Application Areas of Ephemeral Computing: A Survey

Carlos Cotta[1], Antonio J. Fernández-Leiva[1], Francisco Fernández de Vega[2], Francisco Chávez[3(✉)], Juan J. Merelo[4], Pedro A. Castillo[4], David Camacho[5], and María D. R-Moreno[5]

[1] Dept. Lenguajes Y Ciencias de la Computación, Universidad de Málaga, Malaga, Spain
{ccottap,afdez}@lcc.uma.es
[2] Dept. Tecnología de los Computadores y de las Comunicaciones, Universidad de Extremadura, Merida, Spain
fcofdez@unex.es
[3] Dept. Ingeniería En Sistemas Informáticos Y Telemáticos, Universidad de Extremadura, Merida, Spain
fchavez@unex.es
[4] Dept. Arquitectura Y Tecnología de los Computadores, Universidad de Granada, Granada, Spain
{jmerelo,pacv}@ugr.es
[5] Dept. Ingeniería Informática, Universidad Autónoma de Madrid, Madrid, Spain
{david.camacho,gema.bello}@uam.es

Abstract. It is increasingly common that computational devices with significant computing power are underexploited. Some of the reasons for that are due to frequent idle-time or to the low computational demand of the tasks they perform, either sporadically or in their regular duty. The exploitation of this (otherwise-wasted) computational power is a cost-effective solution for solving complex computational tasks. Individually (device-wise), this computational power can sometimes comprise a stable, long-lasting availability window but it will more frequently take the form of brief, ephemeral bursts. Then, in this context a highly dynamic and volatile computational landscape emerges from the collective contribution of such numerous devices. Algorithms consciously running on this kind of environment require specific properties in terms of flexibility, plasticity and robustness. 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. Deploying bioinspired techniques on this scenario, and conducting analysis and modelling of the underlying Ephemeral Computing environment will also pave the way for the application of other non-bioinspired techniques on this computational domain. Computational creativity and content generation in video games are applications areas of the foremost economical interest and are well suited to Ephemeral Computing due to their intrinsic ephemeral nature and the widespread abundance of gaming applications in all kinds of devices. In this paper, we will explain why and how they can be adapted to this new environment.

N.T. Nguyen et al. (Eds.): TCCI XXIV, LNCS 9770, pp. 153–167, 2016.
DOI: 10.1007/978-3-662-53525-7_9

Keywords: Ephemeral computing · Bioinspired optimization · Evolutionary computation · Complex systems · Autonomic computing · Distributed computing

1 Ephemeral Computing: What and Why

This paper revolves around the notion of *Ephemeral Computing* (Eph-C) which can be 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 computational tasks"*. The main goal in Eph-C is thus making an effective use of highly-volatile resources whose computational power (which can be collectively enormous) would be otherwise wasted or under-exploited. Think for example of the pervasive abundance of networked handheld devices, tablets and, lately, wearables –not to mention more classical devices such as desktop computers– whose computational capabilities are often not fully exploited. Hence, the concept of Eph-C partially overlaps with ubiquitous computing, volunteer computing and distributed computing. Due to these research fields deal with the concept of ephemerality are explained in next section, but exhibits its own distinctive features, mainly in terms of the extreme dynamism of the underlying resources, and the ephemerality-aware nature of the computation which autonomously adapt to the ever-changing computational landscape. These concepts not trying to fit to the inherent volatility of the latter but even trying to use it for its own advantage.

In light of the computational context described above, it is clear that the algorithmic processes deployed onto it should be flexible (to work on a variety of computational scenarios), resilient (to cope with sudden failures and with the phenomenon of churn [55]), (self-)adaptive (to react autonomously to changes in the environment and optimize its own performance in a smart way), and intrinsically decentralized (since centralized control strategies cannot consistently comprehend the state of the computational landscape and decisions emerging from them would lag behind the changing conditions of the latter). Some bioinspired algorithms, like the Evolutionary algorithms (EAs) fit nicely into this scenario. However, few works have previously considered the interest of endowing evolutionary algorithms with the capability for coping with transient behaviors in underlaying computer systems. Moreover, in the age when the term Big Data [35] is present in many initiatives requiring large amount of computational resources for storing, processing, and learning from huge amount of data, new methods and algorithms for properly managing heterogeneous computing resources widely distributed along the world are required. Energy consumption must also be considered from the point of view of both algorithms and hardware resources, given the large differences among large computing infrastructures typically devoted to running massively parallel algorithms when compared to smart devices optimized for reducing battery consumption. It is of the foremost interest to research on the basic features allowing to provide efficient and reliable ephemeral evolutionary services.

The paper is structured as follows. Section 2 gives an onverview of Eph-C. Then, we analyze an important parameter in Eph-C: the energy consumption.

