International Journal of Innovative Computing, Information and Control
Volume **8**, Number **1(A)**, January **2012**

ICIC International ©2012 ISSN 1349-4198
pp. 13–32

MICHIGAN VERSUS PITTSBURG APPROACH: A COMPARISON FOR MARKET SELECTION PROBLEM

Ata-Ul-Waheed and A. R. Baig

Department of Computer Science
National University of Computer and Emerging Sciences
A.K. Brohi Road, H-11/4, Islamabad, Pakistan
ataul.waheed@gmail.com; rauf.baig@nu.edu.pk

Received June 2010; revised October 2010

Abstract. *This paper analyses the application of learning classifier systems on a market selection game. The prime objective of an agent in this game is to strive for a better profit on its products when sold in the market. A judicious vision is required on the part of agents while choosing a market to keep in view transportation cost as well as the profit margins on the sale of their products in that market. If a large number of similar products are brought to that market, these will naturally yield low price. In the same way, high transportation cost will also result in decrease of profit margin. Hence, a balance between the two is to be maintained in order to gain high profits for an agent's products. We have proposed a new Michigan-Style algorithm for solving this problem. The proposed algorithm is then compared with Pittsburg classifier system approach to solve the same problem. The former works without a central coordinator, whereas the later works with the help of a central coordinator. Our experiments show that even though Michigan approach works with limited localized information, yet it produces high quality results quickly, consistently and at the same time all this is achieved by using less computational resources. We have also compared Michigan approach with the approach of unplanned coordination as proposed by Ishibuchi et al. in 2001. It also shows that the idea of using Michigan classifier system better suits the agents to solve this problem.*

Keywords: Michigan classifier systems, Pittsburg classifier systems, Evolution of game strategies, Evolutionary algorithm, Agent market selection

1. **Introduction.** In the field of computer science, researchers often face computational problems having multiple possible solutions. It sometimes becomes computationally non-feasible to find the optimal or near optimal solution. Computer scientists have come up with various approaches and techniques for investigating such problems in depth with a view to find a solution to the problem within the constraints of computational resources as well as within feasible amount of time. One set of these approaches draws its inspiration from the principles of natural evolution. Such approaches are grouped together by scientific community as Evolutionary Computation (EC) which is in fact treated as a subfield of artificial intelligence. Over a period of time, EC based techniques have been successfully applied in solving the real-world problems. These evolutionary algorithms perform quite well in solving multi-objective optimization problems. In contrast to mathematical programming technique, a single run of these algorithms provides a collection of possible members of Pareto optimal set. The concept of population in these algorithms corresponds with the possible solutions of a problem. Mathematical programming techniques, on the other hand, require a series of executions to get a set of possible solutions [1]. Evolutionary algorithms have been applied to various fields such as data mining, combinatorial optimization, fault diagnosis, classification, clustering, scheduling,

routing, transportation and time series approximation [2-5,30]. In addition, it has also been applied in the field of Game development for solving various gaming problems [6-11].

Evolutionary Computing paradigm consists of many evolutionary algorithms including Genetic Algorithm (GA). GA has been applied to complex function optimization problems where the target objective function does not exhibit nice properties such as continuity, differentiability, satisfaction of the Lipschitz Condition [14-17]. These algorithms are biologically inspired and follow the Natural law "The Survival of the Fittest". The fittest individual of a population is given a better chance to reproduce and survive to the next generation which results in improved solutions with the evolution of successive generations. This helps the algorithm in exploiting the promising areas of solution space. In order to keep the algorithm's exploring capabilities intact, the inferior individuals are also given some chance of survival and reproduction. This ensures that the algorithm is not trapped into local optima. Since these algorithms maintain several possible solutions in parallel, they exploit the solution space, in search of promising areas, in a better way. Genetic algorithms have been used in the implementation of Learning Classifier algorithms successfully [18]. These algorithms have also been applied in linear and nonlinear optimization problems to find out an exact or at least an approximate solution to these problems.

Learning classifier systems provide algorithms, which are capable of autonomously extracting useful rules about a problem space using evolutionary computation and reinforcement learning techniques. It is a well-established method that breaks down into two main branches.

The first one is the Michigan-approach that operates at the level of individual rules [19,20]. In this approach, an individual is coded as a string of if-then rule and its performance is considered as the fitness value. Instead of the fitness value of a rule-set as a whole, an individual's fitness value is used in a genetic search for finding high quality rules regarding a chosen problem domain. This approach uses both less memory storage and less computation time, in comparison to its counterpart, as it maintains a population of single rule-set only. The generation update only replaces bad rules of the current population with newly generated rules. Due to elitism, good rules of current population are inherited by next generation without any modification. In this way, performance of the algorithm keeps improving with each new generation, as good rules are retained and bad rules are replaced [21]. Some good examples of this approach can be found in [20,22].

The second one is Pittsburgh-approach that operates at the level of whole rule-sets [23,24]. In this approach, a rule-set is treated as an individual and the performance of the rule-set is used as a fitness value. An individual, in this case, corresponds to a number of if-then rules coded as strings. Thus, an evolutionary search is performed for finding rule-sets with high fitness values. Some of the good rule-sets are treated as elite and are inherited by the next generation without any modification. As the performance of each rule is not explicitly evaluated, good rules in a poor rule-set may get eliminated as a result of generation updates. Due to the involvement of a number of rule-sets in a population, this approach requires a large memory storage and a long computation time, as compared with Michigan approach [21]. For further details and good examples of Pittsburgh approach, one may refer to [25-28].

In this paper, we have proposed a novel evolutionary algorithm for Market Selection game and have presented its comparison with other techniques. The market selection game was formulated in [12] and was further investigated in [8,13]. It is a multi-agent game, in which each agent has to sell its products at a market yielding maximum profit. Many products brought to the same market result in low profits. Hence, each agent has to pick the market having least number of products. Previously, the attempts have been made to propose Fuzzy Q-Learning [12] and Genetic Algorithm inspired approaches [8] to solve this problem. To the best of our knowledge, no attempt has been made to apply the concept of Michigan style learning classifier system to address this problem. In the light of above facts, we intend to make the following major contribution in this paper.

- Propose an efficient and effective Michigan-style novel technique for solving this problem that has its inspiration from Ishibuchi method.
- Prove that even by providing very limited local information and a memory function to each agent results in better solutions.
- Compare the effectiveness of our technique with that of Ishibuchi [8].
- Compare Michigan and Pittsburg Style learning classifier system for their applicability and suitability to a multi-agent system such as market selection game.

The rest of this paper is structured as follows: In Section 2, we are explaining Market Selection game and its constraints whereas its subsections describe how the various genetic operators are applied to this game. In the next section, we are describing our proposed Michigan-Style algorithm for this problem and in Section 4, we have shown how this problem can be solved by using Pittsburg approach. Section 5 gives the details of experiments conducted in order to compare these two approaches with each other as well as the Ishibuchi Method [8]. Section 6 provides future directions while the last section provides the conclusion of our investigations.

2. Market Selection Problem.

Market Selection problem was for the first time formulated for games in the study titled "Fuzzy Q-learning for a multi-player non-cooperative repeated game" [12]. In this problem, the game strategies of various agents are simultaneously evolved on the basis of the reward the agent receives by selling its products in the market. In this optimization problem, each of the n agents has to choose one market out of m markets to sell its p products so that the maximum profit is made. The problem constraints the decision of all the agents to be made simultaneously. Thus, no agent will know other agent's market selection choice in advance. Figure 1 shows an instance of the problem in which $n = 50$ and $m = 7$.

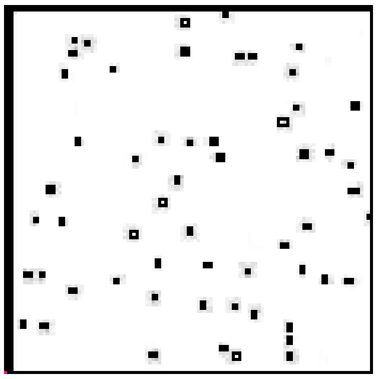


FIGURE 1. Market selection environment

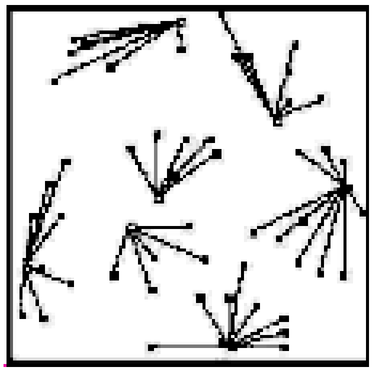


FIGURE 2. Market selection by agents

There are various types of products which can be sold in markets. Each agent may or may not have limited quantities of each type of the products to sell. The products and their quantities may be different for each agent. Each agent can only choose a single market for selling its products.

