Bottom-Up Strategy Planning Model by applying Fuzzy Integral in RTS Game

Peter H. F. Ng, Y. J. Li, H. B. Wang, Y. Li and Simon C. K. Shiu
Department of Computing, The Hong Kong Polytechnic University
PQ806, Mong Man Wai Building, HKPU, Hung Hom, Hong Kong
Email: {cshfng, cshbwang, csyjli, csckshiu}@comp.polyu.edu.hk

Abstract—Game strategy planning is a difficult task in Real Time Strategy (RTS) game AI development. Tree searching technique has been one of the common approaches. However, the increasing use of complicated game rules leads to tree models that are huge and complex, and sometimes even unmanageable. Traversing and modification of the tree structure becomes a time consuming and inflexible task. Our research tries to avoid this top-down strategy planning method and propose a bottom-up approach by apply Fuzzy Integral in extracting useful game strategies from data. We also developed a new fuzzy integral for game decision-making. Compared with the traditional Choquet Integral, we achieved a better result using real Warcraft III battle data and the detail is reported in this paper.

Index Terms—RTS game, Strategy Planning, Feature Interaction, Fuzzy Measure, Fuzzy Integral, Genetic Algorithm, Warcraft III

I. INTRODUCTION

The fundamental game play of a typical RTS game is allocating resources to build an army, design units’ combination and defeat the enemy by killing the opponent units. Control of RTS game can be classified into two types. The first type is macro management which is also named game strategy. It involves resource gathering, base building and technology upgrade. The second type is micro management, i.e., tactics, involves planning of the unit movement and control. Multi-agents Potential Fields of Hagelback[1] and Johan[2] shows the importance of game strategies and tactics development and events such as Open Real Time Strategies (ORTS) AI competitions were also held for many years. Our research focuses on game strategy planning and development. Many RTS game designer suggests a balanced army which involves many unit combinations is a good choice. However, such approach wastes the resource in technology upgrade and is unable to gain massive destroy power in a short time. Professional players seldom use this strategy in real game competition [3]. They developed “unbalanced” army units, which have excellent killing power to certain kinds of enemy units, but they themselves may be easily killed by another special kind of enemy unit too. Thus, many researchers such as Buro [4] and others developed interests in RTS game AI.

II. LITERATURE REVIEW

Most of the researchers applied a top-down strategy planning approach [5][6][7]. They built up a tree structure and the AI compares situations of the game and adopts the strategy that has the closest match. This model becomes inflexible and difficult to implement if the conditions of each node are not well determined. RTS game consists of hundreds of buildings and units. Each of them performs different function in the battle and requires different resources to create. They have different abilities on damaging others or defending themselves. Thus, top-down approach becomes intractable. Furthermore, the combined power of different units is usually “non-linear”, for example a knight ridding on a horse is definitely much more powerful than an individual standalone solider plus a horse near him. Some quantitative interaction method such as analysis of variance (ANOVA) test is available to use to describe relationships among groups of data. However, they assumed that qualitative interactions can be formulated as linear weighted sum and/or statistical variance which is usually not the case in RTS games. Therefore, we adopt Fuzzy Integral to tackle and represent these interactions.

Choquet Integral (CI) has been a general tool for dealing with multiple criteria decision making and is able to model the interactions among different criteria [8]. Suppose a fuzzy measure \( \mu : P(X) \rightarrow [0,1] \) and \( \mu(\emptyset) = 0 \), definition of CI is shown as following.

\[
(c) \int f(x) \circ \mu(x) = \sum_{r=0}^{\infty} (a_r - a_{r-1}) \cdot \mu(x \mid f(x) \geq a_r)
\]

where \( a_r = 0 \) and \( a_0 \leq a_1 \leq a_2 \leq ... \leq a_r \)

Murofushi [9] and Kwon [10] have proven that CI is meaningful when the fuzzy measure is non-monotonic. This property makes it very suitable for RTS game, for example a few types of “suitable” unit combination have much value than the combination of many “wrong” types.

III. METHODOLOGY

A. Strategy Definition

Strategy Case = {Goal, Situation, Scores}

We define the goal of a strategy is the creation of a suitable army, i.e., a suitable unit combination with certain proportion, e.g., 10% peasants, 50% footman and 40% rifleman. Situation is used to cluster cases in our model. We select player race and enemy units as the criteria in our experiment. Depends on the number of training data and game play design, other information could also be selected as the cluster criteria. Scores is the performance evaluation and is used to train up the fuzzy measure. The higher the scores represent the better the unit
combination. These scores are provided by the testing RTS games that we collected from the Web, and through our students.

