Journal of Computer and System Sciences 77 (2011) 293–304

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Distributed redundancy and robustness in complex systems

Martin Randles [a,*], David Lamb [a], E. Odat [b], A. Taleb-Bendiab [a]

[a] *School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, United Kingdom*
[b] *King Abdullah University of Science and Technology, Saudi Arabia*

ARTICLE INFO

Article history:
Received 24 June 2009
Received in revised form 19 September 2009
Available online 1 February 2010

Keywords:
System robustness
Distributed redundancy
Algebraic connectivity
Observer system

ABSTRACT

The uptake and increasing prevalence of Web 2.0 applications, promoting new large-scale and complex systems such as Cloud computing and the emerging Internet of Services/Things, requires tools and techniques to analyse and model methods to ensure the robustness of these new systems. This paper reports on assessing and improving complex system resilience using distributed redundancy, termed degeneracy in biological systems, to endow large-scale complicated computer systems with the same robustness that emerges in complex biological and natural systems. However, in order to promote an evolutionary approach, through emergent self-organisation, it is necessary to specify the systems in an 'open-ended' manner where not all states of the system are prescribed at design-time. In particular an observer system is used to select robust topologies, within system components, based on a measurement of the first non-zero Eigen value in the Laplacian spectrum of the components' network graphs; also known as the algebraic connectivity. It is shown, through experimentation on a simulation, that increasing the average algebraic connectivity across the components, in a network, leads to an increase in the variety of individual components termed distributed redundancy; the capacity for structurally distinct components to perform an identical function in a particular context. The results are applied to a specific application where active clustering of like services is used to aid load balancing in a highly distributed network. Using the described procedure is shown to improve performance and distribute redundancy.

1. Introduction

Robustness is a property that the increasingly prevalent complex, complicated, pervasive, embedded and ubiquitous computing systems, realising visions of modern computing such as cloud based services, will need to establish as their management outstrips the real time capabilities of systems' autonomous response or human reaction/comprehension. In this regard a useful definition of robustness, appropriated from the study of biological systems, is stability against external perturbations and internal variability. It is this biological robustness that is sought for these currently emerging large-scale computer systems. It has been observed and understood for some time now that, for machine resilience at least, there is a specific relationship between robustness and randomness [1], without which errors are amplified through the systems. Thus in order to mimic biological robustness in these systems it is necessary to consider the appropriate aspects that render natural systems robust and consider the best way to introduce random elements into artificial (digital eco-) systems.

[*] Corresponding author.
E-mail addresses: m.j.randles@ljmu.ac.uk (M. Randles), d.j.lamb@ljmu.ac.uk (D. Lamb), inas_odat@ieee.org (E. Odat), a.talebbendiab@ljmu.ac.uk (A. Taleb-Bendiab).

0022-0000/$ – see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcss.2010.01.008

1.1. Theme and context: Learning from biological robustness

Robustness is observed in many biological systems; it is increasingly accepted as a fundamental property of complex evolvable systems [2,3], which for instance enables the persistence of a given function in spite of external or internal perturbations. Many examples of robustness can be found in well studied natural system models, such as ant foraging, herding, flocking, schooling [4], or regulatory networks within cellular and multi-cellular individual organisms [2]. Distributed redundancy (referred to as variety in cybernetics literature or degeneracy in biological systems [5]) has been observed to be ubiquitous in these and many other natural/biological systems, contributing significantly to their resilience [6]; for instance, in genetic code, many varieties of nucleotide sequences encode the same polypeptide or there are many different ways in which communication may be achieved between animals (even within human language) [5]. Distributed redundancy, as a global system emerging property, or phenomenon, arises out of the individual components interactions and distributed connectivity. It is important to note the difference between regular redundancy and distributed redundancy; the distinction is clearly seen in the comparison between design and selection: For an engineered system, redundancy is built into the design to provide fail-safe operation, unplanned interactions are ruled out, specific functions are aligned with particular components and no adaptation is expected in response to failure. A biological system, on the other hand, has no design, it is evolutionary in nature, any part may change or mutate to contribute to a function, there is no fixed assignment of function to components and interactions become very complex. Thus redundancy, in engineered systems, simply consists of providing spare components to identically replace failed or failing components in the system. Biological systems, in contrast, adapt or make different uses of existing components to replace failed or failing system parts. For instance, in the previously given example of communication, in a biological system if communication through speech (say) becomes impossible then other system attributes may be utilised, to accomplish the same outcome, such as sign language, for example. Alternatively if communication in an engineered system through radio (say) becomes impossible then a faulty component is diagnosed and replaced with a working identical replacement. It is thus clear that far from distributed redundancy being selected by evolution, it is rather a necessary condition for effective adaptation: Distributed redundancy refers to different elements facilitating the same outcomes, whereas regular redundancy refers to the function of identical elements.

1.2. Contributions

Whilst the concepts of robustness, distributed connectivity and redundancy might be intuitive, and many classical approaches to robust design already exist, the emergence of robust structure has inspired many computational models and applications in for instance: P2P and self-organising networks management, grid resource optimisation and scheduling and swarm based service compositions [6]; yet it is not fully understood how to characterise or engineer robust structure as a general emergent feature in such computational systems.

Such models, as have been proposed, are generally not appropriate for large-scale decentralised dynamic systems. In addition, there is little engineering understanding as to how to characterise, analyse or measure robustness in these large-scale decentralised dynamic systems. As previously stated, it is thought that robustness is a feature of evolving complex and dynamic systems [2] with engineered robustness facilitating evolution and evolution favouring robust traits. Thus there are structural requirements for systems to be evolvable involving the capacity to produce more robust components. This, allied with the hierarchical modular nature of the structures, suggests a nested bow tie or hour glass structure [7] may best capture the dynamics, where various input and output modules are connected through a conserved core with extensive system control; particularly as this architecture has emerged as an underlying feature of the World Wide Web [8].

The heterogeneous nature of the environments and participants, the specialised computational powers required to drive the processes within the components, whilst handling the vast amounts of resultant data, and the need for extensive communication between components, to facilitate robustness through emergent organisation, means that the modelling environment is required to capture both the dynamically changing nature of the systems at all its levels (from global down to local) and the static aspect of the data set at any discrete time point.

It is proposed to investigate, in this paper, how to move towards bringing aspects of biological robustness (such as distributed redundancy) to computer systems: A technique to increase robustness by demonstrating a measurable increase in distributed redundancy is discussed that uses system rewiring to promote increased connectivity.

