Task Allocation in Multi-Agent Systems with Grammar-Based Evolution

Conference Paper · September 2021

DOI: 10.1145/3472306.3478337

CITATIONS
2

READS
110

4 authors, including:

Dilini Samarasinghe
UNSW Sydney
8 PUBLICATIONS **23** CITATIONS
SEE PROFILE

Erandi Lakshika
UNSW Sydney
32 PUBLICATIONS **109** CITATIONS
SEE PROFILE

Kathryn Kasmarik
Australian Defence Force Academy
141 PUBLICATIONS **1,319** CITATIONS
SEE PROFILE

Task Allocation in Multi-Agent Systems with Grammar-Based Evolution

Dilini Samarasinghe
University of New South Wales
Canberra, Australia
d.samarasinghe@adfa.edu.au

Michael Barlow
University of New South Wales
Canberra, Australia
m.barlow@adfa.edu.au

Erandi Lakshika
University of New South Wales
Canberra, Australia
e.henekankanamge@adfa.edu.au

Kathryn Kasmarik
University of New South Wales
Canberra, Australia
kathryn.kasmarik@adfa.edu.au

ABSTRACT

This paper presents a grammar-based evolutionary model to facilitate autonomous emergence of task allocation for intelligent multi-agent systems. The approach adopts a context-free grammar to determine the behaviour rule syntax. This allows for flexibility in evolving task allocation under multiple and dynamic constraints without manual rule design and parameter tuning. Experimental evaluations conducted with a target discovery simulation illustrate that the grammar-based model performs successfully in both dynamic and non-dynamic conditions. A statistically significant performance improvement is shown compared to an algorithm developed with the broadcast of local eligibility mechanism and a genetic programming mechanism. Grammatical evolution can achieve near-optimal solutions under restrictions applied on the number of agents, targets and the time allowed. Further, analysis of the evolved rule structures shows that grammatical evolution can identify less complex rule structures for behaviours while maintaining the expected level of performance. The results infer that the proposed model is a promising alternative for dynamic task allocation with human interactions in complex real-world domains.

CCS CONCEPTS

• **Computing methodologies** → **Multi-agent systems**; **Cooperation and coordination**; *Genetic programming*.

KEYWORDS

grammatical evolution, multi-agent systems, task allocation

ACM Reference Format:
Dilini Samarasinghe, Michael Barlow, Erandi Lakshika, and Kathryn Kasmarik. 2021. Task Allocation in Multi-Agent Systems with Grammar-Based Evolution. In *21st ACM International Conference on Intelligent Virtual Agents (IVA '21), September 14–17, 2021, Virtual Event, Japan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3472306.3478337

IVA '21, September 14–17, 2021, Virtual Event, Japan

ACM ISBN 978-1-4503-8619-7/21/09…$15.00
https://doi.org/10.1145/3472306.3478337

Multi-agent systems (MASs) are emerging as an enabling technology for designing intelligent virtual agents (IVAs) that are capable of social interactions and human-like decision making, behaviour generation, and planning in large-scale, distributed applications. A key limitation in the current multi-agent models is the lack of human interpretable solution spaces that render these models less ideal to be handled, controlled or monitored by humans. Search and rescue [1], exploration and surveillance [12], and planning [3] are real world applications where IVAs are employed with human controllers for distributed sensing and actions. Although existing literature discusses algorithms targeting optimal distributed decision making, most cannot cater to dynamic conditions and varying constraints, and can only provide black box solutions that cannot be easily interpreted by humans [4].

Grammatical Evolution (GE) [19] is an evolutionary computing technique with the unique ability to represent its solution space in the form of programs (tree structures), utilising a context-free grammar (CFG) to define their syntax. We have previously studied GE in consensus based environments for automatically synthesising cooperative behaviours [23, 24]. In this paper, we investigate GE in automatic emergence of task allocation within a distributed learning environment by addressing the limitations of the existing IVA modeling techniques that limit human comprehension and explainability of the solutions. The task allocation model tested is designed for a situation without human interaction for simplicity of testing, but the technique is directly extensible for MASs with a human controller. The proposed approach which derives solutions based on CFG based representations lends itself to human understanding, and can be examined and reverse-engineered to be adapted in similar environments for social interactions and decision making.The contributions of the paper can be listed as follows:

- A grammar-based evolutionary approach for automatic synthesis of task allocation is introduced.
- The model is investigated under non-dynamic and dynamic conditions to establish its flexibility to adapt.
- The proposed GE approach is compared against algorithms designed based on: (1) the broadcast of local eligibility (BLE) mechanism; and (2) a genetic programming (GP) mechanism.
- The behaviour rules evolved are analysed discussing their rule structure and complexity.

The rest of the paper is organised as follows. Section 1 summarises the literature, and the proposed approach is presented in

Section 2. Section 3 illustrates the experimental setup and results with Section 4 concluding the paper.

1 RELATED WORK

1.1 Task Allocation

Task allocation is mainly of two variants: centralised and decentralised allocation. In centralised mechanisms, a central agent assesses the requirements and tasks, and coordinates the execution process [26]. A human agent or a cognitive tool acts as the allocator [31] in controlling the task distribution. These mechanisms are less robust and are poor in fault tolerance as the entire system risks failure if the central node malfunctions.

