Identification of Dynamical Structures in Artificial Brains: An Analysis of Boolean Network Controlled Robots

Andrea Roli[1,3], Marco Villani[2,3], Roberto Serra[2,3], Lorenzo Garattoni[4], Carlo Pinciroli[4], and Mauro Birattari[4]

[1] Dept. of Computer Science and Engineering (DISI) - Università di Bologna, Italy
[2] Dept. of Physics, Informatics and Mathematics, Università di Modena e Reggio Emilia, Modena, Italy
[3] European Centre for Living Technology, Venezia, Italy
[4] IRIDIA, Université Libre de Bruxelles, Belgium

Abstract. Automatic techniques for the design of artificial computational systems, such as control programs for robots, are currently achieving increasing attention within the AI community. A prominent case is the design of artificial neural network systems by means of search techniques, such as genetic algorithms. Frequently, the search calibrates not only the system parameters, but also its structure. This procedure has the advantage of reducing the bias introduced by the designer and makes it possible to explore new, innovative solutions. The drawback, though, is that the analysis of the resulting system might be extremely difficult and limited to few coarse-grained characteristics. In this paper, we consider the case of robots controlled by Boolean networks that are automatically designed by means of a training process based on local search. We propose to analyse these systems by a method that detects mesolevel dynamical structures. These structures are emerging patterns composed of elements that behave in a coherent way and loosely interact with the rest of the system. In general, this method can be used to detect functional clusters and emerging structures in nonlinear discrete dynamical systems. It is based on an extension of the notion of *cluster index*, which has been previously proposed by Edelman and Tononi to analyse biological neural systems. Our results show that our approach makes it possible to identify the computational core of a Boolean network which controls a robot.

1 Introduction

The design of artificial systems by means of automatic techniques, such as evolutionary computation techniques, is a well-known approach in the AI community since decades. A plethora of studies has been published in the literature with the aim of showing properties, conditions and characteristics of the emergence of intelligent behaviours. A case in point is the study of the emergence of non-trivial cognitive capabilities in robots, such as sensory-motor coordination [9,10].

M. Baldoni et al. (Eds.): AI*IA 2013, LNAI 8249, pp. 324–335, 2013.

Besides these objectives of foundational and investigation flavour, the automatic design of artificial system is currently achieving increasing attention also because it opens the possibility of designing innovative systems or finding solutions for complex problems that are quite hard to solve with classical approaches [3,6]. One of the main advantages of automatic design is that the designer can specify just the minimal set of constraints and objectives on the final system; for example, when designing neural network systems, one can let the design process decide the topology of the network together with the connection weights. This makes it possible to explore larger design spaces than those explored by classical design techniques, but it has the drawback that the resulting system might be very difficult to analyse. In this work, we propose to analyse these systems by a method that detects *mesolevel dynamical structures*, i.e., emerging patterns composed of elements that behave in a coherent way and loosely interact with the rest of the system. The method is based on an extension of the notion of *cluster index*, which has been previously proposed by Edelman and Tononi for analysing biological neural systems [15] and can be used to detect functional clusters and emerging structures in nonlinear discrete dynamical systems. We apply this method to analyse the Boolean network trained to control a robot that must be able to walk along a corridor without collisions. Results show that this makes it possible to identify structures inside the network which perform main information processing jobs. The results we present are preliminary, but, even if at this early stage, we believe they anyway show the potential of this method so that to motivate their diffusion to the AI community. Indeed, the approach is very general as it only requires a collection of states traversed by the system and it does not need information on the topology nor the functions of the elements composing the system. Nevertheless, this method is able to identify structures related to relevant information processing in the actual functioning of the system.

We provide an introduction to the method in Section 2. In Section 3 we illustrate the case study we analyse, namely an application of Boolean networks to robotics. We present the results in Section 4 and we conclude with Section 5.

