Co-evolutionary Agent Self-Organization for City Traffic Congestion Modeling

Luis Miramontes Hercog

Eguiara y Eguren #128. Viaducto Piedad.
México, D.F., 08200 México
Tel.:+5255-55199861, Fax:+5255-55199861
lmhercog@yahoo.co.uk

Abstract. This paper introduces a model based on a multi-agent system that learns using the extended classifier system (MAXCS). The agents are informed in the news, after the weather forecast, the congestion levels of the main city avenues and they decide which avenue to use the following day. The results are encouraging, as the agents, both homogeneous and heterogeneous adapt to the congestion thresholds set by the authorities. The main factors for this adaptation are the reward received by the agents and their perception.

Keywords: Road traffic self-organization, Learning Classifier Systems, XCS, Emergent behavior, heterogeneous agents.

1 Introduction

The government of Mexico City has recently (5th December, 2003) introduced a traffic system that measures the congestion in the main avenues of the city. This measure is done counting the number of cars per minute that cross certain key points in the avenues. Then, the congestion for each of the avenues is reported in the evening news following the weather forecast. It has been shown how a number of agents can adapt to several comfort thresholds in modes of transport if they perceive the same information, in this case the congestion report in the news [10].

This paper introduces a system where heterogeneous agents, that learn from the traffic reports, can co-evolve and produce traffic self-organization using inductive reasoning [1]. Inductive reasoning is ideal to solve ill-defined problems like the one analyzed here. There are avenue comfort thresholds set by the city government, though some agents ignore them. Each agent has a level of traffic tolerance, which is assumed to be +-0.1 of the threshold set by the authorities, in case they ignore the figure suggested by the government. The agents translate the capacity reports into rules, which they evolve using an extended classifier system (XCS) [18,4]. Several experiments with different thresholds and different number of agents show that this evolutionary computation approach for traffic self-organization is feasible and could be used both, by the people – to decide which avenue to take for their journey – and by the city authorities to assess which city routes need expansion or an alternative path (such as another metro line, bicycle lanes, or which public transport mode to improve). The experimental results also show that when all the agents pay attention to the suggestion made by the authorities (homogeneous agents) or they are less tolerant to what the authorities suggest (heterogeneous less-tolerant agents), the comfort threshold in the avenues is achieved. As most of the agents do not pay attention to what the authorities suggest, the avenue usage depends on the reward received, rather than the tolerance of the agents. Furthermore, the multi-agent simulation allows individual behavior analysis. These results are supported by 20 sets of experiments. Section 2 presents the model on which this system is based. Section 3 gives a brief overview of the learning algorithm (XCS). Section 4 introduces MAXCS a multi-agent system that learns using XCS. Section 5 explains the experimental settings followed by the results in Section 6.

2 Traffic Simulation Model and Representation

Each of the main avenues of Mexico City, which are the arteries of the main routes through the city, have recently been provided with car counters and cameras to assess the congestion levels. The government has set a novel strategy of providing this data to the citizens after the weather news every evening. The government makes suggestions of how the congestion can impact the travel time based on the number of cars per minute that pass by the car counters. This model proposes and shows that instead of telling the people what to do, it is better to give them the congestion information and let them learn from it.

Therefore, the model used for this simulation tries to be as realistic as possible and takes only the information that the users receive from the traffic report every evening to plan their journey the following day.

It is assumed that the users modeled are traveling in the same routes in the city and that the number of avenues they can use is limited, therefore they have to make a decision on which avenues to use. Sometimes the users can decide to take an alternative route through the city, hence not congesting the avenues. All three possible choices are parallel, therefore no intersections need to be modeled.

