臺灣二００七年國際科學展覽會

科　　別：數學

作 品 名 稱：替機器人安排作業程序

學校 / 作者：臺北市立建國高級中學　洪錫成
臺北市立建國高級中學　李重毅

個人簡介

我是洪錫成，西元 1989 年出生，目前就讀台北市立建國高級中學三年級。從小受到家人的影響，對數學有著較濃厚的興趣。除了接觸數學之外，我空閒的時間也會從事一些休閒活動，像是打球、彈鋼琴。

很高興能夠參加這次的國際科展，讓我有機會對一些有趣的數學問題有更深一步的探討，也非常感謝辛苦指導過我們的教授與老師。

個人簡介

我是李重毅，出生在一個安平和樂的家庭，現在是台北市立建國高級中學三年級的學生。從國中到高中時期，因爲老師們的引導，我漸漸發現數學是個有趣的科目，有時候我會和幾個好朋友一起討論數學問題，因爲和朋友交流總能讓自己學的更多。

很幸運的，我們通過了國際科展的初選。經過這次研究中，讓我懂得如何有系統的研究一個數學問題。感謝教授老師們的指導，也謝謝和我一起研究的夥伴－洪錫成。

摘要

編號 1~mn 的 mn 個物件已隨機置入 $m \times n$ 階的矩陣中，另外有一行 m 個空格的暫存區供物件暫存用。我們探討將這 mn 個物件移至目標區並按照 1,2,…, mn 的次序排列，所需的移動步數；每一步的移動中，只能移動每一行最頂層的物件到其他行(含暫存區)的最頂層或目標區。

在這篇報告中，我們給出了一個適用於 $n \geq m-1$ 時的移動方法，此方法在一般的情形下，所需的移動次數未必是最少；但是在最不利於移動的情形下，我們證明此方法所需的移動步數爲最少。

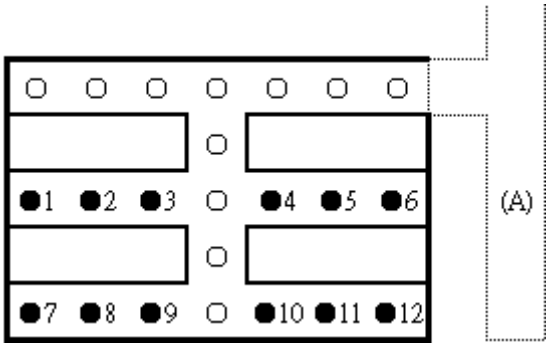
Abstract

There are *mn* objects, numbered from 1 to *mn*, put on an $m \times n$ matrix randomly, and there is another column with *m* blank spaces for temporary storage purpose during moving. In each step of moving, we can only move the top object from one column to the top of another column or to the target pile. The total steps needed to move these *mn* objects to the target pile in increasing order from the bottom to the top is studied in this article. A general method for solving this problem when $n \geq m-1$ is given, and we prove that it provides an optimal solution in the worst cases. However, it may not always provide the minimal steps in all cases.

壹、前言

一、研究動機

在一次數學教室的大掃除中，要把椅子搬出教室外。可是由於教室內有桌子阻隔，所以椅子沒有辦法順利搬出。如下圖，12 個黑點爲 12 張椅子，任一張椅子移動時不能跨越任意其他的椅子。於是我們想問：如果要把椅子按編號大小順序搬出教室外，該怎麼做呢？



二、研究目的

我們可以將前述的問題，推廣如下：編號 1~mn 的 mn 個物件已隨機置入 $m\times n$ 階的矩陣中，另外還有一行 m 個空格的暫存區供物件暫存用。如下圖，A 區是可以供暫存之用的空格，B 區是移動物件的「目標區」。在每次移動中，只能移動每一行最頂層的物件，該物件也只能移動到其他行的最頂層或 B 區。若將每次移動定爲一步，則將這 mn 個物件在目標區由下而上按照 1,2,…, mn 的次序 (即下圖 B 區的順序) 排列，最少要幾步方可完成？

$$\underset{\text{A}}{\begin{bmatrix}\text{O}\\ \text{O}\\ \text{O}\\ \vdots\\ \text{O}\end{bmatrix}\begin{bmatrix}\bullet & \bullet & \bullet & \bullet & \cdots & \bullet\\ \bullet & \bullet & \bullet & \bullet & \cdots & \bullet\\ \bullet & \bullet & \bullet & \bullet & \cdots & \bullet\\ \vdots & \vdots & \vdots & \vdots & & \vdots\\ \bullet & \bullet & \bullet & \bullet & \cdots & \bullet\end{bmatrix}_{m\times n}} \qquad \underset{\text{B}}{\begin{bmatrix}mn\\ \\ \vdots\\ \\ 3\\ 2\\ 1\end{bmatrix}}$$

貳、研究方法或過程

我們首先分別以 3×4 及 5×4 特定的排列情形，作移動過程的說明(見二)。接著說明 3×4 的移動方法，進而推廣到 3×5 等 $n\geq\dfrac{m(m-1)}{2}$ 的情形(見四)。最後結合 5×4 的移動方法，我們找到了對任意 $n\geq m-1$ 時都適用的移動方法(見五)。

一、符號

首先引進若干符號，以利後續的討論。如下圖，最左邊的 O 爲空格子 (暫存區)，a_{ij}

表示位置在第 i 列第 j 行的原始物件 $(1 \le i \le m, 1 \le j \le n, \text{且} 1 \le a_{ij} \le mn)$.

