J Reliable Intell Environ (2015) 1:75–86
DOI 10.1007/s40860-015-0010-6

ORIGINAL ARTICLE

Approaches for engineering adaptive systems in ubiquitous and pervasive environments

Mohamed Bakhouya[1] · **Jaafar Gaber**[2]

Received: 12 August 2015 / Accepted: 16 October 2015 / Published online: 20 November 2015

Abstract This paper describes existing work related to the development of adaptive systems and approaches in ubiquitous and pervasive environments and sheds more light on how features from natural and biological systems could be exploited for engineering adaptive systems. Ubiquitous and pervasive systems are composed of different heterogeneous parts or entities that interact and perform actions favoring the emergence of global desired behavior. Furthermore, in this type of systems entities might join or leave without disturbing the collective, and the system should self-organize and continue performing their goals. Therefore, entities must self-evolve and self-improve by learning from their interactions with the environment. In this paper, the main challenges for engineering these systems are presented by putting more emphasis on the design and the development of distributed and adaptive algorithms that allow system entities to select the best suitable strategy/action in order to drive the system to the best suitable behavior according to the current state of the system and environment changes. We also highlight specific aspects being investigated via illustrative examples in order to show the usefulness of natural and biological system principles for developing adaptive approaches.

Keywords Pervasive and ubiquitous computing · Adaptive systems · Self-CHOP · Natural and biological systems · Bio-inspired algorithms

✉ Mohamed Bakhouya
mohamed.bakhouya@uir.ac.ma

Jaafar Gaber
gaber@utbm.fr

[1] International University of Rabat, 1100 Sala El Jadida, Morocco

[2] Universite de Technologie de Belfort Montbeliard, Rue Thierry Mieg, 90010 Belfort, France

1 Introduction

Ubiquitous and pervasive computing is an emerging paradigm that aims to provide users with access to services all the time, everywhere and in a transparent way, by means of devices embedded in the surrounding physical environment and/or carried by the user [57]. The goal is to develop environments where highly heterogeneous hardware and software resources can seamlessly and spontaneously interoperate, in order to provide a variety of services to users regardless of the specific characteristics of the environment and user devices. As stated in [29], the aim of ubiquitous computing is to provide any mobile device an access to available services in an existing network all the time and everywhere while the main objective of pervasive computing is to provide spontaneous services created on the fly by mobiles that interact by ad hoc connections. In other words, pervasive computing concerns the use of mobile computing technology and recently data coming from social networks to enhance people's interactions during unexpected contexts.

Gaber [29] also points out that most of research works to date in resource management are based on the traditional Client–Server paradigm (CSP). However, this paradigm is impracticable in ubiquitous and pervasive environments and cannot meet simultaneously their related needs and requirements that are scalability and adaptability to dynamic environments. Indeed, network resources must be able to scale, adapt to dynamic conditions in the network, be highly available, and should require minimal human configuration and management [10]. According to Gaber's classification, two alternative paradigms to CSP have been introduced to design and implement ubiquitous and pervasive applications: the adaptive Services-to-Client Paradigm (SCP) and the Spontaneous Service Emergence Paradigm (SEP) [28]. In order to carry out these paradigms, a decentralized and

self-organizing middleware should be able to provide services to users according to their availability and the network status. Such a middleware can be inspired, for example, from a biological system like the natural immune system.

Furthermore, embedded computing systems and devices have also become widespread in various application domains, as evident from their use in products such as PDAs, household appliances, and automotive systems. This has been further accelerated by the advances in silicon technology, which had led to the design of complex SoCs (System-on-Chip) that incorporate many hardware and software block cores [6,14]. For example, most of the microsystem solutions, such as the ones present in mobile communication devices, are a combination of ASICs, microcontrollers, and digital signal processors. However, recent developments in these domains brought new challenges to designers and developers, such as the fact that these systems must provide high-performances while consuming as little energy as possible. Furthermore, future mobile communication terminals will have to support many applications, ranging from web browsing/navigation to real-time multimedia applications such as audio and video communication.

The design of such systems should be highly flexible, adaptable, and meet stringent time-to-market constraints, while providing high-performance and lower energy consumption. Therefore, new methodologies are required to solve the growing complexity of these systems, especially when large number of cores must to be integrated on a single chip. Furthermore, adaptive and dynamic reconfiguration approaches need to be developed to allow these systems to self-configure according to their environments and running applications.

The aim of the work presented in this paper is towards solving some of the technical challenges in realizing adaptive systems. The general context of the research is the design and development of distributed and adaptive algorithms that allow system components to select the best suitable strategy/action and drive the system to the best suitable behavior according to the current state of the system and environment changes. The main goal is to develop run-time mechanisms so that the system autonomously adapts its structure and its behavior during the course of operation. However, several challenges must be overcome to make these computing systems a practical infrastructure for emerging ubiquitous and pervasive applications.

The remainder of this paper is structured as follows. Section 2 presents existing research directions in developing adaptive systems. Section 3 highlights features from natural and biological systems and how they can be used for engineering adaptive approaches. In Sect. 4, we briefly describe some results from past and ongoing work for developing bio-inspired and adaptive approaches. Conclusions and perspectives are given in Sect. 5.

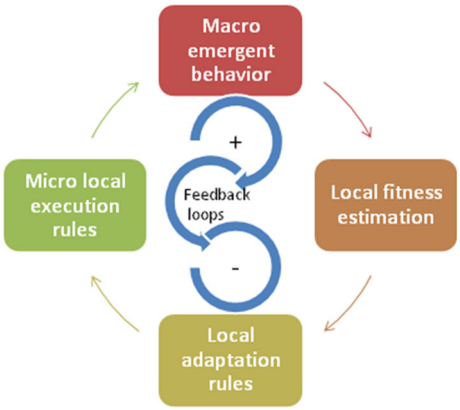

Fig. 1 Design paradigm for autonomic/adaptive systems

2 Related work

During the past few years, research in artificial intelligence, agent-based systems, mobile and autonomous robots, distributed systems, and autonomic systems, has focused on the development of adaptive approaches and systems that modify their own behavior at run-time to address constantly changing environments. Some of these approaches are inspired by features and capabilities seen in natural and biological systems, e.g., human brain, immune systems, ant colony, flocks of birds [10,18,21]. The capabilities of these systems have been exploited in a variety of computation systems and been perceived as an efficient system model for developing adaptive systems and reconfigurable/evolvable hardware systems [3,30–32,47,55]. The objective of such research is to develop autonomic systems with self-aware (e.g., self-configuration, -organization, -optimization) properties at component level and strengthen the self-design and fault-tolerance aspect (emergence of self-* [4,6,46]).

Recent studies have emphasized that designing adaptive systems requires a shift from the current top-down design approach to a bottom-up design approach [39,46]. In a bottom-up design approach as illustrated in Fig. 1, local rules allow system components to collaborate in a distributed manner in order to enable the emergence of behaviors at a global level. However, designing and engineering autonomic/adaptive systems requires answering the following research questions: (1) how to design basic system components in which decisions are distributed and not fully controlled by a single component? (2) How to design strategies (at micro-level) that allow the system to adapt to environment changes (at macro-level) by selecting the best suitable actions/strategies? (3) What are the dynamic rules that drive the system to the expected behavior (i.e., reliable, performance and energy efficient)? (4) What are techniques and tools for studying the effectiveness of these mechanisms and evaluating the expected functionalities and performance metrics?