The agents learn to model their environment by encoding it into rules. Each rule has a condition and action part representing if-then rules. The condition part represents several previous days of congestion information for each of the avenues. Each agent has comfort preferences, which the agents encode as a binary set: 1 represents that the avenue was not congested, 0 that it was. Every evening the agents compare the number of cars per minute in the avenue to their comfort threshold and encode the bit. For example, a bit set as 10 would tell the agent that it was good to use avenue 1 and not avenue 2. Then, each agent has bit sets for every day information. To keep some homogeneity in the system, the agents have the same memory (5 days). The action part tells the agent which avenue to use if any, i.e. 0 alternative route(not using the avenues that are monitored), 1 take avenue 1, 2 take avenue 2, etc. This model is based on previous findings [9].

$$R = \frac{1000}{e^{\left((congestion - CT)^2 \cdot 1000\right)^2}} \quad (1)$$

The agents learn through a reinforcement learning mechanism (XCS) and a reward is given based on equation 1 using the algorithm in Fig. 1, where CT is the avenue comfort threshold. For every step, all agents encode the levels of congestion of their preferred route and using XCS – explained in the next section – they decide which action to take, either one of the avenues, or a side street. As seen in Fig. 1, if the agents decide to take an alternative route they are rewarded if and only if all the avenues are congested – as it would take longer to reach their destination – and in that case they receive half of the maximum reward. Hence there are AV+1 actions for the agents.

In Fig. 1 the variables are: $MAXAV$, constant that indicates the maximum number of avenues. $MAXAGS$, constant that indicates the maximum number of agents. $MAXACTIONS$, constant value MAXAV+1. *maxReward*, constant with value 1000. *alt_route*, Boolean variable that indicates if action 0 was the best. *pay_Reward*, function that calculates the reward value using Eq. 1. *usage*, array indexed by avenue number. *avenue_result*, array indexed by avenue number, showing whether it was good (1) or not (0) to use an avenue. *rewards*, array of size MAXACTIONS+1, indexed by action number and which in rewards[0] keeps the reward for action 0. All the arrays' cells are initialized to zero.

3 Extended Classifier System (XCS)

Learning classifier systems (LCS) [15] are a machine learning system, situated in an environment that presents a problem that the system tries to learn and solve. The learning process takes place in discontinuous time intervals, where the system is presented a state and it gives an answer to it; depending on the answer there is a reward. If the reward is given just after the action is evaluated, the problem is considered a single step problem; if the reward is given after several actions are taken, it is called a multiple-step problem. If the sensors provide enough information for the system to differentiate the different states of the problem, i.e. each state is different to the system's sensors, the problem is called Markovian, if not it is called non-Markovian [17]. A problem is also called stationary if for the same state, the correct action is the same [17].

LCSs learn by evolving simple strings encoded as if-then rules using a genetic algorithm (GA) [13] and a reinforcement learning algorithm (RLA) [17] to determine the usefulness of the rules. The condition composed generally by 0, 1, # (where # is a wild card) and the action a bit set. The RLA is used to update the rules' fitness that can be, for example, the amount of food or the profit that the rule generates when is activated by the state that the system perceives from the environment. XCS [18,4] is a LCS that uses 3 parameters to measure a classifier's usefulness: the prediction (p) of the reward, the error (ϵ) of the prediction and the fitness (F). The fitness of the rule is an inverse function of the error. XCS uses a Temporal Difference (TD) algorithm to update the values, i.e. learning classifier systems are evolutionary reinforcement learning methods.

```
void Calculate_and_Give_Rewards()
{
  for agent=1 to MAXAGS
   usage[agent.action]=usage[agent.action]+1
  next agent

  for avenue=1 to MAXAVS
  if(usage[avenue]/MAXAGS)<=thresholds[avenue] then
       avenues_result[avenue]=1
    else
       avenues_result[avenue]=0
    end if
  next avenue

  alt_route=true
  for avenue=1 to MAXAVS
    if avenues_result[avenue]=1 then
     alt_route=false
  next avenue

  for action=0 to MAXACTIONS
   if action=0 then
     if alt_route=true then
        rewards[action]=maxReward/MAXAVS
     else
        reward[action]=0
     end if
   else
     rewards[action]= avenue_result[action]*
                     getReward(usage[action],
   end if            thresholds[action])
  next action

  for agent=1 to MAXAGS
    update agent values using rewards[agent.action]
  next agent
}
```

Fig. 1. The pseudo code algorithm for the reward calculation and agent update. This algorithm is run every time step after all the agents take their action [11].

