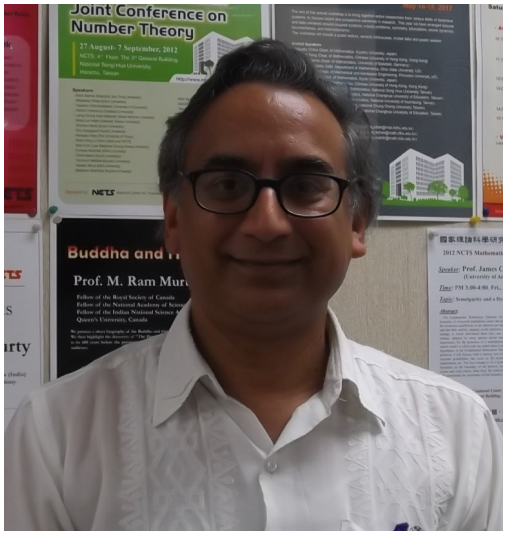
數學傳播　36卷3期, pp. 12-23

How Google Works——搜尋引擎中的線性代數



演　講 : M. Ram Murty 教授
時　間 : 民國101年8月17日
地　點 : 清華大學
　　國家理論科學中心
整　理 : 陳麗伍

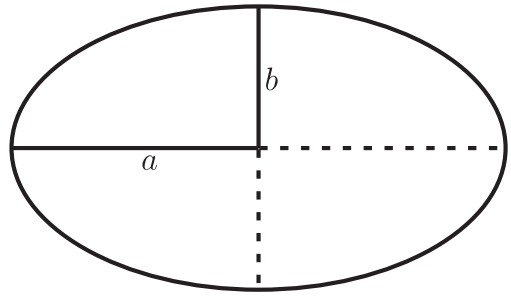
介紹 (國家理論科學研究中心數學組李文卿主任): 大家好, 很高興爲大家介紹皇后大學的 Ram Murty 教授, 他是理論中心的訪問學者, 除了精闢的數論演講以外, 同意做兩場一般性的演講。今天是第一場演講, 第二場在下星期三同一時段同一地點。Murty 教授是一位出色的數學家, 還是一位詩人, 他寫詩算數, 不僅對數學有廣泛的興趣, 在其它領域也有廣泛的涉獵, 哲學就是其中之一。他是皇后大學的研究講座 (Research Chair), 加拿大皇家學會的會士及印度國家科學院院士。我們很幸運他答應給這個演講。今天的講題是"How Google Works"。讓我們歡迎 Murty 教授。

謝謝。能爲各位演講是我的榮幸。這個演講是給一般聽衆準備的科普演講, 只需要了解一些基本的線性代數。演講的小標題就是搜尋引擎中的線性代數。

1995年以前, 搜尋引擎是透過關鍵字的搜尋來運作。但是 Google 搜尋引擎決定以數學計算來決定單一網頁的重要性。特定網頁是重要還是不重要該如何計算? 現在, Google 已經成爲一個名詞, 例如:"堤米, 這些是百科全書, 這是 Google 的前身。" 或是:"卡住了, 查一下 Google。" Google 也是動詞, "你這個問題把我難倒了, 你需要 Google 一下。" Google已經成爲有如聖諭一般的存在。但是依靠網路尋求答案其實是非常危險的。

感謝國家理論科學中心數學組李文卿主任及 Ram Murty 教授同意本刊刊載 Murty 教授於清華大學理論科學中心的演講紀錄。

舉例來說, 有一個叫做 gomath.com[1] 的網站, 是學習基本數學知識的網站, 在這個網站上你可以找到計算橢圓的面積與周長的如下的公式:



$$\text{橢圓:}\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$$

$$\text{周長}=2\pi\sqrt{\frac{a^2+b^2}{2}} \qquad \text{面積}=\pi ab$$

如果你做微積分的作業, 會發現第二個答案是正確的, 第一個答案則完全錯誤。幸好如此, 不然我們就沒有橢圓函數理論與橢圓積分, 因爲計算橢圓周長的公式是由橢圓積分推導出來, 而複雜豐富的橢圓積分理論與代數幾何相關。

引用錯誤的資訊可以造成毀滅性的結果, 譬如以下的例子。1999年9月30日 CNN 有這樣一則新聞"計量災難造成火星探測器損失"[2]。 現在在火星的探測器不是當初發射的探測器。這個失敗的探測器原本應該在1999年9月30日抵達火星, 但是因爲一個簡單的數學錯誤造成它完全消失。CNN新聞說"根據報告, 美國太空總署因爲一組 Lockheed Martin 的工程團隊在輸入參數時使用了英制單位, 而不是政府團隊所使用的公制單位, 造成一個價值一億二千五百萬美金火星探測器的失聯。因爲計算單位的不同使得位於丹佛的 Lockheed Martin 工程團隊與位於帕薩迪納美國太空總署火箭推進實驗室的飛航方向資料無法傳輸。" 所以億萬美元就這樣不見了。這只是探測器的造價, 還不包括研究人員的薪資、心力以及花費的時間, 所以一定要萬分謹慎。我相信他們絕對從中學到一個很好的教訓, 這顯現了 Google 的一些限制。至於另外一個侷限, 我們來看下面的對話:

