Design and Control of Self-organizing Systems

Carlos Gershenson
New England Complex Systems Institute
and Vrije Universiteit Brussel

Mexico City Boston Viçosa
Madrid Cuernavaca Beijing
CopIt ArXives
2007

CopIt ArXives
Mexico City Boston Viçosa
Madrid Cuernavaca Beijing

Published 2007 by CopIt ArXives

Electronically produced using Free Software
and in accomplishment with an
Open Access spirit for academic publications

ABSTRACT

Complex systems are usually difficult to design and control. There are several particular methods for coping with complexity, but there is no general approach to build complex systems. In this book I propose a methodology to aid engineers in the design and control of complex systems. This is based on the description of systems as self-organizing. *Starting from the agent metaphor, the methodology proposes a conceptual framework and a series of steps to follow to find proper mechanisms that will promote elements to find solutions by actively interacting among themselves. The main premise of the methodology claims that reducing the "friction" of* interactions *between elements of a system will result in a higher "satisfaction" of the system, i.e. better performance.*

A general introduction to complex thinking is given, since designing self-organizing systems requires a non-classical thought, while practical notions of complexity and self-organization are put forward. To illustrate the methodology, I present three case studies. Self-organizing traffic light controllers are proposed and studied with multi-agent simulations, outperforming traditional methods. Methods for improving communication within self-organizing bureaucracies are advanced, introducing a simple computational model to illustrate the benefits of self-organization. In the last case study, requirements for self-organizing artifacts in an ambient intelligence scenario are discussed. Philosophical implications of the conceptual framework are also put forward.

Contents

List of Figures

List of Tables

Acknowledgements

This book is one of the outcomes of my PhD at the Vrije Universiteit Brussel between November 2002 and May 2007. It takes several years to build up a PhD. And it is impossible to do so alone. Along these years, many mentors, colleagues, and friends have influenced my research, formation, and life in different aspects.

I am in debt with my promoters, Francis Heylighen, Diederik Aerts, and Bart D'Hooghe, who have shared their knowledge and experience, and whose support has gone well beyond the academic.

Collaborations with Johan Bollen, Jan Broekaert, Paul Cilliers, Seung Bae Cools, Atin Das, Bruce Edmonds, Angélica García Vega, Stuart Kauffman, Tom Lenaerts, Carlos de la Mora, Marko Rodriguez, Ilya Shmulevich, Dan Steinbock, Jen Watkins, and Andy Wuensche have enriched my research. I have learned a lot from interacting with them.

Hugues Bersini, Jean-Louis Denebourg, Marco Dorigo, Bernard Manderick, Gregoire Nicolis, Ann Nowe, Luc Steels, René Thomas, Jean Paul Van Bendegem, Frank Van Overwalle, and other professors at the VUB and ULB have provided me with great advice and inspiration.

Being in Brussels, I had the opportunity to interact with several research groups akin to my interests in both VUB and ULB. I am grateful for the discussions and good times I've had with the members of the Evolution, Complexity and Cognition Group and the Centrum Leo Apostel the AI Lab, IRIDIA, and CeNoLi. The ideas presented here were influenced especially by Ricardo Barbosa, Vassilios Basios, Bart De Vylder, Anselmo García Cantú Ros, Erden Göktepe, Carlos de la Mora, Marko Rodriguez, Francisco Santos, and Clément Vidal.

I appreciate the time, support, and advices given by Juan Julián Merelo Guervós and the GeNeura team at the Universidad de Granada. The five months I spent with them were crucial for the development of this research.

The possibility of a PhD does not come out of nowhere. This research would not have been possible without the formation and influences I received at Sussex University, especially from Inman Harvey, Ezequiel Di

Paolo, Chris Thornton, and Andy Clark; and as an undergraduate in Mexico, especially from Javier Fernández Pacheco, Jaime Lagunez Otero, Pedro Pablo González Pérez, and José Negrete Martínez.

