應用 PC 叢集基因演算法於訓練隱藏式馬可夫模型尋找大腸桿菌的基因

Using a PC-Clustering Genetic Algorithm for Training a Hidden Markov Model to find Genes in E.coli DNA

施文弦、侯玉松、曾亮嘉

中華大學資訊工程系

yshou@chu.edu.tw

摘要

在生物資訊研究領域裡，找出基因位置是一個重要課題。在過去的研究文獻中，曾有學者利用 Baum-Welch 演算法訓練隱藏式馬可夫模型，以找尋大腸桿菌 DNA 中的基因位置。本論文採用基因演算法來訓練隱藏式馬可夫模型，在 PC 叢集上實作，並與 Baum-Welch 演算法互相比較其預測效果。

關鍵詞：隱藏式馬可夫模型、基因演算法、PC 叢集

Abstract

Gene finding is an important research topic in bioinformatics. In the past, researchers used the Baum-Welch algorithm to train hidden Markov models for E. coli's gene finding. In this paper, a genetic algorithm was implemented in PC-clusters for train a hidden Markov model. The prediction results are compared with the Baum-Welch algorithm.

Keywords: Hidden Markov Model, Genetic Algorithm, PC-Cluster

一、概論

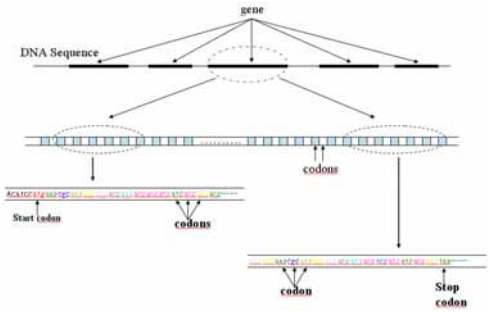
生物體內所有遺傳物質的構造稱為染色體(chromosome)，是由去氧核醣核酸（DNA）與蛋白質所組成[1]，在 1952 年，賀西（Alfred D，Hershey）和蔡斯（Martha Chase）對噬菌體所做的實驗[2]，證實 DNA 是影響遺傳的主要物質。

DNA 的基本單位是核苷酸(nucleotide)，而構成核苷酸的單位是由一個五碳糖(pentose)，磷酸(phosphoric acid)及鹼基嘌呤(purine)或嘧啶(pyrimidine)組成。鹼基共有四種：胞嘧啶（Cytosine，簡稱 C）、胸腺嘧啶（Thymine，簡稱 T）、腺嘌呤（Adenine，簡稱 A）、鳥糞嘌呤（Guaninne，簡稱 G）。

DNA 為雙螺旋結構，核苷酸在螺旋體的內側兩兩相對。G 必與 C，T 必與 A 配對形成鹼基對（base pair），構成雙螺旋體樓梯型排列。

蛋白質由胺基酸序列所組成，其生成必須經過轉錄（transcription）和轉譯（translation）這兩個過程，將 DNA 中某片段的核苷酸序列轉換成胺基酸序列[8]，以構成蛋白質。

基因(gene)是 DNA 上的一個片段，可以提供產生蛋白質的核苷酸序列。生物體的基因結構，依照生物細胞構造而有所不同，可分為真核生物與原核生物二種。原核生物的基因結構比較簡單，基因的核苷酸序列的開端有個初始子(start codon ，在原核生物為 ATG 或 GTG)，接著一連串的密碼子(codon)，最後以終結子(stop codon，在原核生物為 TAA、TAG 或 TGA)。其結構如圖一。



圖一：原核生物的基因結構

本論文的主要目的，在研究原核生物的基因位置預測方法，即輸入一段連續的原核生物 DNA 序列，可輸出這段 DNA 中基因的預測位

置。我們採用文獻[5]的隱藏式馬可夫模型(hidden Markov model，簡稱 HMM)，然後設計基因演算法(genetic algorithm，簡稱 GA)，在 PC 叢集系統上實作其訓練程式，以大腸桿菌為例，輸入其 DNA 序列，以統計其輸出預測結果的靈敏度（Sensitivity）與專一性（specificity），並與 Baum-Welch 的訓練演算法的預測結果相比較，以探討 PC 叢集 GA 在訓練 HMM 的效果。

