數學傳播 38卷1期, pp. 16-21

我和數學研究的第一次邂逅

黃光明

一

1967 年秋, 我進入 Bell Labs 數學統計部門的統計所做基礎研究工作。和我們鄰近的是計算機部門, 當時正是新興學科, Bell Labs 計劃下一年招收 250 計算學博士, 卻發現全美國的產量不足此數。計算機部有一個菲律賓華僑林牲, 常來找我打橋牌, 出去打牌時, 在車裏聊完牌後, 不免就聊起彼此的工作。他告訴我他正在思考的一個問題是電腦裏的一個排序問題; 譬如紐約市有二百萬個電話用戶 (那時電話還是獨占性的, 所以這些全是 AT&T 的用戶), AT&T 要編一本電話簿, 需要把二百萬個用戶的姓名依英文字母次序排列起來。什麼方法可以最快的排好呢? 據說那一年代的電腦, 三分之二的時間都花在排序上, 像發薪水, 就要爲員工排序才便於尋找。

要比較排法, 首先要根據原始的資料是什麼, 及以何種形式儲存在電腦裏來規定我們討論的是那一類排法。爲了便於敍述, 我們把電腦裏的實際操作改成以下較通俗但實質不變的說法。假設每一個名字都寫在一張卡上而另一面則分別寫上一到兩百萬的數字以便區別卡片。這兩百萬張卡都是數字面朝上的, 只有在比兩張卡時, 才可以把它們翻過面來比對英文名字的先後, 然後把比對的結果記在一個記憶箱裡, 再翻回兩張卡。注意一張卡即使以前被比過, 只要一翻回來, 我們對它即一無記憶, 唯一知道的只是它的號碼。譬如我們有三張號碼爲 x, y, z 的卡背面分別寫著 John, Tom, Mary 三個名字等著被排序。我們的比法如果是先比 xy, 看到 John 先於 Tom 後記下 x 前於 y, 再比 xz, 看到 John 先於 Mary, 記下 x 先於 z, 則此時仍不知 y 與 z 孰先, 因爲記憶箱裏保存關於 x, y, z 的資料不能回答這問題。我們並不記得 Tom 和 Mary 這些名字。注意以上比法如果改成第二比是 yx, 則會得到同樣需比三次的結果。但若第一比是 xz 且第二比是 yz 時, 記憶箱的資料是 x 先於 z 先於 y, 則三張卡 (及其上的名字) 就已排好了。但這並不表示後一法比前一法好, 因爲後一法並不能保証比兩次一定能排序成功。例如把 y 和 z 的英文名互換後, xz 和 yz 兩比即不能完成排序。讀者需要自己做幾個簡單的情形, 像三張卡或四張卡, 以充分了解什麼是一個算法。注意每一比有兩個結果, 如果得到結果後仍不能排序時, 則必須再提出下一比。在三或四的簡單情形下, 大概所有合理的比法都會得到相同的比數; 但在五張卡時, 比法就有優劣之分。讀者可以試試能否設計出一個用七比就能完成排序的比法。

二

林牲所思考的問題是: 把 n 張卡排序的最好比法是什麼? 這個比法需要用多少次比較? 雖然目標定爲最好比法, 但很多難題常常無法強求完美結果, 能得到一佳法已經算是成功了。當然佳與不佳不是由提出者說了算, 通常是要算出一個理論上的極限, 譬如 5 張卡有 $5\times4\times3\times2\times1=120$ 種不同的排序, 而任一比較, 如 x 和 y, 都把這 120 種分成兩類, 一類是在排序中 x 先於 y, 另一類是 y 先於 x。一開始我們完全不知正確的排序是 120 種中的那一種, 直到比較 xy 的結果後, 可以排除 60 種。另一次比較, 如 z 和 w, 又可排除 30 種。如果每一比較都能排除一半的排序時, 則因爲 120 大於 2 的六次方, 六次比較是無法保証只剩下一個排序沒有被排除的; 換句話說, 會有多於一個排序符合所有六次比較的結果, 因此排序沒有完成, 即五張卡的排序, 理論極限說至少要七比。當我們提出一個只需七比的比法時, 就知必是最佳比法了。當然, 並不是每一次比較都能排除恰恰一半的, 如果一個比較的兩個可能的結果把尙存的排序分爲不均勻的兩部份時, 其中必有一部份大於一半, 其影響是理論極限數會被提得更高。