I appreciate the discussions, advice, help, and inspiration I received from Maximino Aldana, Mark Bedau, Pamela Crenshaw, Kurt Dresner, Gastón Escobar, Jochen Fromm, Peter Furth, Nagarjuna G., Alexander Gorbulski, Stephen Guerin, David Hales, Dirk Helbing, Uri Hershberg, Bernardo Huberman, Genaro Juárez Martínez, Stuart Kauffman, Iavor Kostov, David Krakauer, Stefan Lämmer, Taivo Lints, Diana Mangalagiu, Peter McBurney, Mike McGurrin, Barry McMullin, Richard Michod, Luis Miramontes, Kai Nagel, Mikhail Prokopenko, Daniel Polani, Andreas Schadschneider, Frank Schweitzer, Cosma Shalizi, Ricard Solé, Sorin Solomon, Steven Strogatz, Hideaki Suzuki, Seth Tisue, Richard Watson, George Whitesides, Stephen Wolfram, Franco Zambonelli, Héctor Zenil, and many others to whom I owe an apology for my bad memory.

Several anonymous referees have helped with their comments, critics and suggestions to improve the ideas presented here.

I thank the minister of Mobiliteit en Openbare Werken of Brussels, Pascal Smet, Kenneth Vanhoenacker, and Philippe Boogaerts from the Ministerie van het Brussels Hoofdstedelijk Gewest for their time and for providing data for the Wetstraat simulations.

This research was partially $upported by the Consejo Nacional de Ciencia y Teconolgía (CONACyT) of Mexico and by the Fonds Wetenschappelijk Onderzoek (FWO) of Flanders.

Gracias a mi familia y amigos, por su apoyo y comprensión, tan lejos y tan cerca...

Надежда моя,
всё это—
невозможно без тебя.
Эта работа
тоже твоя...

Chapter 1

Introduction

"Everybody's Hive"
Carlos Gershenson
2006.
Acrylic on canvas, 65x80 cm.
Merelo Guervós collection.

Our world becomes more complex every day. To cope with it, the systems we design and control also need to become more complex, increasing the complexity of the world. This increase in complexity is characterized by a growth in number and diversity of elements of systems and their interactions. Every year there are more people, more computers, more devices, more cars, more medicines, more regulations, more problems. Since complexity has crawled into all the aspects of our lives, its study is very important for all disciplines. Traditional approaches are becoming obsolete, as they were developed for a simpler world. Thus any advancement in the general understanding of complex systems (Bar-Yam, 1997) will have a potential impact in sciences, engineering, and philosophy.

"Classical" approaches are still very useful, but only in problem domains that are stationary and comprehensible. An exact solution can be found, and this solution will hold. But with an increase of complexity, problem domains become non-stationary, requiring for dynamic solutions that will be able to *adapt* to the changes in the problem domain (Ashby, 1947a).

1.1 Motivation

"Simplicity is no reason, difficulty is no excuse"
—Bruce Edmonds

One conceptual approach to build adaptive systems involves designing the elements of a system to find *by themselves* the solution of the problem. Like this, when the problem changes, the elements are able to *dynamically* find a new solution. We can say that such a system *self-organizes*.

Even when the concept of self-organization (Ashby, 1947b; Heylighen, 2003b) is very promising to solve complex problems, and has been used for more than half a century, it remains somewhat vague, and it is not widespread. The aim of this work is to enhance our understanding of self-organization, and to exploit it to build systems that will be able to cope with complex problem domains. With the experience gained by building such systems, our understanding of them is also increased.

There is as yet no general methodology to design and control self-organizing systems. This book is a step towards developing one, providing new insights to build systems able to solve complex problems.

The role of a control mechanism in cybernetics (Wiener, 1948) is to regulate the variables of a system against perturbations within a viability zone. Thus, the control forces the system into a certain region of the

state space. For example, a thermostat controls the variable 'temperature' to be in a viability zone defined by the user. However, it becomes very complicated to steer the variables of a complex system due to inherent nonlinearities. *How can such systems be steered? And how to design systems that will be steered easily?* These are main questions for my research. There is no universal answer, but the aim is to increase our understanding of complex systems.

1.2 Contributions

The main contributions of this book are the following:

- Notions of complexity and self-organization are put forward (Sections 2.3 and 3.5). These generalize from definitions existing in the literature, aiming for synthesis. Thus, the notions try to be as broad as possible, without losing practicality.

- A general Methodology to aid the design and control of self-organizing systems is proposed. In principle, it could be applied in any domain to help people build and then steer complex systems (Chapter 4). The conceptual framework was developed independently of similar approaches. The stages of the Methodology were inspired in software engineering, but can be applied in other domains as well.

