Decentralized Multi-Agent Clustering in Scale-free Sensor Networks

Mahendra Piraveenan, Mikhail Prokopenko, Peter Wang, and Astrid Zeman

CSIRO Information and Communication Technologies Centre, Sydney, Australia*,
`Mahendra.Piraveenan@csiro.au`, `Mikhail.Prokopenko@csiro.au`,
`Peter.Wang@csiro.au`, `Astrid.Zeman@csiro.au`

1 Introduction

1.1 Multi-Agent Systems and Self-organization

Many interaction processes in complex adaptive systems occur in groups, and in order to organize knowledge, collaboration and a proper distribution of functions and tasks, there is a need to analyze, model and develop computational systems in which several autonomous units interact, adapt and work together in a common open environment, combining individual strategies into overall behavior. The approach to engineering a desired system-level behavior, adopted in this work, is based on a multi-agent system [11], in which the preferred responses emerge as a result of inter-agent interactions.

Multi-agent systems (MAS) represent a new technology to engineer complex adaptive systems. Informally, a MAS is composed of multiple interacting units (agents). Each individual agent can have individual actions, plans, and so on, while all agents work together towards a common goal. It is important to distinguish between agent characteristics and MAS properties. An agent may be described in terms of the following qualities [4, 27, 28]:

- *situatedness* – an agent can receive sensory input from its environment and can perform actions which change the environment in some way; no single agent has access to what everyone else is doing;
- *autonomy* – an agent has control over its own actions and internal state without direct external intervention, and the agents are expected to self-organize and survive on the basis of local, rather than global, information;
- *temporal continuity* – an agent is a continuously running process rather than a function with fixed inputs and outputs;

* Author list is in alphabetical order.

- *adaptability* – an agent makes decisions in accordance with various rules and modifies the rules on the basis of new information;
- *communication* – agents frequently engage in communication with users and each other;
- *multi-platform functionality* – some agents run on low-end platforms, some on high-end platforms.

Some key concepts of MAS are as follows:

- each agent has incomplete capabilities to solve the global problem addressed by the MAS;
- there is no global system control or external coordination;
- data processed by the system is decentralized;
- computation within the system is asynchronous;
- robustness – the system is able to deal with unexpected changes in the environment, and recover from its own and users' errors;
- scalability – the system can be easily extended without a major redesign of its individual parts, in other words, the effort required to extend the system does not increase exponentially with the growth in the number of agents;
- solutions obtained at the system level are not explicitly programmed, and can be interpreted as emergent behavior.

Multi-agent interactions often lead to emergent patterns in overall system behavior [for example, 28]. The emergence of system-level behavior out of agent-level interactions is a distinguishing feature of complex multi-agent systems, making them very different from other complicated multi-component systems, where multiple links among the components may achieve efficient interaction and control with fairly predictable and often pre-optimized properties. However, the concept of emergence is a matter of considerable debate [6]. In particular, emergence is an expected (but not guaranteed) property of self-organization, while the latter is typically defined as the evolution of a system into an organized form in the absence of external pressures. For example, [5] described self-organization as:

> "a set of dynamical mechanisms whereby structures appear at the global level of a system from interactions among its lower-level components. The rules specifying the interactions among the systems constituent units are executed on the basis of purely local information, without reference to the global pattern, which is an emergent property of the system rather than a property imposed upon the system by an external ordering influence."

Despite the huge potential offered by self-organization, a solution based on MAS technology is warranted only when the problem domain is non-trivial, and can be characterized by the following three properties:

- *dynamism* – the problem itself changes concurrently with the problem-solving processes, forcing the latter to adapt;
- *decentralization* – the system's computational architecture is spatially distributed;
- *computational complexity* – the dimension of the full solution search space is exponential in the dimension of the problem representation, for example, optimization problems such as the Travelling Salesman Problem [8] and the Minimum Energy Broadcast Problem [45].

Informally, the problem has 'depth' in three dimensions: time, space, and computation. In the absence of at least one such requirement, it is quite likely that a more conventional approach would be more appropriate. For instance, if a problem is NP-hard and spatially distributed, but static, then it might make sense to establish a predefined hierarchy of problem-solvers that process and channel data to a single point where the global solution is integrated. If a problem is NP-hard and changes in time, but is spatially localized, then again a powerful incremental problem-solver located in a single place is the preferred choice. Finally, if a problem can be solved in polynomial time, but is dynamic and spatially distributed, then a dynamic hierarchy of problem-solvers may be considered. A study of typical trade-offs is described in [34].

1.2 Multi-Agent Networks

A well-known instance of multi-agent systems is a multi-agent network, in particular, a sensor network. A sensor network interconnects (often, wirelessly) multiple spatially distributed autonomous devices (nodes or agents), each capable of sensing, computation and communication, requiring limited memory and power. Typically, a multi-agent network is decentralised. The following summary proposed by [9], captures this requirement with three constraints:

- there is no single central information fusion or coordination centre; no node should be central to the successful operation of the network;
- there is no common communication facility; nodes cannot broadcast results and communication must be kept on a strictly node-to-node basis (although a broadcast medium is often a good model of real communication networks);
- sensor nodes do not have any global knowledge of the sensor network topology; nodes should only know about connections in their own neighbourhood.

The last constraint distinguishes between decentralized systems where each agent still has global information about the group and decentralized systems where an agent has access only to local information from a small subset [22].

Sensor networks may be utilized in various tasks, for instance, monitoring physical or environmental conditions at different locations, search, surveillance, target tracking, mapping and exploration [22]. When networked nodes are controllable and/or have actuators, one may call such network an active sensor network [21]. Typically, control complexity of large multi-agent networks grows rapidly with the number of agents [37], as well as the number of simultaneous and spatiotemporally distributed real-time events. Thus, self-organizing networks are ideally suited to implementing large sensor networks, being both robust to failures of individual nodes and scalable in terms of the number of detectable events, network size, and so on. Since the overall network behavior is spread over multiple reconfigurable communication paths and interactions, an incremental loss of a portion of the multi-agent network will lead to an incremental loss in quality, rather than a catastrophic failure.

In general, a self-organizing multi-agent network is expected to be:

- *sentient* – relying on perception through sensing (but not necessarily conscious);
- *active* – interrogating/probing the environment, and self-inspecting both locally and globally [29, 35];
- *reconfigurable* – reacting in real time, robust to external and internal fluctuations [32], and adapting to significant change through updating sensor layouts, communication protocols, and power consumption modes;
- *coordinated* – behaving coherently as a dynamical system [36]; fusing the data of individual agents into a joint shared model [9, 30];
- *symbiotic* – recognizing and forming relationships of mutual benefit or dependence among various types of agents (for example, nodes in a sensor network monitoring environment may assist in navigation of multi-robot teams, while being powered by the robots when required) [13].

These desiderata call for an efficient network structure or topology. It is obvious that a fixed topology is not likely to meet such objectives as re-configurability, symbiosis, and the like. Instead, one may consider self-organizing formation of links between nodes, leading to adaptive topologies.

1.3 Adaptive Topologies and Dynamic Hierarchies

Dynamic creation and maintenance of optimal topologies in large dynamic networks is a well-recognized challenge. It appears in many different contexts, for example, as dynamic hierarchies in Artificial Life [32, 38], coalition formation in Agent-based Systems [40], decentralized clustering in Multi-Agent Systems [24], dynamic cluster formation in Mobile Ad Hoc Networks [20], decentralized sensor arrays [25, 26, 31], reconfigurable sensor networks [12, 33], and similar. In this Chapter, we consider a sub-problem from this class: dynamic cluster formation in a sensor and communication network without centralized controllers.

There is a distinction between sensor networks and sensor grids, as pointed out in the recent literature, for instance:

> "Whereas the design of a sensor network addresses the logical and physical connectivity of the sensors, the focus of constructing a sensor grid is on the issues relating to the data management, computation management, information management and knowledge discovery management associated with the sensors and the data they generate." [14]

Dynamic sensor-data clustering is a significant issue addressed by sensor grids. The clustering process is aimed at grouping entities with similar characteristics together so that main trends or unusual patterns may be discovered. In the absence of centralized controllers, this process can be described as *self-organization* of dynamic hierarchies, with multiple cluster-heads emerging as a result of inter-agent communications.

Decentralized clustering algorithms deployed in multi-agent networks are hard to evaluate precisely for the reason of the diminished predictability brought about by self-organization. In particular, it is hard to predict when the cluster formation will converge to a stable configuration. The results presented by [31] identified a predictor for the convergence time of dynamic cluster formation in a specific topology (a rectilinear grid), based on the traffic volume of asynchronous inter-agent communications. The work presented here is an extension of the method to scale-free (sensor) grids/networks. In a scale-free network, some nodes are highly connected in comparison to the rest of the nodes in the network. Properties of scale-free networks have been extensively studied in recent times, since a lot of real world networks seem to fall into this category [1, 10, 17, 46, 47]; we shall briefly introduce this important class of networks in the next Section.

