Optimization of an Evaluation Function of the 4-sided Dominoes Game Using a Genetic Algorithm

Nirvana S. Antonio, Cícero F. F. Costa Filho, Marly G. F. Costa, Rafael Padilla

Abstract— **In 4-sided dominoes, the popular way of playing dominoes in Amazonas State, the strategies used for the game are more complex than those adopted in the more traditional 2-sided dominoes, the most popular dominoes game played in Brazil. This work presents the optimization of an evaluation function for the best move in 4-sided dominoes using a genetic algorithm. The evaluation function is composed of terms that incorporate the game's strategies and are defined as: punctuating, facilitating future moves and complicating opponents' moves. Coefficients were defined to determine the importance of each term of the evaluation function and a set of parameters and operators for implementation of the genetic algorithm. The players' ability was calculated by the number of wins in 5,000 matches. The results obtained during the simulations showed that the team (composed of 2 players) that used the evaluation function with its coefficients optimized by the genetic algorithm won in more than 70% of the total matches.**

I. Introduction

Dominoes is a game consisting of rectangular pieces divided in 2 halves, which are marked with black dots indicating a numeric value. Possibly having originated in China, dominoes is a popular game all over the world and can be played in many different ways, depending on the region. In Brazil, the most popular way is called "2-sided dominoes". However, in Amazonas State, another variation of the game is more popular among the locals, dominoes played on four sides.

4-sided dominoes is characterized by being a multi-agent problem [1], because it is not a zero sum game, played by four players divided in two pairs, which have imperfect information about the possible moves of the other. When played by 2 pairs, it is considered a two-player game, as there are only two scores. Therefore, each player must develop strategies based on incomplete information in order to win the game with his/her partner.

N.S. Antonio is with *Universidade Federal do Amazonas/Centro de P&D em Tecnologia Eletrônica e da Informação* – UFAM/CETELI. Manaus. Amazonas. Brasil (tel: +55 92 8414 1406; e-mail: nirvana.sa@gmail.com).

C. F. F. Costa Filho is with *Universidade Federal do Amazonas/Centro de P&D em Tecnologia Eletrônica e da Informação* – UFAM/CETELI. Manaus. Amazonas. Brasil (tel:: +55 92 91464954; e-mail: ccosta@ufam.edu.br).

M. G. F. Costa is with *Universidade Federal do Amazonas/Centro de P&D em Tecnologia Eletrônica e da Informação* – UFAM/CETELI. Manaus. Amazonas. Brazil (tel: +55 92 9128 2404; e-mail: mcosta@ufam.edu.br).

R. Padilla is with *Universidade Federal do Amazonas/Centro de P&D em Tecnologia Eletrônica e da Informação* – UFAM/CETELI. Manaus. Amazonas. Brasil (tel: +55 92 9165 1123; e-mail: eng.rafaelpadilla@gmail.com).

The use of an evaluation function associated with the alpha-beta algorithm for choosing the best move in games of perfect information was first proposed by Shannon [3]. This proposal was implemented for chess [4] and checkers [5]. For games of imperfect information, the use of Shannon's proposal involves complex reasoning about game strategy. It is mentioned for example in the Bridge Baron program [6], developed for Bridge. For 2-sided dominoes, some studies can be found in the literature seeking the best game strategy [7] [9].

In [2], we developed a methodology for choosing the best move based on an evaluation function that combines in its terms information about the present state of the game. To evaluate the developed evaluation function, simulations were conducted for 4-sided dominoes games, in which a pair used the evaluation function for choosing the best move while the other pair performed their moves based only on the dots of the dominoes on the table. The choice of the parameters of the evaluation function was done manually and as result, the first pair won the game in more than 66% of the simulations.

In this study, we will perform the optimization of the parameters that make up the evaluation function applied to some strategies that can be adopted by the player. The optimization will be achieved by the optimization and search technique of the genetic algorithms (GA). GA is a very efficient technique for searching for optimal or near-optimal solutions. Moreover, it is easy to implement and allows for parallel computing. With the optimization process, we hope that the pair that uses the evaluation function to choose their moves performs better than their opponents that use only the punctuations of the pieces on the table as a criterion to choose their moves, considered a basic strategy used by beginners.