老人:「我花了一生尋找人生的意義。」

小孩:「我爸說如果 Google 找不到, 應該就找不到了。」

我可以打包票你用 Google 絕對找不到生命的意義, 這可以說是一個終極的問題。

---

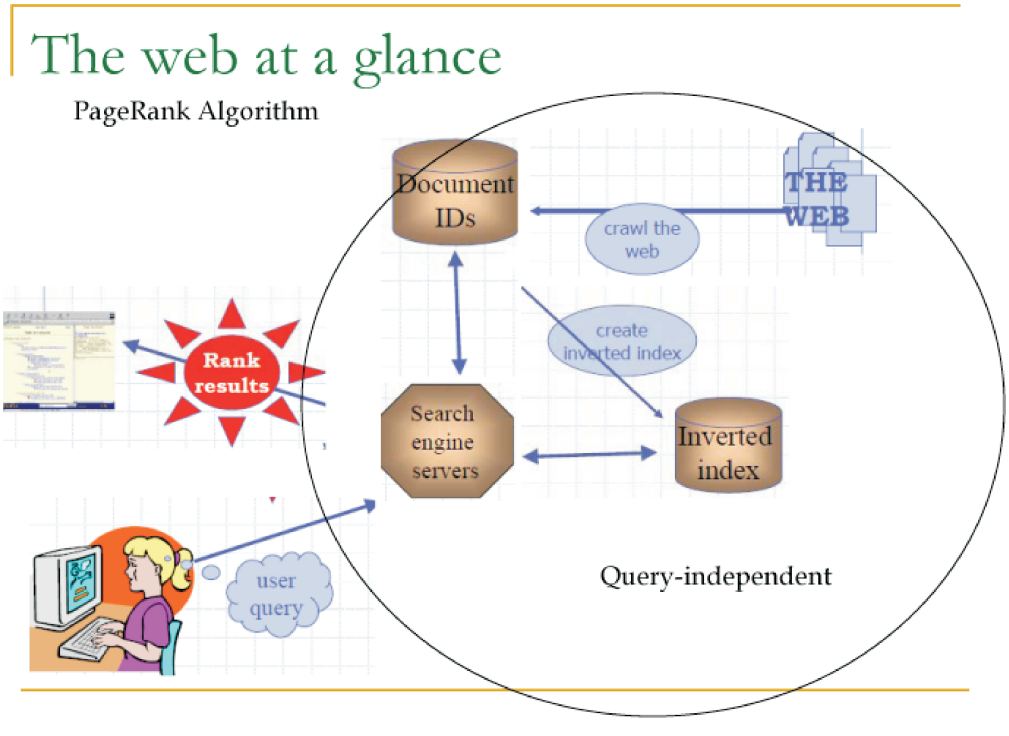
[1] http://gomath.com/geometry/ellipse.php.

[2] http://articles.cnn.com/1999-09-30/tech/9909_30_mars.metric_1_mars-orbiter-climate-orbiter-spacecraft-team?_s=PM:TECH.

現在讓我試著大略解釋一下全球資訊網 (World Wide Web) 的運作。實質上這是一張大連結圖。這張圖有兩個特點。其一, 使用者提出一個查詢後, 搜尋引擎開始針對這個查詢工作。搜尋引擎不停派出虛擬機器人 (cyberbot) 到各網站收集每個網頁的文件識別碼 (document id) 與關鍵字, 並且把這些資料儲存起來。使用者提出的查詢利用這些儲存起來的資料, 根據重要性排出順序。概括的說這就是 Google 的運作過程。

Google有兩個部份。一個是與查詢無關固定的部分, 另一個就是我要講的, 以 PageRank (網頁級別) 演算法爲基礎與查詢相關的部分。數學就是透過 PageRank 演算法進入 Google 的運作。至於虛擬機器人不斷地到各網頁紀錄拍攝關鍵字, 儲存分類文件識別碼後以關鍵字建立倒排索引 (inverted index) 隨時提供存取, 則是非數學的部分。

當使用者鍵入查詢 google 特定的字。你必須把所有有這個字的連結找出來, 接著依照重要性排序。問題就在什麼是重要? 該如何利用數學分析排出網頁的重要性以取代單純的關鍵字查找? 以前是利用關鍵字出現的多寡決定, 網頁上關鍵字出現的頻率越高就越相關 (relevant) 或越熱門。Google 演算法有另一個計算相關性 (relevance) 的操作。

