Review of Recent Work on Computational Intelligence in Games

Fernando Fradique Duarte
University of Aveiro
Aveiro, Portugal
fjosefradique@ua.pt

Abstract—**Games provide an ideal environment for the development and testing of new techniques and technologies particularly in the field of Computational Intelligence. Techniques developed for game-playing are often transferred to other domains such as Psychology and Education further enhancing the scope of their use. Furthermore there is a significant commercial interest in the development of human-like game AI agents. This paper presents a review of the recent work on Computational Intelligence in Games focused primarily on the competitions hosted at the conference on Computational Intelligence and Games. The review provides background on each of these competitions and presents the most recent related work. Four of these competitions, namely the Fighting Game AI competition, the Ms. Pac-Man Vs. Ghost Team competition, the Hearthstone AI competition and the StarCraft AI competition are further reviewed in this regard and a brief summary of their evolution over time is also presented.**

Keywords—Computational Intelligence, Games, Artificial Intelligence, Machine Learning

I. Introduction

Games have been used in scientific research for quite some time particularly in the field of Computational Intelligence (CI) but also in other fields such as Psychology, Sociology and Education [1]. Early research in Artificial Intelligence (AI) used mostly classical two-player board games such as chess [2]. Recently video games have begun to attract such equal attention. One of the reasons video games are interesting in terms of research and particularly CI an AI stems from the wide range of challenges they pose with each genre of game posing its own set of challenges. As an example in Hearthstone a Collectible card game (CCG) the AI agent must deal with hidden information and uncertainty. As another example in StarCraft a Real Time Strategy (RTS) game the AI agent must be able to perform both micro-management tasks (e.g. control units in real-time) as well as macro-management tasks (e.g. plan a higher-level strategy).

Another reason of interest in terms of research as to do with the fact that the techniques developed for game playing can be used in other fields of research such as Education, Robotics and Brain-Computer Interfaces (BCIs). Also, there is significant commercial interest in the development of more sophisticated game AIs that act in a more natural way, adapting themselves to the changes occurring in the environment similarly to the way human players do [1].

Proof of this growing interest are the various conferences that have emerged over the years hosting several competitions related to research on CI and AI in video games such as the conference on Computational Intelligence on Games (CIG) and the conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE). This paper presents a review of the recent work on Computational Intelligence in Games focused primarily on the competitions hosted at the CIG conference. The review provides background on each of these competitions and presents the most recent related work. Four of these, namely the Fighting Game AI competition, the Ms. Pac-Man Vs. Ghost Team competition, the Hearthstone AI competition and the StarCraft AI competition are further reviewed in this regard and a brief summary of their evolution over time is also presented. These 4 competitions were chosen because of one or more of the following reasons: the fact that they are well established, relevant information (e.g. detailed competition results) was readily available, the base games are popular or have commercial versions and ultimately because they span several different game genres. The remainder of this document presents this review.

II. Fighting Game AI Competition

The objective of the Fighting Game AI [3] competition is to promote the research and development of general fighting game AIs that are able to play against various different opponents (other AIs or human players) in any play mode using any character data. In the remainder of this section further background information about the competition is provided as well as a brief overview of the most recent related work. A summary regarding the evolution of the competition over time is presented at the end of the section.

A. Background

Fighting games are a challenging game genre as they require the player to decide on which actions to perform from a set of possible actions in a very short interval of time. In FightingICE, the fighting game platform used for this competition there are 56 possible actions that the player can choose from within a required response time of 16.67 ms.

The FightingICE platform offers an environment for research that allows the design of flexible AIs. The setting of FightingICE takes place on a spatially limited two-dimensional stage. The 2 players or fighters (either human players or programmed AI agents) can move and perform attack and defense actions. Three different game characters can be used, each with its own set of unique skills (different effects achieved) and different requirements that must be met in order to perform these skills (for one of these characters, namely LUD, this data is unknown in advance).

Besides the character data just mentioned the AI agents also have access to the so called frame data containing information about the characters positions and their health points for example. This frame data however is given to the AI agents with a 15 frame delay instead of the current data in order to simulate the reaction delay of human players (this delay constraint can be circumvented for Visual-Based AIs). Agents must perform their actions through simulated key-inputs (imitating the input of a human player). By performing certain sequences of these actions within a given period of time, agents can perform combos to deliver additional damage to the opponent as well as combo-breakers (abort the opponent's combo).

There are 2 leagues in this competition, namely the Standard League and the Speedrunning League. In the Standard League the winner of a round between 2 AIs is the one with Health Points (HP) above zero at the time its opponent's HP has reached zero. In the Speedrunning League the competing AIs fight against a provided sample AI. The winner is the AI that can beat the sample AI in the shortest average time computed over 10 matches.

B. *Related Work*

Various different approaches have been proposed in order to deal with the challenges inherit to this competition. This section discusses some of this most recent work.

In [4] the authors propose a Hierarchical Task Network (HTN) as a planner in order to create sequences of actions or plans. This approach allows the AI agent to make decisions that can take into account long-term goals and high-level strategies allowing the creation of longer plans (i.e. plan sequences of actions further in advance) as opposed to just single actions (i.e. choose the optimal action for the current game state). By generating several alternative such plans the agent can react more suitably to the constant changes occurring in the environment.