II. DOMINOES GAME

Dominoes is a game consisting of 28 rectangular and flat pieces (or tiles). These pieces are divided in two halves that, in the classic form of the game, contain numbers ranging from zero to six dots, marking all combinations between these numbers. It can be played by four players, in pairs or individually, where each player receives seven pieces or, alternatively, can be played by two players, where the 14 remaining tiles are "bought" during the match. There are several ways to play dominoes and the most common form in Brazil is called 2-sided dominoes. However, in Amazonas State, it is played in another less popular way, 4-sided dominoes.

978-1-4577-0011-8/11/$26.00 ©2011 IEEE

In 4-sided dominoes, the games are played by two pairs. Each player must have seven tiles in hand at the beginning of each turn. In the first round, the first piece to be displayed is the 6-6, called "double six", having two halves numbered with 6 dots each. The players have the possibility to play off the 4 sides of this first piece. In Figure 1, we can see a game where 3 points have been opened from the "double six".

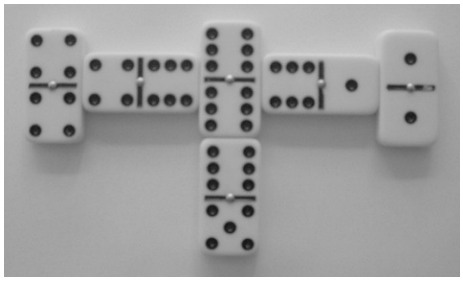

Fig. 1. Example of a move that is scored 15 points

In the following rounds, the player who "hit" (used up all their "stones" (tile pieces) before the other players) in the previous round will start with the double of his choice. The main objective of 4-sided dominoes is to achieve a score of 200 or more points, in one or more rounds of play. During the game, there is the possibility of scoring 5, 10, …, 50 points, in other words, multiples of 5. The chances of scoring can be defined by the following conditions:

P1) The sum of the dots on the ends is a multiple of 5 and the score recorded for the pair is equal to this sum. Figure 1 exemplifies a move equivalent to 15 points, because the sum of the dots on the ends of the double that started the game (2+5+8) is a multiple of 5. However, Figure 2 shows an example of a move where there is no score, because the sum of the dots on the ends is 14 (1+5+8).

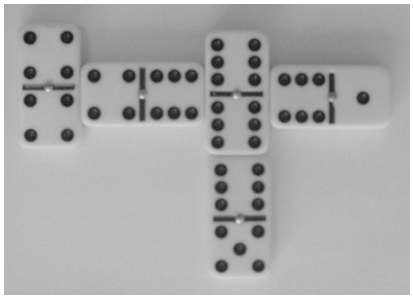

Fig. 2. Example of a move with no score

P2) The opponent player has no piece to fit in any of the 4 ends, so he "skips" his turn instead of playing, impeding an opponents' play is equivalent to a score of 20 points for the pair who caused the skipping.

P3) When the player causes all other players to pass (including his partner). This pass, called a "rooster" is equivalent to a score of 50 points.

P4) When a player "hits", the dots of the pieces left in the hands of the opponents are totaled. This sum is called "the garage". The score corresponds to the highest multiple of 5 less than or equal to this sum. Figure 3 shows an example where the "garage" is worth 10 points since the score of the opponents is 12.

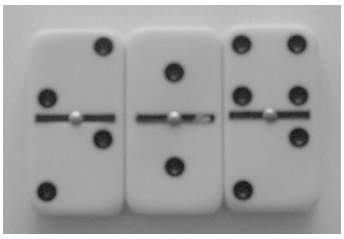

Fig. 3. "Garage" example of 10 points

P5) The last piece discarded by the player who goes out is a "double". The score of this move is equal to 20 points.

