Swarm and Evolutionary Computation 69 (2022) 101017

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Grammar-based cooperative learning for evolving collective behaviours in multi-agent systems

Dilini Samarasinghe*, Michael Barlow, Erandi Lakshika, Kathryn Kasmarik

School of Engineering and IT, University of New South Wales, Canberra, Australia

A R T I C L E I N F O

Keywords:
Grammatical evolution
Heterogeneous multi-agent systems
Team learning
Cooperative coevolution
Swarm intelligence

A B S T R A C T

This paper presents a novel grammar-based evolutionary approach which allows autonomous emergence of heterogeneity in collective behaviours. The approach adopts a context-free grammar to describe the syntax of evolving rules, which facilitates an evolutionary algorithm to evolve rule structures without manual intervention. We propose modifications to the genome structure to address the requirements of heterogeneity, and two cooperative learning architectures based on team learning and cooperative coevolution. Experimental evaluations with four behaviours illustrate that both architectures are successful in evolving heterogeneous collective behaviours. Both heterogeneous architectures surpass a homogeneous model in performance for deriving a flocking macro behaviour, however the homogeneous model is superior for evolving micro behaviours such as cohesion and alignment. The results infer that by placing the entire set of agent rules and their syntax under evolutionary control, effective solutions to complex problems can be evolved when human knowledge and intuition becomes insufficient.

1. Introduction

Multi-agent systems (MASs) are increasingly being adopted as a viable solution to model, study and understand the real life complexities and nonlinear interactions of dynamic systems in many application fields.

The multifaceted nature of most of the real world requirements of MASs could be addressed by employing heterogeneity to flexibly adopt to dynamic conditions and face and recover from failures with maximum robustness. Nevertheless, designing heterogeneous MASs is still a challenge that has only been partially addressed as a result of the related complications. It is particularly more challenging than homogeneous systems due to the need of designing each individual agent or sub-groups of agents separately such that they cooperatively act towards a common goal [1,2]. This requires exploring a substantially diverse search space of different rule components, parameters and values which is a complex, time consuming endeavour. Manually exploring the search space and designing the behavioural rules is difficult and not feasible, as mere intuition is insufficient to foresee which combination of individual rules and/or their components will result in the desired behaviour at the emergent level [3,4]. Autonomous design techniques that have often been explored as alternatives to address these challenges are primarily concentrated only on automating the control of parameters necessary for behavioural rules formulation. They rarely explore the capability for automating the design of the structure of the rules that are being evolved. The rule structures are mostly still either manually designed [5,6] or are represented by an artificial neural network (ANN) [7] that hinders understanding and reverse engineering of the rules for analysis and combination of individual components for complex heterogeneous behaviour generation [8].

Grammatical evolution (GE), first proposed in 1998 [9], is a technique which is intrinsically focused on the structure of the rules being evolved. In contrast to other automated design techniques, it adopts a context-free grammar (CFG) in defining the syntax of the evolving behaviours and has the ability to maintain search space and solution space independent of each other through a separate mapping process. This makes GE an ideal candidate approach that can evolve entire rule structures to explore diverse solutions for complex problem domains of MASs. As such, this paper investigates GE as a potential solution to support autonomous emergence of heterogeneity among agents. The content of this paper further elaborates the challenges in using evolutionary computation for evolving heterogeneous behaviours such as the computational cost associated with the size of the population and premature convergence of solution spaces. We present two cooperative learning architectures; one based on team learning (TL) and another based on cooperative coevolution (CCE) [10], using grammar-based evolution in order to address the said challenges. Experimental evaluations are conducted to present a coherent view of the strengths and weaknesses of

MAS, Multi-agent System; GE, Grammatical Evolution; CFG, Context-free Grammar; TL, Team Learning; CCE, Cooperative Coevolution.

* Corresponding author.
E-mail address: d.samarasinghe@adfa.edu.au (D. Samarasinghe).

https://doi.org/10.1016/j.swevo.2021.101017
Received 18 January 2021; Received in revised form 25 August 2021; Accepted 10 November 2021
Available online 27 November 2021

each mechanism and a selection criteria for using these methods for generation of collective multi-agent behaviours.

In a recent study [11,12] we introduced a GE-based mechanism to synthesise multi-agent behaviours for a homogeneous system, which is capable of reducing human intervention in the rule generation process. In contrast to homogeneous agents, heterogeneous agent rules can complicate the process of learning as the search space for rules becomes proportional to the number of agents in the system increasing the complexity [10]. In this paper, we study how the previous mechanism could be significantly enhanced to address such complications and model heterogeneous behaviours with the following contributions:

1. A GE-based approach for synthesis of heterogeneous multi-agent behavioural rules is introduced. Unlike the existing mechanisms which require the rule structure to be pre-defined, this approach can evolve the entire rule structure from their atomic components based on a grammar which outlines the syntax of rules.
2. A novel encoding mechanism is proposed for GE which can encode multiple behavioural rules (corresponding to different agents) in a single genome. This facilitates the representation of rules required for cooperative learning.
3. Cooperative learning mechanisms based on two architectures: TL and CCE, are proposed for implementation of the grammar-based model. These mechanisms explore means to reduce computational costs associated with expanding agent group sizes and to avoid the evolution process getting stuck in sub-optimal solutions.
4. The effectiveness of the proposed models is analysed based on evolutionary results in a simulation environment with four behaviours and the results are also compared against a homogeneous model.

The rest of the paper is organised as follows. The relevant existing literature is reviewed in Section 2. Section 3 introduces the problem statement in relation to heterogeneous MASs, and presents the general framework of the proposed grammar-based evolutionary approach with the modified genome encoding mechanism. It also introduces the cooperative learning mechanisms used in combination with the grammar-based model. The experimental setups and evaluations are presented in Section 4 and Section 5, respectively. Section 6 discusses the results and Section 7 concludes the paper with possible future directions.

2. Related work

2.1. Evolution of heterogeneous multi-agent systems

Evolutionary algorithms have often been discussed as an effective approach to design MASs [13,14] and have also been used in heterogeneous contexts leveraging behavioural or structural diversity to address more complicated problems. Heterogeneity has long been explored in the context of evolutionary computing in fields such as robotics [15,16], surveillance [17], traffic management [18], hazardous environments [19], mapping and exploration [20], and construction [21]. With the recent advancements in the areas of social networking and data sharing, evolutionary techniques are also being used in MASs that interact with such systems for social networking predictions [22]. Motion detection, social behaviour, and drift detection [23] are other areas that have found interest in evolutionary techniques coupled with MASs to support dynamic pattern mining. More recent MASs-based work has also explored evolutionary algorithms in areas such as game design [24] and graphical simulations [25].

Two cooperative learning architectures: TL and concurrent learning [10], are discussed in the literature to facilitate the evaluation of interactions among agents in determining their contributions at the emergent level. Only a single learner is associated with TL which will search and improve all the behavioural rules of the agents in the system [26]. Hence, multiple agents should be encoded in a single genome which will then learn the behaviours of the entire agent system. It has centralised control over the multi-agent system and lacks the property of breaking a larger problem into manageable sub problems, thus leading a single learner to explore a large solution space. Due to this fact, TL is generally used with evolution of homogeneous agent systems.

Concurrent learning on the other hand, involves multiple learners working on different parts of the agent system. Generally, each agent or agent group has a separate learner to modify their behaviour. CCE is one of the common concurrent learning mechanisms, where separate populations are used to coevolve different agents resulting in a solution space with cooperating sub modules. The evolution is carried out by evaluating each individual for their performance with the individuals of other populations. This approach reduces the workload of the learner by decomposing the problem into more manageable sub-problems [27] and is typically used with heterogeneous evolution.

These learning architectures have been studied with other reward-based techniques such as reinforcement learning [28] and genetic algorithms [27]. More recently, Deng et.al [29] have explored an ant colony optimisation algorithm with cooperative coevolution in a multi-population strategy. In a similar vein, a multi-objective bacterial foraging algorithm [30] and distributed combinatorial optimisation heuristic approaches [31] have been tested with cooperative learning architectures to explore their capacity in collective behaviour evolution. Nevertheless, these architectures have not been experimented with GE in previous literature. The proposed models cater to this gap by investigating modifications to cooperative learning architectures in association with GE in order to support evolution of heterogeneous multi-agent behaviours.

2.2. Challenges in evolution of heterogeneous multi-agent systems

We investigate grammar-based cooperative learning in addressing the following limitations associated with the current evolutionary approaches for heterogeneous MASs:

Human bias in synthesis process: The classic evolutionary approaches concentrate on automatically generating only the parameters associated with formulating behavioural rules rather than exploiting means for automating the generation of the entire rule structure [5,6]. In a heterogeneous context, it is not trivial to determine the aggregated set of local behaviours to result in a desired global behaviour. The approaches that have explored heterogeneity have in fact used potential functions and parameters that require heavy manual intervention [32]. We explore GE as a solution to reduce human bias by evolving the entire rule structure based on a predefined syntax.

Limitations on scalability: The cost of computation is generally associated with the population size [33]. In a typical concurrent learning approach for heterogeneous systems, individual agents have to be learned separately. The search space to be explored increases proportionally as the number of agents increases, thus increasing the complexity of the process causing scalability issues [27]. We explore modifications to the genome structure of GE to represent multiple agents in a single genome, enabling TL with a single learner to be incorporated for heterogeneous learning.

Premature convergence of the solution space: Multi population environments can suffer from premature convergence to equilibrium states. Relative over-generalisation is one of the main causes for co-evolutionary algorithms to converge towards sub-optimal equilibrium states without reaching for the optimal solutions [34]. The populations may deceive each other to settle at a sub-optimal solution in such instances [35]. We investigate extensions to traditional CCE learning to help the evolutionary process escape from sub-optimal solutions.

3. Grammar-based evolutionary approach

This section discusses the proposed grammar-based evolutionary approach along with the modification introduced to the genome structure and the cooperative learning architectures.

3.1. Problem statement

Heterogeneous multi-agent systems are naturally distinguished by the behavioural and/or morphological diversity of the constituent agents. Morphological diversity refers to the structural dissimilarity within agents such as different actuation and sensing capabilities. In contrast, behavioural diversity allows different behavioural specialisations within a group of agents [27].

In the context of this paper, we concentrate on achieving behavioural heterogeneity through individual specialisation of the agent rules within a group of morphologically homogeneous agents. We define a heterogeneous multi-agent system with the tuple $\mathrm{H} = \{ \Psi, R, g \}$ where;

- $\Psi = \{ \psi_1, \psi_2, \ldots, \psi_\mu ; 1 < \mu \}$ is a set of more than one agent ψ, where μ is the number of agents.
- $R = \{ r(\psi_1), r(\psi_2), \ldots, r(\psi_\mu) \}$ is a set of rules $r(\psi)$ that define the agent behaviours, each associated with a single agent in the system.
- g is the goal state the entire agent system is required to achieve defined by the objective function $O(g)$.

The objective $O(g)$ can be achieved by optimising a global fitness function which defines the criteria for the entire set of agents to reach the goal state as defined in Function 1;

$$O(g) \Leftarrow \max \ F(\Psi) \tag{1}$$

However, since agents follow distinct rules, the contributions of the agents towards the final goal can be different. Therefore, in a second approach, the individual contribution can also be taken into account along with the global performance. Function 2 optimises both the global performance as well as the individual contribution;

$$O(g) \Leftarrow \max \ F(\Psi) \ \&\& \ \max \sum_{i=1}^{\mu} F(\psi_i) \tag{2}$$

Given a set Ψ of agents and a goal g, the aim is to optimise the fitness either using the Function 1 or Function 2 to achieve the rule set R which resembles the closest possible behaviour to the goal g.

3.2. General framework

Computational methods that can tackle optimisation problems (such as reinforcement learning, particle swarm optimisation, evolutionary algorithms including genetic algorithms, estimation of distribution algorithms and evolution strategy algorithms) can often also be used to improve emergent behaviour of MASs. Nevertheless, these methods can only be used in exploring the space of states and behaviour parameters of agents [28,33,36] rather than the space of behaviour rule structures. They are limited in their capacity to incorporate the structure of a programme in the evolution process. Genetic programming (GP) [37] has the potential to evolve programme structures, however, has no control over their syntax. As such, GE can be identified as a more logical alternative that has the unique ability to utilise a CFG to control and restrict the solution structures to a desired syntax.

This approach is increasingly being used in evolving behaviours of single agent systems [38–40] as well as homogeneous agent systems [8,11], due to its ability to generate syntactically correct solutions through the evolutionary process with reduced manual intervention. However, evolution of heterogeneous groups of agents has not been explored in a comprehensive manner with grammar-based approaches. As such, this paper focuses on allowing autonomous emergence of heterogeneous rule structures in a cooperative environment where the individual agents succeed or fail in collaboration. For this, two approaches for cooperative learning; using a single learner for the entire team (TL) and using multiple concurrent sub learners (CCE) are explored along with the GE model.

In our previous work [11], we proposed a mechanism to evolve emergent homogeneous behaviours by employing the atomic components of the rule structures; control structures, parameters, preliminary actions and logical and/or relational connectives [11]. This structural definition remains true for aggregated sets of rules as well, since individual rules are combined using logical connectives to form the aggregation. Figure 1 illustrates the syntax formulation mechanism.

The aggregated rules can consist of multiple rule components and the process is started by initialising the first component. It follows an *if-else* control structure, and the process then selects the condition composed of logical and/or relational connectives followed by their parameters. The next part leads to the actions or another *if* condition (resulting a nested rule). It is then followed by the *else* component, which could again be an action or another *if* condition nested within. The process may end there or continue to generate more rule components.

The evolutionary process is outlined in Algorithm 1 and Fig. 2.

