資工情史(2011/11, Bangye Wu)

想起初相見，你輕輕的一聲 hello world 就已敲開了我的心，你那美麗的倩影，如指數成長的 time complexity，瞬間佈滿我的腦海；你的一顰一笑是我的電感，些微的改變就牽動著我生命的磁場，人世間再大的電阻（險阻），只要有你的電容（寬容），就能夠讓我的心像 RC 震盪永不停歇。

我願意成爲你的 Big-O（上界），保護著你，而你是我的 Omega（下界），支持著我；我幻想著我們可以離開紅塵俗世一起到那世外桃源建立 relational database，從此不必再有那煩死人的 addressing mode，我們一起種下 binary tree，不僅種樹可以救地球，生兩個孩子還可以救台灣。

無奈計畫趕不上變化，動態規畫（DP，dynamic programming）也無法處理詭譎的感情事件，我們之間出現了第三者。兩人的世界，Satisfiability 問題總是屬於 P 而易解的（2-SAT），心中有再多的結(node)，只要程度是平衡的（even degree），我總能找到 Euler path 通往你的內心世界；但是三個人的 SAT 問題是 NP-complete，愛情的電梯只能乘載兩人，再好的 elevator algorithm 也解決不了問題，無論我如何努力尋找答案也只是出現 HTTP 404（Not Found）。

我說愛情世界爲了要偵測 match 或 mismatch 必須是個 Stack，後來的應該先離開；你卻說你的心裡只有 Queue，我先來就該先走。面對你這種數位子的邏輯（Digital Logic），我的 compiler 只能 parsing 出 syntax error，就像我的 CodeBlock 永遠 Block 我的 Code，怒火中燒的我只能回答：「ㄡ，ㄡ，屁啦（OOP）」，然後像 tree graph 的 leaves 被你的狂風吹 leave。

曾經你是我的 basis，span 出我所有的空間，如今你我卻已是 linear independent，除了原點外不再有交集，我倆的恩怨情仇，不是排容原理（principle of inclusion-exclusion）可以計算的了的。

雖然離開了你，我的生活卻無法平靜，你像 cycle-stealing，不僅偷走了我的 CPU cycle，還一次偷走了兩個（bi-cycle）害我上課總是遲到。愛你與恨你兩種情緒像是不斷在我心中上演的伯努力試驗而產生的二項式分布（binomial distribution），使得我陷入 Race condition 而人格分裂，對你的渴望總在午夜夢迴（return）時遞迴地呼叫（recursively call），而恨你的情緒也不像拉普拉思（Laplace）經過轉換（Transform）就可以求解。激盪出的是心中的 Deadlock，無法 avoid 也無法 prevent，即使可以 detect 也無法 recover（處理 database deadlock 方式）。

我乞求上蒼讓對你的記憶如同指數分布（exponential distribution）般的memoryless，但是他卻像 touch panel 的 API 似的一碰就來、像噗浪（Plurk）般的前仆後繼、像推特（Twitter）一直朝我推來，搞得我非死不可（FaceBook）。

朋友勸我要用暴力法（Brute-Force）解決問題，爸媽要我學會 divide-and-conquer，但是你在我心中的份量仍然遵循摩爾定律每十八天翻倍，在一年後的今天增長了百萬倍，而被 linked list 所串成的相思細雨，像是散落在程式中的 Bug，永遠滴(De)不完；而數著愛與不愛的玫瑰花辦，則是無窮迴圈永遠落不盡。最後救了我的是專題實作（Project），他把我對你的思念 Project 到努力用功這件事上，我發誓要走出陰霾，這是我和生命所做的 TCP/IP 協定（To Compromise Principle, I Promise）。

===我是分隔線===

前一陣子看到 FaceBook 上流行 XXX 賓果，很是有趣，一些 KeyWord 總是可以引起相同背景的人的一些共鳴，前幾天又看到一個把電感誤會成電桿的笑話，於是一時興起將記憶中的資工人課程中所學的 keyword 串成個小故事。有些牽強附會的地方，笑笑就好。不是資工人看不懂的當然是無可厚非，資工人卻看不懂的，那就……
複習一下吧！