The game of 4-sided dominoes, having more elaborated goals and rules, adopts more complex strategies than those used in 2-sided dominoes, because in addition to scoring, the player should also try to facilitate his (or his partner's) future moves and impede the moves of their opponents. To illustrate the strategies adopted during the game, it is considered that at the beginning of a round, a player has in hand four or more pieces with the same number (Figure 4). In this scenario, the strategy is to try to make this same number present on the most possible sides, since this type of move could force his opponent to skip his turn or may facilitate future moves.

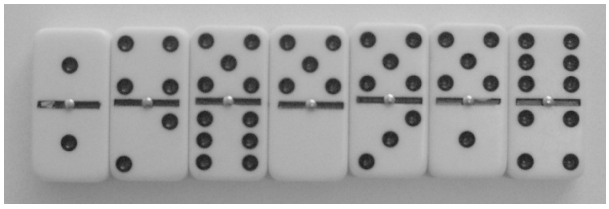

Fig. 4. Example of a set of initial tiles of a player where four of them have the number 5

III. EVALUATION FUNCTION FOR SELECTING THE BEST MOVE

Previously we developed an intelligent agent for 4-sided dominoes that chooses the best move by an evaluation function [2]. The evaluation function consists of the sum of two terms, T_{n1} and T_{n2}, as defined by equation (1).

$$f(n) = T_{n1} + T_{n2} \quad (1)$$

In the expression above, the variable n identifies a move for which the evaluation function is calculated. The term T_{n1} corresponds to points obtained by performing move n. The term T_{n2} incorporates the strategy of the game, corresponding to values that represent how choosing move n can facilitate future actions of the player and make future moves of the opponents even more difficult.

To calculate the terms T_{n1} and T_{n2} of the evaluation function, the current state of the game was used. The state of the game was defined as a set of 7 vectors, $V_i = \{a_i, b_i, c_i, d_i, e_i, f_i, g_i\}$, where a_i corresponds to the number of pieces (tiles) played with numbers 0; b_i corresponds to the number of pieces played with numbers 1, and so on. These vectors express statistics updated every move. The definition of states of 4-sided dominoes is shown below:

- V_0: quantity of pieces in the game for each number.
- V_1: quantity of pieces in the hands of the players for each number
- V_2: quantity of pieces for each number at the tile ends.
- V_3: quantity of pieces already played by the pair for each number.
- V_4: indicates the numbers where the following opponent has already passed. If a_4 is equal to 1, the following opponent has already passed the numbering 0. If a_4 is equal to 0, the following opponent still has not passed the numbering 0, and so on.
- V_5: indicates the numbers where the previous opponent has already passed.
- V_6: indicates the numberings where the partner has already passed.

The term T_{n1} incorporates in the calculation of the evaluation function the points obtained when the move option n is made through situations P_1, P_2, P_3 and P_5, as shown in equation (2).

$$T_{n1} = P_1 + P_2 + P_3 + P_5 \quad (2)$$

It is noteworthy that the T_{n1} does not incorporate situation P_4, which corresponds to the points of the "garage", because they cannot be counted while the round is not complete.

To understand the text that follows, let us define L as a possible option to play, which means, a tile in the player's hand. Each tile has numbers in its two halves. L_1 corresponds to the numbering of the half of the stone that matches one of the values in any tile ends of the game, and L_2 corresponds to the other half that does not match the values at the ends. Thus, if a player has the option to place a piece numbered 5-3 and the game on the table is prepared as illustrated in Figure 1, L_1 will receive the number 5, as it matches one side and L_2 will receive the number 3, which does not match the side.