The simple predictor mentioned above is implemented at two levels:

- *the global level*, where full information on nodes' states and their interconnections is available, and
- *the local level*, where only partial information is obtained within a small selected subset of nodes.

Quantitative measures of multi-agent dynamics can be used as feedback for evolving agent behaviors [36]. Such measures can use either full information on agent states and their inter-connections, or work with partial information, obtained locally: *localisable measures* [34]. Of course localisable measures can be embedded in the agents themselves and be accessible to selected nodes (for example, hubs), controlling agent behaviors during run-time via adaptive feedback. In general, however, the communication role of a hub should not be confused with its possible control role – in some applications, the only information available at the hub is the number of transiting messages and not their content, and the main decision expected from the hub is a decision on whether to interrupt current multi-agent dynamics without knowing specific details of the exchanged messages.

Our immediate goal is predicting when the cluster formation will converge to a stable configuration. In achieving this goal, we consider an underlying time series, the traffic volume of inter-agent communications, and relate its irregularity during an initial interval to the eventual convergence time. Clearly, the shorter the initial interval, the more efficient is the prediction: for instance, when a predicted value exceeds a threshold, agents may adjust parameters and heuristics used in the clustering process.

A simplified version of a decentralized adaptive clustering algorithm operating within a scale-free network, developed for evaluation purposes, is described in the next Section. The proposed predictor for the convergence time of cluster formation is then described, followed by a discussion of the obtained results.

2 Dynamic Cluster Formation Algorithm

The dynamic cluster formation algorithm has been implemented in a scale-free topology. In a scale-free topology, it is not uncommon to find nodes with a degree (the number of connections from a node) that is much higher than the average degree of that network. These highly connected nodes are called *hubs* and can play specific roles in their networks, depending on the network domain. Hubs are often formed by a growth model that shows preferential attachment. That is, when a new node is attached to the network, it is more likely to be attached to a node with a higher degree. A general model of this type of growth is that the probability of a new node being attached to a particular existing node is proportional to the number of connections from that existing node [3]. According to this model, starting from m_0 vertices (that are typically fully connected), at each iteration a new vertex with $m \leq m_0$ edges is attached to old vertices in such a way that the probability of being connected to the existing vertex i is proportional to the degree k_i, and is set to $\frac{k_i}{\sum k_i}$, where the sum is computed over all nodes. If the parameter $m = 1$, then the growth results in a scale-free *tree graph*; otherwise, if $m > 1$, then a scale-free *network* is produced.

A degree distribution (sometimes called *vertex degree distribution*) is the probability distribution function describing the total number of vertices in a network with a given degree. The degree distribution of a scale-free network follows a power law, in the form of $p(x) \approx k^{-\gamma}$, where k is the degree. The power index γ is usually between 2.1 and 3.0 for most biological, social and technological networks [42]. Some scale-free networks may have rapid cut-offs after a certain degree, so that the degree distribution takes the form of $p(x) \approx k^{-\gamma}\phi(k/\xi)$, where $\phi(k/\xi)$ is the step function [42] which introduces a cut-off at some characteristic scale ξ. When ξ is very small, $p(x) \approx k^{-\gamma}\phi(k/\xi)$ and the degree distribution is single-scaled. As ξ grows, a power law with a sharp cut-off is obtained, while scale-free nets are observed for large ξ.

Scale-free networks generally display the *small world* phenomenon, in that the average distance between any two vertices is very small, compared to a regular or randomly connected network. Scale-free networks also tend to have higher clustering coefficients. The clustering coefficient C can be defined as the probability of two nodes individually connected to a particular third node being connected to each other [23]. Formally,

$$C = 3\left(\frac{number\ of\ triangles\ in\ the\ graph}{number\ of\ connected\ triples\ of\ vertices\ in\ the\ graph}\right) \quad (1)$$

where the multiplier 3 indicates that one triangle accounts for three individual nodes that are each connected to two other nodes.

Scale-free networks are generally robust against random attacks, but highly vulnerable against targeted attacks. In other words, removal of random nodes will only slightly affect functionality of the networks, whereas removal of a hub will *drastically* affect network functionality [2]. A sensor grid node within a network communicates only with immediate neighbours: all data are processed locally, and only information relevant to other regions of the grid is communicated as a multi-hop message. A cluster-head may be dynamically selected among the set of nodes and become a local coordinator of transmissions within the cluster. The intersection of cluster-heads and hubs of the scale-free network may be empty – in other words, a cluster-head does not need to have many network connections. On the other hand, a non-hub cluster-head would generate more intense communication traffic. Clusters may adapt, that is, re-form when new data is obtained on the basis of local sensor signals. Importantly, a cluster formation algorithm should be robust to such changes, failures of individual nodes, communication losses, and the like.

As pointed out earlier, our main goal is an analysis of a representative clustering technique in a dynamic and decentralized multi-agent setting, deployed within a scale-free sensor grid, *in terms of the predictability of its convergence time*. We represent a node's sensory reading with a single aggregated value, define 'differences' between cells in terms of this value, and cluster the nodes while minimizing these 'differences'.

The algorithm input is a series of events detected at different times and locations, while the output is a set of non-overlapping clusters, each with a dedicated cluster-head (network node) and a cluster map of its followers in terms of their sensor-data and relative grid coordinates. The algorithm is described in Appendix-A, and involves a number of inter-agent messages notifying agents about their sensory data, together with changes in their relationships and actions. For example, an agent may send a *recruit* message to another agent, delegate the role of cluster-head to another agent, or declare 'independence' by initiating a new cluster.

Most of these and similar decisions are based on the clustering heuristic described by [24], and a dynamic offset range introduced by [26]. This heuristic determines if a cluster should be split in two, as well as the location of this split. Each cluster-head (initially, each agent) broadcasts its recruit message periodically, with a broadcasting period, affecting all agents with values within a particular dynamic offset of the sensor reading detected by this agent. Every recruit message contains the sensor data of all current followers of the cluster-head with their relative coordinates (a cluster map). Under certain conditions, an agent (which is not a follower in any cluster) receiving a *recruit* message becomes a follower, stops broadcasting its own *recruit* messages and sends its information to its new cluster-head indicating its relative coordinates and the sensor reading. However, there are situations when the receiving agent is already a follower in some cluster and cannot accept a recruit message by itself – a recruit disagreement. In this case, this agent forwards the received recruiting request to its present cluster-head. Every cluster-head waits for a certain period, collecting all such *forward* messages, at the end of which the clustering heuristic is invoked on the union set of present followers and all agents who forwarded their new requests [26, 31].

Firstly, all n agents in the combined list are sorted in decreasing order according to their sensor reading value x. Then, a series of all possible divisions in the ordered set of agents is generated. That is, the first ordering is a cluster with all agents in it; the second ordering has the agent with the largest value in the first cluster and all other agents in the second cluster; and so forth (the n th division has only the last n th agent in the second cluster). For each of these divisions, the quality of clustering is measured by the total squared error:

$$E_j^2 = \sum_{i-1}^{z} \sum_{x \in A_{i,j}} \parallel x - m_{i,j} \parallel^2 \tag{2}$$

where z is a number of considered clusters ($z = 2$ when only one split is considered), $A_{i,j}$ are the clusters resulting from a particular division, and $m_{i,j}$ is the mean value of the cluster $A_{i,j}$. We divide E^2 values by their maximum to get a series of normalized values. Then we approximate the second derivative of the normalized errors per division:

$$f''(E_j^2) = \frac{(E_{j+1}^2 + E_{j-1}^2 - 2E_j^2)}{h^2} \tag{3}$$

where $h = 1/n$.

If the peak of the second derivative is greater than some threshold for the division j, we split the set accordingly; otherwise, the set will remain as one cluster. When the clustering heuristic is applied, it may produce either one or two clusters as a result. If there are two clusters, the offset of each new cluster-head is modified. It is adjusted in such a way that the cluster-head of the 'smaller' agents (henceforth, references like 'larger' or 'smaller' are relative to the value x) can now reach up to, but not including, the 'smallest' agent in the cluster of 'larger' agents. Similarly, the cluster-head of 'larger' agents can now reach down to, but not including, the 'largest' agent (the cluster-head) of the cluster of 'smaller' agents. These adjusted offsets are sent to the new cluster-heads along with their cluster maps.