Next, the optimization criterion is presented by means of bioinspired algorithms. Section 5 presents a short revision on bioinspired methods and applications that could be affected by Eph-C characteristics. Finally, the conclusions are outlined.

2 Ephemeral Computing in Perspective

According to the Oxford Dictionary, the term *ephemeral* means "lasting for a very short time". It thus encompasses things or events with a transitory nature, with a brief existence. A number of phenomena and resources in computer science are endowed with that feature (e.g. in computing networking, an ephemeral port is a TCP port, for instance, dynamically assigned to a client application for a brief period of time, in contrast with well known ports) [10]. Ephemeral behaviors can be also observed in the way users collaborate in volunteer networks of computers.

Although ephemeral phenomena naturally arise in several areas such as ubiquitous computing, volunteer computing or traditional research areas like distributed computing, some issues arise when dealing with ephemeral behavior. In cloud computing [3], for instance, the opposite is usually looked for: *persistence*. Although services are commonly associated with computations among autonomous heterogeneous parties in dynamic environments, exceptions must be handled to take corrective actions. Ephemeral services are thus commonly seen more as a problem than a solution [28].

On the other hand, in ubiquitous computing the main goal is to leverage computation everywhere and anywhere, so that computation can occur using any kind of device, in any location, starting and ending at any time and using any format and during any amount of time. The main efforts in this area have been oriented to design and develop the underlying technologies needed to support ubiquitous computing [37] (like advanced middleware, operating systems, mobile code, sensors, microprocessors, new I/O and user interfaces, networks or mobile protocols). However, and in the same way it happens with cloud computing, the main target in ubiquitous computing is to allow stable and persistent computation processes perform a complete execution of the programs. When this area handles the concept of ephemeral devices, services or computation, the main solution is to stop the process, or processes, and resume once new devices are ready [59]. Previous hypothesis and assumptions can be extrapolated to distributed computing, where the concept of ephemeral services can be a problem that could eventually generate a failure in the execution of the process [54].

As stated before, the main focus of Eph-C is different from the above approaches: rather than trying to build layers onto the network of ephemeral resources in order to "hide" their transient nature and provide the illusion of a virtual stable environment, Eph-C applications are fully aware of the nature of the computational landscape and are specifically built to live (and optimize their performance) in this realm. Note that this does not imply the latter have a lower-level vision of the underlying computational substrate, or at least not markedly so. In fact, most low-level features can be abstracted without precluding attaining a more accurate vision of this fluctuating substrate.

To some extent, some of these ephemerality issues are also present in areas such as volunteer computing (VC) [53], whereby a dynamic collection of computing devices collaborate in solving a massive computational task, decomposing it into small processing chunks. Most VC approaches follow a centralized master/slave scheme though, and typically deal with resource volatility via redundant computation. A much more decentralized, emergent approach can be found in amorphous computing [1], but that paradigm is more geared towards programmable materials and their use to attack massive simulation problems. Massive problems are also the theme in ultrascale computing, where issues such as scalability, resilience to failures, energy management, and handling of large volume of data are of paramount importance [30,46]. Note however that Eph-C is not necessarily exascale nor it is oriented towards supercomputing.

3 Energy Consumption of Algorithms

When dealing with Eph-C, a relevant parameter to be taken into account is energy consumption, given that mobile devices frequently provide hardware resources to run programs, and battery consumption is always a concern for this kind of devices. During the last decades, researchers have focused on energy consumption for some kind of algorithms; for instance, encryption algorithms, that are frequently required in wireless communication, consume significant amount of computing resources such as CPU time, memory, and battery power, which affects every mobile device with wireless connection [51]. Sorting algorithms have been also analysed from this point of view [11].

Researchers have been interested in finding a proper way to map energy consumption to program structure [18]. But not always this mapping can be easily found, particularly when dealing with stochastic algorithms. The focus has been typically placed on the infrastructures behind the algorithms and the way they are exploited and offered to companies, which is the case for cloud models [8]; and not so frequently on the way algorithms can be optimized to better exploit those infrastructures. Thus, the term *Green Computing* has emerged when refering to the practice of using computing resources more efficiently while maintaining or increasing overall performance [26].

But we are more interested in the relationship between algorithms and the time and energy required to solve a problem. Although initially we could find a direct relationship between CPU cycles and energy required to run an algorithm, when optimization problems are faced by means stochastic algorithms affected by a series of parameters, such a relationship is not so straightforward, especially when different hardware architectures and operating systems can be chosen to run the algorithms: x86 family based computers running Linux or Windows, ARM based mobile devices with Android, etc. Moreover, even in a single platform and a given algorithm, different data structures that could be employed within the algorithm also affect energy consumed by some physical components of the computer, such as cache memories, and therefore can also be optimized to save energy, even when the time required to run an algorithm may not change [2].

