IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 12, 2021, accepted September 21, 2021, date of publication September 27, 2021, date of current version October 12, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3115768

A Systematic Review of Coevolution in Real-Time Strategy Games

EHAB Z. ELFEKY[1], (Member, IEEE), SABER ELSAYED[1], (Member, IEEE), LUKE MARSH[2], DARYL ESSAM[1], MADELEINE COCHRANE[2], BRENDAN SIMS[2], AND RUHUL SARKER[1], (Member, IEEE)

[1]School of Engineering and Information Technology, University of New South Wales Canberra, Australian Defence Force Academy, Canberra, ACT 2600, Australia

[2]Australian Department of Defence, Defence Science and Technology Group, Edinburgh, SA 5111, Australia

Corresponding author: Ehab Z. Elfeky (e.elfeky@unsw.edu.au)

This work was supported by the Australian Department of Defence, Defence Science and Technology Project, under Grant RG204054.

ABSTRACT Real-time strategy (RTS) games are a subgenre of strategy video games. Due to their importance in practical decision-making and digital entertainment over the last two decades, many researchers have explored different algorithmic approaches for controlling agents within RTS games and learning effective strategies and tactics. Among the techniques, coevolutionary algorithms proved to be one of the most popular and successful algorithms for developing such games, in which players can compete or cooperate to achieve the given game's mission. However, as many alternative designs exist with their analysis and the applications reported in diverse publications, a review covering the evolution of such algorithms would be valuable for researchers and practitioners in this domain. This paper aims to provide a systematic review by highlighting why and how coevolution is used in RTS games and analysis of the recent work. The review conducted follows procedural steps to identify, filter, analyse and discuss the existing literature. This structured review articulates the purposes of using coevolution in RTS games and highlights several open questions for future research in this domain.

INDEX TERMS Coevolution, real-time strategy games, computational intelligence, evolutionary computation, systematic review.

I. INTRODUCTION

In 2017, the annual revenue of the global video games industry passed 110 Billion USD [35]. In recent years, between 2017 and 2020, the video game industry has generated the highest annual revenue and revenue growth compared to all other forms of media entertainment industries [78]. Video games can be classified into five genres: traditional games (i.e., puzzle and board games), simulation games (i.e., sport and racing games), strategy video games (i.e., planning strategies, such as Chess and Starcraft), action video games (i.e., player actions are connected to the game environment, such as Tomb Raider and Resident Evil), and lastly fantasy games (i.e., players solve the game while they explore its environments such as Myst and Syberia) [16].

(Strategy Video Games): Strategy video games mainly involve tasks that require tactical planning and actions such as creating buildings, managing resources and supervising many units such as Chess, Starcraft and Warcraft [59]. Strategy video games can be subdivided based on the actions permitted for players over a given period of time inside the game; that is, either turn-based games (TBS) or real-time strategy (RTS). In the former, every player is given a dedicated period of time to put their strategy into action. Players are not allowed to take any action during the allocated time for other players. On the other hand, players in RTS games have no such restrictions on when they implement their strategies. In other words, players can do as many actions as they like even though other players have decided to go with a no actions' strategy [36]. Although developing an RTS game environment is challenging, a recent study showed that researchers developed equivalent game environments for RTS and TBS games [3]. Such attention from researchers reflects the popularity and the importance of RTS games, which is considered the focus of this paper.

(RTS and Coevolution): In RTS games, players are given one or more tasks to accomplish within the game environment with different levels of challenges, such as: levels, maps,

The associate editor coordinating the review of this manuscript and approving it for publication was Danilo Pelusi.

game rules, textures, stories, items, quests, music, weapons, vehicles and characters [68]. In some scenarios, when developing a commercial RTS game, the target is to create more challenging and attractive game contents as well as more sophisticated game AI, also referred to as non-player characters (NPCs) [21], [36]. Military games, in other cases, are developed to explore different strategies in defence/attack situations [31]. Researchers have tried different approaches to control the agents within such RTS games and to learn effective strategies and tactics; amongst them, coevolution was shown to perform well [26].

(Brief Introduction to Coevolution): Generally speaking, in the evolutionary computation (EC) domain, coevolution refers to evolving individuals with a shared goal and different roles in single or multiple populations [77]. Motivated by the work in [4], where the authors introduced a competitive fitness function to evolve strategies in a turn-based game called Tic Tac Toe, Reynolds [63] introduced the first coevolution technique in RTS games. He designed a coevolutionary system that could approach quality evolved players without reaching the known optimality, supported by genetic programming (GP) as the evolutionary methodology. The technique obtained a type of Pursuit-Evasion game called *Game of Tag*. Since then, many researchers have investigated the utilisation of coevolution features, such as competing populations with coupled fitness to fit in modelling RTS games, especially those with two opposing players (e.g. predator and prey games) [54].

(Related Reviews): To the best of the authors' knowledge, no research has been conducted to review coevolutionary algorithm (CA) approaches in RTS games. The two most relevant studies in the literature date back to 2013; (1) Lara-Cabrera *et al.* in [36] reviewed the computational intelligence (CI) applications in RTS games by covering existing approaches that deal with RTS games, their challenges and possible remedies; (2) Ontanon *et al.* [55] conducted a limited survey on how to use AI in a Starcraft game (a kind of RTS game). Due to the lack of a comprehensive review and the importance of coevolution in handling RTS games, this paper conducts a systematic review that includes how coevolution is used in RTS games and discussions on the unanswered questions in this field. The review follows a systematic approach as described in [74].

(Paper Organisation): This paper is organised as follows: Section II covers a theoretical background regarding RTS games and coevolution individually, while Section III describes the methodology used to carry out this systematic review and its significant findings. And lastly, Sections IV and V draw out the future work recommendations and conclusions, respectively.

II. THEORETICAL BACKGROUND

This section gives an overview of RTS games and coevolution.

A. RTS GAMES

(Game Number of Players): While most RTS games take place between two players, games can range from a single player, as in [38], [57], to being massively multiplayer, as in [61], [66], [67]. In some two-player games, each player has control over several homogeneous/heterogeneous units, and makes a decision on behalf of them; in this case the game takes place between two teams of players.

(Game Tasks): As previously mentioned, in RTS games, players are given a task to accomplish, such a task can be seen as managing resources to achieve a predefined goal(s). Managing a resource can be something like collecting/spending material resources, such as gold, producing/positioning units (i.e., tanks and aeroplanes), training/forming agents (i.e., soldiers and workers), constructing buildings, upgrading technologies, scouting/controlling map zones, and generating good build orders [36], [65].

(Game Management Depth Level): A player task in RTS games can be categorised based on their management level into two groups (1) macromanagement; and (2) micromanagement. The former requires long-term planning, such as gathering/spending resources, producing units, constructing buildings, upgrading technologies, scouting map zones, and generating good build orders. Usually, the better macromanagement skills players have, the larger army and/or economy can be achieved. In contrast, micromanagement tasks require short-term planning, such as unit formation/positioning planning using efficient communication between units. Usually, the better micromanagement skills players have, the longer their units stay alive in both attack and defence modes [1], [12], [36], [40], [55].

(Challenges in RTS): Over the last two decades, researchers have reported many challenges associated with RTS [14], [36], [55], which are summarised below:

- *Resource Management:* Part of RTS games is to gather resources from the local environment and to use them in creating/upgrading units, weapons, and buildings. Players are required to gain a competitive resource management strategy compared to their opponents.
- *Decision-Making Under Uncertainty:* Unlike most TBS games, RTS games are partially observable; where players neither have complete information about the game environment nor their opponents' formation and actions. As a result, players need to have a prediction model with an acceptable error margin.
- *Spatial and Temporal Reasoning:* RTS games depend mainly on a game map. Recognition of both player's and building's location on the map and the distances between them, including their elevation, affects players' actions and their impact. Moreover, temporal reasoning plays an essential role in both tactical and strategic reasoning.
- *Collaboration:* In many RTS games, players are expected to build their own army/team. Collaboration is required to take place between different units owned by the player to yield actions and tactics that are able to achieve the desired goal of the game.

- *Adversarial Real-Time Planning:* As explained earlier, there are no playing turns in RTS games. This makes it more challenging for players as their opponent(s) may not wait for their action, and so they may change the game before the other player(s) react.
- *Opponent Modelling and Learning:* Like other games, players in RTS games need to be able to model their opponents' actions and behaviour by utilising the game's history.
- *Path Planning:* Finding a path between two points on the map is a challenging task in RTS games. Players need to take many factors into consideration including the number of agents and units, their formation, the location of a dynamic opponent's units, and the game environment's obstacles.
- *Content Generation:* Anything apart from NPC's behaviour and the game engine that affects playing inside the game is considered as game content, such as: maps, resources, obstacles, weapons, units, and buildings. Having such a variety of game contents is expected to produce an almost unlimited number of different games.
- *Planning:* Ontanon *et al.* [55] encapsulated some aspects of the following challenges: Adversarial real-time planning, resource management, and collaboration. He utilised the predefined decision management level (Macro and Micro management) to categorise these challenges in two levels: Long-term planning and short-term planning.
- *Learning:* Ontanon *et al.* [55] considered learning as a challenge for RTS games and categorised it into three different forms:
  - *Prior Learning:* significant work in the literature has utilised prior information such as game maps.
  - *In-Game Learning:* reinforcement learning (RL) and NPC's modelling has been utilised to enhance players performance while playing a game.
  - *Intergame Learning:* utilising the experiences gained from prior games to increase the capability of playing RTS games.
- *Domain Knowledge Exploitation:* Utilising domain knowledge (strategies, build orders, replays, etc.) to develop better evaluation functions.
- *Task Decomposition:* Problem dimensionality is another a challenge in RTS games. Although researchers recently tried to handle this issue by breaking down this problem into a set of smaller sub-problems, such as tactics, reactive control, strategy, terrain analysis, and intelligence gathering. Finding the optimal decomposition of such tasks itself is a challenge.

B. COEVOLUTION

(Coevolution Definition & Types): Coevolution takes place when two different populations (species) evolve in parallel, where each population's fitness function is affected by shared information from the other population [20], [62]. When this shared information positively drives the receiving population's fitness value, it will be considered as cooperative coevolution. This can be seen as increasing the fitness values of one species leading to improved fitness values of the other species. In contrast, competitive coevolution takes place when the shared information affects the receiving population's fitness function negatively [20], [52], [62].

(Competitive Fitness): When one species evolves a good strategy that increases its fitness at the same time as driving the other species' fitness down, the other species will not stay silent. The corresponding evolutionary algorithm (EA) will evolve a counter strategy to improve the situation of this species in the game. Such behaviour keeps going and pushes both species to improve their own strategies yielding what is called an arms race [1], [26], [31], [34], [75]. The successful CA seeks to preserve such behaviour for the longest time possible. In other words, it aims to give enough time for each species to go through many loops where each species develops its own strategies, tests it against other species, receives the opponents' reaction and updates its strategies accordingly [26].