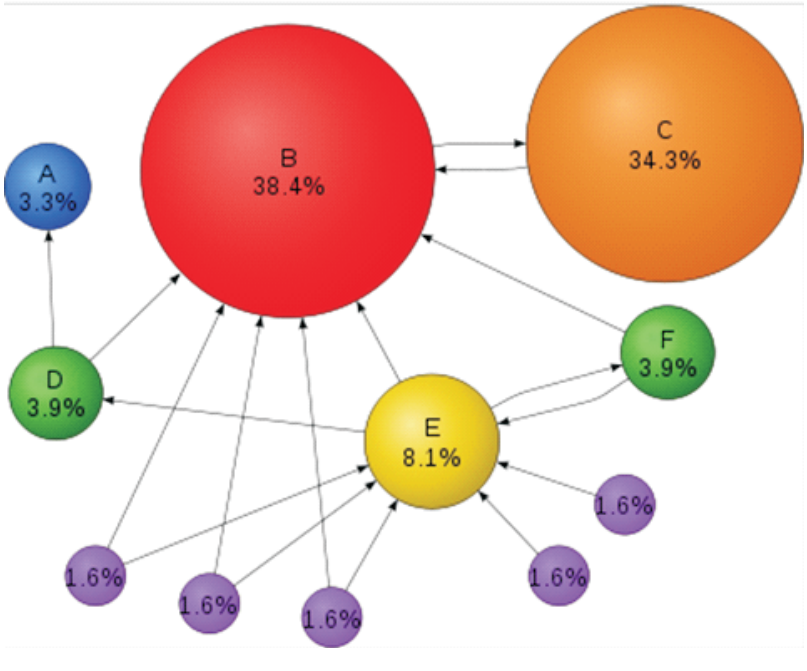


所以網路其實是一張很大的圖, 網頁就是節點 (nodes) 或頂點 (vertices)。每一個網頁代表一個頂點, 連到這個網頁的連結或從這個網頁連出去的連結就是邊 (edges), 由內指標 (in arrows) 或外指標 (out arrows) 表示, 這是一個有向圖 (a directed graph)。這張

圖有超過十億的頂點而且每秒持續成長, 理所當然是一個巨大的圖。這個圖應該不會是計算搜尋時使用的圖。當有人提出查詢, 對應到這個特定碼字 (code word) 的一些 URL 就會被找到。只需要用這些節點與 URLs 來建構一張有向圖, 而不必使用代表整個網路的有向圖。有向圖的大小經由這個過程而縮減。

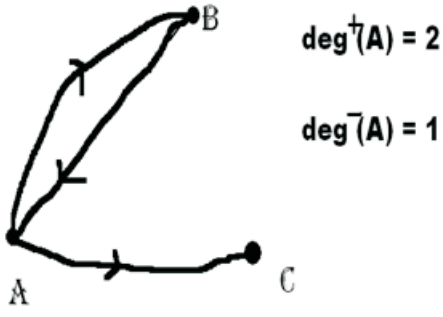
PageRank 演算法是由兩位輟學的研究生 Sergey Brin 和 Larry Page 發現。PageRank 聽起來像是排序網頁 (ranking of pages), 這是一個巧合, 其實是 Larry Page 的姓。他們評等網頁重要性的數學依據如下: 他們提出一個定理, 如果有其它重要網頁指向一個網頁, 這個網頁就是重要的。這聽起像是一個循環定義。如果有一百個人說一個人是偉大的, 我想這個人是偉大的。這就是這個定義的基礎。可以透過矩陣的方式以很簡單且基本的數學表示。他們發明的這個演算法在2001年取得專利。我要解釋的數學現在可以說是衆所皆知, 其它的搜尋引擎也已經在使用類似的演算, 受到專利保護的版本會更詳細, 其中的秘密當然無法得知, 不過基本的概念如下。

這是一個例子。假設有這麼多網站, 然後依照其重要性放大尺寸, 這些網站互相連結有如一個有向圖。我們可以看到, 雖然節點 E 有許多指標 (arrow) 指向它, 但節點 C 的重要性卻高於節點 E。這是因爲節點 B 很重要, 然後由節點 B 指向節點 C 的指標讓節點 C 重於節點 E。下面是這個例子的圖示, 這就是決定重要性的方法。



那該如何透過數學表示呢? 先做一個有向圖並假設他有三個頂點 A, B 和 C。從 A 點連外的三個邊中, 有二個邊向外, 一個邊向內。所以頂點 A 的出次數 (out degree) 是 2,