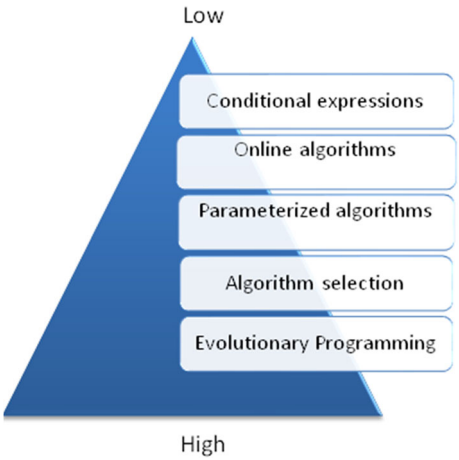

Fig. 2 Adaptation rules and techniques

The main goal is to develop run-time mechanisms so that the system autonomously adapts its structure and its behavior during the course of operation. From algorithms perspectives, several techniques could be used to develop algorithms with increasing level of adaptiveness as illustrated in Fig. 2 [34]. The first technique uses *if/switch statements* to evaluate the local function or expression to select a suitable action. Online parameterized techniques are used to select an action based on inputs and parameters that can evolve over time. The algorithm selection technique chooses the most effective algorithm among a fixed set of available algorithms based on given properties, for a specific task or environment state. The AI-based learning and evolutionary programming provide techniques to select suitable actions and generate new actions. For example, a mechanism inspired by the immune system is proposed in [35,56] for intelligent selection of actions by a mobile robot; it was adapted from a model proposed in [27], in which the authors describe a nonlinear dynamical model using differential equations for the immune system based on the immune idiotypic network hypothesis proposed in [36]. The use of linear equations formulation and iterative methods, which is preferred to a nonlinear system or coupled differential equations that can have multiple attractors, to model adaptive behaviors was proposed in [30]. Action selection algorithms for adaptive behavior emergence can be then modeled by a simple linear system solving. The immune system model has been used in several artificial intelligence approaches [9,30,53].

Despite existing work in developing adaptive algorithms, several challenges must be tackled in order to carry out the bottom-up design approach for engineering adaptive systems. For example, the design and development of adaptive distributed mechanisms, called also self-* features [39], following this bottom-up design paradigm have been mainly studied to develop large and self-adaptive distributed systems. Recently, researchers from the software engineering community have clearly stated that building self-adaptive systems is a major challenge and put emphasis on the effectiveness of using theories from control engineering, with well-established mathematical modeling tools for performance evaluation and stability study, and natural systems [20]. They have highlighted that *feedback loops* are core design elements and should be made explicit in modeling, design, implementation, and validation approaches [17]. Autonomic computing communities have indirectly exploited feedback loops to develop systems that manage themselves according to an administrator's goals. In fact, the IBM concept of MAPE-K (monitor, analyze, plan, execute over a knowledge base) can be also seen as a feedback loop [38].

It is worth noting that in the control engineering field, research has focused on the design and development of complex adaptive systems by emphasizing positive and negative feedback loops also seen in natural and biological systems. Complex systems are *complex* because of the multiple *feedbacks/interactions* among the various components of the system. In other words, actions taken on an element in a system might result in changes in the state of the element and these, in turn, might bring about changes in other linked elements. The effects may trail back to the first element, this is called feedback that can be positive or negative. Positive or self-reinforcing feedback amplifies the current change in the system. Negative or self-correcting feedback seeks balance and provides equilibrium by opposing the changes taking place in the system. The two types of feedback should be combined to insure the stability of the system [16,19,26,42,50]; positive feedback alone pushes the system beyond its limits and, eventually out of control, while negative feedback alone prevents the system from reaching its optimal behavior.

Recently, Jones and De Florio in [23,37] highlight the challenges for designing complex systems in uncertain environments. Authors put emphasis on the necessity of developing tools and techniques for managing interactions of massive numbers and types of entities and for predicting and controlling emergent behavior. Taleb in his book "Antifragile: Things That Gain from Disorder" [54] point out that current systems are fragile since they are developed according to requirements that are defined in advance. He introduced antifragile systems that adapt to uncertain and dynamic environments and therefore, similar to biological and natural systems, become stronger when stressed. These authors highlight clearly that a new design philosophy is required for developing antifragile systems that are able to learn and evolve to better perform in unexpected and uncertain environments [23,24,37,54].

Previous and ongoing studies clearly show the potential of using principles from natural and biological systems for designing adaptive systems. However, despite the variety

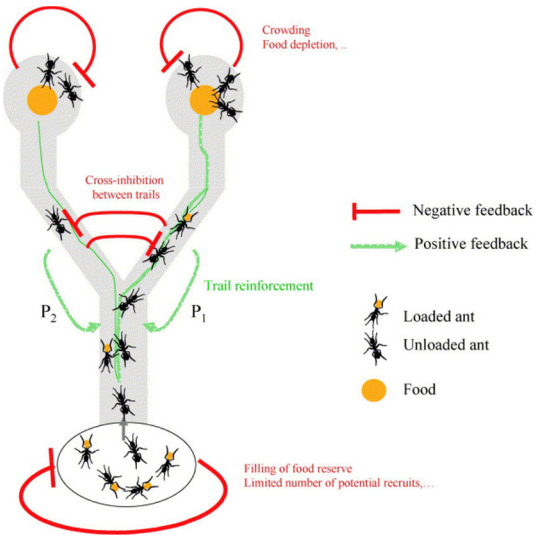

Fig. 3 Feedback loops between ants [25]

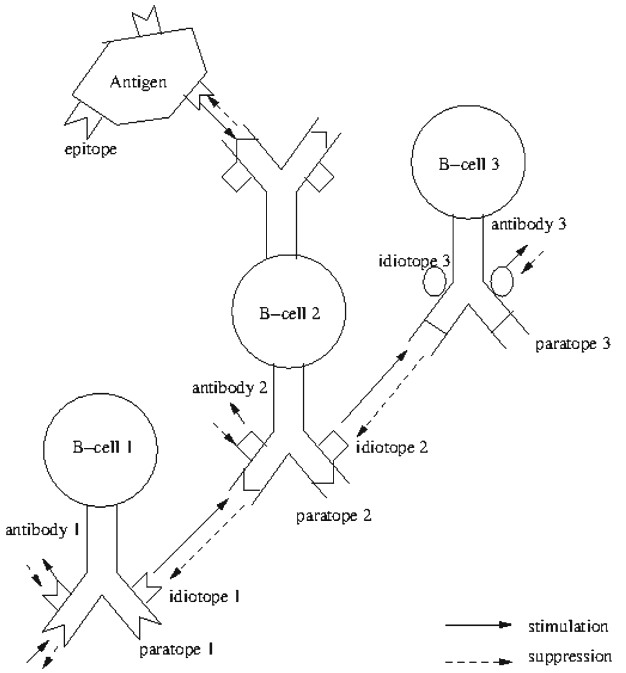

Fig. 4 Feedback loops between B-cells [36,56]

of existing models, there is still no general methodology for designing local adaptation rules. Thus, modeling and studying mechanisms for adaptive systems, while in parallel developing tools required to understand and evaluate them, remains an important and open challenge that needs to be addressed.