The work presented in [5] proposes 4 Dynamic Difficulty Adjustment (DDA) AI agents implementing Monte Carlo Tree Search (MCTS) in order to tailor the difficulty of the game according to the player's skill in real-time and throughout game play as the skill level of the player progresses. In contrast to the more traditional approach were players are required to choose a difficulty level at the beginning of the game which is then kept unchanged until the end, these agents can dynamically change the strategies and behaviors of the opponent AIs or even the environment to better suit the current skill level of the player. This in turn allows the game to be more enjoyable and long-lived as it promotes a better immersion of the player.

Acknowledging the popularity of the MCTS approach in this competition and its limitations, mostly due to response time constraints (i.e. the opponent's behavior is predicted by taking into account only a few randomly selected actions (5) from the set of all possible actions on the opponent's side), the authors in [6] propose the use of an Action Table (AT) in order to encode the opponent's playing patterns and use this knowledge to predict the opponent's playing actions. This means that the AI agent can incorporate the observed playing patterns of its opponent into its decision process in order to further improve its performance against its opponent.

The authors of [7] use a Deep Convolutional Neural Network (CNN) in order to predict the actions of the opponent. This CNN however is trained on non-visual information (i.e. features) such as energy points and position and size of the characters as visual information (i.e. images of the game) is not provided to the AI agents in this competition. Several experiments were performed in order to find the best way to arrange and group these features as well as to compare the results obtained by this CNN with those obtained by a simple Neural Network (NN).

Genetic Programming (GP) is used in [8] in order to help automate the creation of the character fighting strategies. This allows the creation of a wider diversity of AI agents with different strategic skills. These in turn can be leveraged to incorporate more adaptive behavior into the game making it more engaging to the human player. As an example of this, this process can enable the creation of more interesting and realistic AIs (i.e. non-determinist) then those obtained using more established industry techniques such as manually coded Finite State Machines (FSMs) (i.e. deterministic).

In [9] the authors propose to model the opponent using the Neuro-evolution of Augmenting Topologies (NEAT) algorithm. The basic architecture of this algorithm is an Artificial Neural Network (ANN). This ANN is trained using the data collected from the opponent's actions and later used in order to predict the probability associated to each of the movements that the opponent can make so that the AI agent can determine the best countermeasure according to each situation. A Genetic Algorithm (GA) is used to optimize (i.e. change) the architecture of the ANN to better model the opponent over time.

In [10] the authors propose a GA in order to discover combos (sequences of attacks allowing a player to damage the opponent while preventing the opponent from performing any action). While combos are an important feature in many fighting games acting as a rewarding system to the precise execution of commands and thus encouraging the player to continue playing the game and improving his skills, they can also result in unexpected and undesirable behaviors such as the occurrence of long or infinite combos for example which can ruin the gaming experience. Results demonstrated that this approach was able to find combos under various different configurations, some of which (combos) were not known to be possible within the fighting game platform used to perform the tests.

C. *Evolution*

The Fighting Game AI competition started in 2013. In this first edition almost all of the participants (9) used rule-based AIs as their main approach. This approach (rule-based) continued to be the most popular amongst the participants on the next 2 editions in 2014 (6 entries for the 1 character competition) and 2015 (along with FSMs) although various other approaches began to emerge such as Opponent Modeling, GA and Fuzzy Logic. This scenario changed a bit in 2016 with the emergence of a new approach based on the combination of Rule-based algorithms with MCTS (6 entries against 5 for pure Rule-based). This approach dominated the top 3 ranking. Ever since then (also in 2017 and 2018) MCTS combined with several other techniques such as GA and Q-Learning has been the dominant approach used and as dominated the top ranking in terms of results. A brief summary of this evolution is depicted in Table I, showing the most used approach (MUA) as well as the winning approach (WA) and the number of participants (#E) for each year.

TABLE I. SUMMARY OF THE EVOLUTION OF THE COMPETITION IN TERMS OF THE APPROACHES USED OVER TIME

Year	MUA	WA	#E
2013	Rule-based	FSM	10
2014	Rule-based	Dynamic Scripting	10
2015	Rule-based	Rule-based	17
2016	Rule-based + MCTS	Rule-based + MCTS	13
2017	Rule-based + MCTS	MCTS	9
2018	MCTS	MCTS	7

III. Ms. Pac-Man Vs Ghost Team Competition

The goal of the Ms. Pac-Man Vs. Ghost Team [11] competition is twofold: on one hand the competition aims to promote research on cooperation between agents acting in a fairly complex environment in order to achieve a mutual goal (capture Ms. Pac-Man). On the other hand due to the adversarial nature of the Pac-Man game the competition also promotes research on strong AI agents that can get good results in the game (stay alive and score as much points as possible). In the remainder of this section further background information about the competition is provided as well as a brief overview of the most recent related work. A summary regarding the evolution of the competition over time is presented at the end of the section.

A. Background

The Ms. Pac-Man Vs Ghost Team competition started in 2016 and is a revival of the 2 previous competitions also based on the Ms. Pac-Man arcade game, namely the Ms. Pac-Man Screen Capture competition and the Ms. Pac-Man Vs Ghosts competition. An updated game engine, the introduction of Partial Observability (PO) constraints in the game and a new multi-agent approach to develop the ghost agents are some of the improvements introduced [12]. PO is a technique used to impair the ability of the player or agent to completely observe its environment.

Ms. Pac-Man, the arcade game on which the competition is based consists of 5 agents, Ms. Pac-Man and 4 ghosts interacting in a 2D maze environment. Explained in a very simplistic way the ghosts must try to capture Ms. Pac-Man while Ms. Pac-Man collects as much of the pills scattered on the corridors as possible in order to obtain a higher score. From time to time a reversal event occurs (e.g. Ms. Pac-Man eats a power pill) during which the ghosts change into the frightened mode and are instead chased and can be eaten by Ms. Pac-Man.