An enhanced network-rewiring algorithm mediated by algebraic connectivity analysis is developed, which provides a measured increase in the identified robustness metric; distributed redundancy. This is used, for instance, to steer a given network rewiring (generation) process towards robust topologies/configurations. The dynamic nature of current systems means that network rewiring is a frequent occurrence in these systems. For instance load balancing on complex network is often performed based on the creation and deletion of network connections to optimise the placement of work over the network. In particular *active clustering* is a recently investigated technique whereby like services are rewired together to provide an easy distribution of the load on heterogeneous systems. This *active clustering* load-balancing procedure [9] is used in this paper to test the application of the discussed techniques, whereby clustering is mediated by algebraic connectivity: The clustering only proceeds if increased algebraic connectivity (shown in this paper to map to increased distributed redundancy) of the network is observed.

1.3. Structure of paper

The paper is organised as follows: Section 2 describes the system model, used throughout this paper, as a large-scale network graph. Section 3 defines the measurement of distributed redundancy, whilst Section 4 gives description of the techniques used in this paper to engender an increase in distributed redundancy; describing the observer system and details of the methods utilised by the inherent reasoning system to improve the system, the algebraic connectivity measure, the propagation of robust behaviour and the measurement of distributed redundancy. The proposed method is experimentally detailed formally in Section 5: The results are assessed with a simulation showing the increased distributed redundancy induced by an increasing algebraic connectivity at the component level. Section 6 applies this technique to a load-balancing solution and shows that unless robustness is specifically built into the system, through the proposed method, then performance will suffer, as the network will be unable to adapt in an agile way to new circumstances. Section 7 discusses the results and other work in this area, while Section 8 concludes the paper and looks at the future work remaining.

2. The system model

The original motivation to investigate the provision of distributed redundancy came about from the authors' work on the analysis of communities that emerge in large scale Service-Oriented Architectures (SOA) [10] or the Internet of Services (IoS) [11] following the application of self-organising algorithms; see [12] for an example analysing the communities that arise as a result of an application of load balancing based on beehive dynamics. Here the resource and application layers of the system interacted with many cross-layer dynamics engendering the emergence of communities. It is desirable to make these underlying vital communities as robust as possible in supplying the major functionality to the system. This paper addresses the problem of bringing the robustness of self-organising biological systems to computational systems by proposing a distributed Observer System to bridge the gap between network layers; including micro to *meso* to global level outcomes: The *meso*-level is conceived of as an intermediate perspective giving an accurate abstraction of component behaviour whilst maintaining a global view of emergent features (a fuller description is available in [13]). The specification for the observers' deliberation is based in a particular form of mathematical logic that permits counterfactual reasoning giving efficient modelling strategies for handling the systems' openness.

For the purposes of this paper the system consists of a set of nodes spread across a geographical area: This set of nodes may be modelled by a network graph $G = \langle V, E\rangle$ where V is the set of network nodes and E is the set of connections between the members of V. No geographical knowledge is used in this paper and it is not assumed that the nodes necessarily know their position. No initial assumptions are made regarding the networks connectivity; although for the application, scale-free connectivity is required to permit equivalent testing. Each node is uniquely identified for message sending and holds a local view of its environment: It stores a list of the node identifiers for the nodes it is connected to, in its *neighbour-list*. A node can communicate directly with only its neighbours. The system is also assumed to be dynamic with nodes entering and leaving the system at any time causing remaining nodes to update their *neighbour-lists* accordingly. The observer nodes are separated from the system itself and monitor the target system, so that each observer is in the *neighbour-list* of its target nodes and every other observer at the same system level. Thus the observer nodes, in their entirety, possess an abstract (*meso*-level) model of the components' behaviour through the formation of a Connected Dominating Set (CDS) [14] over the network.

Now, as previously discussed, this system needs to be considered as having distributed redundancy. It is suggested that, rather than components being specifically engineered for distinct functions with little interaction, future systems, with the benefits such as nanotechnology, smaller chip sizes and expansive memory, will rely on an evolutionary approach sponsored by distributed redundancy. These systems will be selective, rather than fully pre-programmed, which is a significant requirement to handle unpredictable situations in which programmed planning is impossible and novelty detection becomes very important. As it is thought that distributed redundancy in the previously mentioned ways permits the evolutionary selective approach, it is first necessary to formally consider the measurement of distributed redundancy and state the metrics required for the observer system.

3. Measuring distributed redundancy

As discussed so far, distributed redundancy ought to quantify the property of a system to have distinct, heterogeneous components capable of performing the same function in the system. Thus component output needs to be considered together with the mutuality of the components: In [15] a measure of distributed redundancy is proposed based on mutual information and entropy. If $p(g)$ is the probability distribution for the value of some system component, g in the system G and $p(g,h)$ is the joint distribution for pairs (g,h) with another system $H (h \in H)$ then the usual definition of entropy is

$$I(G) = -\sum_{g\in G} p(g)\log p(g) \quad \text{for the single system } G \quad \text{and}$$

$$I(G,H) = -\sum_{g\in G}\sum_{h\in H} p(g,h)\log p(g,h) \quad \text{for the joint system}$$

Thus, for the system G, entropy $I(G)$ is maximal when all system components contribute equal probabilities ($p(g) = 1/n$) giving $I(G) = \log n$. So that mutual information is given by:

$$MI(G, H) = I(G) + I(H) - I(G, H)$$

Now, the inputs of the systems can be considered against the outputs to measure the distinctiveness of the components with the same outputs; At any time a subset of the system units, O, will be providing the system output. Thus the distributed redundancy of the system can be interpreted as:

$$D(G) = \frac{1}{2}\sum_{k=1}^{n}\bigl\langle MI\bigl(G_i^k, O\bigr) + MI\bigl(G - G_i^k, O\bigr) - MI(G, O)\bigr\rangle \tag{1}$$

where the expression within the brackets $\langle\,\rangle$ indicates the mean taken over i and G_i^k is the ith subset of k elements which it is possible to build from the n components of the system G. Thus it can be observed that $D(G)$ tends to high values when the mutual information in the whole system ($k = n$) and the output system is high and when smaller subsets of components are contributing more to the output than larger subsets. This is fundamentally different from a normal measure of redundancy (where the components are structurally identical), which would be high if the sum of the mutual information between each component and the output system is much larger than the mutual information between the whole system and the output system.

