Expert System in a Gaming Environment

Introduction

Following on from the research this section will explain how an expert system function works within the gaming environment. The goal of this expert system is to select which weapon to use during game play, of which this depends on a number of different variables that are changing every frame, therefore the expert system will make a decision.

This page will explain through the following:

- Analysis of the gaming environment, listing variables, expert knowledge and static data.
- Assumptions made in the creation of the expert system e.g. confidence factor, forward chaining
- How the facts are derived from game play itself.
- How the inference engine decides which rules to run using data from working memory and knowledge base.
- Weapon Selection example using the expert system

Depicting the diagram shown in Types of gaming AI under expert system, there are a few requirements into developing a weapon selection AI system, and they are as follows:

- From game play what are the input variables, or changing environmental data, i.e. target range, target type, weapon range.
- What is the system going to output and why? i.e. type of weapon, why choose a specific weapon, what are the weapon characteristics.

The answers to these questions are derived from the game expert, in this case the makers of this coursework. Therefore the known facts are weapon characteristics (highlighted in the table 1 below), format of target range weapon range, and which weapon is best for achieving its goal i.e. to eradicate the target.

The main reason for developing an AI is to satisfy the objectives and as the gaming environment constantly changes, the bot will constantly have to re-evaluate the situation much like a human player. Creation of an AI will improve game play and provide a challenge with the bot making its own decisions on weapon selection.

Analysis of Gaming Environment

The gaming parameters are shown below that are derived from the game experts.

Predefined Static Variables and Expert knowledge

- Pulse weapon range is 40m.
- Mini gun is effective against all targets, more so at close range.
- Missile weapons are effective against player, more so at long range.
- Bots field of view or target range is 300m.

Facts above are stored into the Knowledge base of the AI system.

Predefined Weapon Data

<table>
<thead>
<tr>
<th>Target (A) / Weapons (B)</th>
<th>B</th>
<th>B</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missile</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mini gun Weapon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Defensive Pulse</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nuke Missile</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Missile</td>
<td>Useless</td>
<td>Effective</td>
<td>Effective</td>
</tr>
<tr>
<td>Player</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Missile</td>
<td>Useless</td>
<td>Effective</td>
<td>Useless</td>
</tr>
<tr>
<td>Player</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Effective</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1 shows weapon characteristic / effectiveness

For example, Missile weapon is useless against Missiles.

Assumptions

Confidence Factor
The confidence factor is a probability given to problem data that the user has entered into the working memory. For example the target that is within target range has a confidence factor of 75%. The 75% value is an independent value from the artificial intelligence and is used in the final result once the inference engine has run. If another fact was added that target was a missile, the confidence factor was 80%.

An example use of the expert system would be if target was in range and a missile, the pulse weapon would be selected for defence. The system would select the pulse weapon put put a confidence factor based upon the facts it used. The Min/Max protocol could be used in this instance. The Min protocol uses the lowest % combination of logic values.

Therefore the pulse weapon was chosen and given a confidence factor of 75% of this facts being correct. The decision that was made with a definite answer being given, however the confidence factor is a rating of the answer being correct given the fact each giving a level of surety.

This game will not use the confidence factor as all of the game parameters are directly read from the computer and would be 100% certain that they are correct. Therefore there is little point using this system in a gaming environment.

**Forward Chaining**

The inference engine will use Forward Chaining this method uses the derived data or facts stored in the working memory and using rules to determine a goal. Backward Chaining is the reverse of this process. The inference engine runs rules chosen by a controlling strategy until the premise is true and thus a consequent is formed using expert data from the knowledge base. The Inference engine will cycle until all relevant rules have fired or it when it has come to a conclusion. In the gaming environment the cycle is run every frame as parameters change every frame.

The assumptions or realities of the games are as follows:

- The confidence factor is not required and therefore not implemented.
- Forward chaining is used in the inference engine.

**Derived Facts**

The expert systems needs to know how to interpret the targets data, such as is it within weapons range. To determine this the knowledge base contains a set of rules (questions).

A rule is most likely to be implemented using an If statement. The if statement takes the form of IF (R logical operator S) then T.

The bracketed section (R logical operator S) is known as the premise or proposition. The T is the result of the premise being true, this is known as the consequent or result. Therefore this boolean expression can derive certain facts that are then stored into the working memory.

The working memory in the gaming environment is a variable in RAM (Random access memory) that is overwritten every time the function is called or hence when the AI system runs a new loop of the knowledge base.

**Decision or Inference Engine**

An inference engine derives conclusions from the knowledge base and working memory. The inference engine is the brain behind the expert system, they are also known as a special type of reasoning engine.

The inference engine is a set of special rules that determines the output of the system. It also determines which rules are relevant for the chosen task at hand and apply only those rules. This inference engine will use the forward chaining method.

The controlling strategy only runs a certain set of rules that are only applicable to the data available this is also known as conflict resolution. A good controlling strategy can also help to increase the speed of a large expert system.