3 Natural and biological systems

Swarm intelligence is the discipline that deals with natural and artificial systems composed of many individuals that coordinate their activities using decentralized control and self-organization principles [18]. As stated by Bonabeau et al. in [15], self-organization is a set of mechanisms whereby structures emerge at the global level of a system from interactions of its lower-level entities. In particular, this principle allows the emergence of collective behaviors that result from the local interactions of the individuals with each other and with their environment. Ants and bees colonies are well studied examples of systems by swarm intelligence community having self-organizing features [15]. Most of research works have focused on the development of adaptive approaches and systems that modify their own behavior at run-time to address constantly changing environments. Authors also put more emphasis on positive and negative feedback loops as a main base of self-organization [15]. For example, biological and natural systems, such as Immune systems, honey Bee, and Ant colonies, have several features and organizing principles (i.e., feedback loops as depicted in Figs. 3, 4, 5) that can be exploited in designing and developing adaptive systems. More precisely, these superorganisms often use feedback loops that allow the system achieving reliable and robust solutions using information gathered from entities [49]. As also stated by Kholodenko in [40], positive and negative feedback loops are key elements of information processing in all biological systems. These feedback loops allow improving information flow and decision making at multiple levels, without centralized control.

For example, the waggle dance stop signals by bees could be seen as a positive feedback to attract the attention of other entities about foraging at a specific location [1,49]. The positive and negative selections and stimulation/suppression in the immune system and the pheromone evaporation and deposit by Ants could be seen as feedback loops [25,27,36,56] (Fig. 3). The biological immune system can be seen as a *massively distributed architecture* with a diverse set of cells distributed throughout the body but *communicating* using chemical signals. There is *no central control* (i.e., distributed); the multitude of independent cells work together resulting in the *emergent behavior* of the immune system. The immune system *evolves to adapt* and improve the overall system performance (e.g., organizational memory).

These systems can be seen as complex collective systems in which the behavior emerges from the product of interactions between individual entities. These entities followed a simple set of rules (i.e., not via top-down mechanism) and react only to their local environment. These features and principles (e.g., bottom-up mechanisms, feedback loops) could be used for designing a scalable, adaptive and efficient framework to bring answers to some of the research questions mentioned above.

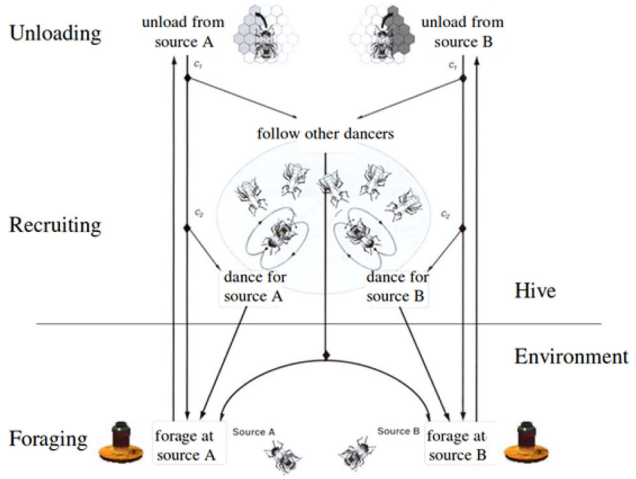

Fig. 5 Feedback loops between bees [41]

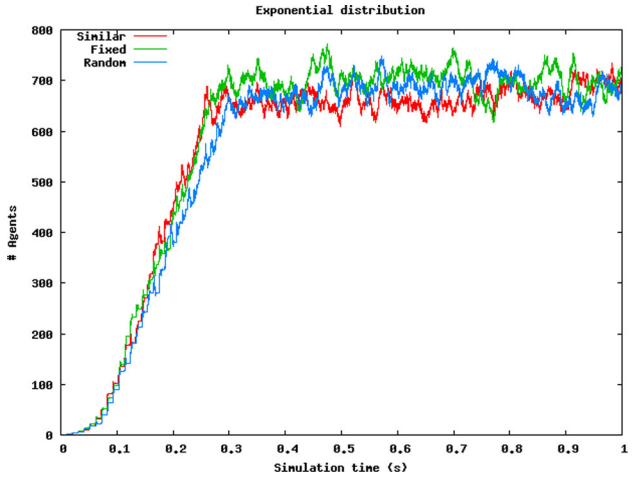

Fig. 6 The evolution of mobile agents' population with exponential distributions

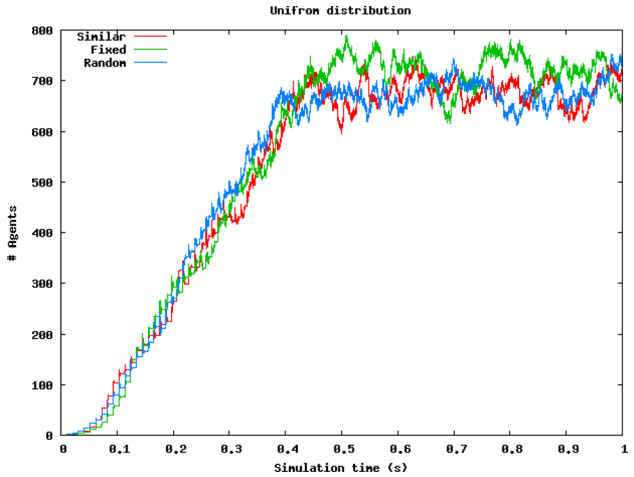

Fig. 7 The evolution of mobile agents' population with uniform distributions

4 Illustrative examples

In this section, we highlight specific aspects being investigated and concern the development of adaptive approaches and follow the *bottom-up design paradigm* in order to shed more light on the usefulness of *natural and biological system* principles for developing adaptive approaches.

4.1 Adaptive resources discovery

A resource discovery approach based on mobile agent paradigm and inspired by the human immune system has been proposed to dynamically regulate the population size of mobile agents that can clone themselves in large distributed environments without any centralized control or global information gathering. Each agent is equipped by a controller equivalent to the immune idiotypic network. An antigen corresponds to the inter-arrival time of agents to a node and provokes an adaptive immune response. Mobile agent behaviors (i.e., actions) are *death* or *kill*, *move* and *clone* and are linked with a stimulation/suppression feedback loop [5,9,13]. Formally x_c, x_m and x_k can be considered as the concentrations associated, respectively, with the *clone*, *move* and *kill* behaviors (i.e., B-cells). Their variations can be expressed for example as follows:

$$\dot{x}_c = (x_k - x_m + m_c - K_c)x_c$$
$$\dot{x}_m = (x_c - x_k + m_m - K_m)x_m$$
$$\dot{x}_k = (x_m - x_c + m_k - K_k)x_k$$

where the values K_c, K_m, and K_k are constants and denote the dissipation factor representing the antibody's natural death of the behavior *clone*, *move* and *kill*, respectively. Variables m_c, m_m, and m_k correspond to the affinity of the antigen with the three respective behaviors (i.e., B-cells). Figures 6 and 7 show the evolution of dynamic agent population size during the simulation when the Uniform and Exponential distributions are applied [5,9].

