Complexity of Agents in Non-stationary Environments: A Partial Computational Model

Vadim Bulitko BULITKO@UALBERTA.CA
Department of Computing Science, University of Alberta, Edmonton, AB, CANADA

Valeriy K. Bulitko GBOOLY@GMAIL.COM
Centre for Science, Athabasca University, Athabasca, AB, CANADA

Abstract

There is an ongoing debate on importance of human-inspired problem-solving methods in Artificial Intelligence. Thus it is of interest to consider problem-solving needs of any intelligent agent. In our previous work we proposed that agent's cognitive complexity should rise as the agent's environment changes more rapidly. We supported our conjecture with a theory but offered no specific model. In this paper we follow up with a simple computational which illustrates two lowest cognitive levels in our theory. To do so we represent agents symbolically as arithmetic formulae. In line with our previous work, the agents survive by finding a fixed point. We show that while the first cognitive level of our theory can be reached easily by individual agents. The second cognitive level in the theory can then be reached by treating the whole population as a single cognitive entity.

1. Introduction

The field of Artificial Intelligence (AI) was once inspired by broadly applicable human problem-solving techniques yet modern AI techniques tend to be performance-driven and use human-independent methods (Langley, 2018a). A natural question is thus whether there are any problem-solving methods needed for *any* intelligent agent. The answer may depend on the problem an agent sets out to solve and the context of the said problem. In our previous work (Bulitko & Bulitko, 2019) we narrowed the question by assuming that an agent's only task is to survive in an Artificial Life (A-life) style setting (Wilensky & Rand, 2015). We then proposed, in the terminology of Langley (2018b), a basic theory that linked agent's survival to finding a steady-state solution which is a fixed point in the agent-environment interaction (detailed below in Section 2). We argued that a non-stationary environment changing more rapidly would call for cognitively more complex agents and described four possible classes of such agents. The conjecture has seen support from biological evolution leading to emergence of human intelligence (Calvin, 2002).

This paper follows our previous work and proposes a concrete computational model to illustrate our abstract theory. The model enables us to conduct computational experiments in which agents have their control/behaviour policy encoded genetically and evolve over time. To keep our model and the findings human-readable we chose to use a simple symbolic representation of agent behaviour policy: a space of arithmetic formulae over a compact context-free grammar. We then

show how a higher cognitive complexity is called for in more rapidly changing environment. In doing so we see how the first two cognitive levels in the theory can be seen with an A-life evolution. Furthermore, we demonstrate that a population of simpler agents can be collectively viewed as a more cognitively complex single agent.

The rest of the paper is organized as follows. First, in Section 2 we briefly recap our previously proposed abstract theory. We then present the primary contribution of this paper, our computational model, in Section 3. Empirical results are found in Section 4, followed by a discussion of current shortcomings and future work in Section 5. The paper is concluded in Section 6.

2. A Recap of Our Theory

We previously considered cognitive agents whose task is to survive in an environment by solving the problems it presents (Bulitko & Bulitko, 2019). Specifically, at time t the agent Φ_a perceives its problem as P_t and proposes a solution S_t: $\Phi_a(P_t) = S_t$. The environment Φ_e then reacts to the solution by producing the next problem for the agent to solve: $\Phi_e(S_t) = P_{t+1}$ and the cycle repeats. Agent survives if and only if finds a *steady-state solution*: a *fixed point* of the composition $\Phi_a \circ \Phi_e$:

$$\Phi_e(\Phi_a(P)) = P. \tag{1}$$

A fixed point of $\Phi_a \circ \Phi_e$ provably exists when Φ_e and Φ_a can be represented as enumeration operators with indecies a and e (Rogers, 1987; Bulitko, 1986). Then problems P_t and the solutions S_t are encoded as sets of natural numbers. Furthermore a fixed point can be computed by the agent.

The environment is *substantially non-stationary* when Φ_e depends on t, $\Phi_e(S_t, t) = P_{t+1}$, and no single fixed point P exists for all t. In that case the agent needs to modify its steady-state solution as the environment changes. Predicting or even tracking changes in the environment may require additional cognitive abilities in the agent. Thus depending on how rapidly Φ_e changes we postulated the following levels of additional machinery that need to be present in the agent.

Level 1: Stationary Environments. In the simplest case $\Phi_e(S_t, t)$ does not depend on t which allows the agent to either compute its fixed point during its life time or have the fixed-point solution encoded in its genes.

Level 2: Slowly Non-stationary Environments. When $\Phi_e(S_t, t)$ substantially depends on t the agent can observe a sequence of problems and solutions $(P_i, S_i), i \leq t$ during its lifetime and use it to derive a model of the environment. In terms of the enumeration operators this means that the agent maps the sequence $(P_i, S_i), i \leq t$ to the current operator index e_t. Knowing that $\Phi_e(S, t) = \Phi_{e_t}(S)$ allows the agent to compute a steady-state solution for the environment as it exists at time t.

