Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of Acceptance: 18 March 2020.

Digital Object Identifier 10.1109/ACCESS.2019.DOI

Team Recommendation Using Order-based Fuzzy Integral and NSGA-II in StarCraft

LIN WANG[1], YIFENG ZENG[3], BILIAN CHEN[1,2], YINGHUI PAN[4] AND LANGCAI CAO[1]

[1]Department of Automation, Xiamen University, Xiamen 361005, China.
[2]Xiamen Key Laboratory of Big Data Intelligent Analysis and Decision-making.
[3]School of Computing, Teesside University, UK.
[4]Department of Information Systems, Jiangxi University of Finance and Economics, China.

Corresponding author: Yifeng Zeng (e-mail: y.zeng@tees.ac.uk).

ABSTRACT As one of the well-known real-time strategy games, StarCraft has become an important benchmark for game artificial intelligence research. Previous works in StarCraft mostly focused on strategic planning and tactical reasoning. One of the key issues in strategic planning, namely unit selection, is to build up an army, so called team in this article, with appropriate units which can gain massive destroy power against enemy's army. It is still a challenge to determine a team that has a good chance of defeating a specified army and there is no any formal algorithm solving it directly at present. Considering that the number of each unit will change during a game, if the player encounters the same enemy for multiple times, we need to select multiple winning troops, which will give the player a number of choices so as to increase the winning chance. In this article, we formulate the team recommendation as a multi-objective optimization problem and propose a novel team recommendation algorithm to solve the problem. We add a normalization of the team size to the order-based fuzzy integral and the normalized order-based fuzzy integral can better estimate the relative combat power of a team. We use genetic algorithm (GA) to learn the fuzzy measure in the fuzzy integral from the StarCraft replay data and adopt Non-Dominated Sorting Genetic Algorithm (NSGA-II) combined with the fuzzy integral for team recommendation. Finally we use a simulator called SparCraft to examine the new algorithm. The experimental results show that our proposed algorithm can recommend winning teams with a high accuracy for ordinary units in StarCraft, and the sizes of recommended teams are mostly not larger than the size of the enemy's team.

INDEX TERMS StarCraft; Team Recommendation; Order-based Fuzzy Integral; NSGA-II

I. INTRODUCTION

IN Real-time strategy (RTS) games, players need to gather resources, control limited resources to build bases, produce units, and then command appropriate units to form a team to defeat an opponent's army in a series of skirmishes. The main differences between RTS games and traditional board games are that the game environments are nondeterministic, and players are allowed to have simultaneous moves, durative actions and receive incomplete information in the gameplay due to fog-of-war. StarCraft is a well-known famous RTS game that has a large size of environmental states and action space. It has become a popular simulation environment for developing artificial intelligence research in many organizations such as FAIR [1], DeepMind [2] and Alibaba [3]. The operations in RTS games can be roughly divided into two categories: macro-management and micro-management. Macro-management (strategy) refers to the long-term planning. It includes resource gathering, base building, unit selection and technological developments. Micro-management (tactics), i.e., the short-term control, refers to a unit control. One of the most popular techniques for tactical decisions is based on game tree search, such as Alpha-Beta [4], UCT [5], and Monte Carlo planning [6]. To reduce the search space and make adversarial game tree search feasible, Barriga *et al.* [7] presented a basic implementation as an example of using Puppet Search in RTS games.

For RTS games, one of the winning keys is to produce sufficient units and select appropriate units to constitute a strong team that can destroy an enemy in a short time. This

is also a great challenge in macro-management, i.e., facing a specified troop under the command of a potential opponent, how to select a similarly sized team from the units we own that can eliminate the enemy. During the game, players will produce different units according to their own build tree. At the same time, in a series of battles, some units will be destroyed, that is, the number of units owned by the player is constantly changing. Therefore, when an enemy army strikes, it is best for the player to have several different solutions and then they can choose a suitable team to win the battle according to the units they currently own. At present, there is no formal approach to address this problem directly.

We first need an approach to evaluate the combat effectiveness of a given team. This is not easy due to different attributes of each kind of unit e.g., the attack or defense capabilities. Furthermore, the interaction between different unit types is non-additive. The combat power of a team of two types of units may be larger or smaller than the sum of the combat power of the two units. It is because some units strengthen each other while some units weaken each other. Fuzzy measure and integral could be a proper approach to handle the non-additive property in the team evaluation. Li *et al.* [8] proposed an order-based fuzzy integral that can evaluate the combat power of a team to some extent. One problem with this integral is that the calculated value is equal when the proportion of each unit of the two armies is the same, which is not in line with the actual situation. We adjust the fuzzy integral to address this problem in this article. Then we design an algorithm based on the fuzzy integral to select appropriate units for a series of battles that take place in StarCraft.

We list the main contributions as follows.

- We add a normalization of the team size to an order-based fuzzy integral, and the normalized order-based fuzzy integral can better learn the fuzzy measure in the fuzzy integral.
- We consider the unit selection problem of StarCraft as a multi-objective optimization problem (MOP) and design a team recommendation algorithm using elitist non-dominated sorting genetic algorithm (NSGA-II) and the normalized order-based fuzzy integral. This is the first team recommendation algorithm for StarCraft, which solves the unit selection problem in the face of a specified enemy's team.
- Experiments with SparCraft are conducted. The experimental results demonstrate that the proposed algorithm has a good performance on the selection of general units in StarCraft.

The remainder of this article is organized as follows. Section II discusses the related work of macro-management and the existing fuzzy integrals for a unit selection. Besides, we analyze the limitations of previous fuzzy integral. Section III reviews the knowledge of fuzzy measure, order-based fuzzy integral and NSGA-II. We propose the new algorithm in Section IV. Then, we compare the performance of the normalized order-based fuzzy integral with order-based fuzzy integral and utilize SparCraft to evaluate our algorithm's performance in Section V. Finally, Section VI summarizes our work and discusses future work.

II. RELATED WORKS

Since team recommendation is an important issue in macro-management, we first review related works on game macro-management. The hard-coded approach uses finite state machine (FSM) to let an AI author hard-code the strategy [9] [10]. A HRL approach using a set of 165 hard-coded macro actions as the low-level has recently been successful in a full-game StarCraft II AI [11]. Ontanon *et al.* [12] [13] studied the use of real-time case-based planning (CBP) in the domain of Wargus (a Warcraft II clone). Synnaeve and Bessiere [14] presented a Bayesian semi-supervised model to learn and predict openings from replays of StarCraft. Justesen *et al.* [15] [16] proposed continual online evolutionary planning (COEP) to continually evolve build-orders itself during the game to adapt to an opponent. In recent years, there are a lot of applications of reinforcement learning and deep learning in macro-management. Justesen *et al.* [16] showed how macro-management decisions in StarCraft can be learned directly from game replay data using deep learning, and an actor-critic approach based on recurrent neural network was applied to exchange information between units [3] [17]. Barriga *et al.* [18] used a deep convolutional neural network (CNN) to select among a limited set of abstract choices in RTS games, and to utilize the remaining computation time for game tree search to improve low-level tactics. In January 2019, DeepMind released AlphaStar [19], which is the first artificial intelligence (AI) system to beat a professional player at the game of StarCraft II. AlphaStar plays the full game of StarCraft II using a deep neural network that is trained directly from raw game data through supervised learning and reinforcement learning techniques.

There is still limited research in regard to a unit selection although it is an important issue in StarCraft. Murofushi *et al.* [20] proved that the Choquet Integral was truly significative for non-monotonic property, but there are very few literatures in a practical application. Li *et al.* [8] proposed three different fuzzy integrals for the unit selection problem: max-based fuzzy integral, mean-based fuzzy integral and order-based fuzzy integral. Their experimental results prove that an order-based fuzzy integral is more appropriate in an interaction detection. They then presented a fuzzy integral named Directional based Fuzzy Integral to support the flank and diversion attack in unit formation planning in RTS games [21]. Nguyen *et al.* [22] proposed an evaluation method that can cope with nonadditive properties and unit properties. This approach was also successfully applied to heuristic search for micro-management in RTS games. Peter and Simon [23] used fuzzy measure and integral to extend artificial potential field. They showed a new direction for extracting behaviors from human players and provided different unit manoeuvers.