For the purpose of this coursework the controlling strategy is very simple, utilizing a single if statement. The rule set of the inference engine determines the weapon selection based upon facts already obtained in the working memory. The system is entirely boolean and can be represented in logic gates form (figure 2 onwards).

**Implemented Example**

The flow diagram below shows how the expert system is executed every frame during game play. The gaming environment changes every frame therefore the AI system will cycle every frame.

1. The game parameters are questioned using fact deriving if statements (shown below) which are then placed into the working memory.

The first section is done via a series of if statements, the pseudo code is below:

- IF target within target radius then set flag = 1
- IF target = missile then set target = missile
- IF target = nuke then set target = nuke
- IF target = player then set target = player
- IF target within range of pulse gun set flag = 1
The purpose of the above code is to determine facts from the gaming environment, and place them into the working memory. This is shown in the top half of the diagram below.

The Inference Engine

The inference engine as described previously firstly decides what rules to execute, this is the simple controlling strategy described before. The pseudo code for this is shown below.

2. The next stage is to construct the controlling strategy if statement clause shown below.
   - IF target is within target range execute weapon selection rules.

Once this control rule has been executed an inner series of if statements will infer a result from the facts in the working memory and knowledge base.

3. Using the forward chaining method, a series of if statements are executed that sorts through the data in the working memory and knowledge base to find a goal. The goal in this case is weapon selection. The pseudo code is shown below:
   - IF target is a missile and at close range select DEFENSIVE PULSE WEAPON
   - IF target is a missile and not at close range select MINI GUN
   - IF target is a nuke missile select MINI GUN
   - IF target is a player and at close range select MINI GUN
   - IF target is a player and not at close range select MISSILE

Therefore the goal selected from the data is either a MISSILE, MINI GUN or a DEFENSIVE PULSE WEAPON. The if statement or inference engine may be understood as boolean logic depicted in the following figures. Boolean logic is used as expert systems have true or false answers, especially in this gaming environment.
5. Once the logic has determined a goal the output is fed back into the game environment, and the process begins again but with new game parameters.

The code can be accessed here.

Testing

Expected Result

The expected result or behaviour of this expert system totals 5 outcomes equivalent to the number of rules in the rule set. The bot will decide which behaviour it will execute depending on different combinations of input variables or premise.

1st Behaviour

The bot should change its behaviour/weapon once it detects a missile/s and activate the defensive pulse to push incoming missile/s away once within close range.

Commentary

- The bot has identified a list of targets that are within its range.
- The bot then enters the inference engine and executes the control rule. This confirms that at least a target is within range.
- The inference engine now knows the fact that a target is within weapons range. The engine also knows what type of enemy from the target selection algorithm.
- The bot then cycles through the rule set and identifies in the first instance, incoming missiles that are outside of the defensive pulse weapon.
- The bot then proceeds to destroy each missile in turn with the mini gun, following rules in "The inference engine" section.
- As the bot changes target after destroying several missiles to missiles entering the range of the defensive pulse weapon. The inference engine has a new set of criteria, of which the best weapon was expertly chosen to be the defensive pulse. The defensive pulse weapon is selected and activated.
- The pulse pushes the missiles away and out of range of the pulse weapons effectiveness.
- The inference engine is run again and finds the same criteria as at the top of the commentary, and destroys the missiles with a mini gun, thus preserving its life.

Evaluation of Behaviour 1

The expert rules react as expected, it clearly shows a forward chaining methodology using a controlling strategy. An analysis of the set of rules used in this demonstration suggest self preservation of the bot is more important than that of defeating an aggressor (player). This would appear to suggest or replicate some form of intelligence.
2nd Behaviour

The bot should change its behaviour/weapon to mini gun to destroy the incoming missile/s when within target range to destroy the missile/s.

Commentary

- The bot has identified targets that are within its range.
- The bot then enters the inference engine and executes the control rule. This confirms that target are within range.
- The inference engine now has stored in working memory the following facts.
  - Target is a missile
  - Target is not short range
  - The bot then cycles through the rule set and identifies in the first instance, incoming missiles that are outside the range of the defensive pulse weapon.
  - The bot then proceeds to destroy each missile in turn with the mini gun as the selected weapon.

3rd Behaviour

The bot should change its behaviour/weapon to mini gun when a nuke missile is within range and destroy the nuke.

Commentary

- The bot is currently engaged in combat by using the mini gun to shoot down incoming missile.
- The player then fires a special nuclear missile that has a red fire trail.
  - During game play the target selection algorithm is run every 2 seconds, during this interim period the bot continues to fire at the standard missiles.
  - The target selection algorithm then switches the nuke missile to highest priority. The bot then tracks the fast moving nuke (shown in the video).
  - The expert system is then run and derives the following facts from game play.
    - Target is Nuke
    - Target is within range
  - The inference engine takes these facts and executes the rule "destroy nuke with mini gun"
  - The weapon choice is therefore a mini gun until the nuke is destroyed.

4th Behaviour

The bot should change its behaviour/weapon to mini gun when a player is detected within close range and destroy the player.