- Three control mechanisms of self-organizing traffic lights are proposed and analyzed with computer simulations. These are able to adapt to changing traffic conditions autonomously, considerably improving traffic flow over traditional control methods. (Chapter 5). A public software laboratory (SOTL, 2005) was developed to test the original methods. Their advantage lies not only in their performance, but also in their simplicity, making it possible to compare them easily with other methods.

- Using the Methodology, solutions are proposed for improving communication within bureaucracies, sensing public satisfaction, dynamic modification of hierarchies, and contextualization of procedures (Chapter 6). The benefits of self-organization in bureaucracies are exemplified with an original simple computational model, "random agent networks". Another public software laboratory was developed to study them .

- Properties required by protocols of self-organizing artifacts in an ambient intelligence scenario are discussed (Chapter 7). These were developed in collaboration with Francis Heylighen.

1.3 Outline

After this introductory chapter, Chapter 2 discusses concepts related to complexity, noting why they require a shift from classical thinking. Several examples are given, and a notion of complexity is put forward.

Chapter 3 discusses the concept of self-organization, proposing in Section 3.5 a practical notion that will be useful for the rest of the book. Before this, the inability to give a strict definition of self-organization is indicated, showing the partial dependence on the observer to judge whether a system is self-organizing or not.

Chapter 4 is the central part of the book, presenting a general methodology to design and control self-organizing systems. For this, a novel conceptual framework is introduced, and the different steps of the Methodology are described. Three case studies are presented to illustrate the Methodology (Chapters 5, 6, and 7). Since these are at different stages of development, some are able to illustrate the Methodology only partially.

In Chapter 5, Self-organizing traffic lights are presented. This project has developed concepts, abstract, and realistic simulations, almost ready to be deployed in a real city, making this the "less incomplete" case study. The simulations show that self-organizing methods outperform considerably traditional methods, even when the former ones use very simple rules and no direct communication. The models are interesting in themselves, as platoons are promoted by the local algorithms, which in certain abstract cases achieve "full synchronization", i.e. they do not stop at all.

The self-organizing bureaucracies project described in Chapter 6 is at an earlier stage, presenting concepts and an abstract simulation. Solutions are proposed for improving communication within bureaucracies, sensing public satisfaction, dynamic modification of hierarchies, and contextualization of procedures. A novel computational model—random agent networks—is presented, along with simulation results, to illustrate the benefits of self-organization in abstract organizations.

Chapter 7 presents self-organizing artifacts, focusing on communication protocols. This chapter contains only concepts, as simulations would only repeat results by de Jong (2000), which were developed in a slightly different context. The concepts presented illustrate a scenario where devices can learn to communicate, with whom to cooperate, and how to del-

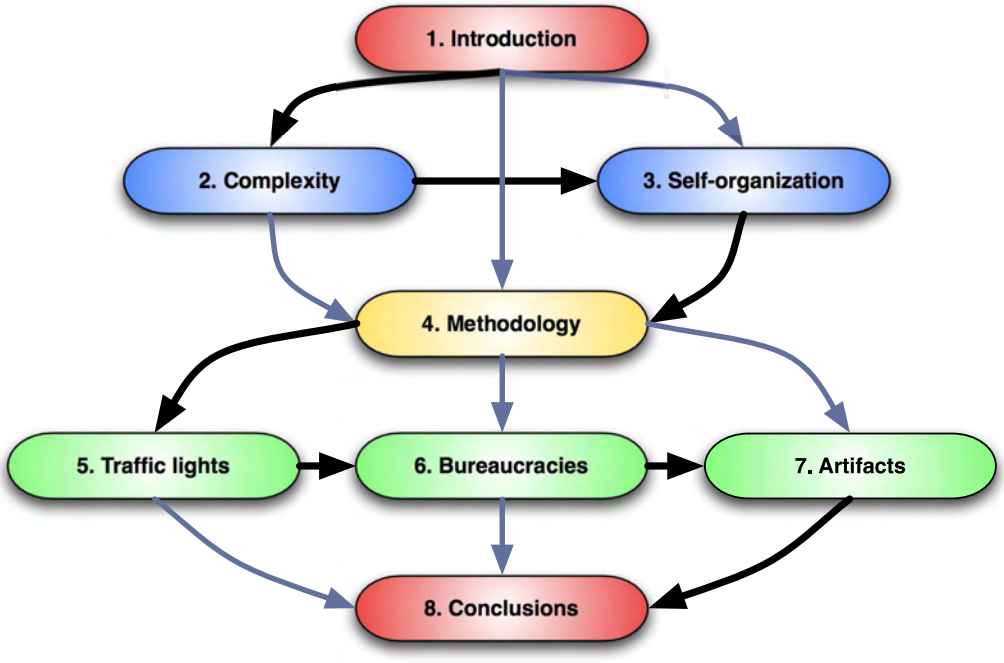


Figure 1.1: Book map. Thicker arrows indicate the suggested order of reading, while thinner arrows indicate alternative paths.

egate and coordinate specialized tasks.

