# DFA Learning of Opponent Strategies

**Gilbert Peterson and Diane J. Cook**

University of Texas at Arlington
Box 19015, Arlington, TX 76019-0015
Email: {gpeterso,cook}@cse.uta.edu

**Abstract**

This work studies the control of robots in the adversarial world of "Hunt the Wumpus". The hybrid learning algorithm which controls the robots behavior is a combination of a modified RPNI algorithm, and a utility update algorithm. The modified RPNI algorithm is a DFA learning algorithm, to learn opponents' strategies. An utility update algorithm is used to quickly derive a successful conclusion to the mission of the agent using information gleaned from the modified RPNI.[1]

## Introduction

Developing single purpose learning algorithms for manipulating robots in a given domain, has given way to using hybrid learning algorithms that yield a more robust behavior. An approach to combine a planner with a reactive agent to play the game of soccer has been explored in the literature (Sahota 1994). In this paper, we are interested in developing a hybrid learning algorithm to manipulate robots in an adversarial game, often known as "Hunt the Wumpus".

In the wumpus world, there are two adversaries, in our version played by robots. The wumpus is given a number of set behavior patterns/strategies to follow to foil the agent. Our agent learns the strategies of the wumpus, and uses the learned information to complete a mission. The wumpus strategies are learned and stored in a DFA together with probabilities indicating the number of times the agent has seen the wumpus follow this strategy. The learned DFA is used to output the next most probable abstract move to be made by the wumpus. The agent computes where the move will lead in the literal space, and using an utility/reward update function derives a path to complete its mission while avoiding the wumpus.

Because the strategies of the wumpus can be simply represented as a finite state machine. The method of learning the strategies of the wumpus was to learn a DFA representation of the strategies.

Researchers have shown that learning the DFA can be done both incrementally (Dupont 1994) and PAC learnably (Parekh and Hanovar 1994). Because the DFA can be learned quickly and incrementally, it makes for an excellent choice in learning the wumpus strategies in an unsupervised, fashion.



## Wumpus World

The Wumpus World domain is based on an early computer game. The basis for the game is an agent who explores an N by N grid world while avoiding a creature named the wumpus. Other elements of the world consist of bottomless pits (which don't affect the wumpus), and bars of gold. The objective of the game is to collect as many of the gold bars as possible, return to the initial grid location [1,1] and exit the cave. The information the agent senses/receives each turn to aid in locating the gold and avoiding the wumpus and pits, is a five element percept. If the agent is in the square containing the wumpus or directly adjacent squares the agent perceives a stench. If the agent is in a square directly adjacent to a pit it will perceive a breeze. If there is a gold bar in the same location as the agent it will perceive glitter. If the agent runs into a wall it will perceive a bump. If the agent shoots its arrow and kills the wumpus it will hear a scream. The actions allowed the agent are to move forward, turn left, turn right, grab gold, shoot the arrow, and climb out of the cave. The action allowed the wumpus are to move forward, turn left, turn right, and do nothing.

Changes to the domain were made to use sonar readings instead of the five element percept, because the robots which are to play the roles of the agent and the wumpus possess a sonar grid. The agent is given the locations of the pits and gold bars. Using the sonar grid, the robot can determine where the walls are, its position in the world, and whether or not it is in horizontal or vertical alignment with the wumpus.

Obstacle were also inserted into the domain to increase the world complexity. If the wumpus and agent are on opposite sides of an obstacle, neither can see the other.

The robots playing the roles of the agent and wumpus are Trilobots. The Trilobot robots have been given an eight element sonar array. The sonar array polls all eight sonars, then rotates 22.5 degrees and polls all eight sonars again. This method gives a total of sixteen distance readings. A picture of the robots in a portion of wumpus world can be seen in Figure 1.

