**Artificial Intelligence**

**Algorithm Development, Object Oriented Design and Reinforcement Learning, In AI Game Play Using MicroRTS**

**BY CRAIG NELSON and Adam Swift**

MicroRTS is a small implementation of a Real Time Strategy Game designed and implemented by Santiago Ontanon as a research project on Machine Learn for a Real Time Strategy Game

Santiago Ontanon is a professor of Computer Science at Drexel University

Craig Nelson is an Assistant Professor at Community College of Philadelphia

Adam Swift is a K12 Computer Science Instructor at the Egg Harbor School District in New Jersey

Craig Nelson and Adam Swift worked collaboratively as research assistant under Santiago Ontanon as an advisor to produce this research project. Parts of this version of the projects forks from the original due to a variance in the intended audiences

The entire collection of documents and modified versions MicroRTS to support this lesson may be downloaded at GITHUB from the following URL: <https://github.com/nelsoncra/Project1>

This project should be cloned and continued

***Research Experience for Teachers in Engineering and Computer Science:  
Machine Learning, Big Data and CS Principles***

**Description:**

The first use of the term Artificial Intelligence: The founding father of the term artificial intelligence will be studied. Some of his accomplishments will be reviewed

“We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College inHanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves.” Computer Scientist - John McCarthy, August 31, 1955

AI is prevalent throughout society including, but not limited to, search engine algorithms, selection of cell towers to transmit phone calls, delivery patterns for Delivery Services, interactive websites, GPS systems, and more. Artificial Intelligence, lies at the core of a vast majority of technological products. Computer Scientist are constantly trying to improve on existing technologies. Computer Scientist are constantly asking Questions; Are you happy with where technology is today? Do you desire greater improvement, more speed, and/or specialization? As we demand more from our products/devices it is critical to understand what AI is, how it works, and ways in which we can improve it. The purpose of this lesson will be to introduce you to AI. This will be attempted by first explaining what AI and how it can be used to enhance technological products/devices. This lesson will briefly review the improvements of AI over the years and the direction it is heading. We will also delve further into the role of AI in video games, specifically MicroRT. However, it is important to keep in mind the vast services and products that rely on AI. And then start to think from an algorithmic view, how does it do that?

There are several goals of the lessons in this activity. The first is to familiarize you with the concept of AI and while doing so, differentiate traditional AI from gaming AI. A second goal is to familiarize you with the basic design of a Real Time Strategy (RTS) game from the context of an Object Oriented Design. A third goal is to reinforce your skills in algorithm development by creating algorithms for basic movement and strategy in a RTS game. A final goal is to allow you to have fun in the process.

MicroRTS is a real time game in which both players act simultaneously with no time to stop and contemplate their next move, unlike a game like chess, in their attempt to win the game. MicroRTS is a research project that can be played by a user against the computer or the computer can play against itself using different/selected AI strategies. MicroRTS can be setup learn from its mistakes and improve its play during the next game it play. However these advance things are beyond the scope of this lesson. The objective in using MicroRTS in this lesson is to understand the basic game play and features of MicroRTS as well as evaluate the role of AI in the game’s development. Students will try to understand the role of different algorithms in controlling built in coded AI features and functions. In groups, students will discuss and evaluate the limitations of algorithms. In the context of a RTS game, an algorithm can be described as a step by step process that can prepare for constant changes in the playing area and new scenarios ahead of time.

There will be a few prerequisites before attempting the activities in this lesson. Before attempting this exercise, students should read the chapter on Artificial Intelligence, Object Oriented Design in a basic RTS game, Reinforcement Learning. Additionally, students should read the brief user’s manual on playing MicroRTS. For students that intend to use the schools resources to play MicroRTS, the game will be already be installed and configured for you. Students that want to play the game on their personal computers should additionally read the installation and configuration guide that is provided with the collection of documents for this lesson. You should have already taken the prerequisite quiz for the lesson. A either a supplement or substitute for the written user’s manual, student may watch the video describing how to play MicroRTS.

**Learning Objectives**

Students should be able to describe the difference between traditional artificial intelligence and artificial intelligence used in gaming agents

Students should be able to describe an algorithm for a game unit’s attack pattern or game unit’s roaming pattern or building placement pattern

Students should be able to describe the fundamental behavior of a real time strategy game.