Algorithm 1 Grammar-based Evolutionary Process.

```
Require: $\mathfrak{R}$: Rule Space$\Omega$: Maximum no. of generations$\beta$: Size of the
    population$CL$: Cooperative learning mechanism (TL/CCE)
Ensure: $I_b$: Best individual genome
 1: procedure GRAMMARBASEDEVOLUTION($\mathfrak{R}, \beta, \Omega$)
 2:     $\rho \leftarrow$ GENERATECFG($\mathfrak{R}$)
 3:     $pop \leftarrow$ INITIALISEPOPULATION($\beta, \rho$)
 4:     $pop \leftarrow$ EVOLUTIONARYALGORITHM($pop, \beta, \rho, \Omega, CL$)
 5:     $I_b \leftarrow$ GETMOSTFITINDIV($pop$)
 6:     return $I_b$
 7: procedure EVOLUTIONARYALGORITHM($pop, \beta, \rho, \Omega, CL$)
 8:     $\omega \leftarrow 0$
 9:     while $\omega \neq \Omega$ do
10:         EVALUATEFITNESS($pop$)
11:         while $validateAgainstCFG(pop) == false$ do
12:             PARENTSELECTION($pop$)
13:             GENETICOPERATIONS($pop$)
14:     return $pop$
```

The rule space which consists of the four types of atomic components (control structure, logical and relational connectives, parameters, and elementary actions) is used to design the CFG syntax illustrated in Fig. 1 (Algorithm 1: Line 2). The designed CFG is then used to generate an initial random population of rules which adhere to the required syntax, which is then fed to the evolutionary algorithm to evolve the desired behaviours (Algorithm 1: Line 3). As identified earlier in this section, we propose two cooperative learning algorithms; TL and CCE, which are presented in detail under Section 3.4. The algorithm presented here describes the structure of the evolutionary process common to both TL and CCE mechanisms (Algorithm 1: Lines 7–14). The fitness is calculated for each individual rule in the population and two parent rules are selected with the highest fitness values to generate offspring. The offspring generated by applying genetic operations are then validated against the CFG to ensure they adhere to the required syntax. The unique mechanisms adopted by TL and CCE with the process of fitness evaluation, parent selection, and applying genetic operations are discussed in the respective Sections.

3.3. Genome structure

In GE, the genome is represented as a binary string referred to as a set of ‘codons’. The codons are consecutive groups of 8 bits each representing an integer value, and can be mapped to a phenotype with syntactically valid solutions based on the grammar. Since a genome represents a single rule, representing each agent separately to retain heterogeneity is computationally expensive, as the population size increases proportional to the number of agents [41].

Therefore, a mechanism is required to represent multiple agents in the same genome so that the cost of exploration during the evolution process would be minimised. In doing so, we propose the encoding

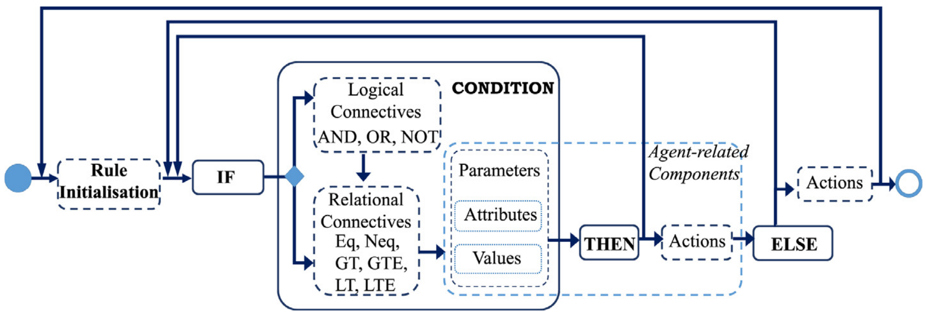

Fig. 1. Syntax followed by the grammar in designing the structure of the behaviour rules. The flow chart describes the steps that should be taken during the formation of a simple or an aggregated rule and allows for the combination of several such rules as required, for achieving complex behaviours.

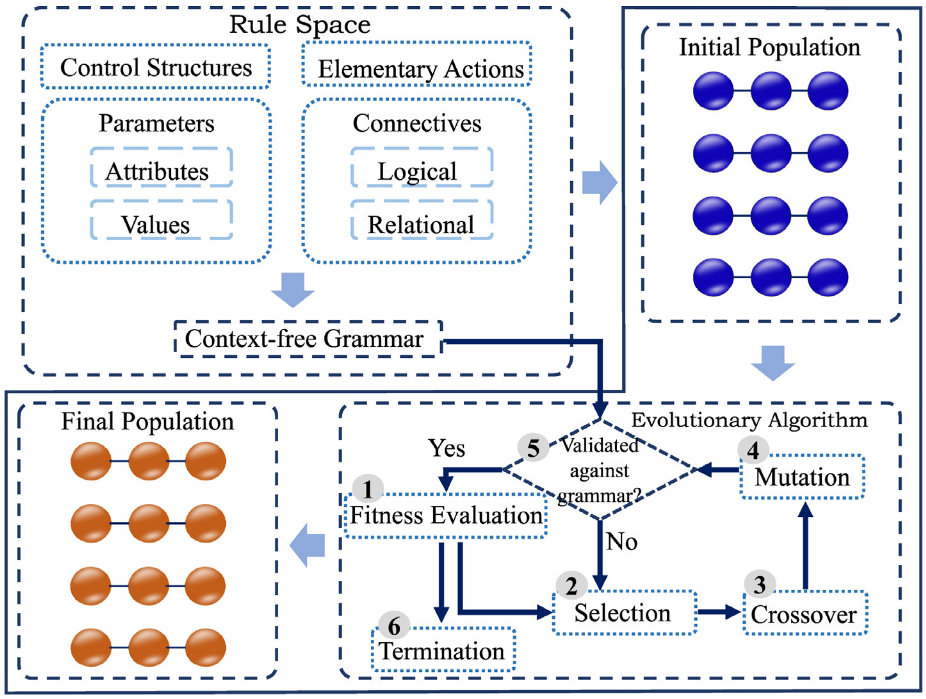

Fig. 2. Grammar-based evolutionary mechanism. Followed by the construction of a context-free grammar based on the rule space, an initial population adhering to the syntax requirements is automatically generated. Every individual in this population is then evaluated for their fitness and parent selection is performed. Offspring are generated by applying crossover and mutation on selected parents. They are validated against the CFG and if they do not comply with the syntax requirements the process is repeated with new parents. The evolutionary process is thus continued until the termination criteria is met. The enclosed section is updated based on the cooperative learning architecture used.

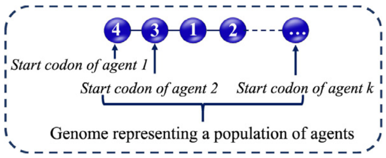

Fig. 3. Proposed encoding mechanism for genomes. A genome consisting of m codons can represent a population of k rules ($k \leq m$). Each rule is interpreted starting from the ith codon ($i \in [1,k]$).

mechanism depicted in Fig. 3 with GE to store multiple agents in one structure.

This proposed genome structure is unique because it can record all characteristics of multiple agents in a single structure while maintaining their heterogeneity. Other evolutionary techniques such as genetic algorithms employ encoding mechanisms which represent multiple variables of a single agent in a single genome. However, they are not capable of representing multiple variables of all agents in one structure without scaling proportionately to the number of agents. This is because the search space and solution space are not independent of each other unlike GE. In contrast, the proposed genome structure uses the same representation of codons, but since they are read in different combinations, they map to entirely different solutions resulting in different agent rules.

In the original GE approach, the genome is mapped to the phenome through the mapping function illustrated in Fig. 4. The function is applied on the corresponding integer value of the codon based on a specified grammar. Multiple rules each starting from the consecutive codon are generated through the same mapping function with the proposed mechanism. For example, in Fig. 4, there are 4 codons [2 4 3 1] and the genome represents two rules, starting from the 1st [2 4 3 1] and 2nd [4 3 1 2] codon respectively.[1]

This mechanism ensures that each agent, although using the same genome, represents different rules based on their respective codon sequence. The only necessary condition is that the number of codons should exceed or be equal to the number of agents that it represents. We adopt the presented genome representation in both TL and CCE contexts for evolution.

3.4. Proposed cooperative learning mechanisms

Traditional coevolutionary algorithms for heterogeneous behaviour generation are limited by scalability and premature convergence issues discussed in Section 2. No single solution could be identified with our investigations that can address both limitations, and as such, both TL and CCE approaches are investigated as means to address each problem. TL, combined with the proposed genome structure is recognised

[1] A description of the syntax and production rules of the grammar used; a detailed illustration of the mapping process with an example rule; and a sensitivity analysis for genetic parameters, is available in supplementary files.

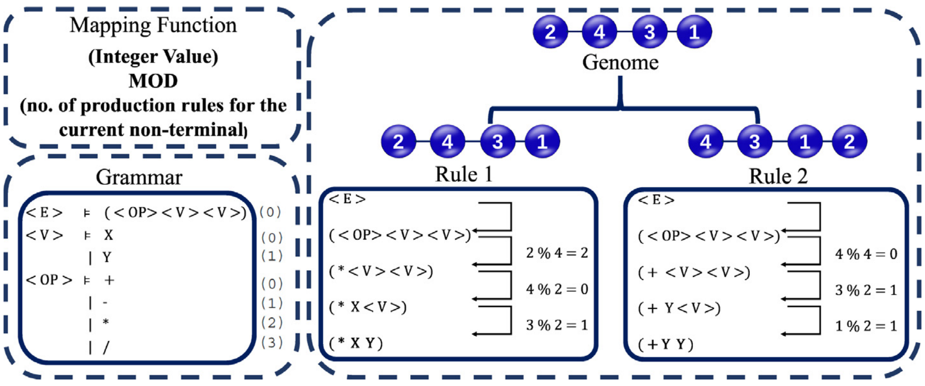

Fig. 4. Example mapping of genotype to phenotype. The genome consists of 4 codons representing 2 rules each interpreted starting from the 1st and 2nd codon. They map to (* X Y) and (+ Y Y) respectively, which are two different rules constructed from the same original genome.

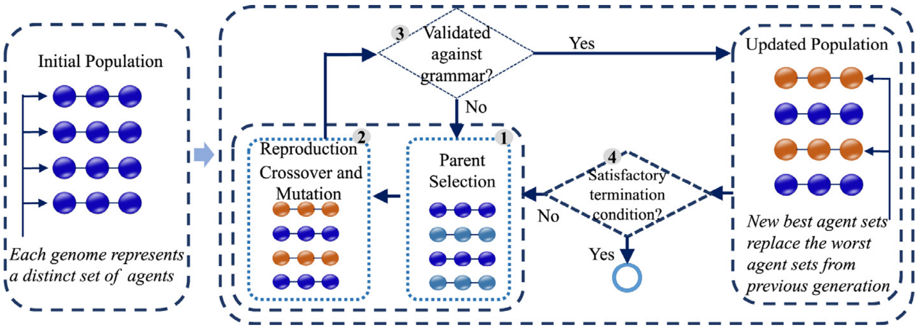
Initial Population
Each genome represents a distinct set of agents
Validated against grammar?
Yes
No
Reproduction Crossover and Mutation
Parent Selection
Satisfactory termination condition?
Yes
No
Updated Population
New best agent sets replace the worst agent sets from previous generation
1
2
3
4

Fig. 5. Proposed team learning mechanism. Each team of agents in the population is separately evaluated for their global fitness and parent selection is performed. The next offspring are generated by applying genetic operators on the selected parents. Each offspring generated is validated against the CFG and the process is repeated with new parents if they do not comply with syntax requirements. The evolutionary process is continued until it meets the termination condition.

as a solution to scalability. Since the entire team can be represented in a single genome, only the genome size is increased with the number of agents, which is less computationally expensive than an increasing population size. The modifications proposed through the extended CCE algorithm are capable of addressing premature convergence issues of traditional CCE. The details of the mechanisms presented here will allow a designer to make decisions on the respective algorithm to choose based on affordable computational resources and expected level of performance.

Team Learning

The aforementioned genome encoding mechanism is used to represent all agents of a team in a single genome for TL. In the proposed approach for heterogeneous systems, the population for evolutionary process is composed of multiple such solutions for global behaviour represented in individual genomes. The evolutionary algorithm is as given in Fig. 5.

Algorithm 2 describes the learning procedure. The population is initialised with multiple genomes, each representing a solution of a different multi-agent group (Algorithm 2: Line 2). During each generation, the genomes are evaluated separately for emergent group behaviour with a global fitness measure (Algorithm 2: Lines 5–6). We adopt steady state replacement (SSR) [42], where the offspring of the selected parents replace the least fit individuals of the last iteration. Hence, based on the fitness values, parent selection is performed on the population (Algorithm 2: Line 11) and genetic operations are applied on the two best solutions identified to generate offspring. The selected parents are first applied with one-point crossover (Algorithm 2: Line 13). As such, a point on both parents is picked randomly and the genome portions from that point are swapped between the two parents. Single point mutation is then applied on the offspring to further facilitate global search of the solution space (Algorithm 2: Lines 15–16). The final offspring solutions generated are then mapped against the CFG to validate that they adhere to the expected syntax (Algorithm 2: Line 17). If the new offspring cannot map to a syntactically correct solution, parent selection and genetic operations will be recurrently performed until they become valid.

