Explaining Evolutionary Agent-Based Models via Principled Simplification

Chloe M. Barnes[1,*], Abida Ghouri[2], and Peter R. Lewis[3,4]

[1,*] *Corresponding author:* Aston University, Birmingham, B4 7ET, UK. c.barnes1@aston.ac.uk

[2]abida.ghouri@outlook.com

[3]Aston University, Birmingham, B4 7ET, UK.

[4]Ontario Tech University, Oshawa, ON L1G 0C5, Canada. peter.lewis@ontariotechu.ca

Abstract

Understanding how evolutionary agents behave in complex environments is a challenging problem. Agents can be faced with complex fitness landscapes, derived from multi-stage tasks, interaction with others, and limited environmental feedback. Agents that evolve to overcome these can sometimes access greater fitness, as a result of factors such as cooperation and tool use. However, it is often difficult to explain why evolutionary agents behave in certain ways, and what specific elements of the environment or task may influence the ability of evolution to find goal-achieving behaviours; even seemingly simple environments or tasks may contain features that affect agent evolution in unexpected ways. We explore *principled simplification* of evolutionary agent-based models, as a possible route to aiding their explainability. Using the River Crossing Task (RCT) as a case study, we draw on analysis in the Minimal River Crossing (RC-) Task testbed, which was designed to simplify the original task while keeping its key features. Using this method, we present new analysis concerning when agents evolve to solve the RCT. We demonstrate that the RC- environment can be used to understand the effect that a cost to movement has on agent evolution, and that these findings can be generalised back to the original RCT. Then, we present new insight into the use of principled simplification in understanding evolutionary agents. We find evidence that behaviour dependent on features that survive simplification, such as problem structure, are amenable to prediction; while predicting behaviour dependent on features that are

typically reduced in simplification, such as scale, can be invalid.

Keywords— *Principled Simplification, Evolutionary Algorithms, Evolutionary Agents, Explainability, River Crossing, Neuroevolution*

1 Introduction

Understanding, explaining, and learning about complex natural phenomena and living systems are some of the aims of artificial life research (Aguilar et al., 2014; Bedau, 2007). This often involves creating simple models that reduce the complexity of the original scenario, such that the phenomena can be studied in detail. Tasks that individuals face in both natural and artificial life can be complex, involving many stages (Brutschy et al., 2014); studying how artificial agents learn to solve these tasks can therefore be beneficial for explaining or predicting more complex artificial scenarios, or phenomena in natural life. For example, Stanton and Channon (2015) have explored how three-dimensional virtual agents can evolve to learn sub-tasks in order to solve increasingly difficult problems. Nicolay et al. (2014) have explored the effect that the order that conflicting tasks are learnt in has on performance in virtual robots. Additionally, Brutschy et al. (2014) show in simulation that interdependent tasks can be broken down and allocated to individuals in a swarm in simulation; further, they show that the approach in simulation can be transferred to physical robots, thus demonstrating the value of performing the simulation experiments. Re-representing the problem with minimal complexity therefore can, in principle, aid understanding of the problem itself.

In summary, this approach to generating such an understanding is to develop a simplified yet sufficiently useful testbed, that facilitates analysis while retaining key problem features, and such that insights can be generalised back to the original problem in the form of explanations. Here, we refer to this as *principled simplification*, and we note that has long been common practice in Computer Science; a familiar example to many will be the simplification of complex vehicle routing problems into the Travelling Salesperson Problem.

In this article, we aim to contribute to an understanding of how and when this principled simplification is valuable, in terms of supporting explainability of complex evolutionary agent-based models. To do this, we explore a simulation environment that is well established in the ALife community, the River Crossing Task (RCT) (Robinson et al., 2007). The RCT was originally developed to explore how agents learn to solve increasingly complex tasks; initially this was done using a novel neural network architecture that separated reactive from deliberative processes. Since then, the testbed has been extended to explore the effect of social learning strategies (Jolley et al., 2016), learning by imitation (Borg et al., 2011), and social action (Barnes et al., 2019), as well as how agents learn in a 3D world (Stanton &

Channon, 2015). One of the difficult features of the RCT and its extensions is that agents must learn sub-tasks in order to achieve their goal. Specifically, while the primary goal is to collect resources in a 2D environment, they must first learn to pick up a stone and then drop it in a river, in order to 'build a bridge' and be able to cross the river safely. This enables them to access all the resource objects, some of which are on the opposite side of the river. They must learn this without any prior knowledge of the task or environment.

In this article, we explore the RCT using the method of principled simplification, in order to generate new understanding of why learning this sub-task is so difficult for evolutionary agents. Specifically, we analyse the Minimal River Crossing (RC-) Task (Ghouri et al., 2020), exploring how and when insights obtained in that analysis also apply to the original RCT from which it is derived. We demonstrate that this aids the explainability of the RCT problem, enabling predictions to be made about how or why agents behave in the way that they do. We also demonstrate that these predictions may also not always be valid, and explore when this is the case. We find that principled simplification can support explanations when structural features of the original problem are retained in the simplified version. However, we further find that when simplification reduces quantitative features of the problem, this leads to predictions based on those quantitative features being invalid.

This article builds on our prior work (Ghouri et al., 2020), in which we first presented the Minimal River Crossing (RC-) Task. It extends this work in four ways: i) an expanded set of experimental results on our original experiments, which provide confirmation of and greater confidence in the original results presented by Ghouri et al. (2020); ii) a new set of experiments comparing evolutionary, random, and hybrid search; iii) corresponding new experiments in the RCT, in order to establish when the RC- results do in fact generalise; and finally iv) new insights into when generalisations such as those studied in this paper are possible.

Therefore, the contributions of this article are twofold: first, we present new insight into why the RCT is challenging for evolutionary algorithms to solve, and in doing so explain observed behaviour of evolutionary agents in the problem; second, we present new insight into the use of principled simplification as a method of explaining complex models, including what to look for in determining whether predictions and explanations may be valid or not. This, in turn, leads to suggestions for further exploration of the method.

2 Explainability of Complex Agent Based Models

From humble beginnings (Schelling, 1971), agent-based modelling is now a serious and dominant method of generating understanding about possible emergent outcomes of complex biological, physical, social, and economic systems (Helbing, 2012). To cite some examples, agent-based modelling has been used to study strategies to tackle climate change (Robalino & Lempert, 2000), by geographers to understand spatial systems (Torrens, 2010), by political scientists to study nation state formation (Cederman, 2002), and quite broadly to explore questions around economies (Farmer & Foley, 2009). Agent-based modelling has also been used extensively in the study of COVID-19 spread, particularly to assess the possible impacts of diferent policies. Maziarz and Zach (2020) assess this as a methodology for that purpose, noting the value that ABMs provide to epidemiology. They further note that this is due to the fact that *"although epidemiological ABMs involve simplifications of various sorts, the key characteristics of social interactions and the spread of SARS-CoV-2 are represented sufficiently accurately."* (Maziarz & Zach, 2020).

In many agent-based models, evolution or other learning techniques are used in order to model incentive-driven behaviour, survivability given limited resources, or population dynamics. Citing three very different examples in order to convey the breadth of topics to which evolutionary agent-based modelling have been applied, Barbosa et al. (2011) use the method to study human migration, Eldridge and Kiefer (2018) to explore the evolution of soundscapes, and Ma and Nakamori (2005) to model technological innovation. Powers et al. (2018) explore methodological issues, comparing evolutionary agent-based models with equation-based evolutionary game theory, arguing for a complementary role for both in answering questions around complex social systems.

Many valuable insights have been gleaned from the use of agent-based models, yet they still remain poorly understood[1] and difficult to explain. In part, this comes from the result of complex interactions between individual agents and the chance of emergent outcomes, as well as the underlying black-box nature of many of the learning algorithms used inside individual agents. Taken together, this makes evolutionary agent-based models notoriously difficult to understand, despite their obvious value. And the ability to understand evolutionary agent-based models is not merely a 'nice to have', as Andras et al. (2018) note, if people do not understand complex computational models, they are less likely to find them trustworthy.

[1]The validity, interpretation, and use of agent-based simulation models for the study of the spread of COVID-19 has been a hot topic in online modelling communities.

In this article, we aim to contribute towards methods for the analysis and explainability of evolutionary agent-based models. More specifically, we are interested in the extent to which principled simplification of a model may allow us to generate insights that can be generalised back to more complex versions of the original model. In order to do this, we explore the specific and well-studied example of the River Crossing Task, using this as a case study.

2.1 The River Crossing Task

The original River Crossing Task (RCT) was developed to investigate how agents can evolve to solve complex tasks using both reactive and deliberative behaviours in a dynamic environment (Robinson et al., 2007). The RCT environment is a 2D grid, with a river of Water separating an agent from its target 'Resource'; the agent must learn appropriate behaviours to cross the river safely, so that it can collect this Resource object and thus achieve its goal. Various other objects exist within the environment: Stones can be picked up and placed in the river to build a bridge, acting as a safe passage across the river; Traps are dangerous items that will kill the agent if stepped upon, giving strongly negative fitness; all 'empty' cells represent Grass, which are safe to pass over. Agents must first learn to avoid the river, otherwise they will drown and again receive a strongly negative fitness. By exploring the fitness landscape, they can also learn that Stones can be used to build bridges; building a bridge itself does not provide any reward, but it does enable agents to reach their target object within the environment - the Resource.