The cluster-head which invoked the heuristic notifies new cluster-heads about their appointment, and sends their cluster maps to them: a *cluster-information* message. There are other auxiliary messages involved in the algorithm but importantly, the cluster formation is driven by three types: *recruit*, *cluster-information*, and *forward* messages. The first two types are periodic, while the latter type depends only on the degree of disagreements among cluster-heads. On the one hand, if there are no disagreements in the clustering (for instance, if a clustering heuristic resulted in optimal splits even with incomplete data), then there is no need to forward messages. On the other hand, when cluster-heads frequently disagree on formed clusters, the forward messages are common. In short, it is precisely the number of forward messages traced in time – the traffic volume of inter-agent communications – that we hope may provide an underlying time series $\{v(t)\}$ for our prognostic analysis, as it exhibits both periodic and chaotic features.

The quality of clustering is measured by the weighted average cluster diameter [49]. The average pair-wise distance D for a cluster C with points $\{x_1, x_2, \ldots, x_m\}$ is given by

$$D = \frac{\sum_{i-1}^{m} \sum_{j-1}^{m} d(x_i, x_j)}{m(m-1)/2} \tag{4}$$

where $d(x_i, x_j)$ is the Euclidean distance between points x_i and x_j. The weighted average cluster diameter for k clusters is given by

$$\bar{D} = \frac{\sum_{i-1}^{k} m_i(m_i - 1)D_i}{\sum_{i-1}^{k} m_i(m_i - 1)} \tag{5}$$

where m_i is the number of elements in the cluster C_i with pair-wise distance D_i. This metric is known to scale well with the size of data points and number of clusters in a particular clustering. It does not, however, account for singleton clusters, while at the same time favouring small clusters.

As pointed out by [26], the algorithm does not guarantee a convergence minimizing this criterion. In fact, it may give different clusterings for the same set of agent values, depending on the relative node locations within the network. The reason is a different communication flow affecting the adjustment of the offsets. Each time the clustering heuristic is executed in an agent, its offsets are either left alone or reduced. The scope of agents involved in the clustering heuristic depends on the order of message passing, which in turn

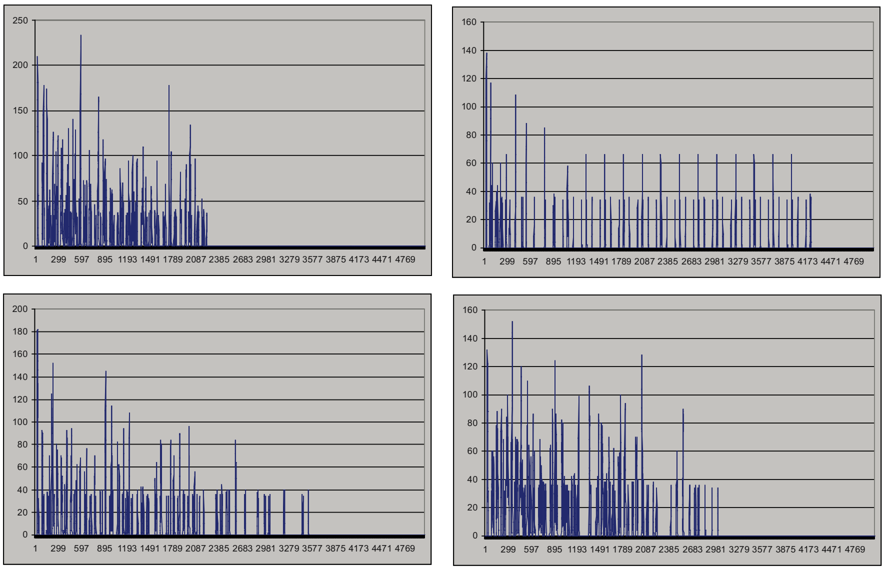
Fig. 1. Varying convergence times T for different experiments tracing the whole communication space

depends on the relative node locations. The adjusted offsets determine which agents can be reached by a cluster-head, and this will affect the result of clustering. Therefore, for any set of agent values, there are certain sequences of events which yield better clustering results than others.

We conducted extensive simulations to determine whether the algorithm is robust and scales well in terms of the quality of clustering and convergence, as measured by the number of times the clustering heuristic was invoked before stability is achieved with each data set – both for rectilinear grids [26] and scale-free networks. While the simulation results show that the algorithm converges and scales well in all cases, and in addition, is robust to dynamics of the sensor data flux, the convergence time varies significantly (Figs. 1 and 2), without obvious indicative patterns – highlighting the need for its better prediction.

3 Regularity of Multi-Agent Communication-Volume

In this Section, we focus on our main objective: prediction of the convergence time T, based on regularity of an initial segment $0, \ldots, \Omega$ of the 'communication-volume' series $\{v(t)\}$, where $\Omega < T$ and $v(t)$ is the number of forward messages at time t. The series $\{v(t)\}$ may be obtained by monitoring the whole communication space, or by monitoring the communication messages through only selected nodes. Given the role played within the scale-free

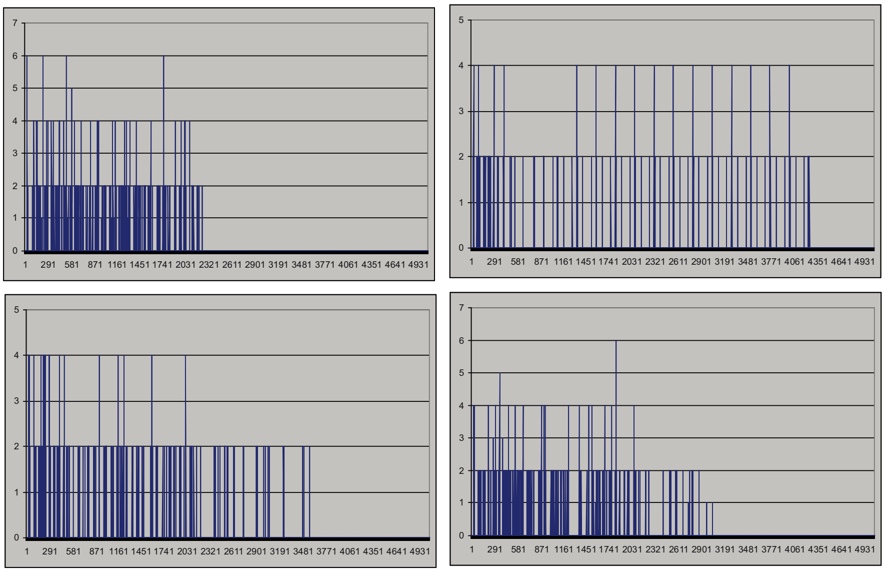

Fig. 2. Varying convergence times T for the same experiments as shown in Fig. 1, but tracing communication traffic through the highest-ranked hub only

network by hubs, we rank the nodes by the number of their connections, and monitor every node, producing multiple series $\{v_h(t)\}, 1 < h < H$, where H is the total number of nodes in the network. Thus, each series $\{v_h(t)\}$ traces the number of messages passed through the first h nodes ranked by the number of connections – for instance, $\{v_1(t)\}$ traces the number of messages passed through the hub with the most connections; $\{v_2(t)\}$ combines the number of messages passed through the first two hubs with the highest number of connections; and $\{v_H(t)\}$ is identical to $\{v(t)\}$, as it traces the messages through all the nodes in the network. The idea is then to determine whether monitoring only a subset of nodes, ideally with h being small, is almost as good as monitoring the whole network.

It is known that in many experiments, time series often exhibit irregular behavior during an initial interval before finally settling into an asymptotic state which is non-chaotic [7] – in our case, eventually converging to a fixed-point ($v(T) = 0$; henceforth, we shall drop the subscript h if it does not matter which series is being considered). The irregular initial part of the series may, nevertheless, contain valuable information: this is particularly true when the underlying dynamics are deterministic and exhibit 'transient chaos' [7, 16]. It was conjectured and empirically verified [31] that the described algorithm for dynamic cluster formation creates multi-agent transient chaotic dynamics.

[31] used the Kolmogorov-Sinai entropy K, also known as metric entropy [19, 41], and its generalization to the order-q Rényi entropy K_q [39]. The

entropy K or K_q is an entropy per unit time, or an 'entropy rate', and is a measure of the rate at which information about the state of the system is lost in the course of time. In particular, the predictor estimated the 'correlation entropy' K_2 using the algorithm of [15]. The predictor based on K_2 uses the initial segment of length Ω of the observed time series $\{v(t)\}$ in 'converting' or 'reconstructing' the dynamical information in one-dimensional data to spatial information in the τ-dimensional embedding space [43], and also depends on the length Ω and the embedding dimension τ. The method for computing the K_2 predictor is described in Appendix-B.

The predictor based on K_2 was used in predicting convergence of cluster formation within rectilinear grids. Here we apply this method to cluster formation in scale-free sensor networks.

For each experiment s, we

(a) select an initial segment of length Ω of the time series; and
(b) compute the regularity predictor: the correlation entropy $K_2(d, r, \Omega)$ for a range of embedded dimensions d and a suitable precision r (see Appendix-B for details).