Another approach for service discovery in large scale network was proposed in [11]. This approach allows organizing resources into communities by creating dynamic affinity relationships with feedback loops to represent services in the network. Peers (i.e., servers) are organized into communities by the creation of affinity relationships, like the idiotypic network [36] created by human immune cells (i.e., peers) against foreign antigens (i.e., user requests). In other words, hardware and software resources are organized into decentralized communities. A community represents a composite service, a set of hardware or software resources that users need to discover and select. In this approach, to create communities, services establish relationships between themselves based on their affinity. Affinity corresponds to the adequacy with

which two services could bind to create a composed service or to point out a similar service.

The establishment of relationship affinities between peers helps solve user requests through collaborations without prior planning. A reinforcement learning mechanism, in the form of *feedback loops*, is used as a gradient ascent/descent, to adjust and dynamically reinforce relationship affinity values according to delivered responses (i.e., user reward/penality). In other words, this reinforcement learning mechanism was used to dynamically adjust and reinforce relationship affinity values according to delivered responses in order to cope with dynamic changes in the network, e.g., services availability and user requests. Therefore, new communities may be created or others may be modified according to dynamic environment changes. Peers may acquire new memberships to new communities or drop themselves from current ones by establishing or deleting affinity relationships. Inside communities, affinity relationships are adjusted as follows:

$$m_{ij}^{(s)}(k+1) = m_{ij}^{(s)}(k) + \mu(satLocal_{ij}^{(s)} - f(m_{ij}^{(s)}(k))),$$

where $m_{ij}^{(s)}$ is the value of the affinity between a resource of the server i and a resource of a server j for a particular service s. f is the logistic equation $f(m_{ij}) = \frac{1}{1+\exp(-m_{ij})}$, μ is a positive value between 0 and 1. $satLocal_{ij}^{(s)}$ is equal to 0 or 1 based on local reward/penality for a particular service s. When all required resources are discovered, the path computed between an end point in the community and the initial entry point will be further reinforced globally by secondary affinity adjustments. The affinity variation for a particular request between a server s_i and a server s_j is determined as follows:

$$\Delta m_{ij}^{(s)}(k) = \mu(satGlobal_{\varphi}^{(s)} - f(m_{ij}^{(s)}(k))),$$

where $satGlobal_{\varphi}^{(s)}$ is the global reward/penalty value regarding the provided service s.

Figure 8 compares a random walk technique with a biased walk technique using the reinforcement learning mechanism. The results show that without reinforcement learning, each peer has no knowledge of the distributed resources provided by other peers and, consequently, the request resolution time is high. Using the reinforcement learning mechanisms, as more time elapses, peers learn from delivered responses leading to an improved performance in request resolution. Furthermore, the biased walk using reinforcement learning provides better results in terms of found resources than the random walk technique as depicted in Fig. 9.

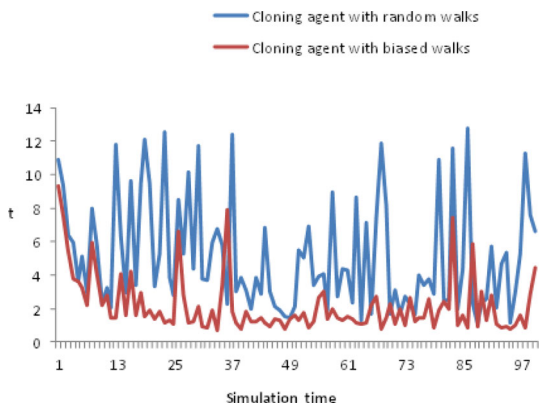

Fig. 8 Request resolution time

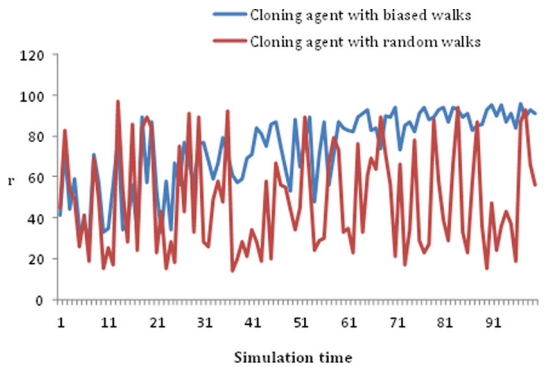

Fig. 9 Discovered resources

4.2 Adaptive broadcasting

The general context of this work is to develop adaptive broadcasting approaches for ad hoc networks (e.g., MANETs, VANETs, WSNs). Most of routing protocols in these networks assume a simplistic form of broadcasting also called flooding. The flooding algorithm is trivial and simple to implement, each node rebroadcast received a message exactly once. However, the flooding is not reliable since most of the nodes are expected to rebroadcast a message at the same time, thus collisions are likely to occur. Furthermore, an increasing number of redundant broadcast messages (called broadcast storm problem) will increase resource utilization, which indirectly affects network performance [48]. More precisely, as rebroadcasting causes trade-off between reachability and efficiency, the core problem is finding a way to minimize the number of redundantly received messages in order to save transmission energy while, at the same time, maintaining good latency and reachability. Therefore, the selection of relay nodes is a major design consideration in broadcasting algorithms.

Several algorithms have been proposed in literature to alleviate the broadcast storm by inhibiting some nodes from rebroadcasting. These algorithms either depend upon certain threshold (e.g., distance, redundant message counts, or

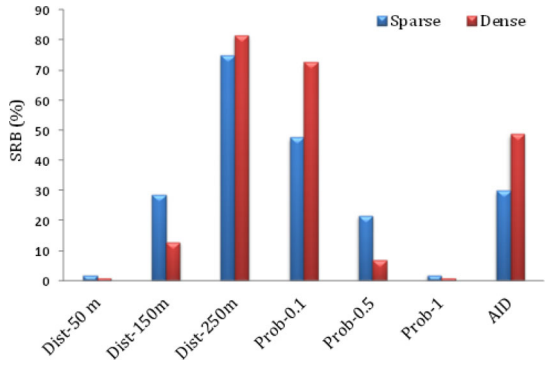

Fig. 10 SRB-save broadcast

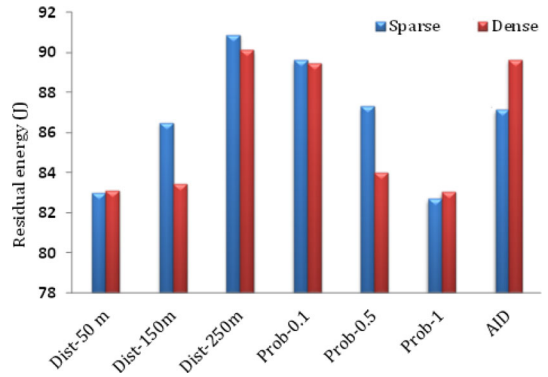

Fig. 11 Remaining energy

broadcast probability) values to estimate the network density, or use sophisticated structures or neighbor topology information to construct a broadcast schedule. However, in dynamic networks, it is difficult, or even impossible, to determine a priori these thresholds (e.g., probability, counter, distance), and to maintain neighbor topology, which requires extra overhead. In this direction, a decentralized and adaptive approach for information dissemination (AID) in dynamic networks is presented in [12,44]. Each node, based on the number of received messages, decides whether or not to rebroadcast a message without the aid of a central controller.

