From Ephemeral Computing to Deep Bioinspired Algorithms: New Trends and Applications

David Camacho, Raúl Lara-Cabrera

Universidad Autónoma de Madrid, Spain

J.J. Merelo-Guervós, Pedro A. Castillo

University of Granada, Spain

Carlos Cotta, Antonio J. Fernández-Leiva

Universidad de Málaga, Spain

Francisco Fernández de Vega, Francisco Chávez

University of Extremadura, Spain

David Camacho[1,], JJ Merelo[1,*], Carlos Cotta[1,*], Francisco Fernandez[1,*]*

Abstract

Ephemeral computing is a term that describes computing systems whose nodes or their connectivity have an ephemeral, heterogeneous and possibly also unpredictable nature. These properties will affect the functioning of distributed versions of computer algorithms. Such algorithms, which are usually straightforward extensions of sequential algorithms, will have to be redesigned and, in many cases, rethought from the ground up, to be able to use all ephemerally available resources. Porting algorithms to an inherently ephemeral, unreliable and massively heterogeneous computing substrate is thus one of the main challenges in the ephemeral computing field. Algorithms adapted so that they can be consciously running on this kind of environments require specific properties in terms of flexibility, plasticity and robustness. Bioinspired algorithms are particularly well suited to this endeavour, thanks to their decentralized functioning, intrinsic parallelism, resilience, adaptiveness, and amenability for being endowed with algorithmic components dealing with both the massive complexity of the computational substrate and that of the problem being tackled. Arranging these components and functionalities in a collection of algorithmic strata results in deep architectures, whereby different layers of optimization are organized into loosely-coupled hierarchies that not only are able to use ephemeral computing environments, but also profit from them by making adaptivity and diversity maintenance features of the algorithm. Moreover, the synergies that arise when these massively heterogeneous computing resources are made available to deep versions of bioinspired algorithms may enable hard real-world problems and ap-

*Corresponding authors

Email addresses: david.camacho@uam.es (David Camacho), jmerelo@geneura.ugr.es (JJ Merelo), ccottap@lcc.uma.es (Carlos Cotta), fcofdez@unex.es (Francisco Fernandez)

plications to be successfully faced in the Big Data context (including but not limited to social data analysis) as well as problems in the areas of computational creativity or computer gaming.

Keywords: Ephemeral Computing, Bioinspired Algorithms, Complex Systems, Deep Bioinspired Architectures

---

1. Introduction

When facing real-world problems, two major issues become crucial: to use problem-domain knowledge to adapt the problem solver to the task at hand, and to exploit available computational resources in the best way. These two issues can be regarded in an abstract sense as fighting the same dragon: complexity. This is very clear, even from a classical standpoint, in connection to the inherent difficulty of the problem, be it because of its size, the intricacy of the data and/or objectives or its potentially dynamic nature. The fact that they are facing complexity also becomes evident in relation to the latter aspect, that is, the use of computational resources, if we consider the increasing prevalence of interconnected techno-social systems composed of heterogeneous layers of resources with a complex dynamics, ultimately driven by human and social factors. These can for example take the shape of non-conventional computational platforms such as volunteer-computing environments, peer-to-peer networks or pool-based systems, just to name a few.

Referring to these non-conventional environments, the notion of *Ephemeral Computing* (Eph-C) [1, 2] has been defined as *"the use and exploitation of computing resources whose availability is ephemeral (i.e., transitory and short-lived) in order to carry out complex and possibly lengthy computational tasks"*. The main goal in Eph-C is thus making an effective use of heterogeneous resources with low availability and high volatility whose computational power (which –if harnessed– can be collectively enormous) would be otherwise wasted or underexploited; for instance, networked smartphones, tablets and, lately, wearables and sensors [3] –not to mention more classical devices such as desktop computers [4]– whose computational capabilities are often not fully exploited. There are some obvious connections to ideas from ubiquitous computing, volunteer computing and distributed computing, although Eph-C focuses more on the dynamics of the nodes and network and on the redesign of *traditional* algorithms to make them better suited to this kind of environment.

The effective exploitation of these platforms for problem-solving, as well as the effective resolution of complex problems requires the use of flexible, robust and adaptive algorithmic methods, capable of coping with the highly dynamic and volatile computational landscape and the massive complexity of the problem. Bioinspired algorithms are particularly well suited to this endeavor, thanks to some of the features they inherit from their biological sources of inspiration, namely decentralized functioning, intrinsic parallelism, resilience, and adaptiveness. In particular, they are prone to augmentation with self-control on their own functioning and/or structure. They are also readily adaptable to different optimization targets by embedding suitable problem-aware algorithmic components. This is crucial in order to tackle the study, analysis and optimization of techno-social systems, whose massive complexity often calls for a bottom-up

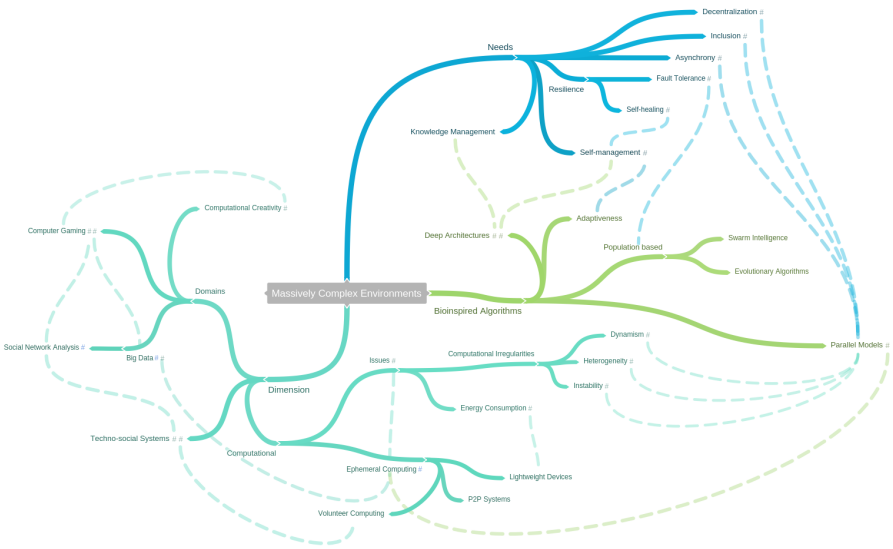

Figure 1: Road map for the use of deep bioinspired algorithms for tackling problems of massive complexity and/or deployment on ephemeral environments.

perspective, understanding the behavior of the system as emergent properties of the interaction of its constituents.

Some bioinspired algorithms such as evolutionary algorithms (EAs) fit nicely into this scenario. However, few works have previously considered the interest of adapting evolutionary algorithms by adding the capability for coping with transient behaviors in the underlying computer systems. Moreover, Big Data [5, 6] has nowadays become a standard issue in many initiatives, in which large amounts of computational resources are required for storing, processing, and learning from huge amounts of data. This underpins the need for managing (often heterogeneous) computing resources widely distributed along the world, a task for which new methods and algorithms are very much welcome.

In this context, and much like deep learning algorithms feature multiple processing layers to learn representations of data with multiple levels of abstraction [7], we can think of *deep bioinspired algorithms* (Deep-Bio) exhibiting multiple interconnected layers contributing the desired characteristics by encapsulating the tools required to tackle the different aspects of the complexity of the problem and the intricacy of the computational substrate, and whose interaction optimize the solving process. Throughout this paper we will elaborate on the need of these techniques, the major issues they have to face, their appropriateness for this purpose, and some of the most distinguished application areas for them, following for this purpose the road map depicted in Figure 1 in which the major themes in this area and their interrelations are shown. As an entry point, we shall begin with an overview of Eph-C in next section.

2. Ephemeral Computing: A Bird's Eye View

Traditionally, complex computational tasks were carried out in specialized devices designed for this purpose. The technological advance in which we are immersed allows us to expand these computing frontiers, since we can count on continuously connected device networks that are prepared to perform complex calculations. But these computing resources are associated with a limited time, since devices will not always be ready to be used. In addition to the association to a limited time, we must also face the heterogeneity of the devices that form the network [8]. All these issues transport us to what we know as "ephemeral computing", a paradigm that allows to perform complex tasks taking advantage of heterogeneous computational resources associated with a limited time. The effective exploitation of these resources requires us to study their capabilities as well as their limitations. Throughout this section we present a perspective of ephemeral computing as well as a study of the main issues at play.

2.1. Ephemeral Computing in Perspective

According to the Oxford Dictionary, the term *ephemeral* means "lasting for a very short time" Therefore, it encompasses things or events of a transitory nature, of brief existence. Current computing resources are endowed with this concept of volatility over time, such as computer networks based on portable devices [9]. We can also observe ephemeral behaviors in the way users work within computer networks [10].