Students should be able to inspect code and identify and describe the use of polymorphism, inheritance and composition in the code of MicroRTS

Students should be able to describe an algorithm using reinforcement learning

r

**Resources:**

Below are a list of resources included with this modified extension of the MicroRTS project. They can be obtained from the following Website: <https://github.com/nelsoncra/Project1>

* Original Source files for MicroRTS
* Game Background - https://code.google.com/p/microrts/
* MicroRTS Handout – basic game description and features
* Chapter on AI, Machine Learning and Real Time Strategy Game Design
* Game Installation and Configuration Guide
* Videos: Watson and Deep Blue (you tube)
* Online Articles - Wired Magazine
* AI Pre Lesson Questionnaire
* mircoRTS Game Play Questionnaire
* AI Post lesson Questionnaire
* Human Agent verses Computer Agent as a NetBeans Project
* Human Agent verses Computer Agent as an Eclipse Project

**Software**

The native version of MicroRTS is a research project that permits one to experiment with a variety of AI agent against AI agent contests using machine learning. The native version of MicroRTS also contains a module that permits a human to play against an AI agent of the game. We extracted this module and added the additional code to run the module as a main method in a Java project. This modified version of the application that plays a human agent against an AI agent can be run simply by clicking run. We created two modified versions of MicroRTS. One as an Eclipse project and the other as a NetBeans Project.

The modified extended versions of the MicroRTS project can be obtained from the GIT Repository at: <https://github.com/nelsoncra/Project1>

**Eclipse**, **NetBeans**, or comparable IDE – currently the only tested IDE’s are Eclipse and NetBeans. NetBeans is an Integrated Development Environment (IDE) In June 2000, NetBeans was made open source by Sun Microsystems, which remained the project sponsor until January 2010 when Sun Microsystems became a subsidiary of Oracle. The Eclipse IDE is an Integrated Development Environment for Java Programmers. The Eclipse Foundation is a community for individuals and organizations who wish to collaborate on commercially-friendly open source software

Two obtain these IDEs visit the following Websites

Website: [https://www.**eclipse**.org](https://www.eclipse.org)

Website: [https://**netbeans**.org](https://netbeans.org)

**GIT Client** and **GIT Hub** with a **GIT Hub Account** - A GIT Hub account will be useful to anyone wanting to extend this research and continue contributing to the open source community. GIT is a Version Control System Client and GIT Hub is a Version Control System Server. See the Installation and configuration document for further details

To obtain the GIT client software and create a GITHUB shared repository visit the following Websites

Website: [**https://git**-scm.com](https://git-scm.com)

Website: [https://**git**hub.com](https://github.com)

TortoiseSVN - TortoiseSVN is an Apache Subversion (SVN) client, implemented as a Windows shell extension. See the installation and configuration document for further details. The Installation instructions for Tortoise can found at this projects website at the following URL: <https://github.com/nelsoncra/Project1>

You can download TortoiseSVN from: Website: <http://tortoisesvn.net/>

Logo[microrts](https://code.google.com/p/microrts/)

[Minimalist Real-Time Strategy Game designed for AI research](https://code.google.com/p/microrts/)

microRTS is a small implementation of an RTS game, designed by Santiago Ontañón in 2013 to perform AI research

Website: <https://code.google.com/p/microrts/>

The lesson contains an AI Pre-Questionnaire and an AI Post-Questionnaire. The purpose of the two questionnaires is to collect data on the success of using MicroRTS as an engagement tool to increase the learning outcomes. The learning outcomes focus on disseminating knowledge about AI, machine learning and the basic behavior of Real Time Strategy Games. Below is a sample each of the quizzes.

**PART 1**

**AI Pre-Questionnaire**

Please take a minute to answer the following questions.

1. What is an algorithm?
2. Can you create an algorithm without knowing the end goal or scenario in which you are acting? Explain.

(Consider – is the activity an isolated activity? **Example** Trying to make a basketball shot on your own vs. trying to make a shot in a game with four other teammates and a defense on the floor)

1. What is artificial intelligence?
2. What technological devices and products utilize artificial intelligence?
3. Are computers smarter than humans?
4. Which does better processing data?
5. Which does better making decisions in real time?

At the end of the lesson students should receive a post evaluation quiz. The purpose of the quiz is to evaluate the difference in understanding of the outcome topics between taking the pre lesson quiz and the post lesson quiz. Below is an example of a Post Lesson Quiz.