Level 3: Rapidly Non-stationary Environments. Note that it takes agent time ticks to collect enough observations (P_i, S_i) so that e_t can be derived from them. Additional time ticks are then needed to compute e_t given the observations followed by computing a fixed point for $\Phi_a \circ \Phi_{e_t}$. Thus if the environment changes rapidly enough by the time the agent has modeled the environment and computed a steady-state solution for it, the environment will have already changed. In that case we

proposed that the agent predicts the environment m steps into the future and computes a steady-state solution not for the current environment but for what it will be by the time such computation is over.

Level 4: Excessively Non-stationary Environments. As we increase the rate of change in the environment we can reach the point where the agent cannot predict the environment far enough into the future and accurately enough to compute a fixed point for it. We proposed that in this scenario the agent needs to find an ecological niche where the environment changes less rapidly (i.e., a *refugia*) so that the agent can survive there.

In our previous paper we illustrated the theory with hypothetical examples but did not present a computational model to go with it. We do so in the next section.

3. Our Proposed Computational Model

There are numerous possible representations of agent's behaviour policies, including heuristic-guided agent-centered search (Koenig, 2001) and neural networks (Silver et al., 2016; Graves et al., 2014, 2016). Likewise, there are a number of methods of evolving such behaviour policies, including evolution of heuristic search agents (Bulitko, 2016), reinforcement learning (Sutton & Barto, 1998; Ackley & Littman, 1991) and neuroevolution (Stanley & Miikkulainen, 2002; Such et al., 2017; Miikkulainen et al., 2017). We strive for simplicity of both our behaviour representation and its evolution and thus use techniques from recent work by Bulitko (2020).

3.1 Representation of Agent Behaviour Policies

Our intention was to illustrate the theory with a basic, easily human-readable model. Thus, we picked a simple representation of the environment and the agents. Each problem P presented to the agent by the environment is encoded by a real number. The agent's solution S is also a real number. The agent's behaviour policy Φ_a is a function of the environment's current problem P_t. The value of $S_t = \Phi_a(P_t)$ is calculated with a formula generated by the following context-free grammar:

$$\Phi_a \rightarrow T \mid U \mid B \tag{2}$$

$$T \rightarrow P_t \mid n \tag{3}$$

$$U \rightarrow \sqrt{\Phi_a} \mid |\Phi_a| \mid -\Phi_a \mid \Phi_a^2 \mid \lfloor \ \rfloor \mid \lceil \ \rceil \tag{4}$$

$$B \rightarrow \Phi_a + \Phi_a \mid \Phi_a - \Phi_a \mid \Phi_a \cdot \Phi_a \mid \frac{\Phi_a}{\Phi_a} \mid \max\{\Phi_a, \Phi_a\} \mid \min\{\Phi_a, \Phi_a\} \mid \mathrm{mod}(\Phi_a, \Phi_a) \tag{5}$$

Here $n \in \{-5, -4, \ldots, 0, \ldots, 5\}$. Parentheses are added as necessary when building a formula and omitted from the grammar above for clarity. If for a given input P_t the formula Φ_a computes a complex number then its imaginary component is discarded. If for any input value the formulae is undefined then $S_t = 0$. If S_t exceeds an *a priori* set maximum $S_{\max}$ then it is set to $S_{\max}$. Likewise, we lower bound S_t at $S_{\min}$. By convention $\forall x \, \mathrm{mod}(x, 0) = x$. The grammar above defines a space of all possible agent behaviour policies $\mathbf{\Phi}$.

3.2 Agent Evolution

A formula representing the environment is fixed while agents' formulae are created randomly and evolve over time in a basic simulated evolution. We use an A-life style evolution (Wilensky & Rand, 2015; Ackley & Littman, 1991) which allows us to simulate a life time of each individual agent as a sequence of problems (P_t) the environment presents the agent with and the solutions (S_t) the agent responds with. This is a finer temporal resolution than in typical evolutionary algorithms which operate at the level of whole generations (Eiben & Smith, 2015).

Our A-life evolution is implemented by Algorithm 1. Time t advances in discrete time steps from 1 to $t_{\max}$ in line 5. The initial population $\mathcal{P}_1$ of N_0 agents is formed randomly in line 2.* Each initial agent is formed by starting with a syntax tree of a single terminal node (T in the grammar above) and mutating it a random number of times. We keep track of each agent's $\Phi_{a,i}$ energy $\varepsilon_{t,i}$ which changes over the agent's lifetime. At birth each agent gets the amount of energy ε_0.

