Project Number 001907

DELIS

Dynamically Evolving, Large-scale Information Systems

Integrated Project

Member of the FET Proactive Initiative **Complex Systems**

Deliverable D5.2.3

Degeneracy and Redundancy in Human-constructed Information Systems

Start date of the project:	January 2004
Duration:	48 months
Project Coordinator:	Prof. Dr. math. Friedhelm Meyer auf der Heide Heinz Nixdorf Institute, University of Paderborn, Germany
Due date of deliverable:	December 2005
Actual submission date:	January 2006
Dissemination level:	PU – public
Work Package 5.2:	Evolved tinkering and degeneracy as engineering concepts
Participants:	Universtitat Pompeu Fabra (UPF), Barcelona, Spain Universita di Bologna (UniBO), Italy
Authors of deliverable:	Javier Macia (javier.macia@upf.edu) Ricard V. Solé (ricard.sole@upf.edu) Sergi Valverde (svalverde@imim.es) David Hales (hales@cs.unibo.it) Ozalp Babaoglu (babaoglu@cs.unibo.it)

Abstract

This report comprises the complete D5.2.3 deliverable as specified for workpackage WP5.2 in Subproject SP5 of the DELIS (Dynamically Evolving Large-scale Information Systems) Integrated Project.

The essential goal of the DELIS project is to understand, predict, engineer and control large evolving information systems. It is desirable for such systems to display functional robustness under internal and external noise and perturbations. Given that many biological and technological networks demonstrate this, it would be of value to understand how this is achieved. The approach of this workpackage is to explore the evolution of simple feed-forward digital logic circuits and to characterise their robustness due to both redundancy and degeneracy. Initial results are presented and preliminary conclusion drawn, also we briefly relate some of these insights peer-to-peer design practices. Possible future work is outlined.

Contents

1 Introduction

One remarkable feature of many biological systems is the presence of a high degree of robustness against perturbations. Such robustness appears at multiple scales. Specifically, it is often found that temporal or permanent loss of some components has very often little or no impact on individual performance. In this context, such entities as a whole are able to cope with a changing world even under complete failure of single units. What is more surprising, it has been shown that the mechanisms underlying such reliable behaviour do not lie in redundancy. By redundancy, we refer to the presence of multiple copies of a given component: the failure of one of them would be compensated by another identical copy. Instead, robustness in biology is largely associated to a distributed property, which has been dubbed degeneracy [1, 9]. By degeneracy we refer to *the ability of elements that are structurally different to perform the same function* [1, 9]

This observation has a number of implications for evolved artificial designs, particularly those performing some sort of information processing or computational tasks, including specifically evolved hardware. In trying to imitate (or get inspired by) natural systems, we might find useful to look into degeneracy. It represents a novel component of artificial design, since most of the literature dealing with robust artificial systems is largely dominated by redundancy. Actually, the early work on reliable computation under the presence of unreliable components was explored by von Neumann, Cowan and Vinograd among others in the context of redundant structures. One prominent conclusion of their work was that, in order to explain the enormous resilience exhibited by biological systems (such as brains) based on redundant functions, a huge, unlikely amount of redundancy was necessary in order to achieve the desired robustness. On the other hand, the type of architectural patterns predicted by these studies was far from the one that we observe in real biological networks.

Exploring the problem of degeneracy involves a number of difficulties. One is obvious: we cannot reduce the adaptive behavior of complex systems to a simple, engineer-based view. Since redundancy does not play the role that it was evolved to play, we need to look at the origins of robustness in a distributed way. Different sub-parts, perhaps different paths, might be able to restore the lost function. In order to measure and understand it, new approaches are required. Evolved hardware provides a perfect framework to explore these issues in detail.

Evolved artificial networks based on the application of evolutionary laws and natural selection allow us to obtain new designs without direct human intervention, in many cases displaying a higher efficiency. One of the relevant interests of these evolved systems resides in the properties that differentiate them from engineered designs. Their architecture strongly deviates from the standard design and their way of functioning is most of the time unknown. In our analysis, we have studied the presence of degeneracy, redundancy and fault tolerance in small-sized evolved cirduits. Our objective was to explore the relationships existing between these three different (although related) properties and how they emerge.