Commentary

- The bot identified the primary target to be the player.
- The bot then enters the inference engine and executes the control rule. This confirms that the player is within range.
- The bot then cycles through the rule set and identifies the best weapon to use given the derived facts in the working memory. Therefore missiles are launched at the player using the targeting algorithm. Notice that the targeting algorithm makes assumptions of where the player is going to be, hence the missiles are launched into the path of the player and explode causing blast effect damage.
- The bot then continues in this fashion until the game parameters change. As the player enters the close range criteria the inference engine will include this fact as it scans through the rules.
- The inference engine consequently fires a rule that switches the weapon to mini gun and causes a large amount of damage to the player.
- As the player retreats the expert system then differs from human behaviour and switches weapon to missiles as the player is no longer within close range.
This allows the player to escape, or so they thought. A missile impacts on the right side destroying the player in quick succession.

**5th Behaviour**

The bot should change its behaviour/weapon to missiles when a player is detected within target range and destroy it.

**Commentary**

- The bot identified the primary target to be the player.
  - The bot then enters the inference engine and executes the control rule. This confirms that the player is within range.
  - The bot then cycles through the rule set and identifies the best weapon to use given the derived facts in the working memory. Therefore missiles are launched at the player using the targeting algorithm. Notice that the targeting algorithm makes assumptions of where the player is going to be, hence the missiles are launched into the path of the player and explode causing blast effect damage.
  - The bot continues in this fashion until the player is destroyed.

**Evaluation of behaviour 5**

The expert system detects the player and fires missiles until other factors such as distance or other elements enter into the game play environment. This is consistent behaviour and very predictable, exactly what an expert system in gaming should produce.

**Unexpected strategy development**

During the testing phase, it was noticed that a combination of rules actually performed an unexpected strategy from the bot. The bot can use the defensive pulse to push the player out of range and continue firing at the missile threat. (This is shown in video).

**Evaluation of unexpected strategy development**

This was an unexpected strategy and shows if the rules are carefully devised, the bot can exhibit signs of intelligent behaviour.

**Conclusion**

An expert system can be used to develop a very simple form of artificial intelligence into a game engine. As proven in the above simulation videos or in actual game play (download the game here).

**Advantages of an Expert system**

The advantage of any expert system is to provide consistency, the bot will use specified weapons for the most appropriate targets, this in turn means the bot will be effective in fulfilling its objectives.

The expert system always decided on a course of action (dependable), there is no middle state, however whether it makes the correct decision depends on the rules and available data being correct.

In this case although the working memory is very simple it has the ability to store a large amount of information and varying scenarios, therefore making the correct weapon selection with a developed rule set almost perfect.
The other advantage of the expert system is that it can be used for a number of users/bots. Therefore expanding the game to have more than one bot using the same expert system is possible.

When designing rules it is often impossible to be absolutely sure how all the rules will effect each scenario. As shown in the videos above it is possible an unexpected result or consequent in a gaming environment.

**Disadvantage of an Expert System**

During game play one disadvantage is predictability from the players point of view. The player can pick out weakness and flaws in the bots tactics if the rule set has not been fully developed or only limited data is provided to the working memory.

Gaming engines generally do not use a sole expert system due to there inherent limitations such as appearing predictable and a lack of common sense shown in behaviour 4. Therefore it is more likely that the expert system is used as part of complex fuzzy/neural system. This leads onto the combination expert/fuzzy system.

When updating the expert system, especially with a new expert inputting the rule set, one must be very careful not to assume a fact from previously derived facts. Therefore when inputting new rule set one method to avoid such an occurrence is to use conflict resolution or a controlling strategy.

**Overall Expert system conclusion**

Expert systems have a part to play in games, however there will become a point where the expert system would become so large it would be impossible to manage in real time. Therefore for RTS games neural networks are used to define a smaller and more likely rule set and behaviour for this type of game.

Expert systems are widely used to assist and not replace experts in the field, i.e NHS. They are also used to determine a fact or goal using the a series of definitive questions (forward chaining) or determine how a goal was achieved (backward chaining) by listing the symptoms, given the NHS example.

**Key points to an Expert system**

The example here should be enough to give you a good concept of the following.

- Knowledge Base
- Rule Set
- Working Memory
- Inference Engine
- Inference Engine rule set
- Forward Chaining
- Controlling Strategy

Other points that where touched on are:

- Confidence Factor
- Min/Max and probabilistic value
- Backward Chaining

Other Expert systems examples can be found [here](#) and can be linked to a whole host of problems such as [Lock control](#).

**Further Enhancements or Game Development**

There are 2 possible further improvements to this expert system in the BiS game.

1. Increase the rule set and accept another gaming variable to give the game more variety.
2. Convolve the expert system with the fuzzy system to develop a combinational artificial intelligence system.

The latter will have a new rule set based upon outcomes of fuzzy logic and have attached to each outcome a probabilistic value. Depending on this level an expert system could then be devised to incorporate this into gaming levels of difficulty.