Evolving Speciated Checkers Players with Crowding Algorithm

Kyung-Joong Kim
Dept. of Computer Science, Yonsei University
134 Shinchon-dong, Sudaemoon-ku,
Seoul 120-749, Korea
uribyul@candy.yonsei.ac.kr

Sung-Bae Cho
Dept. of Computer Science, Yonsei University
134 Shinchon-dong, Sudaemoon-ku,
Seoul 120-749, Korea
sbcho@cs.yonsei.ac.kr

Abstract-Conventional evolutionary algorithms have a property that only one solution often dominates and it is sometimes useful to find diverse solutions and combine them because there might be many different solutions to one problem in real world problems. Recently, developing checkers player using evolutionary algorithms has been widely exploited to show the power of evolution for machine learning. In this paper, we propose an evolutionary checkers player that is developed by a speciation technique called crowding algorithm. In many experiments, our checkers player with ensemble structure shows better performance than non-speciated checkers players.

I. INTRODUCTION

Checkers is a very simple game and easy to learn. Unlike chess, it is simple to move and needs a few rules [1]. Though simple to learn, there are master-level players and naïve players. Maybe, differences among them are experiences, skills and strategies. To teach these properties to machine is not an easy task. Human needs only small time to start the checkers and will improve his performance at each competition. Similarly, researchers attempt to develop game player including the iterated prisoner's dilemma, tic-tac-toe, and checkers by evolutionary algorithm [2]. With respect to checkers, the evolutionary algorithm was able to discover a neural network that can be used to play at a near-expert level without injecting expert knowledge about how to play the game [3,4]. Evolutionary approach does not need any prior knowledge to develop machine player but can develop high-level player.

However, conventional evolutionary algorithms have a property that only one solution often dominates in the last generation (genetic drift). In real world problems, diversity is very useful. To improve the diversity of a population, a number of speciation algorithms have been proposed [5]. In this paper, crowding algorithm is used to improve the diversity of a population. From the last generation, we choose representative checkers player from each species and combine them to play the checkers game.

Figure 1 shows how to obtain better performance by combining speciated multiple solutions. Usually, there are many solutions that have high fitness value in a search space. Speciation techniques can find diverse strategies that survive in genetic search. In this paper, diverse evolutionary checkers players found by speciation techniques are combined by a voting method. The combined player is compared with the fittest player evolved using a simple evolutionary algorithm.

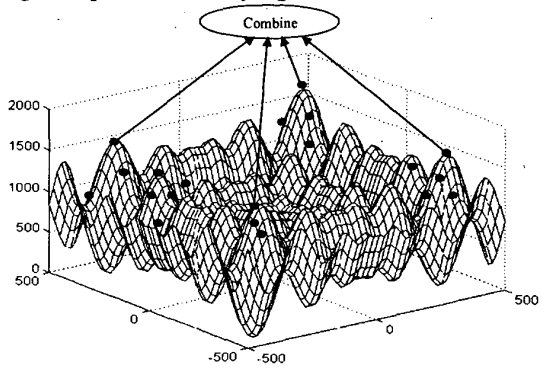

Figure 1. Schematic diagram of achieving better performance by combining speciated multiple solutions.

II. BACKGROUND

In this section, the rules of checkers game are explained. Checkers is a very simple game and needs few rules. Everyone can learn this game at a glance. However, to be the most competitive player, you need a lot fights against many other good players that have different strategies and experiences. Evolutionary algorithms simulate this learning procedure and speciation helps find diverse players.

2.1 *Playing Checkers*

Figure 2 shows opening board in a checkers game. Checkers board has 8 columns and 8 rows and each player has 12 pieces. Each player can move forward diagonally one square at a time. If possible, jumping over an opposing player into an empty square is allowed. In this case, opposing player dies. When a player advances to the last row of the board, it becomes a king who can move forward or backward diagonally. If there is no available player or movable player, the game is over. A draw may be declared upon mutual agreement of the players or in tournament play at the discretion of a third party under certain circumstances.

