以家族競爭基因演算法解資料庫索引選擇問題

唐偉閔　高國峰　廖宜恩
國立中興大學資訊科學學系

E-mail: w9156010@cs.nchu.edu.tw,kfkao@mail.hit.edu.tw, ieliao@nchu.edu.tw

摘要

資料庫索引選擇問題(Index Selection Problem, ISP)是一個 NP－hard 的問題。而基因演算法(Genetic Algorithm , GA)則是解決 NP－hard 問題的一個有效的方式。在以往的論文中，針對 ISP 的 GA 研究，多採取競爭式選擇及均勻交配的標準演化方法。本研究進一步採取了家族競爭演算法(Family Competition Evolutionary Algorithm, FCEA)。FCEA 在傳統的基因變異過程中，加入族群競爭的觀念，使問題在求解的過程中，能夠更具有多樣性及變化性，進而提高找到可行解的機率。本研究並建立了多種不同的案例資料及不同索引個數的測試樣本。從多個測試結果中，確實顯示本研究提出的方法，較諸傳統基因演算法，具有較好的效能。

關鍵詞：基因演算法、資料庫索引選擇、案例資料、家族競爭演算法。

1. 簡介

在資料庫系統中，索引是一個重要的輔助工具，索引建立與否會關係到資料庫效能。如何在一組大量查詢中，找到適合的索引組態更是一個很重要的課題。這個尋找最佳索引組態的問題，被稱為索引選擇問題(Index Selection Problem, ISP)。在抽象的數學模式上，ISP 與著名的工廠選址問題（Uncapacitanted Facility Location Problem，UFLP）[4,11]是類似的。UFLP 和 ISP 都已被證明是 NP-hard 的問題[8]。

對於複雜的 NP-hard 問題，我們經常使用基因演算法來求近似解，以避免尋找最佳解所需耗費的巨大計算成本。在以往的研究中，Jozef Kratica、Ivana Ljubic 和 Dusan Tosic 也提出用基因演算法來解索引選擇問題的做法[7]，經由選擇、交配、突變等基因演化步驟找到最佳近似解。在 Kratica 的基因演算法中，選擇方法是採用競爭式選擇，交配方式是用均勻交配，突變是用單一位元發生突變。我們認為這樣的做法較為單調，很容易陷入局部最佳解。因此，在本研究中，我們採用家族競爭演算法(Family Competition Evolutionary Algorithm, FCEA)[6]，來改善演化的流程。在本論文的實驗結果中，我們證實，針對 ISP 問題，家族競爭演算法的確比 Kratica 基因演算法擁有較佳的效能。採用本論文的做法，不但可以減少 Kratica 演算法需要參數設定的困擾，更可以以較少的時間找到較好的近似解。

除了效能的改善，對於模擬資料的建立，我們也提出一些改進的做法。在 Kratica 研究所採用的測試資料多是使用亂數產生，並不是很符合實際資料庫的狀況。例如說一個讀取的查詢 Q1，單單使用 index1 的獲利是 10，則當資料庫的查詢最佳化程式（query optimizer），決定讓查詢 Q1 同時使用 index1 及 index2 時，其獲利就不應該小於 10。因為若其獲利小於 10，則對一個正常的查詢最佳化程式而言，它會知道單單採用 index1 就會有更好的效能。那麼它就應該要採用 index1 而不該同時採用 index1 及 index2。像這些實際運作時的規則，亂數產生的數據是無法符合要求的。本研究在建立案例資料(Instances)時，同時做索引組態與查詢語句的分析，並限制模擬資料產生的條件，這會比 Kratica 的測試資料更接近實際資料庫的測試案例資料。

本論文其它部分的內容如下。第二章相關研究，將介紹索引選擇和基因演算法的基礎定義及過去的研究成果。第三章介紹家族競爭基因演算法。第四章為系統實作與實驗結果的分析。第五章將描述我們的結論與未來研究方向。

2. 相關研究

索引主要的作用，是在資料庫系統中，額外建立一個只包含資料指標及資料鍵值的檔案。使得系統在查詢時，能更快找到資料。但是對更新(update, insert, delete）資料的查詢而言，不管是否使用索引去找到資料，索引檔將被同步更新，以維護資料的一致性。所以索引對一個同時包含讀與寫查詢的工作負載(Workload)而言，其作用可能是正面的，也可能是負面的。索引選擇問題可描述成：在包括多個查詢的工作負載中，找出有那些該被建立的索引，可以使得整個工作負載的執行效能達到最好的狀態。