**AI in Gaming Post Evaluation**

1. What are the limitations when using AI in real time game play (ie. microRTS)?
2. How does Q-Learning improve this limitation?
3. Does Q-Learning solve the issue of real time change in its entirety?
4. Are computers smarter than humans? Which does better processing data? Decision making in a real time? (**DID YOUR ANSWERS CHANGE**)
5. **Real World Application** – Provide a scenario in which AI could be thrown off by an unexpected interruption. How could Q-Learning improve the AI’s ability to handle this unexpected interruption?

Depending on the amount of time available in the schedule or the magnitude of reinforcement that you want to provide, the lesson provides several suggested supplemental activities that may be utilized during the overall lesson. Listed below are a few of the suggested supplement activities

**Show Videos of More Recent AI’s in Game Action**

Students will be guided to review some recent activity in game development involving AI. During the research of this project, some suggested are the videos on the Deep Blue Chess Challenge and the Watson Jeopardy Challenge.

Chess**:** Deep Blue - 1997

<https://www.youtube.com/watch?v=WGATtigzCNA>

Jeopardy**:** Watson - 2011

[https://www.youtube.com/watch?v=dr7IxQeXr7g\](https://www.youtube.com/watch?v=dr7IxQeXr7g/)

**Group Collaboration on AI Research Publications**

Students will be separated into small groups and provided with varying articles on various areas in society where AI is used. Below are a few suggested sources to start from

**Articles** – below are links to a few suggested articles to be read as a class for in class discussions. A second type of framework for this activity is to break students up into groups to read and then have the smaller groups bring back their findings to share with the entire class

**Links to Suggested Articles**

<http://www.wired.com/2014/06/the-future-of-computer-intelligence-is-everything-but-artificial/>

<http://www.wired.com/2014/06/ai-healthcare/>

<http://www.bostonglobe.com/news/science/2013/10/06/mit-artificial-intelligence-center-backed-federal-grant-learning-from-infant-brain-research/MdPnWBnGv7KA1N3CVssKEO/story.html>

**A Combination (Article and Video) TIC TAC TOE Activity**

The game of TIC TAC TOE provides a small implementation of an AI agent. The Game of TIC TAC TOE is useful because the options for the AI agent are limited in scope. It should be easier for students to follow the flow of the decision processes in tis limited implementation of an AI agent.

One of the suggestions from the research is to provide Videos & Articles to spark introductory discussion on Artificial Intelligence. After basic discussions on AI class will be ready to start hand out activities and demo’s to further their understanding of AI.

Below are links to a PDF file on an AI algorithm for a TIC TAC TOE game and a Video on AI and TIC TAC TOE. Student are given an assignment to read the article and view the video during the lesson.

**Classroom Demo 1 -** TIC TAC TOE – see link for directions

<http://www.cs4fn.org/teachers/activities/intelligentpaper/intelligentpaper.pdf>

**Student Activity 1** – TIC TAC TOE – Creating an AI for when you go second in a TIC TAC TOE game

<http://www.cs4fn.org/programming/noughtscrosses/>

[After completing Activity 1, the class will discuss thoughts about the experiment as well as their answers to the Pre AI Questionnaire.](http://www.cs4fn.org/programming/noughtscrosses/)

Students will be given to study algorithm development in terms of AI game development.

**Describe systems/games we have discussed so far:**

**Watson Video** – filter through all gathered data and map connections.

**Deep Blue** – Let opponent make move then re-evaluate best alternative before making its own move.

**Tic-Tac-Toe** – Strategy created prior to playing game and before knowing what your opponent will do. Does not re-evaluate strategy after each subsequent move by one’s opponent.

Programming Assignment

In a course that is a programming course, students will receive a programming assignment to implement a decision processes to play a game of TIC TAC TOE with an AI agent. In an introductory course, the application will be limited to the logical processes in a TIC TAC TOE game. Emphasis will be placed on the placement decision of the AI agent, not the creation of a complete graphical game.

The lesson is designed to work across a variety of Computer Science and Computer Technology courses. A course where implementing computer programs from algorithm design can add additional tasks to the lesson; such as creating a functional framework of a TIC TAC TOE game as a programming assignment.

Additionally, varying other types of algorithms may be utilized such as minesweeper or Tetris.