4. Techniques for promoting increased distributed redundancy

The complexity of emerging computational systems, driven by the pervasive, heterogeneous nature and scale of the systems, renders control, maintenance and tuning extremely difficult and provides many parallels with biological system complexity. The trade-offs between robustness and complexity or centralised and decentralised control introduces extra factors seldom allowed for in any initial design-time computational system model: As presented this far, a gap remains between global emergent outcome (self-organisation) and the programming model for the component actions/interactions. This, in effect, means that it is not possible to specify all system eventualities at design-time for these systems. Thus, as noted, it is necessary to move from system design in an engineering sense to system selection in an (adaptive) evolutionary approach [3]. This paper is seeking to engender increased distributed redundancy through the introduction of engineered robustness followed by beneficial adaptations sponsored by random events at a local level. It is proposed to consider engineering robustness firstly through the optimisation of a connectivity measure (algebraic connectivity) at the previously mentioned *meso*-organisation level and secondly to promote efficient random propagation via neighbour comparison at the local level with measurable distributed redundancy emerging at the global level.

4.1. The algebraic connectivity metric

At the meso-level in a network system underlying networks of components contribute to the formation of an emergent network model. For instance a company Intranet can be considered as a single node on an Internet scale network graph. Thus increasing the connectivity of an underlying network ought to promote some beneficial effect in the containing network. In order to perform such reasoning it is necessary to portray networks in a manner that permits the extraction of suitable metrics. In this work particular matrices represent the networks: All matrices, of size $n \times n$, are assumed to be of the form

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ \cdot & \cdot & \cdot \\ a_{n1} & \dots & a_{nn} \end{pmatrix}$$

The degree matrix for the graph $G = \langle V, E\rangle$ with $V = \{v_1, \ldots, v_n\}$ and $E = \{(v_i, v_j) | v_i \text{ and } v_j \text{ are linked}\}$ is given by

$$a_{ij} = \deg(v_i) \quad \text{if } i = j \text{ and 0 otherwise}$$

The adjacency matrix is similarly given by:

$$a_{ij} = 1 \quad \text{if } v_i \text{ is adjacent to } v_j \text{ and 0 otherwise}$$

The graph Laplacian is the symmetric, zero-row-sum matrix formed by subtracting the adjacency matrix from the degree matrix:

$$a_{ij} = \bigl(\deg(v_i) \text{ if } i = j\bigr), \quad (-1 \text{ if } v_i \text{ is adjacent to } v_j) \quad \text{and} \quad (0 \text{ otherwise})$$

The spectrum of the graph G consists of the n Eigen values, $\lambda_0 \leqslant \lambda_1 \leqslant \cdots \leqslant \lambda_n$, of the Laplacian matrix, obtained by solving the characteristic equation $\det(L - \lambda I)$ for λ where det is the matrix determinant, L is the Graph Laplacian and I is the identity matrix. 0 is always an Eigen value ($\lambda_0 = 0$ for all Laplacian matrices) and $\lambda_1 > 0$ is termed the algebraic connectivity. The greater the value of λ_1 the more connected the graph is with higher clustering (it is more difficult to split the graph up into separate components). It has been shown that the algebraic connectivity is a highly relevant measure in determining network robustness [16].

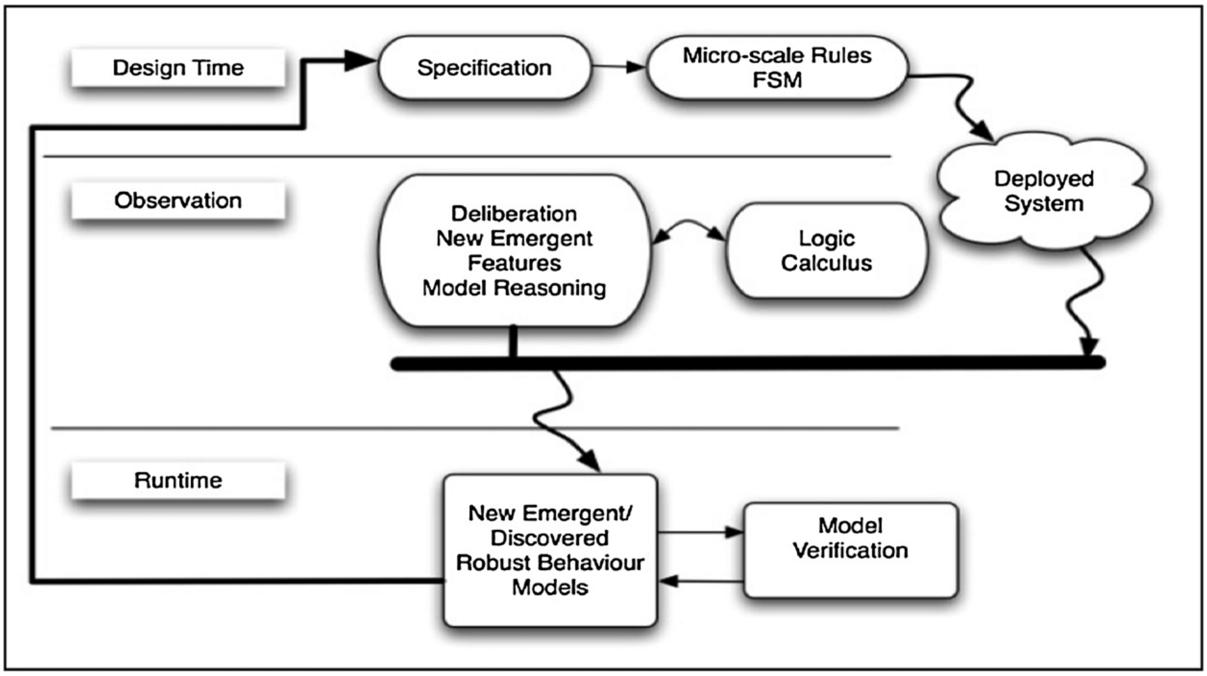

Fig. 1. Observer system process.

4.2. Neighbour comparison/propagation

Network neighbours compare their programming models as Markov Decision Problems: If a component discovers a neighbouring component performing better (based on the reward value of the MDP decided by the observer's global view) then, if it is capable, it adapts itself to the improved model. In this way the benefits of self-organisation, autonomous fault tolerance and fault recovery intervention (immunisation, etc.) are rapidly propagated across the system. It is assumed the components each control a Markov Decision Process (MDP) with the same underlying situation space, S. The component i has actions A_i and reward function R_i. The probability of a transition from situation s_1 to s_2, when action a is attempted, is denoted by: $p(s_2|s_1, a)$.