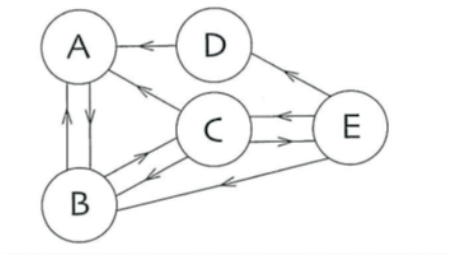
進次數 (in degree) 是 1。



我們用 $r(J)$ 來代表網頁 J 的級別 (rank), 對任何頂點 K, 它的級別是由所有指向它的頂點來決定。所以如果頂點 J 指向頂點 K, 那麼由頂點 J 指向頂點 K 的機率應該是出次數分之一 (1 over the out degree)。也就是可以由出次數知道有幾個邊是從頂點 J 出發的。

網頁 J 有多重要, 我們可以透過以上的敍述將級別做一個循環定義。所以一開始的定義:"如果有其它重要網頁指向一個網頁, 這個網頁就是重要的", 可以用以下的數學公式表示:

$r(K) = \sum_{J\to K} r(J)/\deg^+(J)$ $\deg^+(J)$ 是 J 的出次數



一旦有了這個, 我們馬上可以定義一個矩陣, 可以用馬可夫鍊 (Markov chain) 表示。我們可以用下面的例子說明: 以微型全球資訊網爲例, 假設這個資訊網只有5個節點, 出、入次數如圖。讓 p_{uv} 爲由節點 u 到節點 v 的機率。讓我們來看以下的機率:

由節點 B 到節點 A 的機率有多少?

節點 B 只有 2 個邊向外, 一個邊指向節點 C, 另一個邊指向節點 A。所以, 由節點 B 到節點 A 的機率是 $1/2$。由節點 B 可以到節點 C 或節點 A, 所以 $p_{AB} = 1/2$。

那由節點 C 到節點 A 呢?

由節點 C, 有三個邊向外指。所以由節點 A 到節點 C 的機率是 $1/3$ ($p_{AC} = 1/3$), 因爲節點 C 只有 3 個向外指的邊。由節點 E 到節點 A 的機率是0 ($p_{AE} = 0$), 因爲沒有連

結的邊。所以產生了以下的轉移機率 (transition probabilities)。將這些轉移機率用矩陣表示, 如下圖。

$$P = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{3} & 1 & 0 \\ 1 & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & 0 & 0 \end{pmatrix} \end{array}$$

注意到每個行的總和永遠是1。這些行代表什麼意義呢? 它代表的是每個節點可以連接到的節點。如果你有 n 個邊由一個頂點向外指, 那麼由這個頂點連接到其它頂點的機率是 n 分之 1。每個頂點連接到其它頂點的機率加起來總和一定是 1。所以, $(1\ 1\ 1\ 1\ 1)P = (1\ 1\ 1\ 1\ 1)$。$P^t$ 的特徵值 (eigenvalue) 是1, 特徵向量 (eigenvector) 是 $(1\ 1\ 1\ 1\ 1)$。P 是馬可夫鍊理論 (Markov Chain Theory) 中的移轉矩陣 (transition matrix)。它代表了由 A 點到 B 點的機率的資訊。

讓我們來看一下馬可夫程序 (Markov process)。回到我們的圖, 問一個問題, 如果有一個使用者現在在網頁 C, 他在點擊1次 (1 click) 後會在哪裡? 點擊 2 次 (2 clicks) 後會在哪裡? 點擊 n 次 (n clicks) 後會在哪裡?

把起始值的條件 (initial condition) 用行向量 (column vector) 表示。

$$p^0 = \begin{pmatrix} p(X_0 = A) \\ p(X_0 = B) \\ p(X_0 = C) \\ p(X_0 = D) \\ p(X_0 = E) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

1代表現在在網頁 C, 也就是我們在節點 C。

點擊一次後發生了什麼事? 點擊一次後, 使用者會到哪裡? 我們雖然不知道會發生什麼事情, 但是我們可以從下面知道發生的機率。

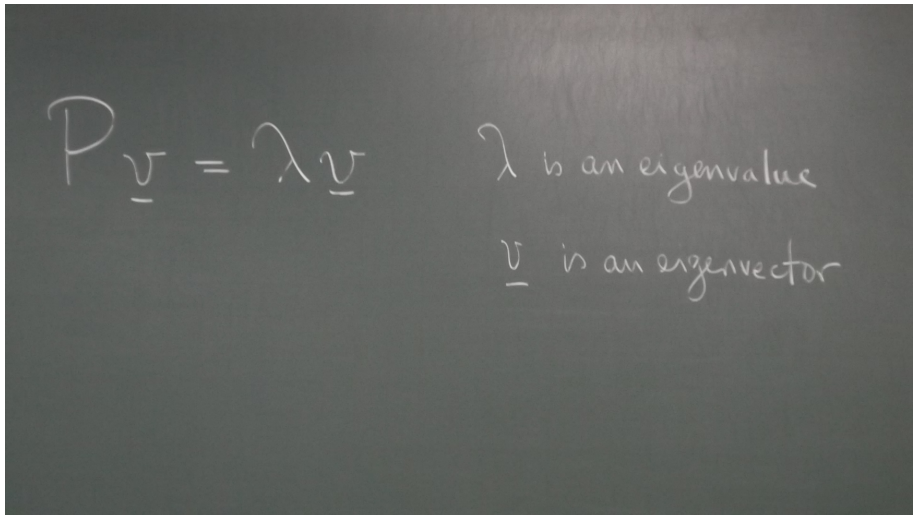
$$p^1 = \begin{pmatrix} p(X_1 = A) \\ p(X_1 = B) \\ p(X_1 = C) \\ p(X_1 = D) \\ p(X_1 = E) \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{3} & 1 & 0 \\ 1 & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ 0 \\ 0 \\ \frac{1}{3} \end{pmatrix}$$