2 Measures of robustness

A first step before we present our results on evolved networks is to properly define a set of quantitative measures of network redundancy and degeneracy. To this goal, we will review the work by Edelman and co-workers and show how these information-based measures can be computed from a given circuit. Degeneracy is defined as the ability of elements which being structurally different can make the same function or generate same output. It is known that biological systems, result from evolutionary processes and natural selection presents important degeneracy levels, i.e. the genetic code or the immune system [5]. On the other hand, we considered that in a system there is redundancy when structurally identical elements make the same function.

We can measure the previous definitions in terms of information theoretic quantities [7]. In all

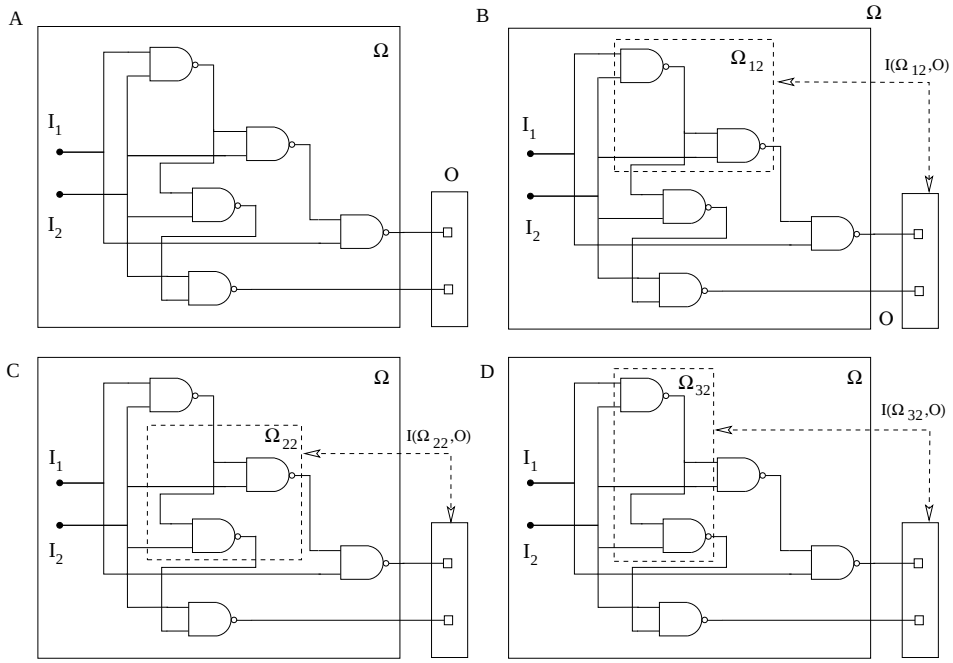

Figure 1: An example of a digital network (A) formed by a small set of NAND gates. Three NAND gates define the X set (and thus $Z = 3$) and two are used as output elements. In order to compute degeneracy (see text) several subsets of the non-output elements must be considered, as indicated in equation (1). Figures B, C and D show the different possible subsets of size 2 that can be considered in the calculation of D_Z.

our definitions, we consider the system as a network of interacting units performing some kind of information processing (in our case, digital circuits) with a well-defined input-output structure. Given a set X formed by $\{X_i\}(i = 1, ..., Z)$ elements connected one to each other and connected with an output layer. Tononi the al. [6] defines the degeneracy of the system as:

$$D_Z(X) = \frac{1}{2}\sum_{k=1}^{Z}\Big(I(X_i^k, O) + I(X - X_i^k, O) - I(X, O)\Big) \tag{1}$$

where X_i^k represents the i-th subset of k elements which is possible to build with the Z elements of the network. The terms in the right-hand side are different contributions to the overall measure, defined in terms of mutual information. Here $I(X_i^k, O)$, for example, indicates the mutual information between X_i^k blocks and the system's output[1]. It can be shown that redundancy $R_Z(X)$ is defined as:

$$R_Z(X) = \Big(\sum_{i=1}^{Z} I(X_i, O)\Big) - I(X, O) \tag{2}$$

thus calculating the overlapping between the information being processed by single basic elements and the rest of the network. If the different elements of the network where independent, we would have $R_Z(X) = 0$. Additional measures (not to be discussed here) include a complexity measure combining previous fetaures as well as network topological attributes.