It is assumed that each component, i, implements a deterministic, stationary policy π_i, inducing a Markov Chain $p_i(s_2|s_1) = p_i(s_2|s_1, \pi_i(s_1))$. Also for all components, i and j, for each action $\pi_i(s)$ it is not necessarily the case that there exists an action $a \in A_j$ such that the distributions $p_j(.|s, a)$ and $p_i(.|s)$ are identical; thus a component can only emulate another component's model if it possesses the required functions. Thus for a component, i, gaining knowledge of another component, j's, transitions, though not necessarily the action that caused the transition, an augmented Bellman equation [17] for the value function, V, follows:

$$R_i + \gamma \max\left\{\max_{a \in A_I}\left\{\sum_{s_1 \in S} p_i(s_1|s, a)V(s_1)\right\}, \sum_{s_1 \in S} p_j(s_1|s)V(s_1)\right\}$$

where γ is a discount on future rewards. Note that situations are used instead of states. Using the logic approach of situation calculus the domain is subject to reasoning whilst the deliberation over quantified situations allows many domain instances to be considered at one time rather than addressing each individual domain instantiation. Recent work on exploiting the propositional nature of such problems [18] has brought about techniques to provide solutions with domain state sizes of over 10^{40} [19].

4.3. The observer system deliberation metrics

In large-scale complex systems there are many interacting components, driven by simple local rules and domain norms, which are mostly ignorant of high-level system goals or states. Nevertheless global states emerge from these interactions that promote robustness and drive evolution. Holland [20] identified the fundamental fact that emergent properties need to be recognisable and recurring. This, at the very least, suggests some sort of cognitive provision is necessary to handle emergent behaviour, complexity, robustness and evolution in a runtime system. Thus the design principles used here rely on the cognitive observer system [21] overlay for peer-to-peer networks or self-organising autonomic systems, shown in Fig. 1.

Briefly described this involves the specification of recognisable phenomena and signatures for emergence, the logical descriptions of component behaviours, the cognitive observer/deliberative functions, the grounding of emergent outcomes and the establishment of recurrence and recognition. This approach can be validated through simulation, whilst system evolution proceeds via runtime adaptation in response to domain sensing. In this paper early results on using this logic based Observer System to promote distributed redundancy, through self-organisation are reported upon. Specifically the benefits of employing mathematical logic, in this instance, include:

- The removal of the need to explicitly enumerate states and their transition functions.
- Behaviour is a consequence of deduction from the systems' description and a propositional account provides an abstract specification to prove properties of the system, entirely within the logic.
- Where deduction is efficient the system specification is also executable; thus rendering a simulator for the system as a side effect of the specification [22].

To handle the dynamism within the domain situation calculus is used.

4.3.1. *Situation calculus*

In situation calculus fluent values, stating what is true in the system, are initialised in the starting situation (S_0) and change from situation to situation according to effect axioms for each action. The partial solution to the resultant frame problem [22] gives successor state axioms that largely specify the system together with action precondition axioms and the initial situation. So an initial situation, S_0 is the start of the situation calculus representation. An action, a, then changes this situation from S_0 to $do(a, S_0)$ with the next action, a_1 say, changing the situation to $do(a_1, do(a, S_0))$ and so on. Thus a situation S_n is comprised of a simple action history: $a_1 a_2 \ldots a_n$. The representation of knowledge and beliefs in the situation calculus is achieved by seeing the world states as action histories or situations with the concept of accessible situations [23]. So if s_1 and s_2 are situations then $(s_1, s_2) \in K_i$ means that in situation s_2 agent i considers s_1 a possible situation with K_i an accessibility relation for agent i. That is all fluents known to hold in situation s_2 also hold in s_1. So accessibility fluents may be specified: $K_i(s_1, s_2)$ meaning in situation s_2 agent i thinks s_1 could be the actual situation. So knowledge for agent $i(knows_i)$ can be formulated in a situation as:

$$knows_i(\phi, s) \equiv \forall s_1 \big(K_i(s_1, s) \rightarrow \phi(s_1)\big)$$
$$\big[\text{alternatively } \forall s_1 \big(\neg K_i(s_1, s) \vee \phi(s_1)\big)\big]$$

This gives rise to a fluent to represent knowledge dynamics in the situation calculus. It is, thus, necessary to distinguish sensing (knowledge producing) actions by writing $SR(sense_\phi, s)$ to denote that the action produced a result for ϕ.

$$SR(sense_\phi, s) = r = \text{value of } \phi \text{ in } s$$

4.4. *Summary*

A number of the required concepts are now defined and in place to experiment with sponsoring an increase in distributed redundancy through network evolution. Firstly there is a method of measuring distributed redundancy. Secondly a component robustness metric, algebraic connectivity may be used to engineer robust components and ensure the selection of robust evolutionary traits. Thirdly the neighbour comparison procedure provides the means by which robustness is propagated throughout the system by local interaction. Finally this is managed, at the *meso*-level, by the observer system, in place to reason not only on the gap between component behaviour and global outcome but also on the algebraic connectivity of components. In this way the initially robustly engineered components are rewired, introducing randomness, with the observer system mediating to mimic selection, any resulting increased robustness is rapidly propagated through the components. It is then possible to measure the change in distributed redundancy in the whole system. The following section provides details of just such an experiment to assess the effect of this process on global system robustness, measured as distributed redundancy.

5. Experiment: The effect of algebraic connectivity on robustness (distributed redundancy)

Fig. 2 provides an overview of a deliberation process: In this example a component consists of a network of linked services, there will be a trigger for evolution, such as a service failure or network optimisation, which will enforce a rewiring, linking to another service (edge realignment of the component network graph).

This rewiring is assessed and where it improves the component it is retained. In this work the robustness measure attribute will be the algebraic connectivity of the component, whilst the effect of this on the distributed redundancy of the network containing the component will be measured.

In this section the strategy to promote distributed redundancy in a large network will be assessed. The Observer System will operate a two-stage strategy to promote and increase distributed redundancy, across and between layers, in models such as the IoS. The network will consist of components themselves comprised of networks. Firstly component networks

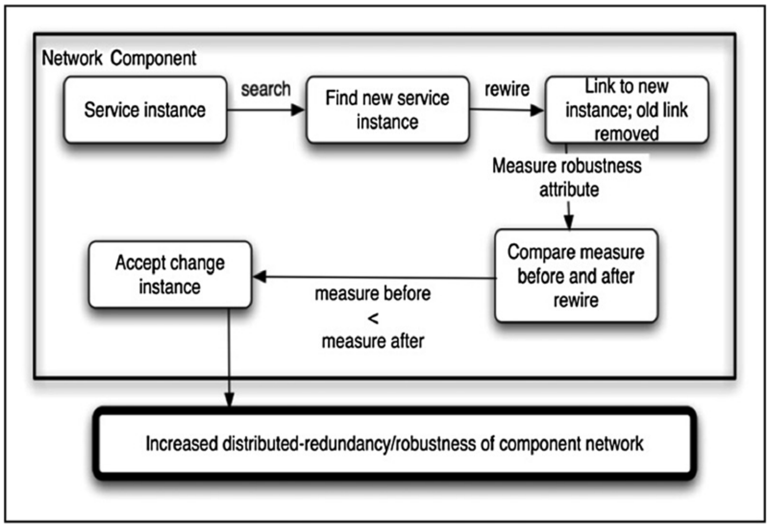
Network Component
Service instance
search
Find new service instance
rewire
Link to new instance; old link removed
Measure robustness attribute
Compare measure before and after rewire
measure before < measure after
Accept change instance
Increased distributed-redundancy/robustness of component network