If we specifically deal with ubiquitous computing, and given that the computation can occur using any kind of device, in any location, energy consumption should be considered when deciding which of the available devices will be employed. But also in a more standard setting, total energy consumed on a given hardware platform could in the future decide which is the preferred one, regardless of the time required to run the algorithm: sometimes the investment applied when finding a solution could be more importantly considered than time to solution.

4 Bioinspired Algorithms and Ephemerality

The term *bioinspired algorithms* usually refers to methods that draw some inspiration from Nature to solve search, optimization or pattern recognition problems. If we focus on optimization problems, the most prominent bioinspired paradigms are evolutionary computation and swarm intelligence. We are particularly interested in this kind of population-based search and optimization algorithms, which have a natural path to distributed computing by simply distributing the population among the different computing nodes, the issue being how to do it in an algorithmically efficient and scalable way. Eph-C, besides the obvious fact that the contribution of a node might come and go at any time, adds several other dimensions to the design of algorithms:

- *Inclusion*: all nodes should have a meaningful contribution to the final result, and they should be incorporated to the distributed system in such a way that they do.
- *Energy Consumption*: Different algorithm parameters influences time to result which in combination with specific hardware architectures behind each node implies a given energy consumption.
- *Asynchrony*: nodes communicate with the others without a fixed schedule due to their different performance.
- *Resilience*: the sudden disappearance of computing nodes must not destabilize the functioning of the algorithm.
- *Emergence*: the nature of the computational environment does not allow a centralized control and requires decentralized, emergent behavior.
- *Self-adaptation*: the algorithm should adapt itself to the changing computational landscape.

This latter issue is particularly important, and encompasses a number of self-$\star$ properties [4] the system must exhibit in order to exert advanced control on its own functioning and/or structure, e.g., self-maintaining in proper state, self-healing externally infringed damage [20], self-adapting to different environmental conditions [48], and even self-generating new functionalities just to cite a few examples, see also [14,16]. Quite interestingly, these properties are frequently intrinsic features of the system, that is, emergent properties of its complex structure, rather than the result of endowing it with a central command. This also

implies there is no need for a central control in the system; every node schedules itself. This decentralization implies a certain fault-tolerance due to the lack of a single point of failure, but it also means resilience must be built into the algorithms present in each node so that their sensitivity to changes in the rest of the system is minimal. This will include measures such as population sizing and the conservation of diversity in each node, as indicated by Cantú-Paz in [13] but taken to new meanings in this context. Indeed, models and algorithms have to be designed to be fault-tolerant [47] so that inclusion of new nodes will be done in a self-adaptive way, but also in such a way that its disappearance from the network will not have a big impact on performance. In fact, VC systems, which are an early example of Eph-C, have been proved to be fault tolerant to a certain point [22], but this fault tolerance will have to be taken into account not just at the implementation level (backing up solutions, for instance) but also at the model and algorithm level, measuring the impact of different churn models [33,49].

Regarding Energy Consumption and Bioinspired Algorithms, we know that they have already been applied to optimize problems related to energy management and consumption, such as HAVC (heating, ventilating and air-conditioning) systems [19]. Yet, to the best of our knowledge, an analysis between the different flavors of available EAs, the parameters affecting them, and the relationship with different available configurations, time to solution, and energy consumed to reach the solution when different hardware architectures are employed have not been addressed. We think that this issue provides a new perspective to apply a multiobjective analysis of the algorithms considering time to solution, energy required, hardware architectures available in relatinship with algorithms configurations and main parameters. And this issue is particularly important when Eph-C is available.

5 Ephemeral Computing-Based Applications

From a practical perspective, the application, development and even deployment of any application that should be executed in an ephemeral environment will need to take into account those features described in Sect. 4. Therefore, the application to ephemeral environments of traditional techniques and methods, as bio-inspired computation, will generate some interesting challenges and opportunities that can be analyzed.

This section provides a short revision on some of those bioinspired methods and applications that could be affected by Eph-C characteristics.

5.1 Big Data and Bio-Inspired Clustering

The data volume and the multitude of sources have experienced an exponential growing with new technological and application challenges. The data generation has been estimated as 2.5 quintillion bytes of data per day[1]. This data comes

[1] http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html.

from everywhere: sensors used to gather climate, traffic, air flight information, posts to social media sites (i.e. Twitter or Facebook as popular examples), digital pictures and videos (YouTube users upload 72 h of new video content per minute[2]), purchase transaction records, or cell phone GPS signals to name a few. The classic methods, algorithms, frameworks or tools for data management have become both inadequate for processing these amount of data and unable to offer effective solutions to deal with the data growing. The management, handling and extraction of useful knowledge from these data sources is currently one of the most popular and hot topics in computing research.