Ontanon [11] and Sharma [12] applied similar approach to index the player actions. They used building or architectures as an index to organize the game actions. In our study, we try to use the possible unit, i.e.: goal that can be produced by player as an index. Game actions are much simplified for the purpose of reducing the complexity of each case description, thus allowing a more focused comparison in this bottom-up strategy planning model.

B. Data Collection and Preprocessing

RTS game involves complicated game play and rules. It is fruitful for decision making research. We selected Warcraft III expansion, The Frozen Throne (WIII: TFT) which is a well known and popular RTS game. It is regarded as the best RTS game on the market because of its game plays. WIII: TFT is famous for its various unit combination and strategy. Over 7 millions were sold and there are concurrently over 200,000 players playing 6,000 battles on line. Hundred and thousand replays can be downloaded from the Internet. In our research, we select professional one verse one competition. This can avoid the effects of micro management and errors from player skill levels. We extracted the data from encrypted battle replay. The data are (1) unit production statistics as strategy goal, (2) enemy unit as the situation and (3) scores as the performance evaluation. Then strategy cases are clustered by situations, i.e., player races and enemy units. Five data sets with the largest amount of cases are selected to perform this research.

C. Bottom-Up Strategy Planning Model

Idea of bottom-up strategy planning model is select a suitable goal that is similar to the current situation. Then generate all the corresponding actions to achieve this goal. The flow is shown in Fig.1.

1. All previous cases will be converted into the strategy case base and used to train the fuzzy measures.
2. Pilot strategy goal will be used to start the game.
3. At each certain time step, situation of player and enemy will be extracted and pass to decision making modules.
4. Performance evaluation will be calculated by fuzzy integral, trained fuzzy measures and player units’ combination.
5. Other alternative strategy goals which contain the same situation of enemy and player will also be filtered out and the best N strategy goals will be selected.
6. Decision making modules will evaluate the additional resources, time and performance scores that required for each goal. Weighted sum calculation could be applied here and to perform a vary strategies control. Following is a sample equation

Weighted sum = W₁ * Scores + W₂ * (1/Additional Time) + W₃ * (1/Additional Resource)

By increasing the W₁ value, the selected strategy will have higher chance to win. If we want an aggressive AI or rushing strategy, we can select W₂ as the major consideration.

Fig.1. Bottom-Up Strategy Planning Model

7. Then it will decide to change the strategy goal or continue the current one.
8. Finally, the action generator is responsible to performs all the actions according to the game play, e.g., build the additional buildings in macro-management or defend in micro-management when the performance measure of fuzzy integral is low. Programmer can only focus on this part and program the actions for each kind of unit. Therefore, they do not need to handle strategy planning and thus reducing the complexity of AI programming.
D. Apply Fuzzy Integral in Decision Making

Performance evaluation is the core of our research. It provides a value for the decision making module to identify the current situation, whether to advance and fight against enemy or change to another suitable strategy goal. Unit combination involves interaction and super additive properties, for example, Power of (Footman + Priest) is greater than Power of (Footman) and Power of (Priest). In a normal battle, player is able to select more than 15 different kinds of unit. Therefore, if the performance prediction is designed to handle unit interaction, then there are more than $2^{15}$ weighting combinations. The time pressure of AI in RTS game is very hard. We suggested using non additive Fuzzy Measure and Integral to solve the performance evaluation problem.

Apply Genetic Algorithm to obtain Fuzzy Measure

First, all the training cases are clustered by situation. Each situation will have its own set of fuzzy measure. Genetic Algorithm (GA) is then applied to obtain the non additive fuzzy measure [13][14], i.e.: $\mu(X)$ for $\mu(\text{Unit Combination})$, for calculating unit combinations and its' interactions. GA has the following advantages to perform this job.
1. Random walk approach (probability mechanism in its iterations) of GA algorithm will prevent falling into local minimal value easily.
2. The capability of random quick search is independent of the problem domain.
3. Searching is started from the whole population, so it is more robust. It leads the comparisons between multi-individuals possible.
4. It is enlightened by evaluation function, the searching process becomes simple.
5. It owes the extensibility; it is easier to be combined with other algorithms.
6. It has been proved both practically and theoretically that under some conditions, GA algorithm is always converging to the optimal solution by the probability of 1.