二、隱藏式馬可夫模型

HMM 是一個有限狀態系統，每一狀態聯繫到其他狀態是個一維或多維的機率分配。狀態之間的轉移是一個機率集合稱為「轉移機率」（transition probabilities），在特定的狀態中，會產生「觀測值」（observation），觀測值也必須符合機率分配。因為 HMM 只輸出「觀測值」，狀態本身是隱藏的，所以稱為「隱藏馬可夫模型」。

HMM 的相關基本術語如下：

N：狀態的個數

$S=\{S_1,.....,S_N\}$ ：為狀態集合

M：狀態中觀測符號的個數

a_{kl}：從狀態 S_k 轉移到狀態 S_l 的機率，$1\le k,l\le N$

$e_l(x_i)$：在狀態 S_l 輸出 x_i 字母的機率，$1\le i\le M$

$A=[a_{kl}]$：狀態轉移機率的矩陣，$1\le k,l\le N$

$E=[e_k(x_i)]$：每一個狀態的輸出機率矩陣，$1\le k\le N, 1\le i\le M$

q_t：在時間 t 的系統狀態

o_t：在時間 t 的輸出觀測符號

$O_i=(o_1,o_2,.....,o_T)$：從時間 1 到時間 T 的觀測符號輸出的序列 O_i

$\lambda=\{A，E\}$：HMM 的所有參數的集合

HMM 參數須符合下列基本規則：

$$1=\sum_l a_{k,l} \quad \text{for } 1\le k\le N \quad ...(1)$$

$$1=\sum_i e_l(x_i) \quad \text{for } 1\le l\le N \quad ..(2)$$

在 HMM 的使用上，必需先解決三個問題[6]：

(1) 求值問題(The Evaluation Problem)

給定一個 HMM 參數集合 λ 及一個觀察的序列 $O=(o_1,o_2,.....,o_T)$，求出這個序列是由此 HMM 所產生的機率，即 P(O|λ)是多少?

(2) 解碼問題(The Decoding Problem)

給定一個 HMM 參數集合 λ 及一個觀察的序列 $O=(o_1,o_2,.....,o_T)$，則可產生這個觀測序列的所有狀態序列中，何者狀態序列發生的機率最大?

(3) 學習問題(The Learning Problem)

給定一個 HMM 參數集合 λ 及一個觀察的序列 $O=(o_1,o_2,.....,o_T)$，如何調整模型的參數{A，E}，使得 P(O|λ)最大?

在過去文獻中，有三種演算法可以分別解決這三個問題：

(1) 求值問題：使用前向推導法（forward）或反向推導法(backward)。

(2) 解碼問題：使用 Viterbi 演算法。

(3) 學習問題：使用 Baum-Welch 演算法。

這三種演算法，請讀者自行參閱文獻[3、6]。

三、預測大腸桿菌基因位置的隱藏式馬可夫模型的結構

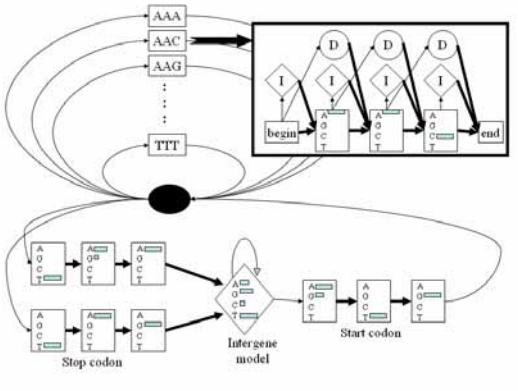
圖二是文獻[5]記載預測大腸桿菌的 HMM 結構圖。對這個模型輸入 DNA 的觀測序列，起始狀態是「基因間」(intergene)狀態，直到遇見初始子（start codon）ATG 或 GTG 之後，轉入編碼區（coding region），在編碼區裡，會不停的重複的把核苷酸序列編碼成胺基酸序列，在轉換的過程中，DNA 序列有可能因實驗誤差遇到插入（Insert）或刪除（Delete）核苷酸的情況，如圖黑框部分，菱形表示插入，圓形表示刪除，直到遇到終結子（stop codon）TAA、TGA 或 TAG，表示基因的結束，及進入「基因間」狀態，為預測下一個基因做準備。