Conclusions and future work are presented in Chapter 8, together with philosophical implications.

A map of the book can be appreciated in Figure 1.1.

1.3.1 How to Read this Book

This book is interdisciplinary, as it combines philosophy and engineering. Different chapters can be read in isolation, if the reader has only partial interest in the topics presented within, especially since each chapter is the outcome of previous papers. To facilitate reading, each chapter contains an abstract.

Chapters 2 and 3 are rather philosophical. Still, the concepts discussed in them are useful to understand the rest of the book. Chapter 4 also discusses philosophical concepts, but it is aimed mainly at engineers. Chapters 5, 6, and 7 present case studies to illustrate the Methodology, but they can be interesting for their own domain.

Throughout the book, different text styles were used to enhance readability. Examples use sans-serif font of pine green color and *notions and definitions use italic sans-serif font of orange red color*. The text is hyperlinked (references, figures, tables, equations, chapters, sections, etc.), making it easy to navigate through the electronic version of this document.

The Bibliography entries include references to the pages where they are cited. Also, URLs are provided for material that is available online. A short Glossary specifies the sense in which different terms are used across the book. An Index may be useful to quickly find relevant topics within the text.

1.3.2 How the Book Was Written

This book is a modified version of my doctoral dissertation (Gershenson, 2007a). It is the outcome of several years of research, which produced several articles. Some of these articles were taken as the basis for the chapters of the book. For reasons of presentation, the order of the Chapters does not coincide with the chronological order in which the ideas presented within were developed. Most of the contents of Chapters 7 and 5 were developed before Chapter 4, and actually the Methodology was abstracted from these case studies, which here are presented to illustrate it. In other words, these two case studies did not rely on the Methodology to be developed. Still, the Methodology is used in the book to explain *a posteriori* these case studies. Similarly, other systems could be developed without the Methodology. What I claim here is that the Methodology will *facilitate* their design by providing an appropriate *language* for the description of complex systems, as it did in the case study presented in Chapter 6.

Chapter 2 is based mainly on Gershenson and Heylighen (2005), but some of the ideas were already developed in Gershenson (2002b). The philosophical ideas of Chapter 3 were presented in Gershenson and Heylighen (2003). Chapter 4 is the central part of the book, presenting a Methodology to design and control self-organizing systems. It is based on Gershenson (2006a). Elements of Sections 2.3 and 3.5 are also introduced in that paper. The Methodology is also outlined in Gershenson (2007b). The major part of the work presented in Chapter 5 was published in Gershenson (2005), except for Sections 5.6 and 5.7. The results of these were obtained by Seung Bae Cools (2006), under the supervision of Bart D'Hooghe and myself, and are also presented in Cools, Gershenson, and D'Hooghe (2007). Chapter 6 is based on Gershenson (2006b) and Chapter 7 on Gershenson and Heylighen (2004).

The next chapter introduces the philosophical concepts that will be useful for understanding self-organization, the proposed Methodology, and the case studies.

Chapter 2

Complexity

This chapter[1] *proposes some basic concepts that help us to think and speak about complexity. We review classical thinking and its intrinsic drawbacks when dealing with complexity. We then show how indeterminacy and unpredictability are related to complexity. Finally, we argue that engineering complex systems will require the design of systems capable of adapting by themselves to unforeseen circumstances.*

[1] Based on Gershenson and Heylighen (2005); Gershenson (2002b, 2006a).