(Memory Mechanism): One of the main challenges in coevolution is forgetfulness; this usually happens when individuals with favourable behaviour do not get good chances for reproduction into the next generations [27]. Populations with a limited number of individuals suffer from short-term memory; individuals in such populations are required to be a winner in the selection process at every generation to secure a place in the next generation. As part of EAs, elitism was used implicitly to serve as short-term memory. In 1997, Rosin and Belew [64] introduced a long-term memory mechanism for a coevolutionary system called Hall-of-Fame (HoF), in which the best player in every generation joins this hall to test other species' players. This concept can also serve to preserve currently unfit individuals with favourable behaviour for later generations when the opponent changes their playing strategy. The mechanism of moving individuals between both short-term and long-term memory was managed later to emulate the psychology of human memory [6]. Moreover, some researchers considered a more precise model of the human brain to add another type of memory (Ultra Short-Term Memory) to the existing two [76].

(Coevolutionary Pathologies): Maintaining an emergent behaviour, such as an arms race for a long period of time is not an easy task due to other behaviours, referred to in the literature as coevolution pathologies. The most recognised ones are:

- *Loss of Gradients:* When all the individuals in a species have the same capability of winning or losing against other species' individuals, the evolutionary process stops as there are no more improvements expected [62].
- *Over-Specialisation:* Specialisation itself is a desired behaviour, where a population's members are well-trained to overcome and win against another population's members. However over-specialisation happens

when this winning capability cannot be generalised to new environments [18], [62].

- *Red Queen Dynamics (Also Known as Mediocre Stable State):* This effect takes place when changes in the competing population leads to instability in the fitness landscape. Such effects stop competing populations from evolving new strategies, instead, they just keep going through loops of actions and reactions with no improvement [26], [28], [50], [54], [62].

(Coevolution Pathologies Solutions): Researchers tried to mitigate the negative effect of such pathologies to allow higher chances of continuing the emerging arms race during the evolutionary process. As previously mentioned, using long-term memory was one of those attempts in which a memory archive, called Hall of Fame, was introduced to preserve the best solutions so far in order to keep the gradient of improvements [1], [9], [11], [37], [51]–[54], [62], [73]. Researchers also imported the concept of shaping from behavioural psychology into the field of learning. By altering some factors in the learning problem, the learner can converge towards some desired behaviours in an environment such as RL [24] and Strategy Games [71]. Coevolution includes implicit shaping by changing the fitness function to include the competitive and/or cooperative effect. Some researchers added an extra level of explicit fitness shaping by changing both the fitness function and the environment during evolution to overcome those pathologies and support the arms race [18], [20], [22]. Another approach was to use competitive fitness sharing, which gives higher fitness for those individuals capable of defeating opponents that did not lose many games. Also, there was an attempt to restrict individuals' interactions to only those in a proximity location on a map [62].

III. SYSTEMATIC REVIEW

In 2014, Borrego *et al.* [13] suggested a methodology, which has seven steps, to conduct a systematic literature review in engineering education. Due to its well regarded approach, this paper follows Borrego's methodology, as discussed in this section.

A. DECIDING TO DO A SYSTEMATIC REVIEW

In their research, Borrego *et al.* recommended to use Petticrew and Roberts questions in [58] to decide whether a systematic literature review is suitable. One of those questions was "*When is an accurate picture of past research, and past methodological research required to promote the development of new methodologies*?", which exactly matches with this review's authors situation. We plan in the future to develop a CA to solve a weapon target assignment problem in an RTS game environment. This is the purpose of carrying out this review.

B. SCOPE AND RESEARCH QUESTIONS

The scope of this systematic review is to investigate existing publications, reported in English, for those studies that have used coevolutionary approaches in RTS games. Two research questions have driven this review as follows:

- How was coevolution used in RTS Games?
- What are the remaining open questions in this regard?

TABLE 1. Keyword groups.

Group 1	*Group 2*	*Group 3*
"Coevolution"	"Game"	"RTS"
"Co-evolution"		"Real-Time Strategy"
"Coevolutionary"		"Videogame"
"Co-evolutionary"		"Video game"

C. INCLUSION AND EXCLUSION CRITERIA

An inclusion criteria was used first to widen the search to minimise the number of missed target papers. Afterwards, the result set is filtered by applying exclusion criteria to exclude irrelevant articles.

- Inclusion Criteria
  - *Database Selection:* Two criteria are considered: the first is the customised search capabilities, which ensure that the search is more precise to avoid missing relevant publications. The second criterion is the percentage of publications covered in the target field, which gives more access to relevant publications.
  - *Search Keywords:* The following combinations of customised search keywords are used; that is three groups of keywords, as per Table 1, are considered, with every group containing relevant keywords. For any publication to be considered in this review, it should have to have at least one keyword from Group 1 and the keyword in Group 2 in the publication's title, abstract, or keywords. In addition, at least one keyword from Group 3 should appear in any field for the publication on the reference database, i.e. references, source title, etc.
  - *Subject Area:* Engineering, Computer Science, Decision Sciences, and Multidisciplinary.
  - *Publication Type:* Article, conference proceeding, and book chapter.
  - *Publication Year:* No Conditions.
  - *Publication Language:* Only English.
- *Exclusion Criteria:* Any publication with any of the following criteria is excluded from the review.
  1) *Full Text Availability:* The full text is not available.
  2) *Game Type:* The work considers TBS games, i.e. is not an RTS game.
  3) *Coevolution:* Coevolution is not used.

D. FINDING AND CATALOGING SOURCES

Google Scholar, Web of Science, and Scopus are the most commonly used databases in this research field. From the authors previous experience, Google Scholar has the lowest customised search options, while Scopus and Web of Science have similarly high options. However, the recent comparison conducted in [43] showed that Google Scholar has the best

coverage among the three databases, while Web of Science has the lowest.

As a consequence and to get the best results, we decided to search for relevant publications in two stages: (1) a database search, considering the criteria mentioned earlier, was done using the Scopus database to utilise its strong customised search capabilities, (2) citation searching (also known as snowball sampling) was conducted using Google Scholar to exploit its strong publication coverage. In the second stage, the references cited by previously identified publications in the first stage were scanned manually and searched for using Google Scholar and were then added to the review list if they did not meet any of the exclusion criterion.

On 15 March 2021, we searched Scopus using the following advanced query. ''(TITLE-ABS-KEY((''Coevolution'' OR ''Co-evolution'' OR ''Coevolutionary'' OR ''Co-evolutionary'') AND (''Game''))) AND (ALL(''RTS'' OR ''Real-Time Strategy'' OR ''Videogame'' OR ''Video game'')) AND (LIMIT-TO (SUBJAREA, ''COMP'') OR LIMIT-TO (SUBJAREA, ''ENGI'') OR LIMIT-TO (SUBJAREA, ''DECI'')) AND (LIMIT-TO (DOCTYPE, ''cp'') OR LIMIT-TO (DOCTYPE, ''ar'') OR LIMIT-TO (DOCTYPE, ''ch'')) AND (LIMIT-TO (LANGUAGE, ''English''))''.

Of the 53 publications discovered, the full text of only one publication was inaccessible, and it was excluded. From the remaining 52, 15 publications did not cover RTS Games and three did not use coevolution, which resulted in a total of 34 publications. Citation searching in Google Scholar was able to provide 19 more publications, which means that the next stage in the review started with 53 publications at hand, as explained in figure 1.

It is worth mentioning that two of the games used in the reviewed papers were turn based, rather than actual RTS games. Bellus Bellum Gratia (BBG) was used in [33], [34], and Planet Wars, the Google AI challenge introduced in 2010, was used in [18], [25], [53]. These papers are still considered in this review as the games used here were adjusted and equipped with features that made them closer to RTS games than TBS games, as will be explained later.

E. CRITIQUE AND APPRAISAL

By scanning the resultant 53 publications, it was found that the purpose of writing those publications could be categorised into the following three reasons:

- to analyse/review an existing system(s) that was used in RTS games.
- to propose a new system to use coevolution in RTS games.
- to update an existing system to use coevolution in RTS games.

Based on this classification, 10 of the reviewed papers were found to fall within the first category and 43 within the second and third categories collectively. The analysis and review publications in the first category are discussed in this section, while the remaining 43 publications are considered the core

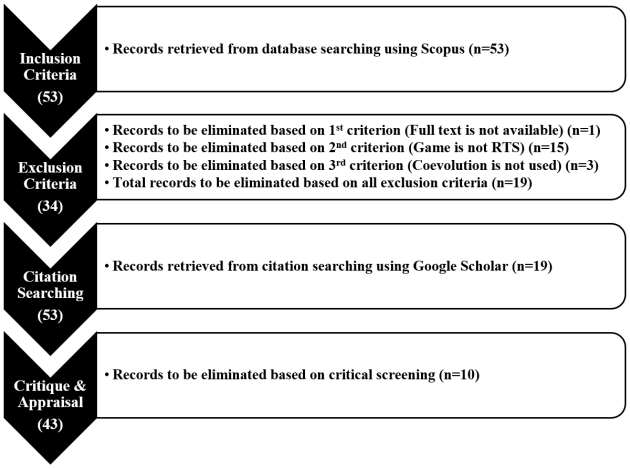

FIGURE 1. Steps of sources finding and the relevant numbers of each stage.

of this review and are handled in details in the following sections.

(Analysis/Review Papers): Miller and Cliff created two technical reports about the coevolution of pursuit and evasion: in the first one [47], the focus was on reviewing the importance and of using coevolution in such a game environment. They reviewed the foundation of this field from both biological and game-theoretical perspectives, such as player's biological ubiquity and mixed strategies' optimality proofs. The focus of the second report [17] was on the simulation methods that can be used (four simulation methods reviewed) and their performance. Buro [14] in 2003 summarised the challenges in RTS games and called on researchers to invest more efforts in such a promising field. In 2005, Livingstone [41] reviewed the studies that used coevolution with RTS games that have hierarchical decomposition that is similar to those in real armies. He reported that although the reviewed studies enable learning within a single layer of decision-making, they generally do not include learning across the given hierarchy. Two review papers were introduced in 2013: Lara-Cabrera *et al.* in [36] reviewed CI applications in RTS games by covering existing CI approaches, the challenges and the solutions. Ontanon *et al.* [55] limited their survey to cover how AI is used in a RTS game (Starcraft). More recently, researchers started reviewing and describing existing environments to design RTS games and related AI players. In 2017, Cook *et al.* [20] reviewed the architecture of an automated game designer that uses cooperative coevolution, called ''ANGELINA'', and discussed how it works. In 2018, Danette Allen [2] proposed a modelling and simulation environment called ''Baseline Environment for Autonomous Modelling''. This NASA environment is designed to support Explainable Artificial Intelligence, which concentrates on producing understandable AI systems. In 2019, Arulkumaran *et al.* [5] highlighted the components of an artificial intelligence (AI) system called AlphaStar, which is the first of its type to beat a

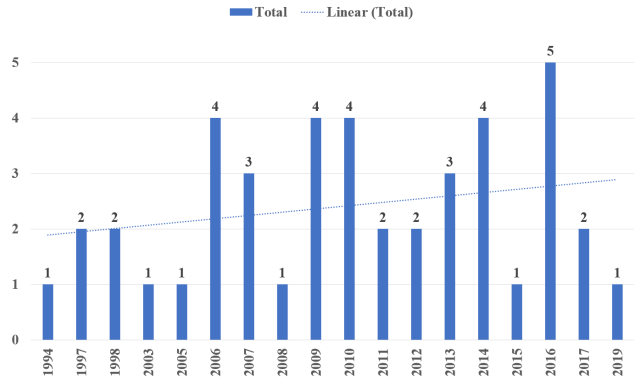

FIGURE 2. Number of publications included in this review per year.

professional player at the game of Starcraft II. They described its internal components, including: Lamarckian Evolution, Competitive Coevolution Algorithms and Quality Diversity. Furthermore, Duarte *et al.* [21] reviewed both planning and learning components in games generally in 2020.