The River Crossing Task can be varied, for example by altering the presence or number of Traps, Resources, and Stones, and changing the width and depth (in terms of number of Stones required in one Water cell to build a bridge) of the river. This means that the RCT is usually considered to be a family of environments, with common learning challenges and tunable difficulty. One learning challenge common across the family, and observed by Robinson et al. (2007), is that it is particularly difficult for agents to evolve successful goal-achieving behaviours when they have to learn sub-goals first (i.e. building a bridge with a Stone), before the possibility of receiving any positive fitness became accessible. They observed that the difficulty of this problem therefore led to evolution resembling random search. To avoid this, agents in Robinson et. al's agents were consequently evaluated on three maps of increasing difficulty – starting with an environment where a bridge was already

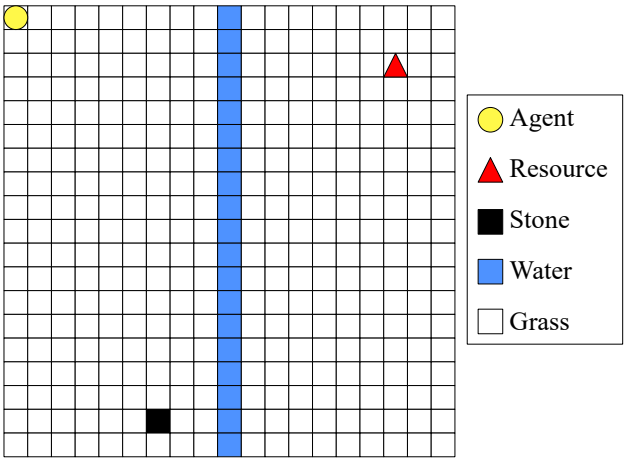


Figure 1: A simplified RCT environment inspired by Robinson et al. (2007), adapted from Barnes et al. (2019). The goal of the agent is to collect the Resource object on the opposite side of the river; to do this, it must learn to build a bridge in the river with a Stone.

built in the river, allowing the agents safe passage to the Resource. This gave agents the opportunity to discover that retrieving the Resource is linked to a high fitness, without the need for learning the sub-tasks of building the bridge first; this involves the learning of risky behaviour as interacting with the river has the potential to lead to a negative fitness.

In this article, we focus on an instance of the RCT family of environments on a 19×19 grid, containing a one-cell wide river of Water in the centre, with one Stone on the same side of the river as the agent, and one Resource on the opposite side. These are the minimum set of objects required for an agent to achieve their goal, and, since agents must first learn to complete sub-tasks before achieving any fitness at all (i.e. building a bridge), to capture the common learning challenge identified by Robinson et al. (2007). Robinson et al. already demonstrate that the ability for agents to exhibit reactive and deliberative behaviours is beneficial when navigating dynamic environments; the RCT configuration shown in Figure 1 is therefore static, and does not vary between the experiments in this study. Further, as one of the primary motivations for this article is to aim for explainable agents through principled simplification, other original RCT objects that are irrelevant for achieving the goal – like Traps, and multiple Stones (Robinson et al., 2007) – are not used. It is also worth noting that while the size of the world impacts the number of moves an agent takes to achieve the goal, it does not change the size of the search space.

Agents use a neural network learning architecture introduced by Barnes et al. (2019), which is a simplified version of other RCT agent learning architectures proposed by Robinson

et al. (2007) and Borg et al. (2011). The first of the two artificial neural networks in this learning architecture is a feed-forward *deliberative* network that determines target locations to move to. The second, *reactive* network is inspired by the shunting equation proposed by Yang and Meng (2000), which deals with navigation towards target locations; this works by overlaying activation values on cells on the grid, centred on a highly positive value at a target location (e.g. a Stone) and which decay in all directions, providing a hill for the agent to climb towards its desired location. Activations can also be highly negative, indicating that the location is to be avoided (e.g. a successful agent may learn to avoid the river unless it has a Stone), or 0, indicating no preference about that location. Further, the shunting equation proposed by Yang and Meng (2000), which allows activity to propagate through the second reactive network in this architecture, only propagates positive activity; negative activity thus does not propagate. In our implementation of the RCT, if the activation landscape is flat, indicating the absence of any goal, the agent does not move[2]. The fitness f for agent A is calculated with Equation 1:

$$A_f = r - w - (c \times \frac{m}{2T}) \tag{1}$$

where r is equal to the number of Resources collected, $w = 1$ if the agent falls into the Water and 0 otherwise, $c = 1$ if there is a cost to movement and 0 otherwise, and m is the number of moves the agent has made. T is a constant representing the total number of moves allowed, where $T = 500$; an agent will fail the task if it falls into the river, or makes the maximum number of moves. By dividing the number of moves by twice the total number of allowed moves for the cost to movement, agents that use the maximum number of moves whilst still achieving the goal ($A_f = 0.5$) will receive a better fitness than those that are Neophobic[3] and do not achieve the goal ($A_f <= 0.0$); successful agents that move excessively are still rewarded, but receive a lower fitness than those that achieve their goal in fewer moves.

[2]Interestingly, whether or not an agent remains stationary or makes a random move in this situation is a detail omitted from Robinson et al. (2007). We discuss the implications of this decision in Section 5.2.

[3]We use the term 'Neophobic' in this paper to refer to agents that evolve to avoid new situations and do not explore the environment. Similarly, we use the terms 'Builders' to refer to agents that successfully build a bridge, and 'Dead' to refer to agents that fall in the river, respectively.

2.2 What Can We Learn from River Crossing Tasks?

Whilst the RCT in design is simple, the task involved is complex for agents to learn to solve – agents must learn complete sub-tasks before any positive fitness becomes accessible. One of the main contributions of this work was an agent learning architecture that could express both reactive and deliberative behaviours; agents could thus navigate dynamic environments with ease without the need for route planning mechanisms, whilst also being able to solve novel and more complex versions of the task when shown them for the first time. An important observation though is that agents could not achieve the task with evolutionary search without first being shown how to achieve the goal (i.e. collecting the Resource), as there was otherwise no incentive to learn to build a bridge.

Later, extensions to the testbed, such as the RC+ task, are designed to increase in complexity such that the final environment cannot be solved by incremental evolution on its own (Borg et al., 2011). Borg et al. evaluated agents on five environments at each generation instead of Robinson et al.'s three, where agents had to exhibit different behaviours to achieve their goal in the final environment, as no Stones were present to build a bridge. As a result, Borg et al. (2011) demonstrated that learning by imitation through transcription errors and cultural transmission can enable agents to achieve goals where incremental evolution cannot; Jolley et al. (2016) showed this was also possible by employing teacher-learner social learning strategies. Borg et al. (2011), and later Jolley et al. (2016), theorised that the resulting fitness landscape had two peaks (the former being for building a bridge to achieve the goal, and the latter relating to solving the final task) – and that the valley between could not be bridged by agents that just learnt via incremental evolution alone. Drawing on the observation of Robinson et al. (2007), that building a bridge in the RCT without first being exposed to the Resource leads to random search, we posit that a similar fitness landscape exists for building a bridge, thus making this a difficult task for evolution to solve.

Other instances in the RCT family include the 3D River Crossing (3D RC) Task and the River Crossing Dilemma (RCD). Stanton and Channon (2015) demonstrated that the RCT task could be extended such that 3D, rigid-body virtual agents could be evolved in a 3D version of the RCT (3D RC Task); agents employed a hybrid neural architecture to express reactive and deliberative behaviours that enabled navigation and interaction with the environment. Further, Barnes et al. (2019) explored how the pursuit of individual

goals was affected when agents coevolved in a multi-agent, gamified version of the task called the River Crossing Dilemma (RCD), in which agents shared an environment with an unknown other. Additionally, a simple technique inspired by the theory of social action was shown to mitigate evolutionary volatility arising from actions of other agents within the environment, without the need for agents to be aware of the existence of others; this unintended influence should be considered when designing systems that are colocated or operate in shared environments.

A common factor in the RCT family of environments is that the seemingly simple act of building a bridge involves learning two implicit things: adapting learnt knowledge that the river is 'safe' *if* a stone is being carried, and that the neutral behaviour of putting a stone in the river is beneficial in the long term. An agent therefore depends on learning this sub-task of building a bridge first in order to achieve its goal, and must endure a period of low fitness during evolution in order to find the optimal solution.

Task decomposition has been shown to be powerful when using a neuroevolutionary approach for learning sequential sub-tasks (Jain et al., 2012), and when using modular connectionist architectures (Jacobs et al., 1991). Nicolay et al. (2014) conclude that, when training a robot to learn two conflicting tasks, learning the 'harder' task before learning both simultaneously is more beneficial than learning both together initially. One issue with current approaches to solving the RCT is that agents use neuroevolution to evolve a combination of two neural networks for reactive and deliberative behaviours (Robinson et al., 2007). As complexity of the task and the sub-tasks involved increases, so does the learning architecture required; this means that the search space also increases, and evolving optimal solutions is more difficult (Federici & Downing, 2006).