2.1 Introduction

In recent decades, the study of complex systems (Bar-Yam, 1997) has been transforming the way we view our world (Morin, 2006; Heylighen, Cilliers, and Gershenson, 2007). As with any scientific advancement, the development of theories and concepts related to complexity have also had implications for philosophy (Gershenson, 2002b; Gershenson, Aerts, and Edmonds, 2007)

In the next section, classical thinking and its drawbacks are reviewed. In Section 2.3, concepts related to complexity are presented, several examples are given, and a recursive notion of complexity is introduced. Sections 2.4 and 2.5 review problems that classical thinking faces when dealing with complexity, and Section 2.6 presents different ways in which systems can cope with complexity.

2.2 Classical Thinking

The majority of scientific models—as well as much of our intuitive understanding—implicitly rely on a "classical" or Cartesian mode of thinking, which is expressed most explicitly in the classical or Newtonian mechanics that dominated the scientific worldview until the beginning of the 20th century. It is based on the following assumptions (Heylighen, 1990a):

- *reductionism or analysis*: to fully understand a system you should decompose it into its constituent elements and their fundamental properties.

- *determinism*: every change can be represented as a trajectory of the system through (state) space, i.e., a sequence of states, following fixed laws of nature. These laws completely determine the trajectory towards the future (predictability) as well as towards the past (reversibility).

- *dualism*: the ultimate constituents of any system are particles, i.e., structureless pieces of matter (materialism). Since matter is already completely determined by mechanistic laws, leaving no freedom for intervention or interpretation, the only way we can include human agency in the theory is by introducing the independent category of mind.

- *correspondence theory of knowledge*: through observation, an agent can in principle gather complete knowledge about any system, creating an internal representation whose components correspond to the components of the external system. This establishes a single, true, objective mapping from the realm of matter (the system) to the realm of mind (the representation).

- *rationality*: given such complete knowledge, in its interaction with the system, an agent will always choose the option that maximizes its utility function. Thus, the actions of mind become as determined or predictable as the movements of matter.

These different assumptions are summarized by the *principle of distinction conservation* (Heylighen, 1989, 1990b): classical science begins by making as precise as possible distinctions between the different components, properties and states of the system under observation. These distinctions are assumed to be absolute and objective, i.e., the same for all observers. They follow the principles of Aristotelian logic: a phenomenon belongs either to category A, or to not A. It cannot be both, neither, in between, or "it depends"[2]. The evolution of the system conserves all of these distinctions, as distinct initial states are necessarily mapped onto distinct subsequent states, and vice versa (causality, see Heylighen (1989)). Knowledge is nothing more than another such distinction-conserving mapping from object to subject, while action is a mapping back from subject to object.

Certainly, we know that these assumptions represent ideal cases that are never realized in practice. Yet, most educated people still tend to assume that a complete and deterministic theory is an ideal worth striving for, and that the scientific method will lead us inexorably to an ever closer approximation of such objective knowledge. However, the lessons from complexity research point in a different direction (Morin, 2006).

2.3 Complexity

What is complexity? Let us go back to the Latin root *complexus*, which means "entwined" or "embraced". This can be interpreted in the following way: in order to have a complex you need: 1) two or more distinct parts, 2) that are joined in such a way that it is difficult to separate them. Here we find the basic duality between parts which are at the same time

[2]These principles can be neglected to comprehend paradoxes (Gershenson, 1998b, 1999).

distinct and connected. Therefore, the analytical method alone won't allow us to understand a complex, as by taking apart the components it will destroy their connections. The elements are mutually entangled, so that a change in one element will propagate through a tissue of interactions to other elements, which in turn will affect even further elements, including the one that initially started the process. This makes the global behavior of the system very hard to track in terms of its elements. Unlike the simple 'billiard-ball-like' systems studied by classical mechanics, complex systems are the rule rather than the exception. Typical examples are a living cell, a society, an economy, an ecosystem, the Internet, the weather, a brain, and a city. These all consist of numerous elements whose interactions produce a global behavior that cannot be reduced to the behavior of their separate components (Gershenson and Heylighen, 2005) .

Complexity is itself a complex concept, as we cannot make an unambiguous distinction between simple and complex systems. Many measures of complexity have been proposed for different contexts, such as computational, social, economic, biological, etc. (Edmonds, 2000). Even more, there is no general definition of *complexity*, since the concept achieves different meanings in different contexts (Edmonds, 1999). Still, we can say that a system is complex if it consists of several *interacting* elements (Simon, 1996), so that the behavior of the system will be difficult to deduce from the behavior of the parts. This occurs when there are many parts, and/or when there are many interactions between the parts. For example, a cell is considered a living system, but the elements that conform it are not alive. The properties of life arise from the complex dynamical *interactions* of the components. The properties of a system that are not present at the lower level (such as life), but are a product of the interactions of elements, are sometimes called *emergent* (Anderson, 1972). Another example can be seen with gold: it has properties, such as temperature, malleability, conductivity, and color, that emerge from the interactions of the gold atoms, since atoms do not have these properties.