In Figure 2, we show an example of the market selection where the line from each agent indicates the selected market (i.e., the line corresponds to the flow of the product). The profit received by an agent by selling its product in the market is inversely proportional to the distance of the agent from the selected market and the number of products brought to the same market by all other agents. In case, the agent is located at a long distance from the market, the higher cost of transport to bring the product to the market will obviously decrease its profit margin. Similarly, if a market receives large quantities of similar type of products, the price of the product at that market will also fall. Hence, the choice of the market for each agent has to be optimized, keeping in mind the product price at that market and the transportation cost involved to bring that product to the market.

2.1. **Chromosome encoding.** Individuals of a population are represented by a chromosome. Each chromosome consists of genes. Each gene is encoded with the help of alphabets which may include binary, numeric, real valued or symbols. Genetic operators such as selection, mutation and cross over are applied to the population of chromosomes to create future generations. In our market selection game numeric encoding is used. Since we will be using 5 markets in our experiments, in our case, each chromosome consists of a single gene with the values ranging from 1 to 5 corresponding to 1st, 2nd, 3rd, 4th and 5th market respectively.

2.2. **Fitness function.** There are n agents and m markets. There are p types of products which can be sold in the markets. Each agent i has quantities q_{ip} of the products to sell. The products and their quantities may be different for each agent. Each agent can choose a single market for selling its products.

The quantity of a product p available in a market j is

$$Q_{pj} = \sum_{i=1}^{n} x_{ij} q_{ip} \tag{1}$$

where $x_{ij} = 1$ if agent i has selected the j^{th} market, otherwise it is zero and q_{ip} is the quantities of product p brought to the market by agent i. The market price MP of the product p at each market in each round is inversely proportional to the quantity of that product Q_{pj} available at that market.

$$MP_{pj} = a_j - b_j(Q_{pj}) \tag{2}$$

where a_j and b_j are positive constants that influence the market price. The total price that an agent i (AP_i) gets for its products, is calculated according to the summation of the market prices of the products at that market weighted by the quantity of that product available with the agent (Equation (3)).

$$AP_i = \sum_{t=1}^{p} q_{it} MP_{tj} \tag{3}$$

The transportation cost of the product from the agent's location to the selected market has a negative effect on the agent's payoff. The transportation cost for an agent i to its selected market j is defined as:

$$TC_{ij} = d_{ij} \tag{4}$$

where d_{ij} is the Euclidean distance between the agent and the market location and can be calculated as following:

$$d_{ij} = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} \tag{5}$$

where X_iY_i and X_jY_j are coordinates of an agent and market respectively.

For the purpose of simplicity, we assume that the transportation cost per unit distance is the same for all agents. Hence, transportation cost (TC) is directly proportional to the distance of agent from the selected market. The payoff/reward (r) of an agent i is

$$r_i = AP_i - TC_{ij} \tag{6}$$

Even though each agent is trying to maximize its own payoff in each round, still the payoff depends upon the market price, which, consequently, depends upon the market selection made by other agents, and hence, the payoff of an agent is dependent on the action of other agents.

Our aim in the market selection game is to look for the optimum collective payoff in minimum rounds of game.

$$R = \sum_{i=1}^{n} r_i \tag{7}$$

In other words, we have to maximize R.

Even though each agent is trying to maximize its own payoff in each round and does so on the basis of local information, yet as a whole the system reaches a sort of equilibrium.

2.3. **Selection probability.** Selection in GA is performed on the basis of the fitness of an individual in order to produce successive generations. The selection is made probabilistically so that the better individuals have an increased chance of being selected and the possibility of selection of weak individuals is also not ruled out. Possibility of selection of weak individuals provides exploration capabilities to the GA. There are several schemes for the selection process, e.g., roulette wheel selection and its variations, tournament, Ranking and linear scaling [15,17].

In our approach, we have used Roulette wheel selection [16] in which each neighbor of an individual is assigned a selection probability using the following formula as used in [8], i.e.,

$$P(s_k^t) = \frac{r_k^t\left(s_k^t\right) - r_{\min}^t\left(N\left(i\right)\right)}{\sum\limits_{k \in N(i)} \left\{r_k^t\left(s_k^t\right) - r_{\min}^t\left(N\left(i\right)\right)\right\}} \tag{8}$$

where $N(i)$ represents all the neighbors of agent i and s_k is the strategy of neighbor k ($k \in N(i)$). $r_{\min}(N(i))$ represents the minimum payoff among the neighbors $N(i)$.

2.4. **Replacement probability.** Each agent's strategy is replaced with the help of the user defined replacement probability parameter. For replacing the strategy of an agent, the new strategy is selected, from one of its neighbors, based upon the selection probability as explained in the previous section. While selecting the new strategy from the neighbors, the current strategy of the agent is also given a chance for reselection. In our implementation, each agent is considered to be its own neighbor; for example, if we keep neighborhood size ($NS(i)) = 4$, it means the i^{th} agent and its three neighboring agents based upon the minimum Euclidian distance from the i^{th} agent. Thus, in some situations, an agent may retain its previous strategy.

2.5. **Mutation.** Mutation operator is applied after applying Replacement procedure to the strategy of each agent. It is applied to the fired rule with a user defined mutation probability P_m. This operator changes the existing strategy of an agent with a different strategy at random (i.e., out of 5 markets values). Both these operations are independent of each other. Thus, both or none of them may be applied to an agent.

2.6. **Stopping criteria.** The genetic algorithm keeps on selecting and reproducing parents for a generation until the termination criteria is met. In our approach, we have used the specific number of generations as stopping criteria for the algorithm. Other possibilities could be to terminate the algorithm when improvement in the best solution is not seen for a specific number of generations or the algorithm achieves an acceptable target value of the evaluation measure.

3. **Application of Michigan Approach.** In Michigan approach, each agent is autonomous in deciding its own strategy based on the localized information available to it. It maintains its own *Rule-Set* consisting of the history of its past decisions and the eventual reward received. The structure of a single rule is shown in Figure 3. Neighbors of an agent are selected during initialization of the algorithm. Since agents in our scenario are not mobile, the neighbors selected initially for an agent remain the same throughout the execution of the algorithm. For selecting neighbors of an agent a method of minimum Euclidean Distance is used. Rule (or chromosome) for each agent is created using the following structure.

$\boldsymbol{N1}$	$\boldsymbol{N2}$	$\boldsymbol{N3}$	**Action**	**Strength**
S_{N1}	S_{N2}	S_{N3}	S_{Agent}	$\sum$PayOff/No. of times this rule is selected

FIGURE 3. Structure of a rule with 3 neighbors

The figure above shows the structure of a rule with three neighbors. In this structure, $N1$ to $N3$ are three nearest neighbors of the agent (excluding the agent). S_{N1} is the strategy of neighbor 1 (i.e., market selected by neighbor 1), S_{N2} is the strategy of neighbor 2 and S_{N3} is the strategy of neighbor 3. S_{Agent} represents the strategy used by the agent (action taken) if the first three conditions are fulfilled. Each chromosome (or rule) has an associated reward/payoff. Initially the Strength is set equal to the payoff. The possible values of all genes are integers between 1 and 5 (because there are 5 markets).

Algorithm

Step 0: Each agent selects a market (strategy) at random.
Iterate the following steps until a termination criterion is fulfilled:

Step 1: The market selection information is used to construct a new rule in the Rule-Set of each agent.

Step 2: The payoff (reward) of each agent is calculated as per (6). The reward is used to update the strength associated with the rule.

Step 3: An Evolutionary Algorithm comprising the following steps is run in each agent's Rule-Set.

- (a) Make three copies of the rule just executed.
- (b) Mutate the consequent (action) part of the first copy by replacing the current strategy of the agent with the strategy of its neighbor 1.
- (c) Similarly mutate the consequent (action) part of the second and third copies by replacing the current strategies with the strategy of neighbor 2 and 3, respectively.
- (d) The rewards of these three rules will be the rewards obtained by the neighbors $N1$, $N2$ and $N3$, respectively.
- (e) If any two (or all three) of the new rules are the copies of each other, they are merged into one rule. In such cases, the reward of the rule created by merging is the average of the merged rules.

(f) Check to see if any of the remaining (after merging) newly created rules is an exact copy of the parent rule. In such cases, the new rule is discarded and the strength of the matching rule of Rule-Set is updated. In case, it does not match with existing rule(s) of Rule-Set, it is added to Rule-Set as it is.