While the learning and interaction techniques explored in this literature (e.g., task decomposition, evolving for modularity, social learning) may be beneficial for designing agents to solve the RCT and its variants well, it is also clear that studying the RCT and its variants support the generation of general insights into how evolutionary agents behave in a variety of different scenarios. Our focus in this article is in line with this latter aim: rather than trying to find *good solutions* for the RCT task, we are interested in exploring and explaining *why the task is difficult* for agents to learn at its core.

As a first step in this study, we therefore require an equivalent problem with reduced complexity, paired with a simplified agent representation and learning architecture.

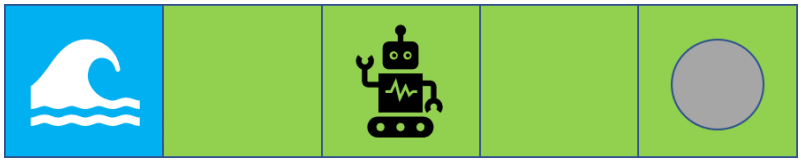


Figure 2: The Minimal River Crossing Task Testbed: a 1×5 grid-world environment with water in the leftmost cell, and grass in all others. There is a stone in the rightmost cell, and an agent in the center cell.

3 The Minimal River Crossing Task Testbed

The Minimal River Crossing (RC-) Task testbed, originally proposed by Ghouri et al. (2020), is a principled simplification of the original River Crossing Task (RCT) (Robinson et al., 2007) and its extensions. The (RC-) is designed to reduce the complexity of the original River Crossing Task while keeping its core properties, such that agent evolution can be studied in closer detail. The intention was to support increased explainability of agent evolution in the RCT, by benefitting from the simpler environment in order to facilitate analysis of the fundamental learning tasks. In this article, we explore the extent to which this proves to be the case.

In order to do this, we explore three questions in the RCT/RC- in this article. First, what effect does a cost to movement (c in Equations 1 and 2) have on agent evolution? Second, what effect does the introduction of a population have on agent evolution, compared with a $1 + 1$ evolutionary algorithm? Third, what impact will the introduction of Random Immigrants (Cobb & Grefenstette, 1993) (typically used to generate increased diversity) have on agent evolution? We use these questions as examples of the sorts of concepts that can be explored in greater detail using this method.

The RC- testbed, depicted in Figure 2, consists of a 1×5 grid, containing Water in the leftmost cell, a Stone in the rightmost cell, and Grass in all others (including underneath the Stone). The goal of the agent, starting in the centre cell, is to build a bridge by first picking up the Stone, and then heading to the Water to build a bridge. In the original RCT environment, it is this act of building a bridge that allows an agent safe passage across the river in order to collect the reward object; thus the act of collecting the reward object is not accessible unless a bridge is built. The RC- therefore simplifies the agent's reward state by only awarding a strongly positive fitness when it has built a bridge. An agent dies if it steps into the Water without carrying a Stone, and receives a strongly negative fitness. Another

simplification present in the RC- is a reduction in the existing objects in the environment to only those that are necessary to achieve the goal; excess Stones, other objects such as Traps, and Resource objects, which can all be seen in other variations of the RCT, are therefore not used in the RC- environment.

3.1 Agent Representation in the RC-

An agent's genome is represented using a pair of integers (R, L), each in the range $[0, 4]$, where R is the number of moves to the right it will take, and L is the subsequent number of moves to the left. Further, a strategy of $(1R, 2L, 3R)$ for example can be simplified to $(2, 0)$; this removes unnecessary complexity from the genome representation, but also means that the genome represents the outcomes of more complex movement strategies in a simple manner. As a result, there are a total of 25 possible agent solutions between $(0, 0)$ and $(4, 4)$, with 12 being valid (e.g. $(1, 2)$) and 13 being invalid (e.g. $(3, 2)$); from the starting position, an agent can take a maximum of two moves to the right, then four to the left, resulting in the optimal solution of $(2, 4)$. The fitness f for agent A is calculated with Equation 2:

$$A_f = Is - Iw - (c \times m) \tag{2}$$

where $s = 1$ if a bridge is built successfully with a Stone and 0 otherwise, $w = 1$ if the agent falls into the Water and 0 otherwise, $c = 1$ if there is a cost to movement or 0 otherwise, and m is the number of moves taken by the agent. I is a constant where $I = 10$, indicating the magnitude of the reward or penalty. Agents can thus be characterised as the Builders (achieve the goal), the Dead (receive a penalty for drowning), and the Neophobic (do nothing).

3.2 Agent Evolution in the RC-

All the experiments in this study evolve agents using an evolutionary algorithm (EA) with the following common parameters. Agents evolve for 100 generations, and experiments are repeated for 100 independent runs; each experiment is also repeated with and without a cost to movement applied to agents (i.e. $c = 0$ and $c = 1$ in Equations 1 and 2), to explore how this impacts evolutionary learning. Agents have no prior knowledge of the task or environment, and are initialised with zero knowledge at the start of evolution. In the (RC-), we operationalise this by initialising all agents to $(0, 0)$; as we will see later, in the RCT this

is typically done by initialising the neural network with uniform random weights. At each generation, the agent in the population with highest fitness is selected to be the 'parent'; tournament size is therefore equal to the population size. Once a parent is selected, an offspring is generated from this single parent by mutation; no crossover occurs.

The mutation operator applied to an agent's genome, (R, L), is as follows. Chromosomes are treated 'piecewise', in the sense that each allele (R and L) is treated independently of the other. Each allele has a 50% chance to mutate, with an equal chance of the mutation being $+x$ or $-x$: the percentage chance of $x = 1$ is 60%, $x = 2$ is 25%, $x = 3$ is 10% and $x = 4$ is 5%. This means that there is a high probability that the offspring is mutated by a small amount, but there is still a very small probability of being mutated by a large amount to enable agents to escape local optima and explore other areas of the solution space. As the search space here is very limited, a fixed percentage chance of mutation is deemed a suitable representation of other common mutation operators – say for example values chosen from a Gaussian distribution in the context of large search spaces. If an invalid solution is generated, new offsprings are created until one is valid. This mutated offspring then replaces the worst performing agent in the population, unless otherwise specified.

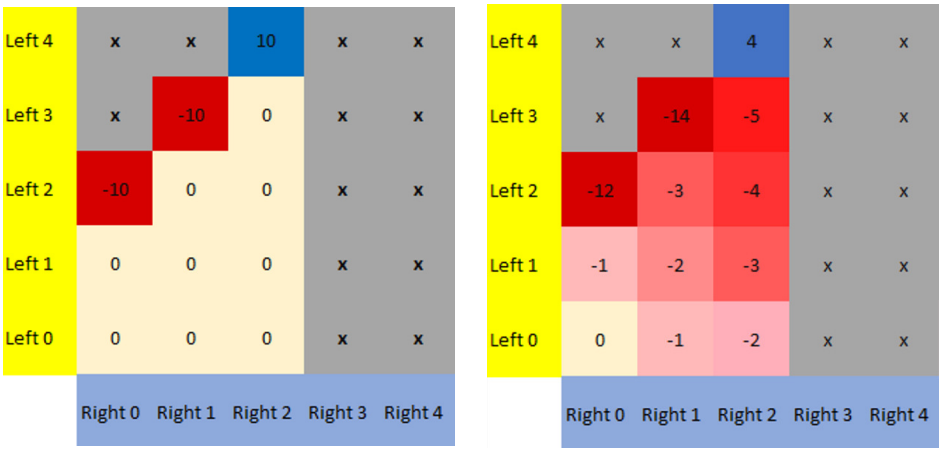
4 Understanding the RC-

This section presents results, both analytical and experimental, concerning the RC-. In doing so, we present insights into the workings of evolutionary processes on agents solving it.

4.1 Landscape Analysis

As an agent's genome is represented using a pair of integers, each in the range $[0, 4]$, the fitness landscape for the RC- environment can be fully mapped with a 5×5 grid. Figures 3a and 3b show the fitness landscapes for the RC- environment, without and with a cost for movement respectively. Out of the 25 possible agent solutions, 12 are valid and 13 are invalid (i.e., agents can make at most two moves to the right, and the number of legal moves to the left is bounded by how many moves to the right they have made first). As the agent has no prior knowledge of the environment or its location within it, restricting the search space to 3×5 would provide additional information and bias the problem.

Without a cost to movement (i.e., $c = 0$), Figure 3a shows that a neutral landscape



(a) *Without* cost to movement.

(b) *With* cost to movement.

Figure 3: Fitness landscapes of all 25 possible agent solutions in the RC- environment, where there is (a) no cost to movement, or (b) a cost of -1 per move. The 13 invalid solutions are indicated with an 'x', whereas the 12 valid solutions show the fitness received if the agent makes the specified moves to the right then left. Agents receive a fitness of $+10$ by moving 2 moves to the right to pick up a Stone, then 4 left to build a bridge in the water. If an agent moves into the Water without first picking up a Stone, it will receive a fitness of -10.

is created; agents can explore without penalty, and there is no indication of proximity to the optimal solution. However, when a cost to movement is introduced ($c = 1$), Figure 3b shows that a trap function is created in the landscape (Nijssen & Bäck, 2003). This means that agents receive a lower fitness as they get closer to the optimum, making it difficult for evolution to find.