Decentralised task allocation algorithms employ a distributed mechanism where each agent is responsible for making their own decision about the tasks to approach. Game theory and negotiation [5], auction [27], and distributed ant colony and particle swarm optimization [13] based mechanisms are examples of this category. These are more robust and fault tolerant in comparison to centralised approaches. Nevertheless, these approaches in most cases cannot cope with dynamic tasks [25] and restrictions on communication [7] and time [28] as the models are difficult to be generalised. This paper aims to address this challenge by deriving a single model that can derive flexible rules based on a generalised grammar, and can adapt to dynamic conditions and constraints to generate near-optimal solutions in different environments.

1.2 Towards Grammatical Evolution

Manually designed IVAs are often non-adaptive and are limited in capacity to change the functions of system components. As a result, automatic synthesis methods such as reinforcement learning (RL) [11] and evolutionary computing [8] have been successfully applied in domains in which human reasoning and judgement is insufficient to comprehend and solve complex problems. However, they have primarily concentrated on automating the control of the parameters necessary for formulating rules rather than the rule structures themselves. These structures are mostly either manually designed [14] or represented by an artificial neural network (ANN) [15, 30] that hinders the understanding of the rules for analysis and reuse for complex behaviour generation [9].

The natural representation of agent behavioural rules can be exploited in the autonomous behaviour-generation process to address this limitation. The inherent nature of the rules is to abide by a syntax (a tree structure) that is formulated by control structures, parameters, actions and logical connectors. GE is a variation of GP that restricts the solution space to a particular syntax through a CFG that serves as a medium through which to translate the solution to valid rules. Hence, it facilitates controlling the structure of evolved populations.

GE has been studied with single-agent systems in the IVA domain such as for horse gait optimisation [17], and for controllers in gaming environments [20, 21] and other multi-agent domains such as robotics and simulations [18, 29]. Nevertheless, the existing GE models have not concentrated on distributed learning environments where IVA-based task allocation is concerned. The focus of this paper is to extend GE towards this domain to enhance the potential of the approach.

2 GRAMMAR-BASED EVOLUTION

2.1 Problem Statement

In formally introducing the task allocation problem, we define a MAS with the tuple M = { Ψ, T, R, G } where:

- Ψ = { ψ_1, ψ_2, .., ψ_μ; 1<μ } is a set of more than one agent ψ, where μ is the number of agents.
- T = { τ_1, τ_2, .., τ_κ; 1<κ } is a set of more than one task τ, where κ is the number of tasks to be covered.
- R is the rule set that defines the agent behaviours.
- G is the goal state the entire agent system is required to achieve defined by the objective function $O(G)$. It could be defined based on different aspects such as maximising the number of tasks covered and minimising the consumption of resources and time spent.

In this paper, the objective $O(G)$ is defined based on maximising the number of tasks covered by all agents:

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

Given a set Ψ of agents and a set T of tasks, the aim is to optimise the fitness to achieve the rule set R to reach goal G.

2.2 Proposed Evolutionary Model

Figure 1 illustrates the evolutionary model. The CFG should be designed prior to the algorithm. The proposed model differs from other GE models as we focus on the most basic or the 'atomic' components of agent behaviour rules to design the CFG. Therefore, the design consists of three primary steps. First, an appropriate rule space which acts as the set of atomic components of the behaviour rules is identified. The four types of components present in any behaviour rule are: (1) the control structure, (2) parameters, (3) logical and relational connectives and (4) elementary actions [24].

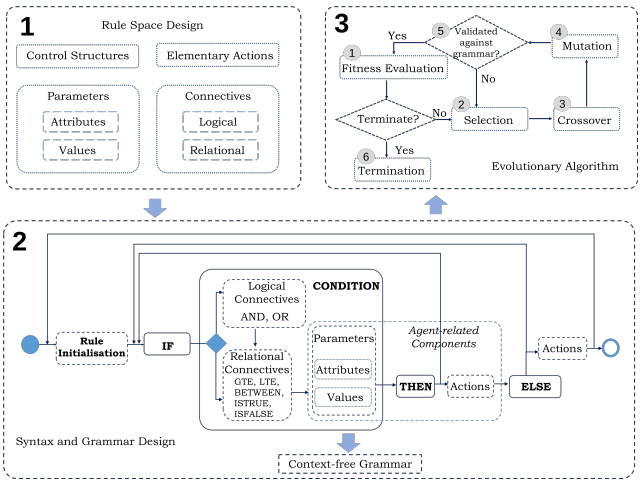

Figure 1: Proposed GE model for automatic synthesis of task allocation in MASs. As the first step, attributes related to the four components of the rule space are identified and the CFG is designed based on this rule space. Finally, the evolutionary algorithm which uses the designed CFG to validate the evolving rules is designed.

The next step is to design a syntax and a CFG as shown in step 2 of Figure 1 to reflect the structure of the rules accurately. The CFG is defined as Gr = $\nu, \gamma, \rho, \varsigma$, where ν is the set of non-terminals, γ is the set of terminals, ρ is the production ruleset, and ς is the start symbol (⟨S⟩). The non-terminals on left of production rules can be replaced with other terminals and non-terminals. The terminals cannot be further decomposed. Figure 2 illustrates these production rules. The flow of operations and grammar components for valid rules can be extracted from this syntax. The rules are initiated with an IF-THEN-ELSE control. The condition (⟨O⟩) formulation has two approaches: with a relational connective and respective parameters; or with a fusion of several relational connectives combined with a logical connective. Followed by the condition is the result if it is true. This portion comprises an action (⟨A⟩) or another IF construct (⟨I⟩) leading to a rule nested within the former rule. The ELSE component could again comprise an action or another nested IF construct. Each behaviour can be composed of multiple rule components combined based on weights ⟨W⟩. The parameters taken into account are: distance of the agent to a target (`distance to target`), distance of neighbouring agents to a target (`distance neighbour to target`) and whether the targets have been discovered or not. The elementary actions include: moving away and towards the target, moving forward and turning at an angle.

