n \text{depth}_T(d_i) \cdot q_i$
- since $\sum_{i=1}^n p_i + \sum_{i=0}^n q_i = 1$
- 使得以上 $E[\text{search cost}]$ 最小的binary search tree稱為 optimal binary search tree.



i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

- 假設給定的key的出現機率為右上表格所顯示, 則左上圖為optimal binary search tree (expected cost=2.75)
- 觀察: 機率最大的key不見得在root (k_5 不在root)

暴力法有多暴力?

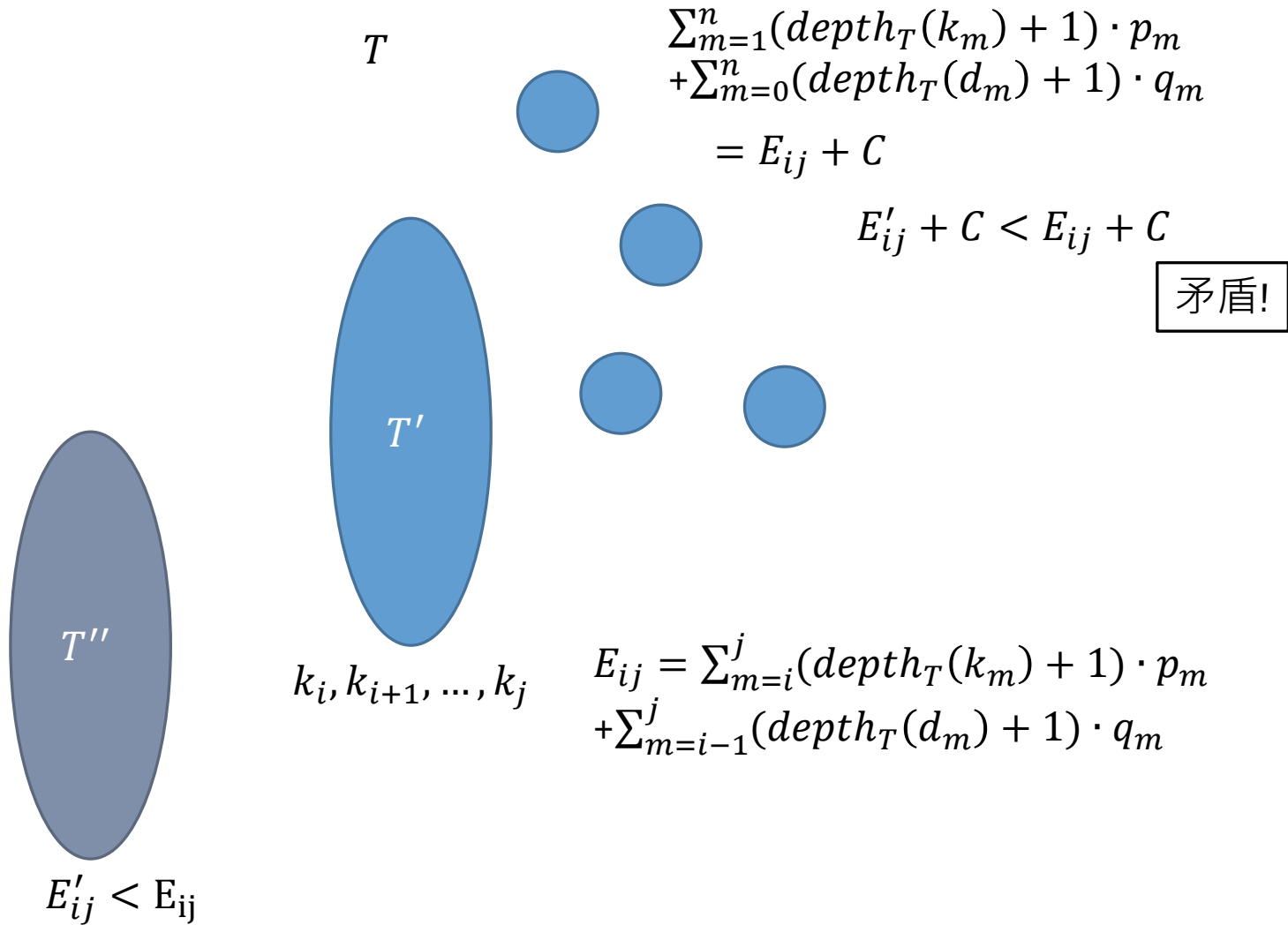
- 上學期也講過(?)...跟上次同樣的
- n個node的binary search tree總共有 $\Omega\left(\frac{4^n}{n^2}\right)$ 個
(Catalan number)