資料庫索引選擇問題已被証實為 NP-hard 問題[8]，且研究出很多方式來解決索引選擇問題。A. Caprara , M. Fischetti and D. Maio [2]用

heuristics based 和 Branch_and_bound 演算法來解索引選擇問題，除了利用 divide and conquer 來準確解索引選擇問題外，並用索引和查詢在測試案例資料的分布特性，求出近似解。雖比不上 divide and conquer 方式準確，確也大大縮短執行時間。A. Caprara and J. J. Salazar Gonzalez [3]提出 branch-and-cut(BnC) 演算法來改善 branch-and-bound 演算法來解決 Uncapacitanted Facility Location Problem(UFLP)問題，Jozef Kratica, Ivana Ljubic and Dusan Tosic [7] 則提出基因演算法來解決索引選擇問題，並和 BnC 較比較，在大部分的測試案例資料，基因演算法有比較好的效能。

2.1 基因演算法

基因演算算是應用達爾文物競天擇原則，利用不斷選擇優良基因交配，加上突變和演化而產生最強物種。基因演算法主要的目的是用自然生物系統的進化過程，來解決各類問題的最佳解，藉由生物每代進行演化，尋得適當問題的最佳解。

基因演算法採用隨機多點同時運做的方式(複製→交配→突變)，而非傳統的單點依序搜尋方式，因此可以避免侷限在區域的最佳解上，而得到問題的最佳解上。如圖 1 是基因演算法流程圖。

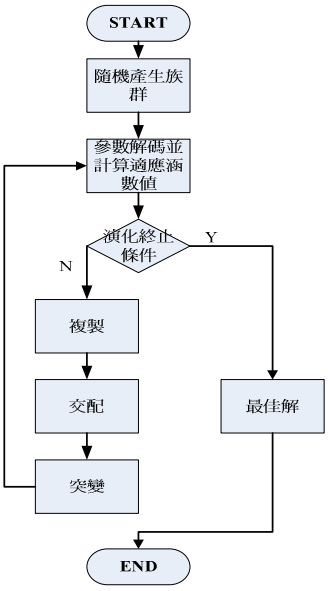

圖 1 基因演算法流程圖

基因演算法運作的幾個主要步驟為:編碼(Coding)、初始化(Initialization)、複製(reproduction)、交配(crossover)、突變(mutation)。其重點概述如下：

● **編碼(coding)：**

定義一個對應的方式，讓每一組解，都分別對應到一個唯一的字串。為了運算方便，該字串的編碼多採用二進位編碼。一組字串，即為基因演算法中的一組染色體。

● **初始化：**

染色體基因字串，在初始時，多由亂數隨機產生的0或1所組成。染色體族群數量大小會因問題而改變。初始族群太多會造成運算太慢，若初始族群太少則會缺少多樣性，易造成局部最佳解。

● **複製(reproduction)：**

是依據每一物種的適應程度，來決定下一代中應被淘汰或複製的一種運算過程。複製過程有兩種形式：(a)輪盤式選擇(b)競爭式選擇。

輪盤式選擇是在每一代演化過程中，以字串應函數大小以比例來決定機率。輪盤所佔的面積愈大所選中的機率就愈大。競爭式選擇是在每一代選擇二個以上字串適應函數最高放入交配池。競爭式有很多不同的形式，Fine Grained Tournament Selection[9,10]是其中的一種演算法。

● **交配 (crossover):**

是依據交配率從父代族群中隨機地取兩個字元串，交換彼此的基因位元，產生兩個新的下一代。基本交配模式有三種(a)單點交配(one-point crossover) (b)兩點交配(two-point crossover)(c)均勻交配(uniform crossover)。

● **突變:**

突變的過程是隨機的選取一物種的字串，並隨機選取突變點，並改變基因字串裡的位元資訊。例如原本是 0 改成 1，原本是 1 改成 0。突變過程發生機率可由突變機率所控制。

● **終止條件(termination criterion)：**

在演化過程中，不可能無窮盡演化下去，所以會設終止條件.終止條件有三種：

1. 依據適應函數的變化，決定是否停止
2. 設定最大演化世代的次數，達到即停止
3. 結合1和2的終止條件.