(g) If the size of Rule-Set exceeds the total number of permissible rules, the rules with the worst rewards are culled. In rule pruning, the rules added in the Rule-Set during the current iteration are exempted from pruning with a view to give them a chance to evolve themselves.

Step 4: Depending upon the replacement probability, a rule is fired from each Rule-Set using the roulette wheel selection. The probability of a rule being fired is proportional to its reward. The consequent part of the rule becomes the new strategy of the agent.

Step 5: The strategy of the agent obtained in Step 4 is mutated in the light of user defined mutation probability parameter.

A new iteration starts.
After the first iteration, we may face the following situations.

(a) In Step 1, it may happen that the rule created from the current market selections does already exist in the Rule-Set. In such cases, the strength of the old rule is updated according to the new reward received.
(b) In Step 3(f), the check for avoiding duplicates is not only limited to the parent rules but it also applies to all other old rules contained in an agent's Rule-Set.

In Step 3(e)-3(f), the rule similarity is checked by comparing the markets selected by one agent & its neighbors with the corresponding markets selected by other agent & its neighbors; for example,

$$\begin{array}{l} S[i]_{Agent} = S[j]_{Agent} \quad \text{AND} \quad S[i]_{N1} = S[j]_{N1} \quad \text{AND} \\ S[i]_{N2} = S[j]_{N2} \quad \text{AND} \quad S[i]_{N3} = S[j]_{N3} \end{array} \tag{9}$$

where i and j represent ith and jth agent respectively.

In Step 3(f), the Strength of existing rule is updated by taking average of the two matching rules' strengths, i.e.,

$$\frac{(\text{Existing Rule's Strength} * X) + \text{ Strength of Newly Created Rule}}{X+1} \tag{10}$$

where $X =$ No. of times the rule has been selected in the past.

4. **Application of Pittsburg Approach.**

In this approach, we have to assume that there is a central coordinator which is trying to maximize the overall payoff of the system (i.e., summation of individual agent payoffs). Information from all the markets has to be made available with this central coordinator after each round.

There is a population of competing chromosomes and each chromosome is a complete list of agent-market allocations encoded as a string of length n as $s = s_1 s_2 s_3 \dots s_n$, where s_i denotes the market selected by the ith agent. Since there are 100 agents, we have a chromosome of 100 genes and each gene's value is an integer number in the range of 1 to 5 because there are 5 markets (the value of a gene is the market selection made by the corresponding agent).

The fitness of a chromosome is the summation of the payoffs of the agents according to the allocations defined in the chromosome. The higher the collective payoff, the better is the chromosome. The fitness (objective) function is:

$$f(\text{chromosome}) = \sum_{i=1}^{n} r_i \tag{11}$$

where r_i is the payoff of the ith agent (as given in Equation (6)) according to the market allocations defined in the chromosome. A search is conducted by means of genetic algorithm to find out the optimum chromosome (which has the maximum summed payoff).

In this approach, we have used the roulette wheel selection mechanism with the linear scaling as mentioned in Equation (8). Previously the scaling mechanism was applied to a single agent but, for Pittsburgh approach, it has been adapted to chromosomes as a whole. Standard uniform crossover and the mutation mechanism, as mentioned in Section 2, are applied.

Algorithm

```
Step 0: Generate the initial population of chromosomes by executing the following steps:
        a. Each agent randomly selects a market.
        b. A chromosome, of length equal to the total number of agents, is created based
           upon the market selected by each agent.
        c. The chromosome is added to the current population.
        d. If the population size is less than the user's specified parameter, return to Step
           a.
Execute the following steps to generate next generation.
Step 1: Calculate fitness of each chromosome by using Equation (11).
Step 2: Select two parents using roulette wheel selection with the linear scaling as in (8).
Step 3: Perform uniform cross over based upon the crossover probability.
Step 4: Mutate the newly generated chromosome based upon the mutation probability.
Step 5: For elitism, add the best individual of previous generation to the current generation.
Step 6: If population size is less than the user's specified parameter, return to Step 2.
Step 7: If the total number of generations is less than the user specified allowed number
        of generations, return to Step 1.
```

5. Experiments.

For our experiments, we have assumed that the agents and the markets are located on the intersections of a 100×100 grid. There cannot be more than one agent or market on an intersection.

In order to analyze the difference between Michigan and Pittsburg approach various experiments were performed. Pittsburgh approach is based upon central coordinator hence the best results should be obtained using that approach. We applied both these approaches to Market Selection Game as explained in Section 2. For both of these approaches, the algorithms were executed for 10000 generations. Though the algorithm may reach effective fitness value in much less number of generations, yet we allowed the execution till 10000 generations in order to ensure that both the approaches are given a fair chance of improvement. No further improvements were observed after 10000 generations. It is important to note that the 10000 generations of our algorithms are analogous to 10000 rounds of market selection game. In each round, each agent is given a chance to select a market, for selling its products, based upon the price (i.e., reward) it will get for those products from the market. An agent's payoff is based upon the market price and distance of the agent from that market. This section is divided into two main portions. In the first portion, we have presented the results obtained by applying our proposed Michigan-style algorithm to the market selection game whereas in the second, we have compared this approach with Ishibuchi method [8] as well as the Pittsburgh-style learning classifying system.

5.1. **Michigan approach.** For our experimentation, we used the value for $a_j = 200$ and $b_j = 3$ in market price equation (Equation (2)). The purpose of specifying these values is the ease of comparison. We would like the Michigan approach to be compared with the approach given by Ishibuchi. In his paper, Ishibuchi has used the same values for a_j and b_j. Our proposed approach is mainly based on four user defined parameters. These parameters are Rule Set Size, Neighborhood Size, Mutation Rate and Replacement Probability. In our computer simulations, we performed experiments using various combinations of these parameters. Though the algorithm proved to be robust against a wide range of combinations but the best results were obtained using the following parameters:

Neighborhood Size $= NS = 4$
Probability of Mutation $= P_m = 0.0$
Rule Set Size $= RS = 5$
Replacement Probability $= Pr = 0.2$

All the results presented in this section are based upon these parameter settings unless stated otherwise. For the purpose of comparison, we have picked up the same environment as mentioned in [8] (see Figure 4). The only difference is that, in [8], the author has specified the coordinates in floating points whereas we have rounded off those values to the nearest integers.

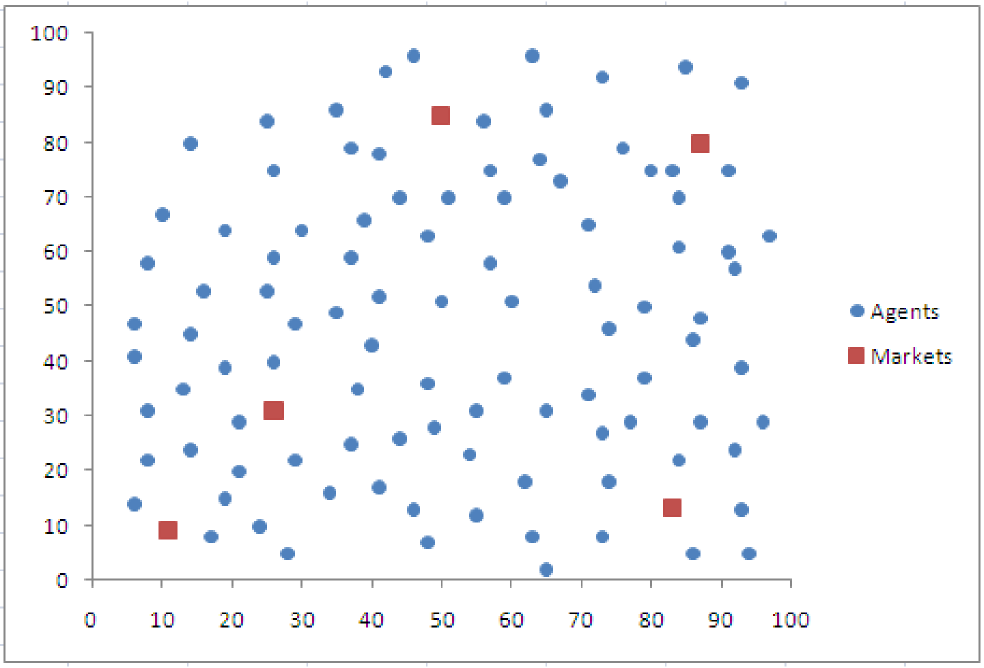


FIGURE 4. Layout of the environment showing 5 markets and 100 agents

The evolution of game strategies using best parameter specification is presented in Figure 5. The figure shows how agents evolve their strategies, gradually, to get higher rewards for selling their products to the nearest less congested market.

