程式設計（112-1）
作業六

作業設計：孔令傑
國立臺灣大學資訊管理學系

繳交作業時，請至 PDOGS（`http://pdogs.ntu.im/`）為唯一的一題上傳一份 C++ 原始碼（以複製貼上原始碼的方式上傳），並且到 NTU COOL 上傳一份 PDF 檔。每位學生都要上傳自己寫的解答。不接受紙本繳交；不接受遲交。這份作業的程式碼的截止時間是 ***10 月 17 日早上八點***，書面報告的截止時間則是 ***10 月 19 日早上八點***。為這份作業設計測試資料並且提供解答的助教是王裕勳和顏子鈞。

本次作業滿分基本上是 110 分，另外有額外加分最多可以加 25 分，得幾分就算幾分。若整學期有 n 份作業，則學期的作業總成績即為 n 份作業的總分除以 n（不論超過 100 與否）。

1 題目敘述

你剛被派任到一家租車公司。最近因為新型的共享交通工具崛起，導致許多傳統的租車客戶轉向其他選擇。另外，多數的車輛投放策略只著重於高需求區域，造成一些站點車輛過多，而某些站點卻常常出現車輛短缺的情況，導致公司無法有效利用所有資源，甚至可能失去大量的潛在顧客。為了改善這個狀況，總經理將一切託付在你身上，希望具有資訊與管理雙專長的你，可以設計一個解決方案（你也可以稱之為演算法、軟體、程式），希望可以產出能夠盡量最大化利潤，又可以同時符合現行車輛、場站條件的車輛投放策略[1]。

這家公司總共有 n 種車型及 m 個場站，每種車型的車輛數不盡相同，已知公司共有 A_i 輛第 i 種車型的車。此外，每一種車型也會為公司帶來不同的營收，第 i 種車型每小時的收費為 R_i 元。此外，車型 i 的預期基礎稼動率（utilization rate）是 U_{ij}，表示我們預期當一輛車型 i 的車被單獨投放在場站 j 時的稼動率。所謂稼動率，是指一輛車被租借的時間佔被投放的時間的比例，例如如果一輛車在一天中被兩次租用，一次 3 小時一次 5 小時，則其當日稼動率即為 $\frac{3+5}{24} = 33.3\%$。稼動率受多種因素影響，包括車輛廠牌、場站位置、鄰近場站的車輛數量、周圍人口數量和附近公司的數量等，不過請放心，公司的其他部門已經做好分析並估算稼動率的任務，因此本題會給定好稼動率的資料。

[1]這份作業的任務，是從孔令傑老師過往和真實企業合作的真實問題簡化而來的。

本題的決策變數為 x_{ij}，代表公司把第 i 種車投放 x_{ij} 輛車至第 j 個場站。例如：$x_{12} = 3$ 表示第 1 種車投放了 3 輛車至第 2 號場站，任務目標即為決定車輛投放方案 x_{ij} 以最大化預期單日租車營收。

1.1 基本模型描述

令 $I = \{1, 2, ..., n\}$ 與 $J = \{1, 2, ..., m\}$ 分別為包含所有車輛與所有場站的集合。為了最大化預期單日租車營收，我們將每日（24 小時）特定車型的費率 $24R_i$ 乘以該車型在特定站點的預期基礎稼動率 U_{ij} 以及車輛數 x_{ij}，最後加總所有車型，即可獲得所有 n 種車輛在所有 m 個場站的總營收

$$\max \sum_{i \in I} \sum_{j \in J} 24 R_i U_{ij} x_{ij} 。 \tag{1}$$

投放車輛當然也有限制，不能任意亂投。首先，針對任一款車型 i，所有場站投放的車輛數不能超過該車型總車輛數上限 A_i。其次，針對任一場站 j，所有車型的總車輛數不可以超過該場站的可用停車位上限 B_j。最後，因為車輛不能分割，決策變數 x_{ij} 屬於非負整數。以上條件寫成限制式會是

$$\sum_{j \in J} x_{ij} \leq A_i \quad \forall i \in I ; \tag{2}$$

$$\sum_{i \in I} x_{ij} \leq B_j \quad \forall j \in J ; \tag{3}$$

$$x_{ij} \geq 0 \quad \forall i \in I \quad \forall j \in J ; \tag{4}$$

$$x_{ij} \in \mathbb{Z} \quad \forall i \in I \quad \forall j \in J 。 \tag{5}$$

以上就是最佳化車輛投放策略的基本模型。

1.2 競食效應

前述的基本模型雖然好，但不夠實際，因為少考慮了車輛之間的「競食效應」(demand cannibalization effect)，就是每多投放一輛車，這輛新投放的車都會搶走鄰近車輛的一點生意，因此降低鄰近車輛的預期稼動率。當然，任兩輛車如果距離越遠，「競食效應」的負面影響應該愈小，也就是說兩個場站離得愈遠，愈不會彼此搶生意。