Algorithm 2 Team Learning.

```
Require: $\rho$: Set of CFG production rules$\beta$: Size of the population$\Omega$: Maximum no. of generations$\ell$: Maximum no. of loops
Ensure: $I_b$: Best individual genome
1: procedure TEAMLEARNING($\rho, \beta, \Omega$)
2:     $pop \leftarrow$ INITIALISEPOPULATION($\beta, \rho$)
3:     $\omega \leftarrow 0$
4:     while $\omega \neq \Omega$ do
5:         for $i \in pop$ do
6:             EVALUATEFITNESS($i$)
7:         $c_{M1}, c_{M2} \leftarrow null$
8:         $valid \leftarrow false$
9:         $loops++$
10:        while $valid == false$ do
11:            $parents \leftarrow$ PARENTSELECTION($pop$)
12:            for $p1, p2 \in parents$ do
13:                $children \leftarrow$ CROSSOVER($p1, p2$)
14:                for $c1, c2 \in children$ do
15:                    $c_{M1} \leftarrow$ MUTATE($c1, prob_{mut}$)
16:                    $c_{M2} \leftarrow$ MUTATE($c2, prob_{mut}$)
17:            if MAPCFG($\rho, c_{M1}$) == $true$ AND MAPCFG($\rho, c_{M2}$) == $true$ then
18:                $valid \leftarrow true$
19:            else if $loops == \ell$ then
20:                $valid \leftarrow true$
21:        $I_W \leftarrow$ GETTWOWORSTFITINDIVS($pop$)
22:        for $I_{W1}, I_{W2} \in I_W$ do
23:            $I_{W1} \leftarrow$ REPLACEINDIV($I_{W1}, c_{M1}$)
24:            $I_{W2} \leftarrow$ REPLACEINDIV($I_{W2}, c_{M2}$)
25:        $I_b \leftarrow$ GETMOSTFITINDIV($pop$)
26:        $\omega++$
27:    return $I_b$
```

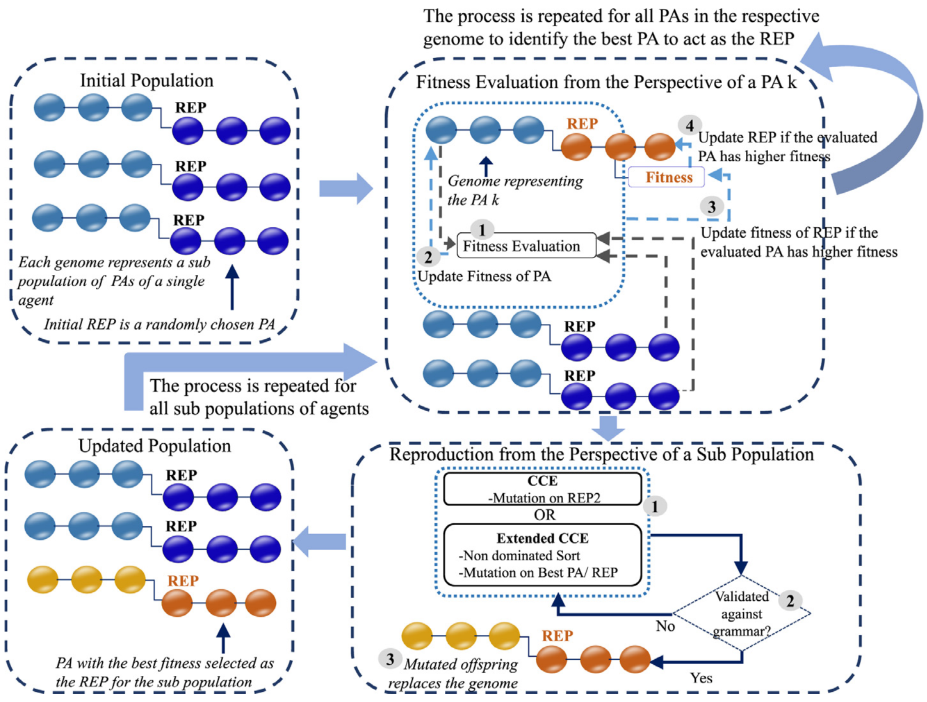
The process is repeated for all PAs in the respective genome to identify the best PA to act as the REP
Initial Population
REP
Each genome represents a sub population of PAs of a single agent
Initial REP is a randomly chosen PA
Fitness Evaluation from the Perspective of a PA k
Genome representing the PA k
Fitness
1 Fitness Evaluation
2 Update Fitness of PA
3 Update fitness of REP if the evaluated PA has higher fitness
4 Update REP if the evaluated PA has higher fitness
The process is repeated for all sub populations of agents
Updated Population
PA with the best fitness selected as the REP for the sub population
Reproduction from the Perspective of a Sub Population
CCE -Mutation on REP2
OR
Extended CCE -Non dominated Sort -Mutation on Best PA/ REP
Validated against grammar?
No
Yes
3 Mutated offspring replaces the genome

Fig. 6. Proposed cooperative coevolution mechanism. The initial population consists of a number of genomes equal to the number of agents in the system. A random PA is selected as the REP at the first generation. Each PA is evaluated for fitness by interaction with REPs from other sub-populations. The REPs are updated at the end of the evaluation phase with the best PA of each sub-population. For the generic CCE approach, the set of genomes are replaced with a mutated version of the REPs for the next generation. For the extended CCE approach, mutation is performed on the best PA/REP after all PAs are sorted based on their local and global fitness values.

A maximum number of loops is used to address the unlikely event when a valid offspring cannot be generated after sufficient number of trials (Algorithm 2: Lines 18–20). When this number is reached, the last generated invalid offspring is placed in the population. During the subsequent fitness evaluation iteration, this offspring receives the worst fitness possible and therefore, it gets replaced immediately afterwards in the next generation. However, this situation is quite rare in practice (was not observed during any of the experimental runs for this study) as there exists a large number of possible codon modifications with parent selections and genetic operators that are capable of generating a valid offspring within a few trials.

Once the offspring is generated, the two least fit individuals from the population are selected which will be replaced with offspring (Algorithm 2: Lines 21–24). Once the evolutionary process is repeated for a desired number of generations and the termination criteria is met, the algorithm returns the best individual solution of the population (Algorithm 2: Lines 25–27).

Cooperative Coevolution Learning

Figure 6 demonstrates the evolution procedure with the CCE approach. The population of solutions is represented by a set of genomes each representing a sub-population of candidate behaviours for a single agent. The goal for the CCE process is to identify the set of best candidate behaviours for the agents that can cooperatively act in the given environment. Each candidate behaviour is evaluated in the presence of other agents during each generation for their capability to represent the actual agent behaviour. We introduce the term ‘phantom agent’ (PA) to refer to these candidate behaviours in the sub-population of the actual agent. During each generation, the PA which results in the best global fitness value based on interaction with the best candidates of all other sub-populations is termed as the ‘representative’ (REP) of that sub-population. Hence, each sub-population consists of multiple PAs which are candidate solutions for REPs, and the best PA takes the position of REP. During the first generation prior to fitness evaluation is begun, a random PA is assigned as the REP for every sub-population. By the end of first generation after iterating through every sub-population, all REPs get replaced with their respective best PA and will continue to be replaced during each generation until the end of the process.

Algorithm 3 further describes the steps of the CCE approach. As discussed above, each sub-population is assigned with a random PA as their REP for the first generation (Algorithm 3: Lines 3–4). During fitness evaluation, each PA in every sub-population is evaluated for their suitability to be elected as the REP. To do so, each PA in a sub-population interacts with an elitist community composed of current REPs of all other sub-populations (Algorithm 3: Lines 7–9). The PA interacts in this community and updates its fitness value based on global performance of the community. At the end of iterating through all PAs for all sub-populations, all existing REPs are updated with the respective best PA of each sub-population unless the existing REP has a better fitness than the new best PA (Algorithm 3: Lines 10–12). Then, offspring are generated. The REP of every sub-population is applied with single point mutation to generate the offspring (Algorithm 3: Line 13). The offspring is then validated against the CFG to ensure they adhere to the syntax. If the offspring is invalid, a different mutation point is chosen, and the process is repeated (Algorithm 3: Lines 22–25). Similar to TL, a maximum number of loops is set to address the rare event that a valid offspring cannot be generated. In that case, after the set number of loops exceeds, a randomly chosen PA is applied with mutation to generate the next offspring (Algorithm 3: Lines 17–19). However, an overrun of the number of loops was not observed during any of the experimental runs conducted. Once the REPs are updated, the evolutionary process is repeated by evaluating their fitness values and applying the genetic operations, until the termination criteria is met.

The genome encoding mechanism proposed above is also used here. Within the genome, the PAs are represented by a different combination of codons, and each PA’s rule should be interpreted starting from the first codon in a circular fashion. In Fig. 4, where the genome is [2 4 3 1], the 1st PA is read from the 1st codon [2 4 3 1], and the 2nd PA is read from the 2nd codon [4 3 1 2].

Figure 7 shows the generation of an offspring for an existing genome where the REP was the third PA. Single point mutation is applied on the REP to generate new offspring and if it is valid (ensuring that it can be mapped to the required number of PAs), it replaces the existing genome. Mutation will be recurrently performed until a valid offspring is generated.

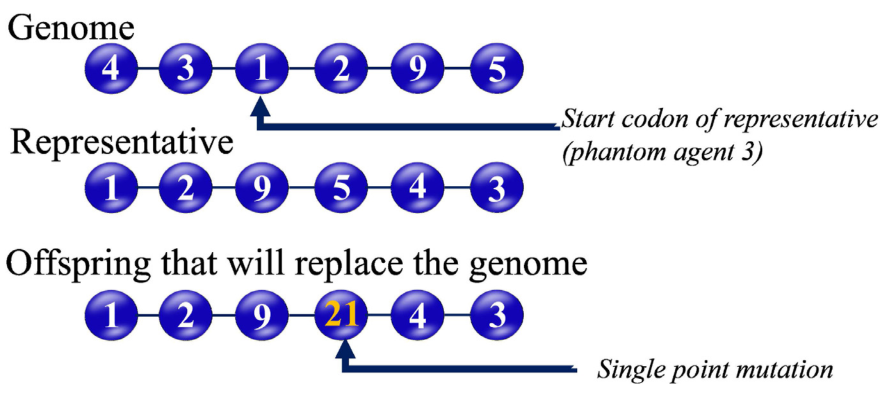
Genome
4 3 1 2 9 5
Start codon of representative (phantom agent 3)
Representative
1 2 9 5 4 3
Offspring that will replace the genome
1 2 9 21 4 3
Single point mutation

Fig. 7. Example generation of offspring from a representative starting from the 3rd codon. Single point mutation is performed on this representative and is used to replace the exising genome for the next generation.

Algorithm 3 Cooperative Coevolution (CCE).

```
Require: ρ: Set of CFG production rulesβ: Size of the populationΩ: Maximum no. of generations
Ensure: $R_b$: Best REP set
1: procedure CooperativeCoevolution($\rho, \beta, \Omega$)
2:     $pop \leftarrow$ InitialisePopulation($\beta, \rho$)
3:     for $i \in pop$ do
4:         $REP_i \leftarrow a\,PA\,randomly\,chosen\,initially$
5:     $\omega \leftarrow 0$
6:     while $\omega \neq \Omega$ do
7:         for $i \in pop$ do
8:             for $PA_k \in i$ do
9:                 EvaluateFitness($PA_k, REP_{set} - REP_i$)
10:        for $i \in pop$ do
11:            if Fitness($REP_i$) > Fitness($PA_{best}$) then
12:                $REP_i \leftarrow PA_{best}$
13:        for $i \in pop$ do
14:            $loops \leftarrow 0$
15:            $valid \leftarrow false$
16:            while $valid == false$ do
17:                if $loops > max\,loops$ then
18:                    $REP_i \leftarrow randomly\,chosen\,PA$
19:                    $loops \leftarrow 0$
20:                $REP_M \leftarrow$ Mutate($REP_i, prob_{mut}$)
21:                for $PA_k \in i$ do
22:                    if MapCFG($\rho, k$) == $false$ then
23:                        loops++
24:                        break
25:                    $valid \leftarrow true$
26:        $R_b \leftarrow$ GetRepresentativeSet($pop$)
27:        $\omega$++
28:    return $R_b$    ▷ addresses a minimisation problem as fitness functions are minimising functions
```

It should also be noted that with the proposed modifications, single point mutation acts as a mechanism to balance both exploration and exploitation of the search space at the same time. Applying only mutation without crossover does not result in a local search which varies only a certain attribute while others remain constant. Rather, this mechanism still facilitates global search. Considering the example illustrated in Fig. 4, if the codon [4] in the genome [2 4 3 1] was changed to [9], it would impact both rules as they become, [2 9 3 1] and [9 3 1 2] resulting in (* Y Y) and (- Y Y) respectively which are both different from their previous rules (* X Y) and (+ Y Y), but still retaining some characteristics from parents. As such, mutation probability is fixed to 1 in order to guarantee variations in the new offspring, and unlike other evolutionary algorithms it does not amount to a complete random exploration. If the probability is made less than 1, it would mean that during every iteration, some sub-populations may remain the same without any novel offspring as no other genetic operation is performed on them, which slows the fitness progression of the overall system.

Extended Cooperative Coevolution Learning

In a heterogeneous context with a common goal, the agents should improve the global fitness cooperatively with the REPs from other sub-populations. The CCE approach looked into improving this fitness, only to discover that the mechanism is still prone to premature convergence as has been observed in the literature with other models that use CCE [27,35].

To illustrate this issue we explain two objectives that motivate an agent in a cooperating heterogeneous environment;

- **global fitness:** the fitness of the global performance achieved by an entire group of agents for a cooperative common goal in the presence of a particular agent.
- **local-fitness:** the individual contribution of a particular agent towards achieving the expected common goal.

For example in an avoidance task; how best every agent avoids every other agent is the global fitness, whereas how best a particular agent is avoiding other agents is that agent's local fitness.