Our algorithm uses random initialization and probabilistic approach in search of good solutions. This creates doubt that an experimental run for a particular specification will give different results for each execution. In order to prove the robustness of our approach we executed 10 independent trials for the same experimental setup. The results of each trial averaged over 100 agents are shown in Table 1. From the table, it is evident that the

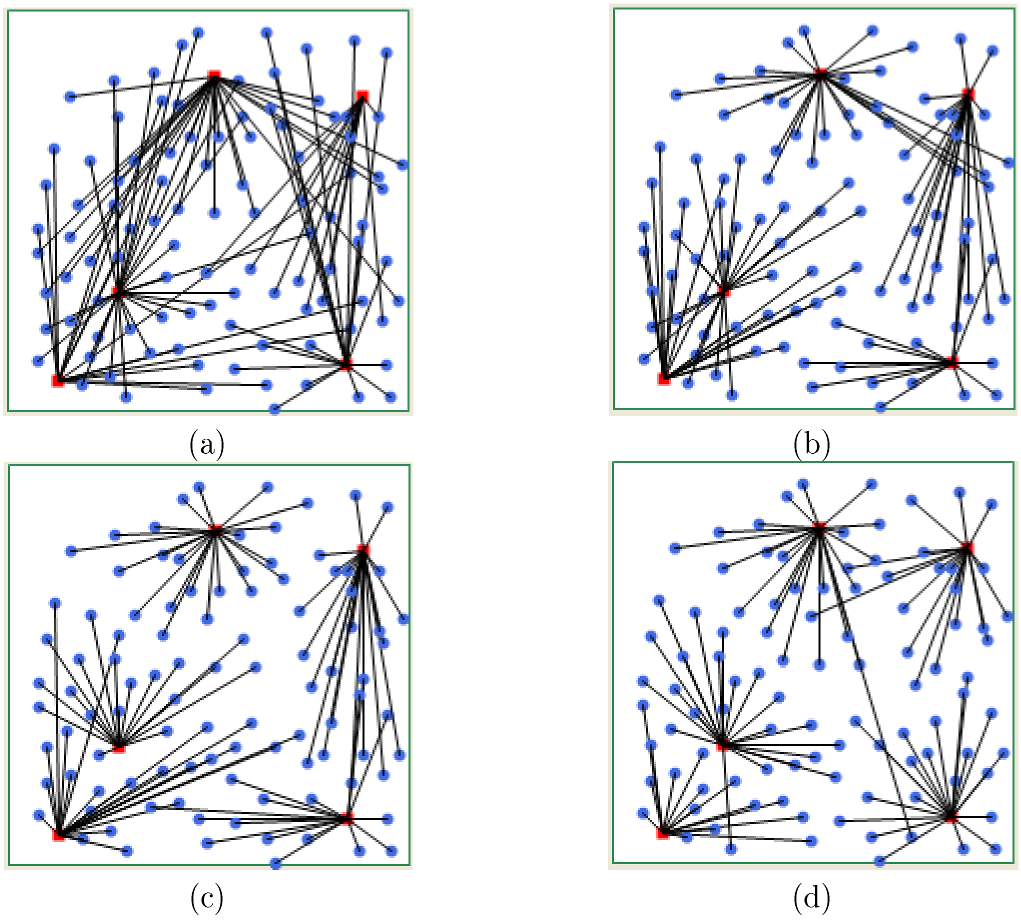


FIGURE 5. Evolutions of market selection during various rounds of game: (a) after 1st round; (b) after 10th round and (c) after 50th round (d) after 500th round

algorithm produces good results consistently. We obtained a payoff of 118.7 or greater in 8 out of the 10 trails performed. This indicates, no matter how agents are initiated with different strategies, during the course of execution they tend to learn from their experience and are able to identify markets that provide higher profits in return to their products.

TABLE 1. Trial-wise average fitness of an agent after 10000 rounds of the game

Trial #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Avg. Fitness	118.6	118.7	118.2	118.7	118.7	118.7	118.9	118.8	118.8	119.0

In order to monitor the evolution of game strategy, we recorded average payoff obtained by an agent in each round of 10 trials. Table 2 summarizes the average fitness obtained by an agent, averaged over 10 independent trials, during various rounds of the game. From this table, it can be seen that the game strategy evolved rapidly during the first 50 rounds and after that minor but steady improvement was observed till the 10000 rounds of the game. Since an agent makes its move depending upon the information received from a limited number of its neighbors, the decision made by an agent in a particular round, based on partial information, sometimes results in a poor payoff as compared with a previous round. This fact can be observed by looking into Table 2 for 1000$^{\text{th}}$ and 5000$^{\text{th}}$ round where a drop in average payoff is visible.

Table 2. Round wise avg. fitness values

Round No.	**Avg. Fitness**
5	99.86
10	107.17
20	113.38
30	116.39
40	117.75
50	118.27
100	118.48
500	118.50
700	118.51
1000	118.36
3000	118.53
5000	118.46
7000	118.62
9000	118.66
10000	118.73

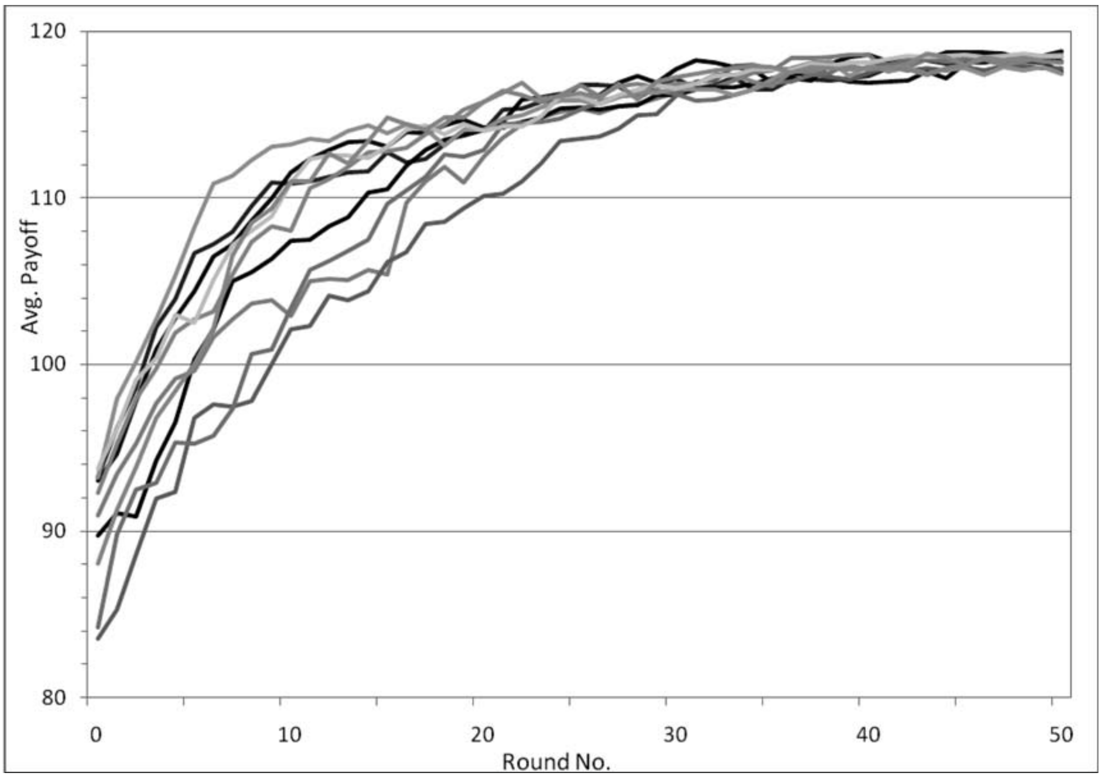


Figure 6. Evolution of average payoff during first 50 rounds of 10 trials

The evolution of game strategies during 10 independent trials is shown in Figure 6. The results show that good solutions are evolved in a very similar fashion during each trial. All the trials start to produce higher average payoffs at the 50th round. Each agent makes its move depending upon the local information available to it. In the absence of global information, agents may make moves which result in sub-optimal solutions. This can result in un-predictable behavior if algorithm is allowed to execute beyond 50 rounds. To analyze this, we have plotted the average payoff for the 10 trials in the next 50 rounds in Figure 7. As shown in this figure, once agents are able to figure out best strategies for themselves, they continue to exercise these strategies. The probability of an agent making a drastically bad move is avoided by the Rule Set memory provided to each agent. The rule set contains the good decisions taken by the agent in the past in order to get high payoff.