Fig. 2. The deliberation process.

will propagate robust behaviour based on the value (utility) given by the Markov Decision Problem formulation: A snippet of pseudo-code to promote this propagation based only on local nodes data illustrates:

```
Initialise node, nodeID, neighbourList, \{behaviours\}, utility
......
for each neighbour in neighbour-list
\{
      if (connection(node, neighbour)==''false'') then
            neighbourList.remove(neighbour)
      endif
\}
if receive(ping) then
      newNeighbour(ping.NodeID)==''true''
      for each neighbour in neighbourList
\{
      if (neighbour=ping.NodeID)then
            newNeighbour(ping.NodeID)==''false''
      endif
\}
      if (newNeighbour(ping.NodeID)==''true'') then
            newNeighbour==ping.nodeID
            neighbourList.add(newNeighbour)
      endif
endif
for each neighbour in neighbour-list
\{
      if (utility < neighbour.utility) then
            \{behaviours\}==neighbour.\{behaviours\}
      endif
\}........
```

Secondly on detection of a fault the network reconfigures, inducing the widely studied Small-World model, as a component searches amongst the separate components for a replacement. At each potential reconfiguration the first non-zero Eigen value (algebraic connectivity) of the network graph is used as a measure of the suitability/resilience of the new network topology. The reconfiguration only proceeds if a higher value is returned. Another pseudo-code snippet for part of the observer programme illustrates:

```
Initialise Observer,nodeList,connectionlist,algebraicConnectivity
.....
for each (aNode in nodeList)
```

```
\{
      if (aNode.available!=''true'') then
            for each (bNode in nodeList)
            \{
                  if (bNode.available.likeMe=''true'') \{BREAK\}
            \}
            connectionList.add((aNode, bNode))
            oldAlgebraicConectivity==algebraicConnectivity
            calculate(algebraicConnectivity)
            if (algebraicConnectivity < oldAlgebraicConnectivity)
then
                  connectionList.remove((aNode, bNode))
            endif
      endif
\}.......
```

Fig. 3 shows a reconfiguring component highlighted within a larger network. This reconfiguring process occurring at the various layers of the system allows the creation of new components performing the same function as previously composed structurally different components; inducing distributed redundancy in the containing network. Thus, through this approach, the resilience of the component programming model, the global outcome and the topology of the network is addressed.

This simulation is encoded in Netlogo [24] with a Mathematica [25] link for Eigen decomposition. In order to simulate heterogeneity each atomic network node is randomly assigned an attribute number between 1 and 50 with the function of a node being the sum of its cooperating network nodes. Thus in order to maintain its function any node must seek a rewiring that links to the same attribute numbers in its network model.

Using an approach to signal grounding [26] within situation calculus, the action based semantics dictate the grounding of an emergent symbolic representation of the system by:

$$do\big(a, do\big(a_1, do(a, s)\big)\big) \quad \text{with} \quad SR(a, s) \neq SR\big(a, do\big(a_1, do(a, s)\big)\big)$$

where a is the sensing action for the algebraic connectivity λ_1 and a_1 is the rewiring action giving

$$knows(\lambda_1 = p, s) \wedge knows\big(\lambda_1 = q, do(a_1, s)\big) \wedge (p < q)$$

Here then the action history $do(a_1, s)$ provides a set of actions by which the emergent feature of higher topological resilience (algebraic connectivity) is observed.

To measure the distributed redundancy a simplified version of Eq. (1) is used, where the subsets of k elements are restricted to the neighbour nodes of each node. Each neighbourhood calculates its function based on the attribute number of its constituents. The level of distributed redundancy for each composed function can then be obtained and the overall level of distributed redundancy calculated as the fraction of repeated heterogeneously composed functions from the total unique functions. Thus a figure of 0 for the distributed redundancy measure means all the components are independent, whilst a figure of 1 portrays the situation where all the components perform the same function.

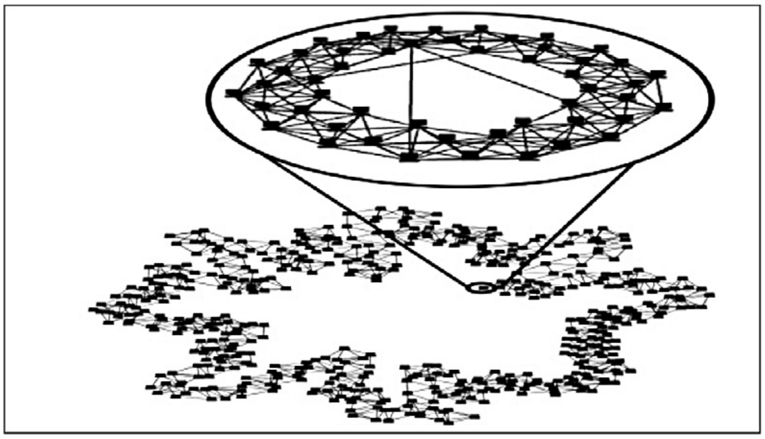

Fig. 3. Hierarchical network structure.

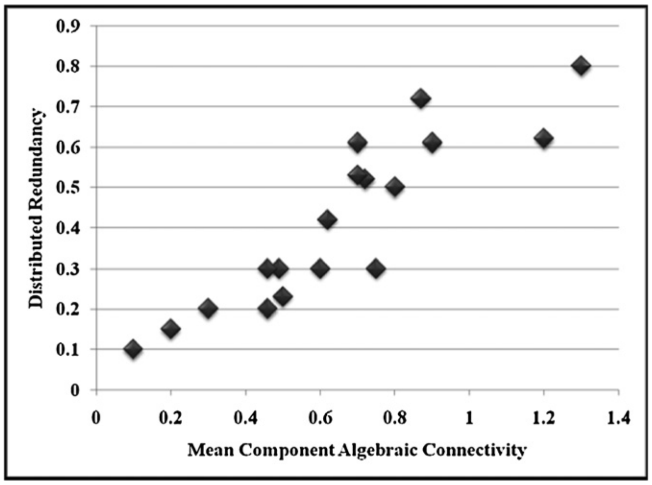

Fig. 4. Distributed redundancy increases with mean component algebraic connectivity.