根據經驗和數據分析，兩個場站間的距離若超過 1 公里，彼此就遠到不會互相影響生意了。因此，我們將任兩場站的距離分為四個等級，如圖 1 所示，組成 $K = \{0, 1, 2, 3\}$ 這個距離等級集合，數字越小表示距離越近，$k = 0$ 代表無距離（場站

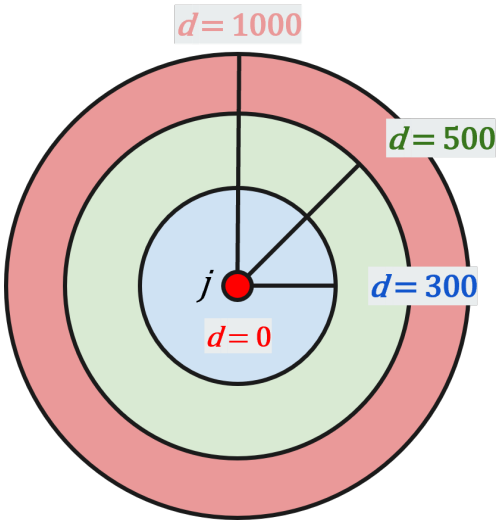

圖 1: 距離範圍圖解

本身），$k = 1$ 代表 1 至 300 公尺，$k = 2$ 則代表 301 至 500 公尺，$k = 3$ 則代表 501 至 1000 公尺，超過 1000 公尺則不考慮。

我們據此定義 $Q^{(k)}$，為特定場站周圍第 k 層內的場站中每多一輛車時，此場站中預期車輛稼動率的下降程度，例如若我們估計得到 $Q^{(0)} = 0.01$，表示當我們新投放一輛車到某個場站，我們預期該場站原有的車輛（不論車型）的稼動率都會各減少 1%；若 $Q^{(2)} = 0.001$，則對於任意一輛車而言，若在距離 301 至 500 公尺內的其他場站多任何一輛車（不論車型），則我們預期這輛車的稼動率會降低 0.1%。我們令任兩個場站 j_1 和 j_2 間的距離為 D_{j_1,j_2} 公尺。

總結以上，你將被給定車型數量 n、場站數量 m、稼動率影響程度 $Q^{(k)}$、每小時費用 R_i、車型車輛數 A_i、場站停車位上限 B_j、基礎稼動率 U_{ij}，以及任兩個場站距離 D_{j_1,j_2} 等資訊。請試著設計一個演算法，能夠一方面滿足限制，一方面最大化考慮競食效應的總預期收益，安排一個最好的車輛投放策略吧！

如果你不排斥閱讀一些數學式子，我們強烈建議你花點時間讀完第 1.3 節，不僅會獲得題目的精確敘述、更能瞭解如何精確地敘述這類的任務，也會累積更好的閱讀數學式子的能力，對你未來學習進階知識很有幫助。但如果你覺得這實在太困難了，請就跳過第 1.3 節，直接進入第 2 節吧。

1.3 考慮競食效應的進階模型

令 $N_j^{(k)}$ 為所有與第 j 個場站距離落在第 k 等級的場站集合，例如 $N_1^{(2)}$ 為與第 1 個場站距離在 301 至 500 公尺的其他場站的集合，而很自然地 $N_j^{(0)} = \{j\}$（只有場站 j 自

己）。本次任務中真正的、更合乎現實的、考慮競食效應的目標式可以被寫成

$$\max \sum_{i\in I}\sum_{j\in J} 24R_i\left(U_{ij} - \sum_{k\in K} Q^{(k)} \sum_{j'\in N_j^{(k)}} \sum_{i'\in I} x_{i'j'} + Q^{(0)}\right) x_{ij}。 \tag{6}$$

看到以上長長的一串先不要太擔心，其實跟基本模型比較，進階模型只是多了

$$-\sum_{k\in K} Q^{(k)} \sum_{j'\in N_j^{(k)}} \sum_{i'\in I} x_{i'j'} + Q^{(0)}$$

來表示競食效應帶來的負面效果。讓我們考慮車型 i 以及場站 j。以 $k=0$ 為例，此時場站 j 中車型 i 的每一輛車的稼動率預期會減少

$$Q^{(0)} \sum_{j'\in N_j^{(0)}} \sum_{i'\in I} x_{i'j'} + Q^{(0)} = Q^{(0)} \sum_{i'\in I} x_{i'j} + Q^{(0)}\ ,$$

