

# 漫談 PCP 定理

吳邦一, 2006/8

小明與小華玩遊戲，小明：「你現在寫下一個兩位數，我有特異功能可以猜到你寫下的數字」，小華於是寫下數字「12」，小明也將他所猜的數字寫在紙上。現在，小明必須說服小華他真的猜對了，而小華則是需要驗證小明是否真的猜對了他的答案，最簡單的方式當然就是小華問小明十位數多少以及個位數是多少，小明如果答對了就真的可以說服小華他是具有特異功能的，反之如果答錯了，也瞞不了小華。

現在我們加一點限制。假設小華只能要求小明回答一個介於 0~9 的數字（也就是一個 digit），小華這樣問：「個位數是多少？」，如果小明回答：「2」小華就相信他，而如果回答的不是「2」小華就選擇不相信他，當然小華也可以選擇問對方十位數，情形也是類似。

我們可以檢查一下，如果小明真的猜對，無論小華如何問，他都會答對，而也就可以說服小華；但是如果他事實上是個騙子，不知道答案的，那麼有多少機率小華會誤以為小明知道答案呢？當然，在此過程中小明是不知道小華的謎底的，所以我們可以假設他會老實的回答小華的問題。以上面小華這種問法，讓我們來算算機率。假設小華在選擇十位數或個位數時是以相等機率隨機選擇的，如果小明的答案的兩個數字都是錯的，那麼無論小華選到哪個數字來驗證都不會受騙；如果小明的答案有一個是錯的另一個是對的，例如說他的答案是 52，那麼小華被騙的機會是  $1/2$ ，因為如果他選到的是錯的（十位數）他就不會被騙，但是如果選到對的數字（個位數），他就會被騙了。

總而言之小華被騙的機率不會超過  $1/2$ 。

以上這個例子就是所謂 PCP(Probabilistically Checkable Proof)的一個描述。在上述例子中，小明是一個提供證明（proof）的人而小華是一個驗證者（verifier），當然，比較正確的講法是一個計算機器，我們這裡把他擬人化了。一個 PCP 系統的中心要角就是這個驗證者，驗證者（小華）的目的是要判斷某件事情是屬實（小明是個有透視心靈的超能力者），針對一個證明 P，驗證者可以使用某些隨機數，然後根據這些隨機數來檢查 P 的某些位置（十位數或個位數），並經過計算（比對）後，判斷是否屬實。我們定義要求驗證者必須做到下列要求：

1. （completeness）如果那件事是真的，必然存在某個證明 P，使得驗證者不管

用哪些隨機數做選取的依據，驗證者都會被取信，也就是說，如果是對的一定存在一個證明，不管怎麼驗證都對而不會被誤判。

2. (soundness) 如果那件事是錯的，對於任何錯的證明，驗證者根據各種可能的隨機數來抽樣檢查而誤判的機率是低的（不超過  $1/2$ ）。

這樣的一個 PCP 系統有何作用呢？我們可以拿驗證者所需要的資源來做為「問題難度的評量標準」。如果需要很多的資源才能達到上述的驗證標準，代表這個問題很難；如果只要一點點資源就可以驗證，則表示是個簡單的問題。

在 NP 的分類方式中，我們在乎的是一個問題需要多少時間去驗證，如果是多項式時間複雜度可以驗證的問題，就是屬於 NP。那麼在 PCP 系統中我們考量的資源是什麼呢？我們考量的有兩項：第一是驗證者使用的隨機變數個數 ( $r$ )，其二是驗證者需要查驗位置個數 ( $q$ )。在正式了理論中我們是以 bit（一個 0 或 1 的數字）來做為單位的，跟上面的例子有一些不同，但是兩者之間只差若干倍，不影響我們的說明。

如果一個問題存在（可以找得出）一個驗證者只需使用  $r$  個隨機變數而查驗  $q$  個位置就可達到驗證的要求標準，我們就稱該問題是屬於  $PCP[r,q]$ 。這裡要稍微提醒， $q$  和  $r$  不一定是常數而可能是輸入的大小  $n$  的一個函數。以 NP 的觀念來看，我們可以說 NP 是屬於  $PCP[0, poly(n)]$ ，這裡的  $poly(n)$  是指他是個  $n$  的多項式函數，原因很簡單，在 NP 的定義中我們不使用隨機數，而查驗了  $poly(n)$  個位置，為何是  $poly(n)$ ？一個問題輸入大小是  $n$ ，他猜測出來的結果（proof）可能是  $poly(n)$  這麼大，他不會超過  $poly(n)$ ，否則我們就無法在多項式時間內查驗完畢。

1990 年開始的一段時間，PCP 有很大的進展而發展出一個很驚人令人意外的結果，PCP 定理現在是這樣的：

$NP=PCP[O(\log n),3]$ 。他的意思是說：對於一個 NP 的問題，存在一種寫下證明的方法以及一種機率驗證的方法，使得驗證者只需要讀 3 個 bits 來達到具有足夠的信心判斷該證明是真是假。