Chromosome Formation

First, a number of chromosomes are generated randomly. Each chromosome represented a fuzzy measure opinion. It is combined by $2^n - 1$ fuzzy measure of $\mu(X)$, i.e., it involves $\mu(x_1)$, $\mu(x_2)$, ..., $\mu(x_1, x_2)$, ..., $\mu(x_1, x_2, ..., x_n)$, each of them represents the fuzzy measure in the sub-set of the unit type. $x_1, x_2, ..., x_n$ represent unit type 1, unit type 2... unit type n and n is the number of unit type. Each fuzzy measure in the sub-set of the unit type is corrected to seven decimal places and then converted into binary form and combined into one chromosome. Population of each generation is set to 50 in our experiment. To simply the fuzzy measure, the unit type that is not produced in all the replays within the same cluster is neglected.

Fuzzy Measure Training in GA

Each chromosome is presented as a fuzzy measure. Unit statistics in each replay will be extracted and processed by function $f$. It is presented as $f(x)$ where x represents a player unit statistic of a replay, i.e.: $f(x)$ = $f(\text{Unit Statistic}_{\text{Replay i, Player}})$. Throughout the function $f$, all the unit statistics in the battle (i.e.: unit(s) of unit type 1... unit(s) of unit type N) are normalized and presented in proportion. $Score_{\text{Warcraft III}}$ is a performance evaluation by Warcraft III and shown as following.

$$Score_{\text{Warcraft III}} = \text{Unit score} + \text{Resource score} + \text{Hero score}$$

(2)

<table>
<thead>
<tr>
<th>ELEMENTS IN $Score_{\text{Warcraft III}}$</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit score</td>
<td>Units Produced, Units Killed, Buildings Produced, Buildings Razed</td>
</tr>
<tr>
<td>Resource score</td>
<td>Gold Mined, Lumber Harvested</td>
</tr>
<tr>
<td>Hero score</td>
<td>Experience Gained</td>
</tr>
</tbody>
</table>

We extracted the $Score_{\text{Warcraft III}}$ of player and enemy in each replay data and defined our score, i.e.: $score(x)$ as the difference between player and enemy.

$$Score(x) = Score_{\text{Warcraft III, player}}(x) - Score_{\text{Warcraft III, enemy}}(y)$$

(3)

The positive $Score(x)$ means the player is willing to win. The negative $Score(x)$, the player is willing to lose. More than 93%
cases obey this phenomenon. The higher the Score\((x)\) represents the higher the chance of winning. Score\((x)\) is then normalized between 0 and 1 in our research. \(\mu(X)\) will be calculated by Fuzzy Integral for each \(f(x)\) one by one. Estimated score will be obtained for each \(f(x)\) and presented as Score\(_{\text{Estimate}}\)\([f(x)]\). Detail steps will be shown in next session. Score\(_{\text{Estimate}}\)\([f(x)]\) will compare with Score\((x)\) from the replay one by one. The result is presented as Score\(_{\text{Different}}\)(x).

Score\(_{\text{Different}}\)(x) = |Score\(_{\text{Estimate}}\)\([f(x)]\) - Score\((x)\)| \hspace{1cm} (4)

All the differences between the real scores and the result of the fuzzy integral in all replays will be summed up and the average is calculated. Then it will become the fitness value of current chromosome. We apply Wang’s fitness value [15] and it is presented as following:

Fitness value = \(\frac{1}{1 + e}\) where \(e = (\sum_{i=1}^{n} \text{score}_{\text{average},i})^\frac{1}{n}\) \hspace{1cm} (5)

**GA Operator**

After fitness calculation for all chromosomes in the same generation, a portion of the existing population will be selected to perform the crossover to form the next generation chromosome. In our research, the chromosome with lower value of fitness will be treated as the fitter chromosome. Roulette wheel selection is used. The fitter chromosome will have a higher chance to be selected; therefore the fitness value can be converged.

Two selected chromosomes will perform crossover operator and generate a new chromosome. One-point crossover has also been applied. A single crossover point on both parents' organism strings is selected. All data beyond that point in either organism string is swapped between the two parent organisms. The resulting organisms are the children. To prevent fall into a local minimum and introducing diversity, mutation operator is used. Mutation rate of this research is 0.05. A probability that a bit in the binary chromosome will be selected and convert into opposite bit.