也就是若場站 j 中所有車型合計共被投放了 v 輛車（$v = \sum_{i'\in I} x_{i'j}$），則場站 j 中車型 i 的每一輛車的稼動率預期會減少 $Q^{(0)}(v-1)$，注意是 $v-1$ 而不是 v 是因為自己不會對自己的稼動率帶來負面影響。再以 $k=1$ 為例，此時場站 j 中車型 i 的每一輛車的稼動率預期會減少

$$Q^{(1)} \sum_{j'\in N_j^{(1)}} \sum_{i'\in I} x_{i'j'}\ ,$$

亦即若距離場站 j 在 1 到 300 公尺的所有場站中所有車型合計共被投放了 v 輛車（$v = \sum_{j'\in N_j^{(1)}} \sum_{i'\in I} x_{i'j'}$），則場站 j 中車型 i 的每一輛車的稼動率預期會減少 $Q^{(1)}v$。依此類推，我們便可以理解這個複雜的目標式了。

2 輸入輸出格式

系統會提供許多筆測試資料，每筆測試資料裝在一個檔案裡。在每個檔案中，第一列存放六個整數 n、m、$Q^{(0)}$、$Q^{(1)}$、$Q^{(2)}$、$Q^{(3)}$，依序代表車型種類數量、場站數量、以及每一個場站周圍第 0、1、2、3 層內的場站中每多一輛車時，此場站中車輛稼動率的下降程度。接下來的資訊分為五部分，依序分別是 n 個每小時費用、n 個車型的總車輛數、m 個場站的可租用之停車格上限、$n\times m$ 個預期基礎稼動率，以及 $m\times m$ 個場站的距離：

- 第二列存放 n 個整數，依序存著 R_1、R_2 直到 R_n，表示各車型的每小時費用。
- 第三列存放 n 個整數，依序存著 A_1、A_2 直到 A_n，表示各車型的車輛數。
- 第四列存放 m 個整數，依序存著 B_1、B_2 直到 B_m，表示各場站的最大車輛數。

- 從第五列開始至第 $4+n$ 列，每列各存放 m 個整數，其中第 $4+i$ 列依序存著 $U_{i,1}$、$U_{i,2}$ 直到 U_{im}，表示第 i 種車輛在 m 個場站的預期基礎稼動率。

- 從第 $5+n$ 列開始至第 $4+n+m$ 列，每列各存放 m 個整數，其中第 $4+n+j$ 列依序存著 $D_{j,1}$、$D_{j,2}$ 直到 D_{jm}，依序表示第 j 號場站與其他每個場站的距離（單位為公尺）。一個場站 j 到自己的距離必定為 0，即 $D_{jj}=0$，且一個場站 j_1 到另一個場站 j_2 的距離會等於 j_2 到 j_1 的距離，亦即 $D_{j_1,j_2}=D_{j_2,j_1}$。

每一列中的任兩個數字之間都用一個空白鍵隔開。已知 $1\le n\le 20$、$1\le m\le 500$、$0\le Q^{(3)}<Q^{(2)}<Q^{(1)}<Q^{(0)}<\min_{i\in I,j\in J}\{U_{ij}\}$、$1\le R_i\le 1000$、$1\le A_i\le 50$、$1\le B_j\le 50$、$0<U_{ij}\le 1$、$1\le D_{j_1,j_2}\le 50000$、$D_{jj}=0$。

讀入資料後，請根據題目的描述，在不同的車型和車輛數量的組合間，嘗試最大化營收，進而安排每種車型在各站點應該停放多少車輛。安排好投放策略後，請依照如下格式輸出你的策略。你總共需要輸出 n 列，第 i 列依序輸出 $x_{i,1}$、$x_{i,2}$ 直到 x_{im}，代表第 i 種車輛在 m 個場站中各投放幾輛車輛，當然你的每個 x_{ij} 都應該要是整數，並且滿足車輛總數上限與場站投放車輛數限制。輸出時請在每一列都用一個逗號隔開相鄰的兩個整數。

舉例來說，如果一共有 $n=2$ 種車型和 $m=3$ 個場站，則一個合理的輸入會是

```
2 3 0.1 0.05 0.03 0.01
50 70
10 5
10 8 7
0.9 0.8 0.7
0.7 0.85 0.75
0 350 800
350 0 450
800 450 0
```

這邊我們安排了一套可能無腦但可行的投放方案，是投放 5、3、1 輛車型 1 的車到場站 1、2、3，再投放 1、3、1 輛車型 2 的車到場站 1、2、3。若要執行這個方案，我們可以輸出

```
5,3,1
1,3,1
```

根據這個投放策略，目標函數值（考慮競食效應後的預期單日總營收）為 3091.2。

3 評分原則

這一題的其中 75 分會根據程式運算的結果給分，共有 25 筆測試資料。你的程式不需要找出真的能最大化目標函數的最佳解（optimal solution）。只要你的程式在時間和記憶體限制內跑完、輸出符合格式規定，且確實是一組可行解（feasible solution，意即符合上文提及的所有限制），就會得到分數。對於每一組輸入，PDOGS 會檢查你的輸出，如果輸出格式不合乎要求或方案不可行，則在該筆測試資料會得到零分；如果合乎要求，且目標式值非負，則對每筆測資，我們依下一段的方式計分。