Ubiquitous computing, voluntary computing or traditional research areas such as distributed computing encompass ephemeral phenomena and behaviors. The services offered by ephemeral computing are commonly associated with executions between autonomous heterogeneous parties in dynamic environments. Therefore, ephemeral services are commonly seen more as a problem than as a solution [11].

We can, for instance, leverage the main advantage of ubiquitous computing, the availability of computing resources anywhere and everywhere, which allows us not to depend on the device, but it entails the difficulty of the heterogeneity of the components. Most efforts in this area are directed towards the design and development of the underlying technologies needed to support ubiquitous computing [12], such as advanced middleware, operating systems, mobile code, sensors, microprocessors, new I/O and user interfaces, networks or mobile protocols. The main objective of ubiquitous computing is to enable stable and persistent computing processes to fully execute programs. When this area handles the concept of ephemeral devices, services or computing, the main solution used so far is to stop the process(es), and resume once the new devices are ready [13].

If we focus on the concepts commented above, we should bear in mind that the focus of Eph-C is completely different, as instead of trying to build layers in the ephemeral resource network to hide their transient nature and provide the illusion of a stable virtual environment, Eph-C applications are fully aware of the computational nature they face and are specifically built to live (and optimize their performance) in this area. Note that this does not imply that the latter have a low level view of the underlying computational substrate, or at least not markedly. In fact, most of the low-level traits of the latter can be

abstracted without excluding the possibility of obtaining a clear picture of this ever-changing environment.

It is worth noting that the issues discussed are present (at least partially so) in areas such as volunteer computing (VC) [14], in which massively large computational problems are broken down into small chunks that are subsequently distributed for individual treatment on a dynamic collection of computing devices. This kind of approaches are mostly centralized, typically following a master/slave scheme, and resort to redundant computation in order to handle the volatility of computational resources. This contrasts with the much more decentralized, emergent approach that amorphous computing [15] provided, although it must be noted that this latter paradigm is more geared towards programmable materials and their use to attack massive simulation problems. Precisely, massively large problems are also central in ultrascale computing, in which issues such as scalability, resilience to failures, energy management, and handling of large volume of data are of paramount importance [16]. This said, notice that Eph-C is not exascale nor it is oriented towards supercomputing (or at the very least, not necessarily so).

2.2. Major Issues in Eph-C

Eph-C deals with the use of ephemeral computing resources in the implementation of (mainly) bioinspired algorithms. Very often, ephemeral computing systems have a techno-social nature [1, 17, 18]. Hence, its dynamics and the analysis thereof usually includes a human-in-the-loop, since the decision of becoming a part of some ephemeral computing systems is taken by a human depending on a number of factors [19].

Due to the existence of multiple devices and human agents, Eph-C systems show emergent behavior, and stand at the edge of chaos [20]. This produces power-law distribution in the amount of computation that is allocated by each user [10], as well as a positive feedback loop between the entropy in the arrival of new users and the performance of the whole system [21]. Complex systems cannot be controlled, but they can be driven by changing the behavior of the connected parts; for example, a gamification strategy can tip the balance and make the users more engaged [22]; however, this strategy cannot be exclusive, since any hurdle to the participation in an Eph-C environment can, in fact, reduce the performance of the whole system.

As stated before, a salient feature of ephemeral environments is their volatility: their constituent computational units are unstable and can cycle from available to non-available in response to uncontrollable external factors. The term *churn* is used to denote the effect on the system as a whole of this constant arrival/exit of computational nodes. Churn causes the computational landscape to be ever-changing, and algorithms running on this kind of systems must face this issue, which impacts them in different ways, e.g., loss of information and communication disruptions. Resilience –among other properties– is then a must for any algorithm running on ephemeral environments, as we will elaborate upon in next section.

The potential heterogeneity [23, 24] of computational nodes (which could very well encompass from tiny sensors to desktop computers) is another important issue in this context (and as a matter of fact of high-performance computing in general). This is often dealt with by adequately balancing the load of the different computing units [25] and finding an appropriate distribution of data

among processors [26]. The situation has its own particularities in the case of optimization approaches since more often than not there is not a certain fixed computation to be performed in minimal time, but a variable computation whose outcome can have different quality to be maximized [27].

The study of self-$\star$ properties [28] in this kind of system is also an interesting challenge, since very diverse such properties emerge as a result of the interaction between ephemeral devices and users. It is as part of this self-adaptive strategy where factors such as energy consumption come into play. Energy- (or, in general, context-) aware algorithms will become increasingly important when energy or any other computational cost measure come into play [29].

3. Deep Bioinspired Algorithms

Bioinspired algorithms use biological metaphors for describing search and optimization methods, generally based on the use of an evolving population or pool of solutions. In this context, we use the term *deep* to indicate several levels of optimization/interoperation, in the sense that there might be algorithmic components working on other kind of algorithms, and so on in different layers of search and optimization. Deep architectures of bioinspired methods can thus stack a plethora of components capable of tackling different features of the computational substrate or of the target problem in multiple scales.

In the rest of the section we will proceed to justify the choice of bioinspired algorithms to work in ephemeral environments to continue with the need and usefulness of using deep architectures (in subsection 3.2).

3.1. Bioinspired Algorithms as the Weapon of Choice

There are several reasons why bioinspired algorithms seem particularly suited to ephemeral environments. The main one is that most of them are based on evolving a population of possible solutions, coding problem solutions as individuals which –depending on the paradigm–, can be either considered chromosomes (in evolutionary algorithms), particles (in particle swarm optimization) or even ants (in ant colony optimization algorithms). This means that they can be easily distributed by simply spreading the population among different computing nodes. The challenge is how to do it in an algorithmically efficient and scalable way in the particular context we are considering.

In this sense, ephemeral environments add several dimensions to the design of algorithms, besides the obvious fact that the contribution of a node might come and go at any time:

- *Inclusion*: the computational system will have a dynamic structure in which all nodes should have a meaningful contribution to the final result. However, since ephemeral environments include varying ranges of computing power, the contribution of every individual node to the final solution might be small and in any case difficult to measure. The challenge is always to include all kind of nodes in a way that it does not create a bottleneck.

- *Sustainability*: As anticipated in a previous section, energy consumption can be a major issue in Eph-Csystems. Given that different algorithmic parameterizations and design decisions can influence the time to find a

solution and even on-line energy consumption, these must be approached in such as way that the system can sustain the computation.

- *Asynchrony*: nodes communicate with each others without a fixed schedule due to their different node capabilities and the fact that they can come or go at any time during the experiment.
- *Resilience*: the sudden disappearance of computing nodes or communication links must not destabilize the functioning of the implementation of the algorithm, and even less so the algorithm itself.
- *Emergence*: the nature of the computational environment does not allow a centralized control and requires decentralized, emergent behavior. This emergent behavior arises from the interaction among the ephemeral components and might include better scaling due to the diversity that the environment lends to the algorithm implementation.
- *Self-adaptation*: the algorithm should adapt itself to the changing computational landscape; this could include self-scaling, for instance, but also self-adaptation of the parameters every ephemeral node is using.

This latter issue is particularly important, and encompasses a number of self-$\star$ properties [28] that the system must exhibit in order to exert advanced control on its own functioning and/or structure. This is a salient feature that will be discussed more in depth in next subsection. As for the remaining ones, bioinspired algorithms remarkably exhibit them in an intrinsic way or can be readily endowed with them. Indeed, bioinspired algorithms are resilient optimization techniques. For example, master-slave parallel models can withstand a reduction in computing resources of up to one order of magnitude [30, 31]. While the situation is different in coarse-grained models such as island-based bioinspired algorithms (in which the loss of a computing node might result in the disappearance of the current incumbent solution [32] and would impair genetic diversity and the progress of the search), it is not difficult to use fault-tolerance techniques such as checkpointing (i.e., saving periodical snapshots of the state of volatile nodes) [33, 34], or endow the algorithm with self-healing techniques [35], as discussed in next subsection.

One of the causes for the resilience of bioinspired algorithms is the use of populations of solutions, which give them built-in redundancy and intrinsic decentralization. It is precisely by the lack of central command (at least in island-based or diffusion-based models) that they can be readily deployed on environments exhibiting plasticity and dynamism in topology and structure. Also as a consequence of this, they do not require synchronization barriers and can naturally adapt to asynchronous functioning [36]. Let us also note *en passant* that the raw material of evolution (i.e., the genetic variance of traits) has been acknowledged as intrinsically ephemeral [37], hence underpinning the amenability of some bioinspired models for working with transient information units. Lastly, bioinspired algorithms have been shown to work on a parameter space that tends to be rife with viable parameters [38], thus implying that there is ample room to adjust these taking into account different issues, such as energy consumption, while maintaining acceptable performance. Of course, finding the best combination of parameters or even joining them together in a workflow to process complex payloads might still be a challenge. Hence the usefulness of

self-management components and ultimately of deep architectures as explained below.