Dynamic Programming出招

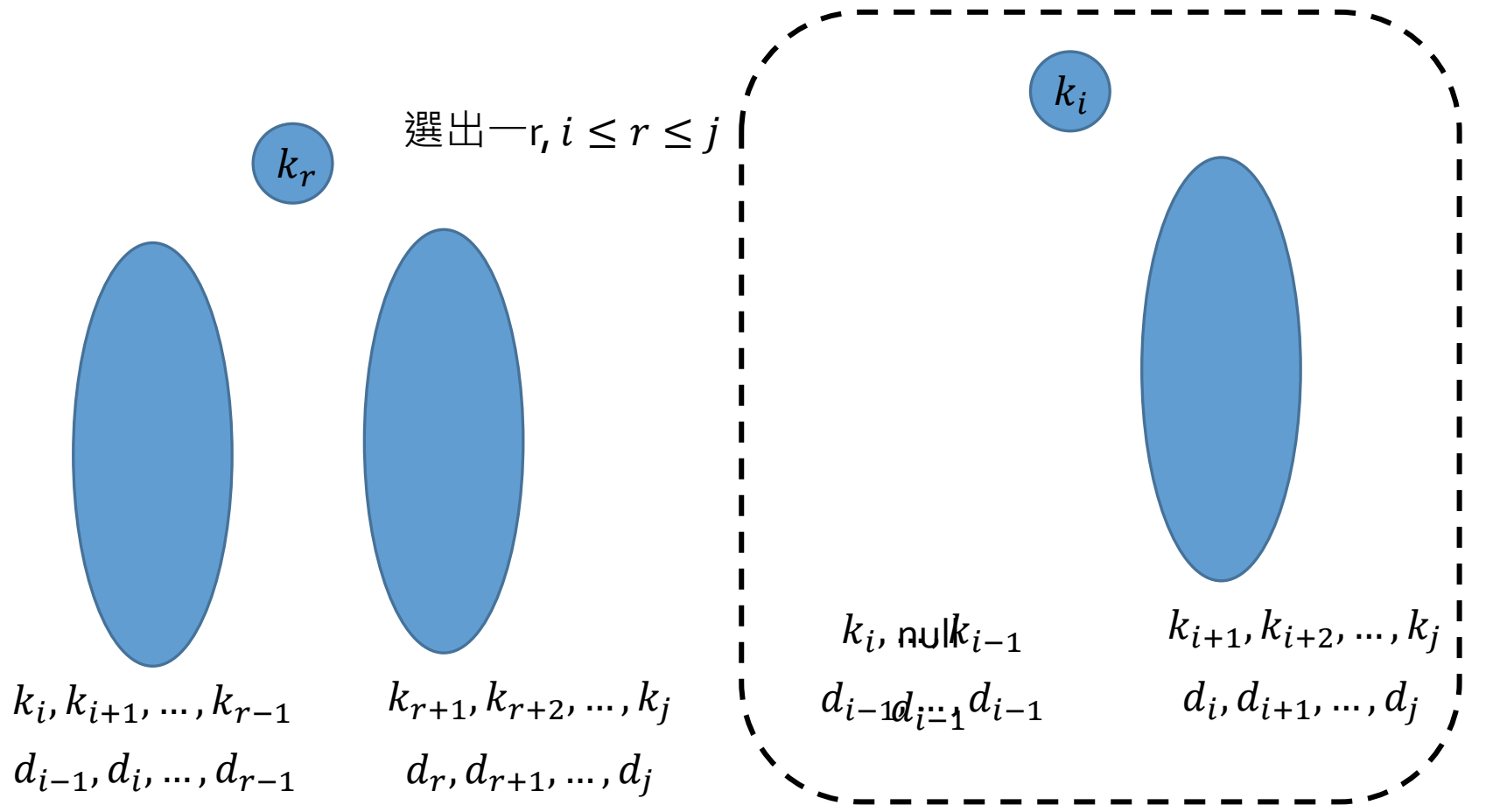
1. 找出Optimal Substructure

- 小觀察: binary search tree的subtree必包含一段連續的key k_i, k_{i+1}, \dots, k_j 及 d_{i-1}, \dots, d_j , $1 \leq i \leq j \leq n$.
- 小定理: 假設 T 為 $K = \langle k_1, k_2, \dots, k_n \rangle$ 之optimal binary search tree. 則 T 之subtree T' , 包含 k_i, \dots, k_j 這些key, 也必定是 $K' = \langle k_i, k_{i+1}, \dots, k_j \rangle$ 這些key的optimal binary search tree.
- 證明: 如果 T' 找出的不是optimal binary search tree, 則表示可以找出一個更好的binary search tree T'' , expected cost比 T' 更好, 則可以用 T'' 取代 T 中的 T' , 得到一個比 T cost更低的binary search tree (矛盾)



Dynamic Programming 出招

- 如何用小問題的答案組出大問題的答案?



Dynamic Programming 出招

2. 列出遞迴式子 (表示花費)

$$\begin{aligned}
 & e[i, r-1] \\
 &= \sum_{m=i}^{r-1} (\text{depth}_{T_L}(k_m) + 1) \cdot p_m \\
 &+ \sum_{m=i-1}^{r-1} (\text{depth}_{T_L}(d_m) + 1) \cdot q_m
 \end{aligned}
 \quad
 \begin{array}{c}