如果讀者做了五張卡的習題, 則一定體會了寫出一個比法的煩瑣。一個用了七比的比法, 並不是只要寫出七個比較來, 而是在不同的比較結果下寫出不同的七種比較。因此如果對每一個 n, 要給出一個比法, 則工作量就像是愚公移山了。所以我們要追求的是一個對所有 n 都成立的比法。可能因此要降低一些答案的質量, 或由最佳變成佳, 但這樣的交換不但是實際的, 有時也是必要的。

當年我聽了林牲的問題, 大感興趣, 就以全力來做這問題, 經過和他一再的探討後, 一周後終於設計出一個巧妙的比法, 對所有 n 都適用。敍述這一比法前, 我們要先介紹「遞歸法」的原則。這個原則就是做 n 張卡問題時, 我們假設卡數比 n 小的問題都已會做, 而這個假設必需在一個起始點上爲眞。譬如在一張卡時, 不需任何比較 (也無可比) 即可排序; 兩張卡時, 很顯明的比一次即可排序, 因此滿足了起始點的要求, 我們就可以在敍述一個 n 張卡的比法時, 隨時用 m 張卡 (m 是一個小於 n 的數字) 的比法來處理任 m 卡的排序。

我們先給出 n 是偶數時的排法:

第一步, 把 n 卡分成 $n/2$ 對比較, 命名第 i 對的先者爲 y_i 後者爲 x_i。

第二步, 引用遞歸法的性質, 把 $n/2$ 個 x_i 排序。因爲 x_i, x_j 可任意換名而無損我們的論點, 可假設次序爲 $x_1, x_2, \cdots, x_{\frac{n}{2}}$ (先寫的在序中爲先)。把 y_1 放在 x_1 之前視同 x_0, 則 $x_0, x_1, x_2, \cdots, x_{\frac{n}{2}}$ 仍是一排好的序列, 稱之爲 X; 把 $y_2, y_3, \cdots, y_{\frac{n}{2}}$ 這一序列(並沒有排好)稱爲 Y。

在給出第三步前, 我們首先驗證把一數 (或一英文字) 插入一長度爲 k 排好的序列內, 需要比多少次。我們從另一方向來討論這個問題比較容易。比一次則顯然 k 只能是 1, 比兩次則 k 可以是 3, 先和中間一數比, 再視比較結果和頭或尾比。一般而論比 t 次, k 可以是 2 的 t 次方減 1, 方法是永遠先和中間一個比, 根據比較結果, 可以丟掉序的半段 (包括中間那個), 剩下

的問題是要插入那剩下長爲 2 的 $t-1$ 次方減 1 的半段, 可以用遞歸法斷定再比 $k-1$ 次即可插入。

第三步是要把 Y 中每一項逐次插入 X 中, 但插入的次序有點講究。先把 Y 分成若干組, 每一組由連續的 y 組成, y 小的組先被插入 X; 但同組的 y 則後面的先插。我們用下例來使讀者明瞭爲何如此。設 $n=6$, 則 $X=x_0,x_1,x_2,x_3$, $Y=y_2,y_3$ 。以後我們會說明 y_2 和 y_3 同屬一組。如果這一組先插 y_2 的話, 則因已知 y_2 先於 x_2, y_2 只需插入 x_0,x_1 這個子序列裏, 比兩次就夠了。但插入 y_2 後, X 就多了一項, y_3 就要插入一個含 x_0,x_1,x_2 及 y_2 四項的子序列裏, 需要三次才能插入。反之, 如果先插 y_3, 再插 y_2, 則容易檢驗二者都是插入一含三項的子序列裏, 各需二次比較, 省了一次比較!