Algorithm 1: Evolution of Agent Behaviours

input : initial population size N_0, maximum simulation time $t_{\max}$, mutation rate μ, regularizer λ, environment Φ_e, bounds on agent response $[S_{\min}, S_{\max}]$, desired problem P_{desired}, minimum survival energy $\varepsilon_{\min}$, minimum parenting energy $\varepsilon_{\text{parent}}$, energy at birth ε_0

1 $t \leftarrow 1$
2 form initial population $\mathcal{P}_t \leftarrow \{\Phi_{a,i}\}$ of size N_0
3 for all $1 \le i \le N_0$ set $\varepsilon_{t,i} \leftarrow \varepsilon_0$
4 $P_t \leftarrow 0$
5 **while** $\mathcal{P}_t \neq \varnothing$ & $t < t_{\max}$ **do**
6 $f_t \leftarrow F(|\mathcal{P}_t|, N_0)$
7 **for** $i = 1, \ldots, |\mathcal{P}_t|$ **do**
8 $S_{t,i} \leftarrow \Phi_{a,i}(P_t)$
9 $S_{t,i} \leftarrow \max\{S_{\min}, \min\{S_{\max}, S_{t,i}\}\}$
10 $P_{t+1,i} \leftarrow \Phi_e(S_{t,i}, t)$
11 $\Delta_{t,i} \leftarrow |P_{t+1,i} - P_{\text{desired}}|$
12 $\Delta'_{t,i} \leftarrow \Delta_{t,i} + \lambda|\Phi_{a,i}|$
13 $\varepsilon_{t+1,i} \leftarrow \varepsilon_{t,i} + f_t - \Delta'_{t,i}$
14 $P_{t+1} \leftarrow \text{mean}_i\, P_{t+1,i}$
15 $\mathcal{P}_{t+1} \leftarrow \mathcal{P}_t \setminus \{\Phi_{a,i} \in \mathcal{P}_t \mid \varepsilon_{t+1,i} < \varepsilon_{\min}\}$
16 **for** $\Phi_a \in \{\Phi_{a,i} \in \mathcal{P}_{t+1} \mid \varepsilon_{t+1,i} \ge \varepsilon_{\text{parent}}\}$ **do**
17 clone parent Φ_a into child Φ'_a and mutate the latter
18 $\mathcal{P}_{t+1} \leftarrow \mathcal{P}_{t+1} \cup \{\Phi'_a\}$
19 adjust the child's and parent's energy
20 $t \leftarrow t + 1$

*A population is a multi-set of agent behaviour policies $\Phi_{a,i}$ thus allowing for multiple copies of the same policy.

At time t the environment presents the same problem P_t to all agents in the current population $\mathcal{P}_t$. Each agent $\Phi_{a,i}$ solves P_t in its own way by producing the solution $S_{t,i} = \Phi_{a,i}(P_t)$ in line 8. The environment reacts to the solution by producing the new problem, $P_{t+1,i} = \Phi_e(S_{t,i}, t)$ specific to the agent $\Phi_{a,i}$ in line 10. Any agent desires to be presented with a problem P_{desired}.[†] Thus the discrepancy between what an agent gets from the environment in response to its action (i.e., $P_{t+1,i}$) and what the agent desires (i.e., P_{desired}) is the agent's *loss* $\Delta_{t,i}$ calculated in line 11.

An agent's energy level is adjusted in line 13. Each agent gains f_t computed in line 6. The amount f_t increases with smaller populations and decreases with larger populations which implements the notion of a global resource shared by all agents in the population. The specific dependency F is detailed in a later section. While the energy gain amount is the same for all agents in the population, energy loss is linked to the agent's loss $\Delta_{t,i}$ regularized with agent's size (line 12). We refer to it as *regularized loss* and denote it by $\Delta'_{t,i}$ henceforth. The agent's size $|\Phi_{a,i}|$ is the number of nodes in the syntax tree that encodes the formula $\Phi_{a,i}$. As with biological agents, larger formulae/brains give the agent a potential for smarter behaviours (i.e., $P_{t+1,i}$ can be driven closer to P_{desired}) but use up more energy.

All agents in the population collectively affect the environment. Specifically, line 14 sets the new global problem P_{t+1} as the mean of the environment's individual responses to each agent. Thus smarter agents help weaker agents drive the global problem P to the desired state P_{desired} whereas weaker agents can counteract such progress. As evolution goes on weaker agents tend to face larger losses which shrink their energy levels. Once an agent's energy falls below a pre-determined threshold $\varepsilon_{\min}$, the agent dies and is removed from the population (line 15).