In the previous CCE approach, only the global fitness is used as the evaluation measure of the PAs. Therefore, the evolutionary algorithm may fail to recognise certain PAs which are in fact worth reproducing, simply because individual contribution of PAs is not measured. If a majority of the team performs inadequately while a particular PA is behaving significantly better, it still results in a poor global fitness value, giving the false impression that the presence of that particular PA cannot make an impact on improving performance. On the other hand, a PA performing marginally well in a good team may get a higher fitness simply because of the contribution of other REPs and not its own performance. Continuous selection of such PAs as REPs lead the solutions towards premature convergence to a local optimal.

To overcome this limitation, the extended CCE version employs local fitness to determine the best PA that gets selected as the REP, and both local and global fitness values when generating offspring. We extend the previous CCE algorithm by utilising a non-dominated sorting mechanism as presented in Algorithm 4 for the selection process to rank the list of PAs and the REP of a particular sub-population. Both local and global fitness values are calculated for each PA during the evaluation process. The REP update is done as described in the previous Algorithm 3, but based on the local fitness measure. Parent selection is improved with the given non-dominated sorting algorithm instead of simply choosing the REP. Fig. 6 illustrates this modification under the reproduction component.

As described in Algorithm 4, all PAs and the REP of a sub-population are ranked based on local and global fitness. The algorithm performs a Pareto ranking [43] such that the best rank will be assigned to the Pareto optimal solutions where none of the two objectives are dominated by

Algorithm 4 Sorting of PAs and REP for a Sub Population.

```
Require: $L$: List of PAs + REP
Ensure: $S \cup D$: Sorted list of PAs + REP
1: procedure NONDOMINATEDSORT($L$)
2:     $O \leftarrow$ SORT($L$)
3:     $S, D \leftarrow null$
4:     ADD($S, first\ element\ of\ O$)
5:     REMOVE($O, first\ element\ of\ O$)
6:     return RECURSIVESORT($O$)
7: procedure RECURSIVESORT($O$)
8:     for $i \in O$ do
9:         for $j \in S$ do
10:            if $i \prec j$ then
11:                REMOVE($O, i$)
12:                ADD($D, i$)
13:            if $j \prec i$ then
14:                REMOVE($S, j$)
15:                if $S\ is\ empty$ then
16:                    ADD($S, first\ element\ of\ D$)
17:        if $!(i \prec j)\ for\ all\ j \in S$ then
18:            $S \leftarrow S \cup i$
19:    if $S\ is\ not\ updated$ then
20:        return $S \cup D\ in\ order$
21:    else
22:        return RECURSIVESORT($D$)
```

others. The ranking priority decreases as the solutions get further away from the most efficient ones. The PA or the REP ranked first will be considered for the new offspring. If the new offspring is invalid, mutation will be recurrently performed until a valid offspring is generated. After a certain number of trials, if a valid offspring cannot be generated, the next PA/REP from the ranked list (instead of a random PA as of the previous CCE approach) is selected to replace the genome.

In contrast to multi-objective optimisation that uses similar Pareto ranking procedures, the two objectives here are partially complementary to each other as the local fitness contributes in calculating the global fitness. Furthermore, the goal here is not to find a representative set of Pareto optimal solutions, rather to rank all solutions based on both objectives, such that their rank can be utilised in identifying the best PA/REP for offspring generation.

4. Experimental setup

To study the evolutionary approaches presented, we employ an agent system where the behaviours of bird like autonomous virtual agents (referred to as boids herein) are evolved to achieve a specified task. The boids model was first introduced by Reynolds in [44] where he simulated the aggregated flocking behaviour of a homogeneous group of birds by handcrafting 3 steering behaviours: alignment, cohesion and avoidance. We define these behaviours under micro behaviours, which are behaviours that cannot be further decomposed into simpler behaviours. The aggregated motion, flocking, is defined as a macro behaviour, a behaviour that can only be achieved with an aggregation of several micro behaviours. Our evaluations particularly concentrate on replacing the handcrafting approach adopted by Reynolds with the proposed automated approaches to evolve these 4 behaviours (alignment, cohesion, avoidance, flocking) in a heterogeneous context with minimal human intervention in the rule design process.

A hybrid architecture is utilised in modelling the boids giving them both deliberative (driven by a common goal) and reactive (interact with neighbours and act based on dynamic changes) properties. The perception of boids is based solely on vision as they sense the neighbourhood based on their vision range, which is also evolved as a part of their behaviour rule.

4.1. Problem definition

Given that,
μ - number of boids
υ - number of phantom agents
ζ - number of rules represented by a single genome
κ - population size
The simulation model consists of a set B of boids (b),

$$B = \{b_1, b_2, \dots, b_\mu\} \tag{3}$$

For TL, the learning population P_{TL} consists of k individuals each defining a separate heterogeneous boid set B_i each represented by a single genome G_{TL_i},

$$P_{TL} = \{G_{TL1}, G_{TL2}, \dots, G_{TL\kappa}\} \tag{4}$$

For CCE approaches, the population P_{CE} consists of μ individuals each representing a sub population for a single boid in the boid set B which is defined by a single genome G_{CE_i},

$$P_{CE} = \{G_{CE1}, G_{CE2}, \dots, G_{CE\kappa} : \kappa = \mu\} \tag{5}$$

A genome G_{TL_i} in TL approach represents rules for all agents (a) in a single set of agents,

$$G_{TL_i} = \{a_1, a_2, \dots, a_\zeta : \zeta = \mu, a_i \mathrel{\widehat{=}} b_i\} \tag{6}$$

A sub population of phantom agents (p_a) corresponding to an agent is represented by a genome G_{CE_i} in CCE approaches,

$$G_{CE_i} = \{p_{a_1}, p_{a_2}, \dots, p_{a_\zeta} : \zeta = \upsilon, G_{CE_i} \mathrel{\widehat{=}} b_i\} \tag{7}$$

Experiments are conducted to evolve the 4 behaviours; alignment: steering in the direction of average heading of neighbours; cohesion: moving closer to neighbours and navigating as a group; avoidance: avoiding collisions among agents by maintaining a distance from neighbours; and flocking: their unified motion as observed in a natural flock.

Quantitative measures are used to evaluate the global fitness of the behaviours as illustrated in Algorithm 5. Alignment is quantified using

Algorithm 5 Fitness Functions for the Behavioural Tasks.

```
Require: $B$:List of boids$\mu$:Number of boids
Ensure: $F_{Co}$: Fitness value for cohesion$F_{Av}$: Fitness value for
        avoidance$F_{Al}$: Fitness value for alignment$F_{Fl}$: Fitness value for
        flocking
1: procedure FITNESSCOHESION($B, \mu$)
2:     $F_{Co} \leftarrow 0$
3:     for $b \in B$ do
4:         $F_{Co} \leftarrow \frac{1}{\eta-1} \sum_{j=1}^{\eta} distance(d_b - d_j) : j \neq b$
5:     $F_{Co} \leftarrow$ AVERAGE($F_{Co}, \mu$)
6:     return $F_{Co}$
7: procedure FITNESSAVOIDANCE($B, \mu$)
8:     $F_{Av} \leftarrow 0$
9:     for $b \in B$ do
10:        $d_b \leftarrow \frac{1}{\eta-1} \sum_{j=1}^{\eta} distance(d_b - d_j) : j \neq b$
11:        if $d_b \leq 500$ then
12:            $d_b \leftarrow \psi$
13:        $F_{Av} \leftarrow -1 + \frac{1}{1+\exp^{-\delta(d_b - \gamma\psi)/\psi}}$
14:    $F_{Av} \leftarrow$ AVERAGE($F_{Av}, \mu$)
15:    return $F_{Av}$
16: procedure FITNESSALIGNMENT($B, \mu$)
17:    $F_{Al} \leftarrow -1\| \sum_{i=1}^{\eta} v_b \|$
18:    $F_{Al} \leftarrow$ AVERAGE($F_{Al}, \mu$)
19:    return $F_{Al}$
20: procedure FITNESSFLOCKING($F_{Co}, F_{Av}, F_{Al}$)
21:    $F_{Fl} \leftarrow (F_{Co} + F_{Av} + F_{Al})/3$
22:    return $F_{Fl}$
```

Table 1
Evolutionary Attributes.

Attribute	TL	CCE / Extended CCE
Individual		
Individual Size	100 codons of 8 bits	100 codons of 8 bits
Maximum Wrappings	50	50
Evolutionary Algorithm		
Population Size	30	30(= boids)
Evolutionary Strategy	SSR	Coevolution
Parent Selection	Tournament (size:5)	Tournament (size:5)
Mutation Probability	0.5	1.0
Crossover	Yes	No
Maximum Generations	1000	1000
Evolutionary Runs	15	15
No. of Phantom Agents	N/A	5
No. of Rules in a Genome	30(= boids)	5(= phantom agents)
Simulation Environment		
No. of Boids	30	30
World Size	850 x 850 units	850 x 850 units
World Nature	Wrap Around	Wrap Around
Agent Speed	3 units per tick	3 units per tick

the order measure which calculates the absolute average of the normalised velocities of all boids [45]. The average distance value of the separation between boids is used to measure cohesion. The avoidance measure from Quera et al. [46] applies the average separation distance (d_b) among boids in the inverse logistic function with the standard parameters $\delta > 0$, $0 < \gamma < 1$, $\psi > 0$. The function is modified with parameter values ($\delta = 100$, $\gamma = 0.99$, $\psi = 1000$) determined experimentally to suit our model. A penalise measure which sets d_b to ψ, if it is less than or equal to 500 units is used to encourage boids to avoid collisions. For flocking behaviour, a combination of the above 3 measures with equal weights for each component is utilised as the fitness measure. All 4 fitness evaluators are modified as minimising functions ranging from values 0–1.

For the local fitness measures of a boid associated with the extended CCE approach, functions are slightly modified to calculate the individual contributions. The alignment measure calculates the difference in velocity from a particular boid to the averaged normalised velocity of group (function in lines 17–18 of Algorithm 5); the cohesion measure calculates the separation distance of the particular boid from the team (function in line 4); and the avoidance measure (function in lines 10–13) calculates the avoidance value of the boid from the team. The flocking measure uses a combination of the said 3 measures.

Furthermore, we emphasise that designing of these fitness functions, as well as selection of the atomic components from which the rule structures are evolved still involve human intervention. It is not within the scope of this paper to completely eliminate human bias by addressing all these aspects, but to limit it as much as possible by concentrating our focus only on the rule structure designing process. Extensive research is required in future to address the other two aspects to completely eliminate bias.

4.2. Evolutionary attributes

Each genome consists of 100 codons of 8 bits. To minimise invalid genomes and allow sufficient complexity for the rules, an experimentally determined maximum wrapping value of 50 is introduced. Therefore, after 50 wraps if it is not mapped to an expression of all terminals, codon set is made invalid. The attributes of the evolutionary set up and the simulation environment for all experiments are as given in Table 1. The attributes of the genetic operations were fixed after preliminary experiments[1].

4.3. Comparison models

To further investigate the performance of our model, we conduct two comparison analyses. First we compare our GE model in homogeneous versus heterogeneous agent contexts. Second we utilise a Particle Swarm Optimisation (PSO) model and a behaviour tree-based Genetic Programming (GP) model to compare it against state-of-the-art approaches.

PSO [47] is a population based heuristic search technique adopted in optimisation problems. However, as discussed in Section 3.2, it cannot incorporate rule structures, as its goal is to optimise an n dimensional vector based on an optimisation formulation. Hence, in the context of MASs, PSO is commonly used for optimising parameters and coefficients associated with agent rules. For the work of this paper, we used the standard boid rules [44] proposed by Reynolds described earlier in this section as the steering vectors (which are the most accepted and state-of-the-art handcrafted boid rules used in the domain) and used a weighted combination of them to derive the movement vector as in Eq. (8):

$$\vec{V} = \varpi_1 \vec{co} + \varpi_2 \vec{al} + \varpi_3 \vec{av} \tag{8}$$

The coefficients ϖ_1, ϖ_2 and ϖ_3 determine the influence of each steering behaviour rule and PSO is used to optimise them. We adopt the implementation in [48] for our analysis. We place 30 particles (= population size of GE model) in the search space to evaluate the fitness functions in Algorithm 5. Each particle iteratively determines its movement in space based on its local best position and the global best position of all particles, eventually converging towards a position (a coefficient value in our case) with a better fitness value. The velocity $\vec{v}_{t+1}$ and position $\vec{s}_{t+1}$ of the particle at the next iteration are modelled as given in Eqs. (9) and (10):

$$\vec{v}_{t+1} = \omega \vec{v}_t + C_1 R_1 (\vec{p}_{best} - \vec{s}_t) + C_2 R_2 (\vec{g}_{best} - \vec{s}_t) \tag{9}$$

$$\vec{s}_{t+1} = \vec{s}_t + \vec{v}_{t+1} \tag{10}$$

where $C_1 = 2$, $C_2 = 2$ are fixed learning factors. $\vec{p}_{best}$ and $\vec{g}_{best}$ are the personal best position of a particle and the global best of any particle, respectively. The inertia weight ω is calculated with Eq. (11) using the values $\omega_{start} = 0.4$, $\omega_{end} = 0.9$ and $iter_{total} = 100$ adopted from the implementation of Alaliyat et al. [48]:

$$\omega = \omega_{start} - \frac{\omega_{start} - \omega_{end}}{iter_{total}} iter_t \tag{11}$$

In contrast to PSO, GP is capable of incorporating rule structures within the evolutionary process. To facilitate a fair comparison, the same components used for the rule space of the proposed GE model are used for the function and terminal sets of the GP model. As GP is limited by closure property [49] which requires every function to be defined for any combination of arguments it may encounter, all functions were modified to accept arguments consisting of any combination of functions and terminals from the set of primitives. The GP algorithm is as given in Algorithm 6, where the genetic operations are performed directly on the selected parent rules based on the fitness values determined by fitness functions in Algorithm 5.