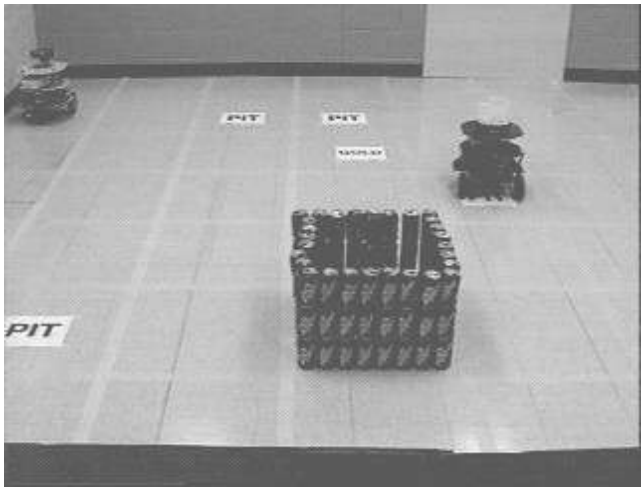

Figure 1

## DFA Algorithm

The behavioral strategies that the Wumpus can employ are to 1) move to a gold bar and circle it clockwise, 2) move to a gold bar and circle it counter-clockwise, 3) attack the Agent, 4) sit or do nothing, and 5) hide from the Agent behind an obstacle until the Agent is close and then attacks. The Wumpus will only follow one of these strategies throughout a given game. A modified RPNI algorithm, infers a DFA structure representative of the Wumpus strategies given a series of general moves. The general moves are derivations of the transitions made by the wumpus as observed by the agent.

The logical choice in using a DFA to learn the Wumpus strategies would be to represent the Wumpus World as the states in the DFA, and the actual action as the transition. For an 8x8 world there are 64 states, representing all locations where the Wumpus can reside. However, information on the Agent position is also needed (in case the Wumpus is attacking the Agent), and this squares the number of possible states. It is also necessary to represent all possible configurations of gold bars and obstacles because they relate to the strategies. With all this information included in the state, there is an immense number of states represented.

Instead of using this representation, each state represents a general move. The general moves abstract away the specifics of the Wumpus World domain. This representation reduces the amount of data that needs to be dealt with and allows the method to be scaled to larger problems. The transitions from state to state instead of being a specific move, since the general moves are now states, are transitions of the Wumpus from making one general move to the state of making another.

The wumpus has a library of five behavioral patterns. These five patterns are to 1) attack the agent, 2) select one of the gold bars, move to it and circle it clockwise, 3)select one of the gold bars, move to it and circle it counter clockwise, 4) select one obstacle, move to it and position himself out of view of the agent, when the agent is within a short distance, the wumpus will switch to the attack the agent pattern, or 5) sit. The wumpus will only follow one of the patterns throughout a given trial.

A modified RPNI algorithm is used to learn a DFA structure representative of the five behavior patterns given a series of general moves. The general moves are derived from the transitions made by the wumpus as observed by the agent. The general moves are 1)move closer to gold bar 1,2 or 3, 2) move closer to obstacle 1,2,3, 3) attack the agent, 4) hide from the agent, 5) sit or do nothing, and 6) move closer to the north wall, south wall, east wall, or west wall. Each of the wumpus behavioral strategies is a combination of several of these general moves. The agent determines the transition made by the wumpus by comparing two verified positions of the wumpus and calculating the relational distances between all the objects in the world and the two positions of the wumpus. The general move is determined by testing all of the relative distances with a set of rules to determine if the wumpus is moving closer to or circle an object in the world. The agent retains the general moves made by the wumpus from the offset of the trial in a transition list.

The algorithm chosen to learn the DFA is a modification of the RPNI algorithm (Oncia and Garcia 1992). The RPNI algorithm performs an ordered search of an accepting DFA by attempting to merge consecutive elements of the prefix set. The prefix set is a set of all of the prefixes of all of the acceptable strings. The first step is to create a PTA(instance+). The PTA(instance+) is a finite automata tree created from only the positive examples. Each state included in the PTA(instance+) has a corresponding element in the set of prefixes of the set of positive instances. Each step of the algorithm attempts to merge a prefix of the set of positive instances with possible suffixes of the PTA(instance+), while remaining consistent with the set of negative instances.

The RPNI algorithm uses a set of positive and a set of negative samples to derive a DFA based on the acceptance of states. For the wumpus world all states are acceptor states since what is being looked for is the next move, not string acceptance or rejection. Because there are no negative samples, we modify the RPNI algorithm to retract based on matching states and pattern repetition rather than on the accepting string.

The modified RPNI algorithm begins by creating a PTA(instance$^{+}$) just as in the original, the instance+ set comes from the transition list. The prefix set that is incrementally searched and merged in the RPNI does not carry over to the modified version. Instead of using the prefix set, the modified RPNI algorithm performs the search space from the initial node. The each path is then searched for possible loops.