The term T_{n2} includes in its calculation two distinct portions related to the game's strategy, as shown in equation (3):

$$T_{n2} = -E_1 + E_2 \quad (3)$$

Portion E_1 considers the possibility of forcing the subsequent opponent to skip his turn. To understand portion E_1, it will be used in the following hypothetical situation: let us suppose that in two of the tile ends there is the number n_1 and the player taking a turn has three pieces in his hands with the same numbering. So, five of the seven possible pieces with numbering n_1 do not belong to the opponent playing next; therefore, the probability of the opponent skipping his turn is higher if all the ends have this numbering. Given the possibility of making the next opponent skip his turn and, thus score 20 points, it is desirable that no piece with numbering $L_1 = n_1$ is discarded. In this case, the option to play a tile in which $L_1 = n_1$ is not desired from a strategic perspective, the negative sign for E_1 is given in the expression (3). The calculation of E_1 is performed according to expression (4).

$$E_1 = \alpha_1.V_0(L_1) + \alpha_2.V_1(L_1) + \alpha_3.V_2(L_1) \quad (4)$$

The more pieces with numbering L_1 on the ends (represented by vector V_2) and in the hands of the player (represented by the vector V_1), the greater the value of E_1 is. On the other hand, the value of E_1 should also be directly proportional to the number of pieces with the numbering L_1 already played (vector V_0). The coefficients α_1, α_2 and α_3 control the importance of each term of E_1 in relation to other parcels compromising the evaluation function.

The portion E_2 aims to facilitate future actions of the player. The calculation of E_2 is performed according to equation (5).

$$E_2 = \alpha_4.V_0(L_2) + \alpha_5.V_1(L_2) + \alpha_6.V_2(L_2) + \alpha_7.V_3(L_2) \quad (5)$$

This expression is very similar to the one proposed for E_1, but L_2 is used instead of L_1 as independent argument, which is the number the player wants to add on the table. There is also a term related to facilitating his partner's play, represented by the vector V_3. In this case, the term examines the pieces played by his partner to the numbering L_2 presented at the tile ends. The coefficients α_4, α_5, α_6 and α_7 control the importance of each term in E_2 in relation to other parcels that compose the evaluation function.

The evaluation function described in equations (1) - (5) involves the main cases considered by a player while deciding the best move at each round. In other words, it proposes a possible strategy to be adopted in 4-sided dominoes. In this work, this strategy will be called "Strategy 1" and its evaluation function will be used to calculate the fitness of the chromosomes during the execution of the genetic algorithm. At the end of the optimization, the genetic algorithm will find the coefficients α_1, α_2, α_3, α_4, α_5, α_6 and α_7 optimized for this strategy.

Besides the strategy outlined above, other strategies were analyzed in this work. In the second strategy, the player gives higher priority only to moves that can promote his own game; in other words, the coefficients α_1, α_2, α_3 (makes the opponents' moves more difficult) and α_7 (analyzes the moves that make his partner's game easier) are set to zero. Therefore, the evaluation function to be optimized by the genetic algorithm for Strategy 2 is made only by the terms of

equation (6). At the end of optimization, the genetic algorithm will find the coefficients α_4, α_5 and α_6 optimized for this strategy.

$$f(n) = T_{n1} + \alpha_4.V_0(L_2) + \alpha_5.V_1(L_2) + \alpha_6.V_2(L_2) \quad (6)$$

In the third strategy, the player only gives higher priority to moves that can promote his partner's game, in other words, the coefficients α_1, α_2, α_3 (makes his opponents' moves harder) and α_5 (analyzes moves that make his own game easier) are set to zero. Therefore, the evaluation function optimized by the genetic algorithm for Strategy 3 is formed only by the terms of equation (7). At the end of optimization, the genetic algorithm will find the coefficients α_4, α_6 and α_7 optimized for this strategy.

$$f(n) = T_{n1} + \alpha_4.V_0(L_2) + \alpha_6.V_2(L_2) + \alpha_7.V_3(L_2) \quad (7)$$

The fourth strategy gives higher priority to moves that make the opponents' actions more difficult, in other words, coefficients α_4, α_5, α_6 and α_7 (facilitates the moves of the pair) are set to zero. Therefore, the evaluation function to be optimized by genetic algorithm for Strategy 4 is formed only by the terms of equation (8). At the end of optimization, the genetic algorithm will find coefficients α_1, α_2 and α_3 optimized for this strategy.

$$f(n) = T_1 + \alpha_1.V_0(L_1) + \alpha_2.V_1(L_1) + \alpha_3.V_2(L_1) \quad (8)$$

The fifth strategy to be considered in this work is the basic strategy of 4-sided dominoes, being the strategy used by novice players. This strategy considers only the term T_{n1} of the evaluation function described in equation (1); in other words, only those points are considered for choosing the best move.