Li *et al.* [8] proposed the order-based fuzzy integral by considering a unit production sequence. Each unit calculates the interaction with the one who has less production. Although the order-based fuzzy integral achieves good results, we find that it can only evaluate the performance of team proportion. Considering two teams with the same proportion, the fuzzy integral of them is equal even if the total number of their units is different. We concentrate on the following two teams/armies as an example.

1) Archon: 2, Dragoon: 3, Zealot: 5
2) Archon: 4, Dragoon: 6, Zealot: 10

The composition ratios of the above two armies are equal, i.e., the proportion of each unit is the same. Hence, the order-based fuzzy integral of the two armies returns the same value. However, it is apparent that two armies' combat effectiveness is different because of the different total number of units. In fact, if we want to estimate the fighting capacity of a team, it is not feasible to use order-based fuzzy integral directly. Hence we adjust the integral for the unit selection, which will be introduced in Section IV-A.

Most of the optimization problems are MOPs in practice. It is difficult to make all the subgoals reach the optimal simultaneously, and only a set of solutions that are compromised by each objective are obtained, which are called the Pareto optimal solutions. Fonseca *et al.* [24] first proposed using MOGA (Multi-objective Genetic Algorithm) to solve MOPs. NPGA (Niched Pareto Genetic Algorithm) proposed by Srinivas and Deb [25] runs faster, but requires an appropriate selection of the size of comparison set, which complicates its practical applications. Kim *et al.* [26] proposed SPEA2+ (Strength Pareto Evolutionary Algorithm) using a more effective crossover mechanism and an archive mechanism to maintain the solution diversity. PAES (Pareto Archived Evolution Strategy) posed by Knowles and Corne [27] can provide a local search operation. Deb *et al.* [28] presented NSGA and NSGA-II exploiting non-dominated sorting. Particularly NSGA-II reduces the computational complexity of NSGA. These methods treat MOP as a whole and mainly rely on domination for measuring the solution quality. Zhang and Li [29] proposed MOEA/D (Multi-objective Evolutionary Algorithm based on Decomposition) using a decomposition method to decompose the MOP into a number of scalar optimization problems. MOEA/D performs similarly to NSGA-II on solving some problems. Louis *et al.* [30] utilized NSGA-II to optimize micro based on a multi-objective formulation of the fitness function that maximized damage done and minimized damage taken, the problem they attacking was controlling teams of autonomous units during skirmishes in real-time strategy games. They cooperatively co-evolved micro for a ranged unit using a parameterized control algorithm along with micro for a melee unit using a pure potential fields approach [31] [32].

III. BACKGROUND: SUBMODULAR FUNCTION

The key to fuzzy integral is fuzzy measure, and we use a genetic algorithm (GA) to learn the fuzzy measure in the fuzzy integral. After learning fuzzy measure with GA, the military's order-based fuzzy integral can be calculated. We use this fuzzy integral and NSGA-II for team recommendation. In this section, we first introduce basic concepts of fuzzy measure and a classic fuzzy integral. Then we introduce the order-based fuzzy integral in detail. At last, we provide the related knowledge of NSGA-II.

A. FUZZY MEASURE AND CHOQUET INTEGRAL

Fuzzy measure is a special non-additive measure, and it replaces the additive property of probability measure with weaker condition as monotonicity [33]. Assuming that (X, σ) is a measurable space, if the mapping $\mu : \sigma \to [0, 1]$ satisfies

1) $\mu(\varnothing) = 0, \mu(X) = 1$;
2) $\forall A, B \in \sigma$, if $A \subseteq B$, then $\mu(A) \leq \mu(B)$;
3) if $\forall i \in N$, $A_i \in \sigma$, and $\{A_i\}$ is monotonous, then $\lim\limits_{n\to\infty} \mu(A_n) = \mu(\lim\limits_{n\to\infty} A_n)$,

then we call μ as the fuzzy measure and (X, μ) as the fuzzy measure space.

Choquet Integral [34] is a classic fuzzy integral. Given a fuzzy measure μ on X, the discrete Choquet integral of a function $f : X \to R^+$ can be written as

$$(c)\int f(x) \circ \mu(X) = \sum_{i=1}^{n} \Big(\big[f(x_i) - f(x_{i-1}) \big] \cdot \mu\big(\{x | f(x) \geq f(x_i)\} \big) \Big),$$

where $x_i \in X$ and $f(x_0) = 0$.

B. ORDER-BASED FUZZY INTEGRAL

For a team A, the set of unit types it contains is $TY = \{x_1, x_2, ..., x_m\}$, then the order-based fuzzy integral [8] estimates its combat power as

$$\begin{aligned} Power(A) &= \int f_A(x) \circ \mu(X) \\ &= \sum_{i=1}^{m} \Big(f_A(x_i) \cdot \mu\big(\{x | f_A(x) \leq f_A(x_i)\} \big) \Big), \end{aligned} \tag{1}$$

where $f_A(x)$ is defined as the proportion of unit type x ($x \in TY$) in A; x_i is the ith unit type; m is the total of unit types. X represents a combination of unit types, e.g., $\{x_1, x_2\}, \{x_2, x_m\}$. $\mu(X)$ is the fuzzy measure of combination X.

C. NSGA-II

In the team recommendation problem studied in this article, we have two objectives. Hence we adopt NSGA-II [28] which is one of the most widely used multi-objective genetic algorithms. The main advantages of NSGA-II are as follows.

- It is a fast non-dominated sorting algorithm and reduces the complexity of computing non-dominated sorting.

TABLE 1: The symbols used and interpretation

Symbol	Interpretation
$\mu(X)$	fuzzy measure of combination X
E	enemy's team
EY	set of unit types of E
EN	set of the number of each unit in EY
$total_E$	total of units in E
C	candidate team
CY	set of unit types of C
CN	set of number of each unit in CY
$total_C$	total of units in C
AC	set of possible unit combinations of C
N'	population size
$Mgen'$	maximum generation
p'_c	crossover probability
p'_m	mutation probability

- It introduces an elite strategy to expand the sampling space. The parent population is combined with its offsprings to produce a new generation. This approach ensures that good individuals are not discarded during the evolution.
- It introduces a congestion degree and congestion degree comparison operator, which not only faciliates the NSGA to specify parameters in an easy manner , but also ensures the population diversity.

IV. TEAM RECOMMENDATION USING ORDER-BASED FUZZY INTEGRAL AND NSGA-II

In this section, we present the team recommendation algorithm using order-based fuzzy integral and NSGA-II for StarCraft, which we will call OFIN in the following text. We first introduce the procedure of using GA to learn the fuzzy measure from the StarCraft replay data and then explain the representation of the team recommendation problem. Finally, we introduce the OFIN algorithm in detail. The symbols used and their explanations are summarized in Table 1.

A. LEARNING FUZZY MEASURE BY GA

To address the problem mentioned in Section II, we normalize the order-based fuzzy integral of Eq. (1). Assuming that the teams to be evaluated are $S = \{A, B, C, D, ...\}$, we adopt Min-Max normalization method and normalize the total number of units of all teams in S. The total number of units of each team is $t_s (s \in S)$. The minimum is t_{min} and the maximum is t_{max}, then the normalized value of A is $n(A) = \frac{t_A - t_{min}}{t_{max} - t_{min}}$. The normalized order-based fuzzy integral of team A can be written below.

$$\begin{aligned} Power2(A) &= n(A) \int f_A(x) \circ \mu(X) \\ &= n(A) \sum_{i=1}^{m} \Big(f_A(x_i) \cdot \mu\big(X_i\big) \Big), \end{aligned} \tag{2}$$

where $X_i = \{x | f_A(x) \leq f_A(x_i)\}$.

We also normalize team A with four different non-linear functions: log function, $L2$ normalization, $Arctan$ function and $Sigmoid$ function. The performance comparison of the five normalization methods will be shown in Section V-B, and the comparison of the effects of using order-based fuzzy integral and normalized order-based fuzzy integral will also be shown in Section V-B.

TABLE 2: Scores of the unit combination extracted from the replay data

Unit combination	Score
Dragoon 5 Zealot 1	700
Arbiter 1 Dragoon 10 Zealot 5	1400
Arbiter 1 Dragoon 6 Zealot 3	1000
Dragoon 13 Reaver 2 Shuttle 1	1500
Dragoon 1 Reaver 1 Zealot 1	1000
Archon 2 Dragoon 10 Zealot 2	6050
Dragoon 4 Zealot 2	400
Dark_Templar 2 Zealot 13	1000
...	...