The interactions between XCS and its environment in a single step problem are cognitive cycles: perception of the environment, activation of the rules that match the current environmental state (forming the match set [M]), action selection (forming the action set [A]), environmental evaluation of the action, reward and reinforcement of the rules that fired the action. The action is selected from all the possible actions in [M], randomly if exploring or deterministically

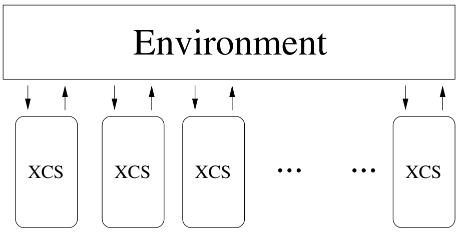


Fig. 2. MAXCS: each agent is a single, independent XCS.

if exploiting. When exploiting, the prediction value ($\frac{\Sigma F_j p_j}{\Sigma F_j}$, where j are the classifiers with the same action) is calculated for all the possible actions in [M], then the action with the highest value is taken. The GA is used to create new rules from fit rules. The GA is invoked when the average of the time stamp of the rules in [A] is greater than the GA threshold (θ_{GA}) [18,4]. The rules generated by the GA are inserted back in the population, and in order to keep the size of the population constant, the rules are deleted based on the estimated size of [A], the fitness and the experience of the rule (for all the deletion techniques and more details about XCS the reader is referred to [18,16,4]).

4 MAXCS: A Multi-agent System That Learns Using XCS

MAXCS [12,9] is a multi-agent system that learns using a learning classifier system, each agent is represented by an XCS (Fig. 2).

The accuracy based fitness, the RLA used by XCS to learn, and its genetic algorithm let XCS evolve accurate and compact rule sets [16]. These rules as in other LCSs, can be read as if-then rules. This property has been very useful for the analysis of the behavior of the agents.

MAXCS exploits the possibility of representing individuals directly, (both their behavior and their interactions) and provides the detailed analysis required to represent agents correctly [7]. A full "dissection" can be made analyzing the population of each agent, as well as their individual answers, providing a clearer result that is easier to understand.

The activation of each set of rules can reveal what is happening individually in the system, unveiling some features of the system that otherwise would remain unexplained.

This multi-agent system satisfies the requirements of multi-agent simulation, based on the idea of a computerized representation of entities' behavior in the world. This computerized representation gives the possibility of representing a phenomenon, which emerges from the interactions of a set of agents with their own operational autonomy [7].

Holland [14] proposed the use of learning classifier systems to represent the agents for an artificial stock market [2], which started the research on agent-based computational economics (ACE), e.g. [14,2].

The rationale behind applying learning classifier systems to social simulation, in this paper, is that they are self-explanatory: by looking at their rule population and the update of the estimation values, a step by step analysis can be made. By having a self-explanatory system, for social simulation, the behavior of the agents can be analyzed in detail and properly explained. From an individual analysis viewpoint this feature can be very valuable. For the work presented here, this can be interpreted as a snapshot of a person's mind at the exact moment of taking the decision of which avenue to take, and the reasons for that decision.

Davidsson [5] has shown how multi-agent based simulation, and other micro simulation techniques, explicitly attempts to model specific behaviors of specific individuals. He compared them favorably to macro simulation techniques that are typically based on mathematical models, where the characteristics of a population are averaged together and the model attempts to simulate changes in these averaged characteristics for the whole population.

Davidsson's [5] findings – combined with Ferrari's [8] – have proved that a mathematical simulation model has to be discarded and restated from the beginning when new variables are added to the system – bring another opportunity niche for the micro-LCS-based simulation presented here.

On the other hand, economists (e.g. [1]) and game theorists (e.g. [3]) have shown that some human decisions are not very rational. Based on this assumption of the lack of rationality in human decisions, MAXCS is used to try to achieve the suggested usage of the avenues in the city, based on inductive reasoning agents [9]. Furthermore, the strategies used by the agents are not set in advance, but they are evolved and chosen autonomously by each agent. Hence, the simulation would provide a framework for assessing the different avenues and their congestion levels and could be used for planning.

5 The Experiment

After the successful use of 10 homogeneous agents and 2 avenues (not shown), a more difficult setting has been selected. First, 101 homogeneous agents with 2 avenues, after this setting yielded self-organization, heterogeneous agents were tested. The heterogeneity is computed randomly by assigning 3 types of agents (type 1, 2 and 3), which take the thresholds told by the authorities, and randomly add or subtract 0.01 to them. Then, during the run, the modulus by 5 operation is computed using the agent number to assign the agent type: 1 for type 1, 2 for type 2, 3 for type 3, leaving 0 and 4 with the thresholds told by the authorities. Therefore, only 40% of the agents pay attention to what the authorities suggest for comfort threshold.