**Playing MicroRTS**

The primary activity of this research and lesson is using MicroRTS as an engagement tool to cultivate higher interest and knowledge in AI, Machine Learning and Real Time Strategy Game Design. For this activity, teacher will introduce students to MicroRTS. They are provided a handout outlining the basic features and instructions for playing MicroRTS. As an alternative or supplement, students will be proved access to short five minute Video Tutorials explaining unit resource gathering, building construction, unit manufacturing, and military unit operations. Within the context of playing microRTS, student learning is guided to focusing on how MicroRTS implements artificial intelligence. While learning about the game play and interacting with and against the different types of AI strategies hard coded into MicroRTS, students will respond to the assignment activities listed below.

**In Game Evaluation**

1. Create a chart identifying and describing the different units on the plating area, their actions, and the roles that the units play in the game?
2. Describe an algorithm (in words) that would explain how a Unit operates.
3. Describe an algorithm (in words) that would explaining the randomness of building construction on the laying area by the AI agent.
4. Describe the behavior of a Real Time Strategy Game. By behavior, what type of functionality or actions should a basic real time strategy game be able to perform?

**Advance Topics while Playing MicroRTS**

In a course where students have introductory level skills in programming, in particular exposure to the Java programming language, a few more additional steps can be added to the MicroRTS activity. During the game play session, students are guided to comment specific method calls and uncomment other method calls. This will be the act of toggling off one type of AI agent strategy and Toggling on a different AI agent strategy to play against. The four AI agent strategies hard coded in MicroRTS are:

* PassiveAI()
* ContinuingNaiveMCTS()
* RandomBiasedAI()
* LightRush()

The human player assumes the role in the MouseController() method as the AI agent that reference variable ai1 will point to. The AI agent that the computer will assume is assigned to reference variable ai2. In this advanced step of the activity, students will be able to comment the method calls that they do not want to execute and remove the comment symbols from the methods that they would like to execute. This can be examined in the snippet of code removed from MicroRTS below. You can examine that reference variable ai1 of AI type is instantiated as a MouseController object of object. It can be clearly seen that the MouseController object inherits from the super type object AI. This implementation of polymorphism can be used to illustrate inheritance and the use of polymorphism. The instructor can point out that the AI data type used to declare the reference variable ai1 is a parent of the object MouseController. The instructor can point out that this practice is used to maintain the stability of applications, so that they do not break when the application is enhanced.

**Example Code Taken MicroRTS outlining the explanation above:**

One can see from the code that reference variable ai1 is instantiated using the constructor from the MouseController object. It can also be seen the instantiation statements for PassiveAI, RandomBiasedAI, and for the LightRush(UnitTypeTable.utt objects are all commented out of the code. The AI strategy that will be contained in the ContinuingNaiveMCTS object during instantiated of the ContinuingNaiveMCTS object.

AI ai1 = new MouseController(w);

// AI ai2 = new PassiveAI();

// AI ai2 = new RandomBiasedAI();

// AI ai2 = new LightRush(UnitTypeTable.utt, new AStarPathFinding());

AI ai2 = new ContinuingNaiveMCTS(PERIOD, -1, 100, 20, 0.33f, 0.0f, 0.75f, new RandomBiasedAI(), new SimpleEvaluationFunction());

During this point, students are guided to observe Java code in the main method of the MicroRTS application. Through this, students can be shown working example of modular development, polymorphism, inheritance and composition in a professionally developed artificial intelligence game.

The commenting and uncommenting of code in the application will permit students to toggle on and toggle off different artificial intelligent strategies that have been hard coded into the MicroRTS game. This will permit the student to play a game against several different AI Strategies. They will be able to record which AI’s agents that they were most successful and least successful against. Afterwards, they should be in a better position to try to explain the different algorithms controlling each of the AI Strategies.

**Reinforcement Learning**

Student will briefly discuss some advance concepts in AI game development. Students are asked to answer several questions regarding machine learning in the context of a real time strategy game. What about games or scenarios where multiple players are acting simultaneously? As a programmer, if you are unaware of what your opponent might do at any given moment, how do you handle this? How can we improve our AI agent based on successes and failures in a previous game?

During this segment of the lesson, one can introduce the concept of Reinforcement Learning to the class. At this point the sophisticated algorithm of Q learning can be examined. Prior to the examination of Q learning one may consider a simpler process. The link below point to a document the details a less complex scenario of Reinforcement Learning using game named Hexapawn.