首先，助教會寫一個程式，在 PDOGS 上取名為「TA-algorithm」，這個演算法會稍微有點聰明（但不會太誇張）。更具體地說，TA-algorithm 的演算法會執行數輪。在每一輪中，演算法會去找在哪個場站多投放哪一個車型的車一輛能最大化考慮了競食效應的目標式值。假設找到的組合是場站 j 和車型 i，那就把 x_{ij} 的值加一，接著進行下一輪搜尋，直到多投放任何一輛車都不會提高目標式值為止。你可能已經發現這個演算法不一定會投放完所有車輛，而針對我們要求解的最佳化模型，最佳解確實也可能不會投放完所有車輛[2]。

計分時，每一筆測試資料原則上佔 3 分。假設你的輸出合乎格式，且投放方案是可行的（feasible，即不違反任何限制式），PDOGS 就會幫你的方案計算目標式值。假設 z 是你的解得到的目標函數值、z_{TA} 是「TA-algorithm」得到的目標函數值，則你在這筆測資的得分就是

$$1.5 + 1.5\left(\frac{\min\{z, 1.4z_{TA}\}}{z_{TA}}\right)。$$

換言之，只要方案可行就會得到 1.5 分（所以輸出一個 $n \times m$ 的全零矩陣其實也可以得分），然後你的投放策略愈好，就愈能得到好成績。要留意的是，如公式所示，如果 $z < 0$，亦即你的解比一輛車都不投放還要糟，那你得到的分數會比 1.5 分還低；與之類似，如果你的解比「TA-algorithm」還要好，你會得到超過 3 分，但最多就是 $1.5 + 1.5 \times 1.4 = 3.6$ 分[3]。

在 25 筆測試資料中，為了讓同學們比較容易拿到分數，前 20 筆測試資料的範圍會限縮在 $n \leq 15$、$m \leq 30$、$A_i \leq 25$ 和 $B_j \leq 25$，同學們即使實作了比較無效率的演算法，也不是很容易在執行時超過 PDOGS 上設定的時間上限。最後 5 筆測試資料則是題目規模相對大，不夠有效率的演算法可能會在時限內跑不完。同學們也可以針對不同規模的題目設計不同的演算法，在讀入資料後根據讀入的參數值大小決定要呼叫哪

[2]這算是這個最佳化模型的一個缺陷，不過因為這門課的重點不是最佳化模型建構，所以我們姑且就還是使用這個模型，這樣會讓盲目地投放全部車輛得到不好的解，進而增加一點演算法開發的挑戰性。

[3]大家可能有發現，我們已經把「TA-algorithm」的演算法告訴大家了。所以這次作業一方面是鼓勵有興趣的同學們積極挑戰，多拿分數的同時也看看自己能做得多好，但另一方面也不會為難同學們，只要能照著 TA-algorithm 的演算法正確實作，就幾乎可以得到滿分。

一個演算法即可。

寫程式之外，你還要寫一份書面報告（所謂「寫」，就是用電腦打的意思），並且上傳書面報告 PDF 檔至 NTU COOL。在報告裡請用文字描述你的演算法（可以用 pseudocode 但不能直接貼 code）、系統的設計（哪個函數做什麼、程式執行的流程等等），以及簡單心得感想。報告***不可以超過四面 A4 紙***。書面報告佔 35 分。

此外，授課團隊會考慮 PDOGS 上的得分以及書面報告上的解法，可能邀請若干同學在合適的上課時間，每位同學用 10 分鐘跟全班同學介紹自己的解法。被邀請的同學可以婉拒上台報告，但上台報告的同學可以獲得本次作業成績最多 10 分的額外加分。

4 繳交方式

請修課的同學們以個人為單位繳交你們的程式和報告。程式方面只要是 C++ 就可以，換言之，可以使用任何語法。書面報告方面可以用中文或英文，並且應該盡量讓自己的報告清爽、好閱讀；NTU COOL 上的「PD_reportFormatGuideline.pdf」有提供最基本的寫作與排版指引，強烈建議所有同學寫報告前先讀過一遍。

有兩件事需要注意。首先，系統會以每位同學的最後一次上傳得到的分數，做為該同學的分數，所以愈傳愈低分是有可能的。其次，原則上 PDOGS 不限制兩次上傳間的時間間隔，但如果有許多同學在同個時間大量地上傳執行時間很長的程式，導致 PDOGS 大塞車，屆時我們會對兩次上傳的時間間隔做出限制。