[1] In general, we have $I(X, Y) = H(X) + H(Y) - H(X, Y)$, where X and Y are two given sub-systems. The entropies $H(X), H(Y)$ are computed by taken into account the probabilities associated to each part, and the joint entropy $H(X, Y)$ provides a measure of correlations between both sub-systems

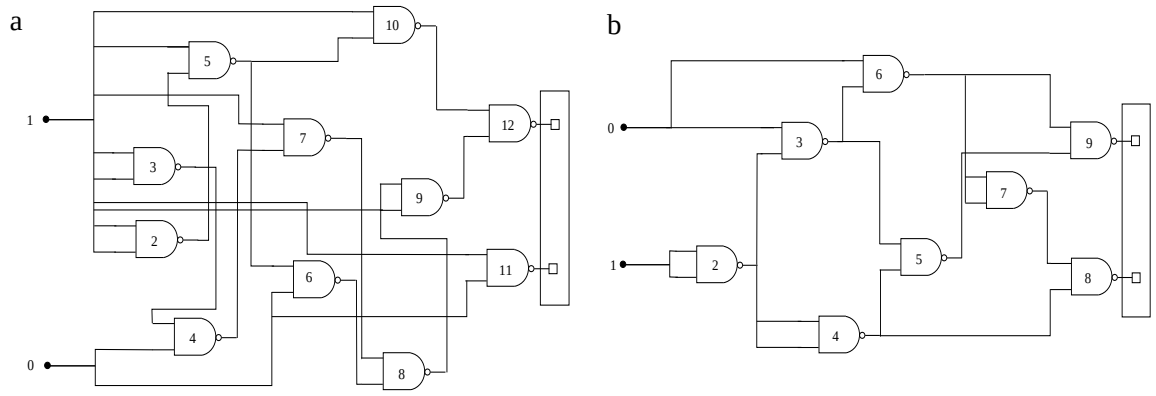

Figure 2: Examples of evolved circuits resulting from a process of evolutionary optimization. In (a) a circuit obtained by conditional evolution is shown. All gates are identical (NAND gates). Nodes 1 and 2 are the input nodes. The circuit outputs are located after nodes 11 and 12. This circuit has a very high fault tolerance value ($\rho = 0.944$). In (b) we show a circuit obtained by neutral evolution. This circuit implements the same logic function than the circuit in (a) but involves a smaller fault tolerance of $\rho = 0.54$.

3 Degeneracy in evolved digital circuits

We have evolved our systems by using two different strategies. Several possible approaches can be followed [3,4] but we restrict ourselves to two simple algorithms. The circuits used in our study are formed by a network of logic gates, which for simplicity are all of the same class (specifically we use S NAND gates), with initially random connections among them. These circuits evolve until being able to implement a certain objective binary function Φ^*, of N inputs and M outputs. An initial population is used, where each circuit will be implementing some arbitrary funcction

$$\Phi_i : \mathbf{I} \rightarrow \mathbf{O} \quad ; \quad i = 1, ..., S \tag{3}$$

with $\mathbf{I} = \{0,1\}^N$ and $\mathbf{O} = \{0,1\}^M$. The only limitation is that, in this case, there are no backward connections, in order to avoid time dependencies (future work will consider the generalization of this scenario). The evolutionary rules are based on the work of J. Miller [3], on which we have imposed additional conditions (selection pressures) in order to canalize the evolution. Two kinds of evolutionary process have been analysed:

Neutral Evolution. The candidate solutions (digital circuits) evolve with a single constraint in the optimization: to perform the desired function. The steps of the algorithm are here:

1. Create a random generated population formed by S NAND gates. Each one of these individuals are formed by Z nodes connected randomly, without backwards connections. This population constitutes the zero generation.