5. Results

5.1. Evolutionary results

We first analyse the evolutionary results of the proposed 3 algorithms: TL, CCE and extended CCE.

Figures 8 and 9 depict the evolutionary results through 1000 generations for each of the 4 behaviours. The shaded regions depict the standard deviation of the results across the number of runs. Statistical evaluations are conducted using non-parametric Mann-Whitney U test for comparisons involving 2 methods and Kruskal-Wallis H test for comparisons involving 3 methods, as the sample sizes are small and not normally distributed. The statistical significance level is specified as $p = 0.05$ and the p values obtained for each comparison are presented in Table 2. The descriptive analysis with respect to the 3 algorithms is presented in Table 3 which is evaluated in terms of the variability of

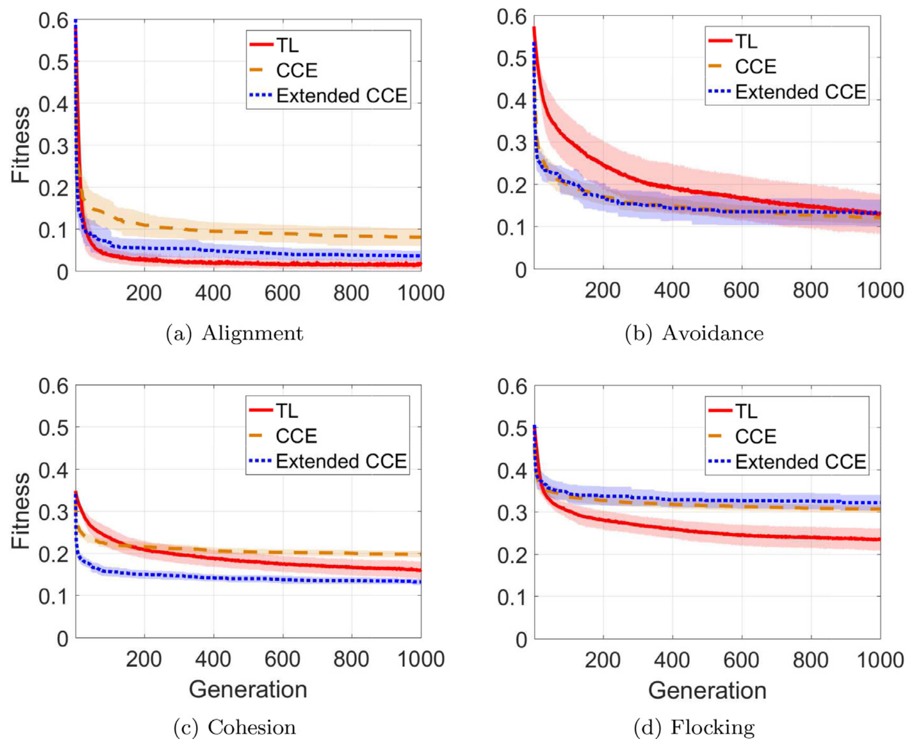

Fig. 8. Average fitness progression of the population through 1000 generations for the 3 micro behaviours alignment, avoidance, cohesion and the macro behaviour, flocking based on the 3 proposed methods. The experimental results are averaged across 15 runs and the shaded areas depict the standard deviation.

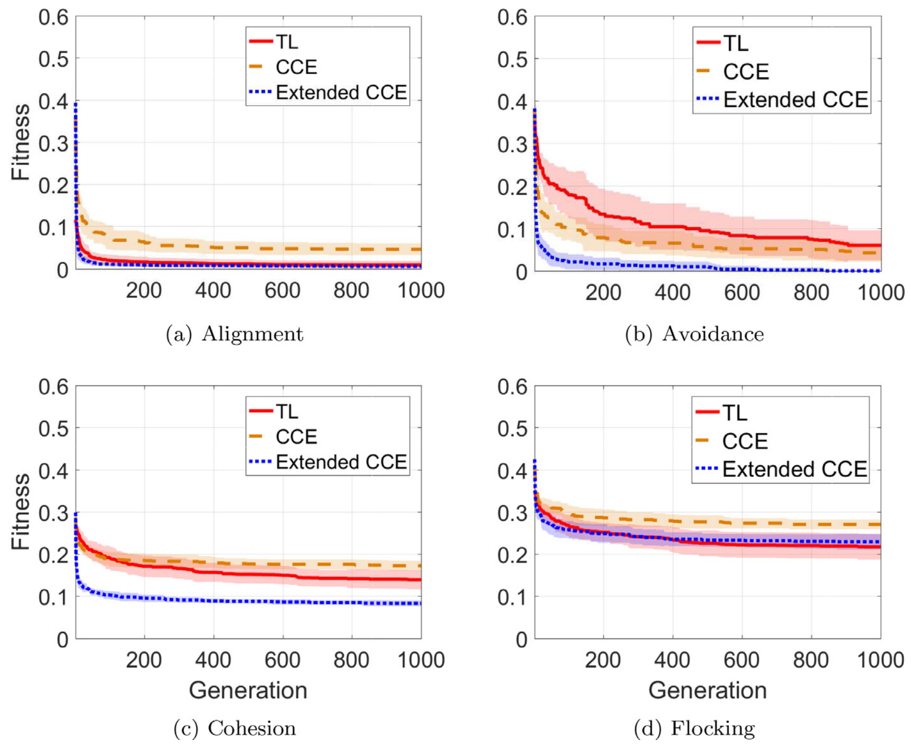

Fig. 9. Fitness progression of the best solution through 1000 generations for the 3 micro behaviours alignment, avoidance, cohesion and the macro behaviour, flocking based on the 3 proposed methods. The experimental results are averaged across 15 runs and the shaded areas depict the standard deviation.

the best solutions achieved after 1000 generations for each of the four beahaviours across 15 runs each.

From the results it is evident that all 3 methods are able to consistently minimise fitness across all 4 behaviours. While the extended CCE approach achieves best performance with avoidance and cohesion, both TL and extended CCE approaches indicate statistically similar performance for alignment and flocking. The statistical results also demonstrate that the extended CCE approach performs better than traditional CCE ($p < 0.001$) in all cases, implying that the modifications applied have significantly improved the performance. In examining the standard deviation of the best solutions for evolution of each behaviour, it is evident that all three approaches are capable of generating consistent solutions as they maintain a low standard deviation (< 0.05) across all four problems tested. The extended CCE approach maintains the lowest standard deviation of the 3 algorithms. CCE approach comes next, however the other statistics proved that the solutions are not near-optimal. Hence,

Algorithm 6 Genetic Programming.

```
Require: $F$: Set of primitive functions
         $T$: Set of terminals
         $\Omega$: Maximum generations
Ensure: $I_b$: Best individual program
 1: procedure GENETICPROGRAMMING($F, T, \Omega$)
 2:     $pop \leftarrow$ INITIALISEPOPULATION($F, T$)
 3:     $\omega \leftarrow 0$
 4:     while $\omega \neq \Omega$ do
 5:         for $pop_i \in pop$ do
 6:             EVALUATEFITNESS($pop_i$)
 7:         $parents \leftarrow$ PARENTSELECTION($pop$)
 8:         $children \leftarrow$ CROSSOVER($pop$)
 9:         $children \leftarrow$ MUTATE($children$)
10:         $pop \leftarrow$ UPDATEWITH($children$)
11:         $I_b \leftarrow$ GETMOSTFITINDIV($pop$)
12:         $\omega + +$
13:     return $I_b$
```

Table 2
Statistical Results Summary for Fitness Measures .

Behaviour	Kruskal-Wallis H-test (*p*-value)	Mann-Whitney *U*-test (*p*-value)
Alignment	6.3150e-07	
TL - CCE		7.4772e-06
TL - Extended CCE		0.5338
CCE - Extended CCE		**3.3918e-06**
Avoidance	4.6659e-07	
TL - CCE		0.3013
TL - Extended CCE		2.9582e-06
CCE - Extended CCE		**7.2818e-06**
Cohesion	3.4798e-08	
TL - CCE		5.7598e-04
TL - Extended CCE		5.0527e-06
CCE - Extended CCE		**3.3918e-06**
Flocking	3.7741e-06	
TL - CCE		2.7983e-05
TL - Extended CCE		0.3401
CCE - Extended CCE		**1.6053e-05**

it can only be concluded that CCE generates consistent solutions which are not satisfactory. TL has the lowest robustness in comparison to the other two, but still with a low standard deviation of < 0.036 for all problems giving both satisfactory and consistent results. In conclusion, both the TL and the extended CCE approaches exhibit a robust performance within and across problems with minimal variability among solutions and satisfactory fitness values.

To further compare the 3 approaches, we analyse the percentage improvement of the best solution across 1000 generations for all 4 behaviours, and the results are depicted in Fig. 10. The extended CCE approach improves the solution space faster than the other 2 methods in all 4 behaviours and finds the best solution earlier than them during the evolution process. The traditional CCE approach has the poorest performance in this aspect as well, with a slow progression of the best solution.

Overall, the results suggest that the traditional CCE method is prone to premature convergence while the modifications introduced with the extended CCE method successfully overcome this limitation. TL and extended CCE methods perform better than the traditional CCE approach, however comparison between the 2 methods suggest that extended CCE finds better solutions faster than TL. Hence, the extended CCE approach is evidently more successful considering both performance and degree of improvement.

5.2. Comparison against other models

Figure 11 compares the fitness distribution of the best solutions obtained with the heterogeneous models against those of the homogeneous model.

The homogeneous model surpasses the heterogeneous methods generating better near-optimal solutions for all 3 micro behaviours alignment, avoidance and cohesion. The traditional CCE model has the worst performance since it is prone to converge at sub-optimal solutions. Extended CCE with heterogeneous agents follows the homogeneous model closely in overall with the second best performance. This result is expected, as evolving multiple rules separately for individual agents such that all agents collaborate in achieving one task is presumably harder than evolving a single rule applicable for every agent in a homogeneous system. Nevertheless, with the macro behaviour flocking, the performance of heterogeneous methods using TL (p = 0.0037 <0.05) as well as extended CCE (p = 0.0128 <0.05) significantly exceed that of the homogeneous model. This observation is significant since it provides evidence to support future exploration of grammar-based heterogeneous agent systems as a potential alternative in designing solutions for more complex tasks for which human intuition could be deficient.

To further investigate the performance of the proposed GE-based model in comparison to state-of-the art model that is frequently used in evolving rule structures, Fig. 12 illustrates the comparison results of GE against PSO and GP in a homogeneous context. We compared the results of the 3 methods in a homogeneous context after 100 generations for all 4 behaviours.

The proposed GE model has a statistically significant improvement in fitness (p <0.05) across all 4 behaviours compared to the GP model. Further, the performance of the GE model exceeds that of the PSO model with alignment, avoidance and flocking (p <0.05) and is comparable to that of the PSO model with cohesion (p = 0.4716 >0.05). This shows that in comparison, our proposed model is capable of surpassing both GP and PSO approaches with the tested behaviours. As PSO can only be used to optimise the parameters rather than the rule structures, the difficulty of manually designing appropriate behaviour rule structures

Table 3
Descriptive Analsyis of the Algorithms.

Algorithm	Behaviour	Minimum	Maximum	Mean	Std Dev
TL	Alignment	0.0004	0.0338	0.0093	0.0095
	Avoidance	0.0244	0.1218	0.0601	0.0354
	Cohesion	0.0890	0.1793	0.1390	0.0224
	Flocking	0.1697	0.2648	0.2168	0.0302
CCE	Alignment	0.0234	0.0763	0.0461	0.0139
	Avoidance	6.6613E-16	0.0731	0.0422	0.0208
	Cohesion	0.1591	0.2054	0.1722	0.0120
	Flocking	0.2518	0.2902	0.2702	0.0115
Extended CCE	Alignment	0.0026	0.0108	0.0054	0.0021
	Avoidance	0.0000	4.774E-15	7.1794E-16	1.2677E-15
	Cohesion	0.0665	0.0970	0.0829	0.0067
	Flocking	0.1934	0.2611	0.2291	0.0185

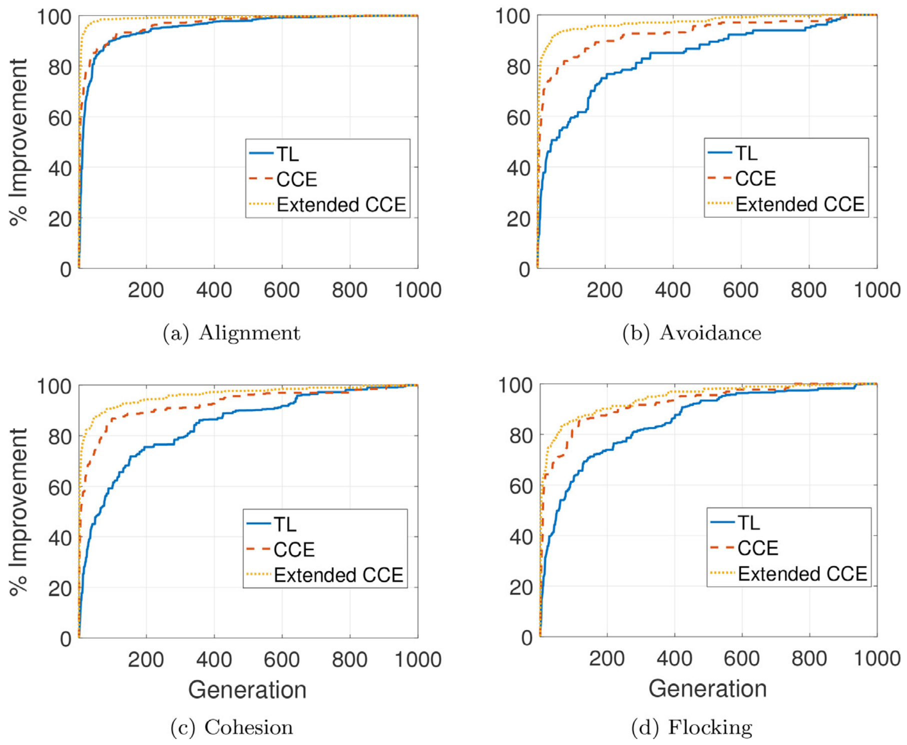

Fig. 10. Improvement of fitness as a percentage over 1000 generations using the 3 mechanisms for the 4 behaviours alignment, avoidance, cohesion and flocking. Experiments were repeated for 15 runs for each behaviour.