3.2. Deep Architectures of Bioinspired Algorithms

Drawing from the analogy with deep learning, deep bioinspired algorithms can be understood as multilayered bioinspired techniques for problem solving in which each layer (or more generally speaking, each component, since one can conceive complex architectures in which no well-defined layered hierarchy is present) deals with a different facet of the problem under consideration or even with different features of the computational environment. The first issue can be readily connected to the broad interpretation of memetic algorithms [39], in which optimization is approached via the orchestrated interplay of components from population-based and local-search metaheuristics. This paradigm allows encapsulating different pieces of problem-knowledge into different algorithmic components, which are subsequently put together aiming to a synergistic functioning. The most classical approach is to embed a local-search component within a global (typically evolutionary) search method [40], an approach that gained popularity and went on to be taken as synonymous to memetic algorithms in the so-called narrow interpretation of the latter. However, it is not uncommon to see memetic approaches that exploit other heuristic add-ons, such as complete techniques, cf. [41] or greedy randomized adaptive search procedures [42], just to name a few.

This idea is also the central tenet of memetic computing [43], in which the focus is on the harmonic coordination of complex computational structures composed of interacting modules (i.e., memes) for problem solving, whose representation is stored and manipulated by the algorithm itself. Therein, the explicit management of memes, that were only implicitly defined by the choice of heuristic add-ons in more classical memetic approaches, is therefore central. Such a treatment was already anticipated in the early stages of the paradigm [44] and put to work in the early 2000s [45] within the so-called multimeme algorithms. This is a theme that to a certain extent is also present in other approaches such as hyperheuristics [46] (in which a high-level heuristic layer is used to coordinate the application of low-level heuristics) or asynchronous teams [47] (in which a computational network of diverse agents -–encapsulating different problem-solving skills-– cooperate).

It is not unusual to have a learning component into the loop so as to modulate the dynamics of the system. In this direction, there exist the notion of memeplex [48] as a collection of co-adapted memes interconnected in a dynamic network. Therein, memes self-organize and engage in collective learning in order to improve the lifetime learning process. Indeed, adaptation and —going beyond— self-adaptation are particularly important in this context in order to alleviate the design effort, boosting the optimization capabilities of the algorithm, or making it more resilient to the glitches in the computational landscape.

As anticipated in the previous subsection, such self-adaptation –and more generally speaking, any form of self-management– can be accomplished via the incorporation of self-$\star$ properties, whereby the system can exert advanced control on its own functioning and/or structure. Fortunately, bioinspired algorithms are very much amenable to self-adaptation [49]; as a matter of fact, self-adaptation has been a part of the paradigm since the 1970s, e.g., in evolution strategies. Relevant examples of self-management properties in this con-

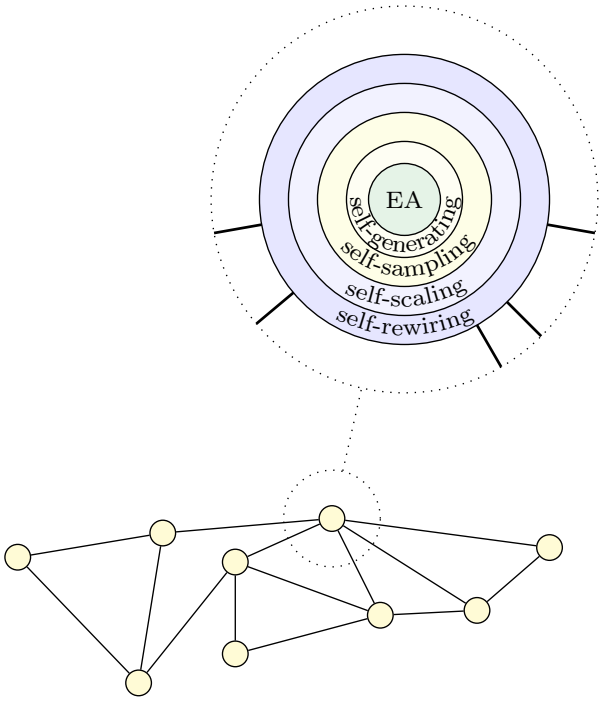

Figure 2: Example of a particular distributed bioinspired algorithm with deep architecture. In this example, each node in a complex network is a basic evolutionary algorithm with different layered self-⋆ properties.

text are, e.g., self-maintaining in proper state, self-healing externally infringed damage [35, 50], self-adapting to different environmental conditions [51], and even self-generating new functionalities just to cite a few examples [52, 53], see also [54, 55]. Figure 2 shows an example of an island-based bioinspired algorithm (adapted from the approach presented in [35]) in which these properties are layered: an external self-rewiring (a form of self-healing) layer tries to keep the connectivity of the node; an underlying self-scaling layer uses existing connections to balance out the size of the population with neighboring nodes; this balancing process may eventually require the use of self-sampling (another form of self-healing) to enlarge the population; at the deepest level sits an otherwise standard evolutionary algorithm, augmented in this case with an external self-generation layer to handle meme evolution. Notice that these properties work in this example at a local level, as opposed to the result of a central monitorization process, which is an additional advantage since decentralization makes the system more resilient and fault-tolerant, cf. [56].

Needless to say, other architectures different to the example in Figure 2 are possible, using other bioinspired algorithms (e.g., swarm algorithms) or hybrids thereof at the bottom level, and not necessarily using explicit meme management. Indeed, all different metaheuristic paradigms mentioned before are prone to result in deep architectures (although of course, one can conceive simple instances of such approaches in which the architectural depth can be regarded

as negligible). Much like it happens in deep learning, one cannot arbitrarily define a depth threshold above (resp. below) which the resulting technique is deep (resp. shallow). This is something that has to be defined on an individual basis in each application domain or even in each problem instance (next section discusses some applications and deep approaches to them). It is nevertheless important to notice that in the context described in this work, depth can be not just a result of the need of exploiting knowledge about the problem being solved, but also a requirement for deployment on certain computational environments, as well as a typical consequence of the integration of self-management capabilities.

4. Areas of Application

From a practical perspective, the application, development or even deployment of any algorithm that could be executed on massively complex settings will need to take into account those features described in Section 3.1. Therefore, such an application will generate some interesting challenges and opportunities that can be tackled. This section provides a short revision on some of those applications that are rife for deep bioinspired algorithms.

4.1. Big Data

Currently, the generation of data has been estimated as 2.5 quintillion bytes per day[1], and its volume continues to grow exponentially. Furthermore, as data come from many different sources, there is a high level of heterogeneity, which is considered as an issue to be tackled [57]. This problem becomes more evident when data are part of Eph-Csystems, since this is an obstacle to interoperability [58] in addition to the ephemeral nature of the data itself [59], making it necessary to deal with data and processing units that can disappear at any given moment [60]. As the amount of data rises, classic data mining and knowledge extraction techniques from the state of the art are not able to cope with it properly.

Deep learning approaches [7] have shown that combining several levels of interoperation –using a layered approach in this case– yield good results when dealing with high dimensional data. As described in Section 3, Deep-Bio approaches combine many components of interoperation/optimization to make profit of the synergy between them. Hence, deep bioinspired algorithms seems to be good candidates to deal with big data, not only for optimizing other algorithms and its parameterization, but also for processing and extracting knowledge from data.

Regarding the former use case, several lines of research have been initiated. EvoDeep [61] is an evolutionary approach and an optimization framework for deep neural networks parameterization. It evolves the parameters and the architecture of this kind of neural networks to maximize its classification accuracy and maintaining a valid sequence of layers at the same time. Another bioinspired technique has been used to optimize an algorithm in [62]. In this case, the authors proposed a back-propagation neural network method based

[1] http://www.vcloudnews.com/every-day-big-data-statistics-2-5-quintillion-bytes-of-data-created-daily/ (last access 22th March 2018)

on MapReduce and a particle swarm optimization algorithm to optimize the initial weights and thresholds of the network, reducing the impact of sample diversity and improving the performance. As illustrated by these and other examples [63, 64, 65, 66], bioinspired algorithms are promising candidates for handling optimization problems involving Big Data which, in turn, might be seen as Deep-Bio systems with several layers of varied algorithms.

On the other hand, other lines of research focus on combining bioinspired algorithms with other big data mining techniques instead of just optimizing the latter. For instance, an ant colony optimization (ACO) algorithm combined with spectral-based methods are the core of SACOC and SACON [67], two algorithms devoted to online clustering on dense datasets. This time, the bioinspired algorithm (i.e., the ACO) is involved directly in the process of knowledge extraction. The MACOC algorithm [68], extends the classical ACOC algorithm [69] including an ACO into a new clustering (medoid-based) approach.

As previously described, big data and its heterogeneity is best suited to Deep-Bio systems. Social media systems and applications have experienced a remarkable growth in the last decade, generating a massive amount of heterogeneous data [70]. The problems related to the efficient representation of knowledge and the management and discovery of patterns in large networks conform the main challenges under study in Social-based analysis and mining area [71, 72, 73]. Hence, it is worth to study the development of this kind of algorithms as well as its deployment on systems that exhibit Eph-C features to deal with the aforementioned problems of efficient representation and pattern discovery. To establish a starting point from which begin to improve the obtained results by incorporating Deep-Bio techniques, below is shown a brief survey on social network analysis methods.