**Q Learning**

**Classroom Demo –** Asimplistic demonstration of machine learning

During the closing activities of this introductory lesson, students can be introduced to a fundamental algorithm in Reinforcement Learning. Such an algorithm is described in the document located at the link below

<http://www.cs4fn.org/teachers/activities/sweetcomputer/sweetcomputer.pdf>

Here is a simple review of Q learning. On its most basic level Q-Learning relies on two sets of parameters: an agent’s state and the agent’s corresponding actions. By simulating different agent states and subsequent actions over a large subset, a Q-Learning algorithm will gather data on the results from different actions by an actor in the same state. Each time the game is run, a state is created and initialized with values including the success or failure of that state. The state is stored in a table representing the history of successes and failures reached after each state. When a game is replayed and the AI agent is faced with a state that matches a state played in the past, and if the original action was successful, that is the action that will be played by the AI agent. If the action was not successful, a new arbitrary action will be played and new data is gathered on the success or failure of that action. This will continue until a reliable amount of runs provides the optimal action for each given state. Positive actions will be assigned a higher likelihood of being repeated while negative actions will see a decrease in their likelihood of being repeated. The correct action might not be made the first time through, but, over time the AI will have an ample data set to determine the best move given different situations. In this respect, the computer is learning. Obviously, the greater number of actions, the more time this learning will require. The fewer the choices as in the TIC TAC TOE game introduced at the beginning of the lesson, the quicker this learning can be developed

**A Reflection Document**

At the end of the lesson, students are required to submit a document outlining a reflection on the successes and failures in the activities assigned during the lesson. For example, students may be asked about the algorithm created prior to having a better understanding of previous contributions to the technology. This exercise is an attempt to explain to students the significance of understanding previous contributions to the problem before attempting their own solution.

**Group Collaboration Roles**

There are activities in this lesson that can and should be completed collaboratively by students. Activities such as algorithm development, design reviews, code inspections, and topic research. Within the context of this lesson’s algorithm development; the activities of attack algorithms, roaming algorithms, and algorithms describing the randomized placement of building structures can be performed collaboratively. Additionally, Research activities could be completed collaboratively. The research and discussions of AI uses in current technology, the research on a design pattern for a real time strategy game, and fundamental concepts in traditional AI and gaming AI can be performed collaboratively.

Within collaboration activities, group members at a minimum should assume some of the following roles outline below, based on the needs of the activity. Some group members may assume multiple roles depending on the configuration, capabilities and limitations of the group members. The ultimate goal is to facilitate the collaborative sharing of knowledge and to strengthen the group members who lack proficiency in the specified categories. Parts of this model is taken form the POGIL model of facilitation of roles in a collaborative learning environment.

**Group Manager**

Each group should have a manager that ensures that the members are fulfilling their assigned roles. There are also ensuring that the group members are completing their tasks on time and ensuring that all group members are participating in the group’s activities and understanding the concepts outlined in the activity.

**Group Recorder**

Each group should have a person assuming the role of group recorder. The group recorder should record the important content in the group’s collaborative discussions on AI, machine learning and Real Time Strategy Game design. The group’s recorder should provide documented insights to the aspirations and obstacles in the group’s journey to completing the assigned task outlined in the lesson presented to the group.

**Group Spokes Person**

The person assuming the role of group spokesperson should provide oral reports to the class on the progress and current state of the group’s status. This should include initial requirements of the AI activity before a project and the presenting the group’s post mortem report on the success and failures of the group’s participation in the activities and the overall lesson.

**Strategy Analyst**

The roles of the Strategy Analyst is to report on the group’s collaborative effort in the leaning process towards artificial intelligence, reinforcement learning and real time game design and learning.

**Algorithm Inspection**

The person overseeing this role should coordinate the inspection of the algorithms to address the problem of the activity of the movable unit in the playing field of the game, resource gathering, and randomized placement of AI agent facilities. In traditional software development, this world be the person overseeing the design review.

**Code Inspector**

The person overseeing this role will lead the inspection of code and point out incidents where the code meets or does not meet standards in object oriented development and software engineering practices.

**Design Pattern Inspector**

This person oversees the high level design of Classes their properties and behaviors. This person examine the class diagrams and assures that the functionality and feature described matches with the requirements of the outlines in specifications.