Systems become more difficult to reduce and separate as the number and complexity of their interactions increases. Since the behavior of the system depends on the elements interactions, an integrative approach seems more promising, as opposed to a reductionist one. Like in Conway's Game of Life (Berlekamp et al., 1982, Ch. 25), it is necessary to "run the tape and see" to which state the dynamics will lead a system. As interactions increase, the state of each element becomes more dependent on the state of other elements, making it difficult to separate them. Taking a common example of deterministic chaos, in the logistic map there is only one variable considered, but the interaction with itself through nonlinear

feedback causes the well studied sensitivity to initial conditions and unpredictability in practice.

Even when there is no general definition or measure of complexity, a relative *notion* of complexity can be useful:

***Notion* 2.3.1** *The complexity of a system C_{sys} scales with the number of its elements $\#\overline{E}$, the number of interactions $\#\overline{I}$ between them, the complexities of the elements C_e, and the complexities of the interactions C_i (Gershenson, 2002b).*[3]

$$C_{sys} \sim \begin{cases} \#\overline{E} \\ \#\overline{I} \\ \sum_{j=0}^{\#\overline{E}} C_{e_j} \\ \sum_{k=0}^{\#\overline{I}} C_{i_k} \end{cases} \tag{2.1}$$

The complexity of an interaction C_i can be measured as the number of different possible interactions two elements can have.[4] For example, everything else being equal, a firm will be more complex than another one if it has more divisions, if its divisions have more employees, if the divisions have more channels of interaction, and/or if its channel of interactions involve more person-to-person interactions. Another example: the complexity of a cellular automaton (CA) (von Neumann, 1966; Wolfram, 1986) will be larger if it has more elements (more memory or space for computations) and/or more interactions (larger neighborhoods). Also, a Boolean CA will tend to have less complexity than a multi-valued one (each cell is more complex, since it can have more possible states); and the complexity of the rules (interactions) will certainly have an effect on the complexity of the CA.

The problem of a strict definition of complexity lies in the fact that there is no way of drawing a line between simple and complex systems independently of a context. For example, the *dynamics* of a system with a complex structure can be simple (ordered), complex, or chaotic. Cellular automata and random Boolean networks are a clear example of this, where moreover, the interactions of their components are quite simple. On the other

[3] This can be confirmed mathematically in certain systems. As a general example, random Boolean networks Kauffman (1969, 1993); Gershenson (2004b) show clearly that the complexity of the network increases with the number of elements and the number of interactions.

[4] Certainly, the number of possible interactions for certain elements is impossible to enumerate or measure.

hand, a *structurally* simple system can have complex and chaotic dynamics, such as the damped, driven pendulum.

Nevertheless, for practical purposes, the above notion will suffice, since it allows the comparison of the complexity of one system with another under a common frame of reference. Notice that the notion is recursive, so a basic level needs to be set contextually for comparing any two systems.

While we do not really need an absolute measure of complexity, using such relative notion may be useful to indicate when it becomes necessary to abandon our simple, classical assumptions and try to develop a more sophisticated model.

2.4 Indeterminacy

"Le certain n'est pas la vérite et l'incertain n'est pas l'ignorance"[5].
—Ilya Prigogine, 1997

Relinquishing classical thinking means giving up the principle of distinction conservation. This implies, first, that we can no longer assume given, invariant distinctions: a distinction made by one observer in one context may no longer be meaningful—or even possible—for another observer or in another context.

This point was made most forcefully in quantum mechanics: in some circumstances, an electron appears like a particle, in others like a wave (Heylighen, 1990a). Yet, according to classical thinking, particle and wave are mutually exclusive categories. In quantum mechanics, on the other hand, the "particle" and "wave" aspects are complementary: they are jointly necessary to characterize the electron, but they can never be seen together, since the observation set-up necessary to distinguish "particle-like" properties is incompatible with the one for "wave-like" properties. This was formulated by Heisenberg as the *principle of indeterminacy*: the more precisely we distinguish the particle-like properties, the more uncertain or indeterminate the wave-like properties become.