Social Data Analysis, which comprises areas such as social-based applications for data mining, data analysis, community detection or social mining among others, has received an increasing attention from the research community [74, 75, 76, 77, 78, 79]. *Social Data Analysis* is inherently interdisciplinary and spans areas such as data mining, machine learning, statistics, information retrieval, natural language processing, semantic web, and big data computing, amongst others. Furthermore, their applications range across a wide number of domains such as health [80], counter terrorism [81] and social network analysis [82, 83]. The area itself, and the main problems and issues that must be solved, are directly related to those kind of massive complex problems, which are the main focus of this work.

Regarding the relationship between bioinspired algorithms and social data analysis, several approaches have been followed [83, 84, 85], specially on clustering, a technique to find hidden information or patterns in unlabeled datasets, e.g., [86]. Some of these approaches are based on bioinspired methods, in which two main types of algorithms can be distinguished: those that combine evolutionary techniques [87], and those that use swarm intelligence approaches (e.g. ant colonies optimization algorithms, artificial bee colonies, particle swarm optimization) [67, 69, 88, 89]. For a deeper insight, Hruschka et al. [90] provide a complete review on evolutionary algorithms for clustering.

Taking into account the aforementioned research works and the current challenges imposed by Big Data problems, it is worth to explore the combination of Big Data and deep bioinspired algorithms as a new and promising research topic (see Table 1). The design of novel Deep-Bio methods might help to an-

Table 1: Some deep bioinspired approaches to Big Data knowledge discovery and social networks

Knowledge extraction	Artificial neural networks	[61, 62]
	Big data optimization	[63, 64, 65, 66]
	Data mining	[67, 69]
Social networks	Social big data	[74, 75, 76, 77, 78, 79]
	Social network analysis	[82, 83, 84, 85]
	Clustering	[87, 88, 89]

alyze and solve, in a distributed way, problems related to this area. Using a deep approach could help to use different kind of both bioinspired and non-bioinspired algorithms to exploit their strengths, in a same way as memetic algorithms or the hybridization between Genetic Algorithms and ACOs with constraints-satisfaction problems (CSP) solvers.

4.2. Computer Gaming

Gaming is a term that we employ here to refer to both videogames, from a general perspective, and gambling, as a specific type of multiplayer online videogame in which players must wager money. Nowadays, gaming constitutes the leading industry in the digital entertainment sector, and one of the main reasons that has provoked its success is the growth of the use of mobile devices to execute videogames. In the context of (perhaps, but not necessarily, massively) multiplayer online games, some of the major issues mentioned in Section 2.2 arise. So, many players access the game through mobile devices that are connected to wireless networks and thus they are exposed to the problem of the disconnection of the game. This can be the result of multiple causes such as a battery depletion or a network failure, to name a couple. The consequence is that the player leaves the game. And this is a huge problem that the industry can not afford. Note that players continuously interact with other players in the game and the sudden disappearance of a player causes an enormous decrease in the feeling of reality (one of the biggest known problems in the gaming experience of the user). Therefore, in gaming it is not only important to manage appropriately the ephemerality of the resources in a network of mobile devices (and its associated problems mentioned in Sect. 2.2 such as energy consumption and the churn, among others) but it is also necessary to manage the ephemeral game resources. In this latter sense, human players can be considered ephemeral resources as they frequently join and leave the multiplayer online game. Moreover, human players can decide to disconnect for their own will, and the consequence is the same as if the disconnection is due to 'technical problems'.

In general terms, multiplayer online games also exhibit a transitory essence in the sense that the execution of the game is influenced by the actions of the players and these has to face the consequences of their acts. This means that the actions of a player influence the universe of the game and thus affect other players. Goals, players' alliances, and even rewards have to be rearranged according to the game progress. In this situation it does not make sense to offer the possibility to save the current state of the game as the actions of the player are not reversible. In other words, the game is continuously being executed. And this means that the (possible sudden) disconnection of a player causes a

sensation of unreality as it represents a disconnection in real time. Eph-C is clearly a paradigm that can alleviate the dire consequences of this problem.

In this context, a multiplayer online game can be viewed as an instance of an ephemeral environment, and thus, it should own some of the features mentioned in Sect. 3.1. Specifically, this kind of games should exhibit resilience (i.e., fault-tolerance) and self-adaptation. The first issue means that the sudden disappearance of a mobile device connected to the game network must not destabilize the functioning of the game. The latter issue implies to deal with the generation of virtual players that should replace the human players that suddenly disconnect from the game. The main difficulty of this process is that the virtual player should perform in a similar way to the disconnected player to avoid the lost of the feeling of reality in the rest of human players. This problem involves player modeling as well as procedural content generation (PCG) – considering game artificial intelligence (AI) as a content of the game. Bioinspired evolutionary algorithms, and evolutionary algorithms in particular, are one of the main techniques that have been used in the scientific literature to create this kind of game AI [91]. However, the ephemeral nature of the resources (i.e., physical resources – in form of mobile device – as well as logical resources – in form of the human player) adds an extra dose of complexity. If we consider each of these ephemeral resources as a layer to manage and having into account that evolution might be needed to generate believable virtual player, Deep-Bio seems an adequate technique to be used here.

Another issue to consider is the huge number of data that massively multiplayer online games (MMOG) generate (and have to manage) during their execution. This basically requires the management of Big Data, and thus the use of Deep-Bio can be defended here as done in Sect. 4.1.

Note also that the technical creation of a videogame can be seen as a multistage process in which there are three main actors: designers, programmers and graphic artists. And the automated creation of content for videogames via procedural content generation techniques [92], is being more and more important. The industrial benefits, economically speaking, are clear: PCG can reduce development costs and enlarge the life of commercial videogames (with the corresponding earnings increase). For this reason, one can find many proposals for the procedural content generation in games although most of them are focused on the generation of a specific type of contents (e.g., maps o non-player characters strategies). Recently, in [93], we have proposed coevolved via a competitive approach two different kind of contents, namely maps and game AI at the same time. The proposal consists of a coevolutionary competitive bioinspired algorithm that can be seen a a two-layer schema in which information from one layer can be used in the other one. Each layer self-adapt to be competitive with respect to the other layer. One can then think in adding more components to create by incrementing the number of layers (e.g., a layer to manage game rules or aesthetics features of the game, just to name a few). Deep bioinspired algorithms, understood as multilayered bioinspired techniques, seem to be perfect to deal with this challenge. The idea of associating distinct layers to the creation of different types of content in the design process (e.g., levels, user interfaces, or mechanics, just to name a few), the programming phase (e.g., game AI or gameplay, among others) or the creation of graphical elements (e.g., 2d/3d content, animations, user interface elements, illustrations, and many others) is valid here, and might even be considered for the generation of whole games in

Table 2: Some deep bioinspired approaches (or related) to Computer Gaming

Techniques	Neuroevolution	[94, 95]
	Deep Learning	[96, 97, 97]
	Evolutionary strategies	[98]
	Progressive neural networks	[99, 100]
	Interactive methods	[101, 102]
Areas addressed	Procedural Content Generation	[93, 103]
	Learning	[93]
	General Video Game Playing	[99, 104, 105]

which there is a layer associated to each of the technical steps involved in the development of a game. Moreover, each layer might be stratified in a number of 'deeper' layers with distinct sub-objectives that might be composed to reach the global aim of each specific layer (e.g., create believable virtual players or beautiful games, among others).

The application of 'deep' techniques to the videogame development is not a misplaced idea and recently some proposals (that demonstrate an increasing interest in this issue) has been launched (see Table 2) , although not necessarily in the context of bioinspired algorithms. So, the generation of game content using machine learning models is attracting attention nowadays [96]. The near future will provide a number of proposals for the automatic generation of contents in games based on neural networks, deep convolutional networks, Markov models, reinforcement learning, n-grams, and deep learning in general, just to name a few. Moreover, a number of methods that operate in several scales (i.e., layers) has already been proposed, as for instances distinct variants of Q-learning (e.g., Deep Q-Network, Deep Recurrent Q-Learning or Dueling Deep Q-network) were applied to Arcade Games [97].

More related to Deep-Bio we can find some proposals that combine deep learning with evolutionary approaches. One of the most interesting approaches is that of neuroevolution [94] that basically consists of a number of networks (possible coded as artificial neural networks) which are trained trough evolution. Here, The evolutionary method might be seen as a layer that covers the different artificial neural networks (ANNs) under evolution and each ANN as a deep layer inside the global architecture. Moreover, neuroevolution might be naturally extended with layers aimed to deal with issues such as self-adaptation (at the level of the whole model or at the level of the ANNs) or self-scaling, just to name a couple. Also, [95] proposes to combine deep learning and neuroevolution to create a bot for a simple first person shooter (FPS) game capable of aiming and shooting based on high-dimensional raw pixel inputs. Evolution strategies to train neural network policies for controlling robots in the MuJoCo physics simulator have also been described in [98].