If a battle involves n types of units $\{x_1, x_2, x_3, ..., x_n\}$, to estimate the combat effectiveness of both sides using Eq. (2), we need to obtain the 2^n-1 fuzzy measures corresponding to all possible unit combinations, i.e., $\mu(x_1), \mu(x_2), ..., \mu(x_1, x_2), ..., \mu(x_1, x_2, ..., x_n)$. Each fuzzy measure represents an interaction of the unit combination.

In StarCraft, each player will obtain a score after each battle as the performance evaluation. The score depends on the player's mining income, economic and technological developments, the number of buildings and units built, and the number of enemies and enemy buildings that are annihilated. Hence, the score can reflect the comprehensive ability of each team. The higher the score is, the more powerful the team is, and the winning team has a higher score than the other. Therefore, we use these scores to learn the fuzzy measure and we can consider the calculated fuzzy integral in Eq. (2) as the estimated score, i.e., the estimated power of the team [8] [35]. To do this, we extract some replay data including the unit combination and the score of each team as Table 2 shows, which will be introduced in detail in Section V-A. This leads to a problem of searching for the fuzzy measure to "best" fit the replay data. We use GA to learn the fuzzy measure from the data.

In the GA development, we adopt real-coded chromosomes and each chromosome consists of 2^n-1 nodes. Each node represents a fuzzy measure and takes a decimal value between 0 and 1. A number of chromosomes will be initialized in a random way. The fitness calculation of one chromosome is described below.

1. Acquire the corresponding fuzzy measure from the chromosome.
2. Extract real scores and the unit production statistic function (aforementioned $f(x)$) from the collected data and normalize the real scores as $score_{real}$.
3. Calculate the fuzzy integral in Eq. (2) as the estimated score $score_{est}$.
4. Calculate the root mean square error (RMSE) over the training data.

$$RMSE = \sqrt{\frac{1}{H}\sum_{i=1}^{H}\left(score_{est}(i) - score_{real}(i)\right)^2},$$

where i is the ith team in the training data, H refers to the total number of armies.

5. Compute the fitness value:

$$Fitness\ Value = \frac{1}{1 + RMSE}.$$

In the evolution process, the chromosome with a high value of fitness will be treated as a better chromosome. The learning process is repeated until the fitness value is stable or the iteration exceeds the maximum generation. After that, a set of reasonable fuzzy measures for our replay data are obtained.

B. TEAM RECOMMENDATION PROBLEM FORMULATION

Team recommendation problem in StarCraft is formulated as follows. Given an enemy's team E, the set of its unit types is $EY = \{e_1, e_2, \cdots, e_K\}$, the number of each unit in EY is $EN = \{n_{e1}, n_{e2}, \cdots, n_{eK}\}$ and the total number of units in E is $total_E$. The units owned by a subject player is C, the set of unit types of C is $CY = \{c_1, c_2, \cdots, c_L\}$, the number of each unit in CY is $CN = \{n_{c1}, n_{c2}, ..., n_{cL}\}$ and the total of units in C is $total_C$. We aim to select several teams from C in order to defeat the enemy team E.

All possible unit combinations of the candidates $AC = \{\{c_1\}, \cdots, \{c_1, c_2\}, \cdots, \{c_1, c_2, \cdots, c_L\}\}$. For each combination $m = \{c_1, c_2, \cdots, c_l\}$ $(1 \le l \le L)$, we need to determine the number of each unit in a combination $\{h_{c1}, h_{c2}, \cdots, h_{cl}\}$, so that the team M consisting of the determined number of units can defeat the enemy. Hence, for the combination m, our goal is to maximize the normalized order-based fuzzy integral of it and minimize the total number of its units, as described in Eq. (3). This is a MOP with two objective functions and the decision variables are the number of each unit in M, i.e., h_{ci} $(i = 1, 2, \cdots, l)$.

$$\begin{aligned} &max\ Power2(M) = n(M) \int f_M(c) \circ \mu(C) \\ &min\ Size(M) = \sum_{i=1}^{l} h_{ci} \\ &s.t.\ 0 \le h_{ci} \le n_{ci}, i = 1, 2, \cdots, l \end{aligned} \tag{3}$$

C. OFIN ALGORITHM

In Algorithm 1, we develop the OFIN algorithm to solve the team recommendation in for StarCraft. N' represents the population size, $Mgen'$ is the maximum generation, p'_c is the crossover probability and p'_m is the mutation probability. For each combination, we adopt NSGA-II to solve the MOP in Eq. (3) and get a set of pareto optimal solutions pop (lines 2-3). The process of NSGA-II is shown in Algorithm 2. In NSGA-II, for each new population, firstly, calculate two parameters of each individual i: the number of individuals who dominate i and the set of individuals that are dominated by i. The whole population is stratified according to these two parameters, namely Fast_Non_Dominated_Sort. Next, the crowding distance of each individual in each non-dominated layer is calculated according to the objective function, i.e., Crowding_Distance_Calculation. Then N individuals with lower non-dominated levels or larger crowdedness are selected as the next new population. For details of NSGA-II, please refer to [28]. Algorithm 3 describes the $Sort(pop)$ process of NSGA-II. After that, the total number of units of each pareto solution and the enemy are normalized. The normalized values are stored in set SN' (lines 4-9). Then, the fuzzy integral of each pareto solution and the enemy can be calculated. We will add the pareto solution with fuzzy integral greater than or equal to the fuzzy integral of enemy $(Power2(E))$ to set T (lines 10-17). Next, for each solution in T, if its total number of units is the smallest in T, we will add it to the target team set WT (lines 18-22). After traversing all the combinations, all the teams in WT are the recommended teams upon the given enemy team.

Algorithm 1 OFIN

Input: $E(enemy\ team), total_E(size\ of\ E)$;
$C(candidate\ team), total_C(size\ of\ C)$;
$AC(all\ unit\ combinations\ of\ C)$;
$\mu(X)(fuzzy\ measure); N', Mgen', p'_c, p'_m$
Output: $WT(set\ of\ recommended\ teams)$

1: $WT \leftarrow \varnothing, T \leftarrow \varnothing$
2: **for** each $m \in AC$ **do**
3: $pop \leftarrow NSGA-II(N', Mgen', p'_c, p'_m)$
4: $S' \leftarrow \varnothing, p \leftarrow 0$
5: **while** $p < N'$ **do**
6: $S' \leftarrow S' \cup sum(pop[p]), p \leftarrow p + 1$
7: **end while**
8: $S' \leftarrow S' \cup total_E$
9: $SN' \leftarrow$ normalize(S')
10: $calculate\ Power2(E), p \leftarrow 0$
11: **while** $p < N'$ **do**
12: $calculate\ Power2(pop[p])$
13: **if** $Power2(pop[p]) \ge Power2(E)$ **then**
14: $T \leftarrow T \cup pop[p]$
15: **end if**
16: $p \leftarrow p + 1$
17: **end while**
18: **for** each $t \in T$ **do**
19: **if** $Size(t) = min(T)$ **then**
20: $WT \leftarrow WT \cup t$
21: **end if**
22: **end for**
23: **end for**
24: **return** WT

Using the OFIN algorithm, we can select some suitable units from the candidates to form a team to fight against the

Algorithm 2 NSGA-II($N', Mgen', p'_c, p'_m$)

initialize pop
$NS \leftarrow Sort(pop)$
$pop2 \leftarrow make\_new\_pop(NS, p'_c, p'_m)$
$gen = 0$
while $gen < Mgen'$ **do**
  $hbpop \leftarrow pop \cup pop2$
  $NS' \leftarrow Sort(hbpop)$
  $Crowding\_Distance\_Calculation(NS')$
  $pop \leftarrow select\ N'\ individuals\ from\ hbpop$
  $pop2 \leftarrow make\_new\_pop(pop, p'_c, p'_m)$
  $gen \leftarrow gen + 1$
end while
return pop

Algorithm 3 Sort(pop)

$S \leftarrow \varnothing, p \leftarrow 0$
while $p < N'$ **do**
  $S \leftarrow S \cup sum(pop[p]), p \leftarrow p + 1$
end while
$SN \leftarrow$ normalize(S)
$p \leftarrow 0, Power2 \leftarrow \varnothing, Size \leftarrow \varnothing$
while $p < N'$ **do**
  $Power2 \leftarrow Power2 \cup Power2(pop[p])$
  $Size \leftarrow Size \cup Size(pop[p]), p \leftarrow p + 1$
end while
$NS \leftarrow Fast\_Non\_Dominated\_Sort(Power, Size)$
return NS

enemy in the case of knowing the opponent's team. In the next section, we will conduct experiments to demonstrate the effectiveness of the above algorithm.