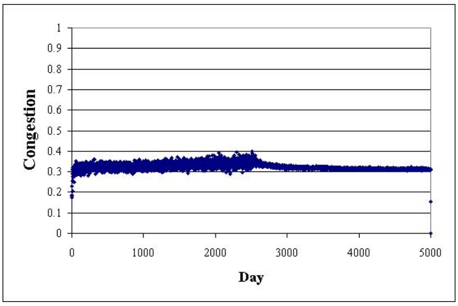
Each agent is represented by homogeneous XCS with conditions of size 5 and 3 possible actions with the parameter values: # probability= 0.333, explore/exploit rate(P_x)= 0.5, crossover rate (χ)= 0.8, mutation rate (μ)= 0.02,

$\theta_{GA} = 25$, minimum error (ϵ_0)= 0.01, subsumption experience 50 and learning rate (β)= 0.2, subsumption deletion (a type of rule "condensation") is used in the GA and the action set, these parameter values have yielded good results before [11]. The populations of the agents are initialized using supersaturation (each 25% of the population is generated with a different random seed) [6]. The covering mechanism is used only when [M] is empty (θ_{mna}) and all three possible actions are generated, taking as initial values p=20.00, ϵ=0.0 and F=0.1. Exploration in these experiments is done by selecting randomly an action with P_x=0.5 from [M] (first 2500 steps). P_x=1.0 for full exploitation trials(last 2500 steps). The former to allow the agents to learn, the latter to test what they have learned. Avenue 1 has a threshold of 30 cars/min and avenue 2 of 50 cars/min., set to 0.3 and 0.5 respectively for the experiments.

Agent heterogeneity is given by how the agents perceive their environment and react to the congestion of the avenues. After the adaptation to the 0.01 random tolerance change 0.1 was tested too. The use of a percentage rather than the exact number of cars per minute allows the system to be flexible and increase the number of agents as it is needed. As the number of agents used in these experiments is 101, the threshold is almost the same as the number of cars per minute tolerated by the agents.

6 Results

The results are presented differently depending on the focus that is given. First, the 20 runs, with different random seeds, are averaged to test if the agents are learning. Figures 3 and 4 show the average of the 20 runs: Fig. 3 shows all 101 homogeneous agents, Fig. 4 shows a congestion coefficient random variation of 0.01 for each agent's perception, which is similar in overall behavior to a variation of 0.1 (only shown as individual behavior in Fig. 6). Then, to unveil the reason for the difference between the runs, an individual run is analyzed. As the individual behavior would get lost with an averaged agent avenue usage.



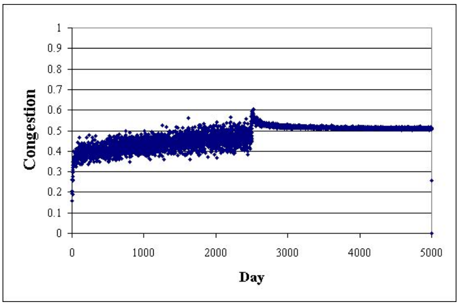


Fig. 3. Average of 20 runs using 101 homogeneous agents with CT_1=0.3 (left) and CT_2=0.5 (right).

A pattern can be perceived in Figs. 3 and 4: the agents adapt to the threshold set very early in the experiment, it takes the agents less than 30 days to achieve the congestion levels set, especially avenue 2 in Fig. 3 is under-congested until day 1500. Once the exploration phase is switched off and the agents use the rules they have learned (day 2500), a closer value to that set by the authorities is achieved in all of the cases.