0-7803-7282-4/02/$10.00 ©2002 IEEE

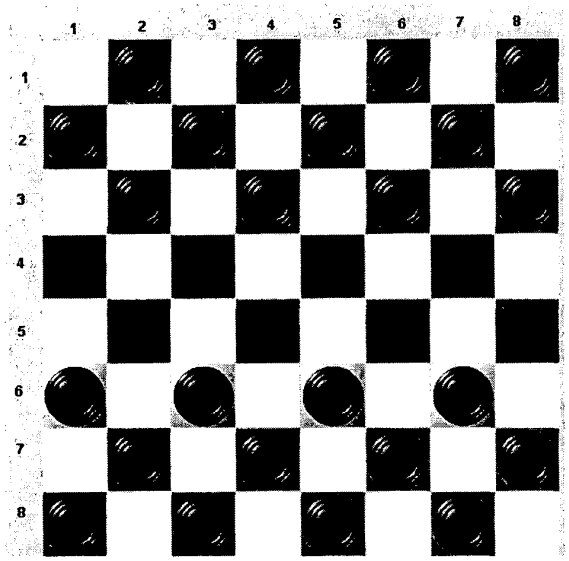

Figure 2. Opening board in a checkers game. The black player moves first (Upper part is red and lower part is black).

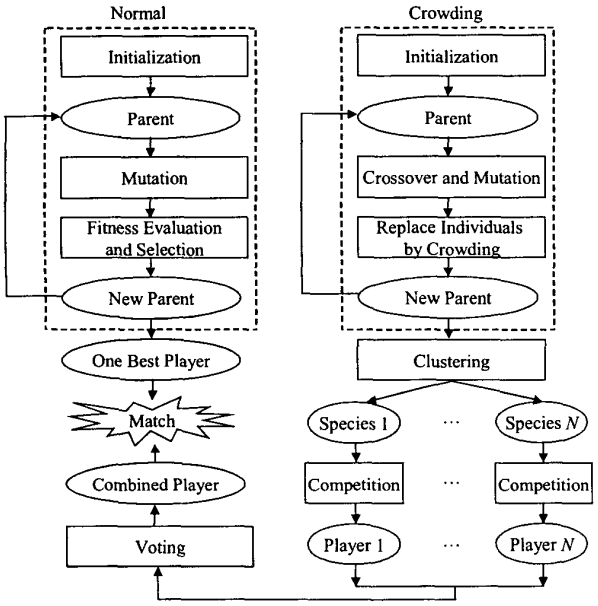

Figure 3. Evolutionary process of the proposed checkers game.

2.2 *Checkers Program*

There are two kinds of checkers programs. One uses end-game databases and expert knowledge, and the other does not use any human knowledge. The former is the program like CHINOOK that is world man-machine checkers champion [6,7]. The latter is the Chellapilla and Fogel's checkers that evolves checkers player by using neural network as evaluator [2,3,4]. Our basic evolutionary checkers system is based on theirs. Different points are to use a speciation technique of deterministic crowding and combine the multiple strategies obtained.

2.3 *Speciation Techniques*

There are many different speciation techniques such as explicit fitness sharing, implicit fitness sharing, and crowding [8]. Fitness sharing is a fitness scaling mechanism that alters only the fitness assignment stage of a GA. Sharing can be used in combination with other scaling mechanisms, but should be the last one applied to, just prior to selection [9]. An extension to the original fitness sharing is implicit sharing [10]. Crowding techniques insert new elements into the population by replacing similar elements [11].

III. EVOLVING CHECKERS PLAYERS

In this section, the evolution of checkers players and a speciation technique are explained. To improve the diversity of a population, deterministic crowding is adopted [8]. Density-based clustering algorithm is used to cluster the speciated population in the last generation [12,13]. From each cluster, one representative strategy is chosen by competition and these strategies are combined for better performance. Figure 3 shows the whole evolutionary process.