On the other hand, every publication in the second and third groups (43 publications) was assessed via collecting some relevant information, such as title and year, game relevant information (e.g. game name and assigned task), coevolution relevant information (e.g. coevolution target and type), and finally some insight information regarding the paper, such as advantages and possible future work. The full list of the information collected is listed and explained in table 2. The last column in this table gives the reader an example of how to fill this template using paper [63] as an illustration.

F. SYNTHESIS

According to the systematic review methodology that this paper follows, arranging and analysing the previously obtained information is the next step to produce justified and supported findings, outcomes, and recommendations. Borrego *et al.* [13] recommended using Petticrew and Robert's internal steps in [58] to carry out the synthesis. Those steps start with organising the publications, then critiquing within them, and conclude with critiquing across them.

1) ORGANISE THE PUBLICATIONS

NVivo was used to organise the publications. NVivo is a tool that helps discover qualitative and mixed-methods research by analysing the unstructured text of all papers found [23].

2) CRITIQUE WITHIN THE PUBLICATIONS

Considering the timeline of the evolution of the use of coevolution in RTS games, we divided them into three stages.

Stage 1 (1994-2004): The main theme of this era was simplicity. Coevolution was used only to evolve game strategies for micromanagement tasks, including unit control, formation and positioning. Reynolds [63] conducted one of the first research studies in this field in 1994. He used GP to co-evolve competitive strategies for a pursuit-evasion game called "Game of Tag" without reaching the known optimality. The results achieved from seven experiments were encouraging in such a new research direction at that time. It is worth mentioning that the author was initially inspired by Angeline and Pollack [4] when the latter introduced a competitive fitness function to evolve strategies in the turn-based game called Tic Tac Toe.

Three years later, two of the Khepera robots, controlled by neural networks (NNs), were physically used in another pursuit-evasion game. The predator robot had a vision module while the prey one was equipped with twice the speed of the predator. By adding a competitive component to an existing evolution experiment, Floreano and Nolfi [28] were able to produce a form of quick adaptation from learning during life with no extra cost of evolution time. By letting the best player in the generation compete (in ten games) against the existing best player, some positive features were evolved, such as obstacle avoidance, visual tracking, object discrimination, and following the prey. The authors concluded that the system dynamics complexity was increased due to the ever-changing fitness landscape resulted from the Red Queen effect of the competing species.

Two other studies have based their work on the same robots. Nolfi and Floreano [54] extended their previous study [28] by exploring the best conditions for arms races behaviour to emerge. They used a Hall of Fame as a coevolution long-term memory, in which a player competes against the best player evolved over the last ten generations. The proposed system showed that coevolution was not able to evolve a general strategy that is able to defeat all other strategies; instead, it could evolve a strategy that is a specialist in defeating the current opponent's strategy. They also explained that this should not be seen as a lack of success, having such a non-dominant strategy gives the opponent more room for improvement and consequently pushes every player to continue the process of the arms race. In 2003, Nitschke [49] constructed two teams of Khepera robots, and ran three experiments to show that strategies that coevolved from a player in a single environment do not work with a team of the same players. This is due to a lack of coordination which leads to physical interference between players from the same team during the game. The study also showed that coevolution in a competitive context does not need direct communication or coordination methods to evolve successful cooperative strategies.

Apart from coevolving competitive robots in pursuit-evasion games, two other competitive/cooperative studies were introduced at an early stage of this field. In 1997, Luke *et al.* [42] developed a system to coevolve two teams of soccer players to compete in the Robocop competition. They utilised the automatic programming of GP to develop algorithmic behaviour for soccer robots. Two separate trees were used in GP, with the first evolving player movements, while the other was dedicated for ball-related actions, such as kick and pass. They also introduced a new crossover operator, root crossover, to allow teams to swap players with their

TABLE 2. Developed template to collect information regarding study publications using [63] as an example.

Field	Description/Options if any	Example
Publication Title	The publication title	Competition, Coevolution and the Game of Tag
Publication Year	The publication year	1994
Publication Type	Article Paper Conference Proceeding Book Section	Conference Proceeding
Game Name	The name of the game	Game of the Tag
Game # of Players	1-player 2-player 2-teams of players Multi-player	2-player
Game Task	The task that players are asked to do: Collect resources Create buildings/units Control units Generate build orders	Control units
Game Management Depth Level	The management depth level associated with the assigned task: Macromanagement Micromanagement	Micromanagement
Agent Controller	How every player actions and decision was controlled? e.g. Expert System Neural Networks(NNs) Simulation Model	Simulation Model
Coevolution Purpose	What is the purpose of using coevolution? Evolve Game Strategies Evolve NPCs Evolve Game Contents	Evolve Game Strategies
Coevolution Type	What is the type of coevolution used? Competitive Cooperative	Competitive
CM	What is the type of CM used? Long-term Short-term Not Reported	Not Reported
Evolution Method	What is the evolutionary algorithm (EA) used to evolve species? e.g. Genetic Algorithm (GA) Genetic Programming (GP)	Genetic Programming
Coevolution Fitness Update	How are the individuals' fitness updated to produce the coevolutionary effect?	Individual fitness is updated based on the average score from playing a series of four games against six other random players in a single population.
Advantages	What is good about this publication?	For the first time, coevolutionary algorithm (CA) were used to evolve strategies in an RTS game. seven experiments were used, and the results were encouraging for exploring such a path. The proposed system was able to approach quality evolved players without being able to reach the known optimality.
Future Work	How can the proposed system get improved?	Increase the difficulty of some simplified aspects such as constant vehicle speed and zero turning radius and momentum. Explore how to enhance the produced strategies by for example using different competitive fitness functions.
Comment	Any other important information	The author was inspired by Angeline and Pollack in which they introduced a competitive fitness function to evolve strategies in a turn-based game called Tic Tac Toe.

strategies. Due to having expensive fitness evaluation, fitness was based on a single game between two random teams. Moreover, instead of assessing the overall team performance, they paired off the teams and assessed every player against its pair. They found that using the number of goals scored solely in the fitness function improves convergence. The proposed system was able to evolve a team of robots that has beaten a hand-crafted team of robots in a soccer game. It is worth mentioning that every player does not communicate with other players even in the same team. Every player receives updated information about the game environment, details such as the ball, the field, and other players both inside and outside the team. In the other study in 1998, Ficici and Pollack [26] developed a new prediction game consisting of three players: one generates output, while the other two players (one is with and the other is against the

generator) predict the generator player. They have proposed a system that provided not only a powerful metric of behaviour, but also the ability to explore convergent and competitive dynamics and their interaction. The fitness used was calculated based on the average score from playing games against all other players in the other two players' populations. They found that the introduced game required a mixture of cooperative and competitive pressures to avoid simple mediocre stable-states and closed-endedness.

Stage 2 (2005-2011): Researchers in this stage started using coevolution in evolving NPCs in addition to the game strategies which were considered in stage-1. Spatial reasoning challenges in RTS games also attracted researchers to this stage; Researchers started developing coevolution systems that use influence maps (IM) as an agent controller to face this challenge.

In 2006, Miles and Louis [45] introduced a new RTS game, called Lagoon, to simulate a sea battle. They developed a coevolution system that uses an A* pathfinder to find a path to reach the required destination, while the strategy of any individual in the GA population was represented via an influence map tree (IMTree). The authors found that coordination behaviour could emerge between the same team members, which made the team look like a real team in nature. This means that IMTrees have the potential to coevolve competitive NPCs for long-term RTS games. Miles and Louis [46] extended their previous work by improving the game to include more features, such as gathering resources, constructing buildings and including more varieties of army units. Furthermore, they developed a spatial reasoning system that analyses the current game status (i.e. IMTree) to determine player spatial objectives. Such objectives were then optimised using GA within the resource allocation domain. The coevolved players were found to be significantly superior to the hand-coded players.

Avery *et al.* [8] designed a sea battle game based on a ''capture the flag'' scenario, in which the boats were used as army units. They developed a competitive coevolution system to evolve game strategies at a low tactical level, such as which boat to attack and which places boats should target. They used an individual IM for every boat in the game, and tested their proposal in an open-ocean scenario with a single flag for the defender team. The proposed system was found to be successful in coevolving coordinated tactics for the attacking team that could not be countered by the defending team, due to the simplicity of the simple operation theatre. The work was extended in [7], by adding a few islands of different sizes to the original map and also by considering a flag for each team to defend. Furthermore, they updated the reasoning system by linking every set of coordinated army entities to a certain map in order to achieve a given task, with the focus on group tactics instead of on agent behaviour. The experimental results showed that IMs were coevolved successfully to produce more interesting coordinated tactics.

As a further extension, the authors in [70] used a complex version of the capture the flag game as a testbed for the coevolution system to investigate the impact of coevolved players on human players' training. They did this by training 16 undergraduate students using either or both of the following AI: the first is a hard-coded opponent, while the other was coevolved. Afterwards, a unified post-test game, different from the training, was exposed to all students. The results showed training using hard-coded or coevolved players did not cause a performance difference between the trained players. Taking this into consideration, designing new training programs could be done more quickly.

In 2008, Keaveney and O'Riordan [32] developed an RTS game called Bellus Bellum Gratia (BBG), which is a TBS game equipped with some RTS game features including simultaneous turns and imperfect spatial information. A competitive coevolution system was proposed in [33] to handle a BBG game. The authors introduced two measurements to assess both tactical spread and attack coordination among player units. Two experiments took place, in which the populations were coevolved using a single map in one experiment and multiple maps in the other one. The results showed that coevolved players using multiple maps were more robust than players coevolved using a single map. They reported that attack coordination is more effective in producing robust strategies than spread coordination, and also that it is easier to evolve better attack coordination in coevolution using a single environment compared to using multiple ones. Two years later, the work was extended in [34] by creating a coordination profile for each strategy (spread/attack coordination managed separately). These profiles were analysed throughout the coevolution process to monitor changes in the coordination and importance of such changes. The results showed that coevolving strategies with more intelligent behaviours takes place at the cost of one or both coordination measurements.

Car racing games have attracted some researchers to build a coevolution system to explore such an interesting game. In 2007, Togelius *et al.* [73] started with a car racing game in which one or two cars compete to score more points within a specific time frame. They wanted to investigate the effect of the number of populations used in the coevolution as well as the agent controller. In terms of population number, they tested generational solo-evolution (no coevolution) vs steady-state evolutionary with a variety of population numbers (1,2, and many). They also explored different types of agent controllers within NNs, modular controllers and GP. They found that multi-population coevolution created a better environment to evolve the controllers, more than either solo evolution or single-population coevolution. They also reported that when using two cars, the modular controller is the best one to be used among the other agent controllers. Moreover, they concluded that larger controllers learn faster when a single population is used in the coevolution.

A couple of years later, Cardamone *et al.* [15] decided to evolve a competitive driver bot to compete in an international competition called ''2009 TORCS endurance world''. Their idea pivoted on utilising the winning driver of the earlier

year's competition ''SIMPLIX'' and coevolving the car setup (16 out of more than 50 available parameters) using one of the existing cooperative coevolution methods in the literature. The results were compared with one of the well-known EAs in the literature, covariance matrix adaptation evolution strategy (CMA-ES), and other GAs. The proposed cooperative system was able to evolve car settings that not only outperforms the other EAs, but also improves one of the existing high performing human developed drivers, i.e. SIMPLIX.