The third step is to design the evolutionary algorithm utilising the CFG to validate the evolving behaviour rules. The search space of GE is represented as binary strings known as 'Codons'. The genetic operations are applied on the codons while the CFG is used in mapping the search space of codons to a solution space of actual rules with the following function:

[Integer value of codon] MOD [Production rules for the non-terminal]

Consider the example given in Figure 3 where the genome consists of six codons (the integer values corresponding to the binary strings are shown). The mapping process starts with ⟨S⟩ being replaced by ⟨S1⟩. Mapping function is not applied as ⟨S⟩ has only one production rule. In the next step, ⟨S1⟩ has two production rules, and the mapping function is applied (5 MOD 2 = 1) replacing it with the production rule ⟨I⟩ ∗ ⟨W⟩. The process is repeated in a similar fashion until a valid expression is reached.

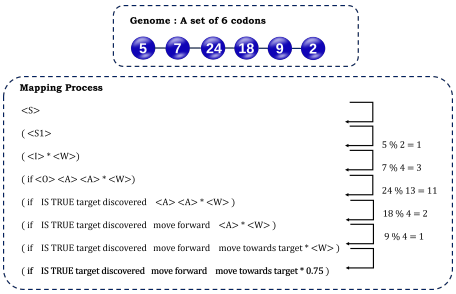

Figure 3: An example mapping of a genome consisting of six codons to a valid behaviour rule based on the CFG.

The evolutionary algorithm is detailed in Algorithm 1. It starts with a valid random population and fitness evaluation is performed. Two parents are then selected and offspring are generated after applying crossover and mutation. The offspring are validated to ensure syntax accuracy against the CFG and if they do not map to

⟨S⟩	⊨	⟨S1⟩	1
⟨S1⟩	⊨	⟨S1⟩ ⟨S1⟩ | ⟨I⟩∗⟨W⟩	2
⟨I⟩	⊨	`if` ⟨O⟩ ⟨I⟩ ⟨I⟩ | `if` ⟨O⟩ ⟨I⟩ ⟨A⟩	4
		| `if` ⟨O⟩ ⟨A⟩ ⟨I⟩ | `if` ⟨O⟩ ⟨A⟩ ⟨A⟩	
⟨O⟩	⊨	`and` ⟨O⟩ ⟨O⟩ | `or` ⟨O⟩ ⟨O⟩	13
		| `LTE distance to target` ⟨distance⟩	
		| `LTE distance neighbour to target` ⟨distance⟩	
		| `LTE distance neighbour to target distance to target`	
		| `GTE distance to target` ⟨distance⟩	
		| `GTE distance neighbour to target` ⟨distance⟩	
		| `GTE distance neighbour to target distance to target`	
		| `BETWEEN distance to target` ⟨distance⟩ ⟨distance⟩	
		| `BETWEEN distance neighbour to target` ⟨distance⟩ ⟨distance⟩	
		| `BETWEEN distance neighbour to target distance to target` ⟨distance⟩	
		| `IS TRUE target discovered`	
		| `IS FALSE target discovered`	
⟨A⟩	⊨	`move away from target` | `move towards target`	4
		| `move forward` | `turn by` ⟨angle⟩	
⟨distance⟩	⊨	`distance value`	1
⟨angle⟩	⊨	`angle value`	1
⟨W⟩	⊨	`weight value`	1

Figure 2: Production rules (ρ) of the grammar (Gr). The range of values for distance and angle are (0 - width of the world], and (0 - 2π] respectively. The number in front of each non-terminal corresponds to the number of production rules associated with it.

valid rules, new parents are selected. The process is repeated until the termination criteria is met.

Algorithm 1 Grammatical Evolution for Task Allocation

Require: ρ: Set of CFG production rules
β: Size of the population
Ω: Maximum no. of generations
ℓ: Maximum no. of loops
Ensure: I_b: Best individual genome

procedure GrammaticalEvolution(ρ, β, Ω)
  $pop \leftarrow$ InitialisePopulation(β, ρ)
  $\omega \leftarrow 0$
  while $\omega \neq \Omega$ **do**
    for $i \in pop$ **do**
      EvaluateFitness(i)
    $c_{M1}, c_{M2} \leftarrow null$
    $valid \leftarrow false$
    $loops++$
    while $valid == false$ **do**
      $parents \leftarrow$ ParentSelection(pop)
      for $p1, p2 \in parents$ **do**
        $children \leftarrow$ Crossover($p1, p2$)
        for $c1, c2 \in children$ **do**
          $c_{M1} \leftarrow$ Mutate($c1, prob_{mut}$)
          $c_{M2} \leftarrow$ Mutate($c2, prob_{mut}$)
      if MapCFG(ρ, c_{M1}) == $true$ AND MapCFG(ρ, c_{M2}) == $true$ **then**
        $valid \leftarrow true$
      else if $loops == \ell$ **then**
        $valid \leftarrow true$
    $I_W \leftarrow$ GetTwoWorstFitIndivs(pop)
    for $I_{W1}, I_{W2} \in I_W$ **do**
      $I_{W1} \leftarrow$ ReplaceIndiv(I_{W1}, c_{M1})
      $I_{W2} \leftarrow$ ReplaceIndiv(I_{W2}, c_{M2})
    $I_b \leftarrow$ GetMostFitIndiv(pop)
    $\omega++$
  return I_b

3 EXPERIMENTAL EVALUATIONS

3.1 Compared Models

We developed two comparison models: one based on BLE and another based on GP to evaluate the performance of GE. As the proposed GE model is rule-based, the comparison models were also chosen from the commonly used rule-based strategies for a fair comparison. GP is the most widely used rule-based automation technique, where as BLE is a commonly adopted manual rule-based approach used for agent-based learning.