3.1 *Representation of Board*

A checkers board has 32 availabe positions where players can move checkers. One board is represented by a vector that has 32 elements. An element in a vector can have one of values $\{-K, -1, 0, 1, K\}$. Zero means an empty position and minus means an opposite player. K means a king.

3.2 *Evolving Neural Network Evaluator*

To find the next move of a player, a game tree is constructed with limited depth. Figure 4 shows a simple game tree. Evaluation of terminal nodes' quality is measured with the feed-forward neural network evolved. Basically, it has three hidden layers and each hidden layer has 91 nodes, 40 nodes, and 10 nodes, respectively. Weights of the neural network are determined by evolutionary procedure described in Figure 3.

Chess, for example, has an average branching factor of about 35, and games often go to 50 moves by each player, so the search tree has about 35^{100} nodes. Pruning allows us to ignore portions of the search tree that make no difference to the final choice, and heuristic evaluation functions allows us to approximate the true utility of a state without doing a complete search. The minimax algorithm is designed to determine the optimal strategy for MAX, and thus to decide what the best first move is [14].

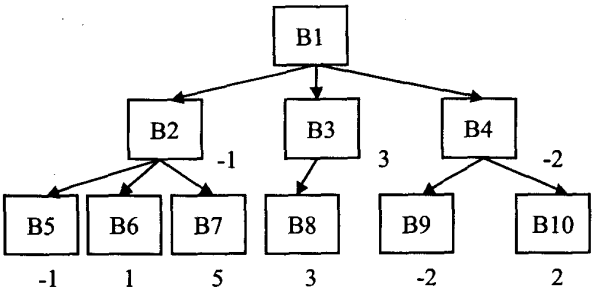

Figure 4. An example of a game tree.

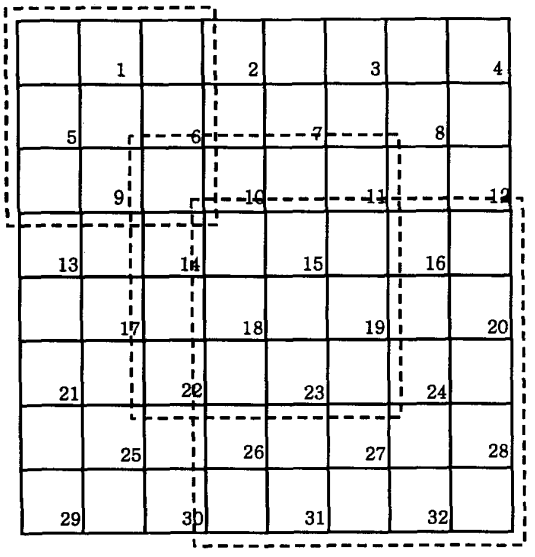

Figure 5. An example of 3×3, 4×4 and 5×5 sub-boards. The 3×3 sub-board contains 1, 5, 6 and 9 squares.

In this game tree, a board is an input of the neural network that produces degree of relevance of the input. One board can have 36 3×3 sub-boards, 25 4×4 sub-boards, 16 5×5 sub-boards, 9 6×6 sub-boards, 4 7×7 sub-boards and 1 8×8 sub-board. 91 sub-boards are used as an input of the feed-forward neural network. Figure 5 shows an example of 3×3, 4×4 and 5×5 sub-boards.

Each individual in a population represents a neural evaluator in a game tree. In fitness evaluation, each individual chooses five opponents from a pool and has a game with the players. The fitness increases 3 in a win while the fitness of an opponent increases 3 in a loss. In a draw, the fitness values of both players increase 1. After all games of individuals, fitness values of all players are determined.

Crowding algorithm is one of the representative speciation methods that attempt to discover diverse species in a search space [11]. The process of crowding is as follows:

- Step 1: Initialization of a population
- Step 2: Shuffling of the population
- Step 3: Insertion of individuals into a queue
- Step 4: Deque two individuals
- Step 5: Recombine and mutate two individuals
- Step 6: Make two pairs of similar offspring and parent
- Step 7: Choose a fit individual in the pair
- Step 8: Enque two individuals into a new queue
- Step 9: Unless the first queue is empty, goto Step 4
- Step 10: Define the second queue as a new population
- Step 11: Unless the generation exceeds Maximum, goto Step 2

The similarity between two neural networks is based on the Euclidean distance of weights and biases of them.