2 The Cluster Index

In the following, we consider a system U, composed of N elements that can assume values in finite and discrete domains and update their state in discrete time. The value of element i at time $t+1$, $x_i(t+1)$, is a function of the values of a fixed set elements in U at time t. The cluster index is defined with the aim of identifying subsets of U composed of elements that interact much more strongly among themselves than with the rest of the system, i.e., subsets whose elements are characterised by being both *integrated* among themselves and *segregated* w.r.t. the rest of the system. The quantity upon which the cluster index is computed is the entropy of single as well as sets of elements of U. The entropy of element x_i is defined as:

$$H(x_i) = -\sum_{v \in V_i} p(v) \ \ log\ p(v) \tag{1}$$

where V_i is the set of the possible values of x_i and $p(v)$ the probability of occurrence of symbol v. The entropy of a pair of elements x_i and x_j is defined upon joint probabilities:

$$H(x_i, x_j) = -\sum_{v \in V_i} \sum_{w \in V_j} p(v, w) \ log\ p(v, w). \tag{2}$$

This definition can be extended to sets of k elements by considering the probability of occurrence of vectors of k values. We can estimate the entropy of each element from a long series of states by taking the frequencies of its observed values as proxies for probabilities. Therefore, the sole piece of information we need is a collection of system states, which can be taken by observing the system in different working conditions. For example, the collection can be obtained by composing several trajectories in the state space; however, there are no requirements on the sequence of these states, because the collection is only used to compute frequencies. Once the data have been collected, relevant sets of elements (clusters, from now on) are evaluated by means of the cluster index. A relevant cluster should be composed of elements *(i)* that possess high integration among themselves and *(ii)* that are loosely coupled to other parts of the system. The measure we define provides a value that can be used to rank various candidate clusters (i.e., emergent intermediate-level sets of coordinated elements). Depending on the size of the system, candidate clusters can be exhaustively enumerated, or sampled, or searched by means of suitable heuristics. The cluster index $C(S)$ of a set S of k elements is defined as the ratio of their *integration* $I(S)$ to the *mutual information* between S and the rest of the system $U - S$. The integration is defined as follows:

$$I(S) = \sum_{x \in S} H(x) - H(S) \tag{3}$$

$I(S)$ measures the deviation from statistical independence of the k elements in S, by subtracting the entropy of the whole subset to the sum of the single-node entropies. The mutual information between S and the rest of the system $U - S$ is:

$$M(S; U - S) \equiv H(S) + H(S|U - S) = H(S) + H(U - S) - H(S, U - S) \tag{4}$$

where, as usual, $H(A|B)$ is the conditional entropy and $H(A, B)$ the joint entropy. Finally, the cluster index $C(S)$ is defined by:

$$C(S) = \frac{I(S)}{M(S; U - S)} \tag{5}$$

Special cases are: $I = 0 \wedge M \neq 0 \ \Rightarrow \ C(S) = 0$ and $M = 0 \ \Rightarrow \ C(S)$ not defined. These cases can be diagnosed in advance. The $0/0$ case does not provide any information, whereas $I(S)/0$, with $I(S) \neq 0$, denotes statistical independence of S from the rest of the system and requires a separate analysis.

$C(S)$ scales with the size of S, so a loosely connected subsystem may have a larger index than a more coherent, smaller one. Therefore, to compare the indices of the various candidate clusters, it is necessary to normalise their cluster indices. To this aim, we need a reference system with no clusters, i.e., an homogeneous system, which we define as follows: given a series of states from the system we want to study, we compute the frequency of each symbol and generate a new random series in which each symbol appears with probability equal to that of the original series. The homogeneous system provides us with reference values for the cluster index and makes it possible to compute a set of normalisation constants: for each subsystem size, we compute average integration $\langle I_h \rangle$ and mutual information $\langle M_h \rangle$. We can then normalise the cluster index value of any subsystem S using the appropriate normalisation constants dependent on the size of S:

$$C'(S) = \frac{I(S)}{\langle I_h \rangle} / \frac{M(S; U - S)}{\langle M_h \rangle} \tag{6}$$