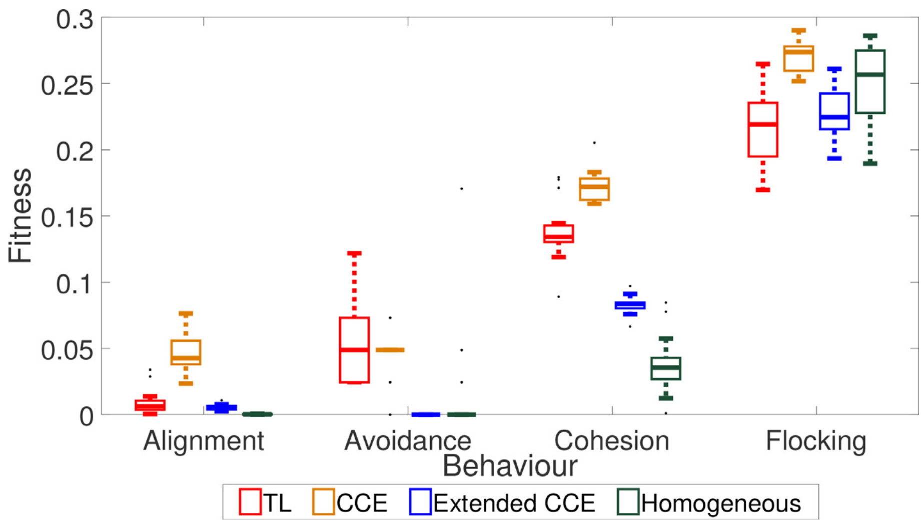

Fig. 11. Comparison of performance of the best solutions averaged across 15 runs of each method against the best solutions of the homogeneous model presented in [11]. The bottom edge, central line and top edge indicate the 25th percentile, median result, and the 75th percentile, respectively. The extensions of whiskers run up to the extreme results which are not deemed outliers. The same evolutionary attributes including crossover and mutation rates, the number of generations, and number of wraps used in the heterogeneous TL model were used for the homogeneous model used for comparison.

to address a problem increases with the complexity of the problem. It limits the potential of the agents to the depth and breadth of intuition of the designer. On the other hand, although GP can evolve rule structures, it cannot validate the syntax of the evolved rule structures leading to more invalid rules. Our model, with the use of a grammatical syntax and a CFG, is evidently more successful than these methods reducing human intervention in rule design process and giving more flexibility to the algorithm to evolve syntactically valid high performing agent rules.

5.3. Computational complexity

Here we conduct an empirical analysis on the computational complexity across space and time, of the 3 algorithms for increasing agent group sizes. Figure 13 illustrates the execution time and memory usage for the 3 algorithms for groups of 10, 20, 30 and 50 agents evolved for flocking behaviour.

The results show that the TL method consumes less time compared to the other 2 methods and the complexity does not scale as the size of the agent group increases. Both CCE methods have significantly longer execution times and scale exponentially as the number of agents increase. On the other hand, the memory consumption of TL method is relatively higher than the two CCE approaches. However, the exponential increase in execution times of the CCE approaches outweigh the relatively less significant requirement of memory by TL as its execution time remains the same for all group sizes. As such, it can be concluded that the TL approach is the better alternative in terms of computational complexity out of the 3 approaches.

5.4. Performance scalability with number of agents

To analyse the scalability of the 3 methods with different agent group sizes, we compare the performance for flocking behaviour across groups of 10, 20, 30 and 50 agents. Fig. 14 illustrates the fitness distribution of the best individuals across these agent groups. The results suggest that

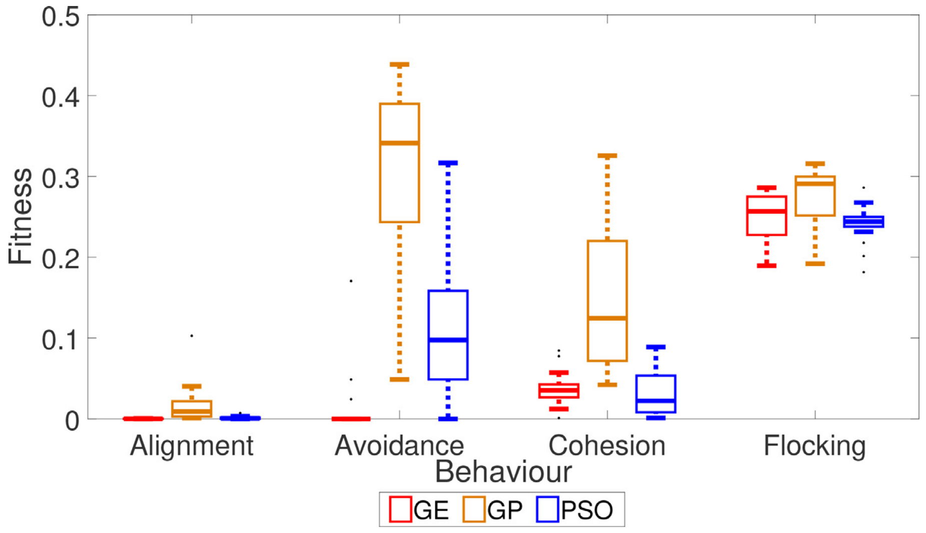

Fig. 12. Comparison of performance of the best solutions averaged across 15 runs of GE, GP and PSO in a homogeneous context. The bottom edge, central line and top edge indicate the 25th percentile, median result, and the 75th percentile, respectively. The extensions of whiskers run up to the extreme results which are not deemed outliers. Crossover and mutation rates used in the GP model are the same as those used with GE and the population size and number of generations are the same across GE, GP and PSO.

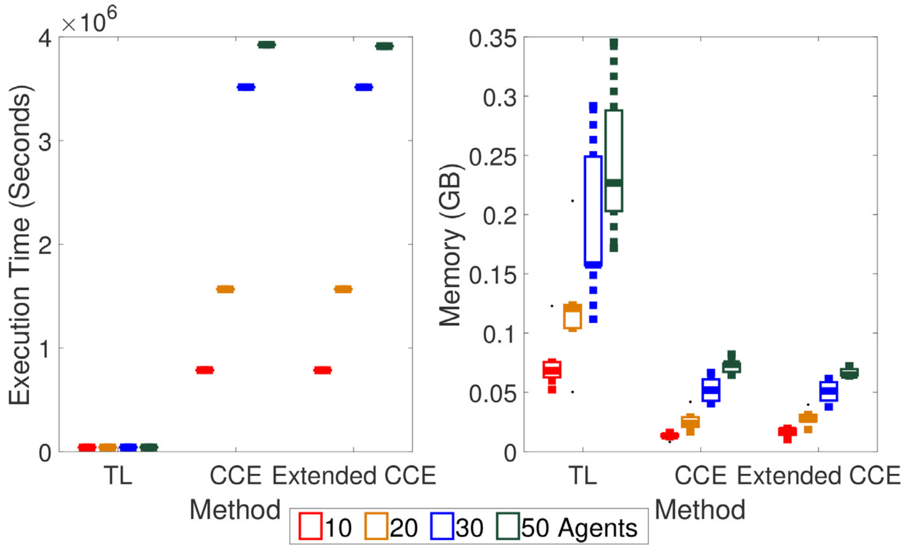

Fig. 13. Comparison of execution times and memory usage of the 3 methods across 5 runs for groups of 10, 20, 30 and 50 agents for flocking over 1000 generations. The bottom edge, central line and top edge indicate the 25th percentile, median result, and the 75th percentile, respectively. The extensions of whiskers run up to the extreme results which are not deemed outliers.

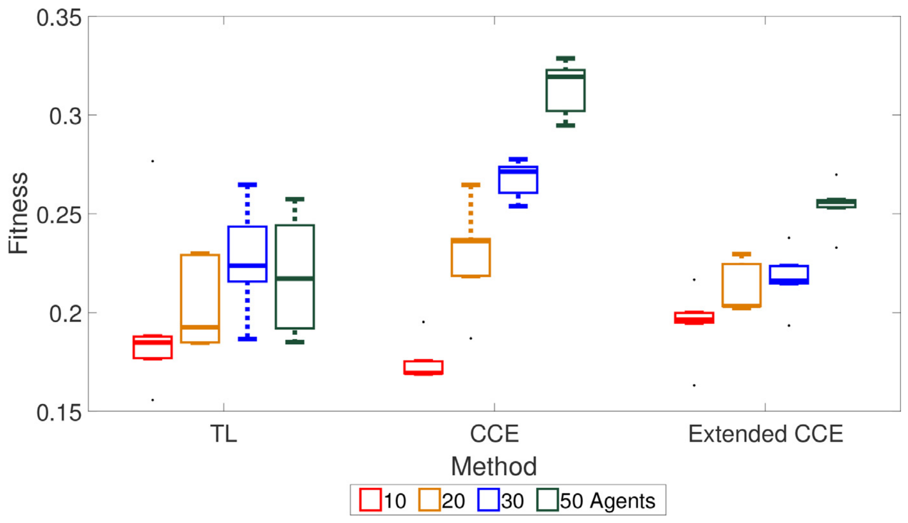

Fig. 14. Comparison of performance of the best individuals across 5 runs for flocking behaviour of each method for groups of 10, 20, 30 and 50 agents. The bottom edge, central line and top edge indicate the 25th percentile, median result, and the 75th percentile, respectively. The extensions of whiskers run up to the extreme results which are not deemed outliers.

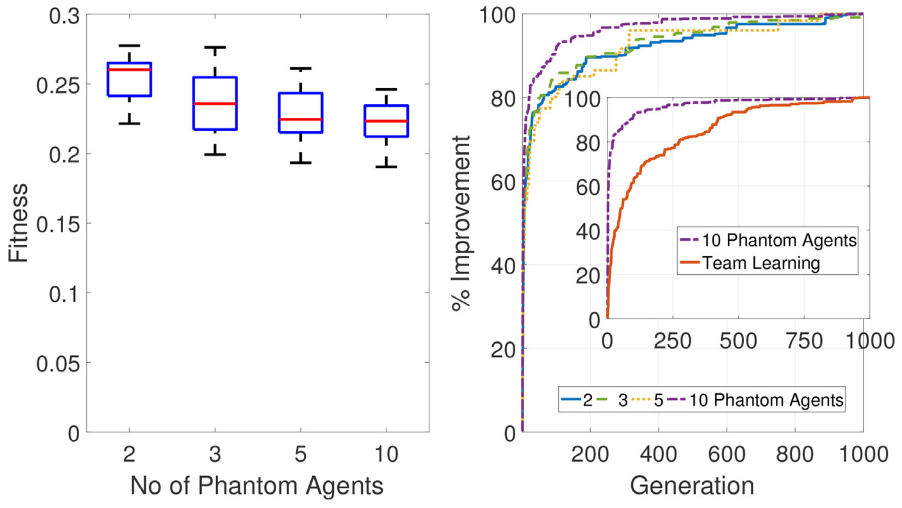

Fig. 15. Scalability with the number of PAs illustrated using flocking behaviour. Left: Comparison of fitness of the best solutions averaged across 15 experiments for varying number of PAs. The bottom edge, central line and top edge indicate the 25th percentile, median result, and the 75th percentile, respectively. The extensions of whiskers run up to the extreme results which are not deemed outliers. Right: Improvement of fitness as a percentage over 1000 generations averaged across 15 runs for varying number of PAs. The embedded plot compares the fitness improvement for flocking behaviour of TL approach against that of extended CCE approach with 10 PAs.

the 3 methods are consistent in their performance across agent groups of increasing size. All 3 methods perform equally well with a group of 10 agents, but as the group size increases the TL and extended CCE approach surpass the performance of the traditional CCE approach. The TL and extended CCE methods show a statistically similar performance (p >0.05).

5.5. Performance scalability with PAs: Extended CCE

To further explore the extended CCE approach, we assess the scalability of the approach based on the number of PAs. The evaluations are conducted considering the flocking behaviour. While keeping the rest of the parameters and attributes the same, we vary the number of PAs used for each sub-population of the evolutionary process.

Figure 15 illustrates a comparison of the fitness distribution of the best solutions for every run using different numbers of PAs. The statistical results (Pearson's r = -0.7100, p = 0.2900 for the number of PAs versus average of the best fitness values) suggest diminishing returns, as the fitness improvement with respect to the number of PAs becomes increasingly less significant. The comparison of the percentage fitness improvement across 1000 generations suggests that a higher number of PAs can learn the solution faster than with a lesser number. The results illustrate that the improvement difference is not significant among 2,3 and 5 phantom agents, but with 10, the evolutionary process consistently improves faster at every stage of the 1000 generations. Furthermore, all 4 versions of PAs outperform the TL approach in terms of percentage fitness improvement significantly (p <0.001).