The competition is composed by 2 tracks. The first track concerns the development of a strong AI agent for Ms. Pac-Man that can operate under a PO constraint and score as many points as possible (stay alive, collect pills and eat as many ghosts as possible). The second track concerns the development of the AI agents for the 4 ghosts. These agents also operate under a PO constraint and must cooperate and try to coordinate their actions in order to prevent Ms. Pac-Man from consuming too much pills and ultimately trap and capture it.

B. Related Work

Various different approaches have been proposed in order to deal with the challenges inherit to this competition. This section discusses some of this most recent work.

In [13] the authors propose an approach to create varying versions of an agent with different playing styles and skills that behave in a designer-specified fashion. In order to achieve this, the authors propose the use of a Neural Network to model the agent. A variant of a generational GA with two-point crossover is then used to evolve these NNs in order to find the most suitable ones. These agents can be used for a variety of purposes such as to automatically collect game metrics, as opponents in multi-player games or to help the developers balancing the games' mechanics by acting as proxy human players with arbitrary skill levels.

The work presented in [14] proposes an approach to evolve a diverse and versatile set of cooperating agents that are able to adapt to the players' skill level using GP. The learning process is based on a combination of cooperative and adversarial coevolution, whereby the Pac-Man agent (1 population) competes against the Ghost Team, composed of 4 cooperatively evolving populations. This adversarial learning scheme enables both types of agents to be evolved simultaneously which in turn somewhat simulates the learning process of an actual human player (playing with the Pac-Man agent).

The authors in [15] propose Case-based Reasoning (CBR) and Reinforcement Learning (RL) techniques to train the agents. This training process is achieved via the use of the Q-learning algorithm. In this case however the Q-table (table used to store the maximum expected future reward for each action at each state) is replaced by a case base (collection of past experiences). The use of cases allows the injection of domain knowledge into the learning process (retrieve similar problems in the case base and adapt their solution to the current context) while also enabling a richer representation of the game state.

Recognizing the importance of human-like agents in order to improve the gaming experience (e.g. challenge or collaborate with the human player to help him achieve a goal or get started with the game) the authors in [16] conduct a study in order to try to understand if it is possible to distinguish between a human player and an AI agent and if so what are the features that best characterize how a players' behavior may be perceived as human-like or not. This study was conducted using 3 AI agents (a strong AI with good performance in the game, a simplified version of the strong AI and an AI presenting a totally randomized behavior) and 5 human players with different experience and skills over the course of 17 recorded games that were later presented to human judges so that these could deliberate whether the specific player was a human or not.

PO is used in various game genres such as in horror games to build suspense for example (sudden appearances of other agents). Used wrongly however, PO can induce negative emotional responses from the player such as anxiety, fear and frustration while playing the game. Given this in [17] the authors conduct some experiments in order to investigate the effect of varying levels of PO on difficulty and enjoyment in a Pac-Man game. AI agents are used in order to assess the effects on difficulty while human players are used to investigate the effects on enjoyment.

Finally in [1] the authors present a summary of the research conducted over the years using the Pac-Man game and variants thereof. The study focuses mainly on research conducted on the field of CI and presents the various approaches used by the participants in this competition and its predecessors over the years. This overview also highlights other fields of research where Pac-Man was used such as in Biology, Psychology, Robotics and Education.

C. *Evolution*

Competitions based on the Ms. Pac-Man arcade game started on 2007 with the Ms. Pac-Man Screen Capture competition (which ran until 2011). Later and building on the success of the Ms. Pac-Man Screen Capture competition the Ms. Pac-Man Vs Ghosts competition was launched and ran for four iterations between 2011 and 2012. In 2016 the Ms. Pac-Man Vs Ghost Team competition was launched featuring some changes relatively to the previous competitions such as an updated game engine and the introduction of PO. The introduction of these changes enriched the competition with a new set of challenges [1].

Over the course of these 3 competitions several different approaches were proposed such as [1] Rule-based, FSM, Tree Search, Monte Carlo (MC), Evolutionary Algorithms (EA), Neural Networks, Neuro-evolution and Reinforcement Learning. In [1] the authors review the literature on these approaches: Rule-based and FSMs can be found as early as 2003 mostly between 2008 and 2012 and still to a lesser degree in 2016 and 2017, Tree Search and MC can be found as early as 2009 mostly between 2010 and 2013 and to a lesser extent in 2014, EAs can be found as early as 1992 and mostly between 2010 and 2013 and also in 2014 and 2016, NNs can be found as early as 1999 and until 2010, Neuro-evolution can be found as early as 2005 and mostly in 2011, 2014 and 2016, lastly RL can be found as early as 2009 and mostly in 2015 and 2016 to a lesser degree. A brief summary of the results obtained in 2018 is depicted in Table II. This summary includes an overview of the winning approaches (WA), considering the top performing agents for both the Ms. Pac-Man controller (team P) and the Ghosts controllers (team G) as well as the number of participants (#E) for each track. It should be noted that very few entrants (only 4) participated on the second track of the competition, 2 of which were the default controllers provided, namely StarterGhostComm and StarterGhost.