Then,

(c) given the estimates $K_2(d, r, \Omega)$ for all the experiments, we correlate them with the observed convergence times T_s by linear regression $T = a + bK_2$ and the corresponding correlation coefficient $\rho(d, r, \Omega)$ between the series T_s and $K_2(d, r, \Omega)_s$;
(d) we determine the embedding dimension $\hat{d}$ and the distance $\hat{r}$ which provide the best fit: the maximum of $\rho(d, r, \Omega)$.

This allows us to predict the time T of convergence to $\nu(T) = 0$, as

$$T = a(\hat{d}, \hat{r}, \Omega) + b(\hat{d}, \hat{r}, \Omega)K_2(d, r, \Omega) \tag{6}$$

for any real-time run that produced the predictor value $K_2(d, r, \Omega)$.

4 Experimental Results

The experiments included multiple scenarios, each of which was defined by a specific scale-free tree graph, Φ (more precisely, a specific degree distribution in the corresponding graph); the number of nodes in the network, H; and the number of events sensed by network, N. For example, two different scale-free tree graphs Φ_1 and Φ_2, both with 400 nodes and 100 events, may be compared (Figs. 3 and 4). In particular, we considered different values for N (for example, N was set to 1/8, 1/4, 1/2 of H), given the same network Φ with fixed H. This approach evaluated the predictor with respect to the dynamics in the communication space brought about by multiple events. In addition,

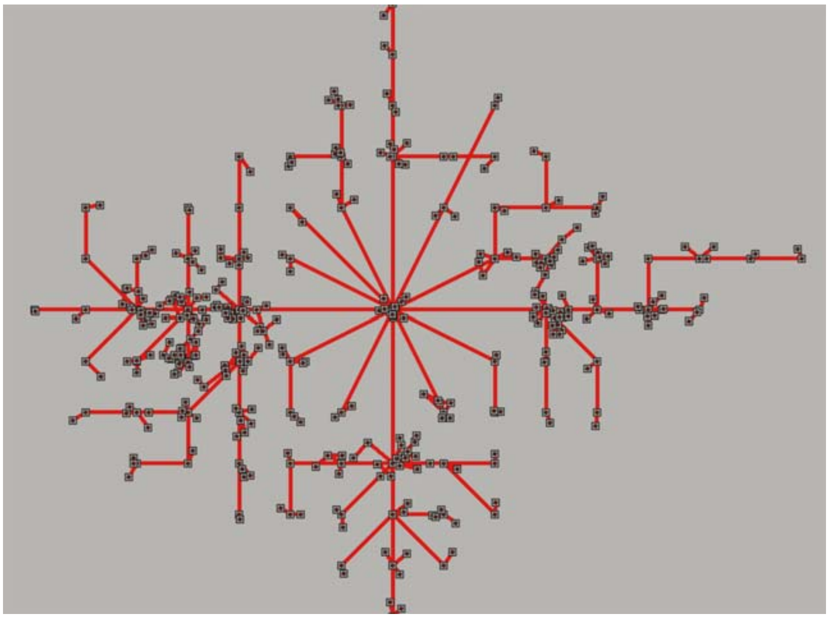

Fig. 3. A scale-free tree graph Φ_1 used in experiments

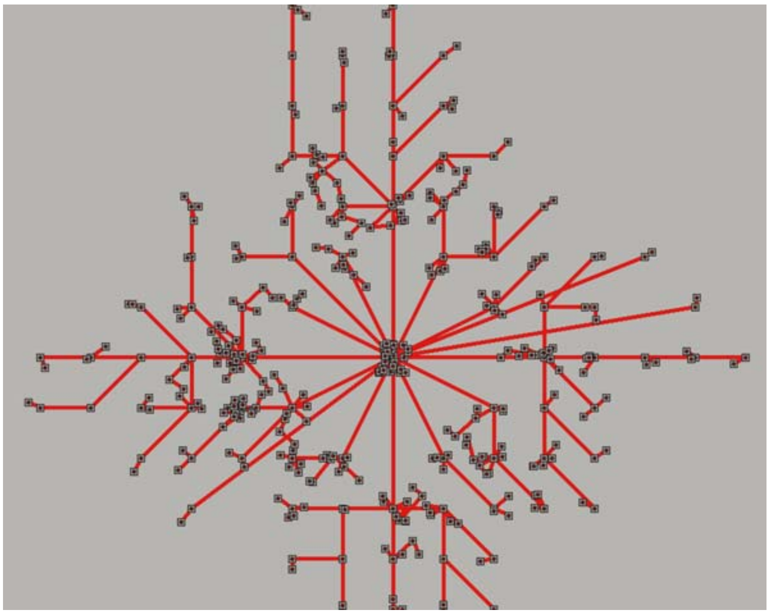

Fig. 4. A scale-free tree graph Φ_2 used in experiments

the predictor was verified by varying degree distributions Φ for a fixed number H – in other words, by considering a different network of the same size. Again, the number of events N was increased as a proportion of H. Finally, we verified the results by increasing the network size H.

In order to estimate the statistical significance of the results, each scenario (fixed Φ, H, N) included 100 runs of the clustering algorithm on a scale-free tree graph, where every run involved N events in random network locations, tracing the communication-volume time series $\{v(t)\}$, as well as multiple series $\{v_h(t)\}, 1 < h < H$. We then selected an initial segment $\Omega = 1500$ (while the longest run is 5000) and carried out the steps b), c) and d) described previously in Sect. 3. These runs produced a 2-dimensional array $K_2(d, r, \Omega)$ for varying dimensions d and precisions r and each run $s(s = 1, \ldots, 100)$. Given the array, the correlation coefficient $\rho(d, r, \Omega)$ between the actual convergence time T_s (standardized series) and the auto-correlation predictor $K_2(d, r, \Omega)$ (standardized series) was determined for the ranges of d and r. The higher the maximal correlation coefficient $\rho(d, r, \Omega)$ is, the more predictive power is contained in the predictor $K_2(d, r, \Omega)$.

The correlation coefficient $\rho(d, r, \Omega)$ is obviously decreased to $\rho_h(d, r, \Omega)$ as the series $\{v(t)\}$ is replaced with the series $\{v_h(t)\}$, if $h < H$. We observed, however, that the difference between $\rho(d, r, \Omega)$ and $\rho_h(d, r, \Omega)$ is insignificant when $h \geq 1$, for at least one embedding dimension. In other words, monitoring a single highest-ranked hub (or two highest-ranked hubs) is sufficient in order to predict the convergence of cluster formation in a scale-free tree graph. This result holds for all the considered scenarios, and supports our conjecture that a localisable predictor is feasible, although in general it is harder to maintain predictability when the number of events is large.

Figure 5 plots the correlation coefficients $\rho(d, r, \Omega)$ for a range of dimensions d and different precisions r in the scenario ($\Phi = \Phi_1, H = 400, N = 50$). The precision $r = 10$ (messages) yields the highest predictive power (shown as the plot with squares). Evaluation of the precision $r = 10$ was continued with higher dimensions d. Figures 6, 8 and 9 plot the correlation coefficients $\rho(d, r, \Omega)$ for a range of dimensions d (and the best precision r) in the scenarios ($\Phi = \Phi_1, H = 400, N = 50$), ($\Phi = \Phi_1, H = 400, N = 100$), and ($\Phi = \Phi_1, H = 400, N = 200$), respectively – based on the series $\{v(t)\}$ tracing the whole communication-space and the series $\{v_1(t)\}$ at the highest-ranked hub only. All results are statistically significant (significance levels of 0.999), and are within reasonable confidence limits (Fig. 7). The results with the alternative network $\Phi = \Phi_2$, as well as different network sizes H, are analogous, and also support the localized predictor based on the series $\{v_1(t)\}$.