IV. OPTIMIZATION OF THE EVALUATION FUNCTION

A. Parameters of the Genetic Algorithm

In order to optimize the evaluation function using GA, we use a combination of parameters for better results, as described below.

Coding: the chromosomes are encoded as a vector of type double, which represent parameters α_1, α_2, α_3, α_4, α_5, α_6 and α_7 of the evaluation function.

Fitness function (*fitness*): the fitness function is responsible for measuring the performance of a chromosome during the simulation of domino matches. The fitness value of a chromosome is given by the number of wins resulting from the performed matches.

Population size: the size of the population directly affects the performance of the algorithm. Very small populations have less genetic diversity and can lead to a faster convergence of the algorithm to a local minimum. On the other hand, very large populations make the algorithm too slow [10]. For the optimization performed in this work, we use the following population sizes: 40, 60, 80 and 100.

Crossover for the optimization process, two techniques of crossing were used: *Two-point Crossover and Uniform Crossover*. The rate of crossover p_c, which defines the probability that the selected chromosomes pass by the process of crossing, will be 0.8 in this work.

Mutation: To implement the mutation process, the operator we chose was the Uniform Mutation and the value of mutation probability, p_m, takes the values 0.1 and 0.5.

Stopping criterion: as stop criterion of the genetic algorithm, we used the total number of generations in the following sizes: 50, 100, 150 and 200.

B. Implementation

For the simulation of 4-sided dominoes, a program was developed in Java with four agents acting as the four players required for the match. In this program, the first pair uses a basic strategy for choosing the best move and the second pair may have its parameters α_1, α_2, α_3, α_4, α_5, α_6 and α_7 adjusted for each simulation. Thus, the simulator of 4-sided dominoes has as input parameters the total number of desired matches and the set of coefficients α of the second pair, returning the number of wins of this pair.

To implement the optimization by genetic algorithm, we use the Genetic Algorithm toolbox of *MATLAB*, the GAtool. A function was created in MATLAB (dfitness.m) responsible for calculating the fitness of each tested chromosome, by adjusting the input parameters of the domino simulator in Java. Besides the fitness function (or *fitness*), the population type was defined as a *Double Vector*, receiving values in the range of -10 to 10.

In the selection process, we used the Roulette method, which simulates a roulette wheel with the area corresponding to each chromosome proportional to its fitness value; in other words, the probability of a chromosome being selected is directly proportional to its area on the roulette wheel.

For the configuration of the reproduction process, the parameter *Elite count*, which indicates how many individuals with the best fitness values will be transmitted directly to the next generation without going through the reproduction, receives values corresponding to 10% of the population size. We used two crossover operators (or crossover): *Two-point* and *Scattered* (also known as *Uniform crossover*).

The purpose of performing various optimizations is to find a combination of the genetic algorithm parameters required to achieve the optimal or near-optimal values for the coefficients α_1, α_2, α_3, α_4, α_5, α_6 and α_7 of the evaluation function, in a way that they allow the pair that uses this function to choose the best move, has a better performance than the pair that only uses the strategy of adding points during the match (coefficient α equal to zero).

The set of parameters defined previously gives us a combination of 64 executions of the genetic algorithm optimization for each strategy, in a total of 256 runs of the genetic algorithm. A summary of all variations of these

parameters is shown in Figure 5:

Population size	Crossover function	Mutation rate	Generations
• 40 • 60 • 80 • 100	• Two-point • Scattered	• 0.1 • 0.5	• 50 • 100 • 150 • 200

Fig. 5. Parameters used for the optimization of genetic algorithm.