GA process will be repeated unit the fitness value is stable or exceed the maximum generation. Therefore, a reasonable fuzzy measure \(\mu(X)\) for training replay(s) is obtained. Maximum generation is 100 in our research.

**E. Fuzzy Integral**

We applied CI in such which is similar to Murofushi example [16], resources allocation problem. We select a testing case from our data cluster 1 to illustrate our idea. Given that the number of unit type is 7, i.e., \(n = 7\) and the unit statistic of is \(X = \{x_1 = 0.1, x_2 = 0.3, x_3 = 0.6, x_4 = 0, x_5 = 0, x_6 = 0, x_7 = 0\}\). Then, ascending order sorting is performed in unit type, i.e.: \(a_1 = x_4 = 0\) , \(a_2 = x_5 = 0\) , \(a_3 = x_1 = 0\) , \(a_4 = x_6 = 0\) , \(a_5 = x_3 = 0\) , \(a_6 = x_2 = 0.1\) , \(a_7 = x_7 = 0.3\) if \(a_1 \leq a_2 \leq ... \leq a_7\). Given fuzzy measure that trained by GA as \(\mu(X) = \{\cdots, \mu(x_7) = 0.74, \mu(x_6, x_7) = 0.98, \mu(x_5, x_6, x_7) = 0.71, \cdots\}\).

(c) \(\int f(x) \cdot \mu(X)\)  
\(= \sum_{i=1}^{n} (a_i - a_{i-1}) \cdot \mu\left(x \mid f(x) \geq a_i\right)\)  
\(= \left((a_i - a_{i-1}) \cdot \mu\left(x \mid f(x) \geq a_i\right)\right) + \cdots\)  
\(= 0 + ((a_i - a_{i-1}) \cdot \mu\left(x \mid f(x) \geq a_i\right)) + \cdots\)  
\(= 0.1 \cdot \mu(x_2, x_6) + 0.2 \cdot \mu(x_2, x_5) + 0.3 \cdot \mu(x_2)\)  
\(= 0.1 \times 0.71 + 0.2 \times 0.98 + 0.3 \times 0.74\)  
\(= 0.489\)

The original score that provided by WIII is 0.41. Therefore, the error is 0.079.

CI is useful in many application areas because of its generalized mean operators but there is missing real application in non-monotonic fuzzy measure. Unit combination of RTS is highly non-monotonic and can be proven by the result of trained Fuzzy Measure. Wasting resource to upgrade and unlock too much different kinds of units may have a poor performance result. We also found out that the CI interaction selection is not very suitable in RTS game. CI considers the least produced unit also has important interactions with all other units. This violates one of the main game design ideas of RTS game play, i.e., powerful unit can dominate the performance. Therefore, we develop a new Fuzzy Integral for decision making based on RTS game play’s characteristics. First, develop a new integral, named mean integral (MI) which is stated as following.

\[m \int f(x) \cdot \mu(x) = \sum_{i=1}^{m} x_i \times \left(\frac{1}{m} \sum_{i=1}^{n} \mu(S_i)\right)\]  
where \(x_i \in S_i\) and \(\forall x \in S_i, x \neq 0\), \(n\) is number of unit type, \(m\) is the number of set which include \(x_i\).

MI tries to find out all the interactions that involve the current unit type and take the average. This approach does not need to deal with the problem of interactions selection. MI can reduce the error in performance evaluation of strategies. Compare with CI, it shows a better result in both training and testing. One of the weaknesses of MI is the training time. The training time of MI is about 100 times more than CI. Therefore, we develop another integral, named RTS game Integral (RI).

Fig. 3 Unit production sequence of RTS game battle

Our idea is come from the unit production sequence of RTS game battle. Battle of RTS game will create labor at the very beginning. Then basic unit is created for attack or defense. Throughout the base and technology tree development, different kinds of advance unit will be unlocked. Players will focus on creating the advance units. Proportion of unit
combination will be dominant by advance units. Usually, the higher proportion of unit or more advance unit should be considered having more interactions with other units. Our proposed fuzzy integral focuses on the highest proportion units and calculates its interaction with all other units first and so on. This approach makes the fuzzy measure distributes according to the unit development stage. Unlike CI, the measure of the battle in later stage with larger set of interactions will not affected by the earlier stage with smaller set of interactions. Hence, unit that is not produced in current battle is not counted. This helps to distinguish the common unit combination. The equation of RI is shown as following.