TABLE II. SUMMARY OF THE COMPETITION RESULTS FOR 2018

Year	Team	WA	#E
2018	P	Modular Multi-objective (Hyper) NEAT	8
	G	Rule-based	4

IV. HEARTHSTONE AI COMPETITION

The objective of the Hearthstone AI [18] competition is to promote the development of fully autonomous AI agents that are able to play in gaming environments featuring uncertainty and hidden information such as in the context of the Hearthstone game. In the remainder of this section further background information about the competition is provided as well as a brief overview of the most recent related work. A summary regarding the evolution of the competition over time is presented at the end of the section.

A. *Background*

CCGs are a popular game genre. This game genre is also interesting for AI research due to the fact that players must deal with hidden information (the cards of the opponent are unknown) and uncertainty stemming from the vast number of possible combinations of states, rules and cards that may result in playing scenarios not even anticipated by the creators of the game.

In Hearthstone, the turn-based card video game used as the base platform for this competition, 2 players play against each other using pre-constructed decks of 30 cards and a selected hero with a unique power (draw a card, summon a minion, heal or deal damage). These cards differ for each hero (e.g. the Mage class offers more spells). Players use their limited mana crystals to draw cards in order to attack the opponent (e.g. cast spells or summon minions). The goal of the game is to reduce the opponents HP to zero.

The competition is composed by 2 tracks, namely the Premade Deck Playing track (PMD) and the User Created Deck Playing track (UCD). In the PMD track all participants receive a list of decks and playout all possible combinations against each other. The winner is determined by the average win rate. This track encourages research on agents that can use their own characteristics and the opponent's deck to win the game. The UCD track allows agents to define their own deck. This track encourages research on finding a deck that can consistently beat a vast amount of other decks and also on optimizing the agent's strategy according to the characteristics of their deck. Again the average win rate dictates the winner.

B. *Related Work*

Various different approaches have been proposed in order to deal with the challenges inherit to this competition. This section discusses some of this most recent work.

In [19] the authors propose a modification to the MCTS algorithm in order to handle randomness and tackle imperfect information. The authors also propose a heuristic (board solver) to tackle the combinatorial complexity of the game (due to the large number of possible attacks and the varying order in which the attacks may be performed). This heuristic generates a sequence of attacks given a game state. Finally and due to the weaknesses of MCTS when dealing with huge branching factors and delayed rewards resulting from the actions taken, the MCTS algorithm is combined with a value network heuristic, specifically a Deep Neural Network that given a state of the game computes the predictions of the game outcome. MCTS uses these predictions to foresee the outcome of a playout without having to simulate it until the end.

An important element of player engagement in card games is the periodical addition of new cards as they potentially provide new gaming strategies. Playtesting is the process used to check new card sets for design flaws The authors of [20] propose an Evolutionary Algorithm (EA) to automate this playtesting process (deckbuilding). The EA creates new card decks which are then played by an AI agent against human-designed decks in order to evaluate their effectiveness. The authors also propose a new heuristic mutation operator to the EA based on the way human players modify their decks in order to limit the space of possible decks.

In the work presented in [21] the authors propose an AI agent based on an Expert System (ES). A symbolic approach with a semantic structure acts as an ontology to represent the static descriptions of the game mechanics and the dynamic game state memories (representation of the current state and the actions that led to it). The amount of expert knowledge represented in the ontology such as popular moves and strategies is reduced as these should be derived by the agent using its knowledge base (static knowledge representing generic information about the game and dynamic knowledge describing the entities currently active on a game session). At runtime the agent uses rules and performs queries on the semantic structure to do reasoning and strategic planning.

In [22] the authors propose an ensemble of various NN models (including CNN) to predict the likelihood of the first player (assuming it is his turn to play) in winning the game given the representation of an arbitrary intra-game state. The datasets used were extracted from a collection of playouts between weak AI agents. The proposed method requires minimal domain knowledge and uses basic feature preprocessing to extract the minion, hero and aggregated features. For the NN models this set of features was further extended to include as additional features the square and logarithm of all features (except minion features and hero class).

The work in [23] proposes a Stacking Generalization (SG) model (various machine learning algorithms stacked) with 2 layers to predict which of 2 AI agents playing against each other will win the game based on the information known at the given time. A Bayesian approach was used in order to optimize the hyper-parameters of the several algorithms used. The final model proposed consists of a conditional model composed of 2 SG models. The first SG model was well optimized and fine-tuned and is used when the test data is Independent and Identically Distributed (IID) (no new cards present). When this is not the case (non-IID scenario) a second SG model, more conservative (not as fine-tuned) is used instead.

Finally in [24] the authors propose an AI agent based on a modified version of the MCTS algorithm that integrates expert knowledge of 2 types in its search process (choose adequate moves). The first type of domain specific knowledge consists of a database of decks that is used to handle imperfect information (the cards that the opponent holds are not known). The second type consists of a heuristic function used to guide the MCTS rollout phase (simulation of the game until a terminal state is reached) in order to reduce the search space of the game (possible moves). Two heuristics, constructed as linear combinations of the features extracted from the given state of the game were tested. The first heuristic included a small number of hand-picked features while the second included additional features.

C. *Evolution*

The Hearthstone AI competition started quite recently in 2018. Regarding the PMD track the top performing agents used simulation-search based algorithms such as MCTS or trained an evaluation function using EA. Concerning the UCD track the top performing agents used several different approaches including MCTS and Greedy EA.

TABLE III. SUMMARY OF THE RESULTS FOR 2018

Year	**Track**	**WA**	**#E**
2018	PMD	MCTS, EA	33
	UCD	MCTS, Greedy EA	17

A brief summary of the results obtained during the 2018 competition is depicted in Table III. This summary includes an overview of the winning approaches (WA), considering the top performing agents as well as the number of participants (#E) for each track.