2. The behavior of each one of these individuals is simulated for the $2N$ possible inputs, comparing its corresponding outputs with the objective function outputs. The fitness of the $i-$ candidate Φ_i is defined as:

$$\mathcal{F}(\Phi_i) = 1 - \frac{1}{M2^N} \sum_{i=1}^{M2^N} |O_i - O_i^*| \tag{4}$$

where $\{O_i\}$ is the set of outputs of the circuit for the different inputs, and $\{O_i^*\}$ is the set of objective function outputs.

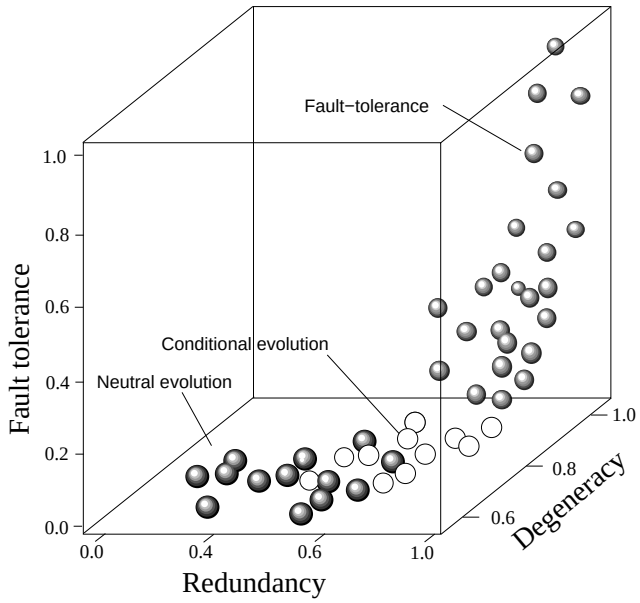

Figure 3: An example of the results obtained from the evolutionary algorithms under different selection pressures. Here different types of target Boolean functions have been chosen and the two types of algorithms are indicated by means of dark (neutral evolution) and light (conditional). There is a clear relation between degeneracy and redundancy.

3. The individual with the greater fitness is chosen to create a new generation formed by the selected individual and $S-1$ random mutations of him. The random mutations can be: (a) Elimination of an existing connection, with probability E_c, (b) Creation of a new connection with probability C_c, (c) Elimination of a node with probability I_n and (d) Creation of a new node with probability C_n.

4. Return to point (2) until F= 1

Conditional Evolution. Basically, this type of evolution is identical to the neutral evolution case. The only difference concerns the 3rd step: The selection of the most optimal individual of each generation is based on the individual that has better fitness and a greater fault tolerance. Fault tolerance ρ is defined as:

$$\rho(\Phi_i) = 1 - \frac{1}{ZM2^N} \sum_{p=1}^{Z} \sum_{i=1}^{M2^N} |O_i - O_i^*| \tag{5}$$

where now the output set of the circuit is obtained under a perturbation of the p-th node. Specifically, for each input all the gates are in a binary state 0 or 1. The perturbation consists of the inversion of the logical state of the gate located at the p-th node. This perturbation is applied for each node of the network.

The evolutionary algorithms successfully find optimal solutions when looking for implementing predefined computations, as defined by Φ^*. Some of our results (for small-size circuits composed by $S \approx 10-16$ elements) are shown in figure 2, for both types of optimization. In figure 3 a more quantitative picture is provided for a range of evolved circuits implementing different Boolean functions. We can see that the values of the degeneracy are much greater for the circuits with conditional evolution, although high levels of degeneracy are found in all cases, thus indicating

that something in this bio-inspired way of designing computational networks leads to distributed robustness. In the case of circuits with conditional evolution the last ones have reached the objective function they can reach high levels of degeneracy evolving to increase their fault tolerance.