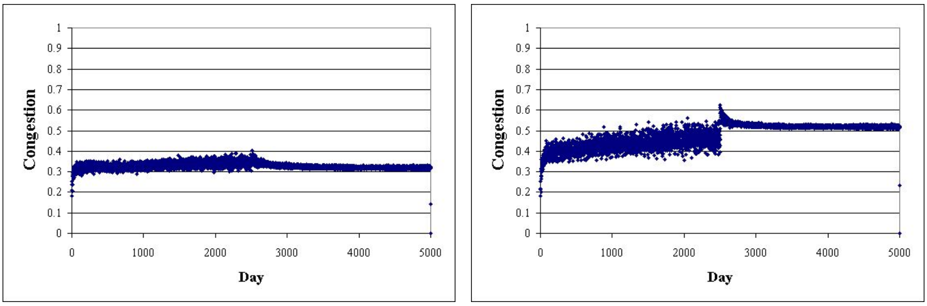


Fig. 4. Average of 20 runs using 101 heterogeneous agents with CT_1=0.3 (left) and CT_2=0.5 (right); 40% of the agents pay attention to the authorities and the rest have +- 0.01 in their thresholds randomly. A +-0.1 change yielded a similar plot, but a different individual behavior.

The experiments with homogeneous agents (Fig. 3) and those with a random 0.01 change (Fig. 4) are more similar than both of the heterogeneous agents. A gap of 0.01 between the congestion wanted and the value achieved by MAXCS. Homogeneous agents yield a value closer to that set by the authorities than heterogeneous agents.

Agents changing their actions all the time produce the variation above the congestion threshold. This phenomenon is produced by the nature of the problem: there is not enough capacity for all the agents to use the avenues, therefore some of them will have to take the alternative routes using little streets in the city. Even though all of them (for the sake of the model) would prefer to use the main avenues, as they would move faster.

The agent individual behavior is analyzed in figures 5 and 7. Each of the squares represents an agent in the system. The squares are shaded depending on the use of the avenues every 100 days: [0,25] white square, [26,50] gray square, [51,75] dark gray square and [76,100] black square. Agent 1 is the top left agent, agent 10 is at the end of the first row and agent 101 is the square at the bottom left. Hence, those agents using an avenue all the time are shaded in black, and those not using it in white, those using it not so often in gray. For example, in Fig. 5 it can be seen that agents 10, 35 and 95 are using always the side streets, while those agents in gray are changing between both avenues and the side streets.

These individual behavior figures are plotted using a single run, as averaging would blur the results, due to the random avenue usage. Random meant as that for different runs, different agents will use different avenues, not random behavior of the agents as such.

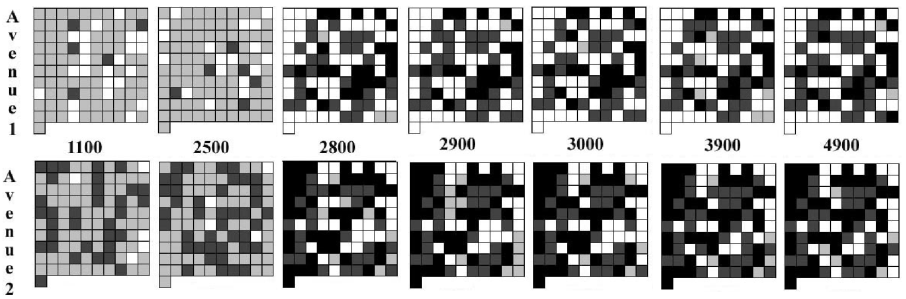


Fig. 5. Individual avenue usage for 101 homogeneous agents.

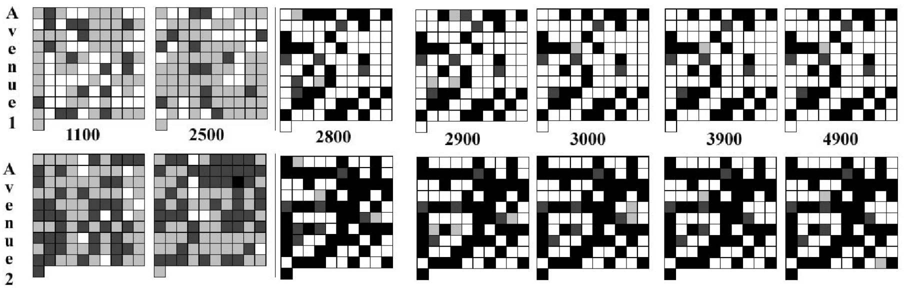


Fig. 6. Individual avenue usage for 101 heterogeneous agents with a 0.1 random change.