Agents whose current energy is at least $\varepsilon_{\text{parent}}$ and who have reached a child-bearing age give birth to other agents in line 17. Each parent asexually produces one child which genetically inherits the parent's formula defining the behaviour policy Φ_a. The genome representing the parent's behaviour policy is a syntax tree, encoding a formula. The child's syntax tree is then randomly mutated by removing, adding and modifying its nodes. The number of such mutations for each child is drawn from the distribution $\lceil 0.1 + \mathcal{E}(\mu) \rceil$ where $\mathcal{E}(\mu)$ is the exponential distribution with the mean μ. The result is a new formula representing the child's behaviour Φ'_a. In line 18 the child is added to the population. The child gets the energy ε_0 which is subtracted from the parent's current energy level in line 19. Once all children are added to the population the next time step begins (line 20).

4. An Empirical Illustration

We will first demonstrate how A-life produces better agents when the environment is stationary. We will then gradually make the environment non-stationary, increasing its rate of change. According to our theory agents in such an environment will need to model the environment to continuously recompute their steady-state solution. Our agents are represented by formulae over a simple arithmetic grammar and lack building blocks, such as memory, to explicitly model the environment. However,

[†]The original theory did not distinguish any particular fixed point (Bulitko & Bulitko, 2019). We do so in this paper to make the computational results easier to analyze. The choice to prefer a problem and not a solution is motivated by the fact that P_t can be viewed as the agent's state while S_t can be viewed as the agent's action in that state. Then the agent can prefer to be in certain states (e.g., where it collects a resource in an A-life setting (Bulitko et al., 2017)).

the evolution itself is a fixed point solver. Thus, a population of agents considered as a single cognitive entity can indeed adapt to a non-stationary environment.

4.1 Stationary Environments and Level-1 Agents

Level-1 agents can be born with formulae that respond to a problem P with a solution S as to drive the next P towards P_{desired}. In this section we assume that the environment is stationary:

$$\forall S, t_1, t_2 \left[\Phi_e(S, t_1) = \Phi_e(S, t_2)\right]. \tag{6}$$

Then we need to solve:

$$\Phi_e(\Phi_a^*(P_{\text{desired}})) = P_{\text{desired}} \tag{7}$$

for a function Φ_a^* in order to get to the desired fixed point P_{desired}. This is conceptually simple to do by first solving $\Phi_e(x) = P_{\text{desired}}$ for $x \in [S_{\min}, S_{\max}]$ and then solving $\Phi_a^*(P_{\text{desired}}) = x$ for $\Phi_a^* \in \mathbf{\Phi}$. However, our agents lack means to explicitly solve either equation. So the burden of solving falls on the evolution. Our A-life evolution is indeed set up to reward agents whose Φ_a approaches Φ_a^* as the absolute difference Δ between the environment's response to the agent's solution and the desired fixed point P_{desired} affects the agent's energy loss. The closer the agent drives the environment to P_{desired}, the less energy it losses on each time step, leaving it more energy to survive and reproduce.

The reasoning is supported by the following computational experiments. Consider the evolution run in Figure 1. The environment is $\Phi_e(S) = 3 - S$ which has a steady-state solution $P_{\text{desired}} = 0$ with the agent policy $\Phi_a^*(P) = 3$:

$$\Phi_e(\Phi_a^*(P_{\text{desired}})) = \Phi_e(\Phi_a^*(0)) = 3 - \Phi_a^*(0) = 3 - 3 = 0 = P_{\text{desired}}. \tag{8}$$

We set all control parameters as per Table 1. Starting with a random population of agent formulae, the evolution produces the following lowest-$\Delta'_{t,i}$ agents: $\Phi_a(P) = \min\{3, \lceil 4 \rceil\}$ at time $t = 1$, $\Phi_a(P) = \min\{3, 4\}$ at time $t = 75$, $\Phi_a(P) = \lceil 3 \rceil$ at time $t = 273$ and finally $\Phi_a(P) = 3$ at time $t = 392$. The corresponding regularized loss is plotted as the curve labeled "lowest Δ'" (Figure 1, right). Note that the final agent produced by the evolution is the best possible in terms of incurred lowest regularized loss: $\Delta' = |\Phi_e(\Phi_a(x)) - P_{\text{desired}}| + \lambda|\Phi_a| = |3 - 3 - 0| + \lambda = \lambda$ as the syntax tree for $\Phi_a = 3$ is the smallest allowed, with only a single node.

Not only the final agent produced by the evolution has the lowest possible loss but also the average regularized loss of the population $\text{mean}_i\, \Delta'_i$ decreases (Figure 1, right). This is because the mean of the solution produced by agents approaches 3 which drives the global problem P produced by the environment towards P_{desired} (Figure 1, left).

In summary, for a stationary environment Φ_e the A-life evolution can solve Equation 7 for an agent policy Φ_a^* and indeed does so in the simple example we walked through above.