4.2 Experimental Setup

The first set of experiments explore the effect that a cost to movement has on evolution, using a 1+1 evolutionary algorithm (EA). The 1+1 EA has been shown to be simple yet effective for simple search problems (Droste et al., 2002; Nijssen & Bäck, 2003), which makes it an appropriate choice for agent evolution in the RC- due to its small search space. As the population size is 1, the EA described in Section 3.2 is adapted such that the single agent in the population will only be replaced if the fitness of the offspring is greater than, or equal to, its own fitness; the EA would in fact instead be a random walk if the offspring always replaced the single agent in the population.

The second set of experiments increase the population size in the EA to 5, to explore the effect of this change on evolution, and then replaces the agent with the lowest fitness (as described in Section 3.2). All ties (in terms of fitness) are broken randomly.

The third set of experiments also use a population size of 5, to explore the effect of Random Immigrants on evolution and learning. Random Immigrants inject randomness into the population, leading to greater diversity (Cobb & Grefenstette, 1993) and the chance to escape local optima. We explore the effect that different probabilities of introducing Random Immigrants into the population has on how agents are able to achieve their goals in the RC- when agents are and are not subjected to a cost to movement. At each generation, the worst-performing agent in the population is either replaced by an offspring of the best, or a Random Immigrant; these experiments are repeated for different levels of Random Immigrants: from 0.1 probability of generating a Random Immigrant instead of an offspring by mutation, in 0.1 intervals, up to 1.0 probability (which resembles random search). These are compared against the results from the second set of experiments, which provide a baseline of 0.0 probability of Random Immigrants. We can therefore compare the behaviour of traditional evolutionary search to random search, as well as the effect that increasing levels of randomness in the EA has on evolution and goal-achievement.

4.3 Cost to Movement and Evolution

When there is no cost to movement, agents encounter a neutral landscape (Figure 3a) in which they can explore without penalty; there is no explicit incentive to exploration, but more importantly there is no inherent *disincentive* to exploration. Figure 4 depicts the evolution of agents evolving with a 1+1 EA, when they are either subjected to a cost to movement or not. 92% of agents evolve to be Builders (those that achieve the goal) when there is no cost to movement and thus are able to explore a neutral fitness landscape (Table 1). In contrast, adding a cost of -1 per move is shown to have a dramatic effect on evolution, as only 10% of agents evolve to be Builders under these conditions. A cost to movement creates a trap in the fitness landscape (Figure 3b), as agents must endure negative fitness in order to evolve a solution that is capable of achieving the goal. Agents are therefore found to be discouraged from exploring when a cost to movement is present, and are encouraged to be Neophobic.

4.4 Population Size and Evolution

Here, we increase the population size from 1, as seen in the 1+1 EA results in the previous section, to 5, to explore the impact that a population size has on evolution in the RC-. As

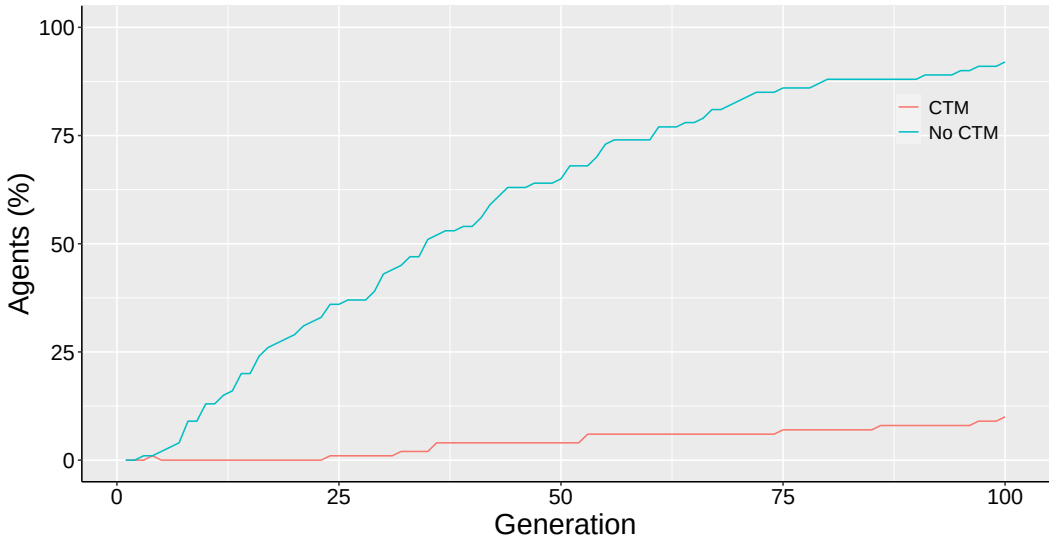


Figure 4: The percentage of runs in which a 'Builder' solution has been found at each generation when agents evolve with a 1+1 EA in the RC- (agents that build a bridge and thus achieve their goal), with and without a cost to movement (CTM). When there is no cost to movement, agents are able to explore the search space without penalty and thus more 'Builder' solutions are found, compared to when there is a trap function in the search space created by a cost to movement.

with the previous set of experiments, no cost to movement means that agents are able to explore without penalty; as a result, more Builder solutions are evolved when there is a neutral fitness landscape than when a cost to movement creates a trap (Figure 5). At the end of evolution, 73% of agents evolve to be Builders with no cost to movement, whereas 14% are Builders when the fitness landscape contains a trap (Table 1). This result is consistent with the 1+1 result, demonstrating that the presence of either a trap function or neutral landscape has the same effect, when the algorithm uses a population. When there is a cost to movement and therefore a trap in the fitness landscape, an increase in population size from 1 to 5 neither helps nor hinders the evolution of Builder solutions; however, we observe that by increasing the population size, fewer agents evolve to be Builders when there is a neutral

Table 1: The percentage of runs in which a 'Builder' solution has been found after 100 generations in the RC-.

Pop Size	CTM	Builders (%)
1	No	92
1	Yes	10
5	No	73
5	Yes	14

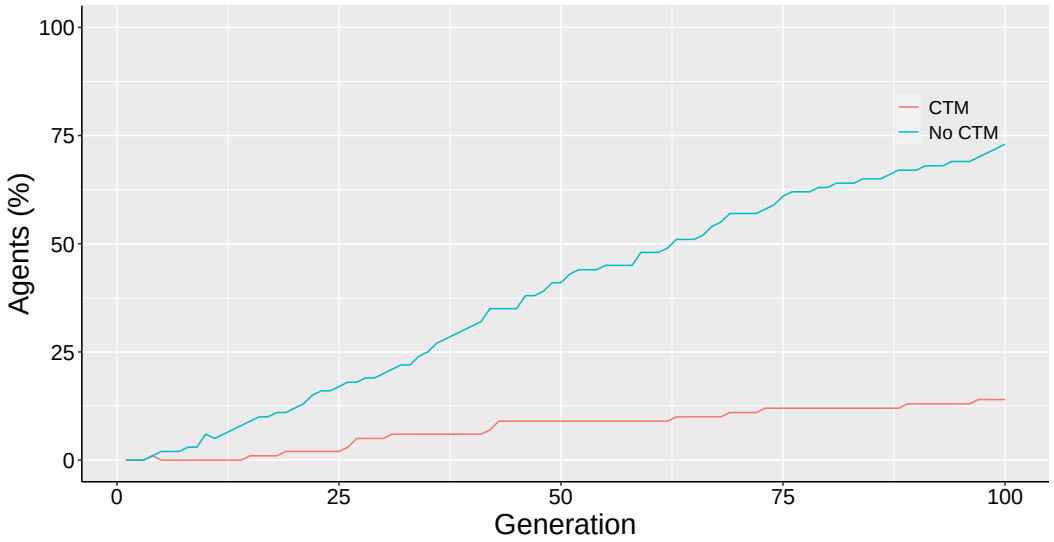


Figure 5: The percentage of runs in which a 'Builder' solution has been found at each generation in the RC- (agents that build a bridge and thus achieve their goal) with a population size of 5 at each generation, with and without a cost to movement (CTM). A trap function is created when there is a cost to movement, making it harder for 'Builder' solutions to evolve compared to when there is a neutral landscape (from no cost to movement).

fitness landscape. As the replacement operator is changed from '*replace with offspring only if it has better fitness*' to '*always replace worst agent with offspring*' when the population size is increased from 1 to 5, one would expect this to be the cause of the slight decrease in Builder solutions that are found, rather than an effect of an increase in the population size.