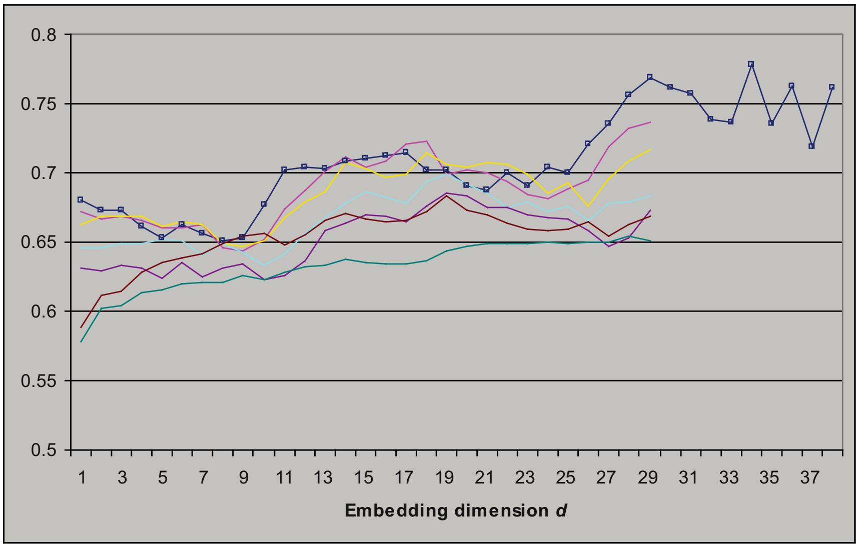

Fig. 5. Correlation coefficient $\rho(d, r, \Omega)$ between series T_s and predictor $K_2(d, r, \Omega)$, for the scenario $(\Phi = \Phi_1, H = 400, N = 50)$, based on series $\{v(t)\}$ tracing the whole communication-space, for a range of precisions r. The precision $r = 10$ (messages) yields the highest predictive power (shown as the plot with squares). Evaluation of the precision $r = 10$ was continued with higher dimensions d

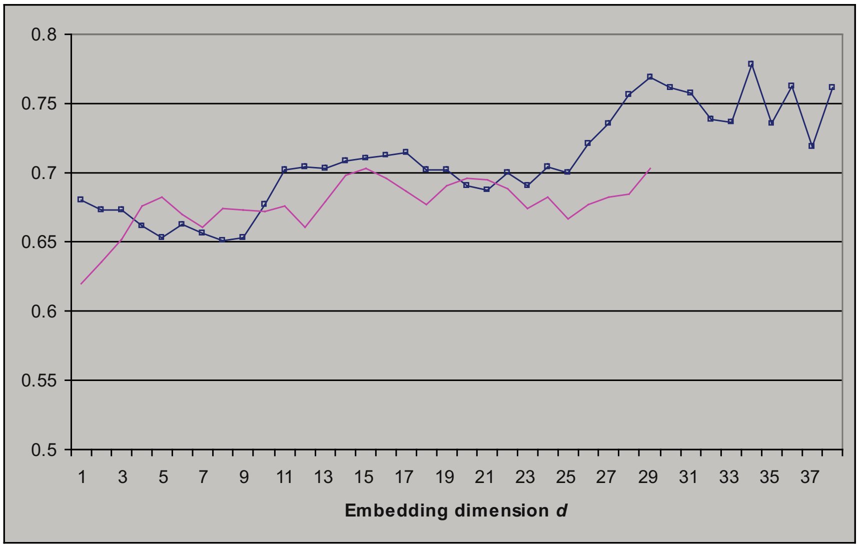

Fig. 6. Correlation coefficient $\rho(d, r, \Omega)$ between series T_s and predictor $K_2(d, r, \Omega)$, for the scenario $(\Phi = \Phi_1, H = 400, N = 50)$, based on series $\{v(t)\}$ tracing the whole communication-space, shown as the plot with squares $(r = 10)$, and series $\{v_1(t)\}$ traced at the highest-ranked hub $(r = 2)$

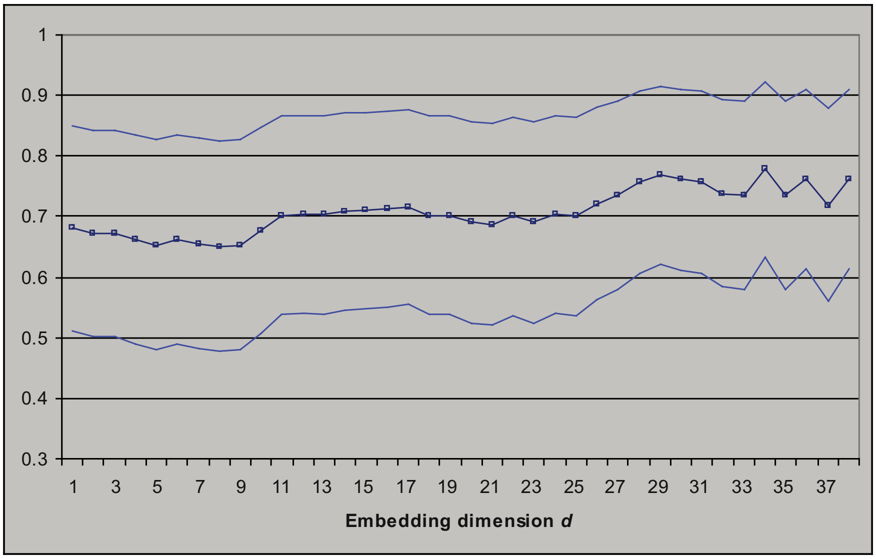

Fig. 7. Confidence limits of the correlation coefficient $\rho(d, r, \Omega)$ between the series T_s and predictor $K_2(d, r, \Omega)$, for the scenario $(\Phi = \Phi_1, H = 400, N = 50)$, based on the series $\{v(t)\}$ tracing the whole communication-space, shown as the plot with squares $(r = 10)$

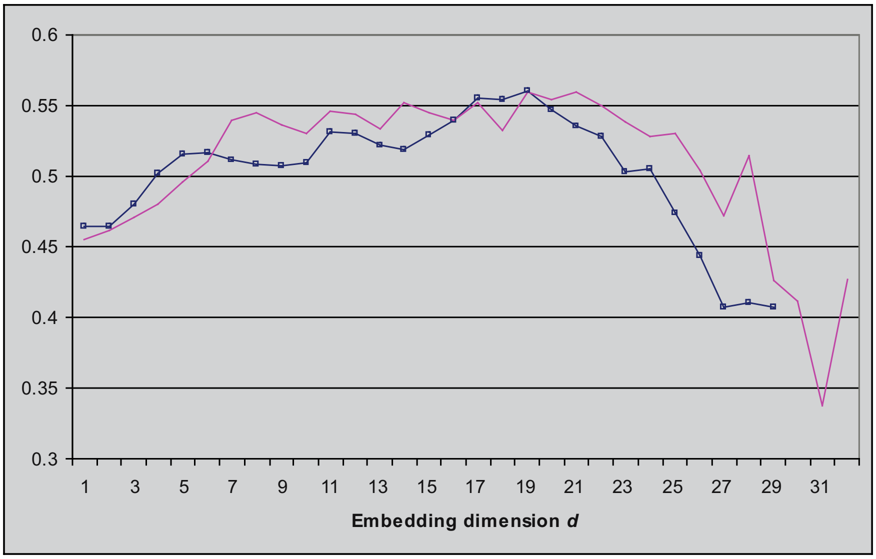

Fig. 8. Correlation coefficient $\rho(d, r, \Omega)$ between series T_s and predictor $K_2(d, r, \Omega)$, for the scenario $(\Phi = \Phi_1, H = 400, N = 100)$, based on series $\{v(t)\}$ tracing the whole communication-space (the plot with squares; precision $r = 20$), and series $\{v_1(t)\}$ traced at the highest-ranked hub $(r = 2)$

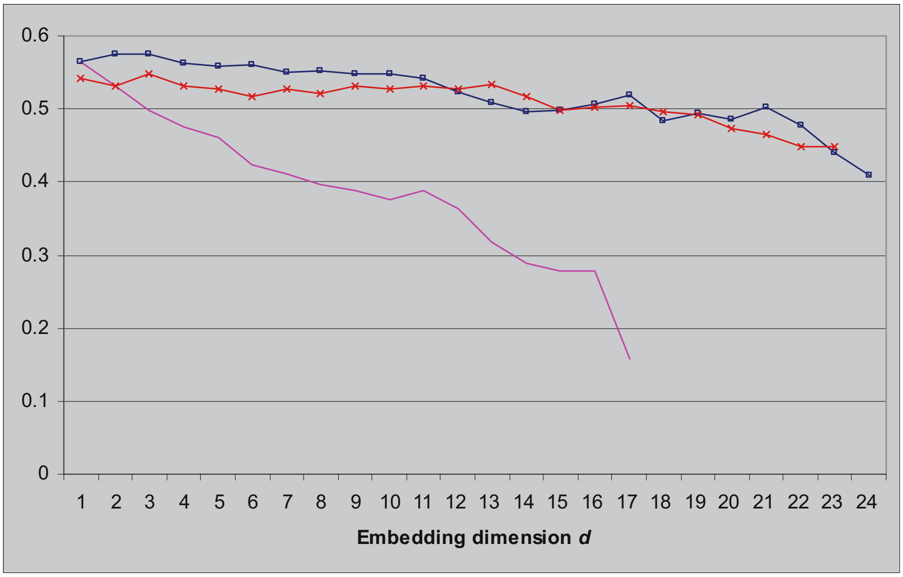

Fig. 9. Correlation coefficient $\rho(d, r, \Omega)$ between series T_s and predictor $K_2(d, r, \Omega)$, for the scenario ($\Phi = \Phi_1, H = 400, N = 200$), based on series $\{v(t)\}$ (squares: precision $r = 40$), the series $\{v_1(t)\}$ traced at the highest-ranked hub ($r = 2$), and series $\{v_2(t)\}$ traced at two highest-ranked hubs (crosses: $r = 4$)

5 An Application Scenario – Distributed Energy Management and Control

Distributed energy refers to the generation of power (for heating and cooling) within close proximity to the point of use. CSIRO is developing a range of small scale distributed energy technologies [18] based on both renewable and fossil fuels (mainly natural gas). A major part of this program is identifying the most efficient ways of integrating large numbers of small generation plants (including solar and wind) into the existing power networks to deliver maximum efficiency with minimum environmental impacts.