2.2 以基因演算法解索引選擇問題

基因演算法的是用二進位編碼方式，把索引引建立與否轉成二進位編碼方式，位元值1表示建立索引，位元值0表示不建立，即可用基因演算法解索引選擇問題。在Jozef Kratica, Ivana Ljubic and Dusan Tosic 的論文 [7]中，其基因演算法中的選擇方法是用Fine Grained Tournament Selection (FGTS)[10]，交配方式是用均勻交配(uniform crossover)[11]，突變是用單一位元發生突變。圖2 是Kratica基因演算法示意圖，把母體放入配交池中，經由交配、突變產生子代，再由FGTS找出下一母體。圖3是FGTS示意圖，設Ftour=5.6，則把子代分成40%和60%二大部分，40%母體每個父代由40%子代亂數取5個子代取最好解當下一次父代，另60%母體每個父代由60%子代亂數取6個子代取最好解當下一次父代。

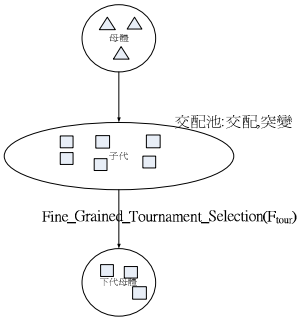

圖 2 Kratica 基因演算法示意圖

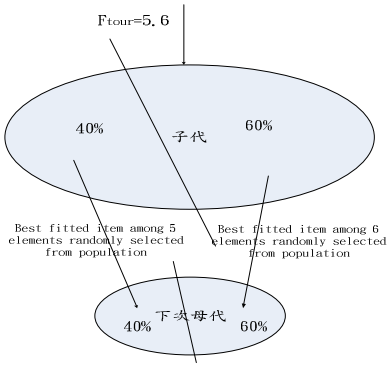

圖 3 是 FGTS 示意圖

3. 家族競爭基因演算法

本節是本論文的核心，我們將詳細描述 ISP 問題正式的數學定義，並且說明其如何運用到家族競爭演算法。

3.1 問題定義

讓 $N = \{1, 2, \ldots, n\}$ 是索引集合，$M = \{1, 2, ..., m\}$ 是查詢集合。每一個索引 j 可以建立或不建立，每一個索引建立都會產生維護成本 f_j 和增益值(gain)。n 個索引都可以建立或者不建立，形成了 $p = 2^n$ 種組態(configuration)，我們用 $P = \{1, 2, ..., p\}$ 來代表所有的組態。令組態 $k \in P$, 查詢 $i \in M$，索引 $j \in N$，則 g_{ik} 表示查詢 i 使用組態 k 的增益值。x_{ik} 是決策變數，其值為0或1，代表查詢 i 是否使用組態 k。f_j 是建立第 j 個索引所需的維護成本。y_j 也是決策變數，代表第 j 個索引是否建立。則我們的目標函式為目為：

$$\max\left(\sum_{i \in M}\sum_{k \in P} g_{ik} \cdot x_{ik} - \sum_{j \in N} f_j \cdot y_j\right) \qquad (3\text{-}1)$$

該目標函式並受限於以下限制式：

$$\sum_{k \in P} x_{ik} \le 1, i \in M \qquad (3\text{-}2)$$

$$x_{ik} \le y_j, \quad i \in M, j \in N, k \in P_j \qquad (3\text{-}3)$$

$$x_{ik}, y_j \in \{0,1\} \quad i \in M, j \in N, k \in P \qquad (3\text{-}4)$$

在每一個可能解 S，$S \subset N$，S 需滿足

$$y_j = \begin{cases} 1, index \quad j \in S \\ 0, otherwise \end{cases} \qquad (3\text{-}5)$$

$$x_{ik} = \begin{cases} 1, query\ i\ uses\ configuration\ k \\ 0, otherwise \end{cases} \qquad (3\text{-}6)$$

公式(3-1)表示建立索引會增加資料庫效能，但也會產生維護成本，把所有增益值總和減去建立索引花費的維護成本即是索引的效能，求其最大值即是我們要的結果。

公式(3-2)表示每一個查詢i最多能用一個組態(configuration)，P_j 表示P包含j索引的組態集。公式(3-3)表示查詢i使用的組態內之索引必須存在。公式(3-4)表示 x_{ik}，y_j 的值只有0或1。公式(3-5)表示若索引j屬於解集合S，則 y_j=1，否則 y_j=0。公式(3-6)表示查詢i使用組態k，則 x_{ik}=1，否則 x_{ik}=0。