V. STARCRAFT AI COMPETITION

The objective of the StarCraft AI [25] competition is to promote research on RTS game AI agents that are able to perform under uncertainty, manage resources and plan high-level strategies. In the remainder of this section further background information about the competition is provided as well as a brief overview of the most recent related work. A summary regarding the evolution of the competition over time is presented at the end of the section.

A. *Background*

RTS games are a challenging game genre as they require the player to handle resource collection (to produce units and buildings), manage the construction of units and buildings (i.e. choose build order) and battle against enemies (control a large number of units in real time). These tasks are often referred to as micro-management tasks (unit control) and macro-management tasks (higher-level game strategy of the player). Such scenarios pose challenges to AI due to their dynamic nature (uncertainty), their huge state-action spaces and the need for both short and long decision making [26].

StarCraft, the base platform used in the competition is a RTS game featuring a strategic military combat simulation. Each player controls 1 of 3 races and must gather resources to expand their base and produce more units (an army). The winner of the game is the player that manages to destroy his opponent’s base.

The competition is organized into a single track, where the participant agents play against each other (1 vs 1 game) in a round-robin tournament. Games are limited to simulate 1 hour of gameplay. The agent with the greatest win percentage over all the rounds played is the overall winner of the competition.

B. *Related Work*

Various different approaches have been proposed in order to deal with the challenges inherit to this competition. This section discusses some of this most recent work.

In [27] the authors propose Continual Online Evolutionary Planning (COEP), an evolutionary-based method capable of performing in-game (during the game) adaptive build order planning and optimization. COEP controls the macro-management tasks of the game (i.e. what builds to produce and in which order) allowing the agent to change its build order dynamically to quickly adapt to the opponent’s strategy. Four mutation operators, namely: clone (build at position *a* becomes the same as the build at position *b*), swap (2 builds swap positions in the build order), add (a build is inserted in the build order along with its requirements-other builds) and remove (a build is moved to the end of the build order) are implemented and used to effectively reorganize pre-existing build orders.

The work in [28] proposes Deep Learning to learn the macro-management tasks directly from game replays performed by highly skilled human players. In order to achieve this, replay files were processed and all events related to macro-management tasks such as material changes were extracted and used to simulate abstract StarCraft games via a build order forward model. The resulting action-state pairs obtained from the actions performed during these abstract games were then used in order to build the training dataset. A fully connected Neural Network was then trained on this dataset and used as the macro-management module of the AI agent.

In [29] the authors conduct a study over 5 algorithms used by AI agents playing StarCraft for resource gathering (choose resource locations in order to maximize the total amount of resources gathered). The algorithms tested are: Built-in (assign unit to go to the previous resource location), Mineral-lock (evenly distribute worker units over all resource locations), Queue-based Scheduling (attach queues to resource locations and designate free worker units to queues), Co-operative pathfinding (optimize the paths of the worker units to their designated resource locations), and Co-operative pathfinding + Queue (a combination of the previous 2 algorithms). The authors conclude that a trade-off between CPU time and resource gathering rate must be made (in general the algorithms that are more CPU intensive can gather more resources).

In [30] the author presents a review of CUNYbot, one of the entrants of the StarCraft: Brood War AI tournament. CUNYbot makes strategic decisions using a low-dimensional economic model (traditionally used to describe the behavior of countries). The parameters of this economic model are optimized between games via a GA algorithm in order to learn the capital/labor ratio for each of the built-in AI races. The bot also implements a reactive strategy based on the Cobb-Douglas model in order to model its opponents during the game and adapt its own strategy accordingly.

The work presented in [26] implements an agent able to pursue long-term plans that may require long sequences of actions to be achieved mimicking the usual behavior of human players in RTS games. In order to achieve this, the game state space is partitioned into a set of clusters (states with similar features) or abstract states. An option or a temporally-extended action in a Markov Decision Process (MDP) is then created for every abstract state and algorithm in the portfolio of game-playing algorithms (such algorithms receive the current state and output an action). The learning agent observes the abstract state and selects an option which acts according to its associated algorithm. The new state is observed as well as the reward received and the agent selects a new option repeating the learning process.

Finally in [31] the authors introduce the Deep RTS environment, an high-performance RTS game (and simulator) created specifically for AI research. Deep RTS supports accelerated learning (50,000 times faster compared to existing RTS games) and features a flexible configuration that enables research in several different RTS scenarios including partially observable state-spaces and map complexity. Deep RTS targets Deep Reinforcement Learning (DRL) research and aims to ease its use in more advanced games (e.g. ease the design of reward models).

C. Evolution

The StarCraft AI competition started in 2010. In terms of the most popular techniques used it is difficult to choose 1 given the plethora of diverse approaches that have been used by the participants such as HTN, Breath First Search (BFS) pathfinding, FSM, Greedy search, GA, MCTS, A*, NN (e.g. Long-short Term Memory (LSTM)) to name just a few. An interesting fact is the large increase on the number of bots using file I/O in order to adapt their strategies (file I/O is allowed and agents can save experience from the rounds they play and use that information to change their strategies for the next rounds) from 4 (2017) to 19 (2018). The number of bots using Machine Learning (ML) techniques for this same purpose (i.e. adapt their strategy) also increased a bit from 2 (2017) to 7 (2018). This may be an indication that participants are trying to devise agents that are more adaptive to the changes occurring in the environment.