The three individual behavior graphics are taken at different steps, mainly the first one (1100) at a time where the agents have adapted already, are still exploring, hence they have not fixed to use a specific avenue. The fixation comes after step 2500, when the GA is not working anymore and the agents are exploiting their knowledge. It can clearly be seen a group of agents which is in light gray shade. These agents are balancing the usage of the avenues and the side streets.[1] The fact that some agents fix to a given avenue, either 1 or 2, leaves with no choice the other agents, but to keep trying the side streets and the avenues. When these vacillating agents take the side streets, they allow the others

[1] These agents are called vacillating agents, and they produce the Nash equilibrium of the problem [11].

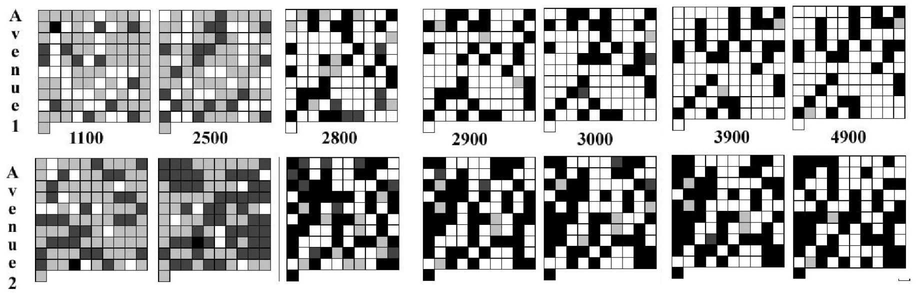


Fig. 7. Individual avenue usage for 101 heterogeneous agents with a 0.01 random change.

to use the avenues without congestion, while when they take the avenues they over-congest them, producing comfort in the side streets. Hence, they balance the system.

From the different experimental settings shown in Figs. 3 and 4, it can be inferred, from the agents behavior, that the main factor for adaptation is the reward, rather than the perception of the agents. Therefore, this simulation can be used by the authorities to set the policies (i.e. rewards) to achieve the congestion levels wanted in the city avenues. It can be seen that heterogeneity is a crucial factor in the agents' behavior (Fig. 7), as they fix quicker to either avenue, leaving 20% of the agents to change between avenues and side streets. Analysis of the evolved rules show very general rules, (e.g. ##########:2) for some agents, but very specific (e.g. 100##11#01:0) for others. This specificity is produced by the initial preference of the agents for the side streets or a given avenue. Even though one would think that the latter example would be more advanced than the former, it is the opposite. The agents that fix to use certain avenue, as the first rule shows: whatever happens take avenue 2, is the best strategy. This is due to the nature of the problem, because by doing this, the agent will profit from the vacillation of those that are changing between the side streets (action 0) and the different avenues (actions 1 and 2).[2] As Littman [17] has pointed out, when agents in a multi-agent problem achieve a Nash equilibrium, it is considered an optimum performance. Hence, the solution that emerged from MAXCS's co-evolution is the optimum performance. This is the reason given by the author to claim that the city authorities and the people in general could use the model presented here to forecast any changes in the avenue congestion or to forecast the avenue usage.

[2] Because a Nash equilibrium is the best possible action, given the other agents' action and no agent can profit better, but by changing strategy [3]. The interested reader can find the game theoretical discussion of the problem in [9].

7 Conclusions

This paper has shown how a homogeneous and heterogeneous multi-agent system can learn from the traffic reports co-evolving to produce traffic self-organization using inductive reasoning. Mainly this co-evolution being an effect of the reward the agents receive, rather than their perception of the traffic information.

Whether they decide to use it or not, the traffic information is relevant for some of the agents, but it is not a crucial factor. The solution has been found in some agents that use consistently either avenue 1 or 2, and leave no option for the others to change from the side streets to the avenues, balancing the avenue congestion.

Another factor that is crucial for the agents' behavior was the level of tolerance set individually, as the agents have learned mainly based on the reward they receive, rather than in their individual perception.