4.3. Computational Creativity

Any creative process requires a number of actors if an appropriate solution is pursued: when human artists work, they do not work alone; a network of "agents" is required to evaluate their work. Thus, art galleries, museums, critics, auction houses and the public are part of a complex network where some of the features mentioned above can be analyzed. This is also the case when it comes to computer-generated art and computational creativity that may be displayed.

Table 3: Some deep bioinspired approaches to Computational Creativity

Computational Creativity	Beginnings	[106, 107, 108]
	Techniques	[106, 107, 108, 109]
	Interactive IEAs	[110, 111]
Areas addressed	Art, Design	[112, 113]
	Music	[111, 114]
	Narratives and Poetry	[115, 116]
	Cooking	[117]

The beginning of computational creativity dates back to the 1960s, when the Cybernetic Serendipity exhibition in London showed some earliest works of computer generated art and music[2]. Although generative art encompasses a wide range of techniques and materials [106], it was soon noticed the potential of computer science in the arts. Since then, the confluence among different but related areas has given rise to a number of techniques and algorithms applied to generate computer based generative computer-based works of art, including Fractals, Neural Networks, Markov Models, Evolutionary Algorithms, to name but a few [108, 107].

Computational creativity has reached popularity in the last decade [118]. Moreover, some of the methods available has gained momentum and attracted the attention of audiovisual industry [119].

The idea behind computational creativity is not to replace human artists by computer algorithms when confronted with creative processes. In reality, we have not seen yet a creative algorithm capable of passing the Turing Test for the arts [112]. Instead, computers need the help of human artists, and a specific algorithm has emerged that provides the perfect medium for including human actions within the main creative algorithm: the Interactive EA (IEA) [120].

When Evolutionary Algorithms are applied, the main evolutionary loop can be altered, so that users are allowed to interact within the algorithm. Their specific activity is narrowed to the essential and fundamental aesthetic evaluation. This small but key change triggered the emergence of this new research area.

IEAs have allowed the development a plethora of previously unseen quality works of art in many different fields: music composition [111], video-games plot induction and story generation [115] or automatic poetry [116]. As seen in the previous section, also PCG benefited from IEAs.

In any case, if we want to improve applications of available IEAs to creative processes a better understanding is necessary of the human creative process itself, and this involves studying human creativity from the point of view of the IEAs, but also considering the interaction of different roles in the creative process: artists, public and critics [113]. Therefore, a hierarchy of connections should be analyzed and contextualized, which are especially relevant when applying deep EA models to creativity. But even when the focus is only over the artists way of working, ephemeral models and their properties must be analyzed when team of artists -human or artificial ones- collaborate when developing art

[2]Cybernetic Serendipity Exhibition: http://cyberneticserendipity.net. Accessed on March 2018

applying evolutionary approaches: highly asynchronous processes; completely distributed and frequently isolated way of working with some interaction along the work: artists work alone in their particular "atelier", and sometimes share their ideas in collective ephemeral activities, such as public exhibition. Interaction with the audience and critics when a new work is exhibited.

Although some authors have already referred to deep versions of the IEA [102], this complete hierarchy of interactions among agents-roles, although required, has been scarcely considered for designing new version of the algorithm. Usually standard IEAs only take into account the role of human beings as fitness evaluators, and the deep series of relationship among artists, audience and critics are lost. In addition, human nature has a number of drawbacks that prevent algorithms from showing their full potential. We can mention human fatigue as one of the main problems leading to an undesirable ephemeral phenomenon: the time of availability of the "fitness evaluators" throughout the experiments.

The problem of human fatigue that is present in IEAs -due the multiple interactions required from the user in charge of aesthetic evaluation- can make it difficult to collect useful data. This well-known problem is already addressed by IEA researchers [109], and solutions often require high computational costs. And this is typically the case: large amount of data and computing infrastructure is required to induce computer-based creative processes.

In-line with the the above referred perspective, a new proposal based on EAs has recently been proposed in order to analyze creativity developed by human artists [113]. Results, have provided clues that may lead in the future to new genetic operators or algorithms. But even so, it requires the participation of as many human artists and people in the audience as possible to properly model human creativity. Results obtained by analyzing human behavior in the context of the algorithm have demonstrated its capacity of self-reflection and adaptation to the circumstances that may arise thorough the experiment [113]. We may notice that this features are part of the deep model described in previous section, where a hierarchy of self-property layers covers the core of the bioinspired algorithm.

Something similar may be observed if we focus on other creativity related areas of applications, such as music (problems such as music transcription, 4-part harmonization, automatic composition, or analysis can be addressed), where high computational costs are associated with computer-based creative processes in this area [114] as well as hierarchies of properties associated to the algorithms and ephemeral properties of the underlying hardware infrastructures must be taken into account.

Therefore, ephemeral behavior is inherent not only to the computer infrastructure we may use to run experiments, but also to the way artists work and react to colleagues and the public. Previous models [113] could be thus studied from this point of view to improve existing methodologies.

Summarizing, we notice how computational creativity researchers in the context of EAs need to consider hierarchical and deep relationships among the components involved in the algorithms. On the other hand massive data can help to train a computer-based learning model, so that after the learning process the algorithm is able to show creativity. And finally, creativity may be exhibited not just in the areas referred above, but in any possible context (check for instance [117], a data-driven approach is demonstrated through a computational creativity system for culinary recipes which can operate either autonomously or

semi-autonomously with human interaction).

5. Outlook and Future Trends

The current development of disciplines such as the Internet of Things and Big Data has originated a new landscape that provides a large number of massive and complex sources of data; features such as heterogeneity, dynamism or extremely large volumes of data can be used to define the basic problems that must be tackled by any algorithm or system in this context. To face real-world problems in this kind of highly complex domains, it is necessary to be aware about how to use problem-domain knowledge, how to adapt the problem solver to the task at hand, and how to exploit all available computational resources in the best way.

This work should be viewed as a revision of the current state of the Ephemeral Computing (Eph-C) as well as an introduction to the concept of Deep Bioinspired Computing (Deep-Bio) research areas. One of the primary objectives has been to show the close relation that can be constructed between these two paradigms. A secondary goal was to illustrate their applicability and, thus, we have focused on three specific areas of applications mentioned before. Of course, other sectors can benefit from both Eph-C and Deep-Bio. This work is also aimed at encouraging other interested researchers to find new methods and applications of these two paradigms, working together synergistically or separately. Indeed, there are many open issues and challenges to be tackled in this field.

The first research area, Ephemeral computing (Eph-C), has been described in terms of those computing systems whose nodes or their connectivity have an ephemeral, heterogeneous and unpredictable nature. We have analyzed the main features that can be used to clearly state Eph-Cas a new paradigm, allowing to perform complex tasks by taking advantage of heterogeneous-limited computational resources. While Eph-C presents some characteristics that are close to volunteer and amorphous computing, the combination of their main features, namely *inclusion, asynchrony, resilience, emergence, and self-adaptation*, defines it more precisely and distinguish it from previous ones though, making Eph-C a concept that includes both of them. However, Eph-C emphasizes infrastructure, its dynamics and its influence on the performance of algorithms; these kind of challenges have to be carried to the definition of new algorithms;

The second area, deep bioinspired algorithms (Deep-Bio), is defined as a methodological approach built on top of varied capsules of problem-knowledge arranged into different algorithmic components. A Deep-Bio algorithm can thus be understood as a multi-layered bioinspired technique for problem solving, in which each layer (or component) deals with a different facet of the problem under consideration, or even with different features of the computational environment. These algorithms are presented as a powerful choice to tackle the issues derived from highly complex domains, including those focusing on Eph-C properties and infrastructure. Some approaches, as memetic algorithms, hyper-heuristics algorithms, pool-based algorithms, or asynchronous teams, are presented as adequate frameworks to develop Deep-Bio methods.

Although there are countless applications that can benefit from the results that may be obtained with the combination of Deep-Bio and Eph-Cwe focused on three areas that have already been built in recent years from the perspective of ephemeral systems. Hence, specific examples of algorithms and methods

designed to tackle over massively heterogeneous and complex domains have been described over a subset of relevant domains of application. This way, several big data and social data analysis problems have been studied, including approaches focusing on the combination of evolutionary strategies and deep learning (EvoDeep), and the hybridization between genetic algorithms and ant-colony systems (SACOC, SACON, MACOC). Computer gaming and their associated problems, such as developing and managing multi-player online games, have been also analyzed as an instance of an ephemeral environment problem. Other potential application of Deep-Bio in this gaming context is the procedural generation of contents stratified in distinct layers, where a number of variants of deep learning methods (as for instance variants of Q-learning such as Deep Q-Network, Deep Recurrent Q-Learning or Dueling Deep Q-network) might be applied to operate in several scales. To conclude, computational creativity is presented as a research area within the context of evolutionary algorithms, which, in turn, need to consider hierarchical and deep relationships among the components involved in the algorithms. Hence, the application of Deep-Bio approaches could help to improve the current state of the art in this area.