4.5 Random Immigrants and Evolution

Injecting randomness into a population via Random Immigrants during evolution increases the diversity and can aid the ability of the population to traverse a search space (Cobb & Grefenstette, 1993). A small proportion (e.g. 0.1) of random immigrants is often sufficient to enable populations to escape local optima or traverse challenging fitness landscapes, and selecting this proportion is typically an important parameter choice in designing an effective evolutionary algorithm. Here, this parameter is then used as a probability when a new offspring is generated: with this probability, instead of producing an offspring by evolutionary operators (crossover and mutation based on the parent), we introduce an individual with a uniformly randomly initialised genome. As we increase the proportion of random immigrants, the algorithm tends towards resembling random search, and at the extreme where the proportion of offspring generated as random immigrants instead of by evolution-

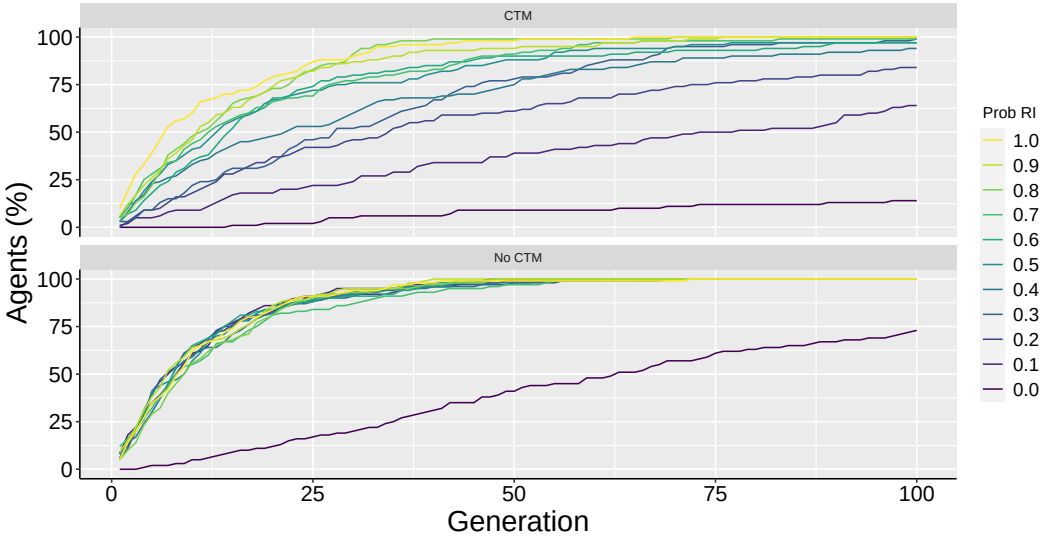


Figure 6: The percentage of runs in which a 'Builder' solution has been found at each generation in the RC- (agents that build a bridge and thus achieve their goal) with a population size of 5, with and without a cost to movement (CTM), and increasing probabilities of introducing a Random Immigrant (RI) at each generation. When there is a cost to movement, and hence a trap function, increased randomness improves the ability of the search to find 'Builder' solutions; indeed random search outperforms evolutionary search in this case. When there is no cost to movement, and hence a neutral landscape, any non-zero quantity of Random Immigrants is sufficient to improve the search process; varying the proportion of Random Immigrants above 0.1 has no further effect.

ary operators is 1.0, the algorithm indeed becomes random search. Since we now know that evolutionary agents in the RC- are faced with either a trap function or a neutral landscape (in the cases of a cost to movement or absence of cost to movement, respectively), here we build on the population-based experiments in Section 4.4, to explore the impact of adding increasing levels of randomness to a population-based search.

The experiments in this section are designed to explore the effect that an increasing probability of injecting Random Immigrants into the population has on evolution and thus the ability to achieve goals in the RC-.

Figure 6 shows these results for agents that evolve with and without a cost to movement. It is immediately obvious that any amount of randomness that is injected into the population is beneficial for evolution in the RC-; the search space is small, with only 12 possible valid solutions, which makes a random jump to a different part of the search space likely to have a positive outcome. This effect is more prominent when agents face a neutral fitness landscape, as random jumps may place the agents closer to the optimum or at the optimum itself. With a cost to movement however, any diversity added by a Random Immigrant must

Table 2: The percentage of runs in which a 'Builder' solution is evolved after 100 generations in the RC-.

CTM	Probability of Random Immigrants ($P(RI)$)										
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
No	73	100	100	100	100	100	100	100	100	100	100
Yes	14	64	84	99	94	97	97	99	100	100	100

be exactly the optimal solution to have any sustained benefit: the closer a solution is in the fitness landscape to the optimum, the worse the fitness is, and thus the solution will be selected out of the population at the next generation. So, while any probability of Random Immigrants has an immediate effect on the evolution of Builders when there is a fitness landscape, this benefit is more gradual when there is a cost to movement, which increases as the probability of introducing a Random Immigrant at each generation increases. This is because there are more chances for a Random Immigrant to 'jump' to the optimum as the probability increases. With a cost to movement, the largest benefits from Random Immigrants are seen with $P(RI) >= 0.8$, as 100% of agents achieve the goal at the end of evolution (Table 2); this is in contrast to a neutral fitness landscape, in which case 73% of agents achieve the goal with traditional evolutionary search, which rises to 100% of agents when any non-zero proportion of Random Immigrants are introduced.

5 Generalising to the Original RCT Problem

The results in Section 4 are interesting in that they provide insight into the structure of the RC- problem environment, and how evolution operates in the RC-. However, our stated aim in producing the RC- is to support the generation of understanding that generalises back to the original RCT from which it was inspired, especially where this understanding would have been difficult to obtain by analysing the original (more complex) problem. An important and related broader question is: can the approach of principled simplification, as we have explored in the case study in this paper, provide such insight, and if so, when? Are there features that can be readily and reliably explored and generalised using this technique, and are there features that are not amenable to analysis using this method?

In this Section, we explore attempts to generalise results from the experiments from Section 4 on the RC- back to the base RCT environment (Figure 1). We note where this is successful and where it is not, discuss possible reasons, and ask: what might we learn from

this?

5.1 Experimental Setup for the RCT

As the search space for agents in the RCT is much larger than in the RC-, the evolutionary algorithm described in Section 3.2 is adapted slightly to accommodate for this change in scale. Agents in the RCT are evolved for 10,000 generations, with each experiment being repeated 100 times with and without a cost to movement. The total cost to movement incurred by an agent is defined in Equation 1; agents may elect to not move, in which case their fitness will be 0.0 – but they will not achieve their goal. Agents are initialised with random weights in the deliberative neural network, and have no prior knowledge of the task or environment at the beginning of evolution. As with the EA design for agents evolving in the RC-, tournament size equals the population size. The individual with the highest fitness is selected to be the parent at each generation, from which an offspring is generated by mutating the weights in the deliberative neural network by a random value from a Gaussian distribution with $\mu = weight$ and $\sigma = 0.01$; crossover does not occur. The worst performing agent in the population is replaced by this mutated offspring, inline with the RC- EA setup described in Section 3.2 – unless the population size is 1, in which case the worst agent is only replaced if the fitness of the offspring is greater than or equal to its own fitness (inline with Section 4.2). The details of the experimental setup for the RCT are the same as for the RC-, described in Section 4.2, so we can ascertain whether results in the RC- can actually be generalised back to the RCT.

5.2 Effect of a Cost to Movement

When agents are evolved with a 1+1 EA in the RC-, as explored in Section 4.3, agents are found to encounter a neutral fitness landscape with no inherent disincentive to exploration when there is no cost to movement; as a result, a much higher percentage of agents are able to achieve their goal when there is no cost to movement compared to when there is a cost to movement. This cost to movement creates a trap in the fitness landscape: the fitness an agent receives gets lower as they approach the optimum; this makes it difficult for successful Builder solutions to evolve because selection pressure in the evolutionary algorithm favours higher-fitness solutions.

Looking instead at how agents evolve in the RCT with a 1+1 EA, Figure 7 paints a

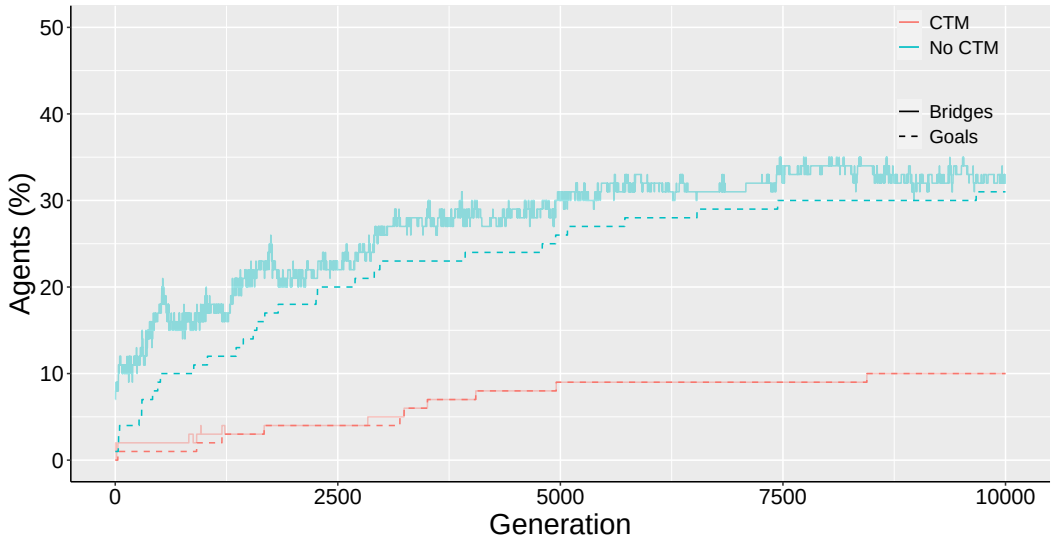


Figure 7: The percentage of runs in which a 'Builder' solution has been found at each generation when agents evolve with a 1+1 EA, with and without a cost to movement (CTM), in the RCT. A higher percentage of runs evolve agents that achieve their goal ('Builders') when there is no cost to movement compared to when a trap function arises through a cost to movement. Further, agents are free to explore the landscape and experiment with behaviours when there is a neutral fitness landscape: agents may evolve to build bridges *without* necessarily learning to achieve their goal when there is no cost to movement, but these two behaviours are strongly linked when there is a trap in the fitness landscape.