In this context, Big Data is a popular phenomenon which aims to provide an alternative to traditional solutions database and data analysis, leading to a revolution not only in terms of technology but also in business. It is not just about storage of and access to data, Big Data solutions aim to analyze data in order to make sense of that data and exploiting its value. One of the current main challenges in Data Mining related to Big Data problems is to find adequate approaches to analyze massive data online (or data streams) [43]. Due to classification methods requires from a previous labelling process, these methods need high efforts for real-time analysis. However, due to unsupervised techniques do not need this previous process, clustering becomes a promising field for real-time analysis. Clustering is perhaps one of the most popular approaches used in *unsupervised machine learning* and in Data Mining [25]. It is used to find hidden information or patterns in an unlabelled dataset and has several applications related to biomedicine, marketing [24], or image segmentation [50] amongst others. Clustering algorithms provide a large number of methods to search for "blind" patterns in data, some of these approaches are based on Bio-inspired methods such as evolutionary computation [21,27], swarm intelligence [9] or neural networks amongst others.

In the last years, and due to the fast growing of large Big Data-based problems, new challenges are appearing in previous research areas to manage the new features and problems that these types of problems produce. New kinds of algorithms, as *online clustering* or *streaming clustering* are appearing to deal with the main problems related to Big Data domains. When data streams are analyzed, it is important to consider the analysis goal, in order to determine the best type of algorithm to be used. We could divide data stream analysis in two main categories:

- *Offline analysis*: we consider a portion of data (usually large data) and apply an offline clustering algorithm to analyze this data.
- *Online analysis*: the data are analyzed in real-time. These kinds of algorithms are constantly receiving new data instances and are not usually able to keep past information. The most relevant limitations of these systems are: the data order matters and can not be modified; the data can not be stored or re-analyzed during the process; the results of the analysis depend on the time

[2] http://aci.info/2014/07/12/the-data-explosion-in-2014-minute-by-minute-infographic/.

the algorithm has been stopped. The main problem of these algorithms is that they need a specific space to update the information. This reduces the possibilities of the new algorithm.

From our previous experience in different complex and industrial problems in different areas from Social Networks Analysis [6,7], Project Scheduling, Videogames, Music classification, Unmmaned Systems, or Bio-informatics, we have designed and developed several bioinspired algorithms for clustering or graph-based computing with the aim to handle Big Data-based problems. We can distinguish from two main types of algorithms, those that have combined evolutionary strategies (mainly genetic algorithms) [42,44] and the second ones which have been designed using swarm intelligence approaches (ant colonies optimization algorithms) [23].

5.2 Social-Based Analysis and Mining

With the large number and fast growing of Social Media systems and applications, Social-based applications for Data Mining, Data Analysis, Big Data computation, Social Mining, etc. has become an important and hot topic for a wide number of research areas [5]. Although there exists a large number of existing systems (e.g., frameworks, libraries or software applications) which have been developed, and currently are used in various domains and applications based on Social Media. The applications and their main technologies used are mainly based on Big Data, Cloud or Grid Computing. The concept of Ephemeral computing has been rarely considered.

Most of the current challenges under study in Social-based analysis and mining are related to the problem of efficient knowledge representation, management and discovery. Areas as Social Network Analysis (SNA), Social Media Analytics (SMA) and Big Data, have as main aims to track, trends discovery or forecasting, so methods and techniques from: Opinion Mining, Sentiment Analysis, Multimedia management or Social Mining are commonly used. For example, when anyone tries to analyze how a Social Network is evolving using a straightforward representation based on a graph, but ignoring the information flow between nodes the information extracted from this analysis will be very limited. Other simple example, based on SNA, is an application that could try to extract behavioral patterns among users connected to a particular social network without taking into account their connections, their strengthens, or how their relationships are evolving through time. Social Big Data analysis, instead, aims to study large-scale Web phenomena such as Social Networks from a holistic point of view, i.e., by concurrently taking into account all the socio-technical aspects involved in their dynamic evolution.

Previous domains could be joined into a more general application area named *Social Big Data*. This area, or application domain, comes from the joining efforts of two domains: Social Media and Big Data. Therefore, Social Big Data will be based on the analysis of very-large to huge amount of data, which could belong to several distributed sources, but with a strong focus on Social media. Hence,

Social Big Data analysis [12,38] is inherently interdisciplinary and spans areas such as Data Mining, Machine Learning, Statistics, Graph Mining, Information Retrieval, Linguistics, Natural Language Processing, Semantic Web, Ontologies, or Big Data Computing, amongst others. Their applications can be extended to a wide number of domains such as health and political trending and forecasting, hobbies, e-business, cyber-crime, counter terrorism, time-evolving opinion mining, social network analysis, or human-machine interaction.