剩下一個問題是如何分組, 亦即每組應含幾項。第一組每項要都能以二次比較插入 X, 很容易看出只能有 y_2,y_3 兩項。第二組每項要都能以三次比較插入 X, 也不難看出只能有 y_4,y_5 兩項。逐步計算, 則第三組可有六項, 第四組可有九項, 每一組有多少項都可推算出, 因此也可算出第三步所需比數。令 $g(n)$ 表示做 n 張卡問題時, 此比法第三步所需的比數。

當 n 是奇數時, 首先拿走一張卡而在剩下的偶數張卡上進行第一步和第二步。在第三步時, 把拿走的卡放入 Y 使其成爲 Y 的最後一項, 它也像其它 Y 中的各項被分組, 再按照規則按序插入 X 中。

現在我們可以用遞歸法來算出排序 n 張卡所需的比數了。第一步的比數是顯而易見的。第二步是遞歸的一步, 如果我們做 n 張卡的問題是眞的從一張卡、二張卡, 一直做到 $n-1$ 張卡, 則做 n 張卡的問題時, 自然第二步要用的比數已知。但當我們不想一磚一石從頭造起, 而想一步到位的話, 則必須有一些技巧, 有一些聰明。設 $f(n)$ 是此法排長度爲 n 的序所需的比較數, 則我們要能根據由經驗而得的一般數學智慧, 及對本題當 n 小的時候做練習所得的結果, 來猜出 $f(n)$ 的方程式, 這樣就可把第二步所需 f 的比數簡單寫成 $f(\frac{n}{2})$ 。當然, 最後要証明所猜的方程式的確是

$$f(n)=\frac{n}{2}+f(\frac{n}{2}+1)+g(n)\quad (n \text{ 是奇數時, } \tfrac{n}{2} \text{ 以 } \tfrac{n-1}{2} \text{ 代替})$$

的解答(式中三項代表了三步各需的比數, $g(n)$ 是已知數) 來確定猜測是對的, 且從 $f(n)$ 的表示式來計算所需比數。

三

這是我做的第一個離散數學的研究, 之前連離散數學的名字也未聽過。如果林牲正經八百的邀請我來一起做一道離散數學問題, 我說不定就以自己毫無背景而辭謝了。即使接受, 也一定趕緊去閱讀基礎入門的書籍, 不知讀到哪年哪月才能累積足夠的信心來做問題。現在竟不知不

覺的跨入一個新領域, 而且從此一跨再跨, 不懂的就問人, 還不懂就去翻書現學現用, 五年有成, 終於從統計所轉到了離散數學所。

我必須承認數學界山頭林立, 不是每個山都輕易讓過客進去開礦的。在這方面離散數學的確是一異數: 第一, 相對而言, 它是一門新數學 (尤其在我那時侯), 因此還有很多別人沒有走過的路。第二是它的應用性強, 尤其電腦是一個離散的機器, 它只會數零和一。所以電腦裏所用的數學基本上就是離散數學, 離散數學也因此隨著電腦應用的推廣而水漲船高。第三是離散數學的問題多半是直觀性的, 它比別的數學更著重解決問題而不是建立結構, 因此不需要穿過一個大廈的重門深院才能進入小姐閨房接觸問題。我之撞入離散國界, 一是幸運的進了 Bell Labs, 那兒聚集了一堆這方面的菁英, 可以從而學之 (從我的書和文章絕大多數都是和人合作可見我受益於人之深)。第二也是我天性接近這一行, 譬如我喜歡打橋牌, 而橋牌的技術就是或然率的理論, 且是離散的那一半。同時, 排橋牌比賽的程序, 像在第幾圈時你應坐在哪一桌和哪一對手打哪一付牌, 則是最難的設計問題, 現在猶未全解。目前做出的一些理論, 我和一些合作者有基礎的貢獻。