In order to compute a statistical significance index (hereinafter referred to as T_c), we apply this normalisation to both the cluster indices in the analysed system and in the homogeneous system:

$$T_c(S) = \frac{C'(S) - \langle C'_h \rangle}{\sigma(C'_h)} \tag{7}$$

where $\langle C'_h \rangle$ and $\sigma(C'_h)$ are respectively the average and the standard deviation of the population of normalised cluster indices with the same size of S from the homogeneous system [2]. Finally, we use T_c to rank the clusters considered.

We have recently applied our method to both artificial test cases and representative natural and artificial systems, such as genetic regulatory networks and catalytic reactions systems [16,17]. In the following, we show that the method can be used also to analyse the network controlling a robot. This case is particularly meaningful because of two main reasons. The first is that it explicitly concerns a system equipped with inputs and outputs, which operates in an environment. Therefore, the relevant structures inside the network are necessarily linked to the interplay between robot behaviour and environment. The second reason is that the networks resulting at the end of the training process are not easily analysable because they have a random topology and the nodes are updated according to Boolean functions: we will show that the method is able to capture relevant structures inside the network without requiring knowledge about network topology and functions.

3 BN-Robot Case Study

We apply our method to analyse the networks trained to control a robot that performs obstacle avoidance. The robot is an *e-puck* [7] and it is controlled by a Boolean network. Boolean networks (BNs) are a model of genetic regulatory networks [5]. BNs have received considerable attention in the community of complex system science. Works in complex systems biology show that BNs provide a

powerful model for cellular dynamics [1,13,14]. A BN is a discrete-time discrete-state dynamical system whose state is a N-tuple in $\{0,1\}^N$ and it is updated according to the composition of N Boolean functions, each of which rules the update of one variable of the system. Usually, BNs are subject to a synchronous and parallel node update, but other update schemes are possible. BNs are extremely interesting from an engineering perspective because of their ability to display complex and rich dynamics, despite the compactness of their description. In a previous work, it has been shown that BNs can be used to control robots [12].

In the case study, the robot must navigate along a corridor avoiding any collision with the walls and possible obstacles and finally reach the exit. At the beginning of each experimental run, the robot is placed within the corridor, far from the exit. During the experiment the robot must advance along the corridor avoiding collisions, and finally, within the given total execution time $T = 120\,\text{s}$, reach the exit. During the execution, if a collision between the robot and the walls of the corridor occurs, the experiment is immediately stopped. Experiments are performed in simulation, by means of the open source simulator ARGoS [11]. The performance measure is the final distance of the robot to the exit (normalised w.r.t. corridor length). The shorter is this distance, the better is the performance of the robot. The robot is equipped with four proximity sensors, placed at positions NE, SE, SW and NW with respect to the heading direction, and with two wheels. At each time step, the readings of the 4 sensors are encoded into the values of the BN input nodes. We use 4 input nodes to encode the readings of the proximity sensors. Values are binarised by introducing a threshold: if the sensor value exceeds the chosen threshold, the corresponding input node value is set to 1. Once the readings of the sensors are encoded in the input nodes, we perform the network state update, and eventually we read and decode the values of the output nodes to set the actuators. Two output nodes are used to set the wheel speeds either to zero or to a predefined, constant value. The robot update frequency is 100 ms. For this case study, we set the network size to 20 nodes. The initial topology of the networks, i.e., the connections among the nodes, is randomly generated with $K = 3$ (i.e., each node has 3 incoming arcs) and no self-connections, and it is kept constant during the training. The initial Boolean functions are generated by setting the 0/1 values in the node Boolean function truth tables uniformly at random. BNs are trained by a local search algorithm which works only on the Boolean functions. At each iteration, the search algorithm flips one bit of a Boolean function. The flip is accepted if the corresponding BN-robot has a performance not worse than the current one.[1] The evaluation of each network is performed on a set of initial conditions, that form the training set. The training set is composed of six different initial orientations of the robot. The six angles are chosen so as to have six equally spaced orientations in the range between $\frac{\pi}{3}$ and $-\frac{\pi}{3}$ (with 0 being the straight direction of the robot towards the exit). In this way, the robot must be able to cope with a wide range of different situations and avoid the walls it detects in any

