Applied Soft Computing 13 (2013) 1136–1146

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

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

Bacterially inspired evolving system with an application to time series prediction

D. Barrios Rolanía[a], J.M. Font[b,*], D. Manrique[b]

[a] Departamento de Lenguajes y Sistemas Informáticos e Ingenieria del Software, Universidad Politécnica de Madrid, 28660 Boadilla del Monte, Spain
[b] Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, 28660 Boadilla del Monte, Spain

ARTICLE INFO

Article history:
Received 5 October 2011
Received in revised form 19 August 2012
Accepted 22 October 2012
Available online 12 November 2012

Keywords:
Natural computing
Genetic programming
Bio-inspired computation
Synthetic biology

ABSTRACT

This paper explores the synergies between evolutionary computation and synthetic biology, developing an *in silico* evolutionary system that is inspired by the behavior of bacterial populations living in continuously changing environments. This system creates a 3D environment seeded with a simulated population of bacteria that eat, reproduce, interact with each other and with the environment and eventually die. This provides a 3D framework implementing an evolutionary process. The subject of the evolution is each bacterium's internal process, defining its interactions with the environment. The evolutionary goal is the survival of the population under successive, continuously changing environmental conditions. The key advantage of this bacterial evolutionary system is its decentralized, asynchronous, parallel and self-adapting general-purpose evolutionary process. We describe this system and present the results of an application to the evolution of a bacterial population that learns how to predict the presence or absence of food in the environment by analyzing three input signals from the environment. The resulting populations successfully evolve by continuously improving their fitness under different environmental conditions, demonstrating their adaptability to a fluctuating medium.

1. Introduction

Natural computing is a highly interdisciplinary field, composed of research areas covering topics ranging from biology, chemistry and physics to the development of mathematical algorithms and software applications [1]. Some of these areas focus on different aspects of nature as a source of inspiration for developing computational techniques [2–5]. Another approach to natural computing engineers biological systems in an attempt to understand nature by defining its constituent parts [6–8]. These engineered biological modules are then used to build fully fledged purpose-specific biological circuits [9–11]. Evolutionary computation and synthetic biology are disciplines that are representative of these two different approaches to natural computing. They come together to create a research field that applies evolutionary techniques to automatically develop *in silico* synthetic biological circuits [12].

Different evolutionary computation techniques can be used for this purpose. Insofar as DNA and RNA strands can be easily represented as fixed-size individuals of the population of a genetic algorithm, this tool is usually applied to automatically generate DNA and RNA structures that have a targeted behavior [13–18]. The evolution of complex biological circuits, such as regulatory networks involving genes, RNA and proteins that interact with each other through chemical reactions, requires the development of *ad hoc* evolutionary algorithms with codification methods to deal with variable-size individuals [19–25]. These evolutionary approaches are specially designed to meet the constraints of their application domain, but none have a crossover mechanism to evolve the population. In these cases the crossover operator suffers the closure problem [26]: it can generate invalid offspring from valid individuals. Consequently the crossover operator is replaced by a controlled mutation operator. This operator randomly performs certain specific mutations within the genome of the individuals in the population.

Additionally, these techniques apply a two-step evolutionary computation process to generate biological circuits, as is usual practice for automatically building intelligent systems [27]. An evolutionary technique is first used to create a specific intelligent system that learns to solve a fixed problem, the system is then installed in the application domain to solve a real-world problem. If the domain is updated with new information or the problem specifications change, the evolutionary program has to be re-run in order to adapt the intelligent system to the new conditions. This offline evolution philosophy clashes with continuously changing execution environment of biological systems, whose distributedness is more consistent with asynchronous and parallel embodied evolutionary algorithm execution [28,29]. The EVE system [30] takes

* Corresponding author. Tel.: +34 913366907; fax: +34 913524819.
E-mail addresses: dbarrios@fi.upm.es (D.B. Rolanía), jm.font@upm.es (J.M. Font), dmanrique@fi.upm.es (D. Manrique).

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.10.012

Table 1
Table of symbols used throughout the paper.

Symbol	Meaning	Symbol	Meaning
E	Environment	r	Radius
$\hat{B}$	Bacterial population	bp	Biological process
$\hat{S}$	Set of environmental signals	ε	Energy
$\mathcal{X}$	Set of states	rp	Response protein
R	Environmental resource (nutrients)	f	Fitness
t	Time	δ_{refill}	Resource refill rate
B_t	Bacterial population at time t	δ_{decay}	Resource decay rate
S_t	Environmental signals at time t	δ_{signal}	Signal decay rate
$[R]_t$	Resource concentration at time t	δ_{ε}	Energy decay rate
X_t	State at time t	$\sigma_{\text{immaturity}}$	Immaturity threshold
φ	Resources function	σ_{death}	Death threshold
b	Bacterium	σ_{quorum}	Quorum sensing threshold
s	Signal	Π_{death}	Death probability
c	Center		

this point into account and proposes an evolutionary framework to continuously self-adapt a population of individuals to a changing environment. Even so, it still includes centralized control mechanisms to maintain a fixed-size population and does not provide a physical environment to simulate its behavior.