This whole area is rife with opportunities for scientific advance. Related to Eph-C problems, the development of adaptive algorithms in massive complex domains will allow to define a new kind of algorithms where problems as heterogeneity, or the unpredictable nature of the resources will be included in the future design of the algorithms taking awareness about these features. On the other hand, and related to Deep-Bio approaches, it will allow the design of new hierarchical and deep bioinspired algorithms, taking advantages on the different strengths of classical methods, for example allowing to move from small-medium large pools of individuals (usually from several hundreds to few thousands) to large, or very large, pools (hundred of thousands or millions). Although the paper has shown several specific applications domains, as an example of current developments in these fields, the authors are aware that other potential domains as energy forecasting and optimization, bioinformatics, audio and video analysis, biometrics, to mention just few, could take advantage on the future design of such algorithms.

Acknowledgements

This work has been supported by Spanish Ministry of Economy and Competitiveness (MINECO) projects: EphemeCH (TIN2014-56494-C4-{1...4}-P), and DeepBio (TIN2017-85727-C4-{1...4}-P), both under the European Regional Development Fund FEDER – Check: `http://ephemech.wordpress.com` and `http://deepbio.wordpress.com`.

References

References

[1] C. Cotta, A. J. Fernández-Leiva, F. Fernández de Vega, F. Chávez, J. J. Merelo, P. A. Castillo, G. Bello, D. Camacho, Ephemeral computing and bioinspired optimization - challenges and opportunities, in: 7th International Joint Conference on Evolutionary Computation Theory and Applications, SCITEPRESS, Lisboa, Portugal, 2015, pp. 319–324.

[2] C. Cotta, A. J. Fernández-Leiva, F. Fernández de Vega, F. Chávez, J. J. Merelo, P. A. Castillo, D. Camacho, M. D. R.-Moreno, Application areas of ephemeral computing: A survey, Trans. Computational Collective Intelligence 24 (2016) 153–167.

[3] J. A. McCann, From IoT to ephemeral computing: Understanding cyber-physical interactions, in: International Conference on Future Networks and Distributed Systems, ACM, 2017, p. 2.

[4] F. Orellana, M. Niinimaki, GridFactory: Distributed computing on ephemeral resources, in: F. Xhafa, L. Barolli, J. Kolodziej, S. U. Khan (Eds.), 2011 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, IEEE Press, 2011, pp. 25–31.

[5] F. J. Alexander, A. Hoisie, A. Szalay, Big data, Computing in Science Engineering 13 (6) (2011) 10–13.

[6] S. Lohr, The age of big data, New York Times 11 February (2012), [Online; accessed 5-September-2014].

[7] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–
–444.

[8] S. Mostaghim, Parallel multi-objective optimization using self-organized heterogeneous resources, in: F. Fernández de Vega, E. Cantú-Paz (Eds.), Parallel and Distributed Computational Intelligence, Vol. 269 of Studies in Computational Intelligence, Springer, 2010, pp. 165–179.

[9] D. Theodoropoulos, G. Chrysos, I. Koidis, G. Charitopoulos, E. Pissadakis, A. Varikos, D. N. Pnevmatikatos, G. Smaragdos, C. Strydis, N. Zervos, mCluster: A software framework for portable device-based volunteer computing, in: IEEE/ACM 16th International Symposium on Cluster, Cloud and Grid Computing, IEEE Computer Society, 2016, pp. 336–341.

[10] J. J. Merelo, P. A. Castillo, P. García-Sánchez, P. de las Cuevas, N. Rico, M. G. Valdez, Performance for the masses: Experiments with a web based architecture to harness volunteer resources for low cost distributed evolutionary computation, in: T. Friedrich, F. Neumann, A. M. Sutton (Eds.), Genetic and Evolutionary Computation Conference, ACM, 2016, pp. 837–844.

[11] M. N. Huhns, M. P. Singh, Service-oriented computing: Key concepts and principles, IEEE Internet computing 9 (1) (2005) 75–81.

[12] K. Lyytinen, Y. Yoo, Ubiquitous computing, Communications of the ACM 45 (12) (2002) 63–96.

[13] B. Wang, J. Bodily, S. K. S. Gupta, Supporting persistent social groups in ubiquitous computing environments using context-aware ephemeral group service, in: Second IEEE International Conference on Pervasive Computing and Communications (PerCom 2004), IEEE Computer Society, 2004, pp. 287–296.

[14] L. F. Sarmenta, S. Hirano, Bayanihan: Building and studying web-based volunteer computing systems using java, Future Generation Computer Systems 15 (5) (1999) 675–686.

[15] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. F. Knight Jr, R. Nagpal, E. Rauch, G. J. Sussman, R. Weiss, Amorphous computing, Communications of the ACM 43 (5) (2000) 74–82.

[16] S. Kamil, J. Shalf, L. Oliker, D. Skinner, Understanding ultra-scale application communication requirements, in: Workload Characterization Symposium, 2005. IEEE International, IEEE, 2005, pp. 178–187.

[17] J. J. Merelo, P. de Las Cuevas, P. García-Sánchez, M. García-Valdez, The human in the loop: volunteer-based metacomputers as a socio-technical system, in: Artificial Life Conference 2016, Complex Adaptive Systems, The MIT Press, Cambridge MA, 2016, p. 648.

[18] Y. Pan, J. White, Y. Sun, J. Gray, Gray computing: A framework for computing with background javascript tasks, IEEE Transactions on Software Engineering (2017) . doi:10.1109/TSE.2017.2772812.

[19] A. Vespignani, Predicting the behavior of techno-social systems, Science 325 (5939) (2009) 425–428.

[20] C. Langton, Computation at the edge of chaos: Phase transitions and emergent computation, Physica D: Nonlinear Phenomena 42 (1-3) (1990) 12–37.

[21] J. J. Merelo, P. de las Cuevas, P. García-Sánchez, M. G. Valdez, A performance assessment of evolutionary algorithms in volunteer computing environments: The importance of entropy, in: G. Squillero, K. Sim (Eds.), Applications of Evolutionary Computation, Vol. 10199 of Lecture Notes in Computer Science, 2017, pp. 806–821.

[22] M. G. Valdez, J. C. Romero, A. Mancilla, J. J. M. Guervós, Exploiting the social graph: Increasing engagement in a collaborative interactive evolution application, in: 2017 IEEE Congress on Evolutionary Computation, CEC 2017, IEEE, 2017, pp. 749–756.

[23] A. Lastovetsky, Heterogeneous parallel computing: from clusters of workstations to hierarchical hybrid platforms, Supercomputing Frontiers and Innovations 1 (3) (2014) 70–87.

[24] D. P. Anderson, K. Reed, Celebrating diversity in volunteer computing, in: 2nd Hawaii International Conference on System Sciences, HICSS '09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 1–8.

[25] M. Beltrán, A. Guzmán, How to balance the load on heterogeneous clusters, International Journal of High Performance Computing Applications 23 (2009) 99–118.

[26] H. Renard, Y. Robert, F. Vivien, Data redistribution algorithms for heterogeneous processor rings, International Journal of High Performance Computing Applications 20 (2006) 31–43.

[27] R. Nogueras, C. Cotta, Analyzing self-⋆ island-based memetic algorithms in heterogeneous unstable environments, The International Journal of High Performance Computing Applications (2016) . doi:10.1177/1094342016678665.

[28] O. Babaoglu, et al. (Eds.), Self-star properties in complex information systems, Vol. 3460 of Lecture Notes in Computer Science, Springer, Berlin Heidelberg, 2005.

[29] A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer, M. Q. Dang, K. Pentikousis, Energy-efficient cloud computing, The computer journal 53 (7) (2010) 1045–1051.

[30] J. L. Jiménez Laredo, P. A. Castillo, A. M. Mora, J. J. Merelo, Resilience to churn of a peer-to-peer evolutionary algorithm, International Journal of High Performance Systems Architecture 1 (4) (2008) 260–268.

[31] D. Lombraña González, F. Fernández de Vega, H. Casanova, Characterizing fault tolerance in genetic programming, Future Generation Computer Systems 26 (6) (2010) 847 – 856.

[32] J. I. Hidalgo, J. Lanchares, F. Fernández de Vega, D. Lombraña, Is the island model fault tolerant?, in: 9th Annual Conference Companion on Genetic and Evolutionary Computation, GECCO '07, ACM, New York, NY, USA, 2007, pp. 2737–2744.