similar picture as Figure 4: more agents are able to evolve the behaviours necessary to achieve their goal when there is a neutral fitness landscape as opposed to when a trap exists. When there is no cost to movement and thus a neutral fitness landscape, agents in the RCT may evolve to build a bridge without necessarily learning how to collect the Resource to achieve their goal; this is because there is no negative consequence for exploring new behaviours, and *disincentive* to learning how to build a bridge – a result that was also observed in the RC- (Section 4.3). Over time, agents may then explore the search space enough to also learn to achieve their goal as well. Conversely, the act of building a bridge is closely tied with the ability to achieve goals when there is a cost to movement present; as each move is costly when a trap function is present, agents learn to only build a bridge when there is a reward. Bridge-building is consequently difficult to learn, because agents must endure a period of low fitness in order to discover the link between bridge-building (low fitness from number of moves), and retrieving the Resource to achieve the goal (positive reward that is only accessible through building a bridge).

Whilst fewer agents overall are able to achieve their goal in the RCT than the RC-, due

Table 3: The percentage of runs in which a 'Builder' solution has been found after 100 generations in the RC-, or 10,000 generations in the RCT.

Experiment	Pop Size	CTM	Builders (%)
RC-	1	No	92
	1	Yes	10
	5	No	73
	5	Yes	14
RCT	1	No	31
	1	Yes	10
	5	No	26
	5	Yes	8

to the increase in problem complexity and size of the search space, the same trend is seen: a higher percentage of agents are able to achieve their goal when a neutral fitness landscape is experienced, compared to when a trap exists as a result of a cost to movement penalty. This can be seen more clearly when looking at Table 3: 92% and 31% of agents achieve their goal with no cost to movement in the RC- and RCT respectively, compared to 10% in both the RC- and RCT when a cost to movement is applied.

It is worth noting that as the fitnesses received by agents will differ when there is or is not a cost to movement applied, presenting and discussing the mean fitness for example would be misleading as the fitnesses are not comparable. Instead, we focus on our primary question of interest, whether or not the agents manage to build a bridge and achieve the goal; fitness is simply a means to an end in this respect. We can therefore objectively see that, regardless of fitness, a cost to movement does in fact influence the ability for an agent to evolve the behaviours necessary to achieve their goal in both the RC- and the RCT, and that by capturing the structural nature of the landscape that causes this, the analysis of the RC- enables us to predict how a cost to movement would affect evolution in the RCT.

Interestingly, in their original presentation of the RCT, Robinson et al. (2007) omit details concerning how an agent behaves when there is no target location given to the shunting network (i.e. the activation landscape is flat). This would be the case for example when the deliberative neural network produces values of 0 for all locations in the landscape, indicating that the agent has no goal to move towards or away from any of them. Yang and Meng (2000) do not discuss this either, since the focus of their work is in how to achieve navigational goals, rather than what to do in their absence. Here, we have designed agents such that if there is no activation signal generated by the deliberative network, the agent does not move, as it has no goal.

An alternative, which also appears reasonable, would be to have the agent move to a random adjacent cell, in the absence of any activation signal. This means that when it had no goal, it would explore the environment in a random walk. At first glance, this may seem an implementation detail of little consequence, but on closer inspection we can see that this has a profound impact on the form of the problem to be solved. Specifically, in the case of there being a cost to movement, this means that every agent would always move at every time point, thereby making m in Equation 1 a constant. This would have the effect of *removing* the trap function from the RCT's fitness landscape, since there would be no option to improve overall fitness by staying still. Assuming that the cost to movement is a proxy for energy usage associated with moving, then this also removes the option for the agent to learn to save energy when it has no knowledge of a goal in the environment. In order to also analyse this using the RC-, one can easily construct a variant of Figure 3b but based on a constant value for m in Equation 2; the result would be a neutral landscape once again.

5.3 Increase in Population Size

In the RC-, similar results are observed when the population size is increased from 1 to 5, as discussed in Section 4.4: more Builders are evolved when there is no penalty for exploration compared to when a trap exists in the fitness landscape, regardless of population size.

When agents evolve in the RCT with a population size of 5, Figure 8 shows that the risk of receiving a lower fitness when moving acts as a deterrent to exploration; learning the sub-task of building a bridge is more difficult as a period of low fitness must first be endured, meaning that these low-fitness but potentially successful solutions are more likely to be replaced during evolution when agents endure a cost to movement. Similarly to the results presented in the previous section and in Figure 7, the act of building a bridge is strongly tied with achieving the goal when agents experience a trap in the fitness landscape; conversely, a larger percentage of agents explore and traverse the fitness landscape to build a bridge without achieving the goal when there is no movement penalty applied.

The results obtained in the RC-, presented in Section 4.4 and Figure 5, corroborate the findings in the RCT: a cost to movement affects the ability of agents to achieve their goals, because a trap in the fitness landscape instead encourages agents to be Neophobic. Fewer agents overall achieve their goal in the RCT compared to the RC-, but as alluded to in

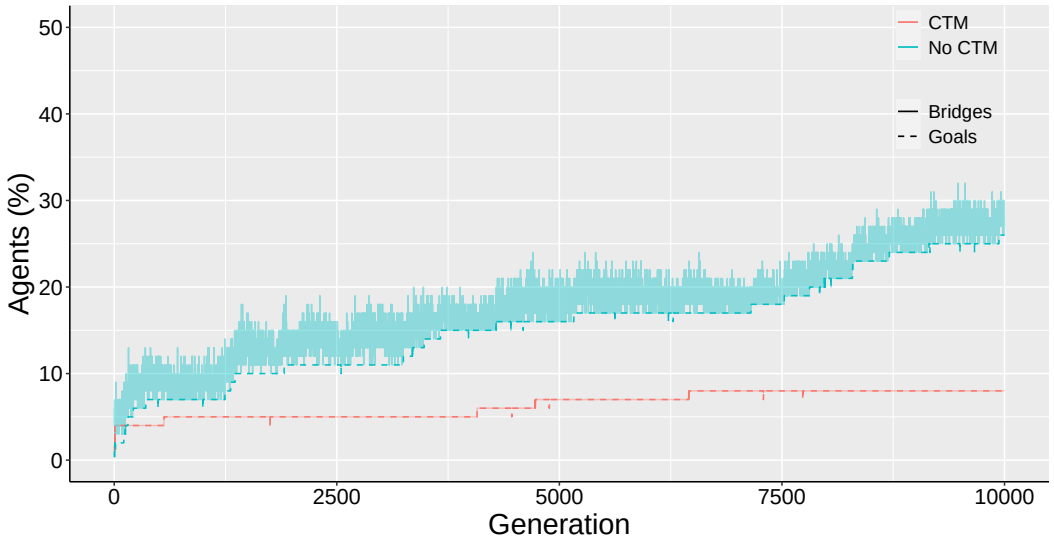


Figure 8: The percentage of runs in which a 'Builder' solution has been found at each generation, when agents evolve with a population size of 5, with and without a cost to movement (CTM), in the RCT. More agents evolve to be 'Builders' (those that achieve their goal) when there is no cost to movement, meaning agents can explore a neutral fitness landscape without penalty. A cost to movement creates a trap function, making it hard for 'Builders' to evolve. As agents can explore without penalty in a neutral fitness landscape, agents may learn to build a bridge without learning to achieve the goal; these behaviours are strongly linked when there is a trap in the fitness landscape.

Section 5.2, this can be expected due to the increase in problem complexity and search space size. 73% and 26% of agents achieve their goal when there is a neutral fitness landscape in the RC- and RCT respectively, compared to 14% and 8% with a cost to movement (Table 3). These results align with what was observed in the RC- (Section 4.4): an increase in population size from 1 to 5 is not observed to help or stifle the evolution of Builder solutions when there is a trap in the fitness landscape, whereas fewer agents evolve to be Builders when there is no cost to movement. The experiments conducted in the RC- are therefore seen to predict the outcome of the same experiments conducted in the RCT.

5.4 Introduction of Random Immigrants

Injecting Random Immigrants into the population was found to have an immediate and sustained effect on evolution in the RC-, as discussed in Section 4.5. This effect was more pronounced in agents that were not subjected to a cost to movement, as any non-zero proportion of Random Immigrants was beneficial to evolution and the pursuit of goal-achieving behaviour. However, when a trap was present in the fitness landscape, the number of

Table 4: The percentage of runs in which a 'Builder' solution is evolved after 100 generations in the RC-, or 10,000 generations in the RCT.