Several experiments with different thresholds and different number of agents show that this evolutionary computation approach for traffic self-organization is feasible and could be used both, by the people – to decide which avenue to take for their journey – and by the city authorities to assess which city routes need expansion or an alternative path (such as another metro line, bicycle lanes, or which public transport mode to improve).

The learning classifier system approach has been successful in achieving optimal performance (in the form of a Nash equilibrium) and in evolving readable rules, which indicate the preference of the agents, hence an individual behavior pattern can be analyzed.

The results reported are encouraging and more research is planned in this direction.

References

1. W.B. Arthur. Complexity in Economic Theory: Inductive Reasoning and Bounded Rationality. *The American Economic Review*, 84(2):406–411, May 1994. http://www.santafe.edu/~wba/Papers/El_Farol.html.
2. W.B. Arthur, J.H. Holland, B. LeBaron, R. Palmer, and P. Taylor. Asset Pricing Under Endogenous Expectations in an Artificial Stock Market. In W.B. Arthur, S. Durlauf, and D. Lane, editors, *The Economy as an Evolving Complex System, II*, Reading, MA., 1997. Addison-Wesley. Proceedings Volume XXVII.
3. K. Binmore. *Fun and games: a text on game theory.* D.C. Heath, 1992.
4. M.V. Butz and S.W. Wilson. An algorithmic description of XCS. In P. L. Lanzi, W. Stolzmann, and S. W. Wilson, editors, *Advances in Learning Classifier Systems. Third International Workshop (IWLCS-2000)*, pages 253–272, Berlin, 2001. Springer-Verlag.
5. P. Davidsson. Multi-Agent Based Simulation: Beyond Social Simulation. In S. Moss and P. Davidsson, editors, *Second International Workshop, MABS 2000*, volume 1979 of *LNAI*, pages 97–102. Springer-Verlag, 2000.
6. L. Davis, C. Fu, and S. W. Wilson. An Incremental Multiplexer Problem and its Uses in Classifier System Research. In W. Stolzman, editor, *Fourth International Workshop on Learning Classifier Systems - IWLCS-2001*, pages 23–32. Springer-Verlag, 2002.

7. J. Ferber. *Multi-Agent Systems: An introduction to distributed artificial intelligence.* Adisson Wesley, 1999.
8. P. Ferrari. A model of urban transport management. *Transportation Research Part B*, (33):43–61, 1999.
9. L. Miramontes Hercog. *Evolutionaty and Conventional Reinforcement Learning in Multi-agent Systems for Social Simulation.* PhD thesis, South Bank University, 2003.
10. L. Miramontes Hercog. *Learning Classifier Systems Applications*, chapter Traffic Optimization using Multi-agent Systems and Learning Classifier Systems. LNAI. Springer-Verlag, 2004. To appear.
11. L. Miramontes Hercog. Multi-agent inductive reasoning using co-evolutionary classifier systems: the Multibar Problem, 2004. Submition to Evolutionary Computation Journal.
12. L. Miramontes Hercog and T.C. Fogarty. Social simulation using a Multi-Agent Model based on Classifier Systems: The Emergence of Vacillating Behaviour in the "El Farol" Bar Problem. In P.L. Lanzi, W. Stolzman, and S. W. Wilson, editors, *Advances in Learning Classifier Systems. Proceedings of the Fourth International Workshop in Learning Classifier Systems 2001.* Springer-Verlag, 2002. LNAI series 2321. ISBN: 3-540-43793-2.
13. J.H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.
14. J.H. Holland and J. Miller. Artificially Adaptive Agents in economic theory. *American economic review*, 81(2):265–370, 1991.
15. J.H. Holland and J.S. Reitman. Cognitive Systems Based on Adaptive Algorithms. In D. A. Waterman and F. Hayes-Roth, editors, *Pattern-directed inference systems.* New York: Academic Press, 1978.
16. T. Kovacs. XCS Classifier System Reliably Evolves Accurate, Complete, and Minimal Representations for Boolean Functions. In Roy, Chawdhry, and Pant, editors, *Soft Computing in Engineering Design and Manufacturing*, pages 59–68. Springer-Verlag, London, 1997.
17. M. L. Littman. Value-function reinforcement learning in markov games. *Journal of Cognitive Systems Research*, (2):55–66, 2001.
18. S.W. Wilson. Classifier Fitness Based on Accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.