V. Results and Discussion

The following results show the optimizations obtained for Strategies 1, 2, 3 and 4 playing against the "basic strategy" game. For Strategy 1, the genetic algorithm performed the optimization of the seven coefficients α. On the other hand, for Strategies 2, 3 and 4, the genetic algorithm performed the optimization of only three coefficients α_i, because the others are set to zero, depending on the type of strategy.

A. *Optimization of the evaluation function of Strategy 1*

The optimization of the evaluation function corresponding to the first strategy was divided into four groups according to the size of the population, receiving the values 40, 60, 80 and 100. For each parameter combination, the amount of wins was registered for pair 2 and the coefficients resulting from the optimization process. Table 1 and Figure 6 show the best results for each optimization group of the evaluation function of Strategy 1 together with the parameters used in the genetic algorithm during the optimization process.

According to the simulations, the best results were obtained in the optimizations with a total of 100 generations, regardless of population size. On the other hand, an\ increase in population size led to an improvement of the results.

TABLE 1
Optimization results of the evaluation function of Strategy 1

Population	Generations	Crossover Operator	p_m	Wins (pair 2)
40	100	Two-point	0.5	69.24 %
60	100	Scattered	0.1	69.68 %
80	100	Scattered	0.5	69.76 %
100	100	Two-point	0.1	70.06 %

The best results obtained by pair 2, which uses Strategy 1, are displayed with the parameters used in Genetic Algorithm.

The first group of the optimizations corresponds to the population of 40 chromosomes. In this group, a total of 5.000 matches were held: pair 2 won 3,462 matches, which represents 69.24% of wins. The second group corresponds to the combinations with population size equals to 60. In this group, the best result is 3,484 winnings in 5,000 matches, which corresponds to a 69.68% yield. In the third group of optimizations for the population of 80 chromosomes, the best fitness value achieved was 3,488 winnings (or 69.76%) for the pair that used the evaluation function with the parameters optimized by the genetic algorithm. The last group of optimizations of the evaluation function performed tests for a population of 100 chromosomes. The best result is 3,503 winnings in 5,000 matches, which corresponds to a 70.06% yield for the pair that used the evaluation function to choose the best move. This was also the best overall result obtained for Strategy 1 and the coefficients α_1, α_2, α_3, α_4, α_5, α_6 and α_7 of the evaluation function are shown in Table 5, in the end of the section.

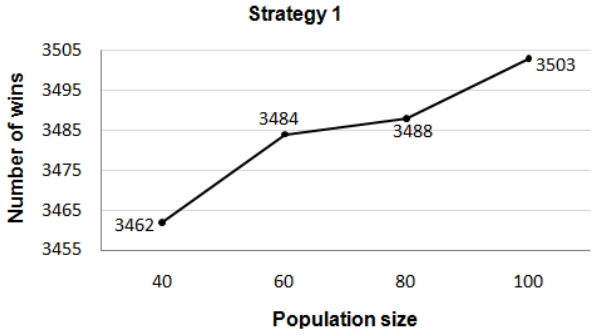

Fig. 6. Overall results of the optimization of the evaluation function of Strategy 1

B. *Optimization of the evaluation function of Strategy 2*

To optimize the coefficients of the evaluation function of Strategy 2, the values for a α_1, α_2, α_3 and α_7 were defined as 0. Once more, the optimization of the evaluation function was divided into groups according to the population size.

Table 2 and Figure 7 show the best results obtained in the optimization of the evaluation function of Strategy 2. We can see that this strategy, where the player focuses only on his own game, the results were worse than those in Strategy 1. The best overall result was 3,355 victories in 5,000 games, which represents 67.1% success for the pair that used the evaluation function to choose the best move. The coefficients α_4, α_5 and α_6 optimized in the evaluation function are shown in Table 5, at the end of this section.