For the Wumpus World the initial node is always sit. The nodes used in the DFA structure of the Wumpus World upon creation will point to themselves. This saves a step,

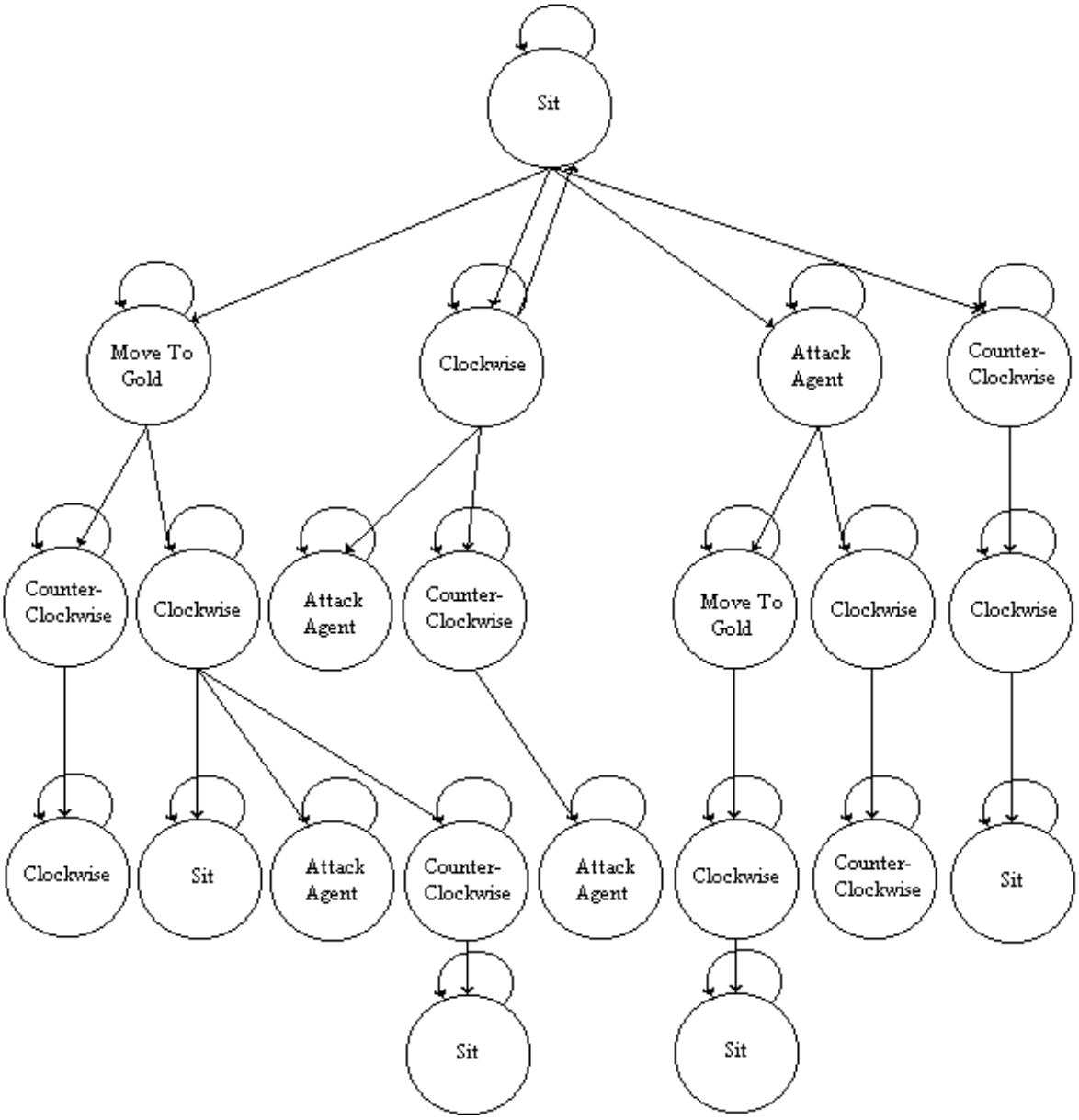


Figure 2

since each transition of a general move to the same general move would be pointed back upon itself. The initial structure will be a sit node which upon receiving a sit returns to itself. Each edge also has a weight attached, the weight represents the number of times the edge has been traversed. The weights of the edges from a given node are used to predict the next most likely transition from that node.

The agent updates a probability attached to each state as it continues to observe the behaviors of the wumpus. These probabilities are then used to predict the next general move to be made by the wumpus. An example learned structure is shown in Figure 2.

## Utility Driven Planning

Once the agent has learned a DFA representing the wumpus behavior patterns, the DFA is used to predict the next move of the wumpus and a utility update function determines the next optimal move for the agent. The agent uses a value iteration algorithm to calculate the utilities of each of the states in the wumpus domain. The value iteration algorithm prescribes a path to gather the gold while avoiding the pits and wumpus. Since the wumpus and agent move simultaneously, the value iteration algorithm is run at each step before the agent selects a move.