5.1. Algebraic connectivity mediated rewiring and distributed redundancy

In Fig. 3 two hierarchical layers of a network system are shown with a particular node, of the 500 node, lower network highlighted to reveal its own internal network: The nodes of this network may also be composed of nested networks. The simulation proceeds by experimentally rewiring, with a probability $p = 0.1$, the components, which commenced as regular lattices of 50 nodes with a connectivity of 6: The highlighted component in Fig. 3 shows one successful application. This induces the well-known 'Small-World model' [27]. The algebraic connectivity of this node is then calculated, if the value, for the newly rewired network, exceeds the previous value then the change is preserved otherwise the rewiring is abandoned.

At various points in the network's evolution the average component algebraic connectivity is calculated together with the distributed redundancy amongst the components: Fig. 4 shows the results. Thus following identification of key communities, as occurred in [12], the major functions of a system can be preserved and made robust, in a biological sense, by an observer system's adjustment of component connectivity to engender an increase in the whole system's distributed redundancy.

6. Application to improve robustness in load balancing

As mentioned in Section 1 there is much interest in engineering robust systems through biological inspiration and self-organisation. More recently the modelling and analysis of massive systems and data sets arising from the uptake of Web 2.0 applications have started to look at network analysis techniques in this regard; taking insights gained through the study of gene regulatory and metabolic networks, as in, for example, [28]. Biological research has provided motivations for dealing with these types of systems, exhibiting the application of dynamic processes on complex networks. In [12] the authors discovered that implementation of a distributed load-balancing solution based on natural insect foraging behaviour (honeybee foraging), at the network application layer caused a particular topology to be induced at the resource layer. This manifested itself as a small number of services (servers) attracting a disproportionate amount of connectivity from cooperating services whilst most services had only a small number of links. In this way well used, vital or similar services may be grouped to deal with load balancing through the topological structure of a large-scale SOA (or Cloud).

More specifically in [8] *active clustering* is considered, as a self-aggregation algorithm, to rewire the network. Application of this procedure is intended to group like-service instances together because many load-balancing algorithms, as exemplified in [29], only work well in cases where the nodes are aware of their like nodes and can easily delegate workload to them.

Active clustering consists of iterative executions by each node in the network:

1. At a random time point the node becomes an "initiator" and selects a "matchmaker" node from its neighbours.
2. The "matchmaker" node searches for selects and causes a link to be formed between one of its neighbours that match the type of the "initiator" node and the "initiator" node.
3. The "matchmaker" removes the link between itself and its chosen neighbour.

This algorithm was studied extensively in [8], showing the organisation of a complex network towards a steady state.

Further works in [30] and [8] have refined or adapted the algorithms. The "fast" algorithm does not allow the removal of a link between like nodes, whereas the "accurate" algorithm maintains the number of links and enforces that links between non-like nodes can only be added if another link between heterogeneous nodes is removed. Full details of the complete algorithm that switches between active fast and accurate as circumstances dictate may be found in [8].

It is noted that this algorithm provides improved results for load balancing; there is, however, no consideration given to the effect on the network topology or the robustness of the rewired network. To address this issue the foregoing work, detailing the achievement of increased distributed redundancy through algebraic connectivity mediated rewiring, is utilised. This means that *active clustering* is only enacted in those cases were the robustness of the system is increased thus eradicating the possibility that the load-balancing strategy will compromise the systems resilience.

6.1. Load balancing with robustness experiment and results

In order to compare the described algorithms an experiment was established using simulations set up in Repast.NET [31]. The experiments were set up replicating the domain described in [8], to allow as direct a comparison of results as possible: A scale-free network of 100 nodes with 10% heterogeneity (i.e. 10 different types of nodes and job types) was used. The experiment was run using active clustering alone and Algebraic Connectivity Mediated Active Clustering (ACMAC): For the ACMAC algorithm, the nodes iterate as before but observer system deliberation determines whether the rewiring proceeds:

1. At a random time point the node becomes an “initiator” and selects a “matchmaker” node from its neighbours.
2. The “matchmaker” node searches for selects and causes a link to be formed between one of its neighbours that match the type of the “initiator” node and the “initiator” node.
3. The observer system calculates the change in algebraic connectivity.
4. If the change is a positive value then the “matchmaker” removes the link between itself and its chosen neighbour, else the link between the initiator and the matchmaker’s neighbour is removed.

Additionally for this experiment node/edge failure is taken into account: In a first simulation, at a certain time point 25% of the nodes (and their edges) were randomly removed from the network. In the second simulation, 25% of the edges were randomly removed from the network. The principal metric used in [8], throughput, is further assessed here under these new experimental conditions: The throughput is the number of completed jobs per elapsed time. The simulations were repeated at least 20 times each.

The results of the simulations are shown in Fig. 5, revealing better long-term outcomes when algebraic connectivity is used with active clustering.

It is shown from the graphs in Fig. 5 that the initial throughput of jobs on the network is slowed by the application of the observer’s algebraic connectivity algorithm. This is due to nodes having to wait for observer response before committing to the active clustering rewire. When failure scenarios are introduced, however, it is clear that, where active clustering has been performed only when it increases the algebraic connectivity of the network, then recovery is much better: In the case of node failure, performance is affected but recovers to near pre-failure values, whilst in the case of edge failure, the induced distributed redundancy ensures almost no discernable effect. This is in contrast to the case where active clustering is not mediated by increasing algebraic connectivity. In this case performance throughput falls and, although some recovery is made, throughput then only reaches a lower stable level. It thus seems to be clear that where active clustering proceeds only in the case of increased algebraic connectivity (previously shown to increase distributed redundancy) then the robustness of the systems operation, at least that measured by the throughput metric, is significantly improved.

7. Related work and discussion

It is generally accepted that network performance, scalability and robustness to various types of perturbations (such as random failures and targeted attacks), all depend on the network’s topology [32]. This is clearly demonstrated by the results of the previous section, where distributed redundancy was induced in the network because the job allocation for load balancing could be spread over multiple network paths, each providing a different solution to the same problem. The stated aim of the work described in this paper was to bring the robustness of biological systems to computational systems by mimicking the distributed redundancy (degeneracy) observed in natural systems. It is acknowledged, by the authors, that the most straightforward way to demonstrate this, in computational systems, is via network path redundancy: The traversal from one node to another can be completed over many different routes with high algebraic connectivity. Nevertheless this method of engendering the required distributed redundancy (robustness) is applicable to most computing domains: The ubiquity of network analysis in computing is well established [32]. This has further implications for improving on biological systems through the currently emerging study of synthetic biology [33]. In the field of computation, network structures (or graphs) offer valuable techniques for modelling and investigating the relationships between computational entities (objects) [34]. In other words instead of performing the difficult calculation of distributed redundancy, when an efficient mapping exists to a network structure, more tractable and widely understood values such as edge and node betweenness can be used instead; more typically used to determine the related issue of community structure [35].