Figures 10 and 11 show the SRB (Saved ReBroadcast) and energy consumption in the context of MANETs. As expected, rebroadcasting causes a trade-off between energy efficiency and SRB [44]. As more rebroadcasts are sent, more energy is consumed. For example, when the distance threshold is fixed to 50 m, more messages are submitted, and then more energy is consumed, but higher reachability is achieved. When the distance threshold is higher (250 m), fewer messages are sent and reachability drops to lower levels, but less energy is consumed. The AID scheme was also evaluated in the context of VANETs. The AID scheme is a more efficient alternative protocol since it increases the number of SRB and the network becomes less congested, resulting in shorter end-to-end message delays.

Other swarm-based distributed broadcasting approaches inspired by Ants and Bees direct and indirect communication principles for VANETs are proposed in [45]. For example, when an abnormal environmental event is noticed on the road surface, a safety message is created to inform other vehicles and roadside units along its way. This is similar to Ant/Bee behavior, i.e., when an Ant/Bee observes a food source it creates pheromone/dance to convey indirectly to other Ants/Bees about route information of that food source. Similarly, when a vehicle v_i observes an event p_j that needs to be disseminated to other vehicles, it will generate a safety message m_{p_j} and will report to RSU (Road Side Unit). This message includes a timestamp t_0, the location information, and an initial relevance value $R^0_{v_i,p_j}(t_0)$ and is disseminated periodically up to a time T, which represents the maximum timespan required to handle the event.

When a node v_k receives a message from another node v_ℓ, we can differentiate between two strategies, G1 and G2. By the strategy G1, information in the header, which is generated by the source node, will not be changed by receivers. Using G2, the receiver node uses the relevance value of intermediate (sender/forwarder) nodes instead of the initial (original) relevance generated by the source node. For example, using G2, a node v_k calculates the new relevance value using the node's v_ℓ information as follows:

$$R_{v_k,p_j}(t+\tau) = \frac{2 * R_{v_x,p_j}(t)}{1+\exp(\frac{d+\lambda\tau s}{D})},$$

where d is the distance between the current location of receiver vehicle v_k and the location where the event is appeared (source). s is the current speed of v_k. The quantity of $\lambda\tau s$ represents the influence of distance variation during the assessment delay τ. λ is a sign, representing direction of the vehicle: -1 (resp. $+1$) if it moving toward (resp. opposite direction) the accident location. It is worth noting that the value of λ equal 1 will cause positive output for R_{v_k,p_j}, which ensures a monotonic increasing function. the value of λ equal -1 implies a negative output for R_{v_k,p_j} and a monotonic decreasing function [29].

Figures 12 and 13 depict relevance values obtained by centralized and distributed approaches. The centralized approach is inspired by bee colony principles, in which communications are indirect via RSUs. The distributed approach is inspired by ants, in which communications and relevance values updating is influenced by intermediate nodes.

4.3 Adaptive transmission range

Recently, nanonetworks have emerged as a communication mechanism to allow tiny nanomachines with limited sensing, computational, storage and power capabilities, communicating using electromagnetic, molecular (i.e., chemi-

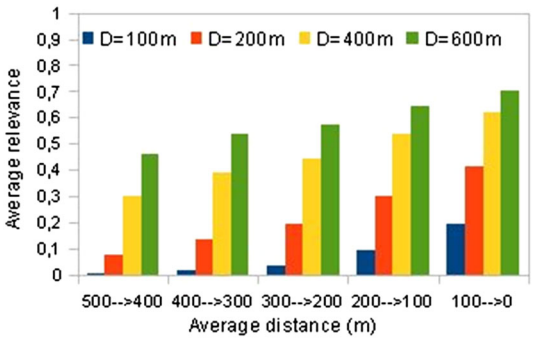

Fig. 12 Distributed approach (ants based)

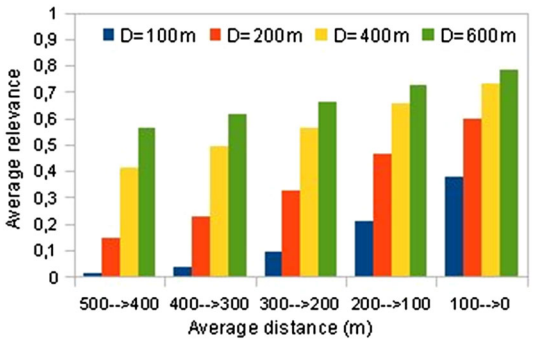

Fig. 13 Centralized approach (bees based)

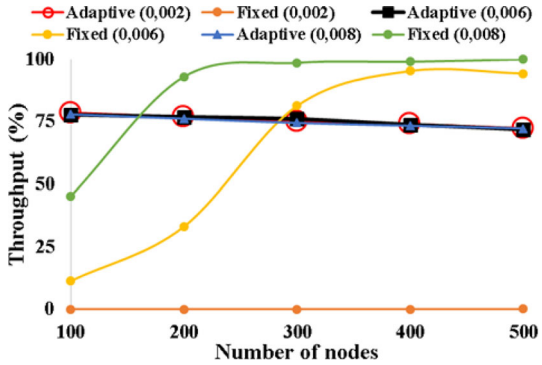

Fig. 14 Throughput vs number of nanonodes

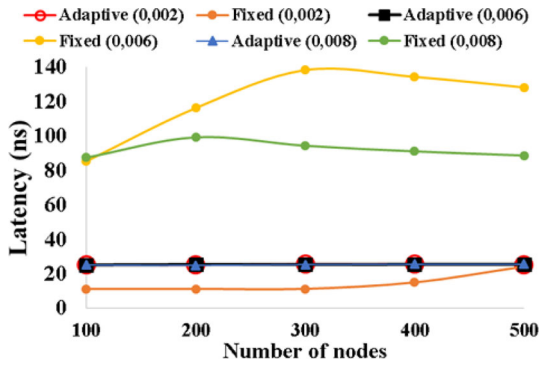

Fig. 15 Latency vs number of nanonodes

cal), nanomechanical, or acoustic communications [2,51]. However, new cost-effective solutions are required for communication among thousands of distributed nanomachines. For example, selecting effective transmission range is a critical issue in design and performance of nanonetworking applications. In high dense network, increasing the transmission range allows quickly the network to sustain higher throughput while maintaining low latency. In low dense network, even with high transmission range, the network could sustain lower throughput. Our preliminary results confirm the research hypothesis stating that different transmission range values need to be assigned based on network density. The aim is to minimize contentions while maintaining good latency and high throughput under different node densities. However, selecting dynamically and in a distributed manner the best suitable range for each node is a difficult issue. In this work, an adaptive transmission range of electromagnetic-based communication mechanism was introduced in [13]. Results showed that this approach provides a good throughput while minimizing latency.