4.2 Non-stationary Environments and Level-2 Agents

Suppose now that the environment is non-stationary (i.e., Condition 6 does not hold):

$$\exists S, t_1, t_2 \left[\Phi_e(S, t_1) \neq \Phi_e(S, t_2)\right]. \tag{9}$$

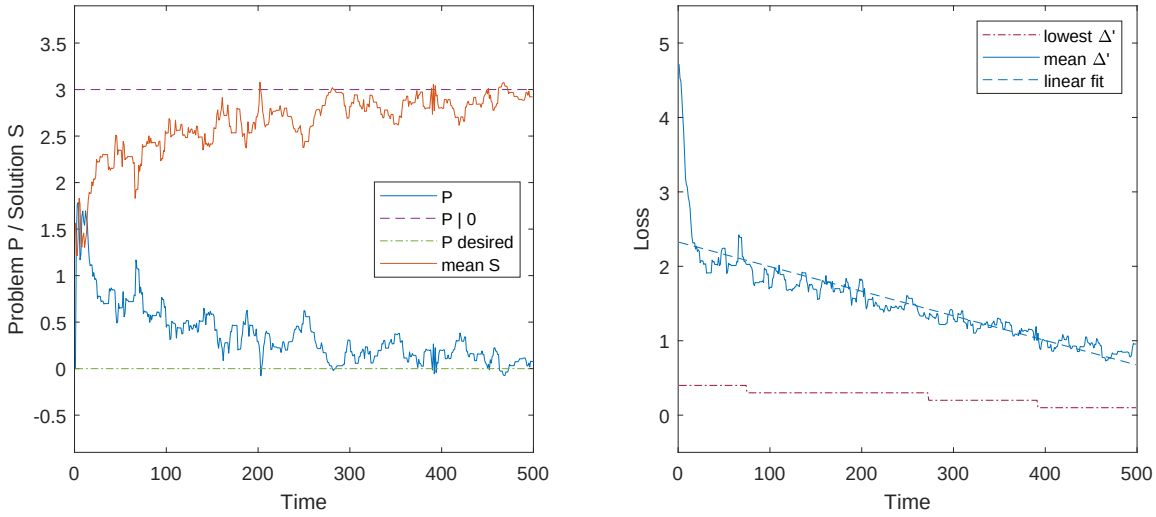

Figure 1. **Left:** evolution of environment's problem P and the agent's solution S. The line $P \mid 0$ represents the environment problem P when $S = 0$. **Right:** mean and lowest regularized losses incurred by the agents.

Table 1. Parameters used for the evolution in Figure 1.

Parameter	**Value**
initial population size	$N_0 = 100$
maximum simulation time	$t_{\max} = 500$
mutation rate	$\mu = 1$
regularizer	$\lambda = 0.1$
bounds on S	$[-10, 10]$
desired problem	$P_{\text{desired}} = 0$
minimum parenting age	5
minimum parenting energy	$\varepsilon_{\text{parent}} = 75$
minimum survival energy	$\varepsilon_{\min} = 10$
energy at birth	$\varepsilon_0 = 50$
energy gain	$F(N, N_0) = 1 + 2\sigma(\frac{1}{10}(N - \frac{N_0}{2})), \sigma(x) = \frac{1}{1+e^x}$

Then it may not be possible to solve:

$$\Phi_e(\Phi_a^*(P_{\text{desired}}), t) = P_{\text{desired}} \tag{10}$$

for an agent policy Φ_a^* if it lacks t as the input.[‡] To illustrate, consider $\Phi_e(S, t) = \sin\frac{\pi t}{500} - S$ which has the period of 1000 with respect to t. The dependence on t precludes any constant solution x to

[‡]Note that Condition 9 does not preclude Φ_e from being time-insensitive for *some* values of S. For instance, $\Phi_e(S, t) = S \sin t$ is insensitive to t when $S = 0$. Thus, it is possible that $\Phi_a^*(P_{\text{desired}})$ produces such S which makes $\forall t_1, t_2\, [\Phi_e(\Phi_a^*(P_{\text{desired}}), t_1) = \Phi_e(\Phi_a^*(P_{\text{desired}}), t_2)]$. We consider this to be a weak non-stationarity of Φ_e and do not take advantage of it.

$\Phi_e(x,t) = 0$. Since for any $\Phi_a \in \mathbf{\Phi}$ the agent's output $\Phi_a(P_{\text{desired}}) = \Phi_a(0)$ is a constant no function Φ_a can make P_{desired} a fixed point of $\Phi_e(\Phi_a(P_{\text{desired}}), t) = P_{\text{desired}}$ for all t.

Our theory in Section 2 suggested that non-stationary environments call for at least level-2 agents which can observe the environment and solve for a fixed point needed for the *current* environment. As the environment changes a new solution will be needed and so on. This paper's computational model confines the agent behaviour to simple formulae which lack any means to observe the environment and repeatedly solve it for a fixed point. However, an evolving *population* of agents can perform the same function. Indeed, as we demonstrated in Section 4.1, the A-life evolution itself can solve $\Phi_e(\Phi_a^*(P_{\text{desired}})) = P_{\text{desired}}$ for the agent formula $\Phi_a^* \in \mathbf{\Phi}$. If Φ_e changes over time then the evolution will produce as series of such formulae. Thus, in our computational model we can view the whole population as a single level-2 agent.