3.2 家族競爭演算法

家族競爭演算法的做法，是在每一父代和其它父代交配突變，在同一個父代所產生的子孫中找出最佳解，當成下一次的父代。因此每一個父代將產生一個族群，族群之間在下一輪迴中又將彼此競爭，藉此增加求解過程中的多樣性及變化性。以下圖4，是家族競爭演算法的示意圖。

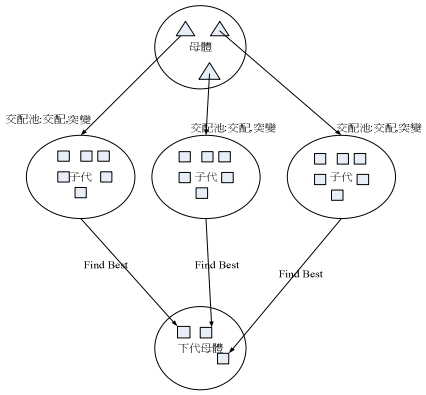

圖 4 是家族競爭演算法的示意圖

比較ISP家族競爭基因演算法和Kratica基因演算法最大的不同，是要把每一個父代看以一個家族，由此家族基因最好者和其它家族基因最好者交配和突變產生下一代。產生的下一代歸此父代的子孫，視為同一家族。同時在由子孫中，找出基因最好者，當下一父代。除此之外，本論文在交配步驟，採用均勻交配(uniform crossover)，突變用位元

突變方式，選擇用家族競爭，決定產生多少子代的參數L則採固定值。如圖5所示，為一個ISP家族競爭基因演算法的流程圖。

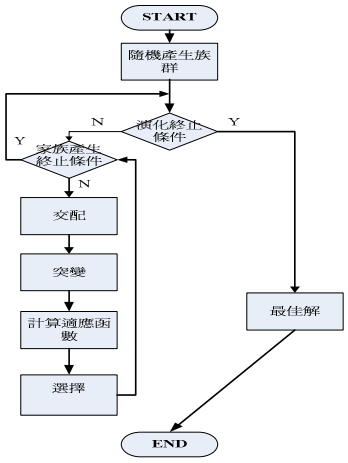

圖 5 ISP家族競爭基因演算法流程圖

在本研究中的基因演算法，幾個設定較為特殊的步驟，再額外說明如下所示。

● 適應函數：

每一個組態中所有查詢增益值的總合，就是適應函數的回傳值。

● 交配:

在交配過成使用均勻交配法[5]，傳入一交配遮罩參數 P_{CRO} ，用此參數控制交配基因位元。

● 突變:

在突變收斂速度,由 pm_{start} 至 pm_{end} ，N_{gen} 是第幾代產生後代， pm_{gen} 是一乖離(Bias)參數值。當 N_{gen} 值愈大時突變的機率愈小。

$$pm = pm_{end} + (pm_{start} - pm_{end}) \cdot 2^{\frac{-N_{gen}}{pm_{gen}}}$$

● 選擇:

在經由交配產生子代的族群中，在同一家族中找最好適應值，當成一下個父代。

4. 系統實作與實驗結果

本節主要在說明系統實作的架構、環境和結果。並分析比較窮舉法、Kratica基因演算法和ISP家族競爭基因演算法，解索引選擇問題的結果。本論文的系統實做，主要分成四個單元: 案例產生器、窮舉法、Kratica基因演算法和ISP家族競爭基因演算法。

4.1 案例資料及效益值函數

案例資料(Instances))是指資料庫的查詢(Query)和索引(Index)的互動行為。在Kratica基因演算法中案例資料是用Class A[1]，Class A包括建立索引的維護成本，和索引與查詢的組態狀況，其建立索引的維護成本是固定值。但在實際資料庫建不同索引不可能所有索引維護成本都是一樣的。所以在本論文的案例資料產生器，將索引的維護成本 f_j 修改為以亂數的方式產生。

除此之外，亂數對於讀出的查詢也會有一些不合理的現象。例如查詢Q1的屬性是讀出時，使用索引index1增益值是50，使用索引index2增益值是80，如果同時使用索引index1和索引index2，因亂數產生增益值，就可能產生低於80的不合理值。本研究案例資料產生器程式是修改Class A[11]產生器程式，把索引維護成本、查詢屬性和增益值限制條件加入程式中。