此 HMM 共有狀態數：742 個，參數個數為：狀態轉移機率 a_{ij} 的個數共 1233 個，狀態輸出機率 $e_j(x_i)$的個數共 2480 個。

四、基因演算法與隱藏式馬可夫模型的訓練

GA 是 John Holland 於 1975 年首先提出[4]。它模仿生物進化的過程，以「優勝劣敗」

的原則來優化解答，逐漸求得最佳解。GA 可以在很大的解集合空間中，快速搜尋出最佳解。GA 的多點搜尋並交換資訊的特性，具備分散式處理的潛力，很適合用於快速求解。



圖二：預測大腸桿菌基因位置的 HMM 結構

GA 的運作大致可以分成幾個階段：

(1) 初始（initialization）：建立族群（population）

(2) 評估（evaluation）：適應值（fitness）的計算。

(3) 選擇（selection）：讓適應值較高的個體較易生存，較低的個體則較易遭淘汰。

(4) 子代（offspring）的產生：利用互換（crossover）、突變（mutation）以及複製(reproduction)的遺傳操作，產生新的個體並加入群體。

(5) 終止（termination）條件：如果演化結果尚未到達所訂定的條件，則回到第 2 階段，繼續下一代的演化。

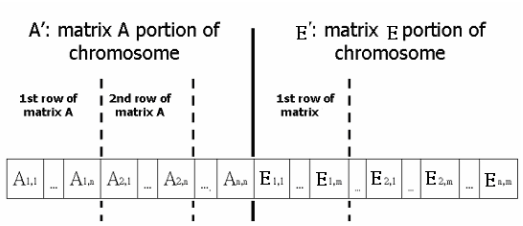
GA 應用於 HMM 的訓練方法，主要是參考文獻[7]，說明如下：

(1) 資料表示方式

在 HMM 中，最主要儲存的參數是狀態轉移的機率及各個狀態中的觀測出現機率，分別是 A 矩陣與 E 矩陣，所以在這裡，將這兩個矩陣，以列為主(row major)的方式，合成為一個陣列，A 矩陣放在陣列前面，E 矩陣放在陣列後面，如圖三。

(2) 適應函數

在這個 HMM 的模型中，我們給 N 組的訓練觀察序列 O_1、…、O_N，則適應函數如下：



圖三：HMM 參數的 GA 資料表示法

$$p_{j=}\frac{1}{N}\left(\sum_{i=1}^{N}p(O_i|\lambda_j)\right)\ 1\le j\le n$$

n： 族群的個體總數

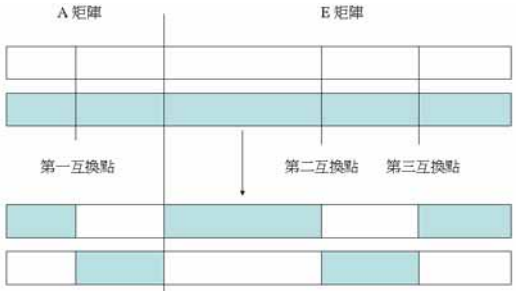
λ_j： 第 j 個個體對應的 HMM 參數

P_j： HMM 模型參數λ_j對所有觀測序列 O_1、…、O_N發生的平均機率，定義為第 j 個個體的適應值。

其中，$P(O_i \mid \lambda_j)$可用前向推導法或反向推導法求得。

(3) 染色體互換

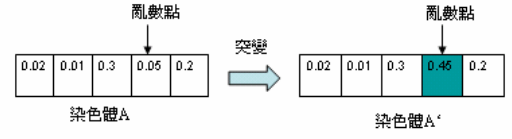
在本論文中，染色體互換的選擇率為 20%。操作方式如圖四，在 A 矩陣取一個亂數點，在 E 矩陣取兩個亂數點，作為兩條染色體的交換點。