V. EXPERIMENTS

We demonstrate the performance of OFIN by simulating battles in the SparCraft platform. In this section, we first describe the dataset and the data preprocessing. Then, we compare the effects of using the order-based fuzzy integral and the normalized order-based fuzzy integral to learn the fuzzy measure. After that, we discuss the the parameter settings of NSGA-II. Lastly, we detail the simulation of the recommended teams and analyze the simulation results.

A. DATASET AND PREPROCESSING

We choose StarCraft: Brood War (SC: BW) as the simulation platform and it is an expansion of StarCraft. It was officially released in 1998 and becomes extremely popular as an e-sport. We focus on SC: BW as it has gained the most popularity within the field of game AI, while the presented approach can be applied to all the games in the StarCraft series as well as similar RTS games. The Brood War Application Programming Interface (BWAPI[1]) is a free and open source C++ framework that is used to interact with SC: BW, and it has been the foundation of several StarCraft AI competitions. Using BWAPI, researchers can write competitive AI for SC: BW by controlling individual units, analyze replays frame-by-frame and examine real-time AI algorithms in a robust commercial RTS environment. Players have the options to save a replay file after each game in StarCraft. Replay files contain the set of actions performed by both players, which the StarCraft engine can use to reproduce the exact events. Therefore, replay files are a great resource for machine learning if one wants to learn how players are playing the game. In this paper, we use BWAPI to extract useful information from replay files to learn the fuzzy measure.

There are three races in StarCraft - Protoss, Terran and Zerg. Each race has different units, buildings, and upgrade options. We only do the experiment of the Protoss v.s. Protoss match-up since this is the only match-up that provides multiple most used types and is available in SparCraft which will be described in Section V-D1. We collect 300 replays of professional one-versus-one competitions of SC: BW from ICCup[2], and write a C++ program in BWAPI and extract 2254 battles from 300 replays. We use 1000 battles among them for the experiments. Hence there are total 2000 armies in the training data because each battle includes two sides. There are 8 units of Protoss involved in the 2000 teams: Dragoon, Zealot, Archon, Dark Templar, Reaver, Shuttle, High Templar and Arbiter. Two kinds of data are included in each battle. One is the unit statistics of the teams of both players, i.e., the type and number of each unit. The other is the score of the team obtained after each battle. Scores are given by the SC: BW system after the winner defeats his opponent, and the scores will be normalized to learn the fuzzy measure.

B. COMPARISON OF LEARNING PERFORMANCE OF FUZZY MEASURE

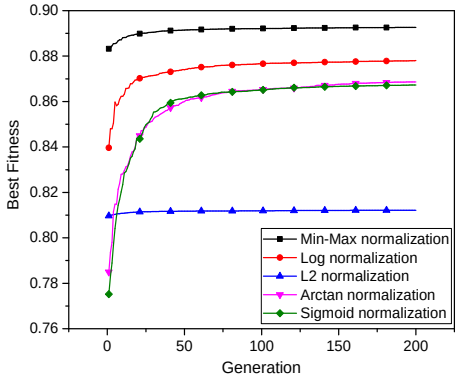
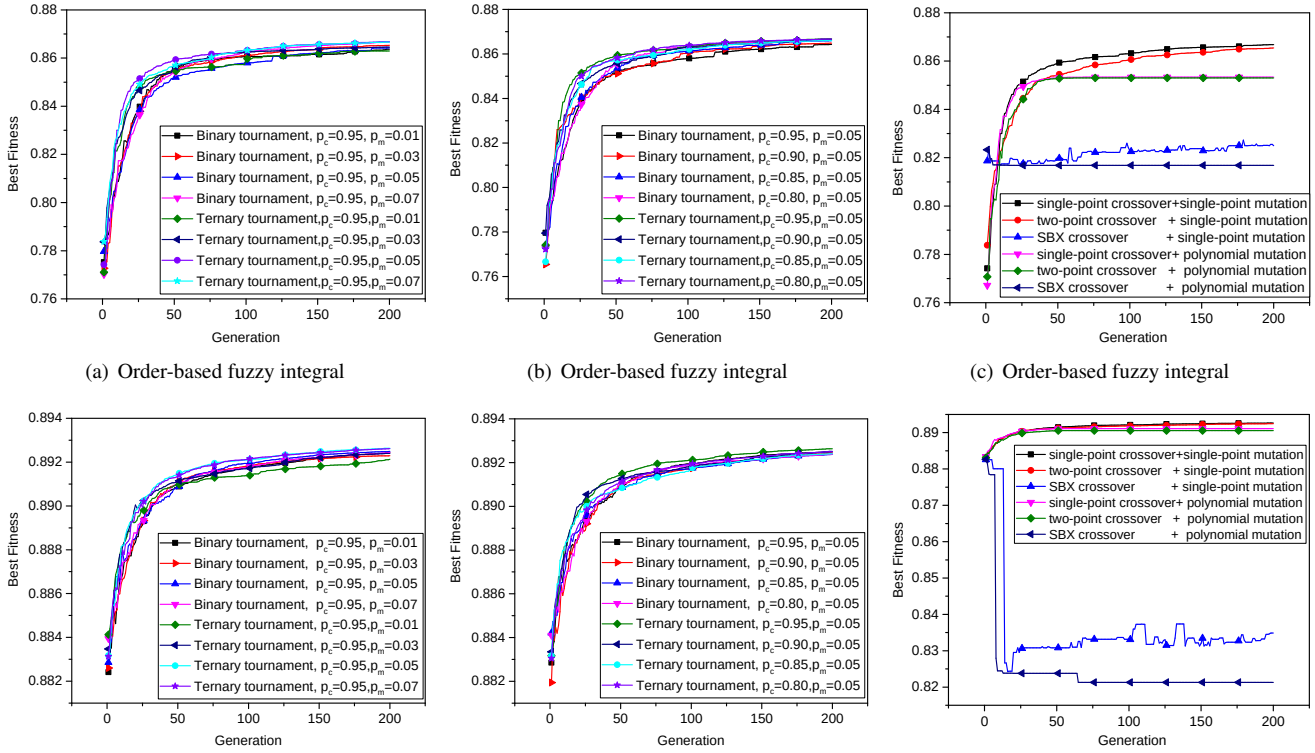


Fig. 1: Comparison of best fitness with different normalization methods.

We use GA to learn the fuzzy measure with the order-based fuzzy integral and the normalized order-based fuzzy integral