In [36] natural connectivity is considered, characterising the redundancy of alternative paths between nodes by measuring the weighted number of closed walks of all lengths on the network. This measure uses the average Eigen value of the graph Laplacian and may provide another means of assessing robustness; albeit one that requires the full calculation of the Eigen value spectrum. For a more distributed solution it is sometimes possible to obtain the Eigen value results and optimise algebraic connectivity through edge addition [37]: Although it is not feasible to search for optimal link addition, good results

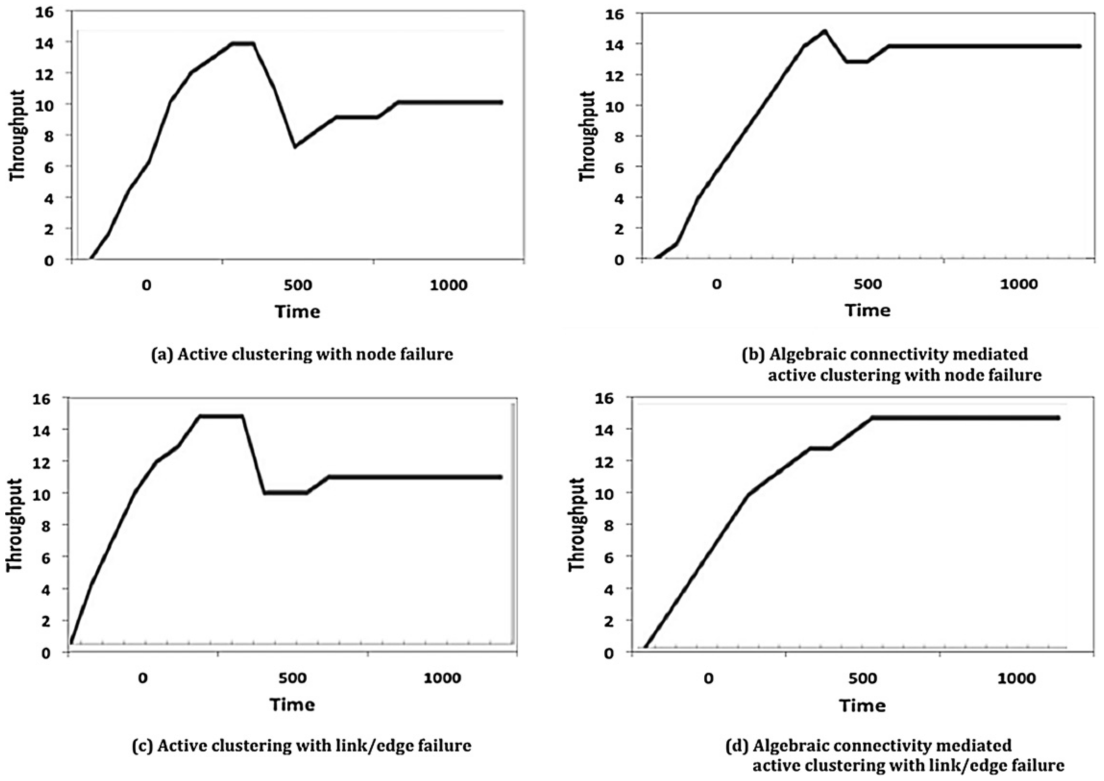

Fig. 5. Throughput for active clustering alone and algebraic connectivity mediated active clustering with: (a) and (b) 25% node failure at $t = 400$ and (c) and (d) 25% edge/link failure at $t = 400$.

can still be obtained using only local information; adding a link between a minimal degree node and another random node engenders a near maximum increase in algebraic connectivity. This method permits a more fine-grained observer system without recourse to a specific global perspective.

8. Conclusion and further works

The robust character displayed by biological and natural systems is widely attributed to their ability to evolve in response to environmental circumstances. This, in turn, is understood to arise as a result of the level of redundancy distributed over the system: There are structurally different components that can perform the same function. This paper has sought to engender such robustness through the promotion of distributed redundancy by optimising component level network structures and ensuring the rapid propagation of the best models of behaviour throughout the system. The results gained show that increased algebraic connectivity, measured as the first non-zero Eigen value of the Laplacian matrix of the network, across the components gives a measurably increased improvement in distributed redundancy.

Similar circumstances arise in market driven models of resource bargaining and acquisition between agents in multi-agent systems, for instance. Furthermore peer-to-peer networks are highly sensitive to content placement, availability, fast search and resilient network structure. In particular, in this paper, load balancing, relevant to large scale services provision or cloud computing scenarios, is considered: To achieve load balancing the network is rewired to group like services together; the rewiring only proceeds if the network is also made more robust measured through increasing algebraic connectivity, which was earlier shown to engender high distributed redundancy in the containing system. The results showed increased resilience to node and edge failure as measured by the load-balancing performance. It is envisaged that the decentralised methods of promoting robustness reported on in this paper can be adapted for application to these and many other scenarios involving large computer systems.

Much further work remains to be completed in this area. It is still not clear what range of methods may be available to promote distributed redundancy or indeed what trade-offs, in terms of increased complexity; robustness or sensitivity may become apparent. Work is still very much on-going in looking at network characterisation in general and via its Laplacian spectrum in particular. Additionally as the computational cost of Eigen value derivation is high for large networks work is progressing in seeking predictable increases in algebraic connectivity using only local actions, without the need to recalculate the spectrum.