[1] This simple local search algorithm is often called *stochastic descent* or *adaptive walk*.

direction. The final evaluation assigned to the robot is computed as the average of the performance across the 6 trials. We executed 100 independent experiments, each corresponding to a different initial network. For each experiment the local search was run for 1000 iterations. The assessment of the performance of the BN-robots is performed on a test set composed of randomly generated initial positions and orientation. All the final BNs achieved a good performance. For further details, we point the interested reader to a comprehensive report on the experiment [4].

4 Identifying Structures in the Boolean Network Controlling a Robot

We analysed the best performing BNs by means of the method based on the cluster index, with the aim of identifying their core structure, if any. It is important to stress that this method detects dynamical relations among nodes of the system, without requiring information on the topology nor on the function computed by the single nodes. The objective is to identify parts of the system which are dynamically coordinated, rather than to discover topological patterns, such as communities and motifs. Furthermore, this technique captures correlations among sets of nodes and not just between pairs.

In the following, we describe the results of two BN-robots we obtained, which are typical examples. In both cases, the BN-robot learned to walk along the corridor without colliding against the walls and obstacles in the path. The data required by this analysis are simply a collection of states of the system. To this aim, we recorded the states traversed by the BN controlling the robot starting from 200 random initial positions. The length of the trajectories is 1000 steps. We then analysed these states, and searched for clusters of size up to 19 (i.e., $N - 1$) by taking 10^4 random samples for each cluster size.

4.1 Network 1

We analysed the collection of BN states related to the test runs of the BN-robot by looking for the cluster with highest significance T_c. The most significant cluster has size 8, with $T_c \approx 48 \times 10^3$. The following clusters have a much lower significance (about 36×10^3), they are of size 9, and they all contain the first cluster of 8 nodes. Therefore, they are not particularly meaningful. The identified cluster is composed of input, output and internal nodes, namely nodes 1, 3, 5, 7, 8, 10, 12 and 18 of the network depicted in Figure 1. Besides input nodes 1 (NW proximity sensor), and 3 (SE proximity sensor) and node 5 which controls the left wheel, the cluster contains nodes involved in internal dynamics.

The nodes of the cluster should convey relevant information on the overall BN dynamics. A way to reckon the impact of a cluster on the system dynamics is to evaluate its *set-influence*. The notion of influence of a set of nodes is a generalisation of the *node influence*, which is defined as the amount of perturbation induced on a set of nodes by a state change in one node. Informally, the influence