BLE [32] compares the eligibility of an agent to perform a task against that of the rest of the group before it can claim it. Therefore, the BLE rule in Figure 4 compares the distance of an agent from a target against that of the neighbours. If the agent is the closest and the target has not yet been discovered, it will claim it.

For a fair comparison, the same GE rule space was used to define the functions and terminals of the GP model. All actions except *turn by* were pooled as terminals that take no arguments. The control structure, connectives and the action *turn by* were categorised as functions that accept arguments. As genetic operations are performed on the rule space itself, GP requires the functions to be defined for any combination of arguments from the rule space. This limitation of GP is known as the closure property [22]. Therefore, the components which do not return a numerical value were modified to return a value to adhere to this requirement.

```
IF [IS FALSE] [target discovered]
 IF [distance neighbour to target] < [distance to target]
   [move away from target]
 ELSE
   [move towards target]
ELSE
 [move away from target]
```

Figure 4: The broadcast of local eligibility (BLE) rule.

Algorithm 2 illustrates the GP process. An initial random population of rules is first evaluated for fitness. Crossover and mutation are then performed on selected parents based on the fitness. The resulting offspring replace the least fit individuals of the population, and the process is repeated until the termination criteria is met.

Algorithm 2 Genetic Programming for Task Allocation

Require: F : Set of primitive functions
T : Set of terminals
Ω : Maximum generations
Ensure: I_b : Best individual program

procedure GeneticProgramming(F, T, Ω)
  $pop \leftarrow$ InitialisePopulation(F, T)
  $\omega \leftarrow 0$
  while $\omega \neq \Omega$ **do**
    for $pop_i \in pop$ **do**
      EvaluateFitness(pop_i)
    $parents \leftarrow$ ParentSelection(pop)
    $children \leftarrow$ Crossover(pop)
    $children \leftarrow$ Mutate($children$)
    $pop \leftarrow$ UpdateWith($children$)
    $I_b \leftarrow$ GetMostFitIndiv(pop)
    $\omega++$
  return I_b

3.2 Experimental Setup

The evolutionary attributes are as given in Table 1 and the task allocation domain is based on a target search environment as illustrated in Figure 5. A set of targets is arranged at random locations and the agents should utilise distributed decision making capabilities to attend to a maximum number of targets within a given time frame. Experiments are conducted in both non-dynamic and dynamic environments. In dynamic environments, the target locations shift randomly during the simulation at every 50 ticks.

Table 1: Evolutionary and environmental attributes of the setup.

Attribute	Value
Evolutionary Algorithm	
Population Size	30
Evolutionary Strategy	SSR
Parent Selection	Tournament (size:5)
Mutation Probability	0.005
Maximum Generations	1000
Evolutionary Runs	15
Simulation Environment	
No. of Agents	5
No. of Targets	30
Time Allowed	500 ticks
World Size	1000 x 1000 units
World Nature	Wraparound
Agent Speed	3 units per tick

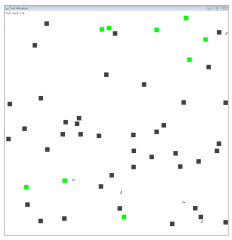

Figure 5: The simulation environment with unallocated targets in black. As they get allocated, targets change to green.

The fitness function for measuring the behaviour improvement is derived based on a minimising function of the normalised total number of targets discovered as given in Equation 2.

$$fitness = 1 - (discovered\_targets \div total\_targets) \quad (2)$$

The GE and GP algorithms are implemented with the steady state replacement (SSR) mechanism [19] where only the two least fit individuals of the population are replaced with the offspring to maintain more valid and fit individuals at the expense of invalid individuals. Agents sense the neighbourhood based on a vision range defined by an angle and distance of vision (fixed at; distance=1000 units and angle=π for all three models for a fair comparison).

3.3 Evolutionary Results of GE and GP

Figure 6 illustrates the evolutionary results of the GE and GP models. The experiments are repeated for 15 runs each, and average fitness progression of the entire population and the best individual over 1000 generations are presented. Mann-Whitney U test is adopted to compare the significance of the difference between two populations as the samples are not normally distributed and the sample sizes are small. The statistical significance level is set at p=0.05.