A decentralized approach to the problem of power load management is described [48], using modeling of direct load management as a computational market. A load, in this context, is any device that consumes electric energy, such as a water heater or an electric motor. Load management involves controlling the loads at the demand side to achieve a better use of energy: better for the utility, the customer or both. [48] define *direct load management* as a process when the utility determines what loads are to be connected, reduced, or disconnected at specific occasions, and contrast it with *indirect load management* when the utility sends some signal to customers, such as price information, and expects them to adjust to this signal. Decentralized multi-agent algorithms become usable in power load management due to the inherent parallelism of the underlying network increasing the computational power. In addition, according to [48]:

> "from the energy utility point of view it is desirable to have a system that hides most details of the different loads while still providing enough information for energy optimization. The system should be able respond to high level control commands for – for example, reduction of the current load in the distribution system by a certain amount".

The US Department of Energy's recent report to Congress on demand response [44] notes that:

> "if you take a system balancing and reliability perspective, active demand gives you another set of tools, another resource on which you can call to enable system balancing to avoid triggering capacity constraints and involuntary interruptions. Furthermore, double-sided markets have important economic efficiency benefits as well as system reliability benefits".

Typically, an efficient demand response is provided by a multi-agent coalition (a cluster of agents), comprising both generators and loads, sharing/fusing some information while trying to cooperatively solve a distributed decentralized problem. We believe that the efficient methods of decentralized dynamic clustering deployed in scale-free power grids will increase the efficiency of the overall solution.

6 Conclusions

We considered decentralized and dynamic cluster formation in scale-free multi-agent sensor grids, and described and experimentally evaluated a predictor for the convergence time of cluster formation. The new predictor estimates regularity of the inter-agent communication space via the 'correlation entropy' (the order-2 Rényi entropy) K_2, and was observed to be well correlated with the time of cluster formation.

The predictor is implemented and experimentally evaluated at the global level, where full information on nodes' states and their inter-connections is available, as well as at the local level, using only partial information obtained within a small selected subset of nodes. In other words, the predictor K_2 does not have to employ full information on nodes' states and their inter-connections – instead it may use only partial information obtained within a small selected subset of nodes. Thus, the analysis and presented results support the deployment of localisable predictors monitoring a small part of the inter-agent communication space – the part contained within the most connected hubs of the scale-free sensor network.

Efficient and reliable algorithms for cluster formation in sensor grids may include a convergence predictor, such as predictor K_2, as a feedback to the algorithms, and this is a subject for future research. Another direction is an evaluation of other decentralized algorithms in terms of multi-agent dynamics within the communication spaces.

References

1. Albert R, Jeong H, Barabási A-L (1999) Diameter of the world-wide web. *Nature*, 401: 130–131.
2. Albert R, Jeong H, Barabási A-L (2000) Error and attack tolerance of complex networks. *Nature*, 406: 378–382.
3. Albert R, Barabási A-L (2002) Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1): 47–97.
4. Butler M, Prokopenko M, Howard T (2001) Flexible synchronisation within RoboCup environment: a comparative analysis. In: Stone P, Balch TR, Kraetzschmar GK (eds) *RoboCup 2000: Robot Soccer World Cup IV (Proc. 4th RoboCup-2000 Workshop*, 31 August – 1 September, Melbourne, Australia), Lecture Notes in Computer Science 2019: 119–128, Springer-Verlag, Berlin.
5. Bonabeau E, Theraulaz G, Deneubourg J-L, Camazine S (1997) Self-organisation in social insects. *Trends in Ecology and Evolution*, 12(5): 188–193.
6. Boschetti F, Prokopenko M, Macreadie I, Grisogono A-M (2005) Defining and detecting emergence in complex networks. In: Khosla R, Howlett RJ, Jain LC (eds) *Proc. 9th Intl. Conf. Knowledge-Based Intelligent Information and Engineering Systems – KES 2005*, 14–16 September, Melbourne, Australia, Lecture Notes in Computer Science 3684: 573–580, Springer-Verlag, Berlin.
7. Dhamala M, Lai YC, Kostelich EJ (2001) Analyses of transient chaotic time series. *Physical Review E*, 64: 1–9.
8. Dorigo M, Gambardella LM (1997) Ant colonies for the Traveling Salesman Problem. *BioSystems*, 43: 73–81.
9. Durrant-Whyte, HF, Stevens M (2001) Data fusion in decentralised sensing networks. *Proc. 4th Intl. Conf. Information Fusion*, 7–10 August, Montreal, Canada, International Society of Information Fusion, Sunnyvale, CA.
10. Faloutsos M, Faloutsos P, Faloutsos C (1999) On power-law relationships of the internet topology. *Computer Communication Review*, 29: 251–262.
11. Ferber J (1999) *Multi-Agent Systems*. Addison Wesley Professional, Reading, MA.
12. Foreman M, Prokopenko M, Wang P (2003) Phase transitions in self-organising sensor networks. In: Banzhaf W, Christaller T, Dittrich P, Kim JT, Ziegler J (eds) *Advances in Artificial Life (Proc. 7th European Conf. Artificial Life – ECAL2003)*, 14–17 September, Dortmund, Germany, Lecture Notes in Artificial Intelligence 2801: 781–791, Springer-Verlag, Berlin.
13. Gerasimov V, Healy G, Prokopenko M, Wang P, Zeman A (2006) Symbiotic sensor networks in complex underwater terrains: a simulation framework. In: Gabrys B, Howlett RJ, Jain LC (eds) *Proc. 10th Intl. Knowledge-Based Intelligent Information and Engineering Systems Conf. – KES 2006*, 9–11 October,

Bournemouth, UK, Lecture Notes in Artificial Intelligence 4253(III): 315–323, Springer-Verlag, Berlin.

14. Ghanem M, Guo Y, Hassard J, Osmond M, Richards M (2004) Sensor grid for air pollution monitoring. In: Cox SJ (ed) *Proc. 3rd UK e-Science All-hands Conf. – AHM 2004*, 31 August – 3 September, Nottingham, UK, Engineering and Physical Sciences Research Council (EPSRC), UK: 106–113.
15. Grassberger P, Procaccia I (1983) Estimation of the Kolmogorov entropy from a chaotic signal. *Physical Review A*, 28(4): 2591–2593.
16. Jánosi IM, Tél T (1994) Time series analysis of transient chaos. *Physical Review E*, 49(4): 2756–2763.
17. Jeong H, Tombor B, Albert R, Oltvai ZN, Barabási, A-L (2000) The large-scale organization of metabolic networks. *Nature*, 407: 651–654.
18. Jones T, James G (2005) The management and control of distributed energy resources (extended version). In: *Proc. 18th Intl. Conf. and Exhibition on Electricity Distribution – CIRED*, June 2005, Turin, Italy, IEE, London, UK: 987–998.
19. Kolmogorov AN (1959) Entropy per unit time as a metric invariant of automorphisms (in Russian). *Doklady Akademii Nauk SSSR*, 124: 754–755.
20. Lin R, Gerla M (1997) Adaptive clustering for mobile wireless networks. *IEEE J. Selected Areas in Communications*, September: 1265–1275.
21. Makarenko A, Kaupp T, Grocholsky B, Durrant-Whyte HF (2003) Human-robot interactions in active server networks. In: *Computational Intelligence in Robotics and Automation for the New Millennium* (Proc. 2003 IEEE Intl. Symposium Computational Intelligence in Robotics and Automation), 16–20 July, Kobe, Japan, IEEE Computer Society Press, Piscataway, NJ, 1: 247–252.
22. Mathews GM, Durrant-Whyte HF, Prokopenko M (2006) Scalable decentralised decision making and optimisation in heterogeneous teams. In: *Proc. IEEE Intl. Conf. Multisensor Fusion and Integration for Intelligent Systems – MFI2006*, 3–6 September, Heidelberg, Germany, IEEE Computer Society Press, Piscataway, NJ: 383–388.
23. Newman MEJ (2002) The structure and function of networks. In: Hossfeld F, Binder E (eds) *Proc. Europhysics Conf. Computational Physics – CCP2001*, 5–8 September, 2001, Aachen, Germany, Elsevier Science, Amsterdam, 147(1–2): 40–45.
24. Ogston E, Overeinder B, Van Steen M, Brazier F (2003) A Method for decentralized clustering in large multi-agent systems. In: Rosenschein JS, Sandholm T, Wooldridge M, Yokoo M (eds) *Proc. 2nd Intl. Joint Conf. Autonomous Agents and Multi-Agent Systems*, 14–18 July, Melbourne, Australia, ACM Press, New York, NY: 798–796.
25. Olsson L, Nehaniv CL, Polani D (2004) Sensory channel grouping and structure from uninterpreted sensor data. In: Zebulum RS, Gwaltney D, Hornby G, Keymeulen D, Lohn J, Stoica A (eds) *Proc. NASA/DoD Conf. Evolvable Hardware – EH'04*, 24–26 June, Seattle, WA, IEEE Computer Society Press, Los Alamitos, CA: 153–160.
26. Piraveenan M, Prokopenko M, Wang P, Price DC (2005) Towards adaptive clustering in self-monitoring multi-agent networks. In: Khosla R, Howlett RJ, Jain LC (eds) *Proc. 9th Intl. Conf. Knowledge-Based Intelligent Information and Engineering Systems – KES'2005*, 14–16 September, Melbourne, Australia. Lecture Notes in Computer Science 3682(II), Springer-Verlag, Berlin: 796–805.