[33] N. Melab, S. Cahon, E. Talbi, Grid computing for parallel bioinspired algorithms, Journal of Parallel and Distributed Computing 66 (8) (2006) 1052–1061.

[34] R. Nogueras, C. Cotta, Studying fault-tolerance in island-based evolutionary and multimemetic algorithms, Journal of Grid Computing 13 (3) (2015) 351—374.

[35] R. Nogueras, C. Cotta, Self-healing strategies for memetic algorithms in unstable and ephemeral computational environments, Natural Computing 16 (2) (2017) 189–200.

[36] E. Alba, J. M. Troya, Analyzing synchronous and asynchronous parallel distributed genetic algorithms, Future Generation Computer Systems 17 (4) (2001) 451–465.

[37] G. P. Wagner, L. Altenberg, Perspective: complex adaptations and the evolution of evolvability, Evolution 50 (3) (1996) 967–976.

[38] M. Sipper, W. Fu, K. Ahuja, J. H. Moore, Investigating the parameter space of evolutionary algorithms, BioData Mining 11 (1) (2018) 2.

[39] F. Neri, C. Cotta, P. Moscato (Eds.), Handbook of memetic algorithms, Vol. 379 of Studies in Computational Intelligence, Springer, Berlin Heidelberg, 2012.

[40] P. Moscato, On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms., Tech. Rep. 826, Technical Report Caltech Concurrent Computation Program, California Institute of Technology, Pasadena, California, USA (1989).

[41] C. Cotta, A. J. Fernández Leiva, J. E. Gallardo, Memetic algorithms and complete techniques, in: Neri et al. [39], pp. 193–204.

[42] J. E. Gallardo, C. Cotta, A GRASP-based memetic algorithm with path relinking for the far from most string problem, Engineering Applications of Artificial Intelligence 41 (2015) 183–194.

[43] Y. Ong, M. Lim, X. Chen, Memetic computation –past, present and future, IEEE Computational Intelligence Magazine 5 (2) (2010) 24–31.

[44] P. Moscato, Memetic algorithms: A short introduction, in: D. Corne, M. Dorigo, F. Glover (Eds.), New Ideas in Optimization, McGraw-Hill, Maidenhead, UK, 1999, pp. 219–234.

[45] N. Krasnogor, J. Smith, Emergence of profitable search strategies based on a simple inheritance mechanism, in: L. Spector, et al. (Eds.), Genetic and Evolutionary Computation Conference 2001, Morgan Kaufmann, San Francisco CA, 2001, pp. 432–439.

[46] K. Chakhlevitch, P. I. Cowling, Hyperheuristics: recent developments, in: Cotta et al. [49], pp. 3–29.

[47] S. Talukdar, S. Murthy, R. Akkiraju, Asynchronous teams, in: F. Glover, G. Kochenberger (Eds.), Handbook of Metaheuristics, Springer, Boston MA, 2003, pp. 537–556.

[48] X. Chen, Y. S. Ong, A conceptual modeling of meme complexes in stochastic search, IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews 42 (5) (2012) 612–625.

[49] C. Cotta, M. Sevaux, K. Sörensen (Eds.), Adaptive and multilevel metaheuristics, Vol. 136 of Studies in Computational Intelligence, Springer, 2008.

[50] R. Frei, R. McWilliam, B. Derrick, A. Purvis, A. Tiwari, G. Di Marzo Serugendo, Self-healing and self-repairing technologies, International Journal of Advanced Manufacturing Technology 69 (5-8) (2013) 1033–1061.

[51] R. Nogueras, C. Cotta, Studying self-balancing strategies in island-based multimemetic algorithms, Journal of Computational and Applied Mathematics 293 (2016) 180–191.

[52] N. Krasnogor, S. Gustafson, A study on the use of "self-generation" in memetic algorithms, Natural Computing 3 (1) (2004) 53–76.

[53] J. Smith, Co-evolving memetic algorithms: A review and progress report, IEEE Transactions on Systems Man and Cybernetics Part B (Cybernetics) 37 (1) (2007) 6–17.

[54] A. Eiben, Evolutionary computing and autonomic computing: Shared problems, shared solutions?, in: Babaoglu et al. [28], pp. 36–48.

[55] J. E. Smith, Self-adaptation in evolutionary algorithms for combinatorial optimisation, in: Cotta et al. [49], pp. 31–57.

[56] R. Nogueras, C. Cotta, Towards resilient multimemetic systems on unstable networks with complex topology, in: G. Papa (Ed.), Advances in Evolutionary Algorithm Research, Nova Science Pub., 2015, pp. 17–23.

[57] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, S. U. Khan, The rise of "big data" on cloud computing: Review and open research issues, Information Systems 47 (2015) 98 – 115.

[58] V. Jirkovský, M. Obitko, V. Mařík, Understanding data heterogeneity in the context of cyber-physical systems integration, IEEE Transactions on Industrial Informatics 13 (2) (2017) 660–667.

[59] J. Michel, C. Julien, J. Payton, G. C. Roman, A spatiotemporal model for ephemeral data in pervasive computing networks, in: 2012 IEEE International Conference on Pervasive Computing and Communications Workshops, IEEE, 2012, pp. 179–184.

[60] M. Höger, O. Kao, P. Richter, D. Warneke, Ephemeral materialization points in stratosphere data management on the cloud, Adv. Parallel Comput 23 (2013) 163–181.

[61] A. Martín, R. Lara-Cabrera, F. Fuentes-Hurtado, V. Naranjo, D. Camacho, EvoDeep: a new evolutionary approach for automatic deep neural networks parametrisation, Journal of Parallel and Distributed Computing (2017) . doi:10.1016/j.jpdc.2017.09.006.

[62] J. Cao, H. Cui, H. Shi, L. Jiao, Big data: a parallel particle swarm optimization-back-propagation neural network algorithm based on mapreduce, PloS one 11 (6) (2016) e0157551.

[63] C. P. Chen, C.-Y. Zhang, Data-intensive applications, challenges, techniques and technologies: A survey on big data, Information Sciences 275 (2014) 314–347.

[64] M. Bhattacharya, R. Islam, J. Abawajy, Evolutionary optimization: a big data perspective, Journal of network and computer applications 59 (2016) 416–426.

[65] S. Kaisler, F. Armour, J. A. Espinosa, W. Money, Big data: Issues and challenges moving forward, in: 2013 46th Hawaii International Conference on System Sciences (HICSS), IEEE, 2013, pp. 995–1004.

[66] Z.-H. Zhou, N. V. Chawla, Y. Jin, G. J. Williams, Big data opportunities and challenges: Discussions from data analytics perspectives [discussion forum], Comp. Intell. Mag. 9 (4) (2014) 62–74.

[67] H. D. Menéndez, F. E. Otero, D. Camacho, Extending the SACOC algorithm through the Nyström method for dense manifold data analysis, International Journal of Bio-Inspired Computation 10 (2) (2017) 127–135.

[68] H. D. Menéndez, F. E. B. Otero, D. Camacho, MACOC: A medoid-based ACO clustering algorithm, in: M. Dorigo, M. Birattari, S. Garnier, H. Hamann, M. A. M. de Oca, C. Solnon, T. Stützle (Eds.), Swarm Intelligence - 9th International Conference, ANTS 2014, Vol. 8667 of Lecture Notes in Computer Science, Springer, 2014, pp. 122–133.

[69] Y. Kao, K. Cheng, An ACO-based clustering algorithm, in: M. Dorigo, L. M. Gambardella, M. Birattari, A. Martinoli, R. Poli, T. Stützle (Eds.), Ant Colony Optimization and Swarm Intelligence, 5th International Workshop, ANTS 2006, Vol. 4150 of Lecture Notes in Computer Science, Springer, 2006, pp. 340–347.

[70] W. Tan, M. B. Blake, I. Saleh, S. Dustdar, Social-network-sourced big data analytics, IEEE Internet Computing 17 (5) (2013) 62–69.

[71] P. Braun, A. Cuzzocrea, C. K. Leung, A. G. M. Pazdor, K. Tran, Knowledge discovery from social graph data, in: R. J. Howlett, L. C. Jain, B. Gabrys, C. Toro, C. P. Lim (Eds.), Knowledge-Based and Intelligent Information & Engineering Systems, Vol. 96 of Procedia Computer Science, Elsevier, 2016, pp. 682–691.

[72] K. S. Rani, Tree representation: Knowledge discovery from uncertain data, Procedia Computer Science 78 (2016) 683 – 690.

[73] G. Manco, P. Rullo, L. Gallucci, M. Paturzo, Rialto: A knowledge discovery suite for data analysis, Expert Systems with Applications 59 (2016) 145 – 164.

[74] G. Bello-Orgaz, J. J. Jung, D. Camacho, Social big data: Recent achievements and new challenges, Information Fusion 28 (2016) 45–59.

[75] W. He, S. Zha, L. Li, Social media competitive analysis and text mining: A case study in the pizza industry, International Journal of Information Management 33 (3) (2013) 464 – 472.