Figure 14 shows the effect of network density on throughput using two strategies: fixed and adaptive. In the beginning of simulations initial values of transmission ranges are same (equal) for both strategies. For the fixed strategy, the transmission range does not change during the simulation, while in the adaptive strategy the initial value changes as simulation time elapse. Simulations are conducted with different transmission ranges (0.002, 0.006 and 0.008). The number of nodes is varied from 100 to 500 in same area by changing the density from sparse to dense networks. Simulations show that when we use fixed transmission range, depending on network density, results for throughput might be different. When we assign a low value for transmission range and fix it, in sparse network, throughput is low, because the probability of finding neighbors in this transmission range is very low; while in dense and medium network, results of throughput gets better. When we use the adaptive strategy, results are more stable for any network density. In this case throughput gets about 75 %.

Figure 15 represents the comparison of average latency when we adapt and fix transmission ranges with dense, medium and sparse networks. Simulation results confirm that the adaptive transmission range strategy outperforms the fixed transmission range strategy in terms of average latency. When the transmission range is fixed to 0.002 the average latency is very low, but throughput value is almost near zero. In this case, if packets are delivered to the destination node, obviously, distance between source and destination nodes is small. In case of fixing the transmission range to 0.006, we observe a high latency compared to other values. It is due to the high number of hops needed for packets to reach the destination node. When we fix the transmission range to

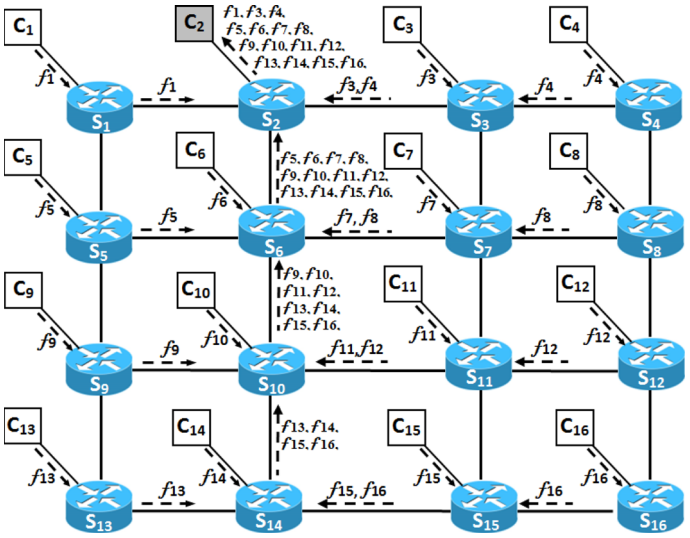

Fig. 16 2D mesh with hotspot traffic pattern

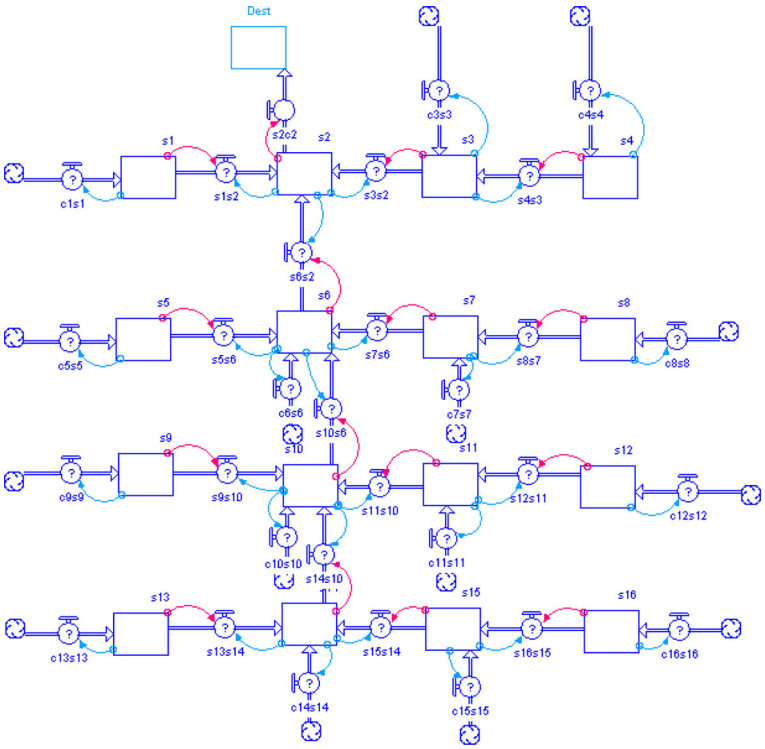

Fig. 17 Feedback loops model using system dynamics

0.008 it can be seen that latency is lower and throughput is higher. However, using adaptive transmission range the average packet delivery time gets smaller for all nanonetworks. It could be observed that since we use adaptive transmission range the average latency is small (between 20 and 30 nanoseconds). This result sheds more light on the usefulness of adaptive approaches in dynamic environments. As shown in these figures, adapting transmission range of nanonodes provides good results in terms of throughput and latency. In case of using adapting rules in a sparse network, the transmission range increases and the average latency and throughput improves.

4.4 Run-time mechanism for congestion avoidance

We have developed a run-time mechanism for congestion avoidance to allow Network-on-chip (NoC) elements to dynamically adjust their inflow by using a feedback control-based mechanism [8]. An NoC is an-chip interconnect infrastructure used in system-on-chip designs to integrate hardware resources or cores [52]. More precisely, these systems are composed of several processing elements (PEs), i.e., dedicated hardware and software components that are interconnected by an NoC. Because of limited processing capacity of the switch and link's bandwidth, incoming packets must

be stored in local buffers before their transmission. These buffers are required to absorb differences in switches speed and burstiness traffic exchanged between the cores. However, because resources are shared, congestion or bottlenecks may be created in some switches, and therefore leading to poor performance. When the network is congested, techniques are required to allow re-routing at run-time traffic from the congested area or by dynamically changing link bandwidth.

The approach we have introduced in [8] mainly uses principles from feedback theory models and systems dynamics techniques [33,50] to prevent the NoC from being overloaded at run-time and then avoid buffer overflows inside the switches. The communication between two cores is characterized as flows that are represented by sequences of hops. The behavior of these data flows can be modeled using a system dynamics mechanism as follows:

$$\dot{x_i} = \left(\lambda_{ji} + \sum_{k \in U_i} \alpha_{ki}\right) - \left(e_{ij} + \sum_{k \in D_i} \alpha_{ik}\right).$$

In this evaluation, 4×4 2D mesh on-chip interconnect is considered as a case study (see Fig. 16), but the approach could be used for any on-chip interconnect. The system dynamics model illustrating feedback loops is depicted in Fig. 17. We first considered hotspot traffic pattern in which the core c_2 was selected to receive all traffic from other cores. This pattern represents the worst case in which all data flows are directed to one core. Indeed, all source cores (except the sink c_2) simultaneously sending data to the on-chip interconnect could lead to excessive load or congestion that impacts the overall performance. The *XY* routing mechanism is used to route flits between source cores and the hotspot core. In this routing technique, flits are first routed along the *X* axis until they reach the column where lies the destination core and then routed along the *Y* axis.