[1]https://github.com/bwapi/bwapi

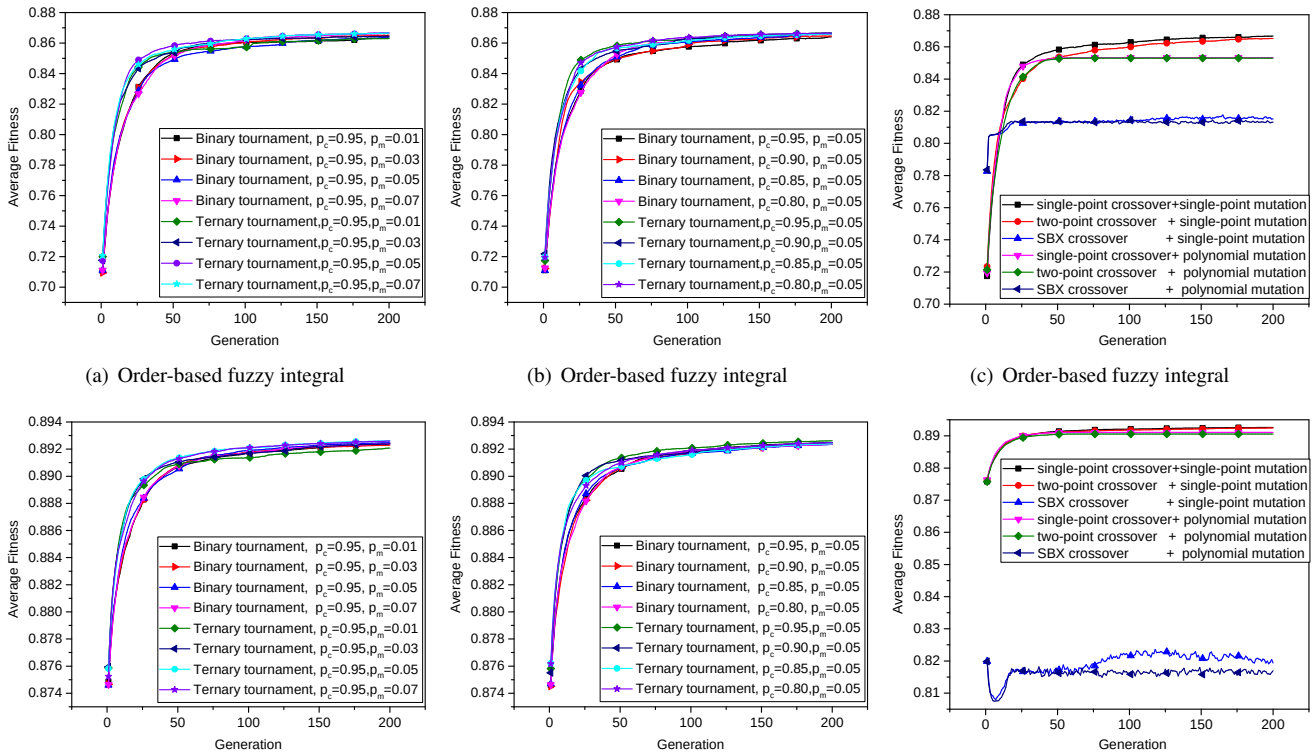
[2]http://iccup.com/en/starcraft/replays.html



(a) Order-based fuzzy integral (b) Order-based fuzzy integral (c) Order-based fuzzy integral

(d) Normalized order-based fuzzy integral (e) Normalized order-based fuzzy integral (f) Normalized order-based fuzzy integral

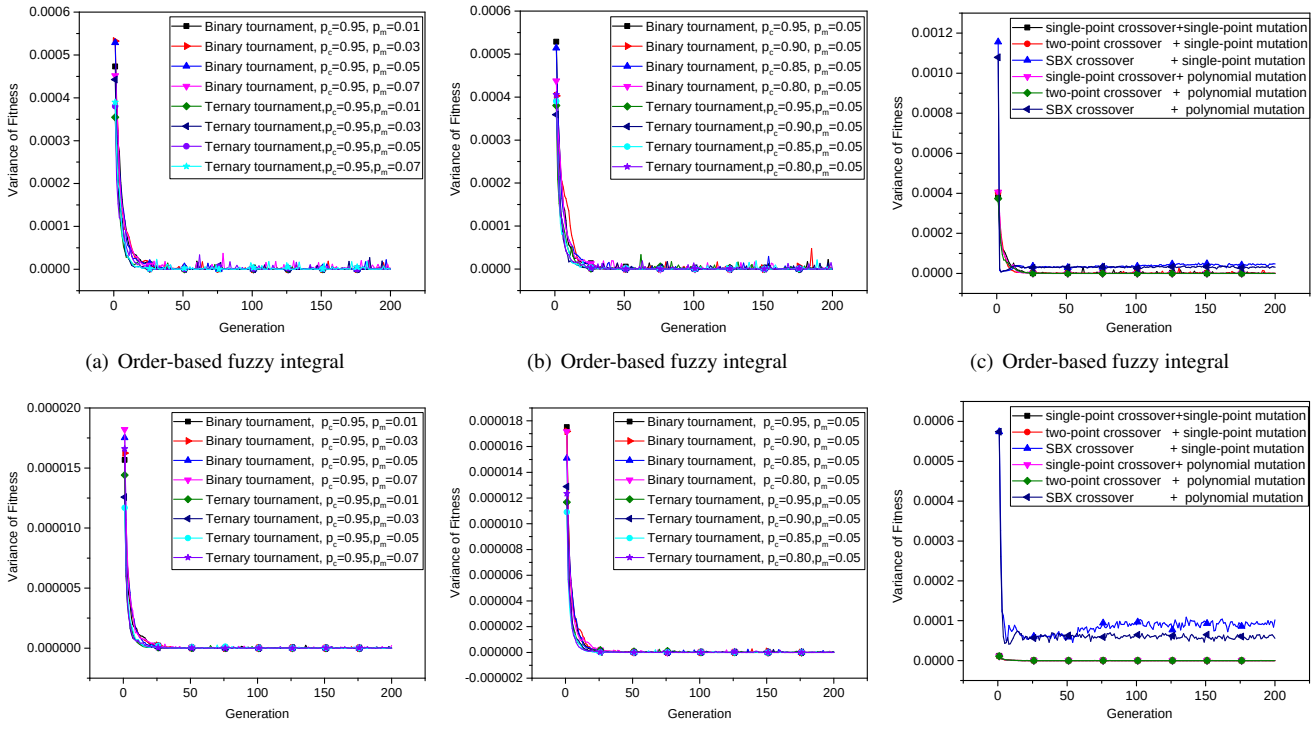
Fig. 2: Comparison of best fitness



(a) Order-based fuzzy integral (b) Order-based fuzzy integral (c) Order-based fuzzy integral

(d) Normalized order-based fuzzy integral (e) Normalized order-based fuzzy integral (f) Normalized order-based fuzzy integral

Fig. 3: Comparison of average fitness



(a) Order-based fuzzy integral (b) Order-based fuzzy integral (c) Order-based fuzzy integral

(d) Normalized order-based fuzzy integral (e) Normalized order-based fuzzy integral (f) Normalized order-based fuzzy integral

Fig. 4: Comparison of variance of fitness

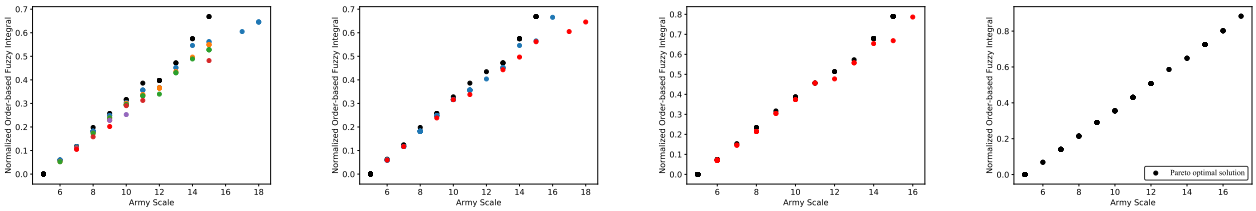
respectively. The comparison of best fitness using the normalized order-based fuzzy integral with different normalization methods is shown in Fig. 1. As expected, the Min-Max normalization performs best. Fig. 2 shows the comparison of best fitness of order-based fuzzy integral and normalized order-based fuzzy integral. Fig. 3 shows the comparison of average fitness. Fig. 4 shows the comparison of variance of fitness. In Figs. 2-4, each column compares the performance of order-based fuzzy integral and normalized order-based fuzzy integral with the same parameters of GA, and each subgraph compares the effect of different parameters of GA on performance. We separately adopt binary and ternary tournament, different p_c, p_m, different methods of crossover and mutation to find the best combination of parameters. Finally, we adopt ternary tournament selection, single-point crossover and single-point mutation, $N = 500$, $Mgen = 200$, $p_c = 0.95$ and $p_m = 0.05$. From each column, it can be clearly seen that the normalized order-based fuzzy integral significantly improves the convergence accuracy contrasted to the order-based fuzzy integral. Since the normalized order-based fuzzy integral takes into account the total number of units in the team, which has a great influence on the combat effectiveness of the team, it can better assess the combat power of the team and better learn the fuzzy measure.