圖四：染色體互換

(4) 染色體突變

在本論文中，突變機率為 0.01，操作方式如圖五，取一個體並在此個體資料上取一個亂數點。將亂數點所指的位子上，再以一個 0~1 之間的亂數取代之。



圖五：染色體突變

(5) 參數調整

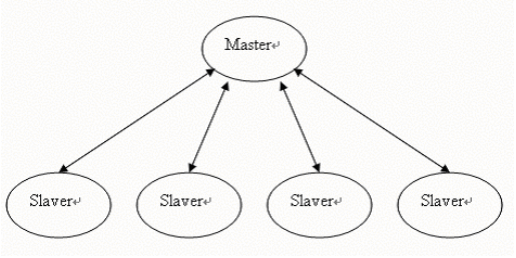
經過了染色體互換與基因突變，改變後的參數時勢必會為違反當初 HMM 模型的定義（式 1、2），所以我們必須對參數作正規化(normalize)的調整，以滿足參數機率和為 1 的限制，調整的公式如下：

$$\bar{a}_{ij} = \frac{a_{ij}}{\sum_{n=1}^{N} a_{in}} \quad 1 \le i \le N$$

$$\bar{e}_j(x_i) = \frac{e_j(x_i)}{\sum_{i=1}^{M} e_j(x_i)} \quad 1 \le j \le N$$

五、PC 叢集式基因演算法

在 PC 叢集式 GA 的網路連接方面，本論文採用的是主僕式(master-slave)架構，如圖六。每一個僕節點都代表一個族群，分別獨立進行 GA，經過一定的演化代數後，每個僕節點挑選部分個體，傳送到主節點，主節點會將所有收集到的個體，平均分配給每個僕節點，即僕節點透過主節點進行個體交流，以達到各族群交流的目標。



圖六：主僕式 GA

六、執行效能及結果

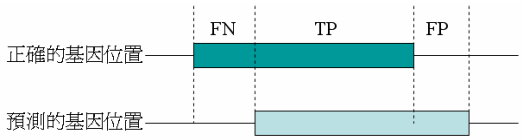
本論文所用之大腸桿菌 k-12 實驗資料來自於美國威斯康辛大學麥迪遜分校（University of Wisconsin, Madison）的 Frederick R. Blattner，所負責的大腸桿菌基因組計畫，並且透過網路資料庫從中取得了 400 條的 DNA 序列，以做為資料訓練及資料測試使用，因為這些 DNA 序列資料本身並包不含基因組的突變異常（插入、刪除、置換）等生物現象，為了還原大腸桿菌基因組最初的基因組原貌，以突變機率 0.01，取亂數的方式在 DNA 序列中加入這些變異。

在硬體方面，Baum-Welch 與 GA 在單機上實作，另在 PC 叢集上實作 GA，單機及 PC 叢集電腦規格如表一：

表一：硬體規格表

	單機	PC 叢集電腦	
		Master*1	Slaver*4
中央處理器	P4 1800	AlthlonMP 1.2G*2	AMD XP 1600+
記憶體	256MB	1G	768MB
硬碟	40G	40G	40G
作業系統	Windows XP	Linux	Linux

在軟體方面，三組程式均以 C++語言撰寫，而 PC 叢集式 GA 是在 MPI 的作業環境下完成[9]。實驗分為三組，第一組為 Baum-Welch 演算法訓練的 HMM 的預設結果、第二組為單機 GA 訓練出來的結果、第三組為 PC 叢集式 GA 訓練出來的結果。將這三種訓練結果後的 HMM 參數，輸入新的觀測序列，分別以 Viterbi 演算法測試，測試後，可以得到這三種基因序列的預測結果，再與已知的基因位置做比對，計算其靈敏度（Sensitivity）與專一性（specificity），如圖七所示。