Figure 18 shows the variation of buffer occupancy at four switches using only analytical evaluation. It shows the total buffer occupancy at these four congested switches. The transmission rates of sources are then adapted to buffers backlog under the size of the sender bucket (core or switch). The control mechanism prevents buffers from overflowing, i.e., the buffer occupancy is always under the actual size limit. For example, the total buffer occupancies at switches s_6 and s_{10} reach the maximum, 100 flits, because these switches have four input buffers and 1 output buffer involved in the data transmission, each has a buffer size fixed to 20 flits. For switches s_2 and s_{14} buffer occupancies reach their maximum (80 flits), since both have three inputs and one output buffer that are receiving and sending flits. The values obtained from the simulations (Fig. 19) match well those obtained from the model. These results confirm the usefulness of including this control mechanism to guarantee the boundedness of the buffer queue lengths.

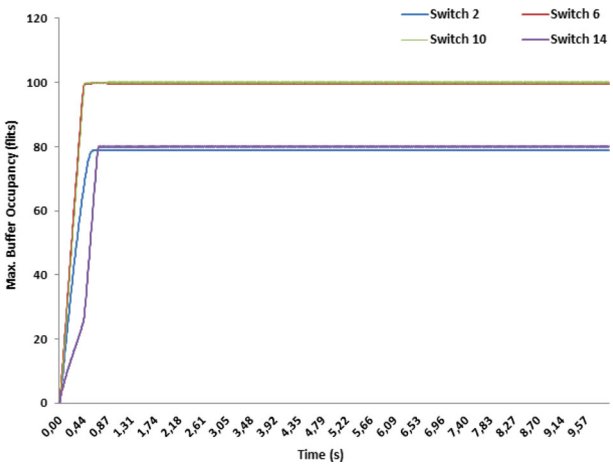

Fig. 18 With adaptation using analytical evaluation

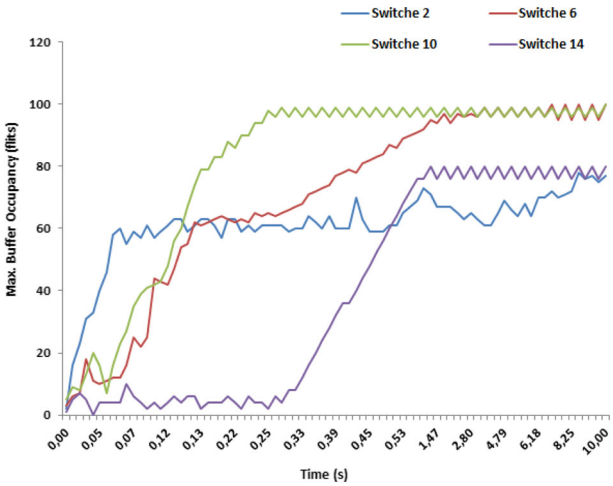

Fig. 19 With adaptation using simulation

5 Conclusions and future work

This paper introduced existing work related to the development of adaptive systems and approaches in ubiquitous and pervasive environments. It highlighted the usefulness of natural and biological systems principles, together with the bottom-up design rules, for designing adaptive algorithms and mechanisms. The illustrative examples provide some insight on how to design local and appropriate methods that allow system components to select the best suitable strategy/action and drive the system to provide the best suitable behavior according to the current system state and environment changes. However, developing models to evaluate self-* mechanisms requires knowledge insight in existing feedback control systems and system dynamics methods. Especially, insight in mechanisms based on the principle of feedback

control and in designing local adaptation rules and mathematical models to evaluate these mechanisms.

Furthermore, analyzing and discovering new emerging behaviors and/or unexpected abnormal behaviors, as well as new opportunities of services emergence, is a challenging issue in designing ubiquitous and pervasive systems. In other words, incorporating local and adaptive rules allows the system's entities to interact and perform actions favoring the emergence of a global behavior that might affect the integrated system. Managing and controlling the whole system behavior still difficult. Methods and tools for formally specifying, verifying, and validating foundational properties of these systems are required [22]. Run-time verification is a relatively new direction of verification, defined in [43] as "the discipline of computer science that deals with the study, development, and application of those verification techniques that allow checking whether a run of a system under scrutiny satisfies or violates a given correctness property" [7]. Currently, we are investigating run-time verification as a verification technique for predicting and controlling emerged behaviors that depend heavily on the environment and operational conditions. This will allow the system to keep evolving by redesigning its structure and its behavior to best suit its context and self-configure to suit users needs.