The value iteration algorithm uses a reward array <x, y,

numgold>, a utility array <x, y, transitions, numgold>, and a model describing the agent's moves to find a path through the wumpus domain. The reward array contains all of the rewards for all of the possible states the Agent can reach. Each state in the reward array consists of an x and y location and the number of gold bars the Agent has in possession at that time. Each state the Agent can be in and each move the Agent can make will have a separate utility value. The state information for the Agent in this table as in the reward array is the x and y location, and the number of gold bars the Agent possesses at the time. The model describing the agent's moves is a set of conditionals that simulate the Agent in the environment. An example would be if the Agent is next to and facing a wall he can not move into it, or if the Agent is facing up and chooses to go forward his y location will increase by one.

Because a close-to-optimal path is desired to get the gold and get out, each location in the reward array is initially set to -0.05. If the agent knows or believes that there is a pit or wumpus in a given grid location, the reward for this location is set to -10.0. This keeps the value iteration algorithm from prescribing a path through a lethal location. Locations with gold are then assigned a reward of 1.0, 2.0, 3.0, and so on for as many gold bars that exist in the world. The incremental values are assigned randomly and not in a specific order to create a shortest path length. Location [1,1] is assigned a value of one more than any gold pieces, so that after the gold has been collected the agent will leave. The utility array at the beginning of each turn is reset to all zeros.

During each turn, the Agent determines the most likely position of the Wumpus, and inserts this into the reward array. The Agent will alternate between updating the utility array and testing the utilities returned to determine if a path to collect all of the gold bars and exit exists. Once a path to reach all goals exists, the Utility Update Function will exit, and the first move of the path is executed. The first move of the path is the move which will have the highest utility for that location and predicted Wumpus location.

A value iteration is used to update the utility values in the array. The value iteration will calculate the utility of each state into a temporary utility array. The new utility value for a state is the sum of the reward for the state and the maximum utility value of all the states reachable from the state in a single move. After recalculating all of the utility values into the temporary utility array, the values then become the new utility values.

## Experimentation and Results

In this section we demonstrate that the modified RPNI algorithm is effective at learning the wumpus strategies and thus aiding the agent in winning games. Each experiment consists of fifty trials executed by the agent. Each trial consists of twenty-five steps, where the wumpus strategy is randomly selected from those available. In Learning Mode the agent moves counter-clockwise around the wumpus world (pits have no effect) to locate and follow the wumpus. Once located, if the agent loses the wumpus, the agent would begin to sweep the world in a counter-clockwise fashion looking for the wumpus.

Four different world definitions exist to test the strategy learning ability of the agent. All four world definitions are based on an 8x8 world with a single agent and a single wumpus. In the first world definition, there is only one gold bar, and no obstacles. Since their are no obstacles, strategies such as hiding from the agent are not tested.

The results from this first world definition show that the algorithm indeed learns the wumpus strategies, but learns so quickly that the learning curve was exceptionally steep. The results also demonstrate how completely the DFA represents the strategies of the wumpus. In order to determine that the agent is indeed learning the wumpus strategies, the second world definition is created. In the second world definition, the agent is given access to the wumpus's orientation and position at all times, God mode. This definition also sees the inclusion of the wumpus selecting one of up to three gold bars to move to in the circle gold bar strategies. These additions were made to increase the world complexity, and expressly slow the learning curve. The results from this experiments sow the learning curve slowing, yet still resulting in a steep rise and a stable maximum. The maximum shown is approximately 84%. This means that on average, the agent missed two or three of the wumpus moves. This relates to the two or three moves, it would take anyone to determine where the wumpus was heading.

The third wumpus world definition introduces the obstacles into the wumpus world model. With the inclusion of the obstacles, the wumpus gains the strategy to choose to hide from the agent. For these tests, the agent is in God mode. The exception to the God mode from the third world is that if the agent and wumpus are on opposite sides of an obstacle, the wumpus's orientation and position are not available. This world shows a slowing in the learning curve and a slightly lower peak success.

The fourth wumpus world definition is the same as the third without God mode on. The information allowed the agent in the definition is only the information the sonars can glean from the environment. The agent is still given the positions of the gold bars and pits, the wumpus position and orientation are not known. It is important to note the amount of information being taken from the agent. The orientation of the wumpus is now not available to the agent because the sonar sensors on the robots can not determine orientation. The sonars can not see into every square of the world, so at best they can return information on about 60 percent of the world. This is the most limited world definition tested with.