4 Summary

The main results of the evolutionary dynamics experiment is that degeneracy seems to be an inevitable outcome of evolved designs, and is strongly related to the fault-tolerant behavior of the circuit. Extensive simulations, using large circuits with many components are difficult to explore due to the computational cost of computing degeneracy as previously defined. New measures, able to capture the essential properties of distributed robustness, but less time consuming, seem to be required. We have also found that increasing the system size has an important influence on increasing the levels of degeneracy and fault tolerance. Although such results might seem reasonable, it is not a simple consequence of a larger number of links, since the system keeps a low number of links, on average. In general, high levels of fault tolerance are strongly tied to high levels of distributed robustness. Future work will consider a spectrum of different fitness functions implementing different types of computations as well as the presence of stochastic fluctuations on the computations.

As discussed in deliverable D5.6.1, the application of robustness tests during the development of peer-to-peer systems is common practice. Designers effectively hill-climb or evolve their protocols (which when executed create and maintain network topologies and interactions) by hand. The main reason for this is the size and complexity of such networks. P2P networks are generally very large (thousands or millions of nodes) and highly dynamic with complex functionality at each node[2]. However, it is possible, as we discussed in D5.6.1, to measure the performance of a given network under various churn characteristics (removing elements of the network and adding new ones) there are practical robustness measures essentially the same as the measure given in equation 5.

It is difficult currently to meaningfully characterise, within given P2P networks, what aspects of robustness are related to degeneracy and what to replicated function. For the reasons given, it would appear that simple information theoretic measures (like mutual information) would not be meaningful for most P2P tasks. However, it may be possible to produce measures for classes of P2P application task that can utilise new measures - for example, in content distribution systems, data is often replicated over different nodes - a form of redundancy. Possible future work could focus on creating meaningful measures for characterising redundancy and degeneracy within evolving P2P systems.

References

[1] Edelman. G. M., Gally, J. A. (2001) Degeneracy and complexity in biological systems. *Proc. Natl. Acad. Sci. vol. 98, 13763-13768.*

[2] Greensted, A. J. and Tyrrell, A. M. (2003) Fault Tolerance via endocrinologic based communication for multiprocessor systems. *Evolvable Systems: From Biology to Hardware, 5th International Conference, ICES 2003, Lecture Notes in Computer Science Vol. 2606.*

[3] Higuchi, T. et al. (Eds.) (1997) *Proceedings 1st International Conference on Evolvable Systems: From Biology to Hardware (ICES96), Lecture Notes in Computer Science 1259: 327-343. Springer-Verlag.*

[2]It is in fact more complex in many deployed P2P systems since protocols do not directly describe networks but rather implement algorithms that when executed over nodes self-organise the network - a kind of morphogensis process

[4] Koza, J. R. (1992) *Genetic Programming: On the programming of computers by means of natural selection.* MIT Press.

[5] Miller, J., Thompson, A., Thompson, P. and Fogarty, T. (editors) (2000). *Proceedings 3rd International Conference on Evolvable Systems: From Biology to Hardware, volume 1801 of LMCS. Springer-Verlag.*

[6] Miller, J., Kalganova, T., Lipnitskaya, N. and Job, D. (2002) The Genetic Algorithm as a Discovery Engine: Strange Circuits and New Principles. *Peter J. Bentley, David W. Corne, editors. Creative Evolutionary Systems. Morgan Kaufmann.*

[7] Miller, J. and Hartmann, M. (2001) Evolving messy gates for fault tolerance: some preliminary findings. *The Third NASA/DoD Workshop on Evolvable Hardware.*

[8] Shannon, C. E., (1948) A Mathematical Thoery of Communication. *The Bell Systems Technical Journal. Vol. 27, 379-423.*

[9] Tononi, G., Sporns, O., Edelman, G. M., (1999) Measures of degeneracy and redundacy in biological networks. *Proc. Natl. Acad. Sci. vol. 96, 3257-3262.*