所以點擊2次後, 在第一個節點的機率是$1/6$, 第二個節點的機率是$4/9$, 第三個節點的機率是$5/18$, 第四個節點的機率是$1/9$, 第五個節點的機率是0。

$$p^2 = \begin{pmatrix} p(X_2=A) \\ p(X_2=B) \\ p(X_2=C) \\ p(X_2=D) \\ p(X_2=E) \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{3} & 1 & 0 \\ 1 & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ 0 \\ 0 \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \frac{1}{6} \\ \frac{4}{9} \\ \frac{5}{18} \\ \frac{1}{9} \\ 0 \end{pmatrix}$$

所以你可以看到, 持續重複 (iterate) 這個馬可夫矩陣 (Markov matrix), 在點擊 n 次後, 你會得到 $P^n p^0$。這就是主要的概念。

那麼矩陣 P 的特徵值與特徵向量在什麼地方出現呢?



$P\underline{v} = \lambda\underline{v}$, λ 是特徵值, $\underline{v}$ 是特徵向量

在矩陣理論 (Matrix Theory) 中矩陣的特徵值與特徵向量因爲不同的原因而重要, 在這裡是以很基本的方式出現。讓我們在計算前先提示一些基本的要點。

任何矩陣都有一個特徵多項式 (characteristic polynomial)。每個特徵多項式的根 λ 都是特徵值。

$$\Delta_{P^t}(\lambda) = \det(\lambda I - P^t) = \det(\lambda I - P)^t = \det(\lambda I - P) = \Delta_P(\lambda)$$

這個多項式一定有根。P 的特徵多項式與它的轉置矩陣 P^t 的特徵多項式是一樣的, 因爲特徵多項式是由行列式 (determinant) 決定的。而一個矩陣 A 的行列式就是 A 的轉置矩陣的行列式。所以 P 的轉置矩陣的特徵多項式就是 P 的特徵多項式。所以矩陣 P 與 P 的轉置矩陣有一樣的特徵值。

已知 P 的轉置矩陣有特徵值 1 與特徵向量 (1,1,1,1,1)。依循上面的邏輯, 我們知道 1 也是 P 的特徵值。但是這並沒有告訴我們任何有關特徵向量的訊息。它只提供了特徵值的訊息。P 和 P^t 有一樣的特徵值, 卻不見得有一樣的特徵向量。這是一個要點。雖然如此, 但是因爲 Frobenius 的一個關於矩陣的有名定理, 我們可以知道特徵值爲 1 的 P 的特徵向量。Georg Frobenius (1849∼1917) 是一位很有名的代數數論家。

注意到特徵值可能是複數, 不過在我們的轉移矩陣 (transition matrix) P 中, 所有的特徵值都是絕對值 (absolute value) ≤ 1 的實數, 並且有一個特徵值是 1。我們的問題就是要找出對應這個特徵值的特徵向量, 而 Frobenius 的定理告訴我們每一個特徵值都有一個對應的特徵向量, 而且這個向量所有的分量都是非負的數。這就是2百50億美元的特徵向量, 支撐著 Google 的祕密。如果要做進一步的閱讀, 有一篇很有意思的文章刊登在 2006 年 SIAM Review 解釋 Frobenius 的定理如何改變了這個世界。

同時這也與 Perron 有關。Oskar Perron (1880∼1948) 是一位數論家。他有一個有名的數論的定理 — Perron 的定理。Perron 將 Frobenius 的定理改進了一些, Perron定理的敍述如下: 令 A 是一個只有正數 entries 的方陣 (square matrix)。讓 $\lambda^* = \max\{|\lambda| : \lambda$ 是 A 的特徵值$\}$。那麼 λ^* 是 A 的一個重數 (multiplicity) 爲 1 的特徵值, 並且有一個所有 entries 都是正數的特徵向量相對應於 λ^*。甚至, 對於其它任何特徵值 λ, $|\lambda| < \lambda^*$。