References

1. Akay B, Karaboga D (2012) A modified artificial bee colony algorithm for real-parameter optimization. Inf Sci 192:120–142
2. Akyildiz IF, Brunetti F, Blazquez C (2008) Nanonetworks: a new communication paradigm. Comput Netw J Comput Telecommun Netw 52(12):2260–2279
3. Babaoglu O, Canright G, Deutsch A, Caro GAD, Ducatelle F, Gambardella LM, Ganguly N, Jelasity M, Montemanni R, Montresor A, Urnes T (2006) Design patterns from biology for distributed computing. ACM TAAS 1(1):26–66
4. Babaoglu O, Jelasity M, Montresor A, Fetzer C, Leonardi S, Moorsel AV, van Steen M (2005) The self-star vision. In: Selfstar properties in complex information systems. LNCS, vol 3460, pp 1–20
5. Bakhouya M (2005) An adaptive approach based on mobile agents and inspired by immune system for service discovery in large scale networks. PhD Thesis, Universite de Technologie de Belfort-Montbeliard, no 34, pp 1–156
6. Bakhouya M (2012) A bio-inspired architecture for autonomic network-on-chip. In: Phan C-V (ed) Autonomic networking-on-chip: bio-inspired specification, development, and verification. Part of the embedded multi-core systems (EMS). Taylor and Francis/CRC Press, Florida, pp 1–19
7. Bakhouya M, Campbell R, Coronato A, De Pietro G, Ranganathan A (2012) Introduction to special section on formal methods in pervasive computing. ACM Trans Auton Adapt Syst 7(1). doi:10.1145/2168260.2168266
8. Bakhouya M, Chariete A, Gaber J, Wack M, Niar S, Coatanea E (2012) Performance evaluation of a flow control algorithm for network-on-chip. In: HPCS proceeding, pp 281–287
9. Bakhouya M, Gaber J (2006) Adaptive approach for the regulation of a mobile agent population in a distributed network. In: 5th International symposium on parallel and distributed computing, pp 360–366
10. Bakhouya M, Gaber J (2008) Approaches for ubiquitous computing. In: Labiod H (ed) Wireless ad hoc and sensor networks. ISTE Publishing Knowledge/Wiley, New York, pp 111–142. ISBN:978190520986
11. Bakhouya M, Gaber J (2010) A query routing approach based on users' satisfaction for resource discovery in service-oriented networks. World Wide Web 13(1–2):61–73
12. Bakhouya M, Gaber J, Lorenz P (2011) An adaptive approach for information dissemination in vehicular ad hoc networks. J Netw Comput Appl 34(6):1971–1978
13. Bakhouya M, Nemiche M, Gaber J (2015) An adaptive regulation approach of mobile agent population size in distributed systems. Int J Intell Syst. doi:10.1002/int.21750
14. Benini L, Micheli GD (2002) Networks on chips: a new SoC paradigm. IEEE Comput 35(1):70–78
15. Bonabeau E, Dorigo M, Theraulaz G (1999) Swarm intelligence: from natural to artificial systems. Oxford University Press, New York. ISBN:0-19-513159-2
16. Borshchev A, Filippov A (2004) From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools. In: The 22nd international conference of the system dynamics society, pp 1–23
17. Brun Y, Serugendo GDM, Gacek C, Giese H, Kienle H, Litoiu M, Müller H, Pezzè M, Shaw M (2009) Engineering self-adaptive systems through feedback loops. In: Software engineering for self-adaptive systems. LNCS hot topics, vol 5525, pp 48–70
18. Caro GD, Dorigo M (1998) AntNet: distributed stigmergetic control for communications networks. J Artif Intell Res 9:317–365
19. Caulfield CW, Maj SP (2001) A case for systems thinking and system dynamics. In: IEEE international conference on systems, man and cybernetics, pp 2793–2798
20. Cheng BHC, de Lemos R, Giese H, Inverardi P, Magee J (2009) Software engineering for self-adaptive systems: a research roadmap. LNCS 5525:1–26
21. Cohen IR (2007) Real and artificial immune systems: computing the state of the body. Nat Rev Immunol 7:569–574
22. Coronato A, De Florio V, Bakhouya M, Di Marzo Serugendo G (2012) Formal modeling of socio-technical collective adaptive systems. In: ASENSIS 2012, SASO conference, Lyon, France
23. De Florio V (2014) Antifragility = elasticity + resilience + machine learning: models and algorithms for open system fidelity, pp 1–7. arXiv:1401.4862
24. De Florio V, Primiero G (2015) A framework for trustworthiness assessment based on fidelity in cyber and physical domains. Procedia Comput Sci 52:996–1003
25. Detrain C, Deneubourg JL (2006) Self-organized structures in a superorganism: do ants behave like molecules. Phys Life Rev 3(3):162–187
26. Estrin D, Govindan R, Heidemann J, Kumar S (1999) Next century challenges: scalable coordination in sensor networks. In: Proceedings of ACM/IEEE MobiCom, pp 263–270
27. Farmer JD, Packard NH, Perelson A (1986) The immune system, adaptation and machine learning. Physica 22D:72–81
28. Gaber J (2006) New paradigms for ubiquitous and pervasive applications. In: First workshop on software engineering challenges for ubiquitous computing, Lancaster, UK
29. Gaber J (2007) Spontaneous emergence model for pervasive environments. In: Globecom workshops, pp 1–6
30. Gaber J (2011) Action selection algorithms for autonomous system in pervasive environment: a computational approach. TAAS 6(1):10
31. Gaber J, Bakhouya M (2006) An affinity-driven clustering approach for service discovery and composition for pervasive computing. In: Proceedings of IEEE ICPS'06, pp 277–280

32. Gordon GWT, Bently PJ (2002) On evolvable hardware. In: Ovaska S, Sztandera L (eds) Soft computing in industrial electronics, pp 279–323
33. Guffens V, Bastin G, Mounier H (2003) Fluid flow network modeling for hop-by-hop feedback control design and analysis. In: Proceedings Internetworking
34. Heinzemann C (2012) Modeling behavior of self-adaptive systems. http://ebookbrowse.com/modeling-behavior-of-self-adaptive-systems5-ppt-d217019628
35. Ishiguro A, Watanabe Y, Kondo T, Uchikawa Y (1996) Proposal of decentralized consensus-making mechanisms based on immune system-application to a behavior arbitration of an autonomous mobile robot. In: AProc of AROB, pp 122–127
36. Jerne NK (1974) Towards a network theory of the immune system. Ann Immunol (Inst Pasteur) 125C:373–389
37. Jones KH (2014) Engineering antifragile systems: a change in design philosophy. Procedia Comput Sci 32:870–875
38. Kephart JO, Chess DM (2003) The vision of autonomic computing. Computer 36(1):41–50
39. Kernbach S, Schmick T, Timmis J (2011) Collective adaptive systems: challenges beyond evolvability. arXiv:1108.5643
40. Kholodenko BN (2006) Cell-signalling dynamics in time and space. Nat Rev Mol Cell Biol 7(3):165–176
41. Krink T. Swarm intelligence-introduction. http://staff.washington.edu/paymana/swarm/
42. Lachhab F, Ouladsine R, Bakhouya M, Essaaidi M (2015) A state-feedback approach for controlling ventilation systems in energy efficient buildings. Proc IRSEC (to appear)
43. Leucker M, Schallhart C (2009) A brief account of runtime verification. J Log Algebraic Program 78(5):293–303
44. Medetov S, Bakhouya M, Gaber J, Wack M (2013) Evaluation of an energy-efficient broadcast protocol in mobile ad hoc networks. In: The 20th international conference on telecommunications
45. Medetov S, Bakhouya M, Gaber J, Zinedine K, Wack M (2014) A bee-inspired approach for information dissemination in vanets. ICMCS, pp 1–6
46. Miorandi D, Pellegrini FD, Mayora O, Giaffreda R (2010) Collective adaptive systems: scenarios, approaches and challenges. ftp://ftp.cordis.europa.eu/pub/fp7/ict/docs/fet-proactive/shapefetip-cas10en.pdf
47. Moore M, Suda T (2002) A decentralized and self-organizing discovery mechanism. In: Proceedings of the first annual symposium on autonomous intelligent networks and systems
48. Ni S, Tseng Y, Chen Y, Sheu JP (1999) The broadcast storm problem in a mobile ad hoc network. In: International conference on mobile computing and networking, pp 151–162
49. Nieh JC (2010) A negative feedback signal that is triggered by peril curbs honey bee recruitment. Curr Biol 20(4):310–315
50. Ogata K (2004) System dynamics, 4th edn. Prentice Hall. ISBN-10: 0-13-142462-9, ISBN-13: 978-0-13-142462-3
51. Piro G, Grieco L, Boggia G, Camarda P (2013) Nano-sim: simulating electromagnetic-based nanonetworks in the network simulator 3. In: SimuTools proceedings, pp 203–210
52. Suboh S, Bakhouya M, Gaber J, El-Ghazawi T (2008) An interconnection architecture for network-on-chip systems. Telecommun Syst 37(1–3):137–144
53. Suzuki J, Yamamoto Y (2000) Building an artificial immune network for decentralized policy negotiation in a communication endsystem: openwebserver/inexus study. In: Proceedings of the 4th world multiconference on systemics, cybernetics and informatics
54. Taleb NN (2012) Antifragile: things that gain from disorder. Random House Publishing Group, New York
55. Upegui A, Thoma Y, Sanchez E, Perez-Uribe A, Moreno J, Madrenas J (2007) The perplexus bio-inspired reconfigurable circuit. In: Second NASA/ESA conference on adaptive hardware and systems (AHS), pp 600–605
56. Watanabe Y, Ishiguro A, Shirai Y, Uchikawa Y (1998) Emergent construction of behavior arbitration mechanism based on the immune system. Adv Robot 12(3):227–242
57. Weiser M (1993) Some computer science issues in ubiquitous computing. Commun ACMs 36:74–84