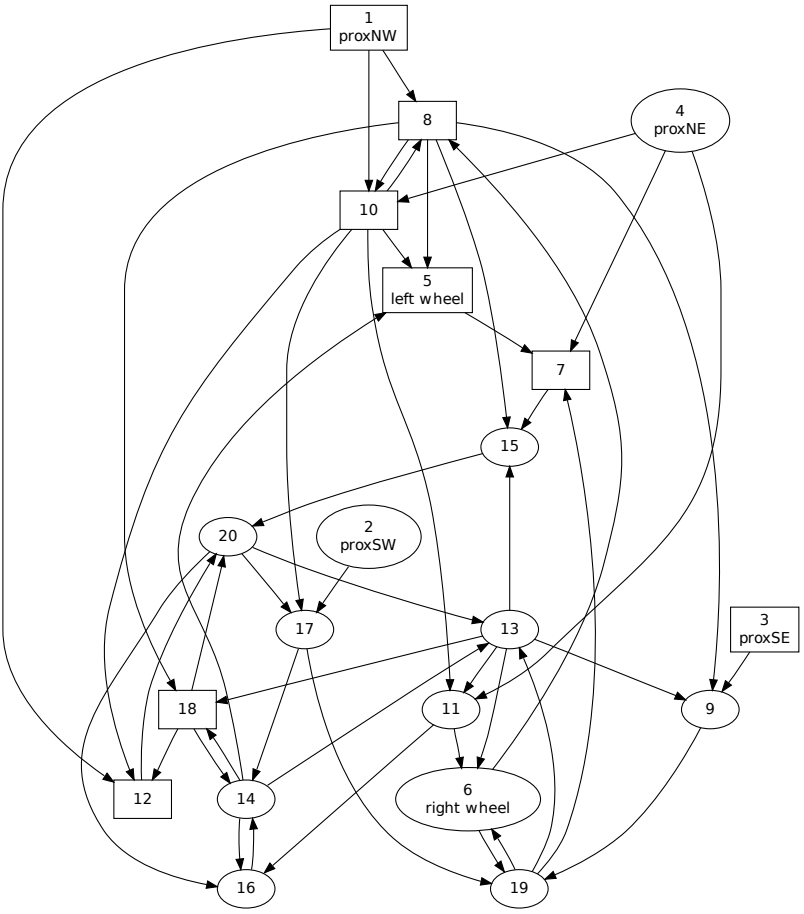

Fig. 1. Structure of BN-1 controlling the robot. Nodes drawn with rectangles are in the most significant cluster.

of a node on the other nodes is the size of the *avalanche* produced by the node perturbation [14]. To estimate the influence of a set S of nodes on the other nodes of the system, $S' = U - S$, we randomly perturb the nodes in S[2] at time t and we compare the state at time $t+l$ in the case with and without perturbation (with time lag $l \in \{1, 2, \ldots, 10\}$). The normalised Hamming distance between the two states is the avalanche produced by perturbing S. We estimated the set-influence of cluster {1,3,5,7,8,10,12,18} and of random clusters of the same size by taking the average avalanche over 100 random initial states. Results are shown in Figure 2. As we can observe, cluster {1,3,5,8,10,12,18} has a higher set-

[2] More precisely, the perturbation is performed by flipping each node state with probability 0.5 .

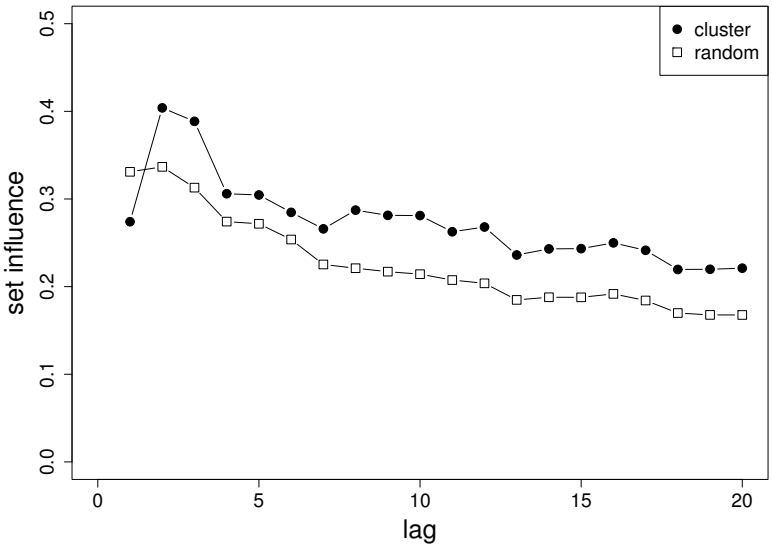

Fig. 2. Set-influence of cluster {1,3,5,7,8,10,12,18} compared with the set-influence of random clusters of the same size (averaged over 100 random clusters)

influence than random sets,[3] hence the functionality of the BN strongly depends on the identified cluster. One might speculate that the most relevant cluster is characterised by a higher set-influence w.r.t. the others, and therefore the search for a significant dynamical structure in the system can be simply performed by inspecting the sets with the highest set-influence. Nevertheless, it should be observed that we are looking for structures relevant for information processing inside the system. These structures are relevant for the *actual functioning* of the system and might not be simply reduced to sets of nodes with high average influence: a node can indeed have high influence by influencing other nodes which are not relevant for the actual dynamics of the system. The next case we present is a prominent example of this phenomenon.