很遺憾我們的移轉矩陣並不符合這個假設。定理的假設要求的是只有正數 (positive entries) 的方陣。因爲條件不符, 所以我們無法應用這個定理, 因此 Frobenius 又出現了。

Frobenius 改進了 Perron 的定理, 提出不可約矩陣 (irreducible matrices)。一個矩陣 A 是不可約的如果存在某個自然數 n 使得 A^n 的 entries 都是正數。

假設我們由這個移轉矩陣開始, 一些 entries 是0, 一些是正數。但如果圖是連通的, 經過幾次點擊 (clicks) 所有的 entries 都會變成正數, 不然你無法由點 A 連到點 B。基本上就是這樣。而我們的移轉矩陣正符合需要的條件, 所以可以應用這個定理。如果矩陣 A 是一個非負數 entries 的不可約方陣, λ^* 是所有特徵值的最大絕對值, 定理告訴我們 λ^* 同時也是矩陣 A 的特徵值, 其重數爲1。此外, 還有一個分量全爲正數的特徵向量與其對應。所以這是一個線性代數的基本定理。

身爲數學家, 我不需要知道這個定理能不能被實際應用在眞實世界, 因爲這個定理本身就是一個漂亮的定理, 是關於特徵值與特徵向量的一個很有意思的描述。讓我們來看看這個理論的實際應用。

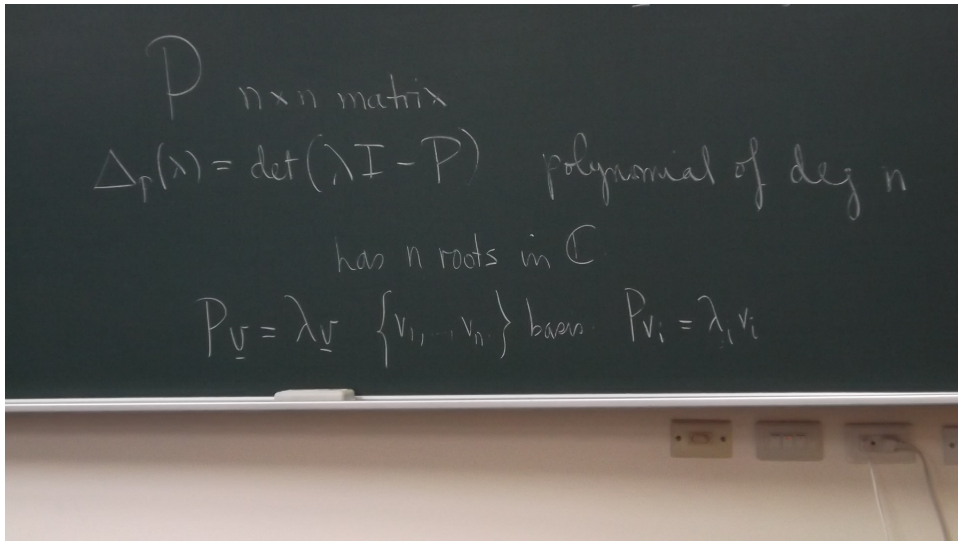
爲了簡便, 讓我們假設矩陣 P 符合以下的條件:

a. P 只有一個特徵值其絕對值是 1 (也就是 $= 1$),

b. 特徵值 1 相對應的特徵空間 (eigenspace) 的維度是1,

c. P 可對角化 (diagonalizable), 也就是特徵向量構成一個基底。

讓我爲不熟悉的人解釋一下。



如果有一個 $n \times n$ 矩陣 P, P 的特徵多項式是一個 n 階的多項式, 代數基本定理告訴我們這個多項式, 有 n 個根, 這些根就是矩陣 P 的特徵值。假設每個特徵值有一組非零向量 v, 同時 Pv 等於 λv。對於每個 P, 假設有一組特徵向量組成的基底, 也就是說, 有 n 個滿足 $Pv_i = \lambda_i v_i$ 的彼此線性獨立的向量。接著假設有一組這類型的基底。這不是一個嚴格的假設, 但是之後會改善。

在這假設之下, 有一個唯一的特徵向量 v 使得 $Pv = v$, 而且 v 有非負數的分量, 分量的總和是1。

我們現在說的是線性代數的基礎定理。而 Google 搜尋引擎如何應用這個定理呢?

前面提到的 Frobenius 的定理導出所有其他特徵值的絕對值絕對小於1。這是很重要的一點。接下來從前面觀察到的性質, 有一個特徵向量的分量總和是1。讓我們計算如果起始點在 C, 在 n 次點擊後, 會到哪裡?