This research takes inspiration from the behavior of bacterial populations [31] in order to create a bacterial evolutionary system that overcomes these drawbacks. This system creates a 3D environment seeded with a simulated variable-size bacterial population. The system simulates how bacteria move [32], grow [33], reproduce and die [34] as they do in nature [35], and they are allowed to interact with the environment and with each other. They execute an asynchronous, decentralized and parallel grammar-guided genetic program that evolves their internal biological processes. The population self-controls its size using an unsupervised and decentralized quorum sensingbased regulatory system [36]. Quorum sensing is an intercellular communication mechanism used by bacteria to shift between two states, representing a low and high population density respectively [37]. Communication is done by producing and sensing acyl homoserine lactones (AHL) signal molecules, also called autoinducers.

This bacterial evolutionary system is a general-purpose technique, because grammar-guided genetic programming (GGGP) assures that any intelligent system whose constraints can be codified using a context-free grammar can implement the biological processes of these bacteria [27]. The bacterial evolutionary system evolves variable-size individuals and, given that GGGP solves the closure problem [38], it is complete, because it includes initialization, selection and replacement operations – via cellular division and death –, as well as a conjugation operator that implements a grammatical crossover [39] by means of which bacteria share genetic material via conjugation [40].

Irrespective of the evolved intelligent system, the bacterial evolutionary system's goal is to assure the survival of the bacterial population through constant adaptation to the changing 3D environment. Population behavior improves continuously through self-adaptation to the constant changes taking place in its surrounding habitat. In this paper we present an application of this system, where bacteria implement a population of Bayesian networks [41] that manage their internal biological processes. These biological processes receive three environmental signals, s_0, s_1 and s_2, and output the expression level of a response protein that regulates the amount of resource that each bacterium harvests from the medium. The amount of existing environmental resource is regulated by time-delayed XOR and XNOR operations performed upon s_0, s_1 and s_2. This way bacteria evolve in order to learn how to predict the presence or absence of the resource by sensing and analyzing the quantity of s_0, s_1 and s_2. The BSim framework, developed by the Bristol Centre for Complexity Sciences [42], has been used to build and simulate the 3D environment. BSim is a modelling tool designed to allow for the study of bacterial populations.

This paper is organized as follows. Section 2 describes the simulation of the 3D environment and the bacterial population. Section 3 explains the components of the evolutionary procedure. Section 4 details the application problem and the results obtained by the evolutionary system, and Section 5 expounds the conclusions. Symbols used throughout this paper are listed in Table 1.

2. Using bacteria as inspiration

2.1. Simulating the habitat

In order to simulate the natural environment of bacteria, we define a biologically inspired 3D dynamic environment as a 4-tuple $E=(\hat{B}, \hat{S}, \mathcal{X}, \varphi)$, where:

1. $\hat{B}$ is a set called bacterial population.
2. $\hat{S}=\{s_0, s_1, \ldots, s_s\}$ is a finite set whose elements are called environmental signals.
3. $\mathcal{X} \subseteq \mathcal{P}(\hat{B}) \times \mathcal{P}(\hat{S}) \times \mathbb{N} \times \mathbb{N}$ is the set of states of system E, where $\mathcal{P}(A)$ denotes, for each set A, the power set (given by all subsets of A) and $\mathbb{N}$ denotes the set of natural numbers, that is, each state is a vector of size 4.
4. $\varphi : \mathcal{X} \to \{0, 1\}$ is a binary function called resources function.

We assume that the following properties hold for the set of states $\mathcal{X}$:

1. If $(B, S, [R], t) \in \mathcal{X}$, then for each $q \in \{1, 2, \ldots, t-1\}$ there exists $(B', S', [R'], q) \in \mathcal{X}$.
2. If $(B_1, S_1, [R]_1, t_1), (B_2, S_2, [R]_2, t_2) \in \mathcal{X}$ are two states with $t_1 = t_2$, then $B_1 = B_2$, $S_1 = S_2$, $[R]_1 = [R]_2$. In other words, there cannot be two different states with the same last entry.

In view of the above properties, the set of states can be construed as a finite or numerable sequence $\mathcal{X}=\{X_t : t=1, 2, \ldots, m\}$, where $m \in \mathbb{N}$ is a fixed number (eventually, $m=+\infty$) and each state is associated with a unique value $t \in \{1, 2, \ldots, m\}$.

In the sequel we assume that t represents the variable time in states $X_t=(B_t, S_t, [R]_t)$. B_t and S_t represent, respectively, the subsets of the bacterial population and environmental signals that are present at time t. $[R]_t$ is called the concentration of environmental resources of X_t. In this way, the set $\mathcal{X}$ describes the evolution of the given 3D dynamic environment.