\[
\left(\int f(x) \cdot \mu(X) = \sum_{i=1}^{n} a_{i} \cdot \mu(\{x | 0 < f(x) \leq a_{i}\}) \right) \\
\text{where } a_{i} \geq \ldots \geq a_{n} \geq a_{1} > 0
\]

If we use RI for calculation of fuzzy integral value instead of CI, the same example previously shown becomes: Given the number of unit type is 7, i.e.: \(n = 7\) and the unit statistic of is \(X = \{x_{1} = 0.1, x_{2} = 0.3, x_{3} = 0.6, x_{4} = 0, x_{5} = 0, x_{6} = 0, x_{7} = 0\}\). Then, descending order sorting is performed in unit type, i.e., \(a_{1} = x_{1} = 0.6\), \(a_{2} = x_{2} = 0.3\), \(a_{3} = x_{4} = 0.1\), where \(a_{i} \geq a_{i+1} \geq a_{n} \). As the fuzzy measure is trained by another different integral, i.e.: RI, the value of fuzzy measure would be totally different and \(\mu(X) = \{..., \mu(x_{1}) = 0.54, \mu(x_{1}, x_{1}) = 0.42, \mu(x_{1}, x_{2}, x_{1}) = 0.43,...\}\)

\[
\left(\int f(x) \cdot \mu(X) \right) \\
= \sum_{i=1}^{n} \mu(\{x | 0 < f(x) \leq a_{i}\}) \\
= a_{1}(\mu(a_{1}, a_{1}, a_{1})) + a_{2}(\mu(a_{1}, a_{1})) + a_{3}(\mu(a_{1})) \\
= x_{1}(\mu(x_{1}, x_{1}, x_{1})) + x_{2}(\mu(x_{2}, x_{2})) + x_{3}(\mu(x_{3})) \\
= 0.6 \times 0.43 + 0.3 \times 0.42 + 0.1 \times 0.54 \\
= 0.438
\]

The original score that given by WIII is 0.41 with the error 0.028. Compare with the error of CI, 0.079, RI gives a better performance prediction in this case.

IV. EXPERIMENTAL RESULT

2,649 replay data in professional completions are collected and useful information is extracted. According to the situation of enemy unit and player race, they are clustered into five data sets. Not all the unit combinations are shown in the data set. In Table II, No. of Combination presents the number of unit combinations that have been used by the player in the data set.

We perform our experiment using Matlab. 70% of cases in each set are used for training and 30% of cases are used for testing. Training error, testing error and training time of our new fuzzy integral with comparison with CI are summarized in following tables.
Mean Integral presents a better result in training and testing as it considers more options in interactions selection. The performance is stable in all five set of data. The average of error is 0.129. It is good enough to be used to determine in the current situation whether the player should advance or not. Currently, the main weakness of MI is time consuming. It is 40 to 180 times more than CI. It is because interaction selection of CI is direct but MI needs to search for all interactions to find out the interactions selection in RI is suitable to Warcraft III game play. Probably, it may explain why MI is similar as RI. The average is 0.139 which is only 0.01 additional searching is required. The training and testing error is directly addressing the required combination. Thus, no additional searching is multiplication of number of unit type, thus the number of additional searching is multiplication of number of unit type, training case, population and generation.

Training time of RI remains the same as CI as it is also directly addressing the required combination. Thus, no additional searching is required. The training and testing error is similar as MI. The average is 0.139 which is only 0.01 different from each other. Probably, it may explain why interactions selection in RI is suitable to Warcraft III game play.

V. CONCLUSION

A bottom-up strategy planning model is proposed to separate the strategy planning and action generating task of various AI programming in RTS game. It is different from the top-down strategy tree search approach especially when conditions among nodes are unable to be determined completely due to increasing use of game units, rules and feature interactions. Fuzzy Measure and Integral is applied in performance estimation by considering the qualitative unit interaction. It is a speedy advisor for decision making in both strategy and tactics. Finally, we also proposed a new Fuzzy Integral which may have a better explanation in RTS game and proven by the data of Warcraft III. Future work includes the development of a fast indexing algorithm to retrieval strategy cases from the case-base so that the training can be done on the background or off line.

VI. ACKNOWLEDGEMENT

This research project is supported by the HK Polytechnic University grants 1-ZV5T, A-PJ18 and G-U523, and the NSFC grant no. 60903088.