Using coevolution to evolve NPCs started in the second stage, as mentioned earlier. Cardamone's work, mentioned above, is one of those attempts. However, Hong and Cho [30] introduced a new RTS game, called ''Build & Build''. Every team in this game gains points from building towns, producing army units, and attacking opponents. Individual fitness was calculated based on the score resulting from playing games against random opponents whose strength was taken into consideration. While the authors' focus was on evolving an NPC's reactive behaviour during the game, they also explored different scenarios for different environment content complexity. The results found were encouraging enough for researchers to invest more effort in this direction.

Priesterjahn *et al.* [60] proposed a two population competitive coevolution system to evolve NPCs in a multi-player computer game called ''Quake3'' using evolution strategies ($\mu + \lambda$). They suggested avoiding the use of hard-coded players during evolution but to use them as part of testing. The coevolved NPCs were able to dominate the existing hard-coded agent on any difficulty level and even on larger maps. Consequently, the proposed algorithm could be utilized for training players in games and environments without existing artificial players.

Based on an existing military game, Thompson *et al.* [72] updated the game, called it ''EvoTanks'', and used it to coevolve NPCs in a competitive environment. Players' efficiency and health factors were considered in evaluating their fitness, which is calculated based on the average of resulting scores from games played in a round-robin league with opponent team agents. Results showed that the coevolved agents had competent reactions and strategies to face different hand coded players.

Dziuk and Miikkulainen [22] explored how to use automatic shaping of coevolution to evolve NPCs with more effective behaviours. By altering both the fitness function and the environment throughout the evolution process, they showed a direct way to shape the coevolutionary procedure. They introduced several automatic shaping methodologies to be used in the coevolution and found that using shaping was beneficial. Also, their work concluded that the shaping concept pivots on increasing diversity, and shaping the environment was more effective than shaping the fitness in this case.

Apart from the previous collective studies, a number of individual works were conducted during this stage: In 2006, Monroy *et al.* [48] explored using a newly introduced Coevolutionary Memory (CM) called the Layered Pareto Coevolution Archive (LAPCA) to coevolve game strategies for a game, ''Pong'', that they modified. They used NeuroEvolution of Augmenting Topologies (NEAT), which is an RL technique that searches both an NNs topology and weight spaces simultaneously to train the player controller NNs. The proposed approach was able to show that LAPCA scales up to NN in complex domains and does not require memory as much as HoF. The game used was found insensitive to the CM technique used, but more work was required to show the effectiveness of LAPCA and HoF as memory techniques.

At the same time, Uchibe and Asada [75] explored how CAs, using soccer robots, could evolve cooperative and competitive emergent behaviour. They used a simplified soccer game where only one team had a goalkeeper and the other team had either one or two playing robots. The novelty in their work came from relying on importance sampling for fitness sharing and employing incremental evolution using multiple schedules. Instead of evaluating every individual in one population against all members of the other population, they used importance sampling to reduce the number of played games, which improved the efficiency of the fitness evaluation process. Moreover, the new system was able to utilise incremental evolution to produce cooperative and competitive behaviour through four types of experiments.

Leigh *et al.* [37] introduced a study to show the potency of using coevolution in both understanding and testing the balance in an RTS game. Their proposed fitness was based on the average fitness resulting from playing six games (three against random players from the hall of fame and three against players selected via shared sampling from the opponent's population). The proposed system, using a simplex heating map as a visualisation tool, was able to bring balance to the game by tuning the game rules and parameters when required during the game.

Menezes and Costa [44] developed a multi-player shooting game, in which Gridbrains (GP Virtual Machine) was used as an agent controller (for vision and audition sensors) and Simulation Embedded EA (SEEA) to provide the evolutionary process. The group fitness in SEEA encouraged the players to evolve cooperative behaviour which yielded efficient shots and better synchronisation across the whole team's shots. Utilising the interaction among the used fitness function, the populations' coevolutionary dynamics and the environmental setup, the proposed system did not need a geographical separation to evolve the species specialisation behaviour.

In the same year, Lichocki *et al.* [38] had a lesson learned from the only 1-player game in this review, ''Gathering Resources Game''. They learned that behavioural sophistication does not always guarantee better performance. The proposed system proved very effective at collecting resources, but performed poorly in situations that required short-term memory like obstacle avoidance.

In 2010, Rawal *et al.* [62] introduced a piece of work to show how to construct concurrent cooperative and

competitive behaviour in a complex Pursuit-Evasion game environment. Their study was based on two experiments: both were between two teams of players where the predator team always had three players while the prey team consisted of either one or two players. The proposed system showed that for both the predators and the prey(s), high level strategies emerged on both competitive and cooperative levels. They also found that interactions among multiple autonomous agents required the coexistence of both cooperative and competitive behaviour in RTS games which have ever-changing dynamics.

Stage 3 (2011-2021): Researchers at this stage started using coevolution to generate game content. They also stopped using NNs as an agent controller.

In 2012, Cook *et al.* [19] introduced the first study to use coevolution to evolve game contents. Their focus was on how to make the procedural content generation task more of a shared creative task between a human designer and the proposed automated designing system. To do this, they proposed using three cooperative species to produce a game environment: Game Map, Game Layout and a list of powerups called a Game Powerset. They proposed a cooperative coevolution system to design simple Metroidvania platform games. They used a cooperative platform System, called ANGELINA, to design the game with a cooperative fitness that was calculated based on the score resulting from playing one game against the fittest member of every other species. 180 human-players tested the system and provided feedback. Some concerns were taken into consideration while the rest were left for future work.

Two years later, Ruela and Guimarães [66] based their work on the assumption that the generation of procedural content of NPCs adapted to human strategies can produce a more interesting and attractive game for human players. They used an existing simulation system, called ''Armageddon Battle Simulator'' (ABS), that simulates the game ''Call of Roma (Caesary)''. They designed the fitness evaluation to assess victory points, offensiveness, defensiveness, usage, and participation of the individual design. They were also able to only rely on the number of soldiers when the fitness was assessed for either the player or the whole team. The coevolutionary GA was able to present a convincing victory rate in all games.

In 2016, Nogueira-Collazo *et al.* [53] proposed a competitive coevolution system with two species: one to evolve players with winning strategies, and the second to evolve game maps that make it harder for the coevolved players to win. They introduced an algorithm with three variants that differed in how they updated HoF members based on measurements that depend on quality and diversity. The proposed algorithm was able to generate maps that favoured both allies and game AI that performed very efficiently.

At the same time, Hintze *et al.* [29] explored how to employ coevolution in adjusting opponents' difficulty in Pursuit-Evasion games. They used Markov networks as the agent controller while instead of evolving two populations independently, both populations were mixed allowing more difficult situations to be handled by the players. The proposed system was able to show that orthogonal coevolution excelled in creating players with various levels of difficulty.

Ruela and Guimarães [67] extended their previous work [66]. A new cooperative framework was introduced to evolve an entire team instead of evolving a single player. They proposed having a team of N heroes as an individual player; the total fitness of those players constitutes the whole team fitness. The developed system was made to choose between random or greedy cooperation according to a proposed α-based method. The new system was able to evolve players with successful strategies, with well-developed formations, against players of Call of Roma.

Nogueira *et al.* [52] started a series of studies; initially they introduced a new RTS game called ''RobotWars''. This game includes two components: a battle generator to generate different scenarios, and a battle simulator to run the game. They proposed an HoF competitive coevolution system where the fitness was calculated based on the average result of playing a game against all opposing HoF members. While they successfully coevolved winning strategies, the coevolution engine could use the final game result to predict the initial players' configurations.

A year later, they [51] extended their previous work [52] to analyse five different HoF implementations for short/long term memory mechanisms. They found that updating the memory members using selection criteria produced a good performance, noting that diversity was the best selection criterion to be used. Moreover, they found that memory size plays an essential role in convergence. In addition to the existing HoF memory archive, a new archive Hall-of-Celebrities (HoC) was introduced in [18]. HoC applies pressure on the optimisation process and differs from HoF when updating its archives based on quality and diversity metrics. They showed that coevolutionary cycling frequency was dropped when hybridising HoF/HoC based algorithms. This work produced victorious strategies over both coevolved and optimised players, which reflect the self-adjustment capacities of the proposed algorithms. Two years later, they extended this work [53] to use coevolution in evolving game contents as explained earlier.

Ballinger and Louis [10] started a series of studies on using GA in coevolving strategies for a two-player game, ''Water craft'', in a competitive environment. They proposed calculating fitness based on a linearly scaled shared fitness score resulting from playing games in a round-robin league with opponent team agents in a teachset. The teachset always had eight members: four random players from HoF, and four players from the opponent population. They compared the coevolved strategies by testing them against three hand-coded players. Results showed that coevolved players preserved a higher average performance over the GA developed players; on the field level, they were also able to destroy the enemies' command centre more frequently, highlighting the robustness of the coevolutionary results. They extended their work [9]

by injecting human strategies in the teachset. The updated teachset always had eight members: three random players from HoF, three players from the opponent population, and two players injected with human player's strategies. They found that teachset with case-injection increased the coevolution convergence without adverse effects on its robustness. These results encouraged them to invest more in the case-injection methodology.

In another study [11], the authors explored four different case injection methods; injection into: only the teach set, only the population, both the teachset and the population, and the last one with no injection. The robustness of the proposed methodology was tested by having the best players compete against five players produced from hand-tuning and coevolution. These five players were never seen during training and were different from the ones used in the injection. It was found that the *only injection in the population* method was the only one that made a difference in the performance. Hence, a coevolutionary population could be influenced by inheriting some playing styles from the injected cases, considering that injected cases might have different levels of influence on the inherited features. These results, along with the robustness test, emphasise a similar conclusion to that reached in their previous study mentioned in the last paragraph. It indicates that using case injection into a coevolutionary population would increase its convergence without compromising its robustness. In 2016, the same authors extended their work [12] when they added branching and iteration to the build-action sequence, aiming to create better build orders by using a more complex representation. Their focus was on finding strong (beat opponents rapidly) and robust (beat many opponents) build orders. Five-action build orders, developed using CA, GA, and Hill Climber (HC), were compared. They found that regardless of whether case injection was used, GA was able to produce strong build orders, while CA produced robust build orders. Furthermore, CA was able to deliver strong build orders when case injection was used in teachset and population.

Apart from the previous collective studies, a number of individual works were conducted during this stage. In 2014, Fernandez-Ares *et al.* [25] developed a system intending to enhance the action controller of the players by improving the relevant behavioural parameters in the Planet Wars game. Planet Wars itself is a pseudo-turn-based game; it allows bots to decide on their following action in one-second micro-turns, which then happen at the same simulated time. They explored three fitness functions to assess the exploration of the bots individually. These functions are (1) Victory Based: the final situation and number of turns required to reach it, (2) Slope Based: the slope of a linear regression line for the ships owned by a player, and finally (3) Area Based: the area (integral of the curve of the live-line of bot's) which reflects the average percentage of ships owned by an individual throughout the game. The proposed cooperative coevolutionary approach significantly saved training time. It was found that slope-based fitness was slightly better than area-based fitness, and the latter is more effective than victory-based fitness.