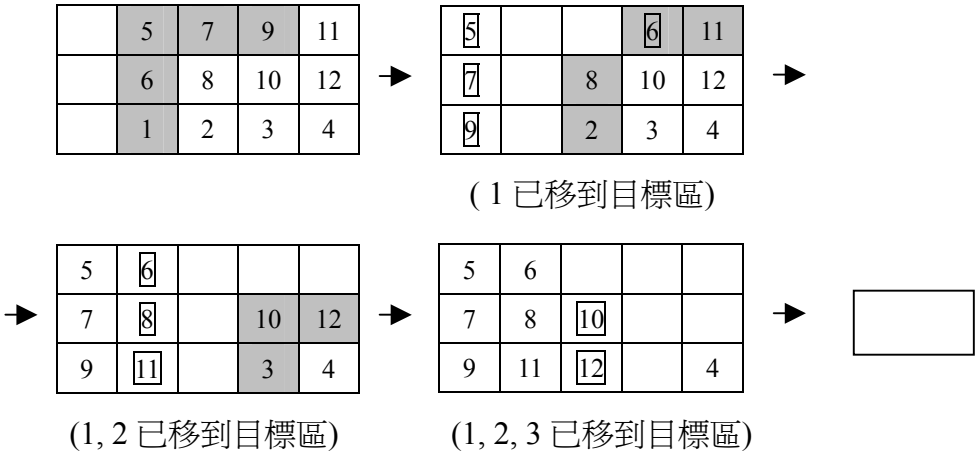
$$\begin{array}{ccccccc} \text{O} & a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ \text{O} & a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ \text{O} & a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{array}$$

不失一般性，令 $\min\{a_{1i}, a_{2i}, \cdots, a_{mi}\} < \min\{a_{1j}, a_{2j}, \cdots, a_{mj}\}$ $(i < j)$，則編號 1 的物件位於第一行，即 $1 \in \{a_{11}, a_{21}, \cdots, a_{m1}\}$. 接著將每個「位置」令爲 A_{ij} ，即在未經任何移動之前，物件 a_{ij} 位於位置 A_{ij} , 如下圖所示：

$$\begin{array}{ccccccc} A_{10} & A_{11} & A_{12} & A_{13} & \cdots & A_{1n} \\ A_{20} & A_{21} & A_{22} & A_{23} & \cdots & A_{2n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ A_{m0} & A_{m1} & A_{m2} & A_{m3} & \cdots & A_{mn} \\ \text{第} & \text{第} & \text{第} & \text{第} & & \text{第} \\ \text{零} & \text{一} & \text{二} & \text{三} & \cdots & n \\ \text{行} & \text{行} & \text{行} & \text{行} & & \text{行} \end{array}$$

二、首先，分別以 3×4 和 5×4 特定的排列情形爲例子，做初步的觀察：(我們以灰色表示準備要移動的物件及其編號，□表示剛移動完成的物件及其編號。)