A more intuitive example of indeterminacy is the well-known ambiguous figure that sometimes looks like a rabbit, sometimes like a duck (see Figure 2.1). While both "gestalts" are equally recognizable in the drawing, our perception—like a quantum observation set-up—is incapable to see them simultaneously, and thus tends to switch back and forth between the two interpretations. Complementary properties, like the rabbit and

[5]What is certain is not truth and what is uncertain is not ignorance.

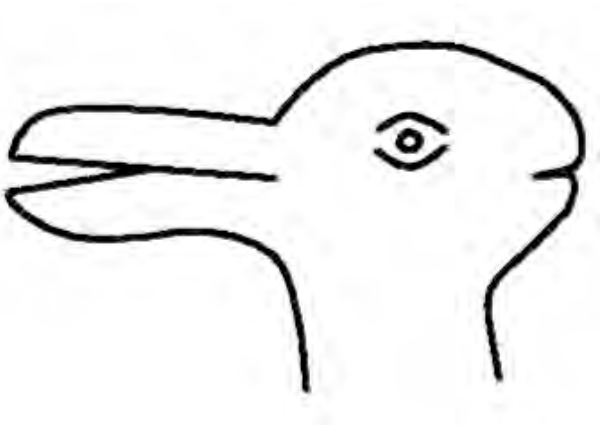


Figure 2.1: Is it a duck, a rabbit, or both?

duck gestalts, are distinct yet joined together. But while we see the one, we cannot see the other!

Because of the correspondence assumption, classical thinking tends to confuse what things are and how we see or know them to be. Thus, observers have engaged in controversies on "what things are", while actually disagreeing on how to model or represent these phenomena. When we speak about a phenomenon it is hard to specify whether we refer to the representation or to the represented, because our language does not make such a distinction, using the verb "to be" for both. To avoid such confusion I have proposed an ontological distinction between "absolute being" and "relative being" (Gershenson, 2002b). The absolute being (abs-being) refers to what the thing actually is, independently of the observer (similar to Kant's *Ding-an-sich*). The relative being (rel-being) refers to the properties of the thing as distinguished by an observer within a context. Since the observer is finite and cannot gather complete information, rel-beings are limited, whereas abs-beings have an unlimited number of features. Since new observers can contemplate any abs-being from new contexts, there exists an infinity of potential rel-beings for any abs-being. We can say that the rel-being is a model, while the abs-being is the modeled. Since we are all limited observers, it becomes clear that we can speak about reality only with rel-beings/models.

We can illustrate this abstract notion by imagining a sphere which is black on one hemisphere and white on the other, as depicted in Figure 2.2. Suppose we can observe the sphere only from one perspective. For some, the sphere will (rel)be white, for others it will (rel)be black, for others it will

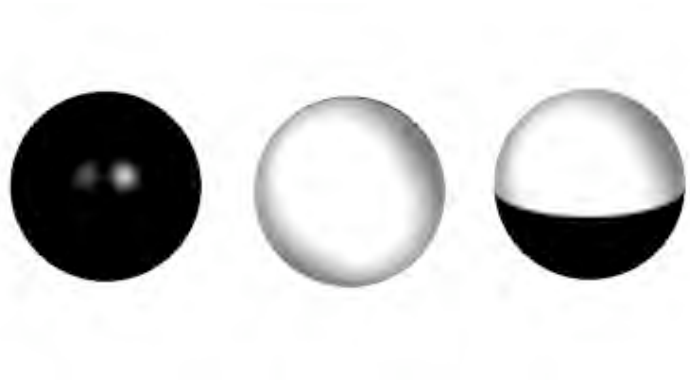


Figure 2.2: The same black-and-white sphere seen from three different angles.

(rel)be half black and half white, and so on. How can we decide which color the sphere (abs)is? Taking an average does not suffice, since it could be the case that more than ninety percent of people see the sphere white, and we would conclude that it is mostly white, while it actually (abs)is half white and half black. The best we can do is to indicate the perspective (context) for which the sphere (rel)is of a particular color. With real systems, we will never reach their abs-being, because there are always more properties (dimensions) than we can be aware of. This task would be like determining the color of an infinite-dimensional sphere when you can only see one two-dimensional projection at a time.