In 2015, Parker [56] proposed a competitive coevolutionary system to evolve NPCs for the Xpilot-AI game. They proposed calculating the fitness every two minutes of gameplay, and it was biased during the calculation time, considering that new biases would be calculated every 15 generations. It was found that fitness biasing performed better in two cases: (1) when finding model parameters that can be learned next to the solution is not possible and (2) when testing a solution would not take a long time and would not disrupt the player's task.

Liu *et al.* [40] produced effective micro-behaviours (tactical for a group of units and reactive control for a single unit) using heuristic search algorithms. They also compared the performance of their bot with two other bots from the literature. Three different scenarios were used: training, testing and head-to-head. They proposed calculating fitness based on the average scores of five games played in ten runs with a randomised seed. The proposed GA quickly evolved adequate micro behaviour to operate both melee and ranged attack units (one of the existing bots from the literature is better against a melee attack, and the other is better against ranged attacks).

Rudolph *et al.* [65] developed a competitive CA to evolve NPCs for the Starcraft ''Brood War'' game. The fitness evaluation considered the situation when a player destroys a building that belongs to the opponent player; a player got rewarded by adding more units as a reinforcement. The coevolution takes place by allowing AIs to compete; the learning during the competition pivots on combining reinforcement and evolutionary learning components from Extended Classifier Systems (XCS). Four different AIs were tested (Defender, Attacker, Explorer and Strategist). They found that the Explorer performed better than other AIs, and that self-adaption at run-time improved every player's performance.

In 2017, Liu *et al.* [39] introduced a simple new battle game between two spaceships with no missiles. A spaceship is considered a winner when facing its opponent's back within a specified proximity. A several-step look-ahead controller managed the following players' actions. Rolling Horizon Genetic Algorithm (RHGA) and Rolling Horizon Coevolution Algorithm (RHCA) considered a macro-action consisting of various specified actions in all the games. Results showed that as the macro-action gets shorter, the relevant performance improves.

Lastly, Adhikari *et al.* [1] conducted a study aimed at evolving high-quality micromanagement strategies in RTS games without the need of an opponent to evolve against. Initially, coevolution takes place between a team of ranged units against a team of melee units. This was followed by coevolution between two teams; each was formed by combining ranged and melee units. Results showed that the proposed approach could find micromanagement strategies

FIGURE 3. The 50 most common words appeared in the reviewed papers. The bigger the word, the more frequent it appeared in the papers. Every word that appears in the figure reflects the frequency of both the word itself and its synonyms, e.g., the word *strategy* represents the frequency of the usage of the words: *scheme*, *schemes*, *strategy*, and *strategies*.

that perform well in unseen scenarios, which indicates its robustness.

3) CRITIQUE ACROSS THE PUBLICATIONS

To give readers an indication about the main keywords found across the reviewed papers, NVIVO was used to create a word cloud for the 50 most common words, see figure 3.

a: COEVOLUTION PURPOSE

Considering the review conducted above, it is found that researchers used coevolution in RTS games for three main purposes:

- Evolving game strategies,
- Evolving NPCs, and
- Evolving game contents.

At the beginning of every stage explained in the previous section, researchers started using coevolution for the purposes mentioned in the same order. Interestingly, the number of publications of every purpose depends on the time researchers started to use coevolution for that purpose. As depicted in Table 3, the number of publications of every purpose was found to be almost double the preceding purpose's number of publications. For example, the figure shows that the number of publications that used coevolution to evolve game strategies (26 publications) is almost double the number of publications that used coevolution to evolve NPCs (12 publications). It is also worth mentioning that considering a new purpose for using coevolution in RTS games does not mean that researchers stopped targeting older purposes, as explained in table 3.

b: NUMBER OF PLAYERS

By looking at Table 4, the reader can notice that only one publication considered a 1-player game, where coevolution is not the best approach to handle such types of games. On the other hand, having two competing sides playing against each other is the best environment for coevolution. Consequently, readers can conclude from the total number column that almost 75% of the considered games are either 2-players

TABLE 3. Number of publications of every coevolution purpose in each stage.

Coevolution Purpose	Stage 1	Stage 2	Stage 3	Total (%)
Evolve Game Strategies	6	11	9	26 (60%)
Evolve NPCs		8	4	12 (28%)
Evolve Game Contents			5	5 (12%)
Total	**6**	**19**	**18**	**43 (100%)**

TABLE 4. Frequency of number of players used across all stages.

Number of players	Stage 1	Stage 2	Stage 3	Total
1-player		1		1
2-player	3	4	7	14
2-teams of players	2	11	5	18
Multi-player	1	3	6	10
Total	**6**	**19**	**18**	**43**

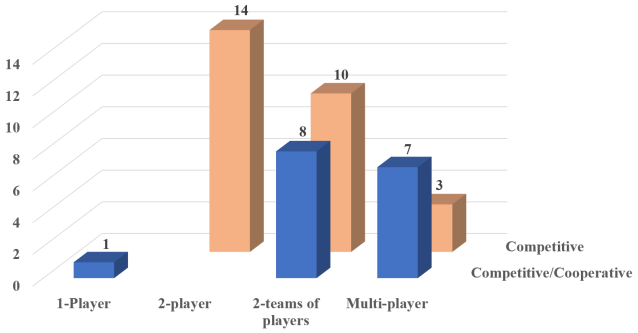

FIGURE 4. Frequency of coevolution type used for every category of players number in the considered publications.

or 2-teams. Comparing this table to figure 4 can explain that considering 1-player games does not give a chance for the coevolution to evolve cooperative behaviours, as there will not be any player to team up with. Hence, cooperative coevolution requires either 2-teams of players or multi-player games to work.

c: MANAGEMENT DEPTH LEVEL & TASKS

Figure 5 shows the relationship between the management level of decisions and the number of players for the games used in the publications reviewed. This figure presents the percentage of the publications that considered RTS games with tasks from strategic and tactical levels (macromanagement and micromanagement) for every category and for every number of players in the game. The figure also shows that the percentage of publications that used coevolution for games involving tasks from the micromanagement level of the decision have an inverse proportional relationship with the percentage of publications that tackled tasks from the macromanagement level of decision. Tasks from the macromanagement level of decisions, such as creating build orders and collecting resources, are more complex than tasks

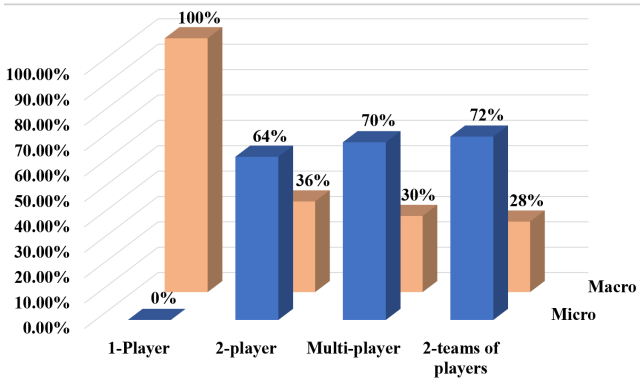

FIGURE 5. Distribution of the number of players against management depth level used.

TABLE 5. Game tasks used in the included publications in this review.

Management Level	Task	References
Macro	Control units.	All references
	Formation/Positioning of units.	[42], [45], [75], [46], [66], [65], [40], [67], [1]
Micro	Collect resources	[30], [60], [46], [37], [44], [38], [22], [19]
	Create buildings/units	[30], [46], [33], [34]
	Create build orders	[10], [9], [11], [12]

from the micromanagement level of decisions, that focus mainly on reaction behaviours. This complexity increases when the number of players increases. This could explain why most researchers used coevolution for games with macromanagement level decision tasks when considering fewer players and vice versa. Table 5, which lists the tasks associated with the games considered in the publications involved in this review, supports the previous analysis. As mentioned, games with micromanagement tactical tasks, such as controlling and positioning units, were used more than those games with macromanagement strategic tasks, such as collecting resources and creating build orders.

d: EAs IN CAs

Table 6 shows that researchers prefer to coevolve using GAs rather than GPs, yet GP was used more frequently than the other EAs. In a further investigation of these two EAs, we found that GA was used in 100% of the publications that used coevolution to evolve game contents (five publications). Moreover, GA was also used in 10 of the 12 publications that used coevolution to evolve NPCs (83.33%). We also found that only one EA was used solely in stage 3 since 2012, that is, the GA. It is worth mentioning that both generational and non-generational (steady-state) GAs were widely used in this regard. It is notable that the five publications that used GP as an evolution methodology all belong to either stage 1 or 2. Although they share similar features, such as evolving game strategies as the coevolution purpose and a simulation model as an agent controller, they differ in other aspects like the game, number of players and the coevolution type considered.

e: AGENT CONTROLLER

In terms of agent controller, we found almost 49% of the publications found it more convenient to develop their own simulation model to simulate a player (agent) controller. Some researchers used either NNs (almost 21%) or IM (almost 16%) as the controller for their game players on a smaller scale. While simulation models were used during all stages, IMs were not used in stage 1 and NNs were not used in stage 3. The need to have a very customised player at very low levels of detail can be seen as the main reason for researchers to prefer using customised simulation models over other techniques of controlling player agents. We also noticed that all researchers used IMs to control players only when they used GA with 2-teams of players games. This can be justified if we know that an IM is a grid for the considered map in the game; this grid is assigned with values based on a customised spatial concept. Such spatial reasoning is usually required for games with many players competing in a defined region such as in games with two teams of players. It is also worth mentioning that neither IM nor NNs were used in coevolution to evolve game contents and environment.

f: COEVOLUTION MEMORY

It was clear that researchers did not pay enough attention to the importance of using or reporting memory techniques to overcome the forgetfulness of CAs, as explained in Section II-B. Table 7, shows that only slightly less than 50% of researchers have reported the memory technique they used in their study. It was also noted that researchers used long-term memory techniques, such as HoF and LAPCA (the latter was used only once) to evolve game strategies. We do not have an explanation for this; however, maybe the lack of information on the used memory technique hinders the appearance of using this technique for other coevolution purposes. Therefore, we broke down the usage of memory techniques to a yearly level to get more insight. Interestingly, the first short memory techniques, such as elitism, were reported in 2009, which is 12 years later than the first reported long-term technique; again, we believe that the missing information has another footprint here. Consequently, readers can now recognise how it is important for researchers to report more details about their proposed systems to support studies like this review. It is also noted that researchers spent limited efforts on employing CM techniques in this field. Researchers can extend the work in this field by exploring various CM techniques, either on an individual level or in combination, and by analysing their effect on coevolutionary systems' performance.

g: COOPERATION VS COORDINATION

We noticed that some researchers used the cooperation and coordination terms interchangeably while they actually are different. As explained earlier, the *cooperation* concept

TABLE 6. Player controller and EAs used by every publication, with player controller on rows and the EA type on columns).

Agent controller	Multi-agent ESP	Evolution Strategies $(\mu + \lambda)$	Evolutionary Programming	Gene Expression Programming	GAs	GP	NEAT
Expert System					[56]		
Influence Map (IM)					[45], [46], [8], [70], [7], [40], [1]		
Learning Classifier System					[65]		
Markov Networks					[29]		
NNs	[62]		[26]		[28], [54], [49], [30], [72]		[48], [22]
Open Loop Monte Carlo Tree Search (OLMCTS)					[39]		
Simulation Model		[60]		[38]	[37], [15], [52], [19], [51], [10], [9], [25], [66], [11], [18], [12], [53], [67]	[63], [42], [75], [33], [34]	
GP					[44]		
Neural Networks vs Genetic Programming vs Modular controller			[73]				

TABLE 7. Number of publications used for each memory technique per coevolution purpose.