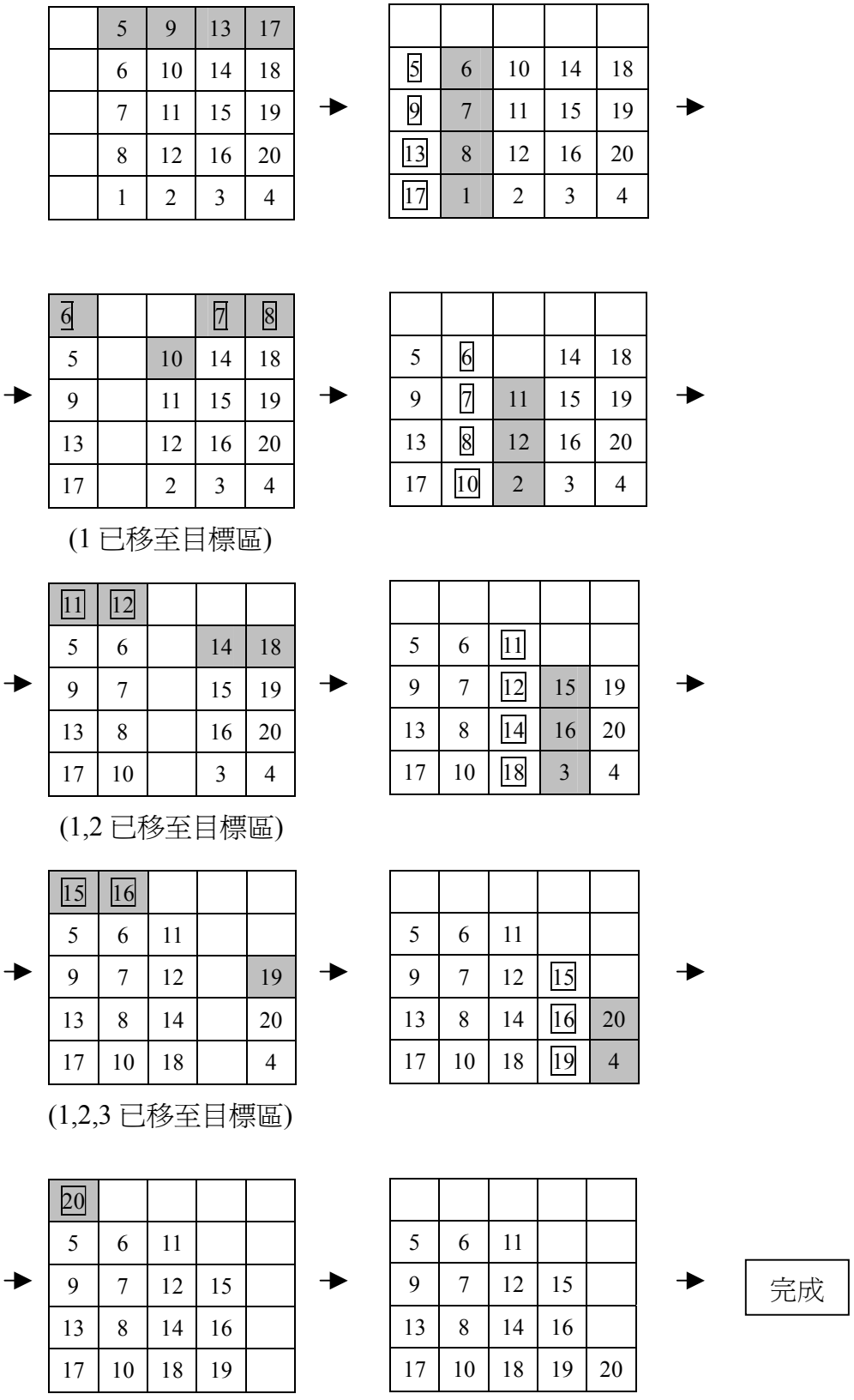
以 3×4 爲例：



上面的做法總共花了 21 步

註：最後每一行的物件皆呈由上到下遞增排列方式暫存，所以可按序將所有物件移動至目標區。

以 5×4 爲例：



上面的做法總共花了 44 步

註：上述的移動方法可以推廣到 $n = m - 1$ 的情形。

三、觀察：

當每一行數值最小的物件在最下層時，我們知道它將會需要比較多的移動次數，因爲除了最下層的物件之外，其他的物件都必須要移動至少兩次，也就是至少多搬一次 (移動到目標區勢必會搬一次)。

爲了使總移動次數降到最少，我們得用這多搬的一次就把它們排序好。所以，我們必須盡可能的在每次排序的過程中，都達到一次排序所能排序的最大值。在一個最不利的初始狀況下，每次只能取每行最上層的物件排序，最多 n 行；又因爲每一行只有 m 個空格，所以一次最多只能排序 $\min\{m,n\}$ 個物件。在這樣的前提之下，「一次排序最多個物件」就是降低移動次數的關鍵。

四、我們發現上述 3×4 的例子中所用的方法，也適用於 3×5 的情形，甚至對任意 $n\geq\frac{m(m-1)}{2}$ 的情形都適用。先以 3×5 的情形爲例做說明：

物件：

O	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
O	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}
O	a_{31}	a_{32}	a_{33}	a_{34}	a_{35}

位置：

A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}
A_{20}	A_{21}	A_{22}	A_{23}	A_{24}	A_{25}
A_{30}	A_{31}	A_{32}	A_{33}	A_{34}	A_{35}
第	第	第	第	第	第
零	一	二	三	四	五
行	行	行	行	行	行

接著我們說明每次移動都要遵守的規則：

規則 1：設已有 s-1 個物件被移動到目標區，若位置 A_{ij} 的編號 s 物件恰好在第 j 行的最上層，則將 s 移動到目標區。

規則 2：當執行下列步驟時，設要移動的是位於位置 A_{ij} 的物件，若它已經按照規則被移至目標區了，則要移動的物件就改爲位於位置 $A_{i+1,\,j}$ 的物件；若該行已經沒有其他的物件可以移動，則不執行該物件的移動。

步驟：

1. 將物件 a_{11}，a_{12}，a_{13} 從位置 A_{11}，A_{12}，A_{13} 由大到小移入第零行 (先移動編號大的物件，使編號大的物件位於底層。)
2. 將物件 a_{21} 從位置 A_{21} 移到位置 A_{13}。此時根據規則，第一行必定可清空。