每一組增益值模擬資料包含查詢的代碼、索引組態及增益值三項資料。在產生增益值的過程中，我們只隨機產生部份組態的增益值，沒有產生的就當作零。這是因為如果要把所有組態的增益值都產生出來，所耗費的紀錄空間會太大。例如，50個索引及20個查詢，就必須紀錄 20×2^{50} 個值，這樣的做法太耗資源也沒有必要。實際上，每個案例我們只產生了2500組增益值資料，在實驗中，這些資料已經足以讓我們觀察到不同演算法的影響。

三種演算法，都需要用到計算效益值的函數cal_gain()。cal_gain()的輸入，為一代表索引建立狀況的二位元字串。此二位元字串長度表示索引個數大小，在每一位元中，0表示索引不建立，1表示索引建立。我們使用二位元字串的值，來檢查模擬案例資料中有那些索引組態是可用的，求出每個查詢使用這些可用索引組態的增益最大值。把每個查詢得的增益值累加，並加總建立索引的維護成本，把增益值總計減維護成本總計即是效益值，也就是cal_gain()的回傳值。

4.2 實驗結果及分析

為了驗證本篇論文提出方法之效能，用窮舉法和Kratica基因演算法[7]，一同比較結果。三種方式實驗條件如下:

1. 窮舉法：用可建立N個索引條件下，則會有 2^N 可能解，因窮舉法太花時間，只建立25個索引和部分30個索引解。
2. Kratica 基因演算法：在初代為母體 200 個，在選擇參數 Ftour=5.6，均勻交配參數 Pcro=85，突變參數 Pmstart=0.01，Pmend=0.002，Pmgen=300，執行 300 代子孫，、分別產生索引數為 25、 30、 35 、40 、45 、50 的解。
3. ISP 家族競爭基因演算法：初代為母體 200 個，子代競爭數 L=3，均勻交配參數 Pcro=85，突變參數 Pmstart=0.01，Pmend=0.002，Pmgen=300，執行 100 代和 300 代子孫，分別產生索引個數為 25、 30、 35 、

40 、45 、50 的解。

在表 1 中，三種演算法以 25 個索引解做比較。t200111 表示案例編號，200 表示增益值產生參數，111 表示亂數種子。增益值參數的可能值包括 200、175、150、125、100、75、50、25，值越大代表增益的效果越強。亂數種子的可能值則包括 111、222、333、444、555。

效益值表示查詢增益值總計減索引維護成本總計，以案例編號前四碼做為群組分類，每個群在不同的亂數種子的效益值都差不多，效益值從上到下都呈群組遞減方式，符合控制增益值參數由大至小排列。在表中效益值為負值表示索引建立的維護成本比增益值高，不值得建立此索引組態。實驗結果窮舉法雖可以找到最佳解，但所花的時間太長，約是其它二種方式的 50 多倍。在 Kratica 基因演算法中，每一測試案例資料模式都需要 10 秒，ISP 家族競爭基因演算法則需要約 7 秒。且測試案例資料模式在家族競爭基因演算法所得的效益值都比 Kratica 基因演算法還好，因此，ISP 家族競爭基因演算法比 Kratica 基因演算法有更好的效能與效益。