Taking into account the nature of Social Big Data sources and the necessary processes and methods that will be required for data processing, the knowledge models, and possibly the analysis and visualization techniques to allow discover meaningful patterns [29], the potential application of Eph-C features could generate a new kind of algorithms that would be suitably applied in ephemeral environments.

5.3 Artificial Intelligence in Computer Games and Ephemerality

The application of artificial/computational intelligence to games (game AI/CI) has seen major advancements in the last decade and has settled as a separate research field [32,60]. One of the main focus of the research is to provide computers with the capacity to perform tasks that are believed to require human intelligence, and that results in a number of interesting sub-fields such as AI-assisted game design, computational narrative, procedural content generation, non-player-character (NPC) behavior learning, NPC affective computing, believable bots, social simulation, and player modeling, among others [36]. Many of the problems that arise in these areas require creativity [34] and cannot be solved just proficiently but also in a human-like style; many interactions and relations emerge naturally in games what creates a complex system that is usually not easy to understand by a human but that can provide interesting results from a human perspective [56]. Moreover, many games have an ephemeral nature, hard to manage computationally. Some game assets (i.e., game contents, NPC behavior/game AI, game goals and even game rules) can be seen as volatile in the sense that one cannot guarantee they occur again. Thus, it does make sense to consider creating them ephemerally.

Furthermore, the recent boom of casual games played in mobile devices provokes that both the design and gameplay of games demand resources that appear and evaporate continuously during the execution of a game. This precisely occurs in the so-called pervasive games (i.e., "games that have one or more salient features that expand the contractual magic circle of play spatially, temporally, or socially" [45]) where the gaming experience is extended out in the real world. Playing games in the physical world requires computations that should be executed on the fly in the user's mobile device and having into account that players can decide to join or drop out the game in each instant. This same situation happens in most of the multiplayer games.

But we should not focus our attention just to this specific genre of games as many areas of application for Eph-C can be easily found in the game universe. So, it is not unreasonable to think about the concept of ephemeral games as

those games that can be only placed once or that expires in some way; one can find many reasons for their creation as for instance: economic intentions (e.g., the player will be supposed to demand extensions of the game in the future) or creative aspects (e.g., provide unique game experiences by playing a game with irreversible actions). In addition, one can think about ephemeral goals/events that have temporary existence in games as these appear (and disappear) as consequences of the actions and preferences of the players. These goals/events are usually secondary (as the main goal is well-defined and related with the primary story of the game) but help significantly to improve the game experience and thus they are critical to increase user satisfaction (incidentally, the maximum objective of games). Other issue to consider is the reversibility of player's actions: most games provide the option to save the current state to reload it later, basically implying that players do not face the consequences of their acts as they can go back to a previous state; while this is interesting (and desirable) in a number of games, it is also truth that it is inconvenient in certain types of games as in multiplayer on-line games (e.g., first-person-shooter, real-time strategy, or role-playing games, among others) where 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 what grants temporality to the nature of game. This transitory essence of games produces important problems that are difficult to manage computationally, and where and how to create the volatile features of a game is a question that remains opens and that Eph-C can help to solve/mitigate.

Other issue to consider is the energy consumption, specially from the hardware resources. Videogames are played on a wide set of platforms including game consoles, personal computers and mobile devices, and all of them consume a large amount of kilowatt-hours per gaming session (not to mention the energy use of the monitors or other pheripherical devices); moreover, even if the user leaves the game for a while (perhaps hours), the gaming plaftform can consume as much energy as in an active play. In mobile devices this issue particularly affects battery consumption. Can Eph-C help to manage the energy cosumption in gaming sessions without negatively affecting the player's game experience, that is to say, without turning off the game platform or enabling the automatic power-down feature built into the device? This is the question that remains open and provides interesting lines of research. Moreover, in games that demand physical activity (such as dance games or pervasive games) it is important to manage the physical effort of the player; in this context Eph-Cmight help to do it by arranging the ephemeral goals wiht this goal in mind.

5.4 Computational Creativity

Computational Creativity has gained attention in the last few years [40]. The idea is not to exclude human artists from creative processes, substituting them by computer algorithms, but instead to extend human creativity by computer aided processes. Although many approaches to the concept are possible, several models arising from the Computational Intelligence (CI) area have been developed in

the last decade. These models include the possibility of human interaction within the algorithms [57]. Art, design and content generations are areas of interest for both CI and audiovisual industry [58].