According to the results, both GE and GP models show potential to minimise fitness (corresponding to an increase in the number of tasks discovered) over the generations. Both models evolve task

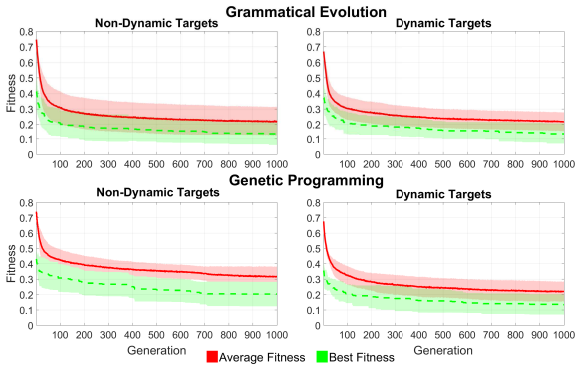

Figure 6: Evolution results for the task allocation behaviours with dynamic and non-dynamic target locations with GE and GP. The progression of the average and minimum fitness of the population are illustrated through 1000 generations. The results are averaged across 15 runs and the shaded areas depict the standard deviation.

allocation in dynamic and non-dynamic environmental conditions. In the non-dynamic environment, GE achieves an average fitness of 0.1378 at the end of 1000 generations for the best solution, which correspond to an average of ≈26 tasks out of 30. The results are in the same range for the dynamic environment and no statistically significant difference (p=0.8504 > 0.05) is observed between the performance in non-dynamic versus dynamic environments.

Conversely, GP only achieved an average fitness of 0.2022 (> 0.1378 achieved by the GE model) for the best solution in a non-dynamic environment. It is a statistically significant performance reduction in comparison to GE (p=0.0394 < 0.05). However, in a dynamic environment, the average fitness of the best solution was 0.1333, which shows that the performance of GP in this situation is in the same range as that of the GE model (p=0.8991 > 0.05).

In conclusion, GE is capable of performing equally well under both dynamic and non-dynamic conditions, whereas GP performs comparatively poorly in non-dynamic environments. Although dynamic environments are intuitively more difficult to master, it can be observed that the GP based agents are exploiting the random nature of targets being shifted in the dynamic environments to attend to tasks that are within close proximity with limited decision making capacity towards actual task allocation task. In contrast, it is not possible to do so with non-dynamic tasks which forces the agents to master task allocation. Their performance in this case falls below that of GE indicating the limitations of GP in achieving optimal task allocation performance.

3.4 Comparison Results

To further explore the performance in diverse conditions, we conduct evaluations in four focus areas with the following variations:

- Time allowed: 250, 500, 750, 1000 ticks
- Number of agents: 3, 5, 7, 10
- Number of targets: 20, 30, 40, 50
- Target location: non-dynamic, dynamic

For each evaluated variable, the other attributes are maintained at the standards given in Table 1. Figure 7 illustrates the number of

targets discovered with each variation for GE, GP and BLE for 15 runs each. The statistical results summary is shown in Table 2.

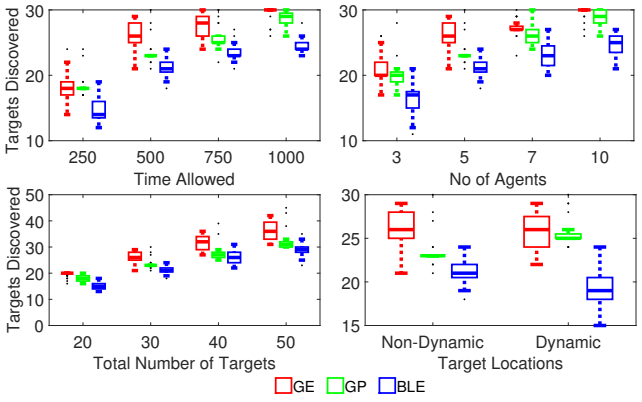

Figure 7: Comparison of number of targets discovered under different variable values for: time allowed, number of agents, number of targets and target location with the GE, GP and BLE models.

With all three approaches, the number of targets discovered increases proportional to the available time. However, GE demonstrates a statistically significant performance improvement for all four variations (p=0.00 < 0.05) compared to BLE, and for two of the four variations (with 500 and 1000 ticks p=0.04 < 0.05) compared to GP. Both GE and GP perform equally well in the evaluation with 250 and 750 ticks. The GE model achieves an optimal fitness discovering all targets in 11 out of the 15 runs as the time reaches 1000 ticks; whereas the times all targets were discovered by GP and BLE were 4 and 0 respectively. Therefore, overall the GE model performs better than GP and BLE under varying time constraints.

The discovered targets also increases proportional to the number of agents with all three models. GE shows a higher performance with all four variations (p=0.00 < 0.05) compared to BLE, and for one variation (with 5 agents p=0.04 < 0.05) compared to GP. However, the average performance of GP across the 15 runs is less than that of GE in all variations. For example, with 10 agents, the GE model discovers all targets in 10 out of 15 runs, whereas with GP it is only 5. BLE is not capable of discovering all targets in any run.

When varying the numbers of targets, both GE and GP perform equally well with a statistically insignificant difference in performance (p=0.07 > 0.05), and can identify all targets when the number of targets is set to 20. However, as the number of targets increase, the number of targets discovered by both approaches is progressively decreased. At the same time, the performance of GP degrades statistically significantly compared to that of GE for all three target variations with 30, 40 and 50 targets ($p < 0.05$) as it discovers less number of targets than GE. Both GE and GP models perform significantly better than BLE in all variations ($p \ll 0.05$).

GE also performs equally well under both dynamic and non-dynamic conditions and perform better than BLE in both instances, and better than GP with non-dynamic conditions. For BLE, it is significantly more difficult to solve the task with dynamic target locations in comparison to non-dynamic locations (p=0.02 < 0.05).

It is also noted that GP finds at least one solution with similar performance to GE in 6 and better performance than GE in 8 out of the 14 variations tested. However, the average performance of GP across all 14 variations is below that of GE. This can be attributed to the exploration and exploitation balance between GE and GP. GP crossover swaps subtrees at random without allowing context preservation in the chromosomes [6] which leads to higher exploration than exploitation. As a result GP conducts more random exploration compared to GE and occasionally finds solutions that perform exceptionally well but it is not a successful strategy on average as shown with the results.