4.2 Network 2

The second BN we analyse has a different topology w.r.t. the first one, because it has been generated with a different random number generator seed. As also the initial BN functions are randomly generated and the search algorithm is stochastic, the training process led to a different solution which anyway achieves the goal.

In this case, the most significant cluster has size 3 and it has a significance value $T_c \approx 1.3 \times 10^6$. The following clusters have a much lower significance (about

[3] Except for the case of $l = 1$, which means that the cluster needs some time to spread the information across the network.

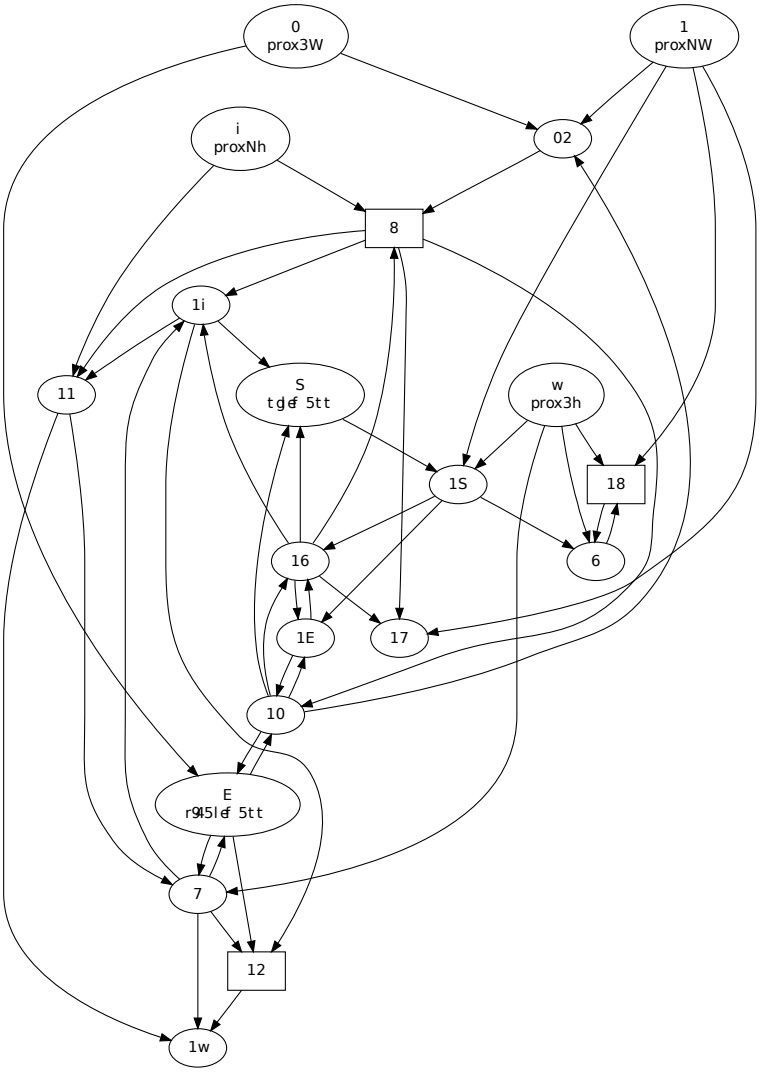

Fig. 3. Structure of BN-2 controlling the robot. Nodes drawn with rectangles are in the most significant cluster.