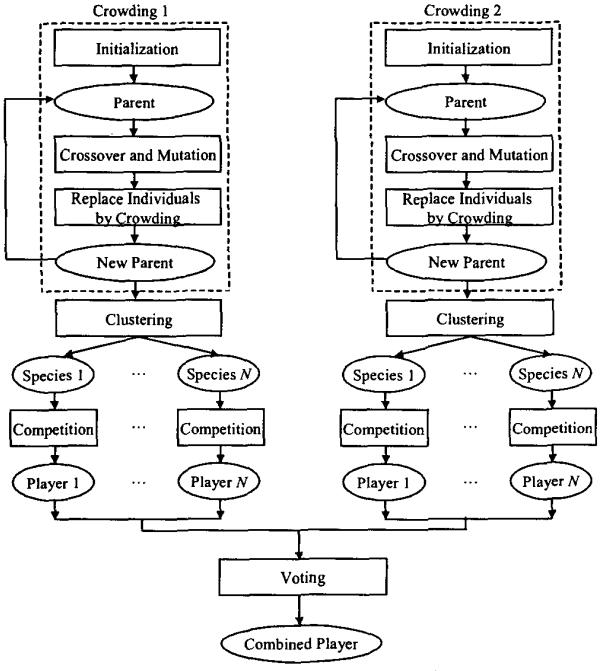

Figure 6. An example of grouping (group size is two).

3.3 *Combining Players*

Moves of combined players are determined using simple voting mechanism. Moves that are selected by many players are chosen. N players' decisions are combined as follows.

$$F(e(x))=S(j) = \max (S(i)) \ (i=1,\ldots, M)$$

$$S(i) = \sum_{k=1}^{N} G_k(i)$$

$G_k(i)$ is 1 if $e(x) = i$, otherwise 0

The number of available positions of a current board is M. Each player can choose one from M positions. Each player's choice is $e(x)$ on the current board status x. A

combined player's choice is $F(e(x))=j$. The number of players is N.

For better performance, grouping representatives in many experiments is proposed. The group size is defined as the number of experiments. Figure 6 explains the concept of grouping with many experiments. In this figure, two experimental results are grouped. Grouping many experiments combines several players speciated. Group size is L. Each sub-group contains $N(1)$, $N(2)$, .., $N(L)$ players. The number of available positions of a current board is M.

$$F(e(x))=S(j) = \max (S(i)) \ (i=1,\ldots, M)$$

$$S(i) = \sum_{l=1}^{L} \sum_{k=1}^{N(l)} G_k(i)$$

$G_k(i)$ is 1 if $e(x) = i$, otherwise 0

3.4 *Clustering Algorithm*

To select the best players, DBSCAN clustering method is adopted [12,13]. In the last generation, clustering algorithm identifies different species. The best player of each species is chosen by league of players in that species.

A density-based cluster is a set of density-connected objects that is maximal with respect to density-reachability. Every object not contained in any cluster is considered to be noise. DBSCAN searches for clusters by checking the Eps-neighborhood of each point in the database. If the Eps-neighborhood of a point p contains more than MinPts, a new cluster with p as a core object is created. DBSCAN then iteratively collects directly density-reachable objects from these core objects, which may involve the merge of a few density-reachable clusters. The process terminates when no new point can be added to any cluster. The basic terms of DBSCAN are defined as follows [12].

Definition 1: (Eps-neighborhood of a point) The Eps-neighborhood of a point p, denoted by $N_{Eps}(p)$, is defined by $N_{Eps}(p) = \{q \in D | \ dist(p,q) \le Eps\}$.

Definition 2: (directly density-reachable) A point p is directly density-reachable from a point q with respect to Eps, MinPts if

1) $p \in N_{Eps}(q)$ and
2) $|NEps(q)| \ge MinPts$ (core point condition).