3. 將位置 A_{22}，A_{13}，A_{14} 的物件由大到小移入第一行。將第二行清空。
4. 將位置 A_{23}，A_{24}，A_{15} 的物件由大到小移入第二行。將第三行第四行清空。
5. 將 A_{25} 移入第四行。則剩下的物件都已按照大小順序排列 (較上面物件的編號較小)，即可將全部的物件搬至目標區。

於是可以知道當 $n \geq \frac{m(m-1)}{2}$ 時：

$$\begin{array}{cccccc} O & a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ O & a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ O & a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{array}$$

根據前面所提到的移動方法，可以發現在最不利的初始條件下，

第二列的 1 個物件 a_{21}

第三列的 3 個物件 a_{31}，a_{32}，a_{33}

第四列的 6 個物件 a_{41}，a_{42}，…，a_{46}

$\vdots$

第 $m-1$ 列的 $\frac{(m-1)(m-2)}{2}$ 個物件 $a_{m-1,1}$，$a_{m-1,2}$，…，$a_{m-1,\frac{(m-2)(m-1)}{2}}$

總共有 $\frac{m(m-1)(m-2)}{6}$ 個物件需要移動三次才能到達目標區。而 a_{m1}，a_{m2}，…，a_{mn} 這 n 個物件只要移動一次就能到達目標區，其餘的則需要移動兩次才能到達目標區。所以，至多需要

$$\frac{m(m-1)(m-2)}{6} + 2n(m-1) + n$$

次移動即可將這 mn 個物件移動到目標位置。

爲什麼只適用於 的情形 $n \geq \frac{m(m-1)}{2}$ 呢？因爲在最不利的初始條件下，第 $m-1$ 列有 $\frac{(m-1)(m-2)}{2}$ 個物件需要移動三次。而在需要移動三次的物件完成排序前，每次清空完某一行後，都需要有 m 行最上方的物件放入該行。所以只有在當 $n \geq \frac{(m-1)(m-2)}{2} + m - 1 = \frac{m(m-1)}{2}$ 時，目前的移動方法才成立。換句話說，如果 $n < \frac{m(m-1)}{2}$，前面的移動方法未必能將物件都移至目標區。

五、前面 5×4 特例的方法，也適用於 $n = m-1$ 的情形。於是將這兩種移動方法結合起來，就得到適用於 $n \geq m-1$ 的移動方法：

$$\begin{array}{cccccc} O & a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ O & a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ O & a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{array} \qquad \begin{array}{cccccc} A_{10} & A_{11} & A_{12} & A_{13} & \cdots & A_{1n} \\ A_{20} & A_{21} & A_{22} & A_{23} & \cdots & A_{2n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ A_{m0} & A_{m1} & A_{m2} & A_{m3} & \cdots & A_{mn} \end{array}$$

1. 移動規則

每一個物件的「移動」時，都要遵守以下幾項規則：

規則 1：設已有 $s-1$ 個物件被移動到目標區，若位置 A_{ij} 的編號 s 物件恰好在第 j 行的最上層，則將 s 移動到目標區。

規則 2：當執行下列步驟時，設要移動的是位於位置 A_{ij} 的物件，若它已經按照規則被移至目標區了，則要移動的物件就改爲位於位置 $A_{i+1,\,j}$ 的物件；若該行已經沒有其他的物件可以移動，則不執行該物件的移動。

規則 3：被「凍結」的物件 (見下列的移動程序 2.(1)) 不再執行程序(1), (2), (3), (4)，直到程序(1), (2), (3), (4)執行完畢後，再一起執行程序(5) (滿足規則 1 的物件仍要依照規則 1 移動)。

2. 移動程序

(1) 將位於 $A_{11}, A_{12}, \cdots, A_{1m}$ 的 m 個物件 (即 $a_{11}, a_{12}, \cdots, a_{1m}$) 移動到空行 (即 $A_{10}, A_{20}, \cdots, A_{m0}$)，滿足自下而上，編號由大到小排列，並將這 m 個物件「凍結」。再將位於 $A_{21}, A_{31}, \cdots, A_{m-1,1}$ 的 $m-2$ 個物件 (即 $a_{21}, a_{31}, \cdots, a_{m-1,1}$) 移動到 $A_{1m}, A_{1,\,m-1}, \cdots, A_{13}$ 的位置，則根據規則， $A_{11}, A_{21}, \cdots, A_{m1}$ 此行被騰空。

(2) 將位於 $A_{12}, A_{13}, \cdots, A_{1,\,m+1}$ 的 m 個物件移動到空行 $A_{11}, A_{21}, \cdots, A_{m1}$ ，自下而上，編號由大到小排列，並將這 m 個物件「凍結」。再將位於 $A_{22}, A_{32}, \cdots, A_{m-1,\,2}$ 的 $m-2$ 個物件移動到 $A_{1,\,m+1}, A_{1m}, \cdots, A_{14}$ 的位置，則根據規則， $A_{12}, A_{22}, \cdots, A_{m2}$ 此行被騰空。