表 1 索引個數 n=25 的三種演算法測試結果

案例資料	窮舉法		Kratica 基因演算法		ISP 家族競爭基因演算法	
	時間(sec)	效益值	時間(sec)	效益值	時間(sec)	效益值
t200111	563.906	6132	9.891	5524	6.985	5918
t200222	601.062	6568	9.813	5956	7.578	6486
t200333	559.781	5982	9.672	5417	7.015	5912
t200444	567.875	6214	9.750	5634	7.047	6130
t200555	547.781	6072	9.703	5634	6.844	5915
t175111	562.297	5161	9.875	4672	7.031	5022
t175222	600.797	5563	9.906	4960	7.438	5323
t175333	559.813	5034	9.922	4592	7.093	4897
t175444	567.782	5231	9.859	4917	7.032	5040
t175555	547.859	5100	9.500	4709	6.828	4892
t150111	562.469	4226	9.562	3579	7.032	4096
t150222	602.015	4554	9.641	4097	7.453	4403
t150333	560.063	4116	9.500	3497	6.922	3963
t150444	567.797	4266	9.594	3864	7.250	4049
t150555	548.234	4177	9.781	3827	6.781	4093
t125111	562.438	3327	9.469	2866	6.937	3158
t125222	601.046	3609	9.797	3128	7.375	3434
t125333	559.954	3223	9.890	2960	7.047	3081
t125444	567.734	3321	9.672	2949	7.062	3137
t125555	548.109	3272	9.609	2784	6.766	3115
t100111	562.500	2459	9.890	1763	7.031	2391
t100222	600.875	2666	9.766	2201	7.562	2551
t100333	560.234	2350	9.672	1811	7.000	2187
t100444	567.656	2414	9.297	2163	6.937	2317
t100555	549.313	2387	9.515	1902	6.797	2191
t075111	562.515	1600	9.735	834	6.906	1353
t075222	600.859	1744	10.047	1399	7.578	1721
t075333	559.875	1564	9.812	1001	7.047	1521
t075444	567.969	1619	9.782	1242	7.047	1480
t075555	548.218	1525	9.750	1174	6.875	1278
t050111	562.500	838	9.625	203	7.047	551
t050222	600.954	910	9.703	517	7.313	712
t050333	561.328	815	9.734	464	7.046	522
t050444	569.141	924	9.531	302	6.985	695
t050555	549.328	734	9.640	487	6.829	587
t025111	563.859	229	10.750	-1410	7.031	-69
t025222	601.750	193	10.563	-1349	7.469	53
t025333	561.265	207	10.515	-902	6.891	76
t025444	568.625	291	10.625	-1206	7.156	-49
t025555	549.125	135	10.031	-803	6.766	-10

在表 2 中，比較 ISP 家族競爭基因演算法和 Kratica 基因演算法在 40 個索引與 50 個索引的測試結果，每一個測試案例資料模式不管在 40 個索引個數或 50 個索引個數，ISP 家族競爭基因演算法所花的時間都比基因演算法還少，效益值都較高，更加說明家族競爭基因演算法會比 Kratica 基因演算法在解索引選擇問題效能還好。

表 2 ISP家族競爭與Kratica基因演算法40個索引與50個索引測試結果

案例資料	40 個索引				50 個索引			
	時間(sec)		效益值		時間(sec)		效益值	
	Kratica	家族競爭	Kratica	家族競爭	Kratica	家族競爭	Kratica	家族競爭
t200111	15.000	8.985	5461	6082	19.125	10.344	5219	5653
t200222	15.219	9.500	5792	6270	18.641	10.625	5127	5831
t200333	14.750	9.078	5202	5712	18.734	10.531	4859	5610
t200444	14.953	9.344	5283	5868	18.922	10.594	5192	5669
t200555	15.375	9.219	5452	6184	18.891	10.484	5455	5888
t175111	15.109	9.140	4287	5072	18.406	10.281	4273	4839
t175222	15.266	9.500	4496	5033	18.641	10.750	4301	4703
t175333	14.625	8.969	4532	4651	18.906	10.406	3933	4512
t175444	15.031	9.235	4239	4920	18.406	10.547	4127	4445
t175555	15.188	9.281	4456	5180	19.031	10.593	4516	4924
t150111	15.000	9.047	3462	4238	18.656	10.250	3113	3548
t150222	15.078	9.328	3335	4050	18.500	10.672	3244	3707
t150333	14.953	9.063	3190	3756	18.672	10.500	2752	3296
t150444	15.172	9.265	3513	4102	18.812	10.531	3178	3766
t150555	15.437	9.172	3648	3995	18.391	10.578	3226	4011
t125111	14.938	8.984	2355	3055	18.719	10.359	2339	2556
t125222	14.968	9.516	2264	3011	18.375	10.656	2556	2624
t125333	14.687	9.047	2012	2815	18.750	10.500	1427	2423
t125444	15.125	9.234	2411	2850	18.984	10.656	1806	2526
t125555	15.141	9.109	2479	3030	18.562	10.469	2246	2774
t100111	15.032	9.078	1448	2121	18.437	10.297	1278	1708
t100222	15.218	9.484	1651	2032	18.750	10.750	1173	1630
t100333	15.078	9.172	1235	1807	18.610	10.422	1236	1547
t100444	14.922	9.171	1623	1999	18.859	10.562	700	1580
t100555	15.281	9.235	1667	1960	18.906	10.594	1241	1877
t075111	15.063	9.094	826	1070	18.484	10.312	-764	682
t075222	15.157	9.438	545	1337	18.484	10.641	-346	708
t075333	14.891	9.157	378	861	18.672	10.500	-1320	278
t075444	15.109	9.281	445	1161	18.609	10.516	373	618
t075555	14.922	9.234	799	1118	18.719	10.515	370	876
t050111	15.078	8.922	-1720	276	18.985	10.344	-1785	-178
t050222	15.110	9.359	-1131	97	18.797	10.640	-2179	-284
t050333	14.860	9.172	-987	21	18.891	10.469	-2637	-479
t050444	14.813	9.344	-881	329	18.671	10.641	-1651	-450
t050555	15.187	9.125	17	217	18.938	10.640	-1483	-202
t025111	15.297	8.968	-2152	-443	18.890	10.297	-3520	-955
t025222	15.328	9.329	-2087	-512	18.641	10.656	-2734	-868
t025333	15.172	9.188	-2322	-765	18.454	10.360	-2427	-1384
t025444	15.375	9.156	-1918	-421	18.281	10.515	-1609	-1069
t025555	15.000	9.141	-1913	-403	18.672	10.500	-2496	-960