A brief summary of the evolution of the competition in terms of the winning approaches (WA), considering the top 3 ranked entrants (and the information available) and the number of participants (#E) for each year is depicted in Table IV (previous years to 2013 are not considered).

TABLE IV. SUMMARY OF THE EVOLUTION OF THE COMPETITION IN TERMS OF THE WINNING APPROACHES OVER TIME SINCE 2013

Year	WA	#E
2013	Greedy Search	8
2014	FSM, Potential Flows	13
2015	FSM, Script-based	16
2016	LSTM, A* with Depth-first Search (DFS)	16
2017	Multi-agent, HTN	20
2018	HTN, BFS	27

VI. OTHER COMPETITIONS

This section presents an overview of the remaining competitions hosted at CIG. Due to space constraints this discussion is more high level. Nevertheless some background information about each of these competitions as well as their research challenges and most recent related work are provided.

A. Short Video Competition

The goal of this competition is to act as a source of interesting videos showcasing CI. The videos are presented in a plenary session and the winners are decided by the vote of the audience.

B. MicroRTS Competition

The goal of the microRTS [32] competition is to motivate research underlying the development of AI agents for RTS games while minimizing the amount of engineering required to participate so that participants can focus their efforts on the research aspects of the competition. Contrary to the StarCraft AI competition, in microRTS agents have access to a simulator (forward model) which they can use to simulate the effect of actions and plans, allowing planning techniques to be developed more easily. The competition is organized into 3 tracks: large state spaces and branching factors, partial observability and non-determinism. Examples of recent related work include several variants of MCTS [33]–[36], RL [37] and Evolutionary Multi-Objective optimization [38].

C. Hanabi Competition

The goal of the Hanabi [39] competition is to motivate research on AI agents capable of playing the cooperative partially observable card game Hanabi. The Hanabi card game is played by 2 to 5 players using a deck of 5 suits (colors) of cards. Players cannot see their own cards but they can see the other player's cards. The goal of the game is to play each suit in rank order (1 to 5). The competition is organized into the Mixed track (agents are paired with a group of unknown agents) and the Mirror track (agents are paired with copies of themselves). The agent that achieves the highest score over a set of unknown deck orderings wins the competition. Examples of recent related work include the use of GA [40], Rule-based [41] and a mixture of Rule-based with MCTS [42], [43].

D. General Video Game AI Competition

The goal of the General Video Game AI [44] competition is twofold: on one-hand promote research on the development of general video game playing controllers, that is a single AI agent capable of playing any game it is given without knowing beforehand which games are to be played and without using a simulator (forward model) for training. On the other hand also promote research on general video game content generation algorithms such as to generate levels for any game or playing rules for any level [45], [46]. The competition is organized in 4 tracks: Single Player Planning, 2-Player Planning, Level Generation and Rule Generation. Examples of recent related work include the use of Deep Reinforcement Learning [47], [48], MCTS [49], a mixture of Rule-based with MCTS [50] and Rolling Horizon Evolutionary methods [51].

E. Angry Birds Level Generation Competition

The goal of the Angry Birds Level Generation [52] competition is to promote research on building computer programs that are able to automatically generate fun and challenging levels for the Angry Birds physics-based puzzle game. The objective of the Angry Birds game is to kill all the pigs using the birds provided. In order to achieve this, the player uses a slingshot to shoot birds at block structures (and destroy them) with pigs placed within and around these structures. The levels generated should also be stable concerning gravity, robust in terms of the objectives of the game (a single action should not destroy large parts of the generated structure) and challenging enough while still being solvable. The game level generators are evaluated on the overall enjoyment of the levels they create. Examples of recent related work include the use of Procedural Level Generation algorithms [53], [54] and Pattern-Struct with Preset-Model [55].

F. Text-Based Adventure AI Competition

The goal of the Text-Based Adventure AI [56] competition is to promote research on AI agents that can play games with text-only interfaces. This competition can also potentially foster new developments in several research fields such as Natural Language Processing (NLP) and Automatic Model Acquisition. The Z-Machine, a text-based game engine is used in order to evaluate the developed AI agents. Agents are scored according to their score while playing an unseen game instance (the dominant criterion) and freedom from a priori bias. Examples of recent related work include the use of Language Models [57].

G. Visual Doom AI Competition

The goal of the Visual Doom AI [58] competition is to promote research on AI agents that can play Doom, a First Person Shooter (FPS) game, using solely the screen buffer (pixels) information to base their decisions upon. Although agents can be developed by using any technique, Machine Learning methods such as DRL are encouraged (also well supported by ViZDoom the Doom-based AI research platform used in the competition). The competition is organized into 2 tracks: single player (finish the Doom game) and multiplayer (compete with other agents in Doom deathmatches). Examples of recent related work include the use of DRL [59], AutoEncoders [60] and Reinforcement Learning with Curriculum learning [61].

Conclusion

This paper presented a review on the recent advances on Computational Intelligence in Games. Special focus was given to the competitions hosted at the CIG conference. These competitions were discussed in terms of their goals and the challenges they pose to researchers. The most recent related work was also presented. Four of these competitions were further reviewed in this regard. A brief overview of the evolution of each of these 4 competitions was also presented in terms of the approaches proposed by their participants over time, when such information was found relevant and available.

References

[1] P. Rohlfshagen, J. Liu, D. Perez-Liebana, and S. M. Lucas, "Pac-Man Conquers Academia: Two Decades of Research Using a Classic Arcade Game," *IEEE Trans. Games*, vol. 10, no. 3, pp. 233–256, 2017.