To illustrate this reasoning empirically we will start with a mild non-stationary environment. There the rate of change with t is low enough that a level-2 agent has enough time to observe the environment and solve it for a fixed point before it changes again. When our level-2 agent is a population of level-1 agents then evolution needs to be fast enough to solve for a fixed point before the environment changes. Consider the non-stationary environment $\Phi_e(S,t) = 3\,\text{round}(\sin\frac{\pi t}{2000}) - S$. It changes slowly in t (the period is 4000) and it also changes in a step-like manner due to rounding to an integer. Thus it is constant in t on each of the steps which gives the evolution time to solve it for a fixed point.

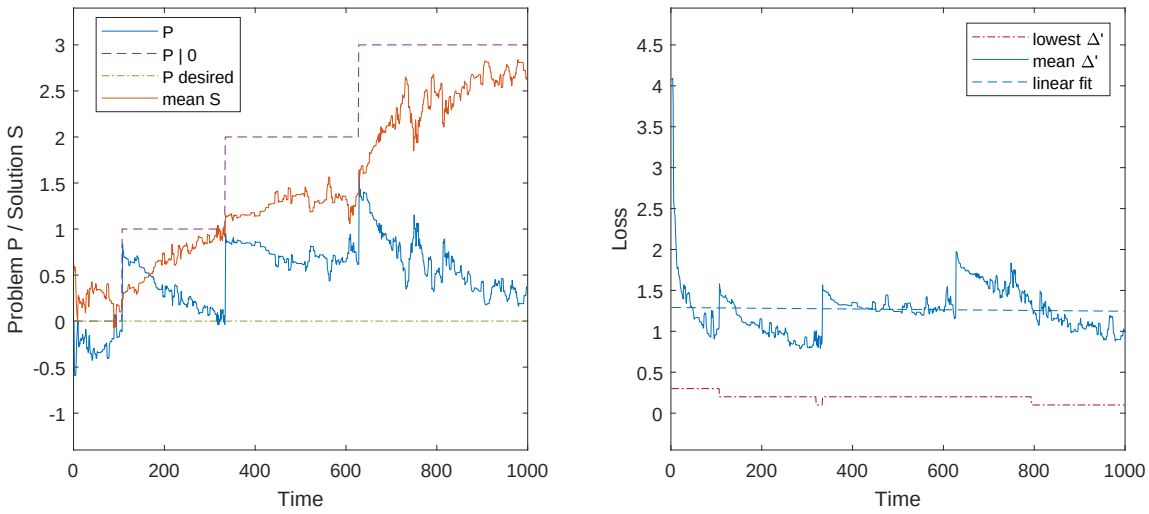

Figure 2. **Left:** evolution of environment's problem P and the agent's solution S. The line $P \mid 0$ represents the environment problem P when $S = 0$. **Right:** mean and lowest regularized losses incurred by the agents.

Figure 2 shows evolution results. The parameters are the same as before (Table 1) but we doubled the maximum simulation time: $t_{\max} = 1000$. The four steps of $3\,\text{round}(\sin\frac{\pi t}{2000})$ are clearly visible as $P \mid 0$ (Figure 2, left). For each step the evolution shapes its population towards driving P to $P_{\text{desired}} = 0$ (the curve labeled P in the left subplot). This is achieved by increasing the agents average response S ("mean S" in the subplot). Correspondingly the average regularized

loss Δ' drops gradually within each step and spikes when the environment abruptly changes at the beginning of the next step (Figure 2, right). The best linear fit remains approximately flat which indicates that evolution is able to keep up with the non-stationary environment.

Looking at the lowest-regularized-loss agent formulae found we see that they match the value of the first term in $\Phi_e(S,t) = 3\,\mathrm{round}(\sin\frac{\pi t}{2000}) - S$ (i.e., $\Phi_e(0,t)$) as shown in Table 2. The best agent at $t = 1$ is $\Phi_a(P) = \lceil\sqrt{-3}\rfloor$ whose real part is 0, matching the first step of $3\,\mathrm{round}(\sin\frac{\pi t}{2000})$. The agent's regularized loss is $|0-0| + \lambda|\Phi_a| = 0.3$. The second step begins at $t = 107$ when $\Phi_e(0,t)$ switches from 0 to 1. The evolution responds with $\Phi_a = \sqrt{1}$ which gives the loss $\Delta = 0$ and the regularized loss $\Delta' = 0.2$. Evolution improves on it at time $t = 320$ by getting rid of the unnecessary square root. The new agent $\Phi_a = 1$ drives Δ' to 0.1 – the best possible value. At time $t = 334$ the next step begins with the rounded sine rising to 2. The evolution matches it with $\Phi_a = \sqrt{4}$ which it is unable to simplify during the step. The final step begins at $t = 628$. It is tracked with $\Phi_a = |3|$ which is improved upon at $t = 794$ with $\Phi_a = 3$. In summary, the environment changed slowly enough for the evolution to solve for the desired fixed point and reach the loss of 0 on all steps. Furthermore, on two out of four time steps the evolution found the most compact formulae for it.