In conclusion, the BLE model often converges towards suboptimal solutions whereas GP performs comparatively better. The GE

Table 2: The statistical results summary of the algorithms. The difference between GE versus other two approaches are shown with the p-values.

Time					**Number of Targets**					**Number of Agents**					**Target Location**				
Var	Min	Max	Avg	p	Var	Min	Max	Avg	p	Var	Min	Max	Avg	p	Var	Min	Max	Avg	p
250					**20**					**3**					**Non-Dynamic**				
GE	0.20	0.53	0.39	-	GE	0.00	0.20	0.05	-	GE	0.13	0.43	0.28	-	GE	0.03	0.30	0.14	-
BLE	0.37	0.60	0.50	**0.00**	BLE	0.10	0.35	0.25	**0.00**	BLE	0.30	0.63	0.46	**0.00**	BLE	0.20	0.40	0.29	**0.00**
GP	0.20	0.43	0.38	0.69	GP	0.00	0.20	0.09	0.07	GP	0.07	0.43	0.33	0.11	GP	0.00	0.30	0.20	**0.04**
500					**30**					**5**					**Dynamic**				
GE	0.03	0.30	0.14	-	GE	0.03	0.30	0.14	-	GE	0.03	0.30	0.14	-	GE	0.03	0.27	0.14	-
BLE	0.20	0.40	0.29	**0.00**	BLE	0.20	0.40	0.29	**0.00**	BLE	0.20	0.40	0.29	**0.00**	BLE	0.20	0.50	0.35	**0.00**
GP	0.00	0.30	0.20	**0.04**	GP	0.00	0.30	0.20	**0.04**	GP	0.00	0.30	0.20	**0.04**	GP	0.00	0.20	0.13	0.90
750					**40**					**7**									
GE	0.00	0.20	0.08	-	GE	0.10	0.33	0.21	-	GE	0.00	0.23	0.09	-					
BLE	0.10	0.30	0.21	**0.00**	BLE	0.23	0.45	0.35	**0.00**	BLE	0.10	0.33	0.23	**0.00**					
GP	0.00	0.26	0.12	0.05	GP	0.03	0.38	0.29	**0.00**	GP	0.00	0.20	0.10	0.29					
1000					**50**					**10**									
GE	0.00	0.13	0.02	-	GE	0.16	0.38	0.27	-	GE	0.00	0.13	0.02	-					
BLE	0.07	0.23	0.18	**0.00**	BLE	0.30	0.54	0.42	**0.00**	BLE	0.10	0.30	0.19	**0.00**					
GP	0.00	0.13	0.05	**0.04**	GP	0.10	0.40	0.34	**0.01**	GP	0.00	0.13	0.05	0.09					

model performs better than both showing potential to generate near-optimal solutions and significant performance improvements.

3.5 Evolved Rules

This section evaluates the nature of the rules evolved with the proposed GE model. Figure 8 illustrates two indicative rules evolved with GE and GP.

The GE rule in Figure 8a aggregates two individual behaviour vectors with weights 0.65 and 0.35 for each. The first rule component with more weight evaluates if the agent is within a distance of 320 units from the target and if the target has not yet been discovered. If both conditions are satisfied, the agent moves towards the target; else moves away from the target. According to the second rule component, if another agent is closer to the target, the agent will keep moving forward; else it will move towards the target. A manual approach to rule design would require identifying all these components and tuning the parameter values, whereas with GE, the entire rule structure including the parameters can be evolved automatically as described. As the rule is syntactically valid, it can be interpreted and extracted from the evolutionary context to be used in similar non-evolutionary environments.

Conversely, the GP rule cannot be interpreted to extract the functioning behind the behaviour vectors generated. The structure does not adhere to a syntax except for *function-terminal* categorisation, and examining the output rule does not reveal any insights of how it functions. As discussed in Section 3.1, GP maintains structure with the use of numerical values returned by terminals. The combination of actions *move towards target*, *move forward* and *turn by* that chose to behave appropriately based on parameter values as of that moment resulted in an acceptable behaviour. However, it does not provide any information that can be used to reverse engineer the behaviour or enhance it to be used in a similar context. Despite both mechanisms using tree structure representations for solutions, this is a significant drawback of GP in comparison to GE. The proposed GE model can therefore be useful in complex environments where human comprehension becomes a limiting factor to understand the problem domain and solutions.

To discuss complexity, Figure 9 illustrates the variation of cyclomatic complexity (CC) [16] determining the number of independent paths through a rule, and the number of individual rules in each behaviour against the fitness variation over 1000 generations.

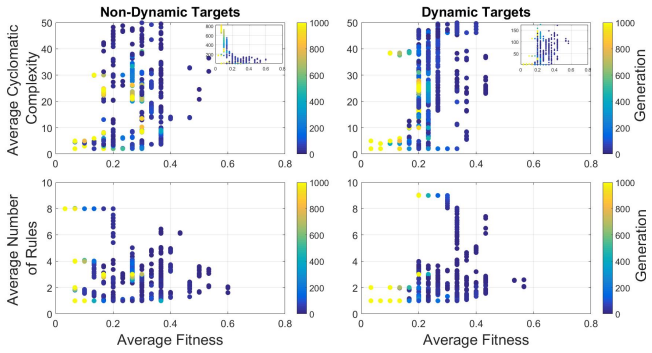

Figure 9: The variation of cyclomatic complexity and the average number of rule components in relation to fitness for 15 runs with the GE model. The values are averaged across all 30 individuals in the population and colour coded based on the generation.