 T \\
 \circlearrowleft k_r \\
 \begin{array}{c}
 \text{---} T_L \text{---} \\
 \text{---} T_R \text{---}
 \end{array}
 \end{array}
 \quad
 \begin{aligned}
 & e[r+1, j] \\
 &= \sum_{m=r+1}^j (\text{depth}_{T_R}(k_m) + 1) \cdot p_m \\
 &+ \sum_{m=r+1}^j (\text{depth}_{T_R}(d_m) + 1) \cdot q_m
 \end{aligned}$$

$k_i, k_{i+1}, \dots, k_{r-1}$ $k_{r+1}, k_{r+2}, \dots, k_j$
 $d_{i-1}, d_i, \dots, d_{r-1}$ d_r, d_{r+1}, \dots, d_j

$$\text{depth}_T(\cdot) = \text{depth}_{T_L}(\cdot) + 1 = \text{depth}_{T_R}(\cdot) + 1$$

- $w(i, j) = \sum_{l=i}^j p_l + \sum_{l=i-1}^j q_l$

包含 k_i, \dots, k_j 的 subtree
所發生的機率

- $e[i, j] = p_r + (e[i, r-1] + w(i, r-1)) + (e[r+1, j] + w(r+1, j))$

條件不同, 使用的 subproblem 不同

- $$e[i, j] = \begin{cases} q_{i-1} & , \text{if } j = i - 1 \\ \min_{i \leq r \leq j} \{e[i, r-1] + e[r+1, j] + w(i, j)\} & , \text{if } j \neq i - 1 \end{cases}$$