Table 2. Evolution tracking changes in the non-stationary environment $\Phi_e(S,t) = 3\,\mathrm{round}(\sin\frac{\pi t}{2000}) - S$.

Time t	**$\Phi_e(0,t)$**	**Lowest regularized loss agent**	**Its Δ'**
1	0	$\lceil\sqrt{-3}\rfloor$	0.3
107	1	$\sqrt{1}$	0.2
320	1	1	0.1
334	2	$\sqrt{4}$	0.2
628	3	$\lvert 3\rvert$	0.2
794	3	3	0.1

Speeding up the rate of change in Φ_e gives evolution less time to find Φ_a which leads to the desired fixed point P_{desired}. Indeed, evolving agents for the environment $\Phi_e(S,t) = 3\,\mathrm{round}(\sin\frac{\pi t}{200}) - S$ which changes 10 times faster tends to keep the lowest regularized loss higher (Table 3, $t_{\max} = 100$). This is because the evolution has less time to find more compact formulae. Note that it still manages to find a fixed point for each of the four steps.

Table 3. Evolution tracking changes in the non-stationary environment $\Phi_e(S,t) = 3\,\mathrm{round}(\sin\frac{\pi t}{200}) - S$.

Time t	**$\Phi_e(0,t)$**	**Lowest regularized loss agent**	**Its Δ'**
1	0	$\frac{0}{4}$	0.3
11	1	$\lVert\lfloor 1\rfloor\rVert$	0.3
34	2	$\left\lfloor\left\lfloor\sqrt{\lVert(-5)\rVert}\right\rfloor\right\rfloor$	0.6
63	3	$\lvert -3\rvert$	0.2

If we make the environment change even faster with t by setting $\Phi_e(S,t) = 3\,\mathrm{round}(\sin\frac{\pi t}{20}) - S$ then the evolution fails to find a fixed point for each of the steps, let alone a compact formula for it (Table 4, $t_{\max} = 10$). Indeed for the last step, $\Phi_e(0,t) = 3$ the best agent found was $\Phi_a = \left(\frac{(-5)}{3}\right)^2 \approx$

2.78 which does not match the 3 and raises Δ' to $\left|3 - \left(\frac{(-5)}{3}\right)^2\right| + \lambda|\Phi_a| \approx 0.622$. Note that each evolution run is based on a random initial population and random mutations when creating children. Thus different results can be observed on different runs. In this section we presented several single evolution runs. We did not conduct a statistical analysis but the runs presented appeared typical.

Table 4. Evolution tracking changes in the non-stationary environment $\Phi_e(S,t) = 3\,\mathrm{round}(\sin\frac{\pi t}{20}) - S$.

Time t	**$\Phi_e(0,t)$**	**Lowest regularized loss agent**	**Its Δ'**
1	0	$\sqrt{0}$	0.2
2	1	1	0.1
4	2	$\left\lceil \frac{\lfloor 4 \rfloor}{\lvert 3 \rvert} \right\rceil$	0.6
7	3	$\left(\frac{(-5)}{3}\right)^2$	0.622

One can increase the cognitive ability of the evolution by enlarging its population. That gives the evolution more agents to choose from and thus a higher chance to find a compact Φ_a that yields P_{desired} as the fixed point before the environment changes again. If we treat the entire population as a single level-2 agent then enlarging the population can be thought of as increasing the speed of the agent's fixed point solver. Naturally to maintain a larger population, the A-life setting has to be modified accordingly so that the agents do not die out quickly, shrinking the population.

5. Current Shortcomings and Future Work

The simple computation model in which the space of agents is represented by a space of simple formulae does not enable individual agents to model the environment and solve for a fixed point. Thus, our individual agents were limited to level 1. We reached level 2 by treating the entire population as a single agent which allowed us to continually re-solve for a fixed point. However, even treating an evolving population as a single agent does not reach level 3 in which agents *predict* the environment. Future work will consider a richer language that can encode standard programming constructs (e.g., branching, loops, memory). Then one can investigate how a fixed-point solver can emerge naturally in the course of an evolution. Once an agent with a fixed-point solver emerges, it can adapt to novel situations in its environment *within* its lifetime. Here it will be of interest to see how such lifetime adaptation interacts with adaptation over generations during evolution. Past work showed such interactions to be non-trivial even in simple environments (Ackley & Littman, 1991).