9.8×10^5) and they are of size 4, all including the first cluster of size 3. The identified cluster is composed of nodes {8,10,18} and it does not contain input nor output nodes (see Figure 3). This result might seem quite surprising, but by looking at the network topology we can observe that (directly or indirectly):

- node 8 collects (and processes) the information coming from sensors SW, NW and NE; moreover, it acts on the right wheel;

- node 10 collects the information coming from node 8 and sensor SE (which produces values obviously correlated with the information coming from the other sensors SW, NW and NE);
- node 18 collects the information coming from sensors SE and NW.

Therefore, these observations help identify sensors SE and NW, and node 8 as the central part of the robot information processing unit, with nodes 10 and 18 playing the role of faithful followers. The set-influence of the cluster is compared against the set-influence of random clusters of the same size in Figure 4. The set-influence of the cluster is lower than the average one; therefore, this sole piece of information would not be sufficient to detect this structure. The reason for low set-influence but high cluster index is very likely that nodes 10 and 18 have low influence on the network, hence low mutual information between the cluster and the remainder of the system. A single node influence analysis is anyway helpful to understand the reason why node 10 is chosen by our method instead of an output node. Node influence is evaluated by computing the *influence matrix* (defined for lag l), in which entry (i, j) is the influence of node i on node j at time l. The influence at lag l is computed as the fraction of times in which the value of node j is affected by a flip in node i occurred l time steps before. If we rank the nodes by influence, among the most five influential nodes on node 10 we find node 6 (left wheel) with the highest influence at lag equal to 1 and nodes 1,2,3 and 4 (i.e., the sensors) for higher time lag values.

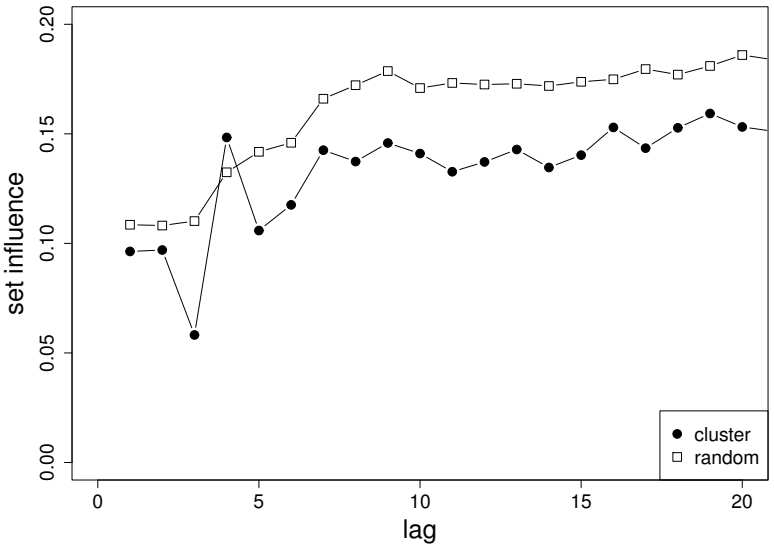

Fig. 4. Set-influence of cluster {8,10,18} compared with the set-influence of random clusters of the same size (averaged over 100 random clusters)

We can conclude that the method based on the cluster index is able to detect subsystems which play a main role in the information processing inside the system: they can be both the causal core of the functioning of the system and a proxy for observing its main dynamical properties.

5 Conclusion

The results we have presented show that the method based on the cluster index can help us detect relevant structures which emerge as the result of the dynamics of the system. This work is at a preliminary stage and further analyses are required to assess the informative power of the cluster index and its possible applications. For example, our method can provide heuristics to reduce the network by pruning irrelevant nodes and links, or it can be used to extract a minimal subset of nodes to observe the system. In addition, whilst in this work we only focused on checking whether our results are meaningful w.r.t. the dynamics of the system, our method can be used as a first step in the identification of *functional modules* of the system [8]. In the future, our method may be applied to detect structures emerging either during the learning process of a single system, or in the evolution of populations of systems, or both.