Definition 3: (density-reachable) A point p is density-reachable from a point q with respect to Eps and MinPts if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i.

Definition 4: (density-connected) A point p is density connected to a point q with respect to Eps and MinPts if there is a point o such that both, p and q are density-reachable from o with respect to Eps and MinPts.

Definition 5: (cluster) Let D be a database of points. A cluster C with respect to Eps and MinPts is a non-empty subset of D satisfying the following conditions:

1) $\forall p, q$: if $p \in C$ and q is density-reachable from p with respect to Eps and MinPts, then $q \in C$. (Maximality)
2) $\forall p, q \in C$: p is density-connected to q with respect to EPS and MinPts. (Connectivity)

Definition 6: (noise) Let $C_1,\ldots,C_k$ be the clusters of the database D with respect to Parameters Eps_i and $MinPts_i$, $i=1,\ldots,k$. Then we define the noise as the set of points in the database D not belonging to any cluster C_i, i.e. noise = $\{p \in D \mid \forall i: p \notin C_i\}$.

Lemma 1: Let p be a point in D and $|N_{Eps}(p)| \ge MinPts$. Then the set $O=\{o \mid o \in D$ and o is density-reachable from p with respect to Eps and MinPts$\}$ is a cluster with respect to Eps and MinPts.

Lemma 2: Let C be a cluster with respect to Eps and MinPts and let p be any point in C with $|N_{Eps}(p)| \ge MinPts$. Then C equals to the set $O=\{o \mid o$ is density-reachable from p with respect to Eps and MinPts$\}$.

TABLE 1. PARAMETERS OF EXPERIMENTS

	Simple GA	Speciated GA
Population Size	100	100
Mutation rate	0.01	0.01
Crossover rate	-	1.0
Generation	50	50

TABLE 2. RESULTS OF EXPERIMENTS

	1:1	1:20
Simple GA Win	24	16
Speciated GA Win	22	37
Draw	22	15
Total Game Number	68	68

TABLE 3. SUMMARY OF THE EXPERIMENTAL RESULTS

	Player	Win	Lose	Draw
1	The coalition of speciated individuals	98	76	76
	Non-speciated individual	76	98	76
2	The coalition of speciated individuals	84	60	56
	Speciated individual	60	84	56
3	The coalition of more than three speciated individuals	104	101	95
	The coalition of two speciated individuals	101	104	95
4	The coalition of speciated individuals	111	22	117
	The coalition of non-speciated individuals	22	111	117

IV. EXPERIMENTAL RESULTS

Table 1 summarizes parameters of simple GA and speciated GA. Table 2 shows the results of experiments. Table 2 shows that the combination of twenty speciated players show better performance than one general player.

There are 68 games in 1:1 match because the number of speciated players is 68. In this match, Simple GA is a bit better than Speciated GA. After combining speciated players, performance gap between two players is large. Voting with duplication means that one can choose a player one or more times. In 68 matches, twenty players are selected at random.

Figure 7 shows one game played by twenty individuals and their choices. In this game, their combination defeats the fittest player. From first player to the 20th player, they vote 1 to previous player that submits the same movement.

If no such player, they vote 1 to self. In this figure, 1, 3 and 10 players dominate the opinion of combined player. Row means the number of votes in each movement for each player and column means one player's number of vote in a game. Shade cell is a selected player for each movement.

Figure 8 shows a dendrogram of population evolved using speciation method. Dendrogram is used to understand the diversity of population. To draw dendrogram, it is required to compute the dissimilarity between two objects in a population and do single linkage clustering [15]. Each individual matches with other 99 individuals and records win, lose, or draw. Each individual is represented with a vector of 100 elements. Each element represents the game result with other 99 players and itself. Elements are one of $\{-1,0,1\}$. They represents lose, draw and win. Match with itself is draw. The dissimilarity of two vectors is sum of different elements at the same position in two vectors.