The results of the fourth world definition show a marked decline in the learning slope, and a lower overall success. A great deal of this is due to the uncertainty of not being able to locate the wumpus. There are sixteen distinct wumpus general moves, and if the agent just randomly selected one of the sixteen, the agent would be

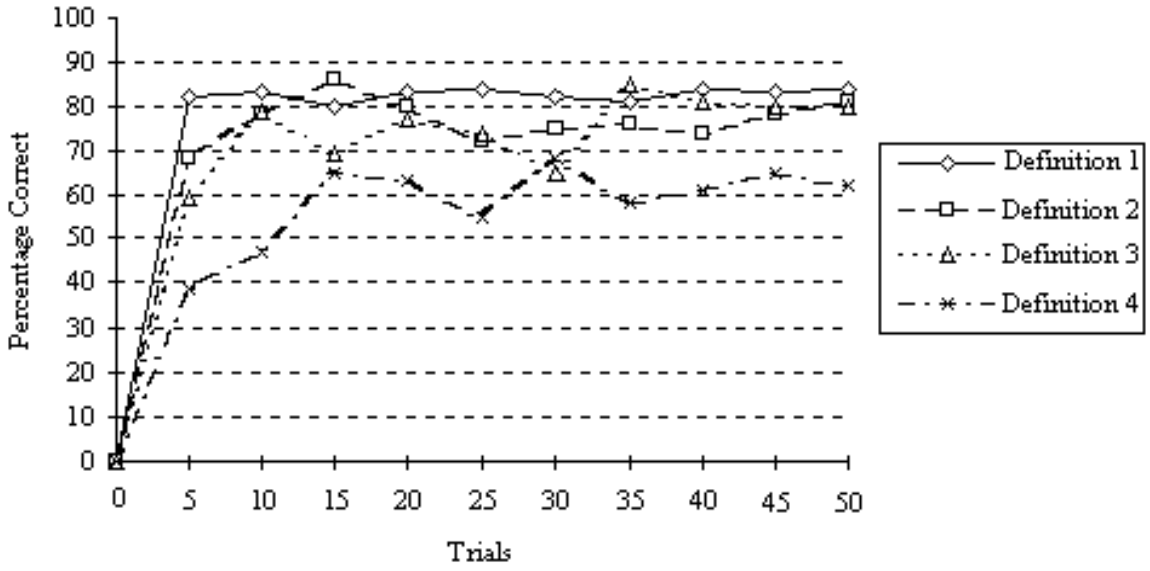


Figure 3

correct seven percent of the time.

The percentage of correct predictions of the wumpus' next move is on average 63% over 25 trials for the world four definition. For the twenty-five step trial, the agent correctly predicted the behavior of the wumpus seventeen of those steps. It takes a couple of moves at the beginning of the trial for the wumpus to get orientated and make its first move before it can be predicted with any certainty what the wumpus is moving toward, this of course is only if the agent has already located the wumpus. The learning curves for all four of the world definitions can be seen in Figure 3.

## Discussion and Conclusion

In adversarial games, the ability to learn a DFA representing the transitions of the strategy of the adversary, is a successful tool to outperform the adversary. The DFA Learning algorithm acceptably learns the behavior patterns of the wumpus. The DFA Learning algorithm learns all of the strategies of the wumpus in one routine structure.

The DFA Learning algorithm in the Wumpus domain was very successful in the smaller definitions or the definitions in which the Agent is given a great deal of information. The DFA Learning algorithm does not do as well in the Wumpus World in which the definition is larger or in which the Agent has less information. Part of this is due to the fact that the sonar sensors can only sense up to sixty percent of the grid locations. Another source of this is the manner in which the general moves are extrapolated from the actual wumpus moves.

## References

Dupont, P. 1994. Incremental Regular Inference. In Miclet, L., and Higuera, C., eds., *Proceedings of the Third ICGI-96, Lecture Notes in Artificial Intelligence 1147,* 222-237, Montpellier, France: Springer.

Parekh, R. G. and Honavar, V. G.. 1997. Learning DFA from Simple Examples. *Proceedings of the Eighth International Workshop on Algorithmic Learning Theory (ALT'97),* Sendai, Japan. Oct 6-8, '97 (To appear). A version of this paper was presented at the Workshop on Grammar Inference, Automata Induction, and Language Acquisition (ICML' 97), Nashville, TN. Jul. 12, ' 97.

Sahota, M. K. 1994. Reactive deliberation: An architecture for real-time intelligent control in dynamic environments. In *Proc. 12th National Conference on Artificial Intelligence*, 1303--1308.

Oncina, J. and Garcia, P. 1992. Inferring Regular Languages in Polynomial Updated Time. *Pattern Recognition and Image Analysis: Selected Papers from the IVth Spanish Symposium.* 49-61.