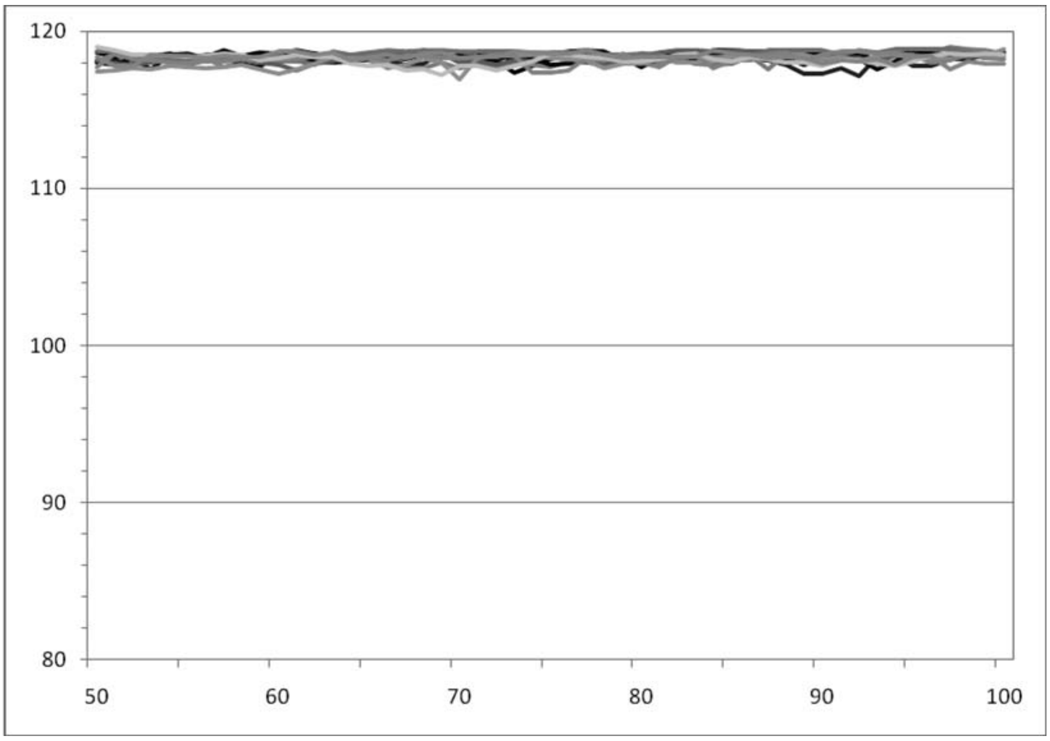


FIGURE 7. Evolution of average payoff (for the 10 trials) from 50th to 100th round

In this section, we have seen how our proposed algorithm solves the market selection problem effectively. In the next section, we would like to investigate its performance in comparison to other alternatives for solving the same problem.

5.2. **Comparison of different approaches.** In this section, we compare Michigan approach with two other methods. The first is based on unplanned coordination mentioned in [8]. We will refer to it as Ishibuchi Method ($\mathrm{M}_{\mathrm{Ishibuchi}}$). The second is based on Pittsburg approach that uses Genetic Algorithm with a concept of central coordinator. We will refer to it as Pittsburg Method ($\mathrm{M}_{\mathrm{Pittsburg}}$). Throughout this section, Michigan based methodology is being referred to as $\mathrm{M}_{\mathrm{Michigan}}$.

5.2.1. *Ishibuchi method ($M_{\mathrm{Ishibuchi}}$).* This method uses unplanned coordination for the evolution of agent's game strategy using Genetic Algorithm based approach. Agents share their strategy with the neighboring agents through localized selection based upon replacement probability [8]. For the purpose of comparison, we performed 10 independent trials of the algorithm based upon the best parameter specifications (i.e., *Neighborhood Size* = 4*, Replacement Probability* = 0.5 *and Mutation Probability* = 0). In each trial, agents were randomly initialized with different strategies. The results of the 10 trials are given in Table 3 below.

By comparing Table 3 ($\mathrm{M}_{\mathrm{Ishibuchi}}$) with Table 1 ($\mathrm{M}_{\mathrm{Michigan}}$), it is evident that the $\mathrm{M}_{\mathrm{Michigan}}$ performs well as compared with $\mathrm{M}_{\mathrm{Ishibuchi}}$. $\mathrm{M}_{\mathrm{Ishibuchi}}$ has achieved an average of 118.2 for the 10 trials in comparison with an average of 118.7 using Michigan approach.

Table 3. Trial-wise average fitness of an agent after 10000 rounds of the game using Ishibuchi method

Trial #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Avg. Fitness	118.0	118.2	118.7	118.5	118.6	118.3	117.9	118.6	117.8	117.8

The results of Ishibuchi method range from 117.8 to 118.7 whereas that of Michigan approach range from 118.2 to 119.01. This shows that our proposed Michigan approach consistently provides better outcome as compared with $M_{\text{Ishibuchi}}$. Michigan approach provided an average payoff 118.7 or greater in 8 out of 10 trials whereas Ishibuchi method reached an average payoff of 118.7 only once during the 10 independent trials.

It is not just the final outcome of the algorithm that matters. How quickly it starts producing good results, is also an important factor. In order to analyze the behavior of algorithm in early rounds of the game, we noted the round of the game in which the particular approach started producing good results, i.e., higher payoffs. For making these comparisons, we considered the average payoff value of 118 or greater as the high payoff. It revealed that Ishibuchi approach was producing good results a little earlier than the Michigan approach (using best parameters). On further investigation, it turned out that the major contributor in Ishibuchi method for this was replacement probability. Consequently, we used the same value of replacement probability (i.e., Pr = 0.5) for both the approaches. The results for the 10 trials of each method are presented in Table 4. It shows the round number of the game in which each trial reached an average payoff of 118 or greater.

Table 4. Trial wise round number in which results entered into good range

Trial #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Avg.**
$\mathbf{M_{Ishibuchi}}$	17	10	27	11	13	11	15	42	14	20	**18**
$\mathbf{M_{Michigan}}$	9	16	10	15	11	19	17	21	17	12	**15**

By looking at this table, it can be concluded that the Michigan approach converges to better results earlier than the Ishibuchi method. Besides this, Michigan approach is pretty much consistent in its behavior as compared to the $M_{\text{Ishibuchi}}$. It varies from 9 to 21 rounds of game as compared to Ishibuchi's 11 to 42.

If the average payoff for the 10 trials is measured at the end of first 50 rounds of game, both algorithm show very much similar pattern with Michigan approach performing slightly better. In case of $M_{\text{Ishibuchi}}$, the average of 10 trials is 118.2 whereas in case of M_{Michigan}, it is 118.3. In order to understand how the two algorithms evolve good strategies beyond these initial rounds, we executed them for 10000 rounds. The 10 trials average payoffs are plotted in the graph (Figure 8) during various round of the game for $M_{\text{Ishibuchi}}$ and M_{Michigan}.

By comparing the results of two approaches in Figure 8, it becomes clear that the Michigan approach really evolves good strategies if it is allowed to execute 10000 rounds of the game, whereas, the $M_{\text{Ishibuchi}}$ does not take real advantage of these extra rounds beyond the initial 1000. This is why Michigan approach clearly shows trends that are much higher in terms of agent's payoffs than the Ishibuchi method.

We made further comparisons by comparing the two approaches in terms of Maximum, Average and Median payoff obtained over 10000 rounds of each trial. The results given in Table 5, reveal the fact that Michigan approach provides better results in all the three categories.