(3) 重複第二步的動作，將位於 $A_{1i}, A_{1,i+1}, \cdots, A_{1,\,i+m-1}$ 的 m 個物件移動到空行 $A_{1,\,i\text{-}1}, A_{2,\,i\text{-}1}, \cdots, A_{\mathrm{m},\,i-1}$ ，自下而上，編號由大到小排列，並將這 m 個物件「凍結」。再將位於 $A_{2i}, A_{3i}, \cdots, A_{m-1,\,i}$ 的 $m-2$ 個物件移動到 $A_{1,\,i+m-1}, A_{1,\,i+m-2}, \cdots, A_{1,\,i+2}$ 的位置，則根據規則， $A_{1i}, A_{2i}, \cdots, A_{mi}$ 此行被騰空。 $(3 \le i \le n-m+1)$

(4) 將位於 $A_{1,n-m+1},\cdots,A_{1,i-3},A_{1,i-2},A_{1,i},A_{1,i+1},\cdots,A_{1,n}$ 的 m-1 個物件移動到空行 $A_{1,i-1},A_{2,i-1},\cdots,A_{m-1,i-1}$，自下而上，編號由大到小排列，並將這 $m-1$ 個物件「凍結」。再將位於 $A_{2i},A_{3i},\cdots,A_{m-1,i}$ 的 $m-2$ 個物件移動到 $A_{1,n-m+1},A_{1,n-m+2},\cdots,A_{1,i-1}$ 和 $A_{1n},A_{1n-1},\cdots,A_{1,i+2}$ 的位置。則根據規則，$A_{1i},A_{2i},\cdots,A_{mi}$ 此行被騰空($n-m+2\le i\le n$).

(5) 此時所有物件都已按照每一行自下而上，編號由大到小的順序排列，即可將全部的物件一一搬至目標區。

六、移動步數的討論

我們知道當 $n\ge\frac{m(m-1)}{2}$ 時，在最不利的初始條件下，第 $m-1$ 列有 $\frac{(m-1)(m-2)}{2}$ 個物件需要移動三次。當 $n<\frac{(m-1)(m-2)}{2}$ 時，則需要移動三次的物件會有一部分被切掉，所以我們在計算需要移動三次的物件數時，需要將每一列要移動三次的物件數與總行數做比較，即需要移動三次的物件數爲 $\sum_{k=2}^{m-1}\min\left(C_2^k,n\right)$，總移動次數爲

$$\sum_{k=2}^{m-1}\min\left(C_2^k,n\right)+2n(m-1)+n.$$

當 $n\ge\frac{(m-1)(m-2)}{2}$ 時，對於 $2\le k\le m-1$，$C_2^k\le n$ 均成立。則總共的移動次數爲

$$\sum_{k=2}^{m-1}\min\left(C_2^k,n\right)+2n(m-1)+n=\frac{m(m-1)(m-2)}{6}+2n(m-1)+n.$$

七、初始條件的討論

將 mn 個物件隨機放入 $m\times n$ 的容器中，最少需要移動 mn 次，最多需要移動 $\sum_{k=2}^{m-1}\min\left(C_2^k,n\right)+2n(m-1)+n$ 次，即可將這些物件全都排序好。若這些物件初始的排列狀況是每一行由上到下遞增的話，那麼只需要移動 mn 次即可完成移動；另一方面，若這些物件初始的排列狀況是以下的情形 (也就是我們之前提到所謂的「最不利的情形」)，則依照我們的方法需要移動 $\sum_{k=2}^{m-1}\min\left(C_2^k,n\right)+2n(m-1)+n$ 次：

$$\begin{array}{ccccccc}
\mathrm{O} & a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\
\mathrm{O} & a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\
\vdots & \vdots & \vdots & \vdots & & \vdots \\
\mathrm{O} & a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn}
\end{array}$$

其中，

$$a_{m1}, a_{m2}, \cdots, a_{mn} = 1, 2, 3, \cdots, n,$$

$$\begin{array}{ll} a_{11} < a_{21} < \cdots < a_{m-1,1} & a_{m-1,1} < a_{22} \\ a_{12} < a_{22} < \cdots < a_{m-1,2} & \text{且} \quad a_{m-1,2} < a_{23} \\ \vdots & \vdots \\ a_{1n} < a_{2n} < \cdots < a_{m-1,n} & a_{m-1,n-1} < a_{2n} \end{array}$$

不過，我們並不保證其他的初始狀況都不需要移動這麼多次。

參、研究結果

(一) 定義$h(m,n)$是在某一個$m \times n$的初始狀況下，將這 mn 個物件按順序排列好所需要花的最小步數，則在$n \geq m-1$時

$$mn \leq h(m,n) \leq \sum_{k=2}^{m-1} \min\left(C_2^k, n\right) + 2n(m-1) + n.$$

特別是當$C_2^{m-1} \leq n$時，

$$\begin{aligned} h(m,n) &\leq \sum_{k=2}^{m-1} \min\left(C_2^k, n\right) + 2n(m-1) + n \\ &= \frac{m(m-1)(m-2)}{6} + 2n(m-1) + n \end{aligned}$$