C. SETTING OF NSGA-II

In the team selection, we apply NSGA-II to each unit combination to find the pareto optimal solution of the combination. We refer to [28] for parameter setting of NSGA-II. We adopt real-coded GA, the binary tournament selection, Simulated Binary Crossover (SBX) operator and polynomial mutation. Parameter settings: $N' = 100$, $Mgen' = 50$, $p'_c = 0.9$ and $p'_m = \frac{1}{n}$ (where n is the number of decision variables). The distribution indexs for SBX and polynomial mutation are $\eta_c = 1$ and $\eta_m = 1$, respectively. Fig. 5 shows the pareto solutions of a combination of four kinds of units when $p'_c = 0.9$ and $p'_m = \frac{1}{n}$, Fig. 6 shows the pareto solutions when $p'_c = 0.9$ and $p'_m = 0.05$, Fig. 7 shows the pareto solutions when $p'_c = 0.95$ and $p'_m = 0.05$, Fig. 8 shows the pareto solutions when $p'_c = 0.95$ and $p'_m = \frac{1}{n}$. We can see that when $p'_c = 0.9$ and $p'_m = 0.05$, solutions can basically converge to the pareto optimal solutions in the 25th generation. Under the other parameter settings, the results are not much different, and they all converge around the 20th generation.

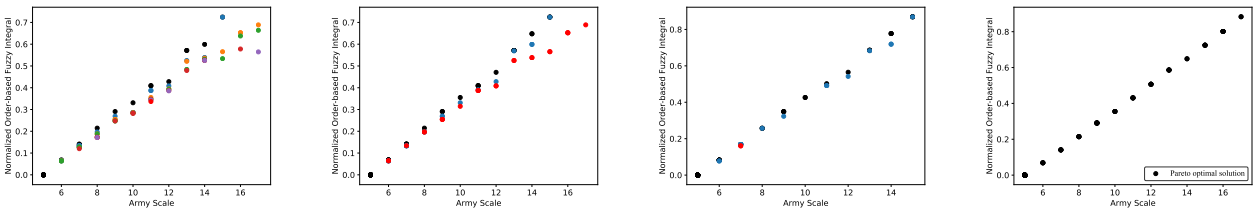
For comparison with NSGA-II, we also apply MOEA/D [29] to find the Pareto optimal solutions of each unit combination. The basic parameter settings are the same as NSGA-II. Fig. 9 shows the Pareto solutions of a combination of four kinds of units when using MOEA/D. The results show that by the 50th generation, solutions have not yet converged to the Pareto optimal solutions.

D. SIMULATION OF RECOMMENDED TEAMS



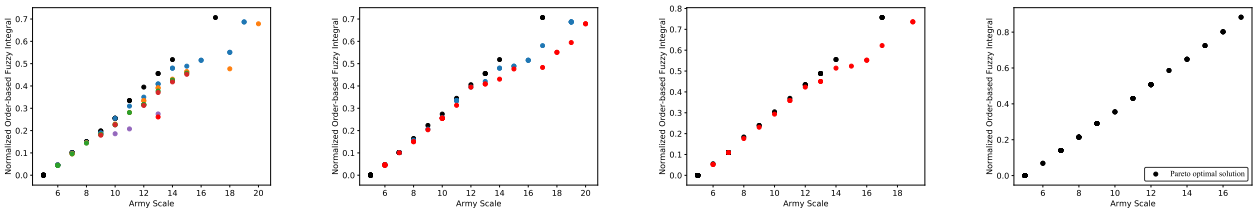
(a) The 1^{th} generation (b) The 2^{th} generation (c) The 3^{th} generation (d) The 20^{th} generation

Fig. 5: Pareto solutions for each generation of a combination of four kinds of units when $p'_c = 0.9$ and $p'_m = \frac{1}{n}$. Points with the same color are solutions on the same non-dominated front. The horizontal axis represents army scale and the vertical axis represents normalized order-based fuzzy integral.



(a) The 1^{th} generation (b) The 2^{th} generation (c) The 3^{th} generation (d) The 25^{th} generation

Fig. 6: Pareto solutions for each generation of a combination of four kinds of units when $p'_c = 0.9$ and $p'_m = 0.05$. Points with the same color are solutions on the same non-dominated front. The horizontal axis represents army scale and the vertical axis represents normalized order-based fuzzy integral.



(a) The 1^{th} generation (b) The 2^{th} generation (c) The 3^{th} generation (d) The 20^{th} generation

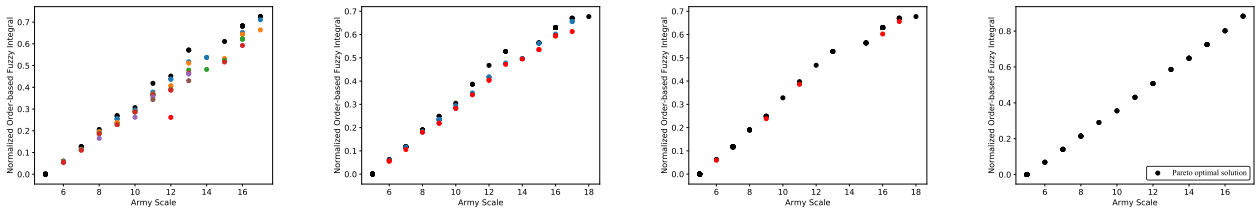
Fig. 7: Pareto solutions for each generation of a combination of four kinds of units when $p'_c = 0.95$ and $p'_m = 0.05$. Points of the same color are solutions on the same non-dominated front. The horizontal axis represents army scale and the vertical axis represents normalized order-based fuzzy integral.

1) SparCraft

We use SparCraft as the simulation platform in the experiments. SparCraft is developed by David Churchill [36]. It is an open source simulation package that can simulate StarCraft combats with a high level of accuracy. In SparCraft, units can be given commands of attack, move and wait [5]. All unit attributes such as size, speed, armor, cooldown times, hit points and weapon types are modelled precisely from StarCraft except for acceleration, with all units having constant speed while moving. The whole upgrades and research are modelled. Nonetheless, spell casters and some units (Reaver, Bunker, Carrier, Transport) are not yet implemented. It does not yet carry out unit collisions or fog of war, either. SparCraft allows battles to be set up and carried out in accordance with deterministic play-out scripts, or by search-based agents. Investigators can easily implement new algorithms, integrate them into this simulator and use it as a test bed for their own research. This simulator can merely support simulations of a limited types of units, only four of the Protoss: Dragoon, Zealot, Archon and Dark Templar. Therefore, our experiment only covers these four units.

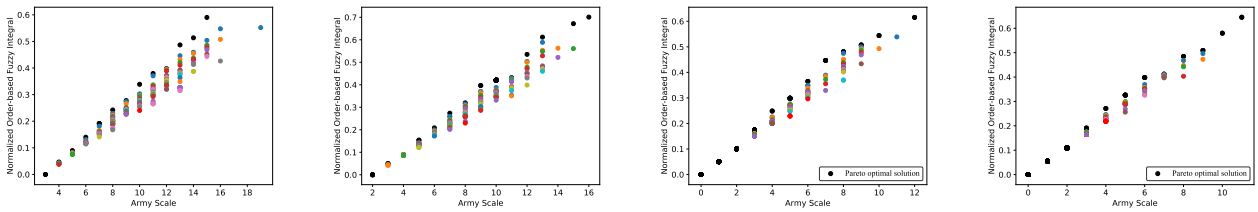
2) Simulation Settings

In our work, we adopt Portfolio Greedy Search for both players, which is proposed by David Churchill and Michael Buro in [5]. Portfolio Greedy Search uses playouts together with the LTD2 evaluation formula [37]. It takes as input an initial combat state, a set of scripts called a portfolio and two integer values I and R. I is the number of improvement



(a) The 1^{th} generation (b) The 2^{th} generation (c) The 3^{th} generation (d) The 20^{th} generation

Fig. 8: Pareto solutions for each generation of a combination of four kinds of units when $p'_c = 0.95$ and $p'_m = \frac{1}{n}$. Points of the same color are solutions on the same non-dominated front. The horizontal axis represents army scale and the vertical axis represents normalized order-based fuzzy integral.



(a) The 1^{th} generation (b) The 5^{th} generation (c) The 20^{th} generation (d) The 50^{th} generation

Fig. 9: Pareto solutions for each generation of a combination of four kinds of units when using MOEA/D. Points of the same color are solutions on the same non-dominated front. The horizontal axis represents army scale and the vertical axis represents normalized order-based fuzzy integral.

iterations and R is the number of responses. As an output, it produces a player move. Using the same search technique for both sides can eliminate the noise caused by different skill levels and playing style of players. We refer to the parameter setting in [5] when we set the Portfolio Greedy Search: I=1, R=0 and time limit=40ms. The portfolio includes two scripts, i.e., NOK-AV and Kiter.