TABLE 2
Optimization results of the evaluation function of Strategy 2

Population	Generations	Crossover Operator	p_m	Wins (pair 2)
40	150	Two-point	0.5	66.22 %
60	150	Scattered	0.5	67.10 %
80	100	Scattered	0.1	66.30 %
100	150	Two-point	0.1	66.46 %

The best results obtained by pair 2, which uses Strategy 2, displayed along with parameters used in the Genetic Algorithm.

C. *Optimization of the evaluation function of Strategy 3*

To optimize the coefficients of the evaluation function of Strategy 3, the values for α_1, α_2, α_3 and α_5 were defined as 0. As in the previous optimizations, the results were divided into groups according to the population size. Table 3 shows

the best results for each group of the optimization of the evaluation of Strategy 3 and the parameters used in the Genetic Algorithm.

Fig. 7. Overall results of the optimization of the evaluation function of Strategy 2

TABLE 3
Optimization results of the evaluation function of Strategy 3

Population	Generations	Crossover Operator	p_m	Wins (pair 2)
40	50	Two-point	0.5	62.38 %
60	150	Scattered	0.1	63.08 %
80	150	Two-point	0.1	63.20 %
100	150	Scattered	0.5	62.90 %

The best results obtained by pair 2, which uses Strategy 3, displayed along with parameters used in the Genetic Algorithm.

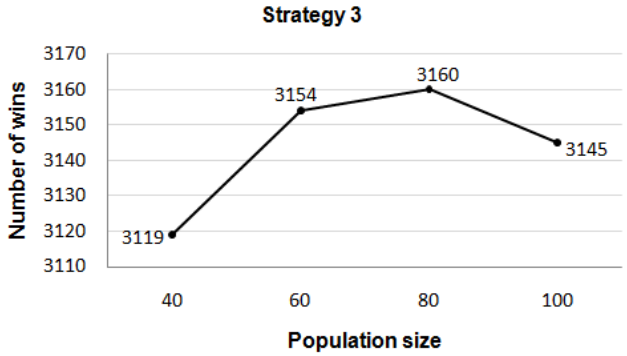

Fig. 7. Overall results of the optimization of the evaluation function of Strategy 3

D. *Optimization of the evaluation function of Strategy 4*

In Strategy 4, the player wants only to impede the moves of his opponents, so the values for the α_4, α_5, α_6 and α_7 were defined as 0. Table 4 and Figure 9 present the best results for each group of the optimization of the evaluation of Strategy 4 and the parameters of the genetic algorithm.

TABLE 4
Optimization results of the evaluation function of Strategy 4

Population	Generations	Crossover Operator	p_m	Wins (pair 2)
40	150	Two-point	0.1	60.98 %
60	200	Scattered	0.1	60.82 %
80	150	Scattered	0.1	61.04 %
100	150	Scattered	0.5	60.98 %

The best results obtained by pair 2, which uses Strategy 4, are displayed along with the parameters used in the Genetic Algorithm.

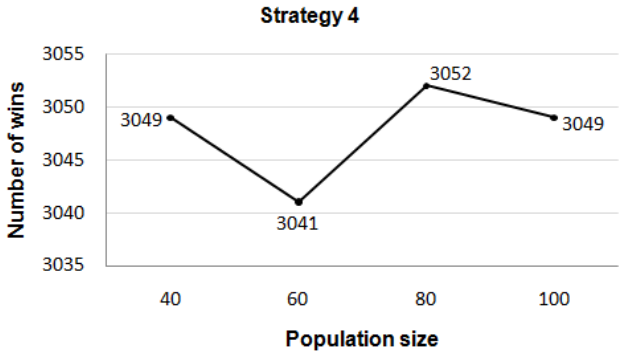

Fig. 8. Overall results of the optimization of the evaluation function of Strategy 4

Observing the above results, we note that Strategy 4 showed the worst results among all tested strategies, whereas the best result was the 61.04% of wins of pair 2, or 3,052 victories in 5,000 matches. The optimized coefficients α_1, α_2 and α_3 in the evaluation function are shown in Table 5, as well as the optimized coefficients for the other strategies.