(二) 在下表中我們給出了若干特定$m \times n$所需步數的上界與下界，其中，

$$(x, y) = \left(mn,\ \sum_{k=2}^{m-1} \min\left(C_2^k, n\right) + 2n(m-1) + n \right)$$

m \ n	$m-1$	m	$m+1$	$m+2$	$m+3$	$m+4$
2	(2,3)	(4,6)	(6,9)	(8,12)	(10,15)	(12,18)
3	(6,11)	(9,16)	(12,21)	(15,26)	(18,31)	(21,36)
4	(12,25)	(16,32)	(20,39)	(24,46)	(28,53)	(32,60)
5	(20,44)	(25,54)	(30,64)	(35,73)	(40,82)	(45,91)

(三) $3 \times n$的證明

1. 我們只要證明了在一個特定的初始排列情形下，任何方法所需要的移動步數都不會比我們的移動方法少，那麼就可以說明$\sum_{k=2}^{m-1} \min\left(C_2^k, n\right) + 2n(m-1) + n$確實是

$h(m,n)$的最小上界，所以接下來的證明我們都只討論最不利於移動的初始條件的情形。在證明$m\times n$之前，我們先証$3\times n$的情形：

2. 當$m=3$時，只要物件排列的初始狀況跟滿足我們之前討論的條件，則第一行至少會有一個物件需要移動三次。因爲在將第一行清空時，可分爲以下兩種情形(參考下圖，○爲空格，●爲物件)：

   (1) 物件 a 移動到右行物件的上方：

   令物件 a 移動到的那行最下方的物件爲 c，這時物件 c 必小於物件 a，於是物件 a 需再移動一次才能到達目標區，也就是移動三次。

   (2) 物件 a 移動到暫存區：

   這時不論物件 b 移動到物暫存區或是右行物件的上方，物件 b 的下方都一定有比它小的物件，所以 b 需要再移動一次才能達到目標區。

○	a	●	●	●	…
○	b	●	●	●	…
○	1	2	3	4	…

3. 故得知至少有一個物件需要移動三次。而我們又已經構造出一種方法使$m=3$時只有一個物件需要移動三次，所以當$n>2$時，我們得到

$$3n\le h(3,n)\le 5n+1$$

(四) $m\times n$的證明

我們首先觀察到幾項移動的規律(見 1.)，提出了「空格」的概念(見 2.)，進而說明了移動過程中大部分物件皆須滿足的條件(見 4.5.)。有了這些基礎，我們針對各種移動情形做討論，並證明了那些情形都無法降低我們的移動步數(見 6.7.8.)，最後利用空格增加的模式計算出需要花三次移動的物件數。(見 9.)

1. 在最不利於移動的初始狀況下，易知每個第 m 列的物件都至少需移動一次，每個不是第 m 列的物件都至少需移動兩次。所以在下面的討論中，將除了這些次數以外的部分稱作多花的步數，並研究至少需要多花幾步。首先，我們發現了以下幾項規律：

   (1) 在a_{mn}被移至目標區前，任何物件如果是移動到未被清空過的區域，那麼該物件移動到的位置下方必有一物件的數值比它小，即該物件至少需多花一步才可移動到目標區。

   (2) 在a_{mn}被移至目標區前，若要從同一行中移動 k 個物件到同一個暫存區，而這 k 個物件是由上到下呈遞增排列，則有$k-1$個物件都需多花 1 步才可移動到目標區(即使在移動這 k 個物件的過程中有移動其他物件也不影響此規律)。

2. 因爲每移動一個物件到目標區後，總共的空格數就會多一個，而一開始有 m 個空格，所以在將第 k 行清空之後，都會有$m+k$個空格以供暫存，其中有 m 個是在第 k 行清空後位於第 k 行的空格。因爲這 m 個空格是每次清空完第 k 行後都會有的，所以接下來我們只討論剩下的 k 個空格。這 k 個空格的發展方式會隨

著不同的移動方法而有所不同，我們先以我們的移動方法來做說明：(。表示空格 ・表示物件 ×表示已排序好的區域)

(1) 第一行被清空後，有 1 個空格，將空格放於第二行

× ○ ○ ● ● ● ● ● ● ●
× ○ ● ● ● ● ● ● ● ●
× ○ ● ● ● ● ● ● ● ●…
× ○ ● ● ● ● ● ● ● ●
⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮

(2) 第二行被清空後，有 2 個空格，我們選擇第三和第四行各一個空格

× × ○ ○ ○ ● ● ● ● ●
× × ○ ● ● ● ● ● ● ●
× × ○ ● ● ● ● ● ● ●…
× × ○ ● ● ● ● ● ● ●
⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮

(3) 第三行清空後，有 3 個空格，於第四行空出兩格，第五行空出一格

× × × ○ ○ ○ ● ● ● ●
× × × ○ ○ ● ● ● ● ●
× × × ○ ● ● ● ● ● ●…
× × × ○ ● ● ● ● ● ●
⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮

(4) 第四行清空後，有 4 個空格，第五行有兩個空格，第六、七行各有一個空格

× × × × ○ ○ ○ ○ ● ●
× × × × ○ ○ ● ● ● ●
× × × × ○ ● ● ● ● ●…
× × × × ○ ● ● ● ● ●
⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮

(5) 第五行清空後，有 5 個空格，第六、七行各有兩個空格，第八行有一個空格

× × × × × ○ ○ ○ ○ ●
× × × × × ○ ○ ○ ● ●
× × × × × ○ ● ● ● ●…
× × × × × ○ ● ● ● ●
⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮

3. 根據以上的圖做類推，我們可以列成下表：

第 k 行被清空	1	2	3	4	5	6	…
這 k 個空格分別位在第 $c_1, c_2, \cdots c_k$ 行	2	3,4	4,4,5	5,5,6,7	6,6,7,7,8	7,7,7,8,8,9	…

而根據上表，我們又可以觀察出以下幾點：

(1) 因爲第 k 行以前都是已排序好的，若安排空格在已排序的區域並不會減少

移動次數，又因爲第 k 行是空的，所以我們的空格都安排在還未排序的區域，也就是 c_i 皆大於 k。

(2) 我們移動方法的空格發展模式爲：每行一次最多增加一個空格，從第 $k+1$ 行開始增加，加到總空格數等於 k 時停止。也就是說，在每行一次最多增加一個空格的前提下，我們把可增加的空格數盡可能地加在較前面(左邊)的行。因爲清空第 k 行時，所需多花的步數取決於第 k 行的空格數(在 6.有更詳細的說明)，所以我們盡可能的選擇空格在較前面的行應該是一種最少步數的選擇。

4. 首先我們要證明，如果欲使總移動次數爲最少，則在移動過程中，除了已排序好的物件外，其餘的物件皆必須滿足下列兩點：

(1) 除了第 m 列的物件之外，每一行的物件皆爲由上至下遞增排列。

(2) 對任意 $1 \le i < n$ ，第 $i+1$ 行的第二個物件小於第 i 行的第 $m-1$ 個物件。

(我們的初始條件也滿足這兩點。)

5. 說明：

假設有一種移動方法，會使至少一行中兩個物件不呈由上到下的遞增排列。那麼此方法必定是在某一次移動後，排列情形才不呈由上到下遞增；也就是說，在移動這一步之前，物件的排列情形與符合上面兩個條件。於是此一步移動必定是在清空第 k 行時，將第 i 行的物件 p，移到第 j 行物件 q 的上方($p > q$, $k \le i$, $k \le j$)。

兩個物件不呈由上到下的遞增排列的好處是可以一次把同一行的兩個物件加入排序；所以，設在清空第 $k+s$ 行後，我們將物件 p,q 與其他要排序的物件一起移動至第 $k+s$ 行。在清空第 k 行到清空第 $k+s$ 行的過程中，會有 s 次排序 m 個物件的時候沒有把第 j 行的物件加入排序，這 s 次的排序可以用來排序其他行，可是相對的在清空第 j 行時就會增加至多 s 個物件需要移動三次；爲了彌補這多花的 s 步，我們必須在 s 次排序的時候將空格盡量選在某些行使該行的空格數較多，但是總空格數是固定的，因此其他行的空格就比較少，也就是我們在清空某些行時會省 t 步，但清空其他行時需多花 t 步，這樣一來，我們並沒有辦法彌補先前多花的 s 步。

總而言之， 刻意使一行中兩個物件不呈由上到下的遞增排列並不會減少總移動步數。

6. 根據以上的說明，我們知道在移動過程中，除了已排序好的物件外，其餘的物件皆必須滿足我們在 4.裡提到的那兩點，所以接下來討論的移動中皆須滿足那兩點。有了這樣的前提，我們可以得知下列幾項事情：

(1) 在將第 k 行清空時，如果第 k 行有 i 個空格，則第 k 行有 $m-i-2$ 個物件需要多花一步。而 i 就是我們將第 $k-1$ 行清空時選擇放在第 k 行的空格數。因爲第 k 行除了第 m 列的物件之外，其餘 $m-i-1$ 個物件是由上到下呈遞增排列。我們只能一次排序一個物件而不增加移動步數，所以剩下的 $m-i-2$ 個物件就需要多移動一次。

(2) 若第 $k-1$ 行被清空時第 j 行有 i_1 個空格，欲使第 k 行被清空時第 j 行有 i_2 個空格，且 $i_1 < i_2$，則第 j 行將有 $i_2 - i_1 - 1$ 個物件需要多花一步。因爲第 j 行除了第 m 列的物件，其餘的 $m - i_1 - 1$ 個物件是由上到下呈遞增排列。如果我們想要增加 $i_2 - i_1$ 空格，我們就必須將 $i_2 - i_1$ 個物件移到別行；但是我們一次只能排序一個物件而不增加移動步數，所以其餘 $i_2 - i_1 - 1$ 個物件就需多花一步。