Memory Technique	Evolve Game Strategies	Evolve NPCs	Evolve Game Contents	Total
Not Reported	9	9	4	22
Long-term		12		12
Short-term	5	3	1	9
Total	26	12	5	43

TABLE 8. Type of coevolutionary memory (CM) used in the publications reviewed.

Year	Short-term	Long-term	Not Reported
1994			[63]
1997		[28]	[42]
1998		[54]	[26]
2003		[49]	
2005			[30]
2006		[48]	[60], [45], [75]
2007			[46], [72], [73]
2008		[37]	
2009	[8], [33], [38]		[44]
2010	[70], [15], [7]		[62]
2011	[34]		[22]
2012		[52]	[19]
2013		[51], [9], [10]	
2014	[25]	[11], [18]	[66]
2015			[56]
2016		[12]	[53], [40], [65], [29]
2017	[67]		[39]
2019			[1]

originated within a coevolutionary context, and it takes place when players of one species performance (fitness) affects the performance (fitness) of players in other species positively. In other words, one species winning influences the winning of other cooperating species, and the same applies in losing situations. On the other hand, the *coordination* concept originated within a multi-agent system context; it refers to heterogeneous agents taking decisions autonomously to achieve a common task [69]. In a coevolutionary context, coordination can take place when players, for example, communicate information, such as teammates/opponents locations and actions to maintain specific behaviour like spread, attack or defend. The source of this confusion between these concepts comes from the similarities they share in many aspects. Both of them share the same target partially: players help each other achieve a given task. Another similarity is that agents (players) that help each other belong to the same team in both concepts. Additionally, both concepts can occur between players belonging either to the same or different population or species. We provide here a guideline to differentiate between cooperation and coordination: In cooperation, players share their fitness, while they share information other than the fitness, like location and actions, in coordination. It is also worth mentioning that when coordination involves sharing players' intentions and beliefs, some researchers call it collaboration [69]. Table 9 shows the coevolutionary type used in the considered publications (competitive/cooperative); it also shows whether or not coordination was used.

TABLE 9. Coevolution type (competitive/cooperative) along with whether coordination was used in every publication reviewed.

	Competition	
	Coordination	No Coordination
Cooperation	[42], [8], [62], [7]	[26], [49], [75], [38], [44], [15], [70], [66], [25], [29], [67], [19]
No Cooperation	[45], [33], [34], [56]	[63], [28], [54], [30], [48], [60], [46], [72], [73], [37], [22], [52], [51], [9], [10], [11], [18], [53], [65], [40], [12], [39], [1]

G. LIMITATIONS, VALIDITY, AND RELIABILITY CONCERNS

The consideration of performance comparison for the reviewed systems was not possible in the current paper due to a few reasons. The proposed systems were developed using different frameworks; besides collecting the related scattered codes over two decades or even creating them from scratch was not a feasible task. Furthermore, the lack of a unified environment that includes the required facilities to compare the validity and practicality of the proposed system was another limitation, even in the case of having the proposed systems' source codes. The latter restriction highlighted a new task that researchers are encouraged to consider in their future work.

IV. FUTURE WORK

Based on this review and analysis of existing literature, future potential work can be categorised as follows: (1) CA related suggestions, (2) RTS games related recommendations, and (3) agent controller related concepts. Table 10 summarises those categories and relevant future work suggestions as well as the publications that reported those suggestions.

CA Related Suggestions: Cover suggestions that were made to improve and explore the issues that are related more to the coevolution part of the proposed system. The most frequent suggestion raised by researchers was using different fitness functions; some researchers have not determined a specific function type (as in [19], [51], [63], [66], and [67]). On the other hand, other researchers were able to suggest specific function types, such as dynamic functions [75], visualisation tools to support determining the fitness [37], automated shaping functions [22], and mathematically approximated functions [25]. Exploration of different CA techniques is another CA related suggestion that was reported by researchers. Extending work on the memory technique LAPCA was suggested in [48] while others suggested doing more investigations on using the memory technique HoF [18], [45], [51], [52].

Improvements in the convergence of CAs needs further attention. The word improvement here reflects two conflicting factors: convergence should be accelerated to reach better solutions quickly, but at the same time it should not be so fast as to avoid premature convergence.

Another CA related suggestion is to explore how to improve CA emergent behaviours in the RTS domain. While some researchers raised this suggestion in a generic way [18], [49], others were more clear in their suggestions. Exploring how to analyze the complex behaviour of an arms race was one of the suggestions [26], [62], at the same time the question of how to bring temporal coordination into the coevolution process was also raised [33].

Exploring the use of cooperative coevolution is a possible way to improve CAs [40], [45], [60], [72]. Further improvement to CAs might be achieved by using a multi-population approach [25], [30], [40], [53]). Utilising the coevolved plans in predicting opponents' plans was also reported by researchers [9], [11], [33], [52]). Another CA related suggestion was to analyze the diversity of the used CA, mainly to solve the issue of selecting the best performing individual during later generations [7], [73], [75].

Exploring, and improving, different optimisation techniques in CAs is still an open direction. Some researchers suggested using multi-objective coevolution [38] to avoid getting trapped in local solutions, while others suggested exploring an entirely different selection method [19]. As well as implementing the coevolution in a physical parallel environment [66], [67]. The last CA suggestion was to explore the effect of using different population sizes on the evolution [42].

RTS Game Related Suggestions: Cover suggestions that were made to improve and explore different aspects of the game part of the proposed system. Exploring using different levels of complexity within the game environment at hand is a possible research direction, as suggested by many researchers. Some researchers have suggested exploration of different settings, features, and actions for the bot (player) that increases the difficulty of their game [7], [9], [15], [30], [34], [40], [45], [63], [75]. Some other researchers suggested adding more features to the game environment, such as barriers, obstacles, weapons, game rules and difficulty levels [19], [29], [34], [39], [46], [53], [72]. Other researchers suggested combining random and structured scenarios to produce robust strategies [1]. There are suggestions to explore different complexities of the game environment by adding rules between environments and players [44]. Some other researchers suggested exploring gaming with no human interaction [22].

Proposing a general system that can cover a wide range of RTS games is another future direction [18], [26], [33], [51], [56], [62], [70]. Researchers suggested increasing the complexity of the used strategy encoding to enable the production of more complex and robust strategies [9], [10], [12], [28], [40], [51]. Exploring different management levels of the game tasks (i.e., covering more of the various tasks from both strategic and tactic management decision levels, i.e. macromanagement and micromanagement) can be further researched [7], [8], [12], [30], [65].

Agent Controller Related Suggestions: Cover those suggestions made to improve and explore the issues related to the used agent controller. The most reported suggestion was increasing the complexity of the agent controller:

TABLE 10. Future work suggestions reported by every publication. Suggestions are categorised based on the part of the work that the suggestion relates to.

Future Work Category	Future Work Suggestion	List of references	Stage 1	Stage 2	Stage 3	Total
CA Related	Explore different fitness functions	[63], [75], [37], [22], [19], [51], [25], [66], [67]	1	3	5	9
	Explore different memory techniques in the used CA	[45], [48], [52], [51], [18], [25]		2	4	6
	Improve the convergence of the proposed system	[54], [60], [19], [10], [53], [1]	1	1	4	6
	Improve CA emergent behaviours in RTS domain	[26], [49], [33], [62], [18]	2	2	1	5
	Explore cooperative coevolution	[45], [60], [72], [40]		3	1	4
	Explore multi-species CA	[30], [25], [40], [53]		1	3	4
	Explore utilising the coevolved plans in predicting opponents' plans	[33], [52], [9], [11]		1	3	4
	Analyse the diversity of CA	[75], [73], [7]		3		3
	Explore different optimisation techniques in the used CA	[38], [19]			2	2
	Explore Implementing CA in a physical parallel environment	[66], [67]		1	1	2
	Explore different population size effect of the evolution	[42]	1			1
RTS Game Related	Explore different complexity of the game environment	[63], [30], [45], [75], [46], [72], [44], [15], [7], [34], [22], [19], [9], [40], [29], [53], [39] [1]	1	10	7	18
	Explore generalisation of the proposed system to cover more games	[26], [33], [62], [70], [51], [18], [56]	1	3	3	7
	Increase the complexity of used strategy encoding	[28], [9], [10], [51], [40], [12]	1		5	6
	Explore different management levels of the game tasks	[30], [8], [7], [12], [65]		3	2	5
Agent Controller Related	Increase the complexity of the agent controller	[42], [30], [72], [37], [8], [15], [62], [9], [10], [11], [65], [40], [39], [1]	1	6	7	14
	Analyse the agent controller performance	[73]		1		1

some researchers suggested increasing the complexity of the controlling algorithm itself by increasing the complexity of the reaction automation part or utilising previous knowledge [1], [9]–[11], [37], [39], [42], [65]. Other researchers suggested increasing their agent controller complexity by introducing more complex input coding or adding sensors to the agent [30], [72]. On the other hand, some researchers pointed out the need to model a special behaviour of the player or the game domain itself [8], [15], [40], [62]. Lastly, analysing an agent controller's performance by better comparing their performance is another open direction [73].

By looking at the summary of the previous reported suggestions in Figure 6, readers can recognise agent controller related suggestions are significantly less in number than other categories. This can be justified by comparing the number of factors that controls an agent controller to each of the CA and RTS games. It is remarkable also that CA related suggestions have attracted researchers slightly more than for RTS games. On the other hand, it is very clear that the number of reported suggestions in stage 1 is significantly less than the other two stages due mainly to the lower number of publications in stage 1. Readers can also notice that stage 3 has got slightly more reported suggestions compared to stage 2.

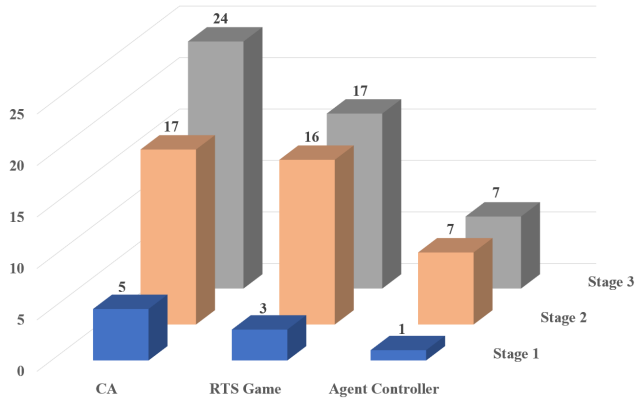

FIGURE 6. This figure shows the distribution of the future work suggestions reported in the publications across the earlier defined stages.

When readers look at Table 10, some future suggestions can be recommended as open areas for researchers to investigate more. The suggestions that have high frequency in the later stages means they are more hot recent topics. Some of those suggestions are stated below.

1) Explore using different complexity of the game environment,
2) Increase the complexity of the agent controller,

3) Explore different fitness functions,
4) Explore different memory techniques in the used CA,
5) Explore generalisation of the proposed system to cover more games,
6) Improve the convergence of the proposed system,
7) Increase the complexity of the used strategy encoding, and
8) Explore different management levels of the game tasks.