[2] R. Miikkulainen, B. D. Bryant, R. Cornelius, I. V. Karpov, K. O. Stanley, and C. H. Yong, "Computational intelligence in games," in *Computational Intelligence: Principles and Practice*, G. Y. Y. and D. B. Fogel, Ed. IEEE Computational Intelligence Society, 2006, pp. 155–191.

[3] "--," 2018. [Online]. Available: http://www.ice.ci.ritsumei.ac.jp/~ftgaic/.

[4] X. Neufeld, S. Mostaghim, and D. Perez-Liebana, "HTN fighter: Planning in a highly-dynamic game," in *Proceedings of the 9th Computer Science and Electronic Engineering Conference*, 2017, pp. 189–194.

[5] S. Demediuk, M. Tamassia, W. L. Raffe, F. Zambetta, X. Li, and F. Mueller, "Monte Carlo Tree Search Based Algorithms for Dynamic Difficulty Adjustment," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 53–59.

[6] M. J. Kim and K. J. Kim, "Opponent Modeling based on Action Table for MCTS-based Fighting Game AI," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 178–180.

[7] N. Duc Tang Tri, V. Quang, and K. Ikeda, "Optimized Non-visual Information for Deep Neural Network in Fighting Game," in *Proceedings of the 9th International Conference on Agents and Artificial Intelligence*, 2017, pp. 676–680.

[8] G. Martínez-Arellano, R. Cant, and D. Woods, "Creating AI Characters for Fighting Games using Genetic Programming," *IEEE Trans. Comput. Intell. AI Games*, vol. 9, no. 4, pp. 423–434, 2017.

[9] T. Kristo and N. U. Maulidevi, "Deduction of Fighting Game Countermeasures using Neuroevolution of Augmenting Topologies," in *Proceedings of 2016 International Conference on Data and*

Software Engineering, 2016.

[10] G. L. Zuin, Y. P. A. Macedo, L. Chaimowicz, and G. L. Pappa, "Discovering Combos in Fighting Games with Evolutionary Algorithms," in *Proceedings of the 2016 Genetic and Evolutionary Computation Conference*, 2016, pp. 277–284.

[11] "--," 2018. [Online]. Available: http://www.pacmanvghosts.co.uk.

[12] P. R. Williams, D. Perez-Liebana, and S. M. Lucas, "Ms. Pac-Man Versus Ghost Team CIG 2016 competition," in *Proceedings of the 2016 IEEE Conference on Computatonal Intelligence and Games*, 2016.

[13] M. Morosan and R. Poli, "Evolving a Designer-Balanced Neural Network for Ms PacMan," in *Proceedings of the 2017 9th Computer Science and Electronic Engineering*, 2017, pp. 100–105.

[14] A. Dockhorn and R. Kruse, "Combining Cooperative and Adversarial Coevolution in the Context of Pac-Man," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 60–67.

[15] F. Domínguez-Estévez, A. A. Sánchez-Ruiz, and P. P. Gómez-Martín, *Training Pac-Man bots using Reinforcement Learning and Case-based Reasoning*. CoSECivi, 2017.

[16] M. Miranda, A. A. Sánchez-Ruiz, and F. Peinado, *Pac-Man or Pac-Bot? Exploring Subjective Perception of Players' Humanity in Ms. Pac-Man*. CoSECivi, 2017.

[17] P. R. Williams, S. M. Lucas, and M. Fairbank, "The Effect of Varying Partial Observability in Ms. Pac-Man," 2018.

[18] "--," 2018. [Online]. Available: http://www.is.ovgu.de/Research/HearthstoneAI.html.

[19] M. Swiechowski, T. Tajmajer, and A. Janusz, "Improving Hearthstone AI by Combining MCTS and Supervised Learning Algorithms," in *Proceedings of the 2018 IEEE Conference on Computatonal Intelligence and Games*, 2018, pp. 445–452.

[20] P. García-Sánchez, A. Tonda, A. M. Mora, G. Squillero, and J. J. Merelo, "Automated Playtesting in Collectible Card Games using Evolutionary Algorithms: A Case Study in Hearthstone," *Knowledge-Based Syst.*, vol. 153, pp. 133–146, 2018.

[21] A. Stiegler, K. Dahal, J. Maucher, and D. Livingstone, "Symbolic Reasoning for Hearthstone," *IEEE Trans. Games*, vol. 10, no. 2, pp. 113–127, 2018.

[22] Ł. Grad, "Helping AI to Play Hearthstone using Neural Networks," in *Proceedings of the Federated Conference on Computer Science and Information Systems*, 2017, vol. 11, pp. 131–134.

[23] D. Deja, "Predicting Unpredictable Building Models Handling Non-IID Data, A Hearthstone Case Study," in *Proceedings of the Federated Conference on Computer Science and Information Systems*, 2017, vol. 11, pp. 127–130.

[24] A. Santos, P. A. Santos, and F. S. Melo, "Monte Carlo Tree Search Experiments in Hearthstone," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 272–279.

[25] "--," 2018. [Online]. Available: http://cilab.sejong.ac.kr/sc_competition2018.

[26] A. R. Tavares and L. Chaimowicz, "Tabular Reinforcement Learning in Real-Time Strategy Games via Options," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 229–236.

[27] 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*, 2017, pp. 187–194.

[28] N. Justesen and S. Risi, "Learning Macromanagement in StarCraft from Replays using Deep Learning," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 162–169.