Environment	CTM	Probability of Random Immigrants										
		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
RC-	No	73	100	100	100	100	100	100	100	100	100	100
	Yes	14	64	84	99	94	97	97	99	100	100	100
RCT	No	27	28	33	32	30	30	19	26	25	35	30
	Yes	8	8	10	6	7	14	10	6	6	5	8

Builders that evolved increased inline with the probability of Random Immigrants; a cost to movement means that Random Immigrants would only be beneficial if the random solution is at the optimal point in the fitness landscape, and, as the probability of Random Immigrants increases, so does the likelihood that a new, random solution would be introduced at the optimum. In all experiments, the traditional evolutionary algorithm was outperformed by either a hybrid algorithm with any amount of randomness introduced, and pure random search.

These experiments were repeated in the RCT, and the results are presented in Figure 9 There are two common observations throughout this study: fewer agents achieve the goal overall when subjected to a cost to movement than without, as there is a disincentive for exploration; fewer agents achieve their goal in the RCT compared to the same experiments conducted in the RC- due to the increase in problem complexity and search space size. These observations still stand true when analysing the number of Builders that evolve in the RCT when different probabilities of Random Immigrants are introduced. When there is a trap in the fitness landscape caused by a cost to movement, both traditional evolutionary search, random search, and hybrids of the two produce similar numbers of Builders at the end of evolution; only 5-14% of agents evolved to be Builders in these experiments, as seen in Table 4. Interestingly, both traditional evolutionary search and random search evolve 8% of agents to be Builders, showing that there is no real difference made by increasing randomness, in the case when there is a trap. We also found that both evolutionary, random and hybrid search evolved more Builder solutions over time when agents did not experience a cost to movement, and had no disincentive for exploration. Here, some amount of randomness in the population was observed to be beneficial for evolution, however traditional and random search were found to be similar: 27% and 30% of agents evolved to be Builders with traditional and random search respectively.

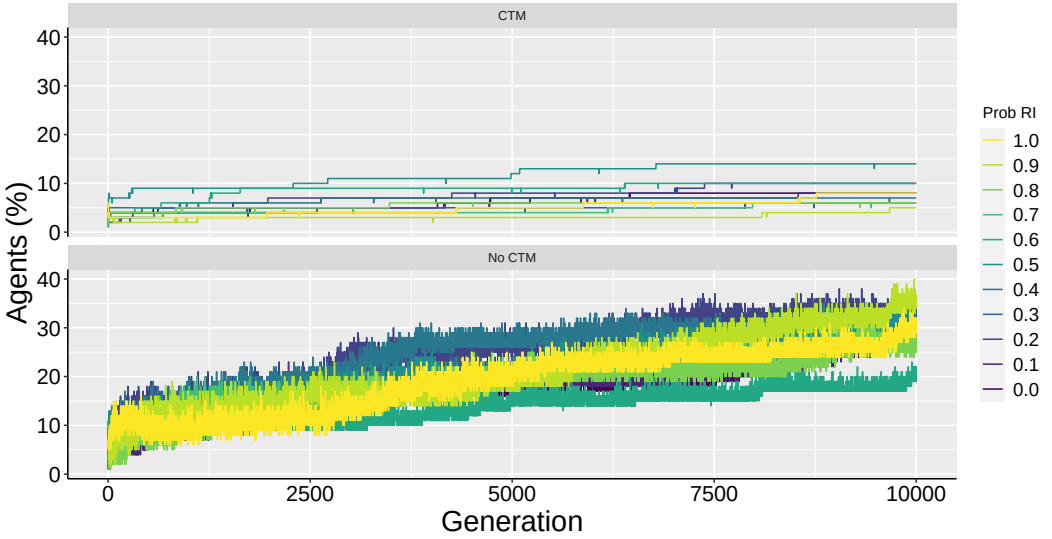


Figure 9: The percentage of runs in which a 'Builder' solution has been found at each generation in the RCT and thus achieve their goal with a population size of 5, with and without a cost to movement (CTM), and increasing probabilities of introducing a Random Immigrant (RI) at each generation. Traditional evolutionary search and random search find a similar number of 'Builder' solutions, both when there is and is not a cost to movement; hybrid search, with any non-zero proportion of Random Immigrants, neither helps nor hinders evolutionary search in general. When agents face a neutral fitness landscape through no cost to movement, more 'Builder' solutions are found than when there is a trap landscape.

There is a divergence from the results of these experiments in the RC- to the results in the RCT for the first time in this experimental study: the RC- cannot predict how evolution will be affected in the RCT, when different probabilities of Random Immigrants are introduced into the population. We see that any non-zero proportion of Random Immigrants is beneficial for agents evolving in the RC- without a cost to movement, but a gradual increase in benefit as the probability of Random Immigrants increases when a cost to movement exists. No such observations were made when agents evolve in the RCT – so what prompted this change? In each experiment, evolution has either nothing to go on, or else is actively pushed away from the goal when a trap exists: structurally, this is the same in both RC- and RCT, in the sense that agents are either in a 'neutral' (no cost to movement) or 'hostile' (cost to movement and thus trap) environment for evolution. However, as the scale changes, the amount of exploration inherent to each of random search and evolution comes into play. Specifically, when the search space is small, the probability of random search finding the goal is increased, compared to in a large search space. Conversely, in a large search space, neither random search nor evolutionary search have any effective 'tactics' to approach the

goal, thus both do roughly as bad as each other: there is no perceptible nor significant difference. In simpler terms, agents in the RCT learn through knowledge encoded in a neural network, however in the RC-, only 12 possible valid solutions exist. This makes it more likely that Random Immigrants in the RC- will fall at the optimal position in the fitness landscape, but is not so straightforward in the RCT. As we approach random search, there is a divergence between the results observed in the RC- and the RCT, due to the quantitative change introduced in the simplification process, and the effect of this on the behaviour of the search process. Specifically, a probability of Random Immigrants of 0.1 - 1.0 in the RCT makes little difference to the number of Builder solutions that are evolved, however the same probabilities in the RC- have a dramatic effect – both with and without a cost to movement – with all probabilities of Random Immigrants producing more Builders than traditional evolutionary search. The reason why we see divergence here, while the results in Sections 5.2 and 5.3 match up to their RC- counterparts presented in Sections 4.3 and 4.4, is because they were based on a qualitative feature of the search landscape, its structure, and the impact of this on searching. Conversely, in this section, the simplification was quantitative in nature, and thus impacted upon the probabilities inherent to each search algorithm.

6 Reflections on Principled Simplification as an Approach to Explainability

We began this article by discussing the approach of simplifying a problem or model in order to aid our understanding of it. The main idea is to create a minimal yet sufficiently useful testbed, that allows us to analyse features of the problem (in this case, when and why it is difficult to solve). In order to do this, the aim is to retain key problem features such that they can be studied, and insights gained on the simpler version can be generalised back to the original problem. Further, such analysis on the simpler variant should be facilitated by this simplicity, which then also provides for clearer explanations. In completing this process, observed results on the original problem can then be explained, using its simpler analogy. As an example, the trap function and neutral landscape in this study are clearly visible in the RC- landscape, visualised in 2D in Figure 3. These figures are valuable tools to understanding and explaining to others why evolution behaves in the way it does.

We refer to this process as *principled simplification*, though we lay no claim to having invented the idea. We do, however, believe that Artificial Life and many fields allied with Computer Science would benefit from a deeper understanding of how and when it works as a method. To this end, one key methodological insight arising from this study is that it is important to distinguish between qualitative features associated with problem structure, problem scale, and finally other quantitative features orthogonal to scale. Concerning the quantitative nature of scale-based features, why would we expect a smaller problem to enable us to predict something that relies on scale in order to determine outcomes? As an example, the impact of Random Immigrants (and other algorithm features of this type, we hypothesise) cannot be predicted using this method, since they rely on relative probabilities associated with different outcomes, that are dependent on the size of the problem search space. And in this particular simplification, problem search space has been reduced. It is conceivable that a different simplification of the RCT could be constructed that preserved aspects of scale, but this is not something that has yet been done. Conversely, both structural and quantitative features unrelated to the scale of the problem can be analysed in simplified versions that preserve them, in a way that can generalise to more complex variants that share those same features. This is the case in our study with the shape of the fitness landscape, which is a major determining factor on the behaviour of evolutionary processes on landscapes of that shape. It is also the case in the observations concerning the impact of a population. While in the latter case we have not extended our pen-and-paper analysis of the RC- to explain this result, empirically we observe a link between the RC- and RCT.

In other words, when considering principled simplification as a tool, ask: is this feature orthogonal to the simplification and hence preserved; or in simplifying, have we altered the feature? Here, we have shown examples of analysis and experimental prediction using a preserved feature, where we were indeed able to predict outcomes in the more complex version of the problem, and one example where, due to the feature being quantitatively dependent on the scale of the simplification, we were not.

7 Conclusions

Evolutionary agent-based models are widely used in a number of applications, but are often built upon a number of assumptions that can inadvertently make agent behaviour and

evolution hard to explain or understand. We demonstrate that principled simplification can be used as a tool to support the explainability of complex evolutionary agent-based models, by presenting an experimental case study that shows behaviour can be explained and predicted in an instance of the River Crossing Task (RCT) through simplification, using the Minimal River Crossing (RC-) Task.