The average CC of the populations during the last generations are generally < 50 with both dynamic and non-dynamic environments. However, during the initial generations, they explore higher complexity values of even around 100-200 (as shown within the inset figures inside the main figures) before converging towards less complex solutions. The number of rules in a single evolved solution also averages to ≈2 in both environments as the fitness improves. Yet, during the evolutionary process, they explore complex behaviours with up to 10 rules. In conclusion, the GE model explores a wider range of solutions with higher complexities during the initial generations and gradually learns to identify a balance between the complexity and performance, thus reducing the complexity of rules while maintaining a better fitness.

4 CONCLUSION AND FUTURE WORK

This paper presented a GE-based algorithm to address the challenges of multi-agent task allocation under different constraints. Task allocation has been attempted with approaches such as consensus and distributed algorithms, graph-based search [2], game theoretic frameworks [33] and swarm robotics [10]. However, they focus on the state and parameter space of agent behaviours rather than their underlying rule structures. Our proposed model cater to this gap, by allowing automatic emergence of task allocation.

```
[Rule1*0.65
        [<if> [<and> [<LTE> distance_to_target 320]
                [<IS FALSE> target_discovered]]
                        move_towards_target
                        move_away_from_target]]
[Rule2*0.35
        [<if> [<GTE> distance_neighbour_to_target distance_to_target]
                move_forward
                move_towards_target]]
```

(a) An indicative rule evolved with GE.

```
[BETWEEN distance_to_target
                [<LTE> move_towards_target
                        [<if> move_forward
                                [<turn_by> move_towards_target]
                                [<or> distance_to_target 220]]]]
```

(b) An indicative rule evolved with GP.

Figure 8: Analysis of rule structures generated by the GE and GP models.

It was compared against BLE and GP in dynamic and non-dynamic environments and under different constraints on agents, targets and time. The results showed that the proposed GE model can generate near-optimal solutions more frequently than the BLE and GP models and can flexibly adapt to varying conditions. The BLE model was unable to successfully perform under constraints. GP performed well in a few variations but the more restrictive conditions led to suboptimal solutions. The GE model maintained a higher performance and evolved rule structures that can cater to different requirements of the environment. Further, GE also showed capacity to explore a wider range of structures to identify relatively less complex solutions with the expected level of performance.

As future research directions, the algorithm can be extended for applicability with a multi-objective optimisation approach where multiple other criteria such as minimising resource consumption and time spent can also be taken into account. There is also potential to explore the GE approach for task specialisation in IVA settings with a human controller. Instead of allocating tasks for each agent, the process of covering a task can be divided into subtasks where each agent specialises in one or more of the subtasks as a means to optimally solve the original task. The observations and potential applications of the proposed model lend valuable evidence to further explore GE as a preferable tool not only to generate intelligent agents with sophisticated capabilities but also to reverse-engineer the evolved rules to be used in similar and related environments with real-world application requirements.

REFERENCES

[1] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa. 2019. LSAR: Multi-UAV Collaboration for Search and Rescue Missions. *IEEE Access* 7 (2019), 55817–55832.

[2] C. Banks, S. Wilson, S. Coogan, and M. Egerstedt. 2020. Multi-Agent Task Allocation using Cross-Entropy Temporal Logic Optimization. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 7712–7718.

[3] Y Bestaoui Sebbane. 2020. *Multi-UAV Planning and Task Allocation.* Chapman and Hall/CRC, New York.

[4] Davide Castelvecchi. 2016. Can we open the black box of AI? *Nature News* 538, 7623 (2016), 20.

[5] Rongxin Cui, Ji Guo, and Bo Gao. 2013. Game theory-based negotiation for multiple robots task allocation. *Robotica* 31, 6 (2013), 923.

[6] P. D'haeseleer. 1994. Context preserving crossover in genetic programming. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence.* 256–261 vol.1. https://doi.org/10.1109/ICEC.1994.350006

[7] R. Doriya, S. Mishra, and S. Gupta. 2015. A brief survey and analysis of multi-robot communication and coordination. In *International Conference on Computing, Communication Automation.* 1014–1021.

[8] Agoston E Eiben and James E Smith. 2003. *Introduction to evolutionary computing.* Vol. 53. Springer.

[9] Eliseo Ferrante, Edgar Duéñez Guzmán, Ali Emre Turgut, and Tom Wenseleers. 2013. GESwarm: Grammatical Evolution for the Automatic Synthesis of Collective Behaviors in Swarm Robotics. In *Proc. 15th Annu. Conf. Genetic and Evolutionary Computation* (Amsterdam, The Netherlands) *(GECCO '13)*. Association for Computing Machinery, New York, NY, USA, 17–24. https://doi.org/10.1145/2463372.2463385

[10] John Harwell. 2020. A Theoretical Framework For Self-Organized Task Allocation in Large Swarms. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems.* 2191–2192.

[11] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research* 4 (1996), 237–285.

[12] Nikolaos Kariotoglou, Davide M. Raimondo, Sean J. Summers, and John Lygeros. 2014. Multi-Agent Autonomous Surveillance: A Framework Based on Stochastic Reachability and Hierarchical Task Allocation. *Journal of Dynamic Systems, Measurement, and Control* 137, 3 (10 2014). 031008.