多麼神奇啊！你能夠想像一個學生做了一個題目而做老師的只要驗證他的答案的 3 個 bits 就能判斷答對或是答錯嗎？

關於 PCP 定理，我們總會疑惑，為何可以檢查少數的位置就可以有足夠的信心判斷真偽，這結果的確超乎我們一般的常識與想像，因為少數的位置正確，但有更多的位置並未被檢查到啊！因為超乎常識而讓我們很難想像。我們以下列的例子或許可以了解其背後的原因，我們來看看前述小華的例子，在那個例子中，小明只被允許回答一個 0~9 的數字，而我們的計算顯示，錯誤的證明而被接受的機率不超過  $1/2$ ，我們是否有別種詢問的方式可以減少誤判的機率呢？我們

來看看下面的做法：

小華要求小明將他所猜的二位數的兩個數字相加(以上述的例子得到 3)，相加的結果如果超過 10 就只取其個位數，然後將所得到的數字寫在原來的數字之前形成一個三位數。例如，小明的答案如果是 52，就會得到 752；如果答案是 38，就會得到 138。接著小華隨機挑選此三個數字其中之一來驗證，如果驗證結果正確就接受，否則就不接受。

我們來檢視一下小華誤判的機率。如果小明的答案是對的，無論他檢查哪一個都是對的而不會誤判，如果小明是錯的呢？我們要分兩種情形來計算。

如果小明的答案有一個數字是錯的，例如 52，那麼他做出來的數字會是 752，三個數字其中會有兩的數字是錯的，因此小華誤判的機率是  $1/3$ 。如果小明的兩個數字都是錯的，例如 94，那相加得到的數可能會正確，本例中為 394，但是原來兩個位置會是錯的，因此小華誤判的機率也是  $1/3$ 。結論是：小華依舊只檢查一個位置，但是誤判的機率可以從  $1/2$  降低到  $1/3$ 。

神奇嗎？其實這是運用了資訊通訊技術中的錯誤偵錯編碼的原理，其根本上的道理就是：運用編碼的方式，可以將局部的資訊擴散到更廣泛的範圍。如此一來，如果原來錯誤在少數地方，就可以在更多的地方出現，而使得更容易被偵測出來。因此很不正式的說，對於 PCP 定理我們有一個重要的認知就是：誤差是可以被某種程度放大的。

PCP 定理雖然有趣，但是我們還是不禁要問：他有什麼用途，能幫助我們什麼呢？不知是幸或是不幸，PCP 定理目前最主要的用途是可以用來證明某些問題不可能存在有效率的近似演算法。一個從 PCP 定理直接得到的結果就是 MAX 3SAT 問題不存在 PTAS（一種可以無窮逼近的近似演算法）。

所謂的近似演算法，是針對最佳化問題來說的。一個最佳化問題是要最大化或最小化某一個目標函數，對於一些已被證明為 NP-hard 的最佳化問題，我們不再尋求快速的演算法可以計算出最佳解，於是退而求其次去設計快速的演算法可以得到近似於最佳解的演算法。一個演算法如果可以保證對任何的輸入，其所求出的解與最佳解的相對誤差不超過  $r$  倍，我們稱呼他為一個  $r$ -approximation 的演算法，我們努力去尋求好的（誤差倍率小的）近似演算法，在有些問題上得到很好的結果，但是在很多問題上，又再度遇上困難，我們找不到好的近似法，但也無法證明出不存在那樣的演算法。在 PCP 定理逐步發展的過程中，越來越多的負面結果（某些問題無法近似到某種程度）被快速的證明出來。

PCP 定理目前所扮演的角色，有點類似 Karp 所證明的 21 個 NPC 問題。Karp 的成果提供了證明某些問題屬於 NPC 的方法，而 PCP 提供了一種方法來證明近

似演算法的負面結果。這些證明都挺複雜，但是我們可以稍微了解一些背後的原理。

這些證明結果大多是類似這樣的陳述：「除非  $NP=P$ ，否則存在某個  $r$  使得某某問題不可能存在有效率的  $r$ -approximation 演算法」。其證明的關鍵在於運用 PCP 定理中正確答案被接受的機率（completeness）與錯誤答案被誤判的機率（soundness）兩者之間的差距，記得嗎？這個誤差可以某種程度的被放大，而在證明過程中，這個差距被轉換為近似解與最佳解的差距，因而推導出如果存在若干倍以內的近似演算法則所求出的解根本就會是最佳解，因而所有  $NP (=PCP[O(\log n), O(1)])$  的問題均會在  $P$  中。

---

#### 【參考資料】

拜網路百科之賜，我不必列出相關了文獻，相關資料可以在下列網址查到：

[http://en.wikipedia.org/wiki/Probabilistically\\_checkable\\_proof](http://en.wikipedia.org/wiki/Probabilistically_checkable_proof)