圖七：預測程式的靈敏度與專一性計算依據

其中，

(1) FN：實際上在該位置為基因，但程式沒有預測為基因。

(2) TP：實際上在該位置為基因，程式預測為基因。

(3) FP：實際上在該位置非基因，程式預測為基因。

所以，

靈敏度（sensitivity）= TP/(FN+TP)
專一性（specificity）= TP/(FP+TP)

本實驗是以 100 條長度為 1~2K bp（base pair）的 DNA 序列當訓練資料 ，300 條長度為 1~2K bp 的 DNA 序列當測試資料，以下分別是 Baum-Welch 演算法，單機 GA，PC 叢集式 GA 的實驗結果。

(1) Baum-Welch 演算法：

靈敏度（%）：85.09
專一性（%）：80.01
執行時間（秒）：960.11

(2) 單機 GA：如表二。

表二：單機 GA 結果

演化代數	執行時間（秒）	靈敏度（%）	專一性（%）
10000	24854.60	77.56	75.76
50000	132863.40	80.94	81.32
250000	496682.80	83.33	84.07

(3) PC 叢集式 GA：如表三。

表三：PC 叢集式 GA 結果

演化代數	執行時間（秒）	靈敏度（%）	專一性（%）
10000	22114.23	82.91	80.94
50000	103144.61	85.43	85.23
250000	463162.71	86.63	84.22

在單機 GA 實驗部分，族群個體 200 個，在每代的演化過程中，篩選 40 個個體進行實驗，交換機率設定為 20%，突變機率設為 5%。PC 叢集式 GA，以一個主行程、四個僕行程為架構，在僕行程的 GA 參數設定上，與單機 GA 相同，並設定每隔 5000 代的繁衍後，僕行程即挑選適應值最佳的前 40 個個體，傳送至主行程，進行族群交流。

從實驗的數據我們可以得知，單機 GA 演化到 250000 代時與傳統的 Baum-Welch 演算法，在靈敏度與專一性互有領先，但差距很小。PC 叢集式 GA 演化到 250000 代時，它的靈敏度與專一性都比的單機 GA 或 Baum-Welch 演算法來的高，這種現象表示 PC 叢集式 GA 的預測結果，優於單機 GA 或 Baum-Welch 演算法。

七、未來展望

訓練 HMM 的 GA 的設計理念，主要是參考文獻[7]，其預測結果，如前一節所示，比 Baum-Welch 演算法好。但是缺點是計算耗時太久，主要原因是計算適應函數的 $P(O_i \mid \lambda_j)$ 時，需使用耗時的前向推導法或後向推導法，而且每一新生子代均需計算其適應值，造成計算量龐大的情況。

另外，雖然 PC 叢集式 GA 的預測結果比較好，但是改善程度約為四個百分點，應該還有改進的空間。我們認為，適應函數 P_j 的計算是僅就訓練資料來設計，若訓練資料的代表性不足，便會影響預測的準確度，如何評估訓練資料的代表性，應當是相當重要的課題。

八、參考文獻

[1] 李千毅譯, *觀念生物學*, 天下文化書坊, 2002

[2] 黃皓陽編譯, *生物學*, 藝軒圖書出版社, 2001

[3] R. Durbin, S.R. Eddy, A. Krogh and G. Mitchison, *Biological sequence analysis Probabilistic models of proteins and nucleic acids.* Camberidge university press, 1998

[4] J. Holland, *Adaption in Natural and Artificial Systems*, The University of Michigan Press, 1975.

[5] A. Krogh, IS. Mian and D. Haussler, "A hidden Markov model that finds genes in E. coli DNA," *Nucleic Acids Res*. Nov 11;22(22):4768-78, 1994.

[6] R. Lawrence, and Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of IEEE*, Vol. 77, No. 2, pp. 257-286, 1989

[7] K.F. Man, K.S. Tang and S. Kwong, *Genetic Algorithms Concepts and Designs*, Springer-Verlag London Limited, 1999.

[8] J. Setubal and J. Meidanis, *Introduction to Computational Molecular Biology* , PWS Pub lishing Company, 1997.

[9] G. William, L. Ewing, and S. Anthony , *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, 1994