讓 $v_1, v_2, \ldots, v_5$ 是矩陣 P 由特徵向量組成的基底, v_1 對應到最大特徵值 1 且 v_1 的分量的和爲 1。因爲 $v_1, v_2, \ldots, v_5$ 是基底, 一個給定的向量 p^0 可以寫成基底的線性組合 $p_0 = a_1v_1 + a_2v_2 + \cdots + a_5v_5$。這些 $a_1, a_2, \ldots, a_5$ 是複數 (complex numbers)。我需要證明 $a_1 = 1$, 證明如下:

$$p^0 = a_1v_1 + a_2v_2 + \cdots + a_5v_5$$

記得向量 $J = (1, 1, 1, 1, 1)$, 將等號的兩邊同時乘上列向量 J, 則

$$Jp^0 = a_1 Jv_1 + a_2 Jv_2 + \cdots + a_5 Jv_5 \text{ 。}$$

根據我們的假設 $Jp^0 = Jv_1 = 1$,
於是我們得到 $a_1 + a_2 Jv_2 + \cdots + a_5 Jv_5 = 1$ 。
另一方面, 對於 $i \geq 2$, $J(Pv_i) = (JP)v_i = Jv_i$, 因爲 P 的每一行總和都是 1 。
但是 $Pv_i = \lambda_i v_i$,
所以 $\lambda_i Jv_i = Jv_i$ 。
因爲 $\lambda_i \neq 1$, 我們得到 $Jv_i = 0$,
所以 $a_1 = 1$ 。

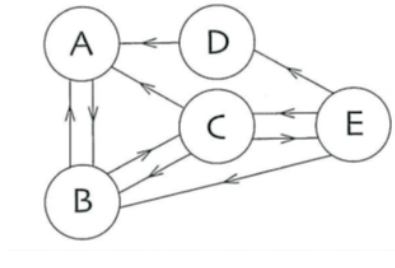
在 n 次疊代 (iterations) 後:

$$\begin{aligned} P^n p^0 &= P^n v_1 + a_2 P^n v_2 + \cdots + a_5 P^n v_5 \\ &= v_1 + \lambda_2^n a_2 v_2 + \cdots + \lambda_5^n a_5 v_5 \text{ 。} \end{aligned}$$

因爲所有這些特徵值 $\lambda_2, \ldots, \lambda_5$ 的絕對值都嚴格小於 1, 當 n 逼近無限, 我們得到 $P^n p^0 \to v_1$ 。

也就是不管起始分佈 p_0 是什麼, 馬可夫程序的穩定向量 (stationary vector) 是 v_1。方程右邊與起點是獨立的。所以這是一個不可思議的定理。這個與 Google 完全沒有關係, 與線性代數是息息相關。當你用一個移轉矩陣的機率, 不停重複後, 得到一個穩定向量。這個穩定向量是 v_1。也就是我們討論的重點, 不管 p_0 是什麼, 馬可夫程序的穩定向量是 v_1。

回到我們的例子, 當計算矩陣 P 對應到特徵值 1 的特徵向量時, 得到的向量是 (12, 16, 9, 1, 3)。這不是一個唯一 (unique) 的特徵向量, 它的任何倍數均爲對應到特徵值1的一個特徵向量。也就是說, 這一定是馬可夫程序對應的穩定向量。所以由這個結果說明網站 B 比網站 A 重要。我們可以把這個除以41做正規化, 所以分量 (components) 的和是1。由這個特定的特徵向量得到節點級別 (ranking) 爲: B, A, C, E, D。



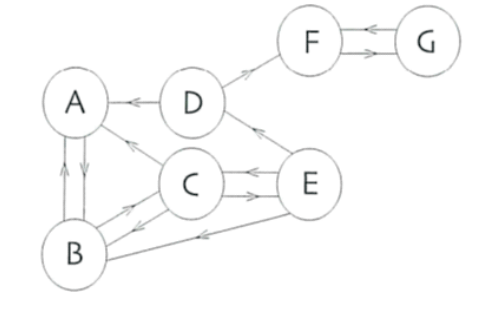
$$P = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{3} & 1 & 0 \\ 1 & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & 0 & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & 0 & 0 \end{pmatrix} \end{array}$$

所以如果你有一個馬可夫矩陣, 在 n 次疊代這個馬可夫矩陣後得到向量 v_1, 這向量對應到任一特徵值爲 1 的特徵向量。它的分量代表連結到那個節點的機率, 也就是網頁的級別順序。所以這就是搜尋引擎的背景計算。

該如何計算特徵向量呢? 這是一個基本的問題。你可以應用 power method。當 n 非常大時計算 $P^n p^0$ 以得到 v_1 的逼近。所以如果你有一個矩陣 P, 你可以一直算到這個矩陣的 n 次方, 看看收斂到哪裡。這就是 power method, 也是計算 p_1 的一種方法。針對大型矩陣計算有一些很有效率的運算。Sergey Brin 和 Larry Page 寫過一篇文章說明馬可夫矩陣大約需要疊代50次 (50 iterations, i.e. $n = 50$) 就可以得到一個好的 v_1 的逼近。

現在我們知道背後的理論了。還有那些問題呢?

