演算法設計方法論 (Design Strategies for Computer Algorithms)

Homework 3
DUE DATE: December 15, 2017

學號：b03902129　系級：資工四　姓名：陳鵬宇

1　問題定義

問題： *A Personnel Assignment Problem Solved by the Branch-And-Bound Strategy*

輸入：

- 一大小為 n 的線性有序集 Persons $P = \{P_1, P_2, \ldots, P_n\}$
  - $P_1 < P_2 < \cdots < P_n$
  - 我們可以想像這些人 (persons) 是由某些標準所衡量，例如身高、年齡、輩份等
- 一大小為 n 的集合 Jobs $J = \{J_1, J_2, \ldots, J_n\}$
  - 這些工作 (jobs) 是部分排序的
  - 每個人必需被指派到一個工作，即

$$\forall \text{person } P_i, 1 \le i \le n, \exists f(P_i) = \text{some } J_i, 1 \le i \le n.$$

  - 並且要求：
    - $f(P_i) \le f(P_j) \Rightarrow P_i \le P_j$.
    - $i \ne j \Rightarrow f(P_i) \ne f(P_j)$.

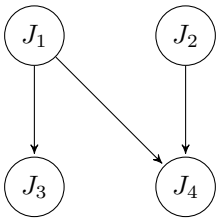
輸出： 最小的花費 (Cost) $= \sum_{i,j} C_{ij} X_{ij}$

- C_{ij} 為將 P_i 指派 J_j 的花費
- 若 P_i 被指派到 J_j：$X_{ij} = 1$，否則 $X_{ij} = 0$

2　解法敘述

2.1　例子描述

我們以下用一個例來說明之：給定 $n = 4$，那麼輸入會有 4 位 persons 和 4 個已經被「部分」排序的 jobs，以下為「部分」排序 jobs 的圖：



透過觀察此有向圖，我們知道：

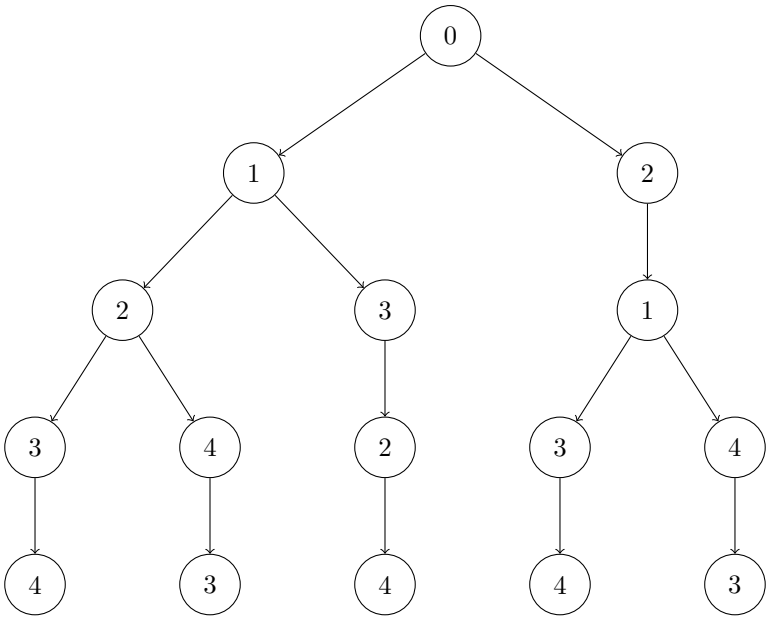
- 因為 J_1 為 J_3, J_4 的父節點，所以 J_1 必在 J_3, J_4 前。
- 因為 J_2 為 J_4 的父節點，所以 J_2 必在 J_4 前。

以下是所有可能的拓撲排序 (topologically sorted sequences):

1. J_1, J_2, J_3, J_4
2. J_1, J_2, J_4, J_3
3. J_1, J_3, J_2, J_4
4. J_2, J_1, J_3, J_4
5. J_2, J_1, J_4, J_3

透過列出所有可能的拓撲排序，我們可以根據以下規則得到一樹狀圖表示之:

1. 先選取其中任一個沒有父節點的節點，在我們上圖舉的例子為 J_1 或 J_2。
2. 該節點即為所有可能拓撲排序的第一個元素。
3. 移除該節點，回到步驟 1. 遞迴地做直到所有節點都被移除。



此外，我們會有如下的 C_{ij} 表格:

Persons \ Jobs	J_1	J_2	J_3	J_4
P_1	29	19	17	12
P_2	32	30	26	28
P_3	3	21	7	9
P_4	18	13	10	15

我們可以對每個列 (行) 同減該列 (行) 的最小值，使每一行至少有一個 0，並且記錄所有被刪減的值的和 (Total)，這個動作很重要!