[29] M. L. M. Rooijackers and M. H. M. Winands, "Resource-Gathering Algorithms in the Game of StarCraft," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 264–271.

[30] B. S. Weber, "Standard Economic Models in Nonstandard Settings - StarCraft: Brood War," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 417–424.

[31] P. Andersen, M. Goodwin, and O.-C. Granmo, "Deep RTS: A Game Environment for Deep Reinforcement Learning in Real-Time Strategy Games," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 149–156.

[32] "---," 2018. [Online]. Available: https://sites.google.com/site/micrortsaicompetition/home.

[33] A. Uriarte and S. Ontanon, "Single Believe State Generation for Partially Observable Real-Time Strategy Games," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 296–303.

[34] S. Ontañón, "Combinatorial Multi-armed Bandits for Real-Time Strategy Games," *J. Artif. Intell. Res.*, vol. 58, no. 1, pp. 665–702, 2017.

[35] N. A. Barriga, M. Stanescu, and M. Buro, "Game Tree Search Based on Non-Deterministic Action Scripts in Real-Time Strategy Games," *IEEE Trans. Games*, vol. 10, no. 1, pp. 69–77, 2018.

[36] Z. Yang and S. Ontanon, "Learning Map-Independent Evaluation Functions for Real-Time Strategy Games," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 301–307.

[37] P. A. Andersen, M. Goodwin, and O. C. Granmo, "Deep RTS: A Game Environment for Deep Reinforcement Learning in Real-Time Strategy Games," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 149–156.

[38] R. Dubey, J. Ghantous, S. Louis, and S. Liu, "Evolutionary Multi-objective Optimization of Real-Time Strategy Micro," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 133–140.

[39] "--," 2018. [Online]. Available: https://comp.fossgalaxy.com/competitions/t/11.

[40] R. Canaan, H. Shen, R. Torrado, J. Togelius, A. Nealen, and S. Menzel, "Evolving Agents for the Hanabi 2018 CIG Competition," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 409–416.

[41] M. Eger, C. Martens, and M. A. Cordoba, "An Intentional AI for Hanabi," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 68–75.

[42] M. J. H. van den Bergh, A. Hommelberg, W. A. Kosters, and F. M. Spieksma, *Aspects of the cooperative card game Hanabi*. Amsterdam, The Netherlands: Springer, 2016.

[43] J. Walton-Rivers, P. R. Williams, R. Bartle, D. Perez-Liebana, and S. M. Lucas, "Evaluating and modelling Hanabi-playing agents," in *Proceedings of the 2017 IEEE Congress on Evolutionary Computation*, 2017, pp. 1382–1389.

[44] "--," 2018. [Online]. Available: www.gvgai.net.

[45] K. Ahmed, M. C. Green, P.-L. Diego, and J. Togelius, "General Video Game Rule Generation," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 170–177.

[46] D. Perez-Liebana, J. Liu, A. Khalifa, R. D. Gaina, J. Togelius, and S. M. Lucas, "General Video Game AI: a Multi-Track Framework for Evaluating Agents, Games and Content Generation Algorithms," *CoRR*, vol. abs/1802.1, 2018.

[47] W. Woof and K. Chen, "Learning to Play General Video-Games via an Object Embedding Network," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 285–292.

[48] R. R. Torrado, P. Bontrager, J. Togelius, and J. Liu, "Deep Reinforcement Learning for General Video Game AI," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 316–323.

[49] C. F. Sironi and M. H. M. Winands, "Analysis of Self-Adaptive Monte Carlo Tree Search in General Video Game Playing," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 396–400.

[50] A. Dockhorn and D. Apeldoorn, "Forward Model Approximation for General Video Game Learning," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018.

[51] R. D. Diego Perez-LiebanaGaina and S. M. Lucas, "Rolling Horizon Evolution Enhancements in General Video Game Playing," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 88–95.

[52] "--," 2018. [Online]. Available: https://aibirds.org/other-events/level-generation-competition.html.

[53] M. Stephenson and J. Renz, "Procedural Generation of Levels for Angry Birds Style Physics Games," in *Proceedings of the 2016 IEEE*

Conference on Computational Intelligence and Games, 2016, pp. 1–8.

[54] M. Stephenson and J. Renz, "Generating Varied, Stable and Solvable Levels for Angry Birds Style Physics Games," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 288–295.

[55] Y. Jiang, T. Harada, and R. Thawonmas, "Procedural Generation of Angry Birds Fun Levels Using Pattern-Struct and Preset-Model," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 154–161.

[56] "--," 2018. [Online]. Available: http://atkrye.github.io/IEEE-CIG-Text-Adventurer-Competition/.

[57] B. Kostka, J. Kwiecieli, J. Kowalski, and P. Rychlikowski, "Text-based Adventures of the Golovin AI Agent," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 181–188.

[58] "--," 2018. [Online]. Available: http://vizdoom.cs.put.edu.pl/.

[59] K. Shao, D. Zhao, N. Li, and Y. Zhu, "Learning Battles in ViZDoom via Deep Reinforcement Learning," in *Proceedings of the 2018 IEEE Conference on Computational Intelligence and Games*, 2018, pp. 389–392.

[60] S. Alvernaz and J. Togelius, "Autoencoder-augmented Neuroevolution for Visual Doom Playing," in *Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games*, 2017, pp. 1–8.

[61] Y. Wu and Y. Tian, "Training Agent for First-person Shooter Game with Actor-critic Curriculum Learning," in *Proceedings of the 5th International Conference on Learning Representations*, 2017.