In conclusion, it can be said that performance is proportional to the number of PAs, however it is less evident as the number increases. Still, better results can be expected within a fewer number of generations as the number of PAs increase. Increasing the number of PAs means an increase in the computational cost due to the addition of more branches in the solution space to explore. Therefore, identifying a balance between expected performance and expendable resources is required.

5.6. Rule complexity

Figure 16 demonstrates the complexity measured in terms of cyclomatic complexity [50] which is a common quantitative metric used to determine the number of independent paths through a rule or programme, and the number of individual rules in each aggregated rule for all 4 behaviours.

The rules evolved in the heterogeneous context with all 3 methods are more complex than those evolved in the homogeneous context, despite solving the same tasks and being treated with identical evolutionary attributes. Rules evolved in the homogeneous environment maintain an average cyclomatic complexity less than 100, and the number of rules in each aggregation average to less than 2. All 3 methods in the heterogeneous environment have higher cyclomatic complexities around 100–200 and number of rules within 2–6 in general. Interestingly, although extended CCE overcomes premature convergence issues associated with traditional CCE with better solutions, it does not reflect a proportional increase in complexity of the evolved rules.

Figure 17 compares the variation of cyclomatic complexity in relation to the fitness for all evolutionary runs over 1000 generations. Each graph should be interpreted from right to left starting from the first generations with higher fitness values (as we experiment on a minimising function) represented in blue, moving left along the x axis to lower fitness values in the later generations shifting the colour to yellow in the generations close to 1000. All 15 runs are plotted in each graph. The comparison results for number of rules in relation to fitness also show a similar pattern as shown in Fig. 18.

There is not enough evidence to identify a correlation between complexity and fitness of rules. However, a few interesting observations provide insights on the strategies used by the methods to explore the search space. The horizontal line formations observed during the last generations of the homogeneous model (Fig. 17d) reveal that it explores a broader range of rules with diverse complexity levels at the beginning of the evolution process, and as it converges to a solution, the population starts exploring only slight variations from the best solutions. TL (Fig. 17a) is quite constraint in its strategy and limits its explorations to a closer neighbourhood of complexity. The horizontal paths observed from the beginning to end of each run provide evidence for this. The CCE methods (Fig. 17b, c), are more relaxed in their strategy and explore a wide range of solutions. The vertical line formations during the last generations show that although the REPs converge to a solution and maintain a steady fitness (hence the x values remain the same forming vertical lines), the PAs still keep searching over a broader spectrum (y values move across a wider range) of solution with diverse complexities.

Similarly, a correlation cannot be observed between the number of rules and the fitness as shown in Fig. 18. However, the homogeneous model explores a broader range of rules at the beginning of the evolution process, and as it converges to a solution, settles for aggregated rules with around 2 individual rules in general, but go up to 8 rules in some cases. TL, similar to cyclomatic complexity variations, remain restricted in its strategy and from the beginning of the evolutionary process, limits its explorations to a closer neighbourhood. In both the CCE methods, the phantom agents explore varying number of rules from 2 - 10 throughout the evolutionary process even after the representatives have converged to a solution maintaining a steady fitness.

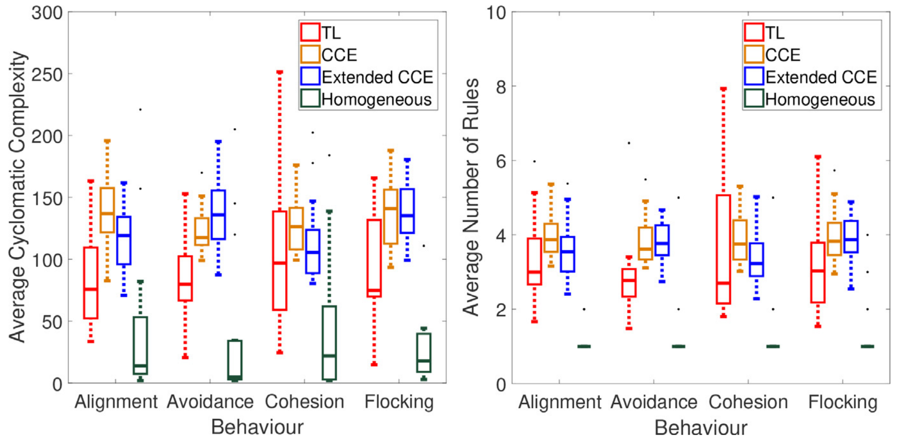

Fig. 16. Complexity analysis of the evolved rules with the 3 heterogeneous models and the homogeneous model across 15 runs for the 4 behaviours alignment (AL), avoidance (AV), cohesion (CO) and flocking (FL). Right: The average cyclomatic complexity of the rules averaged across all agent behaviours for 30 agents. Left: The average number of individual rules in each aggregated set of rules averaged across all agent behaviours for 30 agents. The bottom edge, central line and top edge indicate the 25th percentile, median result, and the 75th percentile, respectively. The extensions of whiskers run up to the extreme results which are not deemed outliers.

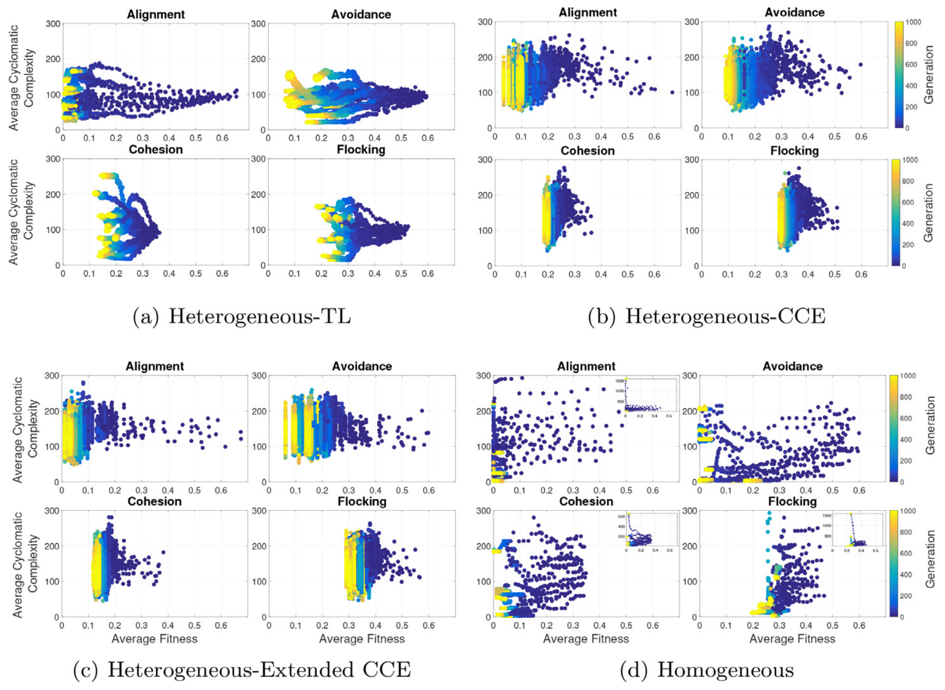

Fig. 17. Variation of cyclomatic complexity of rule structures in relation to fitness for 15 evolutionary runs with the TL, CCE, extended CCE methods against the homogeneous model over 1000 generations. The values are averaged across all agents of a group of 30 for each generation. The individual values are colour coded based on the generation.

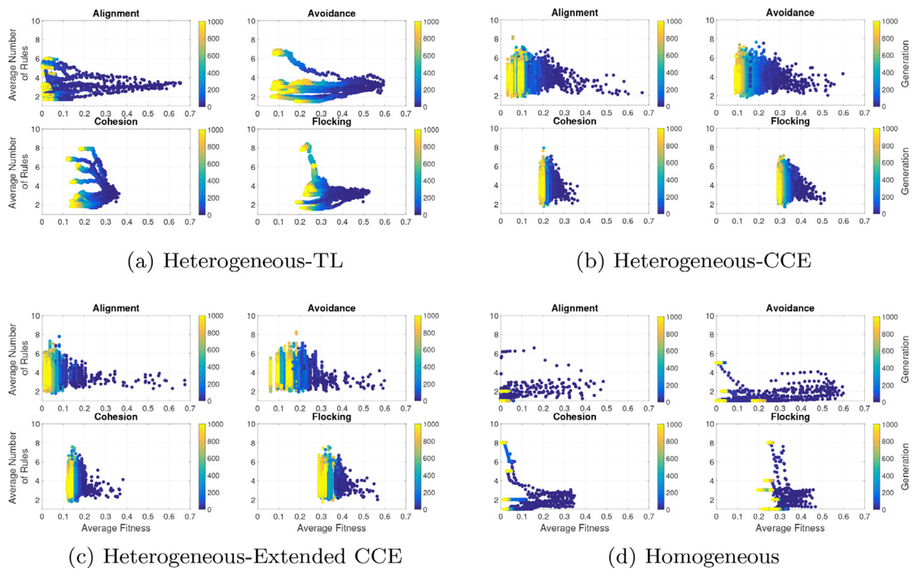

Fig. 18. Variation of number of rules in relation to the fitness of the rule structures for 15 evolutionary runs with the TL, CCE, extended CCE methods against the homogeneous model over 1000 generations. The values are averaged across all agents of a group of 30 for each generation. The individual values are colour coded based on the generation.

6. Discussion

This paper described a novel grammar-based cooperative learning approach to evolve heterogeneous multi-agent behaviours. The method enables autonomous emergence of heterogeneity within the agent system with minimised human intervention. A CFG ensures the generation of valid rules based on a syntax while enabling heterogeneity.

GE has not been a popular choice in conjunction with coevolution in heterogeneous contexts due to the limitations associated with representation of individuals. The encoding mechanism proposed here for the genomes overcomes this issue as it provides a way to represent multiple individuals in a single genome. It can be conveniently translated into sub-populations of individuals being separately evolved in a CCE approach as well. The proposed extended CCE approach demonstrated to be the most successful, considering the performance and fitness improvement, in evolving heterogeneous multi-agent behaviours. Further, we presented a TL approach to represent individuals and evolve heterogeneous behaviours with successful results in contrast to the existing approaches where TL has mostly been studied in homogeneous contexts [51,52].

Based on the experimental evaluations, the following deductions can be made with regard to the presented approaches;

1. Grammar-based evolution with both homogeneous and heterogeneous agent systems work successfully in autonomous generation of complex behavioural rules without the need of manual hand crafting of rules.
2. Our TL mechanism is successful in evolving complex heterogeneous behaviours. However, it performs relatively poorly with micro behaviours compared to a homogeneous model. Nevertheless, TL is a computationally less expensive mechanism as the execution time does not grow with the number of agents in the system.
3. Our extended CCE mechanism can overcome the limitations of the traditional CCE method and demonstrates the best performance out of the 3 mechanisms for heterogeneous MASs and performs nearly optimally as a homogeneous model for micro behaviours. However, it is computationally expensive in comparison to TL. The cost increases proportional to the number of agents.
4. The comparison of homogeneous and heterogeneous agent systems results in further evidence to support the claims in literature describing the limitations of homogeneous agent systems with regard to generation of complex behaviours [53,54], from a GE view point. Our results demonstrate that, while the homogeneous agent system is more successful in micro-behaviours which cannot be further decomposed into simpler rules, the heterogeneous approach is more advantageous with macro-behaviours as the complexity of the requirements increase. Both the TL and extended CCE approaches significantly outperform homogeneous results with macro behaviours.
5. Proposed GE model shows potential in overcoming the limitations of PSO and GP mechanics which are widely adopted in MAS designing by surpassing them in performance as tested within a homogenous context.
6. The rule complexity analysis show that the rule structures evolved within a heterogeneous context are more complex than those within a homogeneous context addressing the same tasks. Furthermore, it shows that the TL method is more restrictive in its strategy of search space exploration while the CCE methods explore a wider spectrum of solutions during the evolutionary process.

The presented evaluations explored the widely tested domain of boids and their commonly tested collective behaviours: alignment, avoidance, cohesion, and flocking. The promising results further support the applicability of the proposed GE-based cooperative evolutionary architectures in real world domains. There exists a diverse array of real-world requirements for MASs where heterogeneity can be useful in adopting to dynamic conditions. For example, these results can be easily translated into the behaviour rules of robot/drone systems working on surveillance and exploration. It is to be expected that an agent system performing a surveillance task may face issues such as certain agents losing sensing abilities due to hardware failures, unforeseen damages to agents causing to lose team members, and dynamic environmental changes that require certain agents to act flexibly and adopt to the environments. The heterogeneous context within which the agents were tested in our simulations ensure such difficulties can be overcome by the agent team due to their ability to adjust the behaviours dynamically to suit the conditions of the entire team. Robot teams that are required to navigate in terrains with obstacles, for example, worker robots that have to cooperatively stock shelves in large warehouses by navigating through the space avoiding collisions with each other and shelves, is another example where these architectures could be used. The navigation paths of the agents could be evolved such that each agent is optimised to identify the best path to reach the shelves in a short time while avoiding collisions as the environment gets updated based on the positions of other agents. Given the recent advancements in the graphics and visualisations related to games and other animations which demand high fidelity visualisations and simulations, these architectures can also be used in designing simulations that closely represent the behaviour of real-world agent groups. As the agents can be evolved to cooperatively act in the environment with appropriate heterogeneous behaviour rules, the fidelity of the simulations can be increased. Further, as the grammar and the syntax can be modified to suit any real-world domain and include the respective actions, the proposed architectures can be directly extended towards any real-world application domain that can benefit from heterogeneous multi-agent interactions given that a fitness criterion can be defined to evaluate the behaviour rules being evolved.