Interactive EAs (IEAs) as well as Human-Based Computational Intelligence [31] are interesting starting points in the area. With the advent of IEAs, and their possibilities for developing creativity (applications can be found in music composition [15], videogames plot induction and story generation [41], automatic poetry [39], etc.), a new problem arose: user fatigue due to multiple evaluations required when the fitness function is substituted by a human in charge of aesthetic evaluations. It is thus required to improve available IEAs so that new autonomous software tools can be developed and this implies a better understanding of human creative processes. Recently, a new proposal based on EAs has tried to analyze creativity when developed by human artists [17]. Results have provided clues that may lead in the future to new genetic operators or algorithms. Yet, high computational costs are associated with computer based creative processes [52], and distributed infrastructures are required. Among the possibilities, Eph-C-based models share some features with the way a team of artists can collaborate when developing evolutionary art following 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, where the interaction with the audience and critics is exhibited. Therefore, ephemeral behavior is inherent to the way artists work and react to colleagues and the public. Previous models [17] could be thus studied from this point of view to improve existing methodologies.

6 Conclusions

Ephemeral computation provides an interesting new, and promising, research area with significant differences when it is compared against other areas as grid computing, or traditional distributed computing. Although Eph-C presents some features close to volunteer computing or amorphous computing, the combination of their main features: *inclusion, asynchrony, resilience, emergence, and self-adaptation*, defines it more precisely.

Therefore, the main focus of Eph-C is different from the above approaches. Rather than trying to build layers onto the network of ephemeral resources in order to "hide" their transient nature and provide the illusion of a virtual stable environment, Eph-C applications are fully aware of the nature of the computational landscape and are specifically built to live (and optimize their performance) in this realm.

Related to the application of traditional methods and techniques from Machine Learning to Big Data problems, our previous experience has shown the high performance that bioinspired algorithms can achieve in huge, open and dynamic problems, showing how bioinspired approaches can be used to improve

the performance of unsupervised approaches. In the near future, and taking into account the new restrictions and features imposed by Eph-C environments, a new suit of algorithms able to efficiently handle the new challenges in data management and knowledge discovery in large Big Data-based problems will be studied and analyzed.

Acknowledgements. This work is supported by MINECO project EphemeCH (TIN2014-56494-C4-1-P, -2-P, -3-P and -4-P) – Check http://blog.epheme.ch.

References

1. Abelson, H., Allen, D., Coore, D., Hanson, C., Homsy, G., Knight Jr., T.F., Nagpal, R., Rauch, E., Sussman, G.J., Weiss, R.: Amorphous computing. Commun. ACM **43**(5), 74–82 (2000)
2. Álvarez, J.D., Colmenar, J.M., Risco-Martín, J.L., Lanchares, J., Garnica, O.: Optimizing l1 cache for embedded systems through grammaticalevolution. Soft Comput. **20**, 1–15 (2015)
3. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., et al.: A view of cloud computing. Commun. ACM **53**(4), 50–58 (2010)
4. Babaoglu, O., Jelasity, M., Montresor, A., Fetzer, C., Leonardi, S., Moorsel, A., Steen, M.: The self-star vision. In: Babaoglu, O., Jelasity, M., Montresor, A., Fetzer, C., Leonardi, S., Moorsel, A., Steen, M. (eds.) SELF-STAR 2004. LNCS, vol. 3460, pp. 1–20. Springer, Heidelberg (2005). doi:10.1007/11428589_1
5. Bello-Orgaz, G., Jung, J.J., Camacho, D.: Social big data: recent achievements and new challenges. Inf. Fusion **28**, 45–59 (2016)
6. Bello-Orgaz, G., Menéndez, H., Okazaki, S., Camacho, D.: Combining social-based data mining techniques to extract collective trends from twitter. Malaysian J. Comput. Sci. **27**(2), 95–111 (2014)
7. Bello-Orgaz, G., Menendez, H.D., Camacho, D.: Adaptive k-means algorithm for overlapped graph clustering. Int. J. Neu. Syst. **22**(05), 1250018 (2012)
8. Berl, A., Gelenbe, E., Di Girolamo, M., Giuliani, G., De Meer, H., Dang, M.Q., Pentikousis, K.: Energy-efficient cloud computing. Comput. J. **53**(7), 1045–1051 (2010)
9. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press Inc., New York (1999)
10. Borella, M.S., Grabelsky, D., Nessett, D.M., Sidhu, I.S.: Method and system for locating network services with distributednetwork address translation. US Patent 6,055,236 (2000)
11. Bunse, C., Hopfner, H., Mansour, E., Roychoudhury, S.: Exploring the energy consumption of data sorting algorithms inembedded and mobile environments. In: Tenth International Conference on Mobile Data Management: Systems, Services and Middleware, MDM 2009, pp. 600–607. IEEE (2009)
12. Cambria, E., Rajagopal, D., Olsher, D., Das, D.: Big social data analysis. Big Data Comput. **13**, 401–414 (2013)
13. Cantú-Paz, E.: A survey of parallel genetic algorithms. Calculateurs paralleles reseaux et systems repartis **10**(2), 141–171 (1998)
14. Cotta, C., Sevaux, M., Sörensen, K. (eds.): Adaptive and Multilevel Metaheuristics. SCI, vol. 136. Springer, Heidelberg (2008)