(吳邦一, 2011/11), 本文歡迎分享轉載，原 PO 部落格爲：

http://bangyewu.wordpress.com/2011/11/08/%e8%b3%87%e5%b7%a5%e6%83%85%e5%8f%b2/

===我又是分隔線===以下是增加的說明======

有些梗不大明顯，以下把文中的專有名詞列出部分做些許說明，大一大二的同學有些地方看不懂，或許過一兩年再看又會別有收穫，好奇心找谷維琪（Google+Wiki）大概都查得到（包括 hello world，相信吧）：

1. Hello world：大部分資工人的第一支程式。
2. 指數成長：函數值隨變數的增加而指數成長
3. Time complexity：演算法所需時間隨資料量大小的成長情形。
4. 電感
5. 電阻
6. 電容
7. RC 電路：電子電路中的一種基本電路，由電阻(R)串接電容(C)組成。
8. Big-O：評估演算法效率 upper bound 的一種數學表示方式。

9. Omega：評估問題難度 lower bound 的一種數學表示方式（複雜度表示法此處非電阻的歐姆）。
10. Relational database：目前大多數資料庫的類型。
11. addressing mode：CPU 指令的資料定址模式。
12. 種下 binary tree，不僅種樹可以救地球，生兩個孩子還可以救台灣：binary tree 是資料結構，每個節點最多兩個分支節點（稱爲 children），這是一個學生在期末考卷上寫下的笑話。
13. DP：演算法設計設策略之一
14. Satisfiability problem(SAT)：第一個被證明爲 NP-complete 的問題。
15. P: polynomial-time solvable problems(已經證明存在有效率演算法的問題)
16. 2-SAT：SAT 問題的一種簡化型，屬於 P。
17. 「心中有再多的結(node)，只要程度是平衡的（even degree），我總能找到 Euler path 通往你的內心世界」：Euler path 的充分必要條件，部分名詞取雙關語。
18. 3-SAT：SAT 問題的一種簡化型，屬於 NP-complete
19. NP-complete：一類很難找到有效率解法的問題。
20. 「愛情的電梯只能乘載兩人，再好的 elevator algorithm 也解決不了問題」：有個故事是這樣說的。有個女滿臉失意站在電梯前按了往上的按鈕，一會兒，電梯打開了，裡面有兩個人，女子想了一下 說：「太擠了，我搭下一班」。奇怪嗎？他後來搭下一班電梯上樓和電梯裡的那位男子簽了離婚協議書。另外，elevator algorithm 是 OS 課的專有名詞，是目前電梯常用的演算法。
21. 「無論我如何努力尋找答案也只是出現 HTTP-404」：HTTP 404 是開啓網頁卻找不到檔案時的 error message。
22. Stack：資料結構，last-in-first-out。
23. 偵測數學運算是中括號是否正確配對(match)可以用 stack。
24. Queue：資料結構，first-in-first-out。
25. Digital Logic
26. Compiler：將程式設計師所寫的程式轉換成計算機可以執行的指令的一種軟體。
27. Parsing：compiler 中的一個動作。
28. syntax error：程式的語法不合規定，初學程式者最常碰到的錯誤。
29. 「就像我的 CodeBlock 永遠 Block 我的 Code」：CodeBlock 是現在資工學生常用來編寫程式的 IDE（整合發展環境），此處的 Block 應該是區塊之意，因爲他可以將一個 block 的 code 展開與收起來，但 Block 可做「阻擋」解釋，取其雙關語，用 codeblock 寫程式 永遠不過關。
30. OOP：（Object Oriented programming）物件導向程式設計。
31. tree graph：無環路的連通圖