V. CONCLUSION

This paper conducted a systematic review to show why and how CA are applied in RTS games. We started with this reason for conducting this review, showing its scope and research questions. Then, we identified the document sources for the study and the possible databases and applied the predefined inclusion and exclusion criteria. Afterwards, the considered publications were scanned, organised, and analysed individually and collectively.

This review showed that CAs were used in RTS games for three main purposes: (1) evolving game strategies, (2) evolving NPCs (Game AI), and (3) evolving game environment contents. It was found that GAs are the most commonly used EAs in this regard. Furthermore, we found that researchers in this field prefer to simulate the agent (player) controller over using other controllers that require training.

This paper explained how the attention of researchers in this field towards using/reporting the memory techniques used for CAs is not up to the expected level. It is very important not only to explore more coevolution memory techniques individually or in a hybridised mode, but also to investigate their effect on the performance of the overall coevolutionary systems.

This review was able to conclude that the most frequent future work suggestions from the publications reviewed were to: explore different complexity for the game environment, increase the complexity of the agent controller, explore various fitness functions, explore other memory techniques in the used CA, explore generalisation of the proposed system to cover more games, improve the convergence of the proposed approach, increase the complexity of the used strategy encoding, and explore different management levels of the game tasks.

This paper also highlighted another limitation that requires more attention from researchers in this field; that is the lack of having a common environment to facilitate comparing the proposed systems and to validate the practicality of the suggested ideas.

REFERENCES

[1] N. K. Adhikari, J. S. Louis, S. Liu, and W. Spurgeon, "Co-evolving real-time strategy game micro," in *Proc. IEEE Symp. Comput. Intell. (SSCI)*, Nov. 2019, pp. 1990–1997.

[2] B. D. Allen, "Serious gaming for building a basis of certification for trust and trustworthiness of autonomous systems," in *Proc. Aviation Technol., Integr., Oper. Conf.*, Jun. 2018, p. 3844.

[3] P.-A. Andersen, M. Goodwin, and O.-C. Granmo, "Deep RTS: A game environment for deep reinforcement learning in real-time strategy games," in *Proc. IEEE Conf. Comput. Intell. Games (CIG)*, Aug. 2018, pp. 1–8.

[4] J. P. Angeline and B. J. Pollack, "Competitive environments evolve better solutions for complex tasks," in *Proc. 5th Int. Conf. Genetic Algorithms*. San Francisco, CA, USA: Morgan Kaufmann, 1993, pp. 264–270.

[5] K. Arulkumaran, A. Cully, and J. Togelius, "AlphaStar: An evolutionary computation perspective," in *Proc. Genetic Evol. Comput. Conf. Companion*, Jul. 2019, pp. 314–315.

[6] P. Avery, Z. Michalewicz, and M. Schmidt, "Short and long term memory in coevolution," *Int. J. Inf. Technol. Intell. Comput.*, vol. 3, no. 1, pp. 1–30, 2008.

[7] M. P. Avery and J. S. Louis, "Coevolving influence maps for spatial team tactics in a RTS game," in *Proc. 12th Annu. Genetic Evol. Comput. Conf. (GECCO)*, 2010, pp. 783–790.

[8] M. P. Avery, J. S. Louis, and B. Avery, "Evolving coordinated spatial tactics for autonomous entities using influence maps," in *Proc. IEEE Symp. Comput. Intell. Games (CIG)*, Sep. 2009, pp. 341–348.

[9] A. C. Ballinger and J. S. Louis, "Finding robust strategies to defeat specific opponents using case-injected coevolution," in *Proc. IEEE Conf. Comput. Intell. Games (CIG)*, Aug. 2013, pp. 1–8.

[10] A. C. Ballinger and J. S. Louis, "Robustness of coevolved strategies in a real-time strategy game," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2013, pp. 1379–1386.

[11] A. C. Ballinger and J. S. Louis, "Learning robust build-orders from previous opponents with coevolution," in *Proc. IEEE Conf. Comput. Intell. Games (CIG)*, Aug. 2014, pp. 1–8.

[12] C. Ballinger, S. Louis, and S. Liu, "Coevolving robust build-order iterative lists for real-time strategy games," *IEEE Trans. Comput. Intell. AI Games*, vol. 8, no. 4, pp. 363–376, Dec. 2016.

[13] M. Borrego, M. J. Foster, and J. E. Froyd, "Systematic literature reviews in engineering education and other developing interdisciplinary fields," *J. Eng. Educ.*, vol. 103, no. 1, pp. 45–76, Jan. 2014.

[14] M. Buro, "Real-time strategy games: A new AI research challenge," in *Proc. 18th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2003, pp. 1534–1535.

[15] L. Cardamone, D. Loiacono, and P. L. Lanzi, "Applying cooperative coevolution to compete in the 2009 TORCS endurance world championship," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 1–8.

[16] E. Choi, S.-H. Shin, J.-K. Ryu, K.-I. Jung, S.-Y. Kim, and M.-H. Park, "Commercial video games and cognitive functions: Video game genres and modulating factors of cognitive enhancement," *Behav. Brain Functions*, vol. 16, no. 1, pp. 1–14, Dec. 2020.

[17] D. Cliff and G. F. Miller, "Co-evolution of pursuit and evasion II: Simulation methods and results," in *Proc. 4th Int. Conf. Simul. Adapt. Behav.* Cambridge, MA, USA: MIT Press, 1996, pp. 506–515.

[18] M. N. Collazo, C. Cotta, and A. J. Fernández-Leiva, "Virtual player design using self-learning via competitive coevolutionary algorithms," *Natural Comput.*, vol. 13, no. 2, pp. 131–144, Jun. 2014.

[19] M. Cook, S. Colton, and J. Gow, "Initial results from co-operative co-evolution for automated platformer design," in *Applications of Evolutionary Computation* (Lecture Notes in Computer Science), vol. 7248. Berlin, Germany: Springer, 2012, pp. 194–203.

[20] M. Cook, S. Colton, and J. Gow, "The ANGELINA videogame design system—Part I," *IEEE Trans. Comput. Intell. AI Games*, vol. 9, no. 2, pp. 192–203, Jun. 2017.

[21] F. F. Duarte, N. Lau, A. Pereira, and L. P. Reis, "A survey of planning and learning in games," *Appl. Sci.*, vol. 10, no. 13, p. 4529, Jun. 2020.

[22] A. Dziuk and R. Miikkulainen, "Creating intelligent agents through shaping of coevolution," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2011, pp. 1077–1083.

[23] B. Edhlund and A. McDougall, *NVivo 12 Essentials*. Stallarholmen, Sweden: Form & Kunskap, AB, 2019.

[24] T. Erez and D. W. Smart, "What does shaping mean for computational reinforcement learning?" in *Proc. 7th IEEE Int. Conf. Develop. Learn.*, Aug. 2008, pp. 215–219.

[25] A. Fernandez-Ares, M. A. Mora, M. Garcia-Arenas, J. J. M. Guervos, P. Garcia-Sanchez, and A. P. Castillo, "Co-evolutionary optimization of autonomous agents in a real-time strategy game," in *Applications of Evolutionary Computation* (Lecture Notes in Computer Science), vol. 8602. Berlin, Germany: Springer, 2014, pp. 374–385.

[26] S. G. Ficici and J. B. Pollack, "Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states," in *Proc. 6th Int. Conf. Artif. Life*, 1998, pp. 238–247.