All of the tested strategies showed significant results compared to the basic strategy, used by beginner players. However, the evaluation function proposed initially represented by Strategy 1, achieved the best results because it combines the three strategies in its equation.

The best result obtained in this study, 70.06% yield, was also higher than the results obtained in [2] and [9]. In [2], the same intelligent agent was used to choose the best move, but the definition of the coefficients α_i was done manually. His best result was 66% wins. In [9], where the research was based on 2-sided dominoes, an intelligent agent was also developed to choose the moves. The agents conducted the decision of the move by inferences and assumptions based on previous moves of other players. However, the stored data was only related to the numbers the players played. Seven different strategies were implemented, including a traditional strategy based on the strategy adopted by real players. However, to validate the intelligent agent, only one of the two players varied his strategy in each simulation; the others played with the traditional strategy. The best result obtained in this study was 54% wins and is not considered a significant result because it was within the tolerance limits set by the author due to the random characteristics of the game.

TABLE 5
Optimized coefficients for Strategies 1, 2, 3 and 4

Est.	α_1	α_2	α_3	α_4	α_5	α_6	α_7
1	-6.63	-3.828	-3.858	-4.694	0.543	6.052	2.419
2	0	0	0	-9.417	2.262	7.065	0
3	0	0	0	1.18	0	7.265	5.811
4	-3.733	-3.233	-4.912	0	0	0	0

VI. CONCLUSIONS

This work presented the results of the coefficients optimization process of the evaluation function to choose the best move in 4-sided dominoes. To adjust the coefficients, we used the technique of searching and optimization of Genetic Algorithms applied to find the best combination that allowed the largest possible number of wins. Four different strategies were evaluated and to obtain the best combination of coefficients of the evaluation functions, the genetic algorithm was run several times with different settings. The optimization results of the evaluation function were superior to the results obtained in previous work, where 66% was obtained. Among the tested strategies, Strategy 1, which is a combination of the other three strategies, had the best results. Given these results, we conclude that the performance of the genetic algorithm was excellent, being superior to the others. Analyzing and comparing our results with results from other studies, ours proved to be more effective in the choice of the best moves in 4-sided dominoes. It is believed that the methodology applied to choose the best move can also be applied to other games where players have imperfect information.

REFERENCES

[1] K. P. Sycara, "Multiagent systems". AI Magazine, 1998, v. 19, n. 2, pp. 79–92.

[2] N. S. Antonio, C. F. F. Costa Filho and M. G. F. Costa, "Proposta de uma heurística para o jogo de domino de 4 pontas," in *Proc. VII Brazilian Symposium on Computer Games and Digital Entertainment – Computing Track*, Belo Horizonte, 2008, pp. 24–30.

[3] C. E. Shannon, "Programming a computer for playing chess". Philosophical Magazine, 1950, 41(4), pp. 256–275.

[4] M. S. Campbell, A. J. Hoane, F. H. Hus, "Deep Blue," Artificial Intelligence, 2002, 134(1-2), pp. 57–83.

[5] J. Shaeffer, *One Jump Ahead: Challenging Human Supremacy in Checkers*. Berlin: Springer-Verlag, 1997.

[6] S. J. J. Smith, D. S. Nau, T. A. Throop, "Success in spades: Using AI planning techniques to win the world championship of computer bridge," *in Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, Wisconsin, AAAI Press, 1998, pp. 1079–1086.

[7] B. S. Chlebus, "Domino-tiling games", *Journal of Computer and System Sciences,* v. 32, n. 3, 1986, pp. 374–392.

[8] H. C. Yen, "A multiparameter analysis of domino tiling with an application to concurrent systems". Theoretical Computer Science, 2002, v. 98, n. 2, pp. 263–287.

[9] A. G. de S. Garza, "Evaluating Individual Player Strategies in a Collaborative Incomplete-Information Agent-Based Game Playing Environment," in *IEEE Symposium on Computational Intelligence and Games*, 2006, pp. 211–216.

[10] D. Ashlock, *Evolutionary Computation for Modeling and Optimization*, Springer, 2006.