15. Diaz-Jerez, G.: Composing with melomics: delving into the computational world formusical inspiration. Leonardo Music J. **21**, 13–14 (2011)
16. Eiben, A.E.: Evolutionary computing and autonomic computing: shared problems, shared solutions? In: Babaoglu, O., Jelasity, M., Montresor, A., Fetzer, C., Leonardi, S., Moorsel, A., Steen, M. (eds.) SELF-STAR 2004. LNCS, vol. 3460, pp. 36–48. Springer, Heidelberg (2005). doi:10.1007/11428589_3
17. Fernández de Vega, F., Navarro, L., Cruz, C., Chavez, F., Espada, L., Hernandez, P., Gallego, T.: Unplugging evolutionary algorithms: on the sources of novelty and creativity. In: IEEE Congress on Evolutionary Computation (CEC), pp. 2856–2863. IEEE (2013)
18. Flinn, J., Satyanarayanan, M.: Powerscope: a tool for profiling the energy usage of mobile applications. In: Second IEEE Workshop on Mobile Computing Systems and Applications, Proceedings, WMCSA 1999, pp. 2–10. IEEE (1999)
19. Fong, K.F., Hanby, V.I., Chow, T.-T.: HVAC system optimization for energy management by evolutionary programming. Energy Buildings **38**(3), 220–231 (2006)
20. Frei, R., McWilliam, R., Derrick, B., Purvis, A., Tiwari, A., DI Marzo Serugendo, G.: Self-healing and self-repairing technologies. Int. J. Adv. Manuf. Technol. **69**(5–8), 1033–1061 (2013)
21. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Reading (1989)
22. Lombraña González, D., Jiménez Laredo, J.L., Fernández de Vega, F., Merelo Guervós, J.J.: Characterizing fault-tolerance of genetic algorithms in desktop grid systems. In: Cowling, P., Merz, P. (eds.) EvoCOP 2010. LNCS, vol. 6022, pp. 131–142. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12139-5_12
23. Gonzalez-Pardo, A., Camacho, D.: Solving project scheduling problems through swarm-based approaches. Int. J. BioInspired Comput. (IJBIC) (2015, inpress)
24. Haider, P., Chiarandini, L., Brefeld, U.: Discriminative clustering for market segmentation. In: Proceedings of the 18th ACM SIGKDD international conferenceon Knowledge discovery and data mining, KDD 2012, pp. 417–425. ACM, New York (2012)
25. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco (2006)
26. Harmon, R.R., Auseklis, N.: Sustainable it services: assessing the impact of green computing practices. In: Portland International Conference on Management of Engineering & Technology, PICMET 2009, pp. 1707–1717. IEEE (2009)
27. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, Cambridge (1992)
28. Huhns, M.N., Singh, M.P.: Service-oriented computing: key concepts and principles. IEEE Internet Comput. **9**(1), 75–81 (2005)
29. Kaisler, S., Armour, F., Espinosa, J.A., Money, W.: Big data: issues and challenges moving forward. In: 46th Hawaii InternationalConference on System Sciences (HICSS), pp. 995–1004. IEEE (2013)
30. Kamil, S., Shalf, J., Oliker, L., Skinner, D.: Understanding ultra-scale application communication requirements. In: Proceedings of the IEEE International Workload Characterization Symposium, 2005, pp. 178–187. IEEE (2005)
31. Kosorukoff, A.: Human based genetic algorithm. In: IEEE International Conference on Systems, Man, and Cybernetics, vol. 5, pp. 3464–3469. IEEE (2001)
32. Lara-Cabrera, R., Cotta, C., Fernández-Leiva, A.J.: A review of computational intelligence in rts games. In: IEEE Symposium on Foundations of Computational Intelligence, pp. 114-121. IEEE Press, Singapore (2013)