32. leaves：葉子，另 leave=離開。葉子離開樹是因爲風而不得已還是自己想離開，又或者只是時機的問題。
33. Basis：線性代數中一個空間的基底，爲一組線性獨立且數量等於 dimension 的向量。
34. span：基底的線性組合可以生成空間中的任一向量，稱爲 span the space。
35. linear independent：線性代數名詞，一群向量不能由其中幾個組合成另外一個。
36. principle of inclusion-exclusion：離散數學中計算多個集合聯集交集的一種計算法則。
37. 「你像 cycle-stealing，不僅偷走了我的 CPU cycle，還一次偷走了兩個（bi-cycle）害我上課總是遲到。」cycle-stealing 是計算機結構的專有名詞，CPU 與周邊裝置皆需使用 Bus 到 memory 存取資料，在 CPU 一個指令執行的某些 cycle 中，周邊裝置乘隙搶用 Bus 稱爲 cycle-stealing，好像偷走了 CPU 的 某些 cycle。又英文中兩個 cycle 就是 bicycle(腳踏車)。
38. 伯努力試驗：結果只有兩種之一的試驗，如拋銅板出現正面或反面。
39. 二項式分布（binomial distribution）：一種機率分布，描述如拋 N 次銅板會出現 K 次正面的機率。
40. Race condition：電路中，輸出結果會因爲兩個輸入訊號到達順序不同而發生未知結果而產生的錯誤。在作業系統(OS)或資料庫管理中，多個程序的執行根據相同的狀態也會發生類似的錯誤，例如同時對一個銀行帳號執行兩個提出全部款項的動作，如果在第一個提款程序尚未將餘額變更前，讓第二個提款程序檢查目前餘額是否足夠，就會造成兩筆都提領成功的錯誤。（別去試了，現在 database 系統都能正確處理這種狀況，不會產生 race condition）
41. Return：計算機程式指令
42. recursively call：計算機程式指令的一種類型
43. Laplace Transform：（工程數學，數位訊號處理）將訊號由 time domain 轉換爲 frequency domain，經常用於數位語音與數位影像的處理。
44. Deadlock：（OS, database）不同程序各自擁有部分資源卻在彼此等待對方釋放手中的資源，會造成你等我我等你而造成大家都動不了的窘境，類似於黑羊白羊在獨木橋上相遇而彼此不肯退讓的情形。
45. 處理 database deadlock 方式：avoid, prevent, detect and recover
46. 指數分布（exponential distribution）：一種機率模式
47. 指數分布的 memoryless：指數分布的重要特性，電子產品的故障機率大致符合此特性，因此有出廠前 Burn-in 測試。
48. touch panel
49. API：application programming interface，上游軟體開發者所撰寫的程序提供給下游程式設計者在程式中呼叫以完成某些工作的介面。

50. 噗浪（Plurk）：現今流行的網路社群平台之一。
51. 推特（Twitter）：現今流行的網路社群平台之一。
52. FaceBook：現今流行的網路社群平台之一。
53. Brute-Force：演算法設計策略之一，列舉所有可能一一測試。
54. divide-and-conquer：演算法設計策略之一，把一個問題切割成若干個小問題在一一解決，通常切割出與原問題相同但資料量較小的問題，因此可用 recursive 解決。
55. 「但是你在我心中的份量仍然遵循摩爾定律每十八天翻倍，在一年後的今天增長了百萬倍」：摩爾定律稱計算機硬體的容量與速度每 24 個月倍增，這裡改成十八天，一年大約 20 個十八天，所以會增加 2 的 20 次方倍大約是 1M（一百萬）。
56. linked list：一種資料結構，處理動態序列資料。
57. debug：程式中的錯誤稱爲 Bug，將錯誤找出改正稱爲 debug，是程式設計者最頭痛的工作。
58. 無窮迴圈：迴圈是程式中一種重要結構，通常是對不同資料做相同運算，但迴圈控制如果沒寫好會造成一直在執行迴圈而不離開的錯誤。
59. 「最後救了我的是專題實作（Project），他把我對你的思念 Project 到努力用功這件事上」：資工系學生多半有專題實作課（Project），但 project 在線姓代數裡是投射投影之意。
60. 「我發誓要走出陰霾，這是我和生命所做的 TCP/IP 協定（To Compromise Principle, I Promise）。」：TCPIP 是目前常用的網路通訊協定，後面瞎掰了個他的縮寫，不過也有其意義，人生有時候是要懂得妥協的，尤其在陷入困境走不出去 的時候。