Non-speciated evolutionary algorithm sets population size as 10 and generation number as 50. Speciated evolutionary algorithm sets population size as 100 and generation number as 50. Mutation rate is 0.01 and full crossover is adopted. The number of leagues is 5. Evolving checkers using speciation needs 10 hours in Pentium III 800MHz (256MB RAM). In the experiments, 5 best players are evolved from 5 runs of non-speciated evolutionary algorithm. 10 speciated players are evolved from 2 runs of speciated evolutionary algorithm. In this case, each speciated evolutionary algorithm produces 5 strategies. The number of the best players in the last generation is not determined because clustering algorithm can identify 1,2,3,4 and more species after clustering.

The number of the best players depends on the number of species in the last generation. Non-speciated evolutionary algorithm uses only mutation but speciated evolutionary algorithm uses crossover and mutation. Non-speciated evolutionary algorithm is the same with original Chellapilla and Fogel's checkers. Table 3 summarizes the results.

V. CONCLUSIONS

In this paper, neural network is used to validate board and min-max search finds optimal board. Neural network evaluator is evolved using evolutionary algorithm. Evolutionary algorithm has a shortcoming that finds only one high-fitness solution. Like other problems in real world, evolving checkers also needs the diversity of population. To improve the diversity of population, speciation method is applied to simple evolutionary algorithm.

In this paper, crowding algorithm is applied to original evolutionary algorithm. In the last generation, we cluster the individuals of population and choose one representative player from each species. From the experimental result, players that are evolved using the speciation method show higher performance than the best. Combining diverse players shows better performance than solo player. However, the more player is, the poorer performance combined player is.

Future works are evolving population using another speciation method like fitness sharing and compare our works with original Chellapilla and Fogel's checkers.

\	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1		0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2		0	6	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0
3	3	0	6	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0
4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6		0	5	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0
7	2	0		0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0
8		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9		0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
10	2	0		0	0	0	0	0	6	0	0	0	0	0	6	0	0	0	0	0
11		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	2	0	5	0	0	0	0	2	0		0	0	0	0	0	0	0	0	0	0
14	2	0		0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0
15		0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
16		0	5	0	0	0	0	1	0	0	0	0	0	0	6	0	0	0	0	0
17	2	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	2	0		0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
19	2	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	2	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	2	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	8	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	5	0		0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0
26		0	7	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0
27		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7. Selection of players by voting principle.

Acknowledgments

This research was supported by Brain Science and Engineering Research Program sponsored by Korean Ministry of Science and Technology. The authors would like to thank to Mr. Donghyuk Kang and Inchang Park for their help to implement the system.

References

[1] D. Clark, "Deep thoughts on deep blue," *IEEE Expert*, vol. 12, no. 4, pp. 31, 1997.

[2] K. Chellapilla, and D. B. Fogel, "Evolution, neural networks, games, and intelligence," *Proc. IEEE*, vol. 87, pp. 1471-1496, Sept 1999.

[3] K. Chellapilla, and D. B. Fogel, "Evolving neural networks to play chekcers without relying on expert knowledge," IEEE Trans. On Neural Networks, vol. 10, pp. 1382-1391, Nov 1999.

[4] K. Chellapilla, and D. B. Fogel, "Evolving an expert checkers playing program without using human experties," *IEEE Transaction on Evolutionary Computiona*, vol. 5, no. 4, pp. 422-428, August 2001.

[5] S. Mahfoud, "Niching methods for genetic algorithms," *Doctoral Dissertation*, University of Illinois Urbana, 1995.

[6] J. Schaeffer, R. Lake, P. Lu and M. Bryant, "Chinook: The world man-machine checkers champion," *AI Magazine*, vol. 17, no. 1, pp. 21-29, 1996.

[7] J. Shaeffer, J. Culberson, N. Treloar, B. Knight, P. Lu, and D. Szafron, "A world championship caliber checkers program," *Artificial Intelligence*, pp. 273-290, vol. 53, no. 2-3, 1992.