假設我們有一個跟下面一樣的圖。



注意到如果使用者到節點 F 後, 會進到一個迴圈, 這個馬可夫程序的穩定向量毫不意外的是 $(0, 0, 0, 0, 0, 1/2, 1/2)^t$, 斜向 (skewed to) 頂點 F 與 G。要克服這個困難, 你可以加入一些臨時的邊或連結回原本的節點。Brin 和 Page (PageRank 演算法的發現者) 建議加一個代表使用者的"品味 (taste)"的隨機矩陣 Q 到 P, 所以最後的移轉矩陣是 $P' = xP + (1 - x)Q$, $0 \leq x \leq 1$。矩陣 Q 是一個完全由 1 所構成的 $n \times n$ 矩陣, 除以節點數 n, 事實上近乎零矩陣。但是這樣就不會有無限迴圈的問題。在這份文章中, 他們建議最佳值 (optimal value) 是 $x = 0.85$。

以上就是我的演講。2008年 Springer 出版的"Mathematics and Technology"中, 作者 C. Rousseau 和 Y. Saint-Aubin, 詳細介紹了這個演算法。2006年 Princeton University Press 出版的 "Google's PageRank and Beyond, The Science of Search Engines", 作者 A. Langville 和 C. Meyer, 也是本不錯的參考書籍。當然, 前面我提到的由 SIAM Review 出版, 2006年第49期, 569-581頁, K. Bryan 和 T. Liese 合寫的 "The 25 billion dollar eigenvector" 也是不錯的參考。有趣的是 Brin 和 Page 從來沒有完成他們的博士學位, 反而走上了另一個方向, 認識到他們可以把論文應用到商業用途上。

我想 Google 已經是生活中不可缺的一部分。謝謝你們聽講, 祝你們好運, 有個愉快的 Google 日。

國家理論科學研究中心數學組李文卿主任: 謝謝你給了一場非常精彩的演講, 有任何問題嗎?

問題1: 我記得在紐約時報上讀過一篇文章說有人知道這個演算法, 並且利用這個演算法將他們的網站排名拉到第一。所以如果有使用者搜尋購物網站, 他們的購物網站會是第一個選擇, 許多顧客都不知道, 一直到紐約時報報導後, 這件事情才被爆出來。後來 Google 試著修正這個問題, 但是很明顯的任何人都可以利用這個伎倆拉高他們網站搜尋的排序。你有任何建議嗎?

回答1: 完全沒有。我是一個完完全全的數學家。當然我同意你的看法, 當一個系統的運作被破解後, 就會有人試著影響運作。然後你就必須要有新的演算法。至於要怎麼產生新的演算法我一點頭緒都沒有。不僅如此, 這件事有一個反面的影響。似乎有一些政治家與公司試著透過 Google 內部利益交易想將他們的網站排序買高甚至排到第一。一旦大局開始腐敗, 就一定會影響其價值。我想這又必須要回到一開始我指出的定理, 如果有其它重要網頁指向一個網頁, 這個網頁就是重要的。對像我一樣研究數學的人, 閱讀特定論文的人數一定不會太多。網路上可能會有一篇很有價值的文章, 但是沒有其他網頁指向它。因爲知道的人不多, 你有可能會發現一個不爲人知的知識寶礦。所以傳統的研究方法, 例如到圖書館獵書仍然是很有價值的。我認爲 Google 不是所有啓發的來源。但一旦有一個如此強大的工具在公共的領域裡, 不免會帶來許多使用不當的遺憾。我沒有解決的辦法。目前已經可以看到不少駭客, 變更他人網站等等, 在這樣的世界裡, 我們遺失了學術的純粹。

問題2: 典型移轉矩陣的大小是多少?

回答2: 很龐大。當你鍵入你的查詢, 它會先丟出數千個網站。我想他們要處理的是 mega 矩陣。我的猜測是通常會有15頁的回應。每一頁大概有30個連結。我估量矩陣的大小應該不會大於 10,000 × 10,000。這是我的猜測。

問題3: 針對這些計算, 他們需要一些特別快速的演算法嗎?

回答3: 這是不同的方向。這可以是另一場演講, 可惜我不是這方面的專家。這應該與快速矩陣計算有關。簡單的說, 就是用矩陣做乘法。轉移機率的計算既快又簡單, 只要把圖做出來做就可以。接下來是必須連乘極多次, 可能可以調整計算速度。

國家理論科學研究中心數學組李文卿主任: 讓我們再次謝謝講者。謝謝。

*感謝中央研究院數學研究所研習員陳昱宇協助校對本文中數學的中文翻譯。