- NOK-AV (No-OverKill-Attack-Value): Units will attack an opponent unit with the highest damage per frame (or hit points) within range when it is able to fire. However, it will not attack units that have already been targeted and can be defeated by other friendly units this round. It will instead choose the next priority target, or wait if such a target does not exist.
- Kiter: Units will attack the closest enemy unit within their weapons' range, and it tends to move a fixed distance away from the closest enemy when it can not fire.

To run the simulation, we should specify what initial states of both players will be used. There are three categories of states that can be specified by the game settings file: State Description File, Symmetric State, and Separated State. The latter two methods can only specify the same units for both players. Consequently, they can not meet our experimental requirements. Therefore, we adopt the first method, i.e., State Description File. States are defined by an external file which lists all initial units and their properties. In the description file, we can assign the initial units each player controlled and the initial position of each unit on the map.

3) Simulation Results

Since both sides use the same search algorithm, it is only the different unit composition that determines the final outcome. Therefore, after determining the team of the player and the enemy, only one simulation is required for each battle, the accuracy of our experiment is determined by how many outcomes this algorithm can predict correctly (i.e., how many teams recommended can win). We conduct a series of experiments and repeat each experiment for 50 times. The results of the experiments are fairly stable and shown in Table 3. In addition, we adopt MOEA/D, random strategy and heuristic method (i.e., based on experience) for the team recommendation. The comparison of the recommended accuracy of the four methods is shown in Fig. 10.

4) Result Discussion

In Table 3, we can see that, for most enemy troops, the proposed algorithm can recommend winning teams with a high accuracy, and it is worth mentioning that the total number of units of each recommendation is smaller than or equal to the total of enemy units. However, for the enemy armies with a large proportion of Archon, the accuracy of the recommendation of our algorithm is relatively low. Considering that Archon has sputter damage, i.e., special skill, when Archon has a large proportion of the enemy, our algorithm cannot accurately recommend all winning teams.

TABLE 3: Results of the experiments
(Candidates: A:5, Dr:5, Z:5, Da:5)

Enemy	Recommendation				Can win?	Accuracy
	A	Dr	Z	Da		
Z:4, A:2, Dr:3, Da:5	4	5	5	0	Yes	80%
	4	4	0	4	Yes	
	5	0	5	4	Yes	
	0	5	5	4	No	
	3	3	4	3	Yes	
Z:5, Dr:2, Da:4	5	0	0	5	Yes	83.3%
	2	4	4	0	Yes	
	3	3	0	3	Yes	
	3	0	5	2	Yes	
	0	4	4	2	No	
	2	2	3	2	Yes	
Z:3, A:2, Da:5	5	5	0	0	Yes	71.4%
	5	0	0	4	Yes	
	2	4	4	0	Yes	
	3	3	0	3	Yes	
	3	0	5	2	Yes	
	0	4	4	2	No	
	2	2	3	2	No	
Z:3, A:4, Da:2	3	5	0	0	No	25%
	4	0	5	0	Yes	
	5	0	0	3	Yes	
	0	4	5	0	No	
	0	5	0	4	No	
	2	3	3	0	No	
	4	4	0	0	Yes	
	0	4	0	4	No	
	4	0	0	4	No	
	3	0	3	2	No	
	0	3	3	2	No	
	2	2	3	1	No	
A:3, Dr:6	4	5	0	0	Yes	28.6%
	5	0	0	4	Yes	
	2	4	4	0	No	
	3	3	0	3	No	
	3	0	5	2	No	
	0	4	4	1	No	
	2	2	3	2	No	

A-Archon, Dr-Dragoon, Z-Zealot, Da-Dark Templar.

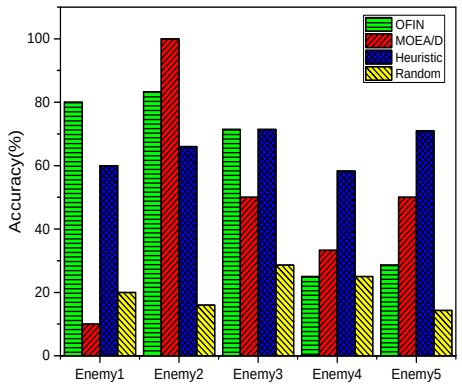


Fig. 10: Comparison of the recommended accuracy of four methods.

This is because that the units with special skills interact with other units more complexly, and we can't learn all the situations in which their skills are released through our replay data. As can be seen from Fig. 10, among the four methods, the recommended accuracy of OFIN in the experiments of Enemy1 and Enemy3 is the highest. In the second experiment of Enemy2, the accuracy of MOEA/D is 100%. However, it recommends only one team. Besides, the results recommended by MOEA/D in all experiments are unstable and they depend on the population size and the number of iterations. In the latter two experiments involving powerful units, although the heuristic strategy has the highest accuracy, the size of the team selected is relatively large since we always tend to choose a large team by experience. By contrast, the team selected by OFIN is relatively small. Overall, the proposed OFIN performs best when considering accuracy, stability and the team size at the same time. In summary, our new algorithm achieves satisfactory results for the ordinary units in StarCraft.

VI. CONCLUSION AND FUTURE WORK

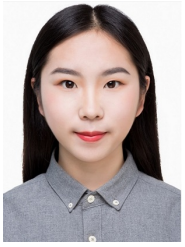
In this paper, we propose a team recommendation algorithm (OFIN), which combines NSGA-II with the order-based fuzzy integral in StarCraft. We initially solve the problem of how to select several winning teams from candidate units when an enemy team is given. Firstly, we use the normalized order-based fuzzy integral as a measure of the overall combat effectiveness of a team in RTS games. To calculate the fuzzy integral, we need to get the fuzzy measure in the fuzzy integral. Hence we collect a lot of replays of professional competitions of StarCraft as our training data and use GA to learn the fuzzy measure. Experiments show that the normalized order-based fuzzy integral can better approximate the replay data compared to the order-based fuzzy integral and it can better learn the fuzzy measure. Then, we formulate the team recommendation problem in StarCraft as a MOP with two objectives that maximize the normalized order-based fuzzy integral and simultaneously minimize the size of the team. After that, we solve the MOP using NSGA-II. Finally, we select the winning teams based on the fuzzy integral of Pareto solutions and the team size. We conduct several experiments in StarCraft and the results show that the OFIN algorithm can suggest teams that can win for a specified enemy troop with a high accuracy, and the scale of each recommendation is no larger than the size of the enemy. For units that are more powerful or have special skills, OFIN performs slightly worse. Therefore, OFIN has a good guiding significance for players to choose appropriate units in the face of ordinary enemy units in StarCraft.

OFIN cannot achieve good performance for some special units, and this is exactly one of the problems that we will try to address in the future. In addition, our current research assumes that the player knows the composition of the enemy army without considering the fog of war. However, due to the existence of the fog of war in a real-world actual game environment, players cannot see the formation of the enemy's army, and they cannot judge the enemy's attack time in advance. Therefore, in order to apply the OFIN algorithm to the actual game process, we will conduct more research in the future, including inferring the moment of the enemy's

attack and the army to be dispatched. Achieving a dynamic team recommendation is our ultimate goal.