Qualitative simplifications to the structure of the problem or environment can facilitate the explainability of agent behaviour and evolution, as results in the simplified testbed are shown to generalise back to the original problem. Specifically, we demonstrate this by exploring how the addition of a cost to movement affects the ability of evolution to find goal-achieving solutions in the RCT – the results of which can be predicted with the simplified RC-. Agents have no disincentive to exploration when there is no cost to fitness, and the resulting neutral fitness landscape allows agents to experiment with behaviours without penalty; with a cost to movement however, a trap arises in the fitness landscape, which deters and pushes agents away from the optimum, thus making it harder for evolution to find successful solutions. Similarly, there is evidence that qualitative predictions associated with quantitative features unrelated to problem scale can also be made. This is the case in experimentally examining the size of the population; this is a quantitative feature of the evolutionary algorithm that affects its behaviour but, unlike Random Immigrants that are sampled from the problem space, is not dependent on problem space size.

Further, what might not be so obvious is how implementation details regarding the setup of the environment and evolutionary algorithm may also affect evolution. In previous work (Ghouri et al., 2020), we found that when agents were subjected to a cost to movement in the RCT – with no option *but* to move at each timestep and thus incur the maximum penalty if the goal was not achieved – the resulting fitness landscape does not contain a trap but instead becomes neutral. By instead allowing agents to intentionally decide whether to move or not at each timestep (as we have done in this paper), the landscape stops being neutral and a trap is again present. Another factor to consider in testbed design is how initialisation may affect evolutionary search; in this experimental study, agents that evolve in the RC- are initialised at zero $(0, 0)$, but it is not difficult to imagine that random initialisation could have a completely different effect on evolution when the search space is small. Considering the RC- has 12 possible valid solutions, if all agents in a population size of 5 are unique, nearly half of the search space is already explored before evolution even begins. Whilst an increase in diversity that arises from a larger population size may potentially influence

the number of Builders that evolve than a 1+1 EA, incremental evolution may still be problematic for agents that are far from the optimum depending on where in the search space they begin. We therefore show that features that appear to be minor implementation details can become a critical factor in how evolution proceeds – the implications of which would not have become obvious nor easily understood without a study of the environment via principled simplification.

Arguably the most interesting discovery arising from this experimental study surrounds when principled simplification *cannot* predict or explain agent evolution or behaviour in ways that can generalise back to the original task from which it was inspired. Simplification of structural aspects of the environment or algorithm, such as a cost to movement or larger population size (as is explored in this article), is shown to be able to predict how evolution will be affected when experiments are repeated in more complex environments. However, when elements that are influenced by the scale of the problem itself are introduced, such as changes that are affected by the size of the search space, agent behaviour and evolution can no longer be explained or predicted due to a difference in magnitude between the simplified and original problems. Our exploration of the introduction of Random Immigrants is a prime example of when principled simplification is ineffective for explaining or predicting agent behaviour when evolving in the original task, because randomness is likely to have a greater impact as the search space decreases in size.

Finally, we note that in this study we have not proven that this approach will or will not work in a given situation. Simply, we have provided evidence in support of our claims. We therefore recommend that this study act as a template for future similar studies, in order that further evidence concerning the validity of predictions based on different forms of feature can be obtained.

References

Aguilar, W., Santamaría-Bonfil, G., Froese, T., & Gershenson, C. (2014). The Past, Present, and Future of Artificial Life. *Frontiers in Robotics and AI*. https://doi.org/10.3389/frobt.2014.00008

Andras, P., Esterle, L., Guckert, M., Han, T. A., Lewis, P. R., Milanovic, K., Terry Payne, C. P., Pitt, J., Powers, S. T., Urquhart, N., & Wells, S. (2018). Trusting intelligent

machines: Deepening trust within socio-technical systems. *IEEE Technology and Society Magazine*, *37*.

Barbosa, H. S., de Lima Neto, F. B., & Fusco, W. (2011). Migration and social networks — an explanatory multi-evolutionary agent-based model. *IEEE Symposium on Intelligent Agent (IA)*.

Barnes, C. M., Ekárt, A., & Lewis, P. R. (2019). Social action in socially situated agents. *Proceedings of the IEEE 13th International Conference on Self-Adaptive and Self-Organizing Systems*, 97–106.

Bedau, M. A. (2007). Artificial life. In M. Matthen & C. Stephens (Eds.), *Philosophy of biology* (pp. 585–603). North-Holland.

Borg, J. M., Channon, A., & Day, C. (2011). Discovering and maintaining behaviours inaccessible to incremental genetic evolution through transcription errors and cultural transmission. *Proceedings of the European Conference on Artificial Life 2011*, 102–109. https://doi.org/http://dx.doi.org/10.7551/978-0-262-29714-1-ch019

Brutschy, A., Pini, G., Pinciroli, C., Birattari, M., & Dorigo, M. (2014). Self-organized task allocation to sequentially interdependent tasks in swarm robotics. *Autonomous Agents and Multi-Agent Systems*. https://doi.org/10.1007/s10458-012-9212-y

Cederman, L.-E. (2002). Endogenizing geopolitical boundaries with agent-based modeling. *Proceedings of the National Academy of Sciences*, *99*(suppl 3), 7296–7303.

Cobb, H. G., & Grefenstette, J. J. (1993). Genetic algorithms for tracking changing environments. *Proceedings of the 5th International Conference on Genetic Algorithms*, 523–530.

Droste, S., Jansen, T., & Wegener, I. (2002). On the analysis of the (1 + 1) evolutionary algorithm. *Theoretical Computer Science*.

Eldridge, A., & Kiefer, C. (2018). Evolutionary agent based models of the acoustic niche hypothesis. *ALIFE 2018: The 2018 Conference on Artificial Life*, 296–303.

Farmer, J., & Foley, D. (2009). The economy needs agent-based modelling. *Nature*, *460*, 685–686.

Federici, D., & Downing, K. (2006). Evolution and development of a multicellular organism: Scalability, resilience, and neutral complexification. *Artificial Life*.

Ghouri, A., Barnes, C. M., & Lewis, P. R. (2020). A Minimal River Crossing Task to Aid the Explainability of Evolutionary Agents. *Artificial Life Conference Proceedings*, 36–43.

Helbing, D. (2012). Agent-based modeling. In D. Helbing (Ed.), *Social self-organization*. Springer.

Jacobs, R. A., Jordan, M. I., & Barto, A. G. (1991). Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive Science*. https://doi.org/10.1016/0364-0213(91)80006-Q

Jain, A., Subramoney, A., & Miikulainen, R. (2012). Task decomposition with neuroevolution in extended predator-prey domain. *Artificial Life 13: Proceedings of the 13th International Conference on the Simulation and Synthesis of Living Systems, ALIFE 2012*. https://doi.org/10.7551/978-0-262-31050-5-ch045

Jolley, B. P., Borg, J. M., & Channon, A. (2016). Analysis of social learning strategies when discovering and maintaining behaviours inaccessible to incremental genetic evolution. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9825 LNCS*, 293–304. https://doi.org/10.1007/978-3-319-43488-9_26

Ma, T., & Nakamori, Y. (2005). Agent-based modeling on technological innovation as an evolutionary process [Advances in Complex Systems Modeling]. *European Journal of Operational Research*, *166*(3), 741–755.

Maziarz, M., & Zach, M. (2020). Agent-based modelling for sars-cov-2 epidemic prediction and intervention assessment: A methodological appraisal. *Journal of Evaluation in Clinical Practice*, *26*(5), 1352–1360.

Nicolay, D., Roli, A., & Carletti, T. (2014). Learning multiple conflicting tasks with artificial evolution. *Communications in Computer and Information Science*. https://doi.org/10.1007/978-3-319-12745-3_11

Nijssen, S., & Bäck, T. (2003). An analysis of the behavior of simplified evolutionary algorithms on trap functions. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/TEVC.2002.806169

Powers, S. T., Ekárt, A., & Lewis, P. R. (2018). Modelling enduring institutions: The complementarity of evolutionary and agent-based approaches. *Cognitive Systems Research*, *52*, 67–81.

Robalino, D. A., & Lempert, R. J. (2000). Carrots and sticks for new technology: Abating greenhouse gas emissions in a heterogeneous and uncertain world. *Integrated Assessment*, *1*, 1–19.

Robinson, E., Ellis, T., & Channon, A. (2007). Neuroevolution of agents capable of reactive and deliberative behaviours in novel and dynamic environments. *Advances in artificial life* (pp. 1–10). Springer. https://doi.org/doi:10.1007/978-3-540-74913-4_35

Schelling, T. C. (1971). Dynamic models of segregation. *The Journal of Mathematical Sociology*, *1*(2), 143–186.

Stanton, A., & Channon, A. (2015). Incremental Neuroevolution of Reactive and Deliberative 3D Agents. *European Conference on Artificial Life (ECAL)*, 341–348. https://doi.org/10.7551/978-0-262-33027-5-ch063

Torrens, P. M. (2010). Agent-based models and the spatial sciences. *Geography Compass*, *4*(5), 428–448.

Yang, S. X., & Meng, M. (2000). An efficient neural network approach to dynamic robot motion planning. *Neural Networks*, *13*(2), 143–148. https://doi.org/10.1016/S0893-6080(99)00103-3