我們雖然憑著熱情一口氣的解了問題, 卻因沒有經驗犯了一個研究者大忌, 沒有去查文獻。等到要把論文寫出時, 才想到這一步。查完以後, 心情由雲端掉到谷底。原來在 1959 年, 我們想出的比法已經被發表了, 而且從此在學界被稱爲 Ford-Johnson 法, 冠上了兩位首先發表者的名字。我們等於只做了一道練習題而已。從此知道了一個研究題只有一個勝利者, 和人競爭除了鬥心智外, 還要搶時間。而到底先做問題還是先查文獻也是一個雞生蛋蛋生雞的問題。我個人的看法是研究者是被問題吸引的, 所以聽到一個好問題就無法遏止的去思考; 但是當做出部份結果有希望寫論文時, 就必須認眞的去搜索文獻以免步人後塵。這時做研究正是每天有收穫之時, 抽出做研究的時間去做枯燥的查文獻工作, 當然是非常不甘心的事, 但這是買保險, 不然你的研究就是沙上的圖畫。

就我而言, 這一段經驗倒不全是負面的, 首先是體會了做出題目時心底那一聲阿哈的歡呼, 再者, Ford 是一個有名的學者, 我能想出和他一樣的結果, 也與有榮焉, 知道自己有一定做研究的能力。另外, 做研究的特點就是解了一個問題後就有五個新問題產生。所以走過了一條路, 就必定更接近下一條路。我們很快的投入了下一個問題: 做了編電話薄引起的排序問題後, 很自然的考慮訂正電話薄引起的併序問題。電話公司要編 1968 年新電話薄, 但不想從頭做起, 就用上 1967 年用戶的排序, 先把離開的用戶名字勾除, 再把當年新加入的用戶名字排序 (不一定要這樣做, 但也是一種方法)。剩下要做的就是把新舊用戶的兩種排序合併成一個排序用在 1968 的電話薄裏, 稱爲併序問題。

四

這次林牲和我照規矩來, 先查了文獻確定沒有人做出什麼重要結果, 然後自己設計了一個

簡單的比法, 不但勝過所有別人提出之比法, 而且證明了它和理論極限所需之比數相差極小。計算理論的開山祖師 Knuth 在他的經典作 The Art of Computing 的卷七中, 用七、八頁的篇幅報導了這一個比法, 並稱之爲 Hwang-Lin 法。

Ford-Johnson 法和 Hwang-Lin 法是不是最佳法呢? 要證明最佳基本上就是要証明此法所需的比數等於理論極限, 而 Ford-Johnson 法在 $n = 12$ 時第一次和理論極限有違, 前者比數是 30 而後者是 29, 引起衆說紛紜。1965 年 Wells 靠著他強大的新電腦, 搜索了所有可能的算法 (排除不合理者), 給出了沒有一個比法可以只用 29 次做到的結論, 更引起 Ford-Johnson 法是最佳一說的衆矚塵上。但 1980 左右, Manacher 首先擴展了 Hwang-Lin 併序法, 再提出用併序法來做排序的問題, 就是先把 n 個名字分成兩群, 各自排序, 再用併序法把這兩個已排好的序合而爲一。他證明對無限多的 n, 這個方法勝過 Ford-Johnson 法, $n = 191$ 是最小的這樣一個 n 。