References

[1] J. von Neumann, Probabilistic logics and the synthesis of reliable organisms from unreliable components, in: Collected Works, vol. 5, Pergamon Press, Oxford, 1963.
[2] H. Kitano, Biological robustness, Nature Review Genetics 5 (2004) 826–837.
[3] M. Kirschner, J. Gerhart, Evolvability, PNAS 95 (1998) 8420–8427.
[4] H. Parunak, Go to the ant: Engineering principles from natural multi-agent systems, Ann. Oper. Res. 75 (1997) 69–101.
[5] G.M. Edelman, J.A. Gally, Degeneracy and complexity in biological systems, PNAS 98 (24) (2001) 13763–13768.
[6] M. Mamei, R. Menezes, R. Tolksdorf, F. Zambonelli, Case studies for self-organization in computer science, J. Syst. Archit. 52 (2006) 443–460.
[7] M. Csete, J. Doyle, Bow-ties, metabolism and disease, Trends in Biotechnology 22 (2004) 446–450.
[8] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener, Graph structure in the Web, Computer Networks 33 (1–6) (2000) 309–320.
[9] E. Di Nitto, D.J. Dubois, R. Mirandola, F. Saffre, R. Tateson, Applying self-aggregation to load balancing: Experimental results, in: Proceedings of the 3rd International Conference on Bio-Inspired Models of Network, Information and Computing Systems (Bionetics 2008), 25–28 November 2008, Article 14.
[10] C. Schroth, T. Janner, Web 2.0 and SOA: Converging concepts enabling the Internet of services, IT Professional 9 (3) (2007) 36–41.
[11] R. Ruggaber, Internet of services SAP research vision, in: 16th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2007), 2007, p. 3.
[12] Martin Randles, A. Taleb-Bendiab, David Lamb, Cross layer dynamics in self-organising service oriented architectures, in: K.A. Hummel, J.P.G. Sterbenz (Eds.), IWSOS, 2008, in: Lecture Notes in Comput. Sci., vol. 5343, Springer, Berlin, 2008, pp. 293–298.
[13] Jan Sudeikat, Martin Randles, Wolfgang Renz, A. Taleb-Bendiab, A hybrid modeling approach for self-organizing systems development, Communications of SIWN 7 (May 2009) 127–134.
[14] Hamamache Kheddouci, Olivier Togni, Minimum feedback vertex sets in distance graphs and circulant graphs, Discrete Math. Theor. Comput. Sci. 10 (1) (2008) 57–70.
[15] G. Tononi, O. Sporns, G.M. Edelman, Measures of degeneracy and redundancy in biological networks, PNAS 96 (1999) 3257–3262.
[16] A. Jamakovic, P. Van Mieghem, On the robustness of complex networks by using the algebraic connectivity, in: A. Das, et al. (Eds.), Ad Hoc Sensor Networks, Wireless Networks, Next Generation Internet (NETWORKING, 2008), in: Lecture Notes in Comput. Sci., vol. 4982, Springer, Berlin, 2008, pp. 183–194.
[17] R. Price, C. Boutilier, Imitation and reinforcement learning in agents with heterogeneous actions, in: Proceedings of the AISB'00 Symposium on Starting from Society – The Application of Social Analogies to Computational Systems, Birmingham, UK, 2000, pp. 85–92.
[18] C. Boutilier, R. Reiter, R. Price, Symbolic dynamic programming for first-order MDPs, in: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01), Seattle, 2001, pp. 690–697.
[19] C. Guestrin, D. Koller, R. Parr, S. Venkataraman, Efficient solution algorithms for factored MDPs, J. Artificial Intelligence Res. 19 (2003) 399–468.
[20] J. Holland, Emergence: From Chaos to Order, Oxford University Press, Oxford, 2000, pp. 3–7.
[21] M. Randles, H. Zhu, A. Taleb-Bendiab, A formal approach to the engineering of emergence and its recurrence, in: Proceedings of ICAC'07-EEDAS Workshop, Jacksonville, Florida, USA, 2007.
[22] R. Reiter, Knowledge in Action, MIT Press, Cambridge, MA, USA, 2001.
[23] R.C. Moore, A formal theory of knowledge and action, in: J.B. Hobbs, R.C. Moore (Eds.), Formal Theories of the Commonsense World, Ablex Publishing Corporation, Norwood, NJ, 1985, pp. 319–358.
[24] U. Wilensky, NetLogo Simulation Software Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, USA, http://ccl.northwestern.edu/netlogo.
[25] Mathematica, Version 6, Wolfram Research, Inc., Champaign, IL, USA, 2007.
[26] M. Randles, A. Taleb-Bendiab, P. Miseldine, Addressing the signal grounding problem for autonomic systems, in: Proceedings of International Conference on Autonomic and Autonomous Systems (ICAS06), Santa Clara, USA, July 19–21, 2006, p. 21.
[27] D.J. Watts, S.H. Strogatz, Collective dynamics of small-world networks, Nature 393 (1998) 440–442.
[28] K. Takemotoa, C. Oosawa, Modeling for Evolving Biological Networks with Scale-Free Connectivity, Hierarchical Modularity, and Disassortativity, Math. Biosci., vol. 208, University Press, 2007, pp. 454–468.
[29] G. Cybenko, Dynamic load balancing for distributed memory multiprocessors, Journal of Parallel and Distributed Computing 7 (2) (1989) 279–301.
[30] E. Di Nitto, D.J. Dubois, R. Mirandola, Self-aggregation algorithms for autonomic systems, in: Proceedings of the 2nd International Conference on Bio-Inspired Models of Network, Information and Computing Systems (Bionetics 2007), 10–12 December 2007, pp. 120–128.
[31] Repast Organization for Architecture and Development, http://repast.sourceforge.net, 2003, accessed 10th August 2009.
[32] M.E.J. Newman, A.-L. Barabási, D.J. Watts, The Structure and Dynamics of Networks, Princeton University Press, USA, 2009.
[33] Richard Kitney, Synthetic biology: Scope, applications and implications, The Royal Academy of Engineering, UK, 2009; http://www.raeng.org.uk/news/publications/list/reports/Synthetic_biology.pdf, accessed 21st June 2009.
[34] S. Papadopoulos, A. Skusa, A. Vakali, Y. Kompatsiaris, N. Wagner, Bridge bounding: A local approach for efficient community discovery in complex networks, arXiv:0902.0871, February 2009; http://arxiv.org/pdf/0902.0871, accessed 21st June 2009.
[35] M.E.J. Newman, Detecting community structure in networks, Eur. Phys. J. B 38 (2004) 321–330.
[36] Jun Wu, Yue-Jin Tan, Hong-Zhong Deng, Yong Li, Bin Liu, Xin Lv, Spectral measure of robustness in complex networks, arXiv:0802.2564v1 [cond-mat.stat-mech], 2008; http://arxiv.org/abs/0802.2564v1, accessed 21st June 2009.
[37] H. Wang, P. Van Mieghem, Algebraic connectivity optimization via link addition, in: Proceedings of the 3rd International Conference on Bio-inspired Models of Network, Information and Computing Systems, Hyogo, Japan, November 25–28, 2008, ICST (Institute for Computer Sciences Social-Informatics and Telecommunications Engineering), 2008.