Acknowledgements. This article has been partially funded by the UE projects "MD – Emergence by Design", Pr.ref. 284625 and "INSITE - The Innovation Society, Sustainability, and ICT" Pr.ref. 271574, under the 7th FWP - FET programme.

References

1. Aldana, M., Balleza, E., Kauffman, S., Resendiz, O.: Robustness and evolvability in genetic regulatory networks. Journal of Theoretical Biology 245, 433–448 (2007)
2. Benedettini, S.: Identifying mesolevel dynamical structures. Tech. rep., ECLT (European Center for Living Technologies) (2013)
3. Floreano, D., Dürr, P., Mattiussi, C.: Neuroevolution: from architectures to learning. Evolutionary Intelligence 1(1), 47–62 (2008)
4. Garattoni, L., Pinciroli, C., Roli, A., Amaducci, M., Birattari, M.: Finite state automata synthesis in boolean network robotics. Tech. Rep. TR/IRIDIA/2012-017, IRIDIA, Université Libre de Bruxelles, Brussels, Belgium (September 2012)
5. Kauffman, S.: The Origins of Order: Self-Organization and Selection in Evolution. Oxford University Press, UK (1993)
6. Lipson, H.: Evolutionary robotics and open-ended design automation. In: Cohen, B. (ed.) Biomimetics, pp. 129–155. CRC Press (2005)
7. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., Martinoli, A.: The e-puck, a robot designed for education in engineering. In: Gonç alves, P., Torres, P., Alves, C. (eds.) Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, vol. 1, pp. 59–65 (2009)

8. Müller, G., Wagner, G., Callebaut, W. (eds.): Modularity – Understanding the Development and Evolution of Natural Complex Systems. The Vienna Series in Theoretical Biology. The MIT Press (2005)
9. Nolfi, S., Floreano, D.: Evolutionary robotics. The MIT Press, Cambridge (2000)
10. Pfeifer, R., Scheier, C.: Understanding Intelligence. The MIT Press (2001)
11. Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G., Ducatelle, F., Birattari, M., Gambardella, L., Dorigo, M.: ARGoS: A modular, multi-engine simulator for heterogeneous swarm robotics. Swarm Intelligence 6(4), 271–295 (2012)
12. Roli, A., Manfroni, M., Pinciroli, C., Birattari, M.: On the design of Boolean network robots. In: Di Chio, C., et al. (eds.) EvoApplications 2011, Part I. LNCS, vol. 6624, pp. 43–52. Springer, Heidelberg (2011)
13. Serra, R., Villani, M., Barbieri, A., Kauffman, S., Colacci, A.: On the dynamics of random Boolean networks subject to noise: Attractors, ergodic sets and cell types. Journal of Theoretical Biology 265(2), 185–193 (2010)
14. Serra, R., Villani, M., Graudenzi, A., Kauffman, S.: Why a simple model of genetic regulatory networks describes the distribution of avalanches in gene expression data. Journal of Theoretical Biology 246, 449–460 (2007)
15. Tononi, G., McIntosh, A., Russel, D., Edelman, G.: Functional clustering: Identifying strongly interactive brain regions in neuroimaging data. Neuroimage 7, 133–149 (1998)
16. Villani, M., Benedettini, S., Roli, A., Lane, D., Poli, I., Serra, R.: Identifying emergent dynamical structures in network models. In: Proceedings of WIRN 2013 – Italian Workshop on Neural Networks (2013)
17. Villani, M., Filisetti, A., Benedettini, S., Roli, A., Lane, D., Serra, R.: The detection of intermediate-level emergent structures and patterns. In: Liò, P., Miglino, O., Nicosia, G., Nolfi, S., Pavone, M. (eds.) Advances in Artificial Life, ECAL 2013, pp. 372–378. The MIT Press (2013)