27. Prokopenko M (1999) On situated reasoning in multi-agent systems. In: *Hybrid Systems and AI: Modeling, Analysis and Control of Discrete and Continuous Systems*, AAAI Technical Report SS-99-05, March, AAAI Press, Menlo Park, CA: 158–163.
28. Prokopenko M, Butler M, Howard T (2001) On emergence of scalable tactical and strategic behavior. In: Stone P, Balch TR, Kraetzschmar GK (eds) *RoboCup 2000: Robot Soccer World Cup IV (Proc. 4th RoboCup-2000 Workshop)*, 31 August – 1 September, Melbourne, Australia, Lecture Notes in Computer Science 2019, Springer-Verlag, Berlin: 357–366.
29. Prokopenko M, Wang P (2004) On self-referential shape replication in robust aerospace vehicles. In: Pollack J, Bedau MA, Husbands P, Ikegami T, Watson RA (eds) *Artificial Life IX (Proc. 9th Intl. Conf. Simulation and Synthesis of Living Systems)*, 12–15 September, Boston, MA, MIT Press, Cambridge, MA: 27–32.
30. Prokopenko M, Wang P (2004) Evaluating team performance at the edge of chaos. In: Polani D, Browning B, Bonarini A, Yoshida K (eds) *RoboCup 2003: Robot Soccer World Cup VII (Proc. 7th RoboCup-2003 Springer-Verlag, Berlin:Symposium)*, Padua, July, Lecture Notes in Computer Science 3020: 89–101.
31. Prokopenko M, Piraveenan M, Wang P (2005) On convergence of dynamic cluster formation in multi-agent networks. In: Capcarrére MS, Freitas AA, Bentley PJ, Johnson CG, Timmis J (eds) *Advances in Artificial Life (Proc. 8th European Conference – ECAL 2005)*, 5–9 September, Canterbury, UK. Lecture Notes in Computer Science 3630, Springer-Verlag, Berlin: 884–894.
32. Prokopenko M, Wang P, Price DC, Valencia P, Foreman M, Farmer AJ (2005) Self-organising hierarchies in sensor and communication networks. *Artificial Life* (Special issue on Dynamic Hierarchies), 11(4): 407–426.
33. Prokopenko M, Wang P, Foreman M, Valencia P, Price DC, Poulton G (2005) On connectivity of reconfigurable impact networks in ageless aerospace vehicles. *J. Robotics and Autonomous Systems*, 53: 36–58.
34. Prokopenko M, Wang P, Price DC (2005) Complexity metrics for self-monitoring impact sensing networks. In: Lohn J, Gwaltney D, Hornby G, Zebulum R, Keymeulen D, Stoica A (eds) *Proc. NASA/DoD Conf. Evolvable Hardware – EH-05*, 29 June – 1 July, Washington, DC, IEEE Computer Society Press, Los Alamitos, CA: 239–246.
35. Prokopenko M, Poulton GT, Price DC, Wang P, Valencia P, Hoschke N, Farmer AJ, Hedley M, Lewis C, Scott DA (2006) Self-organising impact sensing networks in robust aerospace vehicles. In: Fulcher, J (ed) *Advances in Applied Artificial Intelligence*. Idea Group, Hershey, PA: 186–223.
36. Prokopenko M, Gerasimov V, Tanev I (2006) Evolving spatiotemporal coordination in a modular robotic system. In: Nolfi S, Baldassarre G, Calabretta R, Hallam JCT, Marocco D, Meyer J-A, Miglino O, Parisi D (eds) *From Animals to Animats 9 (Proc. 9th Intl. Conf. Simulation of Adaptive Behavior – SAB2006)*, 25–29 September, Rome, Italy, Lecture Notes in Computer Science 4095, Springer-Verlag, Berlin: 558–569.
37. Pynadath DV, Tambe M (2002) Multiagent teamwork: analyzing the optimality and complexity of key theories and models. In: Castelfranchi C, Johnson WL (eds) *Proc. 1st Intl. Joint Conf. Autonomous Agents and Multiagent Systems – AAMAS2002*, 15–19 July, Bologna, Italy, ACM Press, New York, NY: 873–880.

38. Rasmussen S, Baas NA, Mayer B, Nilsson M, Olesen MW (2001) Ansatz for dynamical hierarchies. *Artificial Life*, 7(4): 329–353.
39. Rényi, A (1970) *Probability theory.* North-Holland, Amsterdam, The Netherlands.
40. Sandholm T, Lesser V (1995) Coalition formation among bounded rational agents. In: Mellish C (ed) *Proc. 14th Intl. Joint Conf. Artificial Intelligence – IJCAI95*, 20–25 August, Montréal, Québec, Canada, Morgan Kaufmann, San Francisco, CA: 662–671.
41. Sinai YG (1959) On the concept of entropy of a dynamical system (in Russian). *Doklady Akademii Nauk SSSR*, 124: 768–771.
42. Solé RV, Ferrer-Cancho R, Montoya JM, Valverde S (2002) Selection, tinkering and emergence in complex networks – crossing the land of tinkering. *Complexity*, 8(1): 20–33.
43. Takens F (1981) Detecting strange attractors in turbulence. *Dynamical systems and Turbulence*, Lecture Notes in Mathematics 898, Springer-Verlag, Berlin: 366–381.
44. US Department of Energy (2006) Benefits of Demand Response in Electricity Markets and Recommendations for Achieving Them. *A Report to the United Stated Congress Pursuant to Section 1252 of the Energy Policy Act of 2005*, February.
45. Wieselthier JE, Nguyen GD, Ephremides A (2000) On the construction of energy-efficient broadcast and multicast trees in wireless networks. *Proc. 19th Annual Joint Conf. IEEE Computer and Communications Societies – INFOCOM2000*, 26–30 March, Tel Aviv, Israel: 585–594.
46. White JG, Southgate E, Thompson JN, Brenner S (1986) The structure of the nervous system of the nematode C. elegans. *Philosophical Transactions of the Royal Society of London – Series B: Biological Sciences*, 314(1165): 1–340.
47. Williams, RJ, Martinez ND (2000) Simple rules yield complex food webs. *Nature*, 404: 180–183.
48. Ygge F, Akkermans JM (1996) Power load management as a computational market. In: Tokoro M (ed) *Proc. 2nd Intl. Conf. Multi-Agent Systems – ICMAS'96*, 9–13 December, Kyoto, Japan, AAAI Press, Menlo Park, CA: 393–400.
49. Zhang T, Ramakrishnan R, Livny M (1997) BIRCH: a new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2): 141–182.

Decentralised Clustering Algorithm

This Appendix reproduces the description of the clustering algorithm proposed by [26].