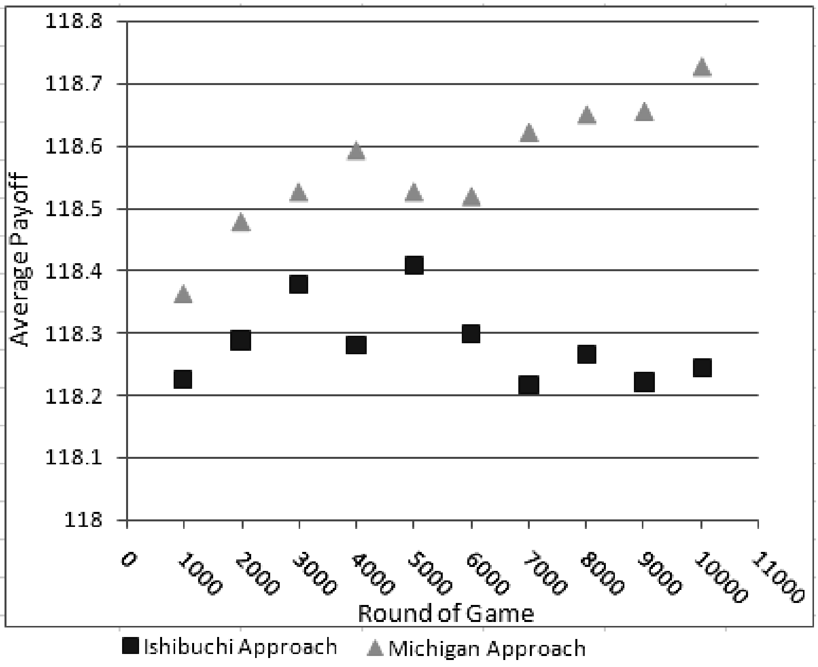


FIGURE 8. Average payoff for the 10 trial at various rounds of the game using Ishibuchi and Michigan methods

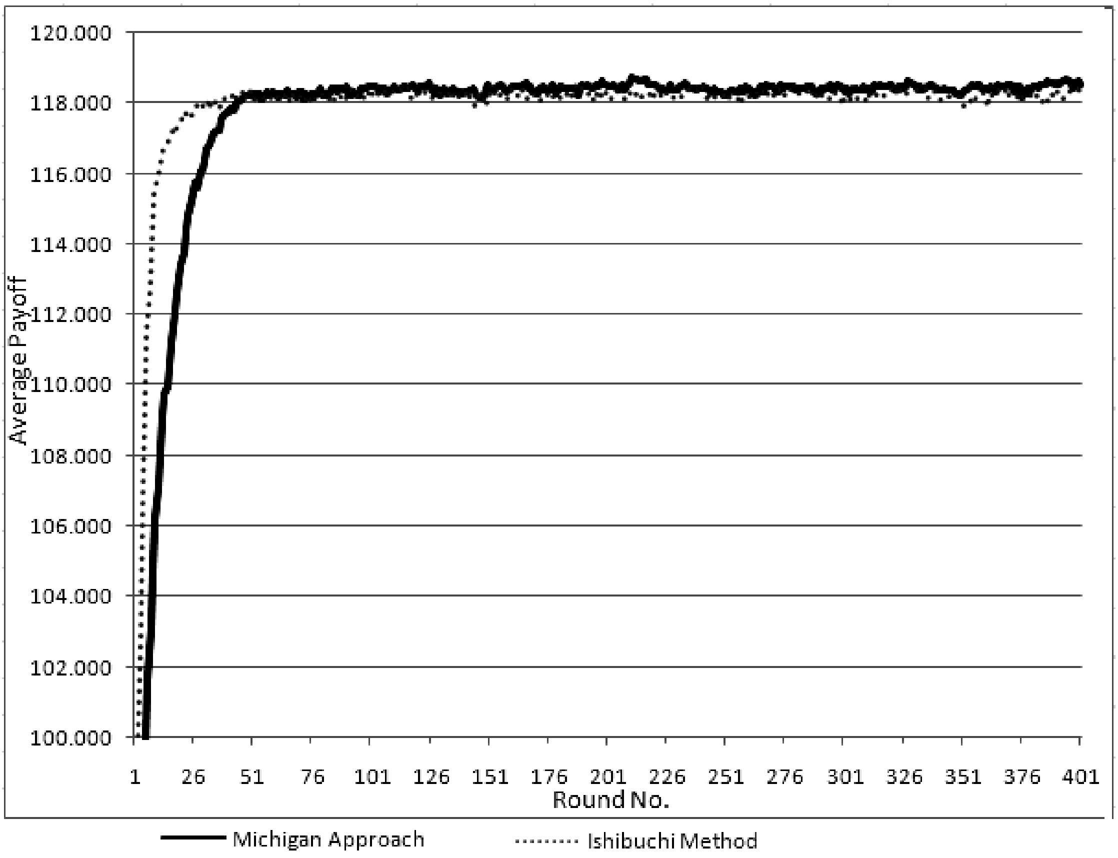


FIGURE 9. Average payoff obtained by the two methods for 10 trials of each approach

TABLE 5. Trial-wise max, avg. and median values based on 100 agent's avg. payoff calculation of each round

Trial #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$\mathbf{M_{Ishibuchi}}$ – **Max**	119.09	119.19	119.03	119.09	119.13	119.11	119.12	119.07	119.09	119.19
$\mathbf{M_{Michigan}}$ – **Max**	119.17	119.15	119.18	119.15	119.17	119.15	119.20	119.18	119.16	119.17
$\mathbf{M_{Ishibuchi}}$ – **Avg.**	118.20	118.22	118.19	118.21	118.23	118.22	118.22	118.19	118.22	118.20
$\mathbf{M_{Michigan}}$ – **Avg.**	118.40	118.40	118.50	118.50	118.50	118.60	118.50	118.60	118.50	118.50
$\mathbf{M_{Ishibuchi}}$ – **Med.**	118.26	118.28	118.25	118.26	118.28	118.28	118.28	118.26	118.28	118.26
$\mathbf{M_{Michigan}}$ – **Med.**	118.52	118.46	118.62	118.52	118.61	118.67	118.55	118.62	118.59	118.59

This fact clearly indicates that the Michigan approach gives us superior results to the Ishibuchi method. However, it would be interesting to analyze the evolution of game strategies in initial rounds of the game by using both these methods. Figure 9 shows the 10 trials' average payoff during first 400 rounds of the game. It is evident from this graph that the improvement of average payoff in case of $\mathrm{M_{Ishibuchi}}$ method is a little quicker than the $\mathrm{M_{Michigan}}$. However, Michigan approach dominates the Ishibuchi method in terms of higher average payoff values after initial 50 rounds of the game. All the analyses presented above also support this argument.

5.2.2. *Pittsburg approach ($M_{\mathrm{Pittsburg}}$).* In order to compare the performance of Michigan approach with a Central Coordinator based Pittsburg algorithm, we performed various experiments. Since Pittsburg approach based algorithm tries to optimize strategies in order to get high overall payoff for all agents (instead of each individual agent), it is quite obvious that it will deliver high quality results. We wanted to analyze Michigan approach by keeping Pittsburg approach as a benchmark. To perform these experiments, we have used the following parameter specifications:
(a) Population size = 100
(b) Crossover probability = 0.8
(c) Mutation probability = 0.001
(d) Maximum iterations = 10,000 generations
(e) $a_i = 200$ and $b_i = 3$
These parameters were used for the 10 trials performed for Pittsburg approach based on different initial strategies. The average payoff of an agent, after 10000 generations, turned out to be 119.265 for all the 10 trials. The evolution of average payoff for these 10 trails is shown in Figure 10. As improvement in average payoff is very similar in 10 trials, it is difficult to distinguish the results from each other. When we compare the trend of Figure 10 with that of Figure 6, we reach the conclusion that both these approaches evolve good results in a very similar fashion.

Figure 11(a) shows the final strategies of agents corresponding to the average payoff of 119.265. It is interesting to note that Michigan approach was also able to provide near optimal results as it was able to achieve an average payoff up to 119.011 (as in Table 1 trial 10). The market selection corresponding to this result is given in Figure 11(b). When one compares Figure 11(a) with Figure 11(b), one can easily conclude that Michigan approach provided almost identical outcome as did its counterpart. The ability of Michigan approach to deliver near optimal results is also proved from the Maximum Average Payoff values given in Table 5. This table shows that each trial of Michigan approach was able to achieve a maximum payoff of 119.2 approximately. In case of Michigan approach, even the average payoff 118.7 obtained over all the 10 trial is quite effective considering the less amount of resources (in terms of CPU and memory) requirement of this approach over Pittsburg approach.