7. 根據 6.(2)我們知道，在清空第 k 行時，若要在第 j 行增加 i 個空格，那麼第 j 行會有 $i-1$ 個物件需要多花一步。而我們發現，如果我們這麼做了，那麼在清空第 j 行時，會比原本少至多 $i-1$ 個物件需要多花一步；但當我們在第 j 行多增加 i 個空格的時候，相對的就是減少了其他行空格的數目，這樣比較起來，在第 j 行多增加 i 個空格並不會使總移動步數變的比原本每一行最多只增加一個空格的還要少。所以我們只需討論每一行最多增加一個空格的情形即可。

8. 在 3.(2)裡我們有討論到，我們的移動方法是盡可能的將所有的空格都選在前面(左邊)的行。假設有一種空格增加的方式跟我們不相同，那麼它必定是在清空第 k 行時，在第 i 行不選空格，而將這空格選在第 j 行 $(i < j)$；這樣的選擇會使得清空第 j 行時至多多一個物件不需要多花步數，但在清空第 i 行的時候就會多了一個物件需要多花步數；整體而言，這樣的空格增加方式並不會比較有利，而且把空格選在較後方的行將會更有可能浪費空格(因爲較後方的該行可能已經沒有物件需要多花步數，將空格多增加在該行就無法減少步數)。由此我們知道，所有其他的空格增加方式都不會比我們的好。

9. 最後，我們利用空格增加的模式來計算需要花移動三次的物件數。我們用一個較簡易的方法來表達空格數的發展情形，在清空第 k 行後，第 $k+1, k+2, k+3, \cdots$ 行的空格數如下：(後面的項都是 0)

k	空格
0	0, 0, 0, 0, …
1	1, 0, 0, 0, …
2	1, 1, 0, 0, …
3	2, 1, 0, 0, …
4	2, 1, 1, 0, …
5	2, 2, 1, 0, …
6	3, 2, 1, 0, …

於是可以發現此數列的演變有底下規則：

(1) 數列的每一項總和等於 k

(2) 第 $k-1$ 個數列刪去首項 a 再把前 $a+1$ 項都加 1 就形成第 k 個數列

例如 k=6 時數列爲 3, 2, 1, 0, 0, 0, …

⇨ 2, 1, 0, 0, 0,…(刪去首項 3)

⇨ 2+1, 1+1, 0+1, 0+1, 0,…(前 3+1 項都加 1)

⇨ 產生 $k=7$ 時的數列 3, 2, 1, 1, 0, …

我們可以計算出總空格數爲 $1\times 0+2\times 1+3\times 2+\cdots$

所以，需要花三步移動的物件數爲

$$1\times(m-2)+2\times(m-3)+3\times(m-4)+\cdots+(m-2)\times 1=\frac{m(m-1)(m-2)}{6}$$

與我們之前的結果相同！

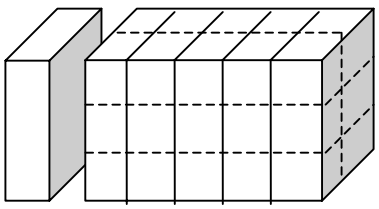
肆、討論與應用

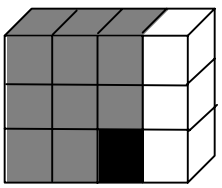
(一) 當 $n<m-1$ 時，因爲一次能排序的個數較少，我們得移動更多次才能完成；再加上能夠調度物件的空間並不多，我們還沒有找出一般的方法來解決 $n<m-1$ 的情況，也就是不同的初始狀況有不一樣的移動方法來完成。

(二) 後續問題的考慮：

1. 如果初始的狀況不是一個完整的矩形，也就是每一行的個數不都是 m 個；如果每個編號的物件不只一個；如果暫存區不是存放 m 個物件，或著不只一個暫存區，那麼移動的方法和次數有什麼變化？

2. 原本的問題是把這 mn 個物件放入 $m\times n$ 的二維容器中，再將它們按序取出。如果將問題推廣爲：把 mnk 個物件放入 $m\times n\times k$ 的三維容器，那麼若根據二維的規則做延伸可以得到這 $m\times n\times k$ 的容器是由 k 個分別獨立不相干的 $m\times n\times 1$ 的面所組成，而暫存區亦爲一個 $m\times n\times 1$ 的面，如左下圖是 $m=3$，$n=2$，$k=5$ 的狀況，即一個 $3\times 2\times 5$ 的容器；而若我們想要移動某一個面的某個物件，則該物件的上方及前方 (左方) 必須沒有物件擋著，如右下圖，我們將一個 $3\times 4\times 1$ 的面轉過來看，若我們要移動黑色的物件，則灰色的格子都必須清空。此時二維的方法推廣到三維是否適用？若將規則再增加，變得只能看到表層的物件，那麼又會是怎麼樣的情形？





(三) 生活中的應用：

若我們要對 mn 筆亂序的資料進行排序，而這 mn 筆資料間又有一定的關聯性(類似於我們每一行的關係)，則可以將我們的方法輸入電腦或機器人，使它們幫我們將這 mn 筆資料排序好。

評語

題目有趣，結果細分爲多種情況而分別討論。