[27] G. S. Ficici and B. J. Pollack, "A game-theoretic memory mechanism for coevolution," in *Genetic and Evolutionary Computation*, E. Cantú-Paz, J. A. Foster, K. Deb, L. D. Davis, R. Roy, U.-M. O'Reilly, H.-G. Beyer, R. Standish, G. Kendall, S. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. A. Potter, A. C. Schultz, K. A. Dowsland, N. Jonoska, and J. Miller, Eds. Berlin, Germany: Springer, 2003, pp. 286–297.
[28] D. Floreano and S. Nolfi, "God save the red queen! Competition in co-evolutionary robotics," in *Proc. 2nd Conf. Genetic Program.*, 1997, pp. 1–9.
[29] A. Hintze, S. R. Olson, and J. Lehman, "Orthogonally evolved AI to improve difficulty adjustment in video games," in *Applications of Evolutionary Computation* (Lecture Notes in Computer Science), vol. 9597. Cham, Switzerland: Springer, 2016, pp. 525–540.
[30] H. J. Hong and B. S. Cho, "Evolving reactive NPCs for the real-time simulation game," in *Proc. IEEE Symp. Comput. Intell. Games (CIG)*, Apr. 2005, pp. 86–93.
[31] R. W. Johnson, M. E. Melich, Z. Michalewicz, and M. Schmidt, "Coevolutionary TEMPO game," in *Proc. Congr. Evol. Comput.*, vol. 2, Jun. 2004, pp. 1610–1617.
[32] D. Keaveney and C. O'Riordan, "Abstract model of a real time strategy game," Nat. Univ. Ireland, Galway, Ireland, Tech. Rep. NUIG-IT-011008, 2008.
[33] D. Keaveney and C. O'Riordan, "Evolving robust strategies for an abstract real-time strategy game," in *Proc. IEEE Symp. Comput. Intell. Games*, Sep. 2009, pp. 371–378.
[34] D. Keaveney and C. O'Riordan, "Evolving coordination for real-time strategy games," *IEEE Trans. Comput. Intell. AI Games*, vol. 3, no. 2, pp. 155–167, Jun. 2011.
[35] K. Kopeć, "Indirect financing in culture on the example of tax relief in the video game industry," *Edukacja Ekonomistów Menedżerów*, vol. 54, no. 4, pp. 45–60, 2019.
[36] R. Lara-Cabrera, C. Cotta, and J. A. Fernandez-Leiva, "A review of computational intelligence in RTS games," in *Proc. IEEE Symp. Found. Comput. Intell. (FOCI)*, Apr. 2013, pp. 114–121.
[37] R. Leigh, J. Schonfeld, and J. S. Louis, "Using coevolution to understand and validate game balance in continuous games," in *Proc. 10th Annu. Conf. Genetic Evol. Comput.*, 2008, pp. 1563–1570.
[38] P. Lichocki, K. Krawiec, and W. Jaskowski, *Evolving Teams Cooperating Agents for Real-Time Strategy Game*. Berlin, Germany: Springer, 2009, pp. 333–342.
[39] J. Liu, D. Perez-Liebana, and S. M. Lucas, "Rolling horizon coevolutionary planning for two-player video games," in *Proc. 8th Comput. Sci. Electron. Eng. (CEEC)*, Sep. 2016, pp. 174–179.
[40] S. Liu, S. J. Louis, and C. A. Ballinger, "Evolving effective microbehaviors in real-time strategy games," *IEEE Trans. Comput. Intell. AI Games*, vol. 8, no. 4, pp. 351–362, Dec. 2016.
[41] D. Livingstone, "Coevolution in hierarchical AI for strategy games," in *Proc. IEEE Symp. Comput. Intell. Games (CIG)*, Apr. 2005, pp. 190–194.
[42] S. Luke, C. Hohn, J. Farris, G. Jackson, and J. Hendler, "Co-evolving soccer softbot team coordination with genetic programming," in *Robot Soccer World Cup I*, H. Kitano, Ed. Berlin, Germany: Springer, 1997, pp. 398–411.
[43] A. Martín-Martín, E. Orduna-Malea, M. Thelwall, and E. D. López-Cózar, "Google scholar, web of science, and scopus: A systematic comparison of citations in 252 subject categories," *J. Informetrics*, vol. 12, no. 4, pp. 1160–1177, Nov. 2018.
[44] T. Menezes and E. Costa, "Coevolution of competing agent species in a game-like environment," in *Applications of Evolutionary Computing* (Lecture Notes in Computer Science), vol. 5484, Berlin, Germany: Springer, 2009, pp. 263–272.
[45] C. Miles and J. S. Louis, "Towards the co-evolution of influence map tree based strategy game players," in *Proc. IEEE Symp. Comput. Intell. Games (CIG)*, May 2006, pp. 75–82.
[46] C. Miles, J. Quiroz, R. Leigh, and J. S. Louis, "Co-evolving influence map tree based strategy game players," in *Proc. IEEE Symp. Comput. Intell. Games (CIG)*, Apr. 2007, pp. 88–95.
[47] G. F. Miller and D. Cliff, "Co-evolution of pursuit and evasion I: Biological and game-theoretic foundations," School Cogn. Comput. Sci., Univ. Sussex, Brighton, U.K., Tech. Rep. CSRP311, 1994.
[48] G. A. Monroy, K. O. Stanley, and R. Miikkulainen, "Coevolution of neural networks using a layered Pareto archive," in *Proc. 8th Annu. Conf. Genetic Evol. Comput. (GECCO)*, Jul. 2006, pp. 329–336.
[49] G. Nitschke, "Co-evolution of cooperation in a pursuit evasion game," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, vol. 2, Oct. 2003, pp. 2037–2042.
[50] J. Noble and A. R. Watson, "Pareto coevolution: Using performance against coevolved opponents in a game as dimensions for Pareto selection," in *Proc. Genetic Evol. Comput. Conf.*, 2001, pp. 493–500.
[51] M. Nogueira, C. Cotta, and J. A. Fernandez-Leiva, "An analysis of hall-of-fame strategies in competitive coevolutionary algorithms for self-learning in RTS games," in *Learning and Intelligent Optimization* (Lecture Notes in Computer Science), vol. 7997. Berlin, Germany: Springer, 2013, pp. 174–188.
[52] M. Nogueira, M. J. Galvez, C. Cotta, and J. A. Fernandez-Leiva, "Hall-of-fame competitive coevolutionary algorithms for optimizing opponent strategies in a new game," in *Proc. 13th Int. Conf. Intell. Games Simul. (GAME-ON)*, 2012, pp. 71–78.
[53] M. Nogueira-Collazo, C. C. Porras, and A. J. Fernandez-Leiva, "Competitive algorithms for coevolving both game content and AI. A case study: Planet wars," *IEEE Trans. Comput. Intell. AI Games*, vol. 8, no. 4, pp. 325–337, Dec. 2016.
[54] S. Nolfi and D. Floreano, "Coevolving predator and prey robots: Do 'arms races' arise in artificial evolution?" *Artif Life*, vol. 4, no. 4, pp. 35–311, 1998.
[55] S. Ontanon, 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 Trans. Comput. Intell. AI Games*, vol. 5, no. 4, pp. 293–311, Dec. 2013.
[56] G. Parker, "Punctuated anytime learning for autonomous agent control," in *Studies in Systems, Decision and Control*, vol. 27. Cham, Switzerland: Springer, 2015, pp. 89–107.
[57] D. Perez, S. Samothrakis, S. Lucas, and P. Rohlfshagen, "Rolling horizon evolution versus tree search for navigation in single-player real-time games," in *Proc. 15th Annu. Conf. Genetic Evol. Comput. Conf. (GECCO)*, New York, NY, USA, 2013, pp. 351–358.
[58] M. Petticrew and H. Roberts, *Systematic Reviews in the Social Sciences: A Practical Guide*. Williston, ND, USA: Wiley, 2005.
[59] D. Pinelle, N. Wong, and T. Stach, "Using genres to customize usability evaluations of video games," in *Proc. Conf. Future Play Res., Play, Share Future Play*, New York, NY, USA, 2008, pp. 129–136.
[60] S. Priesterjahn, O. Kramer, A. Weimer, and A. Goebels, "Evolution of human-competitive agents in modern computer games," in *Proc. IEEE Int. Conf. Evol. Comput.*, Jul. 2006, pp. 777–784.
[61] J. Putzke, K. Fischbach, D. Schoder, and A. P. Gloor, "The evolution of interaction networks in massively multiplayer online games," *J. Assoc. Inf. Syst.*, vol. 11, no. 2, pp. 69–94, 2010.
[62] A. Rawal, P. Rajagopalan, and R. Miikkulainen, "Constructing competitive and cooperative agent behavior using coevolution," in *Proc. IEEE Conf. Comput. Intell. Games*, Aug. 2010, pp. 107–114.
[63] W. C. Reynolds, "Competition, coevolution and the game of tag," in *Proc. 4th Int. Workshop Synth. Simul. Living Syst.*, 1994, pp. 59–69.
[64] C. D. Rosin and R. K. Belew, "New methods for competitive coevolution," *Evol. Comput.*, vol. 5, no. 1, pp. 1–29, Mar. 1997.
[65] S. Rudolph, S. V. Mammen, J. Jungbluth, and J. Hahner, "Design and evaluation of an extended learning classifier-based StarCraft micro AI," in *Applications of Evolutionary Computation* (Lecture Notes in Computer Science), vol. 9597. Cham, Switzerland: Springer, 2016, pp. 669–681.
[66] A. S. Ruela and F. G. Guimaraes, "Coevolutionary procedural generation of battle formations in massively multiplayer online strategy games," in *Proc. Brazilian Symp. Comput. Games Digit. Entertainment*, Nov. 2014, pp. 89–98.
[67] A. S. Ruela and F. G. Guimarães, "Procedural generation of non-player characters in massively multiplayer online strategy games," *Soft Comput.*, vol. 21, no. 23, pp. 7005–7020, Dec. 2017.
[68] N. Shaker, J. Togelius, and M. J. Nelson, *Procedural Content Generation in Games*, 1st ed. Cham, Switzerland: Springer, 2016.
[69] M. P. Singh, A. S. Rao, and M. P. Georgeff, *Formal Methods in DAI: Logic-Based Representation and Reasoning*. Cambridge, MA, USA: MIT Press, 2020, pp. 331–376.
[70] G. Smith, M. P. Avery, R. Houmanfar, and J. S. Louis, "Using co-evolved RTS opponents to teach spatial tactics," in *Proc. IEEE Conf. Comput. Intell. Games (CIG)*, Aug. 2010, pp. 146–153.
[71] M. Szubert, W. Jaśkowski, P. Liskowski, and K. Krawiec, "Shaping fitness function for evolutionary learning of game strategies," in *Proc. 15th Annu. Conf. Genetic Evol. Comput. Conf. (GECCO)*, New York, NY, USA, 2013, pp. 1149–1156.
[72] T. Thompson, J. Levine, and G. Hayes, "EvoTanks: Co-evolutionary development of game-playing agents," in *Proc. IEEE Symp. Comput. Intell. Games*, Apr. 2007, pp. 328–333.

[73] J. Togelius, P. Burrow, and S. M. Lucas, "Multi-population competitive co-evolution of car racing controllers," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2007, pp. 4043–4050.
[74] P. V. Torres-Carrion, C. S. Gonzalez-Gonzalez, S. Aciar, and G. Rodriguez-Morales, "Methodology for systematic literature review applied to engineering and education," in *Proc. IEEE Global Eng. Educ. Conf. (EDUCON)*, Apr. 2018, pp. 1364–1373.
[75] E. Uchibe and M. Asada, "Incremental coevolution with competitive and cooperative tasks in a multirobot environment," *Proc. IEEE*, vol. 94, no. 7, pp. 1412–1424, Jul. 2006.
[76] Y. Wang, Y. Qi, and Y. Li, "Memory-based multiagent coevolution modeling for robust moving object tracking," *Sci. World J.*, vol. 2013, pp. 1–13, Jan. 2013.
[77] C. H. Yong and R. Miikkulainen, "Cooperative coevolution of multi-agent systems," Univ. Texas, Austin, TX, USA, Tech. Rep. AI01-287, 2001.
[78] C. J. Young, "Game changers: Everyday gamemakers and the development of the video game industry," Ph.D. dissertation, Dept. Inf., Univ. Toronto, Toronto, ON, Canada, 2018.

EHAB Z. ELFEKY (Member, IEEE) received the Ph.D. degree in computer science from the University of New South Wales (UNSW) Canberra, Australia, in 2009. He is currently a Research Associate with the School of Engineering and Information Technology, UNSW. His gained expertise includes areas but not limited to solving optimization problems in a parallel environment for both large-scale and multi-objective domains. His current research interests include evolutionary computation and real-time strategy games using coevolutionary Intelligence.

SABER ELSAYED (Member, IEEE) received the Ph.D. degree in computer science from the University of New South Wales (UNSW) Canberra, Australia, in 2012. He is currently a Senior Lecturer with the School of Engineering and Information Technology, UNSW. His research interests include evolutionary computation and scheduling and swarm control using computational intelligence. He served as the Chair for the IEEE Computational Intelligence Society (ACT Chapter), from 2019 to 2020. He also serves as an associate editor for one international journal and an organizing committee member of different conferences in the evolutionary computation field.

LUKE MARSH received the bachelor's degree in computer science from Newcastle University, in 2001, and the master's degree in sciences from The University of Adelaide, in 2008. He is currently a Discipline Leader with the Defence Science and Technology Group, with over 15 years of experience applying AI and computational intelligence to real world problems. His main research interests include the application of computational intelligence and machine learning algorithms to military simulations for tactics learning.

DARYL ESSAM received the B.Sc. degree in computer science from the University of New England, Australia, in 1990, and the Ph.D. degree from the University of New South Wales, Australia, in 2000. Since 1994, he has been with UNSW at Canberra, where he is currently a Senior Lecturer. His research interests include genetic algorithms, with a focus on both evolutionary optimization and large-scale problems.

MADELEINE COCHRANE received the degree (Hons.) in robotics and mechatronics engineering from Swinburne University of Technology and the bachelor's degree in computer science from Swinburne University of Technology, in 2018. She is currently a Researcher with the Defence Science and Technology Group, Australia. Her current research interest includes distributed multi-agent decision making.

BRENDAN SIMS received the Bachelor of Mechatronic Engineering degree (Hons.) from The University of Adelaide, in 2008. He has been employed with DST Group, since 2009, and in that time has worked as part of the Land Division (formerly, Land Operations Division). His research interests include agent-based distributed decision-making methods and their application to autonomous systems in scenarios of interest to the Australian Defence Force.

RUHUL SARKER (Member, IEEE) received the Ph.D. degree from Dalhousie University (former, TUNS), Canada. He has been a Professor with the School of Engineering and IT (SEIT), and the Director of Faculty PG Research, UNSW Canberra (located at ADFA), since May 2015. He served as the Deputy Head for the School (Research) of SEIT, from 2011 to 2014. His research interests include decision analytics, CI/evolutionary computation, operations research, and applied optimization.