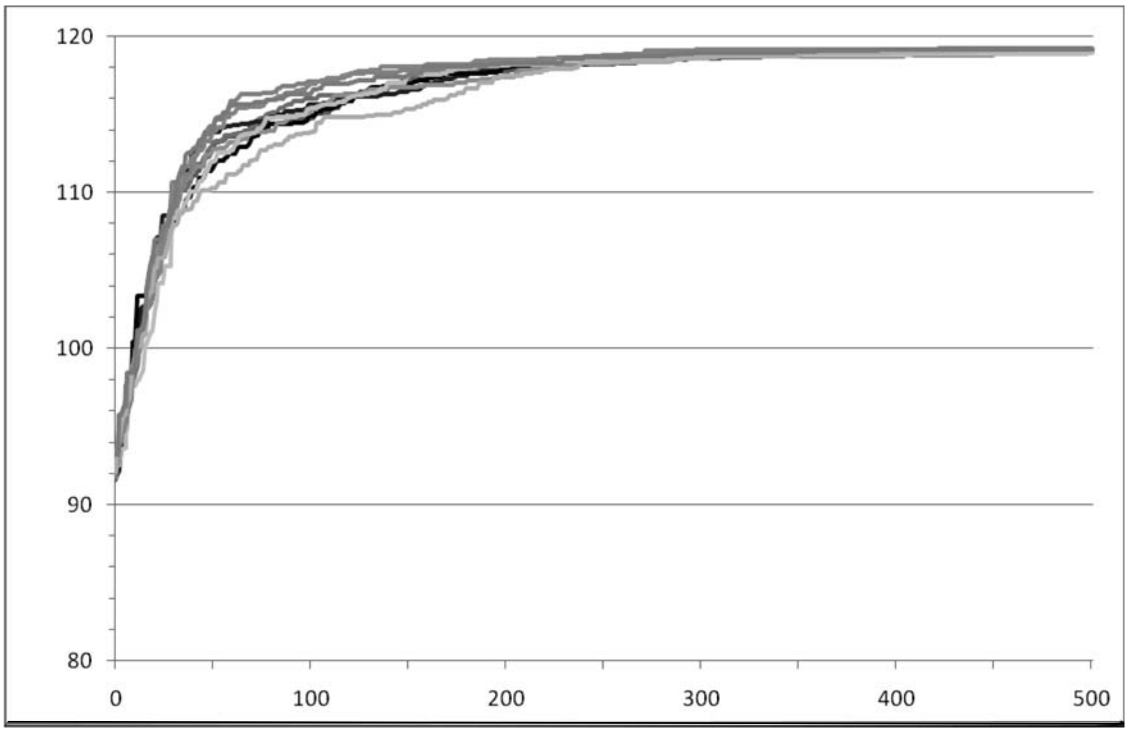
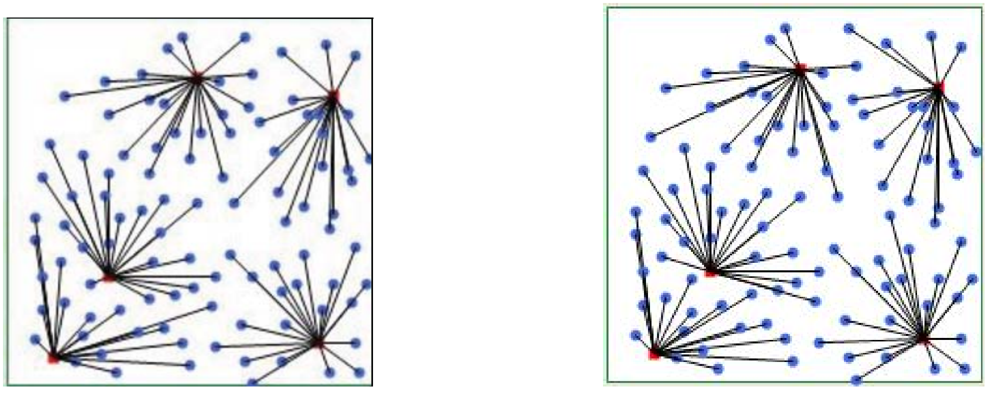


FIGURE 10. Evolution of average payoff using Pittsburg approach during first 500 rounds of the game



(a) Pittsburg approach (b) Michigan approach

FIGURE 11. Market selection after 10000 rounds of the game

Figure 12 shows the comparison of two approaches after first 50 rounds of the game. It is very clear from this graph that Michigan approach was able to deliver far better results than Pittsburg approach at the end of 50th round. The highest average payoff was 118.5 in case of Michigan approach, whereas Pittsburg approach only improved as high as 114.23 at the end of 50th round.

Figure 13 presents the evolution of average payoff during first 50 rounds of Pittsburg approach. When we compare Figure 13 with Figure 6, it becomes quite apparent that the Michigan approach converges to higher payoffs much faster in comparison to the Pittsburg approach.

We then analyzed two approaches thoroughly in order to identify the exact point where both of them started to deliver high average payoffs. The results of our investigation are presented in Table 6 below. The dominance of Michigan approach is clear from this table.

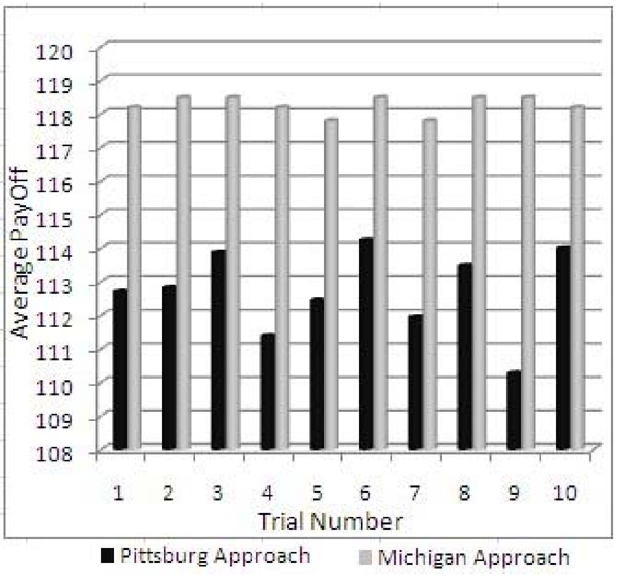


Figure 12. Trial wise avg. payoff at the end of 50^{th} round

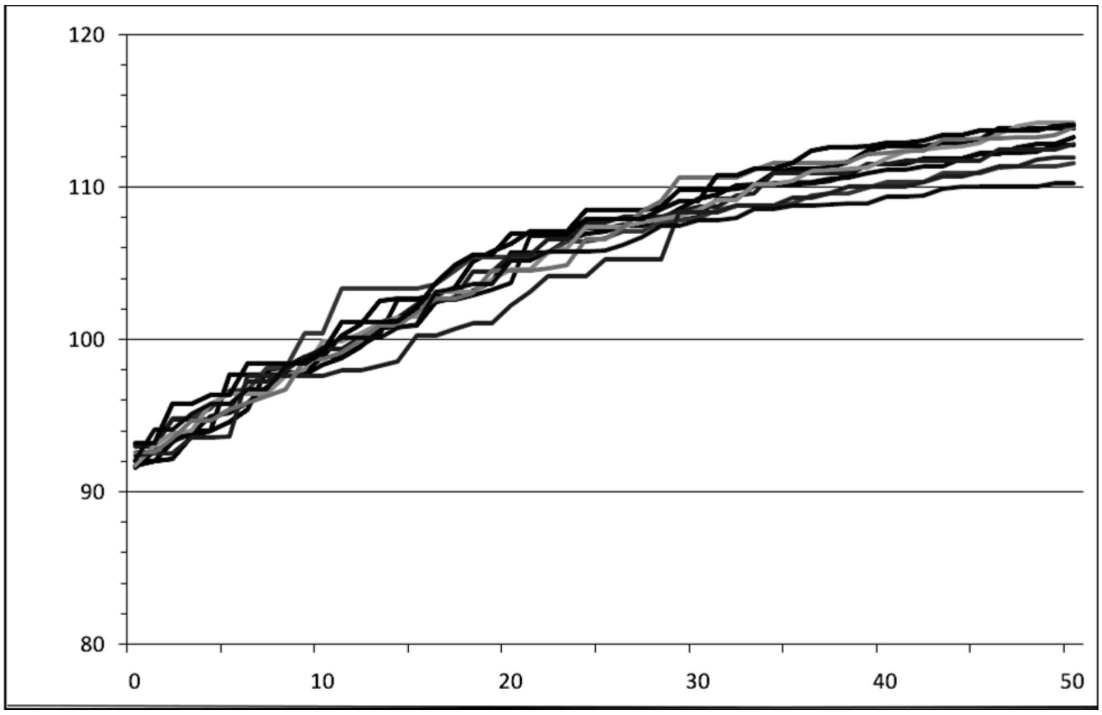


Figure 13. Evolution of average payoff using Pittsburg approach during first 50 rounds

On an average, Michigan approach only took 42 rounds of games to achieve high average payoffs whereas Pittsburg approach achieved this in an average of 194 rounds.