33. Laredo, J.L.J., Castillo, P.A., Mora, A.M., Merelo, J.J., Fernandes, C.: Resilience to churn of a peer-to-peer evolutionary algorithm. Int. J. High Performance Syst. Architect. **1**(4), 260–268 (2008)
34. Liapis, A., Yannakakis, G.N., Togelius, J.: Computational game creativity. In: Proceedings of the Fifth International Conference on Computational Creativity (ICCC 2014) (2014)
35. Lohr, S.: The age of big data. New York Times, 11 February 2012. Online. Accessed 5 Sept. 2014
36. Lucas, S.M., Mateas, M., Preuss, M., Spronck, P., Togelius, J., (eds.) Artificial and Computational Intelligence in Games, vol. 6. Dagstuhl Follow-Ups. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2013)
37. Lyytinen, K., Yoo, Y.: Ubiquitous computing. Commun. ACM **45**(12), 63–96 (2002)
38. Manovich, L.: Trending: the promises and the challenges of big social data. In: Debates in the Digital Humanities, pp. 460–475 (2011)
39. Manurung, H.: An evolutionary algorithm approach to poetry generation. PhD thesis, University of Edinburgh. College of Science and Engineering. School of Informatics (2004)
40. McCormack, J., D'Iverno, M.: Computers and Creativity. Springer, Heidelberg (2012)
41. McIntyre, N., Lapata, M.: Plot induction and evolutionary search for story generation. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 1562–1572. Association for Computational Linguistics (2010)
42. Menéndez, H.D., Barrero, D.F., Camacho, D.: A genetic graph-based approach for partitional clustering. Int. J. Neural Syst. 24(03) (2014a)
43. Menéndez, H.D., Otero, F.B., Camacho, D.: Extending the SACOC algorithm through the Nystrom method for bigdata analysis. Int. J. Bio-Inspired Comput. (2016, in press)
44. Menéndez, H.D., Otero, F.E.B., Camacho, D.: MACOC: a medoid-based ACO clustering algorithm. In: Dorigo, M., Birattari, M., Garnier, S., Hamann, H., Montes de Oca, M., Solnon, C., Stützle, T. (eds.) ANTS 2014. LNCS, vol. 8667, pp. 122–133. Springer, Heidelberg (2014). doi:10.1007/978-3-319-09952-1_11
45. Montola, M., Stenros, J., Waern, A.: Pervasive Games. Morgan Kaufmann, Boston (2009)
46. Network for Sustainable Ultrascale Computing. The future of ultrascale computing under study (2014). Online, Accessed 8 Sept. 2014
47. Nogueras, R., Cotta, C.: Studying fault-tolerance in island-based evolutionary and multimemetic algorithms. J. Grid Comput. (2015a). doi:10.1007/s10723-014-9315-6
48. Nogueras, R., Cotta, C.: Studying self-balancing strategies in island-based multimemetic algorithms. J. Comput. Applied Math. (2015b). doi:10.1016/j.cam.2015.03.047
49. Nogueras, R., Cotta, C.: Towards resilient multimemetic systems on unstable networks with complex topology. In: Papa, G. (ed.) Advances in Evolutionary Algorithm Research. Nova Science Pub. (2015c, in press)
50. Pascual, A., Barcéna, M., Merelo, J.J., Carazo, J.-M.: Application of the fuzzy Kohonen clustering network to biological macromolecules images classification. In: Mira, J., Sánchez-Andrés, J.V. (eds.) IWANN 1999. LNCS, vol. 1607, pp. 331–340. Springer, Heidelberg (1999). doi:10.1007/BFb0100500

51. Prasithsangaree, P., Krishnamurthy, P.: Analysis of energy consumption of RC4 and AES algorithms in wireless LANs. In: Global Telecommunications Conference, GLOBECOM 2003, vol. 3, pp. 1445–1449. IEEE (2003)
52. Reis, G., de Vega, F.F., Ferreira, A.: Automatic transcription of polyphonic piano music using genetic algorithms, adaptive spectral envelope modeling, and dynamic noise level estimation. IEEE Trans. Audio Speech Lang. Process. **20**(8), 2313–2328 (2012)
53. Sarmenta, L.F., Hirano, S.: Bayanihan: building and studying web-based volunteer computing systems using Java. Future Gener. Comput. Syst. **15**(5), 675–686 (1999)
54. Sharmin, M., Ahmed, S., Ahamed, S.I.: SAFE-RD (secure, adaptive, fault tolerant, and efficient resource discovery) in pervasive computing environments. In: International Conference on Information Technology: Coding and Computing, ITCC 2005, vol. 2, pp. 271–276. IEEE (2005)
55. Stutzbach, D., Rejaie, R.: Understanding churn in peer-to-peer networks. In: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement, IMC 2006, pp. 189–202. ACM, New York (2006)
56. Sweetser, P.: Emergence in Games. Game Development. Charles River Media, Boston (2008)
57. Takagi, H.: Humanized computational intelligence with interactive evolutionary computation. In: Fogel, D.B., Robinson, C.J. (eds.) Computational Intelligence: The Experts Speak, pp. 207–218. Wiley (2003)
58. Togelius, J., Yannakakis, G.N., Stanley, K.O., Browne, C.: Search-based procedural content generation: a taxonomy and survey. IEEE Trans. Comput. Intell. AI Games **3**(3), 172–186 (2011)
59. Wang, B., Bodily, J., Gupta, S.K.: Supporting persistent social groups in ubiquitous computing environments using context-aware ephemeral group service. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications, PerCom 2004, pp. 287–296. IEEE (2004)
60. Yannakakis, G., Togelius, J.: A panorama of artificial and computational intelligence in games. IEEE Trans. Comput. Intell. AI Games **7**(4), 317–335 (2015)