Each agent is initially a follower to itself, and its followers' list will contain only itself. Each agent is also a cluster-head initially (a singleton cluster). The communication messages (shown in italics) are 'flooding' broadcasts. The algorithm involves the following steps carried out by each cell (agent) which sensed the value x (henceforth, references like 'larger' or 'smaller' are relative to this value):

1. It keeps broadcasting its *recruit* message initially (*recruit* messages will always contain the followers' list of an agent). This broadcasting is done periodically, with a broadcasting-period P, affecting all agents with values within a particular offset of the value x of this agent – in other words, with values between $x - \varepsilon$ and $x + \varepsilon$. The offset ε is initially set to a proportion α of its agent value: $\varepsilon = \alpha x$;
2. If an agent in a singleton cluster receives a *recruit* message from a 'smaller' agent, it ignores it;
3. If an agent p in a singleton cluster receives a *recruit* message from a 'larger' agent q in a singleton cluster, it becomes its follower, stops broadcasting its own *recruit* messages and sends its information to its new cluster-head q: an *acceptance* message with its relative coordinates and the agent-value x. It also stores details of the cluster-head q: the agent-value x_q and relative coordinates;
4. If an agent p in a singleton cluster receives a *recruit* message from a 'larger' agent q which does have other followers, it ignores the message: simply because the 'larger' agent q would also receive and handle a *recruit* message from p itself (see step 6);
5. If an agent receives an *acceptance* message from some potential follower agent, it adds the agent involved in its followers' list;

6. If a member of a non-singleton cluster, either the head or a follower, receives a *recruit* message (either from a 'larger', 'smaller' or 'equal' agent), it *forwards* it to its present cluster-head;
7. After forwarding a *recruit* message to its cluster-head, a follower ignores further *recruit* messages until the identity of its head has been re-asserted (as a result of the clustering heuristic being invoked somewhere);
8. The cluster-head waits for a certain period W, collecting all such *forward* messages (the period W, called heuristic-period, should be greater than $2P$). At the end of the heuristic-period, the clustering heuristic is invoked by the cluster-head on the union set of followers and all agents who *forwarded* the messages. The 'largest' agent in any resulting cluster is appointed as its cluster-head;
9. The cluster-head which invoked the heuristic notifies new cluster-heads about their appointment, and sends their cluster maps to them: a *cluster-information* message;
10. A cluster-head stops sending its *recruit* messages P cycles before it invokes the clustering heuristic. If it is re-appointed as a cluster-head, it resumes sending *recruit* messages;
11. If an agent receives *cluster-information* message it becomes a cluster-head. If it was already a cluster-head with a cluster map, it erases that cluster map and accepts the new cluster map. It also *notifies* all its new followers;
12. A follower will periodically get *recruit* messages from its cluster-head. If this does not happen for a while, then it means that this follower is no longer in the followers' list of its cluster-head. Then it will make itself a cluster-head and start sending its own *recruit* messages. The offset of these *recruit* messages will be determined by the offsets it had when it was a cluster-head the last time (not necessarily the same as ε).

Because of the unpredictable timing of the clustering heuristics being invoked in various agents, it is possible that a cluster-head keeps a particular agent as its follower even after its offset ε has changed and this particular agent is now out of range. To counter this, the cluster-head checks its followers' list periodically and removes agents with values out of range. It is also possible that a node detects a new sensor reading, possibly increasing the agent-value by a large amount. If this agent was a follower, it immediately becomes a cluster-head and *updates* its former cluster-head. The former cluster-head will delete it from its followers' list.

Depending on the nature of the set of agent values, the offset ε may be initially too small to reach any other agent. To counter this, an agent periodically (with a period δ) increases its offsets exponentially until a certain limit: $\varepsilon_{k+1} = \max(2\varepsilon_k, \beta \text{x})$, where $\varepsilon_0 = \varepsilon$ initially, and β is the limit proportion (for example, the initial ε_0 may be $0.01x$ and after 5 periods the offset would become $\varepsilon_5 = 0.32x$). Alternatively, the increase will stop when the offsets of an agent have been reset by the clustering heuristic. When the clustering heuristic is applied, it may produce either one or two clusters as a result. If there are two clusters, the offset of each new cluster-heads is modified. It is adjusted in such a way that the cluster-head of the ‘smaller’ agents can now reach up to, but not including, the ‘smallest’ agent in the cluster of ‘larger’ agents. Similarly, the cluster-head of ‘larger’ agents can now reach down to, but not including, the ‘largest’ agent (the cluster-head) of the cluster of ‘smaller’ agents. These adjusted offsets are sent to the new cluster-heads along with their cluster maps.

Predictor K_2

This Appendix describes the estimation method adopted by [34].

Suppose that the d-dimensional phase space is partitioned into boxes of size r^d. Let $P_{1_0,\ldots i_{d-1}}$ be the joint probability that a trajectory is in box i_0 at time 0, in box i_1 at time Δt,..., and in box i_{d-1} at time (d$-$1)Δt, where Δt is the time interval between measurements on the state of the system (in our case, we may assume $\Delta t = 1$, and omit the $\text{limit}_{\Delta t \to 0}$ in the following definitions). The order-2 Rényi entropy K_2 is defined as

$$K_2 = \lim_{\Delta \to 0} \lim_{r \to 0} \lim_{d \to \infty} \frac{1}{d\Delta t} ln \sum_{i_0 \cdots i_{d-1}} P^2_{i_0 \cdots i_{d-1}} \tag{1}$$

It is well-known that KS Entropy $K = 0$ in an ordered system, K is infinite in a random system, and K is a positive constant in a deterministic chaotic system. Grassberger and Procaccia (1983) considered the 'correlation entropy' K_2 in particular, and capitalised on the fact $K \geq K_2$ in establishing a sufficient condition for chaos $K_2 > 0$. The Grassberger and Procaccia algorithm estimates the entropy K_2 as follows

$$K_2 = \lim_{r \to 0} \lim_{d \to \infty} \lim_{N \to \infty} ln \frac{C_d(N,r)}{C_{d+1}(N,r)} \tag{2}$$

where $C_d(N,r)$ is the correlation integral

$$C_d(N,r) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N} \theta(r- \parallel V_i - V_j \parallel) \tag{3}$$

Here θ is the Heaviside function (equal to 0 for negative argument and 1 otherwise), and the vectors V_i and V_j contain elements of the observed time series $\{v(t)\}$, 'converting' or 'reconstructing' the dynamical information in one-dimensional data to spatial information in the d-dimensional embedding space: $V_k = (v_k, v_{k+1}, v_{k+2}, \ldots, v_{k+d-1})$ [28]. The norm $\parallel V_i - V_j \parallel$ is the distance (sometimes called precision) between the vectors in the d-dimensional space, for example, the maximum norm

$$\parallel V_i - V_j \parallel = max_{\tau=0}^{d-1}(\nu_{i+\tau} - \nu_{j+\tau}) \tag{4}$$

Put simply, $C_d(N, r)$ computes the fraction of pairs of vectors in the d-dimensional embedding space that are separated by a distance less than or equal to r. Since we consider only an initial segment of the times series, we simply set $N = \Omega$, estimating the entropy as

$$K_2(d, r, \Omega) = ln\left(\frac{C_d(\Omega, r)}{C_{d+1}(\Omega, r)}\right) \tag{5}$$

Resources

1 Key Books

Barabási A-L (2002) *Linked: The New Science of Networks.* Perseus Publishing, Cambridge, MA

Bonabeau E (1999) *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, New York, NY

Ferber J (1999) *Multi-agent systems.* Addison Wesley, Reading, MA

Rényi A (1970) *Probability Theory.* North-Holland, Amsterdam

2 Key Survey/Review Article

Solé RV, Ferrer-Cancho R, Montoya JM, Valverde S (2002) Selection, tinkering and emergence in complex networks – crossing the land of tinkering. *Complexity* 8(1): 20–33

3 Organisations, Societies, Special Interest Groups

Santa Fe Institute
http://www.santafe.edu/index.php

The New England Complex Systems Institute (NECSI)
http://necsi.org/

Institute for the Study of Complex Systems (ISCS)
http://www.complexsystems.org/

The Society for Modeling and Simulation International (SCS)
http://www.scs.org/

The ARC Complex Open Systems Research Network (COSNet)
http://www.complexsystems.net.au/

4 Research Groups

The ARC Centre of Excellence for Autonomous Systems, Sydney, Australia
http://www.cas.edu.au/home.html

The Adaptive Systems Research Group, University of Hertfordshire, UK
http://homepages.feis.herts.ac.uk/~comqdp1/#asrg

The Intelligent Interactive Distributed Systems group, Vrije Universiteit Amsterdam, The Netherlands
http://www.iids.org/

The Distributed Intelligence group, CSIRO ICT Centre, Sydney, Australia
http://www.ict.csiro.au/page.php?cid=40

Center for Complex Networks Research, The University of Notre Dame
http://www.nd.edu/~networks/index.htm

The Complex Systems Lab, The Universitat Pompeu Fabra
http://complex.upf.es/~ricard/complexnets.html

Networks and Chaos: Kaneko's Lab
http://chaos.c.u-tokyo.ac.jp/

5 Discussion Group, Forum

Complexity Digest
http://www.comdig.org

6 Key International Conferences/Workshops

International Conference on the Simulation of Adaptive Behavior (SAB)

European Conference on Artificial Life (ECAL)

Int. Conf. on the Simulation and Synthesis of Living Systems (ALife)

Intl. Joint Conf. Autonomous Agents and Multi-Agent Systems (AAMAS)

Intl. Conf. Multisensor Fusion & Integration for Intelligent Systems (MFI)