Table 6. Trial wise round number in which higher average payoffs start to emerge (i.e., 118 or greater)

Trial #	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Avg.**
Pittsburg Approach	216	197	203	199	222	187	185	137	238	159	**194**
Michigan Approach	43	32	37	46	52	40	38	47	38	43	**42**

The above mentioned facts indicate that although Pittsburg approach achieves higher payoffs due to the presence of central coordinator, yet good enough results can be achieved using Michigan approach faster and with less amount of computing resources. In many cases, it is important to note how quickly an algorithm converges to a near optimal solution. Furthermore, there is an inherent deficiency in Pittsburgh approach, i.e., the fitness is evaluated at the rule-set level which means GA only receives high-level feedback from the fitness function and it is incapable of evaluating the contribution of individual rules in the overall fitness of the rule-set. This means that it requires additional effort for generating optimal populations. This extra effort compounded with the additional computational resources required to operate at the population level is the main hindrance in the formulation of efficient implementation [29].

6. **Future Work.** The market selection scenario used in this paper is very simplistic. It only assumes that each agent has one type of products and it must sell all its products at one market only. Each product sold at one market has the same price and it is the market conditions that determine the price of every product. Market price is a fixed decreasing function and each market has the same mechanism for calculating its price. The location of market and its surrounding population does not have any impact on the price. If we modify some of these assumptions, we can create complex scenarios. Application of Michigan and Pittsburg approach for such scenarios will prove to be an interesting future research task [8].

In our approach, mutation probability surprisingly did not play an effective role which is against the normal working of an evolutionary algorithm. In fact, it was observed that the algorithm performed well in the absence of mutation probability (i.e., $P_m = 0$). In our future research, we would like to investigate this fact further to see how this parameter can be effectively utilized to achieve high quality solutions.

We have proposed a new Michigan style algorithm to solve market selection problem. The concepts presented in our approach can also be adopted and applied to other real world scenarios such as El-Farol Bar problem [31]. Similarly, the application of prominent learning classifier systems such as XCS and ZCS can also be looked into as a possibility of solving Market Selection problem.

Some other real world problems, where concentration of agents leads to a poor reward, can also be formulated using the same approach. For example, if many students apply for the same department, the entrance exam will become tough. The same route, chosen by many drivers at the same time, causes traffic congestion, i.e., poor reward for the drivers [8].

7. **Conclusion.** In this paper, we have proposed a Michigan-style novel algorithm for the evolution of game strategies among agents with localized selection and coordination. Contrary to the Pittsburg approach, it operated at an individual agent level to maximize its payoff. Each agent knew only the strategies of its neighbors and there was no mechanism for it to know the strategies of all other agents and make moves accordingly. It has been proved quite sufficiently with the help of experiments that this greedy approach (improvement of payoff by each individual agent regardless of others) works well for market selection game as the overall average payoff of all the agents achieve higher values. The results of Michigan approach (without any central coordinator) were then compared with the Pittsburg approach (having central coordinator) and Ishibuchi method of unplanned coordination (without any central coordinator). It was proven that the Michigan approach provide much better overall average payoffs than the Ishibuchi method. In addition to this, it indicates better utilization of the increasing generations of evolutionary

algorithm. The strategies evolved by Michigan method were superior to the strategies evolved by Ishibuchi method.

In its comparison to Pittsburg approach, Michigan approach was able to evolve good strategies much quicker and with less amount of computing resources. It was also proven that the localized selection of Michigan approach was not a hindrance in the way of evolution of good solutions. The Michigan approach was able to evolve almost as good strategies as evolved in Pittsburg approach. It also has the quality of providing solutions having higher payoffs in much less time and rounds of games. All the three approaches mentioned above, evolved the strategies in a very similar fashion.

REFERENCES

[1] C. A. Coello Coello, A comprehensive survey of evolutionary-based multiobjective optimization techniques, *Knowledge and Information Systems*, vol.1, no.3, pp.269-308, 1999.

[2] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, John Wiley & Sons, Ltd, West Sussex, England, 2002.

[3] Y. Guo, X. Cao and J. Zhang, Constraint handling based multiobjective evolutionary algorithm for aircraft landing scheduling, *International Journal of Innovative Computing, Information and Control*, vol.5, no.8, pp.2229-2238, 2009.

[4] Y. Wu, P. Ji and T. Wang, An empirical study of a pure genetic algorithm to solve the capacitated vehicle routing problem, *ICIC Express Letters*, vol.2, no.1, pp.41-45, 2008.

[5] F.-T. Lin and T.-R. Tsai, A two-stage genetic algorithm for solving the transportation problem with fuzzy demands and fuzzy supplies, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(B), pp.4775-4785, 2009.

[6] J.-I. Kushida, I. Nakaoka, K. Kamei and Y. Hoshino, Application of co-evolutionary system for strategy developments of teams in the same generation to team match-up games, *International Journal of Innovative Computing, Information and Control*, vol.5, no.11(A), pp.3667-3676, 2009.

[7] J.-I. Kushida, I. Nakaoka, K. Kamei, N. Taniguchi and Y. Hoshino, A coevolutionary system for strategy development in poker games, *International Journal of Innovative Computing, Information and Control*, vol.4, no.12, pp.3259-3272, 2008.

[8] H. Ishibuchi, R. Sakamoto and T. Nakashima, Evolution of unplanned coordination in a market selection game, *IEEE Transactions on Evolutionary Computation*, vol.5, no.5, 2001.

[9] L. Davis, The evolution of strategies in the iterated prisoner's dilemma, in *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann, San Mateo, 1987.

[10] R. Axelrod, *The Evolution of Cooperation*, Basic Books, New York, 1984.

[11] Special issue on prisoner's dilemma, *BioSyst.*, vol.37, no.1-2, 1995.

[12] H. Ishibuchi, T. Nakashima, H. Miyamoto and C. H. Oh, Fuzzy Q-learning for a multi-player non-cooperative repeated game, *Proc. of the 6th IEEE Int. Conf. Fuzzy Systems*, Barcelona, Spain, pp.1573-1579, 1997.

[13] H. Ishibuchi, C. H. Oh and T. Nakashima, *Competition between Strategies for a Market Selection Game*, Complexity Int., 1999.

[14] L. Davis, *The Handbook of Genetic Algorithms*, Van Nostrand Reingold, New York, 1991.

[15] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[16] J. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, 1975.

[17] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, New York, 1994.

[18] L. Dong, H. Zhang, X. Ren and Y.-L. Li, Classifier learning algorithm based on genetic algorithms, *International Journal of Innovative Computing, Information and Control*, vol.6, no.4, pp.1973-1981, 2010.

[19] L. B. Booker, D. E. Goldberg and J. H. Holland, Classifier systems and genetic algorithms, *AI*, vol.40, pp.235-282, 1989.

[20] J. H. Holland, Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems, in *Machine Learning: An Artificial Intelligence Approach*, R. Michalski, J. Carbonell and T. Mitchell (eds.), Los Altos, Morgan Kaufmann Publishers, 1986.

[21] H. Ishibuchi, T. Nakashima and T. Murata, A hybrid fuzzy genetics-based machine learning algorithm: Hybridization of Michigan approach and Pittsburgh approach, *Proc. of Fuzzy IEEE*, 1999.

[22] S. Wilson, Classifier systems and the Animat problem, *Machine Learning*, vol.2, pp.199-228, 1987.

[23] S. F. Smith, *A Learning System Based on Genetic Adaptive Algorithms*, Ph.D. Thesis, University of Pittsburgh, 1980.

[24] B. Carse, T. C. Fogarty and A. Munro, Evolving fuzzy rule based controllers using genetic algorithms, *Fuzzy Sets and Systems*, vol.80, pp.273-293, 1996.

[25] K. A. de Jong, W. Spears and D. F. Gordon, Using genetic algorithms for concept learning, *Machine Learning*, vol.13, no.2-3, pp.155-188, 1993.

[26] C. Z. Janikow, A knowledge-intensive genetic algorithm for supervised learning, *Machine Learning*, vol.13, no.2-3, pp.180-228, 1993.

[27] S. Sen, L. Knight and K. Legg, Prototype based supervised concept learning using genetic algorithms, in *Evolutionary Algorithms in Engineering Applications*, D. Dasgupta and Z. Michalewicz (eds.), Springer, 1997.

[28] S. F. Smith, Flexible learning of problem solving heuristics through adaptive search, *Proc. of the 8th IJCAI*, 1983.

[29] A. Barry, J. Holmes and X. Llor, Data mining using learning classifier systems, in *Applications of Learning Classifier Systems*, L. Bull (ed.), Springer, 2004.

[30] Z. Wang and M. Li, A hybrid coevolutionary algorithm for learning classification rules set, *ICIC Express Letters*, vol.4, no.2, pp.401-406, 2010.

[31] W. B. Arthur, Complexity in economic theory: Inductive reasoning and bounded rationality, *The American Economic Review*, vol.84, no.2, pp.406-411, 1994.