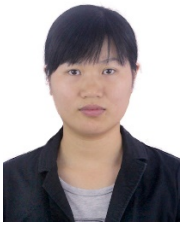
REFERENCES

[1] Y. Tian, Q. Gong, W. Shang, Y. Wu, and C. L. Zitnick, "Elf: An extensive, lightweight and flexible research platform for real-time strategy games," in Advances in Neural Information Processing Systems, San Mateo, CA, USA, 2017, pp. 2659–2669.
[2] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser et al., "Starcraft ii: A new challenge for reinforcement learning," arXiv preprint arXiv:1708.04782, 2017.
[3] P. Peng, Q. Yuan, Y. Wen, Y. Yang, Z. Tang, H. Long, and J. Wang, "Multiagent bidirectionally-coordinated nets for learning to play starcraft combat games," arXiv preprint arXiv:1703.10069, 2017.
[4] D. Churchill, A. Saffidine, and M. Buro, "Fast heuristic search for rts game combat scenarios," in Eighth Artificial Intelligence and Interactive Digital Entertainment Conference, Stanford, CA, USA, 2012, pp. 112–117.
[5] D. Churchill and M. Buro, "Portfolio greedy search and simulation for large-scale combat in starcraft," in 2013 IEEE Conference on Computational Inteligence in Games (CIG), 2013, pp. 1–8.
[6] A. Uriarte and S. Ontañón, "Game-tree search over high-level game states in rts games," in Tenth Artificial Intelligence and Interactive Digital Entertainment Conference, Raleigh, NC, USA, 2014, pp. 73–79.
[7] N. A. Barriga, M. Stanescu, and M. Buro, "Game tree search based on nondeterministic action scripts in real-time strategy games," IEEE Transactions on Games, vol. 10, no. 1, pp. 69–77, 2017.
[8] Y. Li, P. H. Ng, H. Wang, S. C. Shiu, and Y. Li, "Apply different fuzzy integrals in unit selection problem of real time strategy game," in 2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), Taipei, Taiwan, 2011, pp. 170–177.
[9] D. Fu and R. Houlette, "The ultimate guide to fsms in games," AI game programming Wisdom, vol. 2, pp. 283–302, 2004.
[10] S. OntaÃśÃşn, G. Synnaeve, A. Uriarte, F. Richoux, D. Churchill, and M. Preuss, "A survey of real-time strategy game ai research and competition in starcraft," IEEE Transactions on Computational Intelligence and AI in games, vol. 5, no. 4, pp. 293–311, 2013.
[11] P. Sun, X. Sun, L. Han, J. Xiong, Q. Wang, B. Li, Y. Zheng, J. Liu, Y. Liu, H. Liu et al., "Tstarbots: Defeating the cheating level builtin ai in starcraft ii in the full game," arXiv preprint arXiv:1809.07193, 2018.
[12] S. OntaÃśÃşn, K. Mishra, N. Sugandh, and A. Ram, "Case-based planning and execution for real-time strategy games," in Proceedings of the 7th International Conference on Case-Based Reasoning (ICCBR-07). Springer, 2007, pp. 164–178.
[13] S. Ontañón, K. Mishra, N. Sugandh, and A. Ram, "Learning from demonstration and case-based planning for real-time strategy games," in Soft Computing Applications in Industry. Springer, 2008, pp. 293–310.
[14] G. Synnaeve and P. Bessiere, "A bayesian model for opening prediction in rts games with application to starcraft," in 2011 IEEE Conference on Computational Intelligence and Games (CIG'11), Seoul, South Korea, 2011, pp. 281–288.
[15] N. Justesen and S. Risi, "Continual online evolutionary planning for in-game build order adaptation in starcraft," in Proceedings of the Genetic and Evolutionary Computation Conference. ACM, 2017, pp. 187–194.
[16] N. Justesen, P. Bontrager, J. Togelius, and S. Risi, "Deep learning for video game playing," IEEE Transactions on Games, 2019. [Online]. Available: http://dx.doi.org/10.1109/TG.2019.2896986
[17] K. Shao, Y. Zhu, and D. Zhao, "Cooperative reinforcement learning for multiple units combat in starcraft," in 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Honolulu, Hawaii, USA, 2017, pp. 1–6.
[18] N. A. Barriga, M. Stanescu, F. Besoain, and M. Buro, "Improving rts game ai by supervised policy learning, tactical search, and deep reinforcement learning," IEEE Computational Intelligence Magazine, vol. 14, no. 3, pp. 8–18, 2019.
[19] O. Vinyals, I. Babuschkin, J. Chung, M. Mathieu, M. Jaderberg, T. Ewalds, P. Georgiev, J. Agapiou et al., "Alphastar: Mastering the real-time strategy game starcraft ii," https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/, 2019.
[20] T. Murofushi, M. Sugeno, and M. Machida, "Non-monotonic fuzzy measures and the choquet integral," Fuzzy sets and Systems, vol. 64, no. 1, pp. 73–86, 1994.
[21] P. H. F. Ng, Y. Li, and S. C. K. Shiu, "New fuzzy integral for the unit maneuver in rts game," in The 5th International Conference on Pattern Recognition and Machine Intelligence (PReMI 2013). Kolkata, India: Springer, 2013, pp. 256–261.
[22] T. D. Nguyen, K. Q. Nguyen, and R. Thawonmas, "Heuristic search exploiting non-additive and unit properties for rts-game unit micromanagement," Journal of Information Processing, vol. 23, no. 1, pp. 2–8, 2015.
[23] A. H. Ng and S. C. Shiu, "Collaborative maneuver using fuzzy integral for rts game," in 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Istanbul, Turkey, 2015, pp. 1–8.
[24] C. M. Fonseca, P. J. Fleming et al., "Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization." in Icga, vol. 93, no. July. Citeseer, 1993, pp. 416–423.
[25] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," Evolutionary computation, vol. 2, no. 3, pp. 221–248, 1994.
[26] M. Kim, T. Hiroyasu, M. Miki, and S. Watanabe, "Spea2+: Improving the performance of the strength pareto evolutionary algorithm 2," in International Conference on Parallel Problem Solving from Nature. Springer, 2004, pp. 742–751.
[27] J. Knowles and D. Corne, "The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation," in Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), vol. 1. IEEE, 1999, pp. 98–105.
[28] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," IEEE transactions on evolutionary computation, vol. 6, no. 2, pp. 182–197, 2002.
[29] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," IEEE Transactions on evolutionary computation, vol. 11, no. 6, pp. 712–731, 2007.
[30] S. J. Louis and S. Liu, "Multi-objective evolution for 3d rts micro," in 2018 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2018, pp. 1–8.
[31] N. K. Adhikari, S. J. Louis, and S. Liu, "Multi-objective cooperative coevolution of micro for rts games," in 2019 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2019, pp. 482–489.
[32] R. Dubey, J. Ghantous, S. Louis, and S. Liu, "Evolutionary multi-objective optimization of real-time strategy micro," in 2018 IEEE Conference on Computational Intelligence and Games (CIG). IEEE, 2018, pp. 1–8.
[33] M. Sugeno, "Theory of fuzzy integrals and its applications," Doct. Thesis, Tokyo Institute of technology, 1974.
[34] W. A. Lodwick and J. Kacprzyk, "Studies in fuzziness and soft computing," in Fuzzy optimization: Recent advances and applications, vol. 254. Springer, 2010.
[35] Y. Li, P. H. Ng, H. Wang, S. C. Shiu, and Y. Li, "Applying fuzzy integral to performance evaluation in real time strategy game," in 2012 International Conference on Machine Learning and Cybernetics, vol. 1. IEEE, 2012, pp. 289–295.
[36] D. Churchill, "Sparcraft: open source starcraft combat simulation," http://code.google.com/p/sparcraft/, 2013.
[37] A. Kovarsky and M. Buro, "Heuristic search applied to abstract combat games," Conference of the Canadian Society for Computational Studies of Intelligence, vol. 3501, pp. 66–78, 2005.



LIN WANG received her B.S. degree in department of automation at Shandong University of Science and Technology in 2017. She is currently pursuing her Master's degree in department of automation at Xiamen University. Her research interest is team recommendation for games.

YIFENG ZENG received the Ph.D. degree from National University of Singapore, Singapore, in 2006.He is a Professor with the School of Computing,Teesside University, Middlesbrough, U.K. He is also an Affiliate Professor with Xiamen University,Xiamen, China. His research interests include intelligent agents, decision making, social networks, and computer games. Most of his publications appear in the most prestigious international academic journals and conferences, including Journal of Artificial Intelligence Research, Journal of Autonomous Agents and Multi-Agent Systems, International Conference on Autonomous Agents and Multi-Agent Systems, International Joint Conference on Artificial Intelligence, and Association for the Advancement of Artificial Intelligence.



BILIAN CHEN received her Ph.D. degree from The Chinese University of Hong Kong in 2012. Now she is an associate professor in Xiamen University. Her research interests include operational research, optimization theory and recommendation system.

YINGHUI PAN received her Ph.D. degree from Xiamen University in 2012. Now she is an associate professor in Jiangxi University of Finance and Economics, China. Her research interests include intelligent agents and machine learning. Her articles often appear in AAMAS, AAAI and other top AI conferences.

LANGCAI CAO received his Ph.D. degree in automation from Xiamen University in 2011. Now he is an associate professor in Xiamen University. His research interests include research and development of information system, process intelligence and machine learning.