至於 Hwang-Lin 法, 易知它不是最佳併序法 (在特定 n 上可改進)。最佳法的討論走另外一條路。令 m, n 爲兩序之長且 $m < n$, 我們討論 m 小時的最佳法。$m = 1$ 時最佳法即前面討論過的把一數插入一數列之法。$m = 2$ 時, 1972 年 Hwang 和 Lin 發表了最佳法。$m = 3$時, 1980 年 Hwang 發表了最佳法, 已需用二、三十頁來敍述。另有二篇博士論文用了更長的篇幅給出同樣的結果, 其中 Murphy 的一篇, 也給出了 $m = 4$ 的結果, 可能因爲太長而未發表。看看這個討論的方向, 文章出得愈來愈慢, 篇幅愈來愈長, 我不鼓勵大家去做。如果要做的話, 可考慮下題 (此題我曾在數學傳播季刊第二卷第四期「整合與合併問題」懸賞四千台幣):

問題 1. 令 $M(m, n)$ 爲把長度分別爲 m 和 n, $m \leq n$, 的兩序列合併爲一所需最少的比數。証明

$$M(m + 1, n) \geq M(m, n + 1) \text{ 。}$$

五

一個問題被解決後, 就有了自己的生命。往往在很多年後, 在一個聽來無關的領域裏, 忽然出現一個新問題就像是你做過問題的翻版。這事在併序問題上也發生了。我們知道聖誕樹上燈泡是串成一線的, 一個壞了就整串都不亮了。碰到不亮時, 業者怎麼查出哪些是壞燈泡呢? 通常是選兩個燈泡 x, y, 在 x 和 y 上通電, 如果 xy 間這一串燈泡亮了, 就表示 xy 間沒有壞燈泡, 不亮就是至少有一個壞的, 就要想出一個方法來繼續進行。雖然表面上好像和併序是完全不同的問題, 但是兩者的數學程式是一樣的, 把所有的好燈泡看成一個序列, 所有的壞燈泡看成另一個序列, 則一個問題的解法可以翻譯成另一個的。對 x 到 y 這一串燈泡同時的測試稱爲群試。

當然燈泡也許不值錢, 一整串不亮, 丟掉就是, 不值得去找壞的。但群試其實主要是用於醫學測試上, 最早是徵兵時驗血, 找出那些有性病的入伍者淘汰掉。如果一一測試, 十萬人就要十

萬次, 用群試法, 則只是幾百次 (群試是把多人的血液樣本抽一些混在一起試, 如通過則這群人一起通過), 現在則已在非洲多國用來測試 AIDS 患者。我早先就做群試問題, 不過是或然率型的, 此時看到了離散型的群試問題可以借用併序法的成果, 即大力介紹這一新觀點, 及因而得到的新成果, 前後寫了文章和書。群試又可以用到通訊網路上, 也寫了文章, 拿了專利。最近又用在 DNA 測試上 (寫了一本書)。所以你看我的第一次數學研究一個失敗的經驗竟引出了後面這麼多好事。做研究眞好!

誌謝: 本文承陳宏賓博士改寫成符合投稿規定的 pdf 形式, 深表感謝。並謝邵子凡、任元兩學長閱讀初稿, 賜以鼓勵。

參考資料

1. Ford, L. R. Jr. and Johnson, S. B., A tournament problem, *Amer. Math. Monthly*, 66(1959), 387-389.
2. Hwang, F. and Lin, S., A simple algorithm for merging two linearly-ordered sets, *SIAM J. Comput.*, 1(1972), 31-39.
3. Knuth, D., *The Art of Computer Programming*, Vol.7, Addison-Wesley,1975, 6, 2, 4.
4. Wells, M., Note on the fact that $S_{12} > \log 12!$, *Proc. IFIP65 Congress*, 2(1965), 497-498.
5. Hwang, F. and Lin, S., Optimal merging of 2 elements with n elements, *Acta Informatica*, 1(1971), 148-158.
6. Hwang, F., Optimal merging of 3 elements with n elements, *SIAM J. Comput.*, 9(1980), 298-320.

—本文作者爲國立交通大學應用數學系講座教授 (已退休)—

小品

短文一

作者: 蘇珮瑄、陳若瑜

一個零對另一個零說:「我們在一起吧! 我許你一個無限大的未來。」

—本文作者爲中國科技大學行銷與流通管理系學生—