The theory also postulated level 4 when a non-stationary environment changes so rapidly that even a level-3 agent is perpetually behind in its predictions and consequently its steady-state solutions. We suggested that such circumstances may give rise to refugia — ecological niches sought by the agents where the environment changes less rapidly, allowing the agents to keep up and thus survive. Our computational model presented above does not address this situation since our environment does not have a spatial localization and thus cannot feature faster and slower changing areas. Nor can our agents relocate to such areas as they also lack any spatialization. Future work will employ a richer A-life simulation where the agents can move about a heterogeneous envi-

ronment. Furthermore, machine learning methods can then be applied to detecting emergence of refugia automatically (Soares et al., 2018).

Finally, the idea that more complex cognitive abilities can first emerge at the level of population and then migrate into individual agents calls for further computational modeling. For instance, an individual agent can first run an evolution of behaviour policies $\Phi_a \in \mathbf{\Phi}$ in its mind and then implement the best policy in the environment. While such simulation of control policies in the agent's mind has been manually implemented in game-playing agents in the form of lookahead search since the early days of Artificial Intelligence, it is of interest to consider how the cognitive *ability* to conduct such an internal policy simulation can transition from a population into individual agents, in the course of an evolution.

6. Conclusions

The primary contribution of this paper is the proposal of a simple computational model to partially illustrate an existing abstract theory on the rise of cognitive complexity in agents as a response to more rapidly changing environments. The model represents agent behaviour policies as arithmetic formulae over a compact context-free grammar. We then run an A-life-style evolution of such formulae and observe individual agents reaching the preferred fixed point emerge in the evolution. When the environment becomes non-stationary our space of agent's behaviour policies is insufficiently rich for a single agent to track changes in the environment and re-compute a fixed point. However, treating the whole population as a single agent enables us to do so. This prompts the question of whether higher cognitive functions first emerged in a distributed form in a population before embedding themselves in individual agents.

References

Ackley, D., & Littman, M. (1991). Interactions between learning and evolution. *Artificial life II*, *10*, 487–509.

Bulitko, V. (2016). Evolving real-time heuristic search algorithms. *Proceedings of the International Conference on the Synthesis and Simulation of Living Systems (ALIFE)* (pp. 108–115).

Bulitko, V. (2020). Evolving initial heuristic functions for agent-centered heuristic search. *Proceedings of the IEEE Conference on Games (COG)*.

Bulitko, V., & Bulitko, V. K. (2019). Increase in agent complexity in non-stationary environments. *Proceedings of the Annual Conference on Advances in Cognitive Systems (ACS), Poster Collection* (pp. 1–9).

Bulitko, V., Carleton, S., Cormier, D., Sigurdson, D., & Simpson, J. (2017). Towards positively surprising non-player characters in video games. *Proceedings of the Experimental AI in Games (EXAG) Workshop at the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)* (pp. 34–40).

Bulitko, V. K. (1986). *Models of processes in industry, economy and ecology*. Naukova Dumka.

Calvin, W. H. (2002). *A brain for all seasons: Human evolution and abrupt climate change*. University of Chicago Press.

Eiben, A. E., & Smith, J. E. (2015). *Introduction to evolutionary computing*. Springer.

Graves, A., Wayne, G., & Danihelka, I. (2014). Neural Turing machines. *CoRR*, *abs/1410.5401*. From `http://arxiv.org/abs/1410.5401`.

Graves, A., et al. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, *538*, 471–476.

Koenig, S. (2001). Agent-centered search. *Artificial Intelligence Magazine*, *22*, 109–132.

Langley, P. (2018a). Planning systems and human problem solving. *Advances in Cognitive Systems*, *7*, 13–22.

Langley, P. (2018b). Theories and models in cognitive systems research. *Advances in Cognitive Systems*, *6*, 3–16.

Miikkulainen, R., et al. (2017). Evolving deep neural networks. *CoRR*, *abs/1703.00548*. From `http://arxiv.org/abs/1703.00548`.

Rogers, Jr, H. (1987). *Theory of recursive functions and effective computability*. MIT Press.

Silver, D., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*, 484–489.

Soares, E. S., Bulitko, V., Doucet, K., Cselinacz, M., Soule, T., Heck, S., & Wright, L. (2018). Learning to recognize A-life behaviours. *Poster collection: The Annual Conference on Advances in Cognitive Systems (ACS)*.

Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, *10*, 99–127.

Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., & Clune, J. (2017). Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567*.

Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction*. MIT Press.

Wilensky, U., & Rand, W. (2015). *An introduction to agent-based modeling*. MIT Press.