5. 結論與未來研究方向

針對 ISP 問題，窮舉法雖可以找到最佳解，但所花的時間太長。而 Kratica 基因演算法雖不用花太多時間找最佳解，但必需在選擇方法時輸入競爭選擇參數來尋找最佳近似值，而且所花的時間比 ISP 家族競爭基因演算法多，效益值比 ISP 家族競爭基因演算法低。ISP 家族競爭基因演算法讓每個家族有獨立的交配和突變行為，在選擇方式直接找家族最好的，減少調整競爭參數，除了執行效率較高外，更有較好的效益。

不管用Kratica基因演算法或家族競爭演算法所得到的解都是近似解，還是有誤差值，即使增加子孫代數，也無法很快逼近最佳解，如何在短時間把誤差值縮小甚至等於最佳解是重要的研究方向。

參考文獻

[1] A. Caprara and J.J Salazar Gonzalez,"Separating Lifted Odd-hole Inequalities to Solve the Index Selection Problem",Discrete Applied

Mathematics, 92:111–134, 1999.
[2] A. Caprara, M. Fischetti, and D. Maio,"Exact and Approximate Algorithms for the Index Selection Problem in Physical Database Design", IEEE Transactions on Knowledge and Data Engineering, 7(6):955-967, 1995.
[3] A. Caprara and J.J. Salazar Gonzalez,"A Branch-and-Cut Algorithm for a Generalization of the Uncapacitated Facility Location Problem", Trabajos de Operativa TOP, 4(1):135–163, 1996.
[4] D. W. Tcha and B.Y. Lee,"A Branch-and-Bound Algorithm for the Multilevel Uncapacitated Facility Location Problem", European Journal of Operational Research,18(1):35–43, 1984.
[5] G. Syswerda,"Uniform Crossover in Genetic Algorithms", In 3th International Conference on Genetic ALgorithms, ICGA'89, pp. 2–9, 1989.Morgan Kaufmann.
[6] Jinn-Moon Yang and Cheng-Yan Kao,"A Family Competition Evolutionary Algorithm for Automated Docking of Flexible Ligands to Proteins",IEEE Transactions on Information Technology in Biomedicine, Vol. 4, No. 3,pp.227-229,September 2000.
[7] Jozef Kratica, Ivana Ljubic ,and Dusan Tosic,"A Genetic Algorithm for the Index Selection Problem",In Günther R. Raidl et al., editors, Applications of Evolutionary Computing: EvoWorkshops2003, Vol 2611, LNCS, pp.281-291, University of Essex, England, UK, 14-16 April 2003.
[8] J. Krarup and P.M. Pruzan,"The Simple Plant Location Problem: Survey and Synthesis",European Journal of Operational Research, 12:36–81, 1983.
[9] V. Filipovic,"Proposition for Improvement Tournament Selection Operator in Genetic Algorithms (in serbian)",Master's thesis, Faculty of Mathematics, Belgrade, 1998.
[10] V. Filipovic, J. Kratica, D. Tosic, and I. Ljubic,"Fine Grained Tournament Selection for the Simple Plant Location Problem", In 5th Online World Conference on Soft Computing Methods in Industrial Applications, WSC5, pp.152–158, 2000.
[11] Y. A. Kochetov and E. N. Goncharov,"Probabilistic Tabu Search Algorithm for the Multi-stage Uncapacitated Facility Location Problem",In B. Fleischmann, R. Lasch,U. Derigs, W. Domschke, and U. Rieder, editors, Operations Research Proceedings 2000, pp.65–70. Springer, 2000.