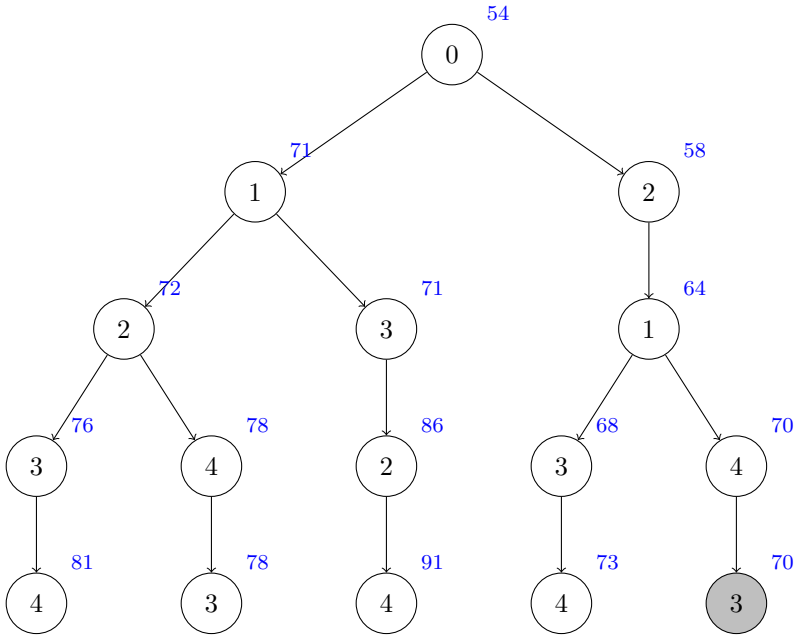
1. 先對每列分列同減該行最小值，即 $12, 26, 3, 10$

Persons \ Jobs	J_1	J_2	J_3	J_4
P_1	17	7	5	0
P_2	6	4	0	2
P_3	0	18	4	6
P_4	8	3	0	5

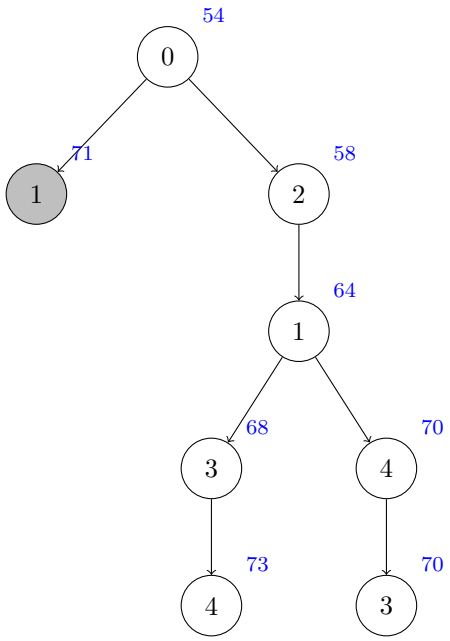
2. 我們發現第 2 行任一元素皆不為 0，再對該行同減最小值 3，此時每一行皆至少有一個 0

Persons \ Jobs	J_1	J_2	J_3	J_4
P_1	17	4	5	0
P_2	6	1	0	2
P_3	0	15	4	6
P_4	8	0	0	5

Total $= 12 + 26 + 3 + 10 + 3 = 54$. 這就是此題的 lower bound，因為每一個分支 (即分別指派 P_1, P_2, P_3, P_4) 都會至少花費 54 單位。所以我們可以將這個值拿來當初始值 (還未指派工作給任一人)



透過檢查最右下分支的子節點 3 (花費 $= 70$) 可以知道，任何比 70 單位花費更高節點的都不會是我們所求的解，故可以刪掉左邊節點 1 (花費 $= 71$) 下面的所有子節點。此時我們可以得到我們的 upper bound $= 70$。



如果我們在前面沒有先對所有的結點做減縮 (reduce) 的動作，考慮以下的指派工作:

$$P_1 \to J_1, C_{11} = 29,$$

即始我們將所有工作都指派完畢得到了 upper bound= 70，也無法 bound 住 $P_1 \to J_1$，因為他的花費只有 $29 < 70$，這也是為什麼一開始做減縮的動作這麼重要，可以幫助我們大副減少搜索的時間。

3 閱讀心得

這次的心得報告不像以往是閱讀 paper，而是一本原文的演算法教科書。Branch and Bound 和之前學習過的線性歸劃有些概念很相似，這些問題都是 NP hard，且都在處理整數問題，但我們仍然希望能找到一個較好的解法，即便他不是多項式時間。

不論是 Branch and Bound，或是之前曾經學到的 Prune and Search，他們的目的都是希望能夠減少「不必要的」計算過程，那些可能是早早就已經被發現不影響結果的，例如比 lower bound 低，或是比 upper bound 高，這跟期中考第 2 題要去找出 x_{right}, x_{left} 來刪去越界的線去加速整體計算時間有異曲同工之妙。

我覺得在學習演算法的過程中，是充滿愉快的，在 Google 相關資訊時，能夠挖掘到海量相關的內容，這種自我學習獲得知識的感覺真的很美好。每一個演算法就像是現實生活中某些事情的投影，我們致力於去降低時間複雜度、增加效率，現實生活中也是一樣，我們都希望事情能夠事半功倍，例如: 當我們想念好演算法，就會用挑比較好的參考資料去學習之。像老師您喜歡品茗，也會用較好的茶具泡茶吧!