[76] J. Song, T. M. Song, J. R. Lee, Stay alert: Forecasting the risks of sexting in Korea using social big data, Computers in Human Behavior 81 (2018) 294 – 302.

[77] P. D. Vecchio, G. Mele, V. Ndou, G. Secundo, Creating value from social big data: Implications for smart tourism destinations, Information Processing and Management (2017) . doi:10.1016/j.ipm.2017.10.006.

[78] A. Hussain, E. Cambria, Semi-supervised learning for big social data analysis, Neurocomputing 275 (2018) 1662 – 1673.

[79] T. Ruiz, L. Mars, R. Arroyo, A. Serna, Social networks, big data and transport planning, Transportation Research Procedia 18 (2016) 446 – 452.

[80] G. Bello-Orgaz, J. Hernandez-Castro, D. Camacho, Detecting discussion communities on vaccination in Twitter, Future Generation Computer Systems 66 (2017) 125–136.

[81] R. Lara-Cabrera, A. Gonzalez-Pardo, D. Camacho, Statistical analysis of risk assessment factors and metrics to evaluate radicalisation in Twitter, Future Generation Computer Systems (2017) . doi:10.1016/j.future.2017.10.046.

[82] R. Lara-Cabrera, C. Cotta, A. Fernández-Leiva, An analysis of the structure and evolution of the scientific collaboration network of computer intelligence in games, Physica A: Statistical Mechanics and its Applications 395 (2014) 523–536.

[83] A. Gonzalez-Pardo, J. J. Jung, D. Camacho, ACO-based clustering for ego network analysis, Future Generation Computer Systems 66 (2017) 160 – 170.

[84] A. Said, R. A. Abbasi, O. Maqbool, A. Daud, N. R. Aljohani, CC-GA: a clustering coefficient based genetic algorithm for detecting communities in social networks, Applied Soft Computing 63 (2018) 59 – 70.

[85] M. Guerrero, F. G. Montoya, R. Baños, A. Alcayde, C. Gil, Adaptive community detection in complex networks using genetic algorithms, Neurocomputing 266 (2017) 101 – 113.

[86] A. C. Gabardo, R. Berretta, N. J. de Vries, P. Moscato, Where does my brand end? An overlapping community approach, in: G. Leu, H. K. Singh, S. Elsayed (Eds.), Intelligent and Evolutionary Systems, Springer International Publishing, Cham, 2017, pp. 133–148.

[87] H. D. Menéndez, D. F. Barrero, D. Camacho, A genetic graph-based approach for partitional clustering, International Journal of Neural Systems 24 (03) (2014) 1430008.

[88] A. Abraham, S. Das, S. Roy, Swarm intelligence algorithms for data clustering, in: O. Maimon, L. Rokach (Eds.), Soft Computing for Knowledge Discovery and Data Mining, Springer, 2008, pp. 279–313.

[89] J. Handl, B. Meyer, Ant-based and swarm-based clustering, Swarm Intelligence 1 (2) (2007) 95–113.

[90] E. Hruschka, R. Campello, A. Freitas, A. de Carvalho, A survey of evolutionary algorithms for clustering, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 39 (2) (2009) 133–155.

[91] G. N. Yannakakis, J. Togelius, A panorama of artificial and computational intelligence in games, IEEE Transactions on Computational Intelligence and AI in Games 7 (4) (2015) 317–335.

[92] N. Shaker, J. Togelius, M. J. Nelson, Procedural content generation in games, Computational Synthesis and Creative Systems, Springer, 2016.

[93] M. Nogueira Collazo, C. Cotta, A. J. Fernández Leiva, Competitive algorithms for coevolving both game content and AI. A case study: Planet wars, IEEE Transactions on Computational Intelligence and AI in Games 8 (4) (2016) 325–337.

[94] S. Risi, J. Togelius, Neuroevolution in games: State of the art and open challenges, IEEE Trans. Comput. Intellig. and AI in Games 9 (1) (2017) 25–41.

[95] A. P. Poulsen, M. Thorhauge, M. H. Funch, S. Risi, DLNE: A hybridization of deep learning and neuroevolution for visual control, in: IEEE Conference on Computational Intelligence and Games, CIG 2017, New York, NY, USA, August 22-25, 2017, IEEE, 2017, pp. 256–263.

[96] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmgård, A. K. Hoover, A. Isaksen, A. Nealen, J. Togelius, Procedural content generation via machine learning (PCGML), CoRR abs/1702.00539. arXiv:1702.00539.

[97] N. Justesen, P. Bontrager, J. Togelius, S. Risi, Deep learning for video game playing, CoRR abs/1708.07902. arXiv:1708.07902.

[98] T. Salimans, J. Ho, X. Chen, I. Sutskever, Evolution strategies as a scalable alternative to reinforcement learning, CoRR abs/1703.03864. arXiv:1703.03864.

[99] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, R. Hadsell, Progressive neural networks, CoRR abs/1606.04671. arXiv:1606.04671.

[100] J. Gideon, S. Khorram, Z. Aldeneh, D. Dimitriadis, E. M. Provost, Progressive neural networks for transfer learning in emotion recognition, CoRR abs/1706.03256. arXiv:1706.03256.

[101] P. D. Sørensen, J. M. Olsen, S. Risi, Interactive super mario bros evolution, in: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, GECCO '16 Companion, ACM, New York, NY, USA, 2016, pp. 41–42.

[102] P. Bontrager, W. Lin, J. Togelius, S. Risi, Deep interactive evolution, CoRR abs/1801.08230. arXiv:1801.08230.

[103] M. Cook, M. Eladhari, A. Nealen, M. Treanor, E. Boxerman, A. Jaffe, P. Sottosanti, S. Swink, Pcg-based game design patterns, CoRR abs/1610.03138. arXiv:1610.03138.

[104] D. P. Liebana, S. Samothrakis, J. Togelius, T. Schaul, S. M. Lucas, A. Couëtoux, J. Lee, C. Lim, T. Thompson, The 2014 general video game playing competition, IEEE Trans. Comput. Intellig. and AI in Games 8 (3) (2016) 229–243.

[105] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.

[106] P. Galanter, What is generative art? complexity theory as a context for art theory, in: In GA2003 – 6th Generative Art Conference, 2003.

[107] P. Bentley, Evolutionary design by computers, Morgan Kaufmann, 1999.

[108] P. J. Bentley, D. W. Corne, Creative evolutionary systems, Morgan Kaufmann, 2002.

[109] M. Frade, F. Fernández de Vega, C. Cotta, Automatic evolution of programs for procedural generation of terrains for video games, Soft Computing 16 (11) (2012) 1893–1914.

[110] H. Takagi, Interactive evolutionary computation: Fusion of the capabilities of ec optimization and human evaluation, IEEE 89 (9) (2001) 1275–1296.

[111] G. Diaz-Jerez, Composing with melomics: Delving into the computational world for musical inspiration, Leonardo Music Journal 21 (2011) 13–14.

[112] A. Pease, S. Colton, On impact and evaluation in computational creativity: A discussion of the turing test and an alternative proposal, in: D. Kazakov, G. Tsoulas (Eds.), AISB symposium on AI and Philosophy, 2011, pp. 15–22.

[113] F. Fernández de Vega, C. Cruz, L. Navarro, P. Hernández, T. Gallego, L. Espada, Unplugging evolutionary algorithms: an experiment on human-algorithmic creativity, Genetic Programming and Evolvable Machines 15 (4) (2014) 379–402.

[114] G. Reis, F. Fernández de Vega, A. Ferreira, Automatic transcription of polyphonic piano music using genetic algorithms, adaptive spectral envelope modeling, and dynamic noise level estimation, IEEE Transactions on Audio, Speech, and Language Processing 20 (8) (2012) 2313–2328.

[115] N. McIntyre, M. Lapata, Plot induction and evolutionary search for story generation, in: 48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2010, pp. 1562–1572.

[116] H. Manurung, An evolutionary algorithm approach to poetry generation, Ph.D. thesis, University of Edinburgh. College of Science and Engineering. School of Informatics. (2004).

[117] L. R. Varshney, F. Pinel, K. R. Varshney, D. Bhattacharjya, A. Schörgendorfer, Y. Chee, A big data approach to computational creativity, CoRR abs/1311.1213. `arXiv:1311.1213`.

[118] J. McCormack, M. D'Iverno, Computers and creativity, Springer, 2012.

[119] J. Togelius, G. N. Yannakakis, K. O. Stanley, C. Browne, Search-based procedural content generation: A taxonomy and survey, IEEE Transactions on Computational Intelligence and AI in Games 3 (3) (2011) 172–186.

[120] H. Takagi, Humanized computational intelligence with interactive evolutionary computation, in: D. Fogel, C. Robinson (Eds.), Computational Intelligence: The Experts Speak, Wiley, 2003, Ch. 15, pp. 207–218.