[13] AM Senthil Kumar and M Venkatesan. 2019. Task scheduling in a cloud computing environment using HGPSO algorithm. *Cluster Computing* 22, 1 (2019), 2179–2185.

[14] E. Lakshika, M. Barlow, and A. Easton. 2013. 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)*. 82–89. https://doi.org/10.1109/ALIFE.2013.6602435

[15] Nam Le, Anthony Brabazon, and Michael O'Neill. 2019. The Evolution of Self-taught Neural Networks in a Multi-agent Environment. In *Applications of Evolutionary Computation*, Paul Kaufmann and Pedro A. Castillo (Eds.). Springer International Publishing, Cham, 457–472.

[16] T. J. McCabe. 1976. A Complexity Measure. *IEEE Transactions on Software Engineering* SE-2, 4 (1976), 308–320.

[17] James E. Murphy, Michael O'Neill, and Hamish Carr. 2009. Exploring Grammatical Evolution for Horse Gait Optimisation. In *Genetic Programming*, Leonardo Vanneschi, Steven Gustafson, Alberto Moraglio, Ivanoe De Falco, and Marc Ebner (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 183–194.

[18] Aadesh Neupane and Michael Goodrich. 2019. Learning Swarm Behaviors using Grammatical Evolution and Behavior Trees. In *Procc. 28th Int. Joint Conf. Artificial Intelligence, IJCAI-19.* International Joint Conferences on Artificial Intelligence Organization, 513–520. https://doi.org/10.24963/ijcai.2019/73

[19] M. O'Neill and C. Ryan. 2001. Grammatical evolution. *IEEE Transactions on Evolutionary Computation* 5, 4 (Aug 2001), 349–358. https://doi.org/10.1109/4235.942529

[20] Diego Perez, Miguel Nicolau, Michael O'Neill, and Anthony Brabazon. 2011. Evolving Behaviour Trees for the Mario AI Competition Using Grammatical Evolution. In *Applications of Evolutionary Computation*, Di Chio et al. (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 123–132.

[21] Diego Perez-Liebana and Miguel Nicolau. 2018. Evolving Behaviour Tree Structures Using Grammatical Evolution. In *Handbook of Grammatical Evolution*, Conor Ryan, Michael O'Neill, and JJ Collins (Eds.). Springer International Publishing, Cham, 433–460. https://doi.org/10.1007/978-3-319-78717-6_18

[22] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. 2008. *A Field Guide to Genetic Programming.* Lulu Enterprises, UK Ltd.

[23] Dilini Samarasinghe, Michael Barlow, Erandi Lakshika, and Kathryn Kasmarik. 2021. Exploiting abstractions for grammar-based learning of complex multi-agent behaviours. *International Journal of Intelligent Systems* (2021). https://doi.org/10.1002/int.22550 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/int.22550

[24] Dilini Samarasinghe, Erandi Lakshika, Michael Barlow, and Kathryn Kasmarik. 2018. Automatic Synthesis of Swarm Behavioural Rules from Their Atomic Components. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Kyoto, Japan) *(GECCO '18)*. ACM, New York, NY, USA, 133–140.

[25] Janaína Schwarzrock, Iulisloi Zacarias, Ana L.C. Bazzan, Ricardo Queiroz de Araujo Fernandes, Leonardo Henrique Moreira, and Edison Pignaton de Freitas. 2018. Solving task allocation problem in multi Unmanned Aerial Vehicles systems using Swarm intelligence. *Engineering Applications of Artificial Intelligence* 72 (2018), 10 – 20.

[26] Onn Shehory and Sarit Kraus. 1998. Methods for task allocation via agent coalition formation. *Artificial Intelligence* 101, 1 (1998), 165 – 200.

[27] Jieke Shi, Zhou Yang, and Junwu Zhu. 2020. An auction-based rescue task allocation approach for heterogeneous multi-robot system. *Multimedia Tools and Applications* 79, 21 (2020), 14529–14538.

[28] V. Singhal and D. Dahiya. 2015. Distributed task allocation in dynamic multi-agent system. In *International Conference on Computing, Communication Automation.* 643–648.

[29] John Mark Swafford, Michael O'Neill, Miguel Nicolau, and Anthony Brabazon. 2011. Exploring Grammatical Modification with Modules in Grammatical Evolution. In *Genetic Programming*, Sara Silva, James A. Foster, Miguel Nicolau, Penousal Machado, and Mario Giacobini (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 310–321.

[30] Ardi et al. Tampuu. 2017. Multiagent cooperation and competition with deep reinforcement learning. *PLOS ONE* 12, 4 (04 2017), 1–15. https://doi.org/10.1371/journal.pone.0172395

[31] N. Tsiogkas, G. Papadimitriou, Z. Saigol, and D. Lane. 2014. Efficient multi-AUV cooperation using semantic knowledge representation for underwater archaeology missions. In *2014 Oceans - St. John's.* 1–6.

[32] Barry Brian Werger and Maja J. Mataric. 2000. Broadcast of Local Eligibility: Behavior-Based Control for Strongly Cooperative Robot Teams. In *Proceedings of the Fourth International Conference on Autonomous Agents* (Barcelona, Spain) *(AGENTS '00)*. Association for Computing Machinery, New York, NY, USA, 21–22.

[33] Vahid Yazdanpanah, Mehdi Dastani, Shaheen Fatima, Nicholas R Jennings, Devrim M Yazan, and Henk Zijm. 2020. Task Coordination in Multiagent Systems. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems.* 2056–2058.