r 有多種選擇

Dynamic Programming 出招

3. 計算花費

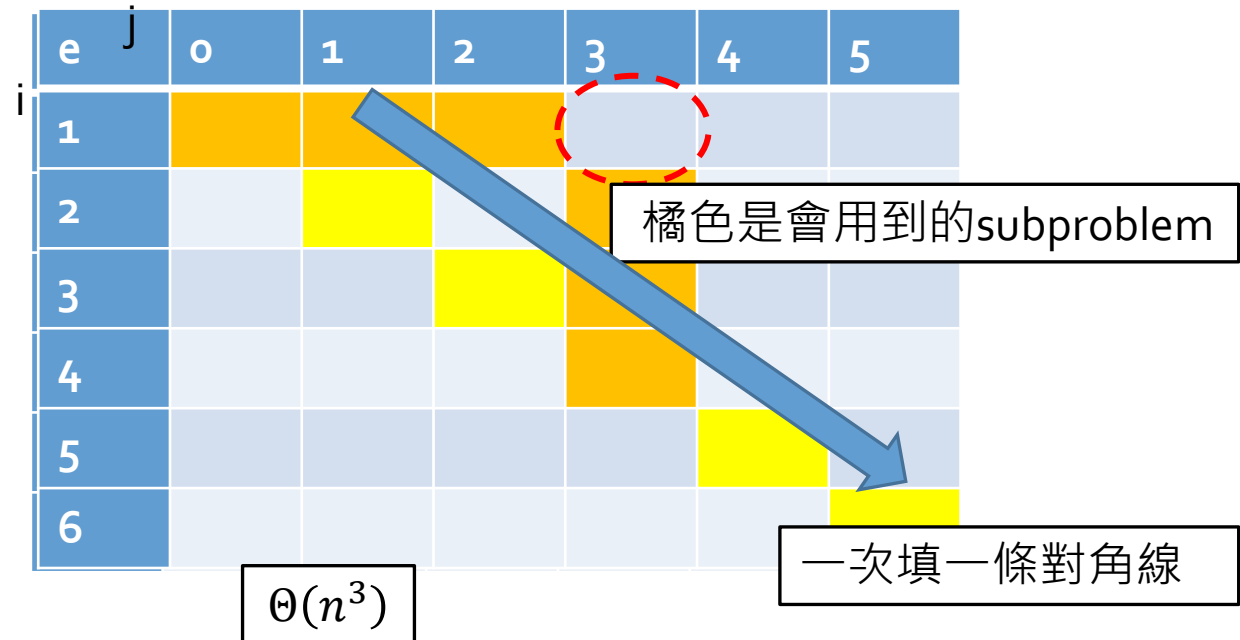
- 填表: e & w
 - $e[i,j]$: $i=1$ to $n+1$, $j=0$ to n
 - $w[i,j]$: $i=1$ to $n+1$, $j=0$ to n
- 為什麼 w 要填表? 不然計算每個 $e[i,j]$ 都需要做 $\Theta(j - i)$ 次加法

$$w[i,j] = \begin{cases} q_{i-1} & , \text{if } j = i - 1 \\ w[i,j - 1] + p_j + q_j & , \text{if } j \neq i - 1 \end{cases}$$

	j	0	1	2	3	4	5
w							
1							
2							
3							
4							
5							
6							

 $\Theta(n^2)$

○ e的填表順序



大家來練習

i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

$$w[i, j] = \begin{cases} q_{i-1} & , \text{if } j = i - 1 \\ w[i, j - 1] + p_j + q_j & , \text{if } j \neq i - 1 \end{cases}$$

$$e[i, j] = \begin{cases} q_{i-1} & , \text{if } j = i - 1 \\ \min_{i \leq r \leq j} \{e[i, r - 1] + e[r + 1, j] + w(i, j)\} & , \text{if } j \neq i - 1 \end{cases}$$

```
Optimal_BST(p,q,n)
```

```
let e[1..n+1,0..n],w[1..n+1,0..n], and  
root[1..n,1..n] be new tables
```

```
for i=1 to n+1
```

```
    e[i,i-1]=qi-1
```

邊界起始值

```
    w[i,i-1]=qi-1
```

```
for l=1 to n
```

```
    for i=1 to n-l+1
```

填表: 兩層迴圈, 對角線順序

```
        j=i+l-1
```

```
        e[i,j]=∞
```

```
        w[i,j]=w[i,j-1]+pj+qj
```

```
        for r=i to j
```

```
            t=e[i,r-1]+e[r+1,j]+w[i,j]
```

```
            if t<e[i,j]
```

```
                e[i,j]=t
```

```
                root[i,j]=r
```

```
return e and root
```

$\Theta(n^3)$

Dynamic Programming出招

4. 印出Optimal Binary Search Tree結果

- 自己回家研究一下😊 15.5-1 on p. 403