7. Conclusion and future work

Our GE mechanism is a promising new approach for MASs adopted in real world domains with complex task requirements where human intuition becomes insufficient in determining the separate individual behaviours that can result in a desired emergent behaviour. The proposed approach models the behavioural system of all agents by generating individual rules from scratch while using heterogeneity to ensure every agent is optimised to achieve a collective task. The significance of the approach lies in its ability to autonomously emerge heterogeneity within the system targeting a common goal.

We identify 4 major directions that can be pursued in future work. As TL can overcome limitations associated with cost in heterogeneous MASs, further analysis is required to understand how the performance could be improved, enabling generation of high fidelity behaviours. Next, the extended CCE approach can be explored to minimise the associated computational cost. As the complexity is associated with the number of agents and PAs in the system, these parameters should be further studied to identify a balance between cost and performance. Third, the proposed model can be applied in multifaceted domains where the behaviours require completing several sub-tasks to achieve the final goal. For example, consider an agent system performing a surveillance task where the agents have to engage in multiple actions such as avoiding threats, coordinating navigation and locating a target. The current results with the grammar-based evolutionary model demonstrates clear potential for achieving behaviours corresponding to such complex requirements which need further evaluations. Finally, although the proposed model is markedly less biased compared to the approaches that do not support rule structure evolution, there does currently exist bias in selecting the atomic components of the rule space and construction of fitness functions. Further investigations are required to completely eliminate bias. For example, adding another evolutionary layer to the GE model to automatically generate the fitness functions from another set of atomic components is an alternative to manual design of fitness functions.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Dilini Samarasinghe: Conceptualization, Formal analysis, Investigation, Methodology, Validation, Writing – original draft, Writing – review & editing. **Michael Barlow:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Erandi Lakshika:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Kathryn Kasmarik:** Conceptualization, Methodology, Supervision, Writing – review & editing.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.swevo.2021.101017.

References

[1] L.E. Parker, Multiple mobile robot systems, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer Berlin Heidelberg, 2008, pp. 921–941.
[2] K. Tumer, A.K. Agogino, D.H. Wolpert, Learning sequences of actions in collectives of autonomous agents, in: Proc. 1st Int. Joint Conf. Autonomous Agents and Multiagent Systems: Part 1, ACM, 2002, pp. 378–385.
[3] P. Husbands, I. Harvey, D. Cliff, G. Miller, Artificial evolution: a new path for artificial intelligence? Brain Cogn. 34 (1) (1997) 130–159.
[4] C.R. Ward, F. Gobet, G. Kendall, Evolving collective behavior in an artificial ecology, Artif. Life 7 (2) (2001) 191–209.
[5] E. Lakshika, M. Barlow, A. Easton, Co-evolving semi-competitive interactions of sheepdog herding behaviors utilizing a simple rule-based multi agent framework, in: Proc. 2013 IEEE Symp. Artificial Life (ALife), 2013, pp. 82–89, doi:10.1109/ALIFE.2013.6602435.
[6] Y.-W. Chen, K. Kobayashi, H. Kawabayashi, X. Huang, Application of interactive genetic algorithms to boid model based artificial fish schools, in: I. Lovrek, R.J. Howlett, L.C. Jain (Eds.), Int. Conf. Knowledge-based and Intelligent Information and Engineering Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 141–148.
[7] N. Le, A. Brabazon, M. O'Neill, The evolution of self-taught neural networks in a multi-agent environment, in: P. Kaufmann, P.A. Castillo (Eds.), Applications of Evolutionary Computation, Springer International Publishing, Cham, 2019, pp. 457–472.
[8] E. Ferrante, E. Duéñez Guzmán, A.E. Turgut, T. Wenseleers, GESwarm: grammatical evolution for the automatic synthesis of collective behaviors in swarm robotics, in: Proc. 15th Annu. Conf. Genetic and Evolutionary Computation, in: GECCO '13, ACM, New York, NY, USA, 2013, pp. 17–24, doi:10.1145/2463372.2463385.
[9] C. Ryan, J. Collins, M.O. Neill, Grammatical evolution: evolving programs for an arbitrary language, in: W. Banzhaf, R. Poli, M. Schoenauer, T.C. Fogarty (Eds.), Genetic Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 83–96.
[10] L. Panait, S. Luke, Cooperative multi-agent learning: the state of the art, Auton. Agent Multi. Agent Syst. 11 (3) (2005) 387–434, doi:10.1007/s10458-005-2631-2.
[11] D. Samarasinghe, E. Lakshika, M. Barlow, K. Kasmarik, Automatic synthesis of swarm behavioural rules from their atomic components, in: Proc. Genetic and Evolutionary Computation Conf, in: GECCO '18, ACM, New York, NY, USA, 2018, pp. 133–140, doi:10.1145/3205455.3205546.
[12] D. Samarasinghe, M. Barlow, E. Lakshika, K. Kasmarik, Exploiting abstractions for grammar-based learning of complex multi-agent behaviours, Int. J. Intell. Syst. (2021) 1–39, doi:10.1002/int.22550.
[13] A. Byrski, R. Dreżewski, L. Siwik, M. Kisiel-Dorohinicki, Evolutionary multi-agent systems, Knowl. Eng. Rev. 30 (2) (2015) 171–186.
[14] D. Karaboga, B. Akay, A survey: algorithms simulating bee swarm intelligence, Artif. Intell. Rev. 31 (1–4) (2009) 61–85.
[15] J.-M. Montanier, S. Carrignon, N. Bredeche, Behavioral specialization in embodied evolutionary robotics: why so difficult? Front. Rob. AI 3 (2016) 38, doi:10.3389/frobt.2016.00038.
[16] M.D, Swarmanoid: a novel concept for the study of heterogeneous robotic swarms, IEEE Rob. Autom. Mag. 20 (4) (2013) 60–71, doi:10.1109/MRA.2013.2252996.
[17] M.A.e.a. Hsieh, Adaptive teams of autonomous aerial and ground robots for situational awareness, J. Field Rob. 24 (11–12) (2007) 991–1014.
[18] A.M. Bretas, A. Mendes, M. Jackson, R. Clement, C. Sanhueza, S. Chalup, A decentralised multi-agent system for rail freight traffic management, Ann. Oper. Res. (2021) 1–31.
[19] R. Sheh, S. Schwertfeger, A. Visser, 16 Years of robocup rescue, Künstliche Intelligenz 30 (3) (2016) 267–277, doi:10.1007/s13218-016-0444-x.
[20] D. Sutantyo, P. Levi, C. Möslinger, M. Read, Collective-adaptive Lévy flight for underwater multi-robot exploration, in: 2013 IEEE Int. Conf. Mechatronics and Automation, IEEE, 2013, pp. 456–462.
[21] J. Werfel, K. Petersen, R. Nagpal, Designing collective behavior in a termite-inspired robot construction team, Science 343 (6172) (2014) 754–758.
[22] F. Yang, Y. Qiao, S. Wang, C. Huang, X. Wang, Blockchain and multi-agent system for meme discovery and prediction in social network, Knowl. Based Syst. 229 (2021) 107368, doi:10.1016/j.knosys.2021.107368.
[23] D.M. Vieira, C. Fernandes, C. Lucena, S. Lifschitz, Driftage: a multi-agent system framework for concept drift detection, Gigascience 10 (6) (2021), doi:10.1093/gigascience/giab030. Giab030
[24] J. Kruse, A. Connor, S. Marks, An interactive multi-agent system for game design, Comput. Games J. 10 (1) (2021) 41–63.
[25] P. Lu, H. Yang, H. Li, M. Li, Z. Zhang, Swarm intelligence, social force and multi-agent modeling of heroic altruism behaviors under collective risks, Knowl. Based Syst. 214 (2021) 106725, doi:10.1016/j.knosys.2020.106725.
[26] J.C. Bongard, The legion system: a novel approach to evolving heterogeneity for collective problem solving, in: European Conf. on Genetic Programming, Springer, 2000, pp. 16–28.
[27] J. Gomes, P. Mariano, A.L. Christensen, Dynamic team heterogeneity in cooperative coevolutionary algorithms, IEEE Trans. Evol. Comput. 22 (6) (2018) 934–948, doi:10.1109/TEVC.2017.2779840.
[28] B.H. Abed-Alguni, S.K. Chalup, F.A. Henskens, D.J. Paul, A multi-agent cooperative reinforcement learning model using a hierarchy of consultants, tutors and workers, Vietnam J. Comput. Sci. 2 (4) (2015) 213–226.
[29] W. Deng, J. Xu, Y. Song, H. Zhao, An effective improved co-evolution ant colony optimisation algorithm with multi-strategies and its application, Int. J. Bio-Inspired Comput. 16 (3) (2020) 158–170.
[30] B. Niu, J. Liu, L. Tan, Multi-swarm cooperative multi-objective bacterial foraging optimisation, Int. J. Bio-Inspired Comput. 13 (1) (2019) 21–31.
[31] C. Hinrichs, M. Sonnenschein, A distributed combinatorial optimisation heuristic for the scheduling of energy resources represented by self-interested agents, Int. J. Bio-Inspired Comput. 10 (2) (2017) 69–78.
[32] T. Yan, X. Xu, Z. Li, E. Li, Flocking of multi-agent systems with unknown nonlinear dynamics and heterogeneous virtual leader, Int. J. Control Autom. Syst. (2021) 1–9.
[33] D. Chen, C. Zhao, Particle swarm optimization with adaptive population size and its application, Appl. Soft Comput. 9 (1) (2009) 39–48.
[34] L. Panait, Theoretical convergence guarantees for cooperative coevolutionary algorithms, Evol. Comput. 18 (4) (2010) 581–615.
[35] J. Gomes, P. Mariano, A.L. Christensen, Avoiding convergence in cooperative coevolution with novelty search, in: Proc. 2014 Int. Conf. Autonomous Agents and Multi-agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 1149–1156.
[36] E. Conti, V. Madhavan, F.P. Such, J. Lehman, K. Stanley, J. Clune, Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents, in: Advances in Neural Information Processing Systems, 2018, pp. 5027–5038.
[37] J.R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, vol. 1, MIT press, 1992.
[38] D. Perez-Liebana, M. Nicolau, Evolving behaviour tree structures using grammatical evolution, in: Handbook of Grammatical Evolution, Springer International Publishing, 2018, pp. 433–460.
[39] J.E. Murphy, M. O'Neill, H. Carr, Exploring grammatical evolution for horse gait optimisation, in: Eur. Conf. Genetic Programming, Springer, 2009, pp. 183–194.
[40] P. Eilert, Learning behaviour trees for simulated fighter pilots in airborne reconnaissance missions: a grammatical evolution approach, Linkping University, Artificial Intelligence and Integrated Computer Systems, 2019 Master's thesis.
[41] G.S. Nitschke, M.C. Schut, A. Eiben, Evolving behavioral specialization in robot teams to solve a collective construction task, Swarm Evol. Comput. 2 (2012) 25–38.
[42] M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. 5 (4) (2001) 349–358, doi:10.1109/4235.942529.
[43] M. Ehrgott, Multicriteria Optimization, vol. 491, Springer Science & Business Media, 2005.
[44] C.W. Reynolds, Flocks, herds and schools: a distributed behavioral model, in: Proc. 14th Annu. Conf. Computer Graphics and Interactive Techniques, in: SIGGRAPH '87, ACM, New York, NY, USA, 1987, pp. 25–34, doi:10.1145/37401.37406.
[45] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, O. Shochet, Novel type of phase transition in a system of self-driven particles, Phys. Rev. Lett. 75 (1995) 1226–1229, doi:10.1103/PhysRevLett.75.1226.
[46] V.Q.J. Quera, F. Salvador Beltrán, R. Dolado i Guivernau, Flocking behaviour: agent-based simulation and hierarchical leadership, J. Artif. Soc. Social Simul. 13 (2) (2010).
[47] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, 1995, pp. 1942–1948 vol.4.
[48] S. Alaliyat, H. Yndestad, F. Sanfilippo, Optimisation of boids swarm model based on genetic algorithm and particle swarm optimisation algorithm (comparative study), in: 28th Eur. Conf. Modelling and Simulation, 2014, pp. 643–650.
[49] L. Vanneschi, R. Poli, Genetic programming — introduction, applications, theory and open issues, in: G. Rozenberg, T. Bäck, J.N. Kok (Eds.), Handbook of Natural Computing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 709–739, doi:10.1007/978-3-540-92910-9_24.
[50] E. Lakshika, M. Barlow, A. Easton, Understanding the interplay of model complexity and fidelity in multiagent systems via an evolutionary framework, IEEE Trans. Comput. Intell. AI Games 9 (3) (2017) 277–289.

[51] Y. Suzuki, T. Arita, A comprehensive evaluation of the methods for evolving a cooperative team, Artif. Life Rob. 10 (2) (2006) 157–161, doi:10.1007/s10015-005-0354-8.

[52] M. Quinn, L. Smith, G. Mayley, P. Husbands, Evolving formation movement for a homogeneous multi-robot system: teamwork and role-allocation with real robots, Congit. Sci. Res. Pap. (2002).

[53] E. Tuci, Evolutionary swarm robotics: genetic diversity, task-allocation and task-switching, in: M. Dorigo, M. Birattari, S. Garnier, H. Hamann, M. Montes de Oca, C. Solnon, T. Stützle (Eds.), Swarm Intelligence, Springer International Publishing, Cham, 2014, pp. 98–109.

[54] M. Quinn, A comparison of approaches to the evolution of homogeneous multi-robot teams, in: Proc. 2001 Congr. Evolutionary Computation (IEEE Cat. No.01TH8546), vol. 1, 2001, pp. 128–135, doi:10.1109/CEC.2001.934381.