[8] T. Back, D. B. Fogel, and T. Michalewicz, *Evolutionary Computation 1 & 2*, IOP Publishing Co, 2000.

[9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Massachusetts, 1989.

[10] P. Darwen and X. Yao, "Every niching method has its niche: Fitness sharing and implicit sharing compared," *Proc. of Parallel Problem Solving from Nature (PPSN) IV*, vol. 1141, Lecture Notes in Computer Science, Springer-Verlag, Berlin, pp.398-407, 1996.

[11] S. Mahfoud, "Crowding and preselection revisited," *Parallel Problems Solving from Nature*, vol. 2, pp. 27-36, 1992.

[12] M. Ester, H.-P. Kriegel, J. Sander and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Knowledge Discovery and Data Mining 1996*, pp. 226-231, 1996.

[13] J. Han, and M. Kamber, *Data Mining Concepts and Techniques*, Morgan Kaufmann, 2001.

[14] S. Russell and P. Norvig, *Artificial Intelligence: A mordern approach*, Prentice Hall, 1995.

[15] A. D. Gordon, *Classification: Methods for the Exploratory Analysis of Multivariate Data*, Chapman and Hall, 1981.

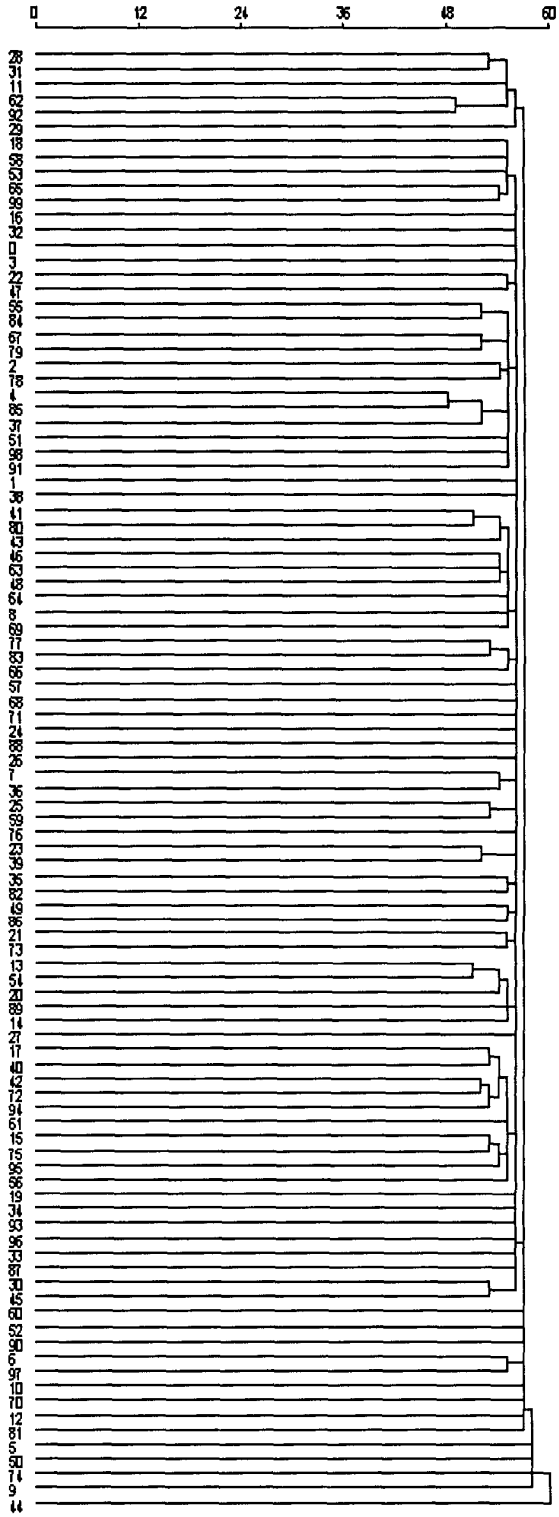

Figure 8. Dendrogram of speciated population.