With simple systems such as the 3-dimensional sphere, the number of rel-beings is limited. However, complex systems have so many types of components and interactions that observers can have rel-beings that are so different that it may appear impossible to recognize them as aspects of the same thing. For example, people in cognitive science have debated on the "true" nature of cognition (symbolic, behavioral, neural...), when actually there are many different aspects of cognition that can be studied using different paradigms (Gershenson, 2004a). In another example, organizations have been described using metaphors such as an organism, a machine, a brain, a community, a market, and a political power game; and models such as a hierarchy, a network, and a linear input-output system. For a classically thinking executive, this constant shift in models and concomitant management styles is bewildering, as it seems that only one (or none) of these approaches can be correct. Yet, an organization has both mechanistic and organic aspects, is simultaneously a cooperative commu-

nity and a competitive arena, a rule-bound system and an open, creative environment, a hierarchy and a network.

There is no absolute, "best" model, as different rel-beings are appropriate for different contexts, and different purposes (Beer, 1966; Heylighen, 1990b). With a classical way of thinking, we can spend all our efforts in trying to decide what *is* the system. Complex thinking, on the other hand, allows us to contemplate different representations at the same time (e.g., by proposing a metarepresentation (Heylighen, 1990b)), in order to have a less-incomplete understanding of the system. To tackle concrete problems, we can then choose the representation that is most appropriate for that specific context, being well aware that a different problem may require a radical shift in representation. For example, if we are interested in the reasoning aspects of cognition, it might be appropriate to model it as a knowledge-based system. But if we are interested in the adaptive aspects of cognition, then a behavior-based system might be more appropriate. Or, when tackling internal conflicts it may be useful to see a firm as a network of interdependent communities; when optimizing production, as a matter- and information-processing mechanism.

Note that, even when we will not find "absolutely true" models, experience will give us a pragmatic feedback for deciding which ones are more useful *for a specific context*, i.e. we are not claiming that models are entirely subjective.

2.5 Nonlinearity and Chaos

"Invention, it must be humbly admitted, does not consist in creating out of void, but out of chaos" —Mary Shelley

According to classical thinking, distinctions are invariant not only over observers, but over time. The principle of determinist causality can be formulated as "equal causes have equal effects", or equivalently as "effects co-vary with their causes". This is nothing more than a statement that the distinctions between causes or initial states must necessarily carry through to their effects, and vice-versa. While we may hold this principle to be true at the level of absolute being, i.e., the complete things-in-themselves, it is in general not true at the level of relative being, i.e. the coarse, finite distinctions made by an observer. In other words, determinism does not in itself imply predictability. This can be inferred most directly from the existence of (deterministic) chaos, which follows from the nonlinearity that characterizes complex systems.

A system is linear if effects (outputs) are proportional to their causes (inputs). For example, if you put twice as much ore in your furnaces, the plant will produce roughly twice as much steel. This can be understood through the principle of conservation of energy and matter: the amount that comes out depends directly on the amount you put in (though there will of course be a few losses here and there). But what happens if (part of) the output is redirected and added back to the input? In principle, the next output will be larger, since it uses both the input and the previous output, and therefore no longer proportional to the input alone. The next output will be even larger, as it uses not only the new input but the two previous outputs. For example, a firm can reinvest some of the money it gets for its products to increase production. Increasing production brings in more money and thus further increases production, leading to an exponential growth in output.

Thus, nonlinearity can be understood as the effect of a causal loop, where effects or outputs are fed back into the causes or inputs of the process. Complex systems are characterized by networks of such causal loops. In a complex, the interdependencies are such that a component A will affect a component B, but B will in general also affect A, directly or indirectly. A single feedback loop can be positive or negative. A positive feedback will amplify any variation in A, making it grow exponentially. The result is that the tiniest, microscopic difference between initial states can grow into macroscopically observable distinctions.

This is called *sensitive dependence on initial conditions*, and is a defining feature of *chaos*. Because the initial difference is too small to perceive, the principle of causality cannot help us in predicting the final outcome. A well-known example of such a difficult-to-predict, chaotic system is the weather, as the fluttering of a butterfly in Brazil can grow into a hurricane devastating Texas. The observation that small causes can have large effects is obvious in social systems as well. For example, during a tense negotiation the tiniest hint of a smile on the lips of a CEO may create the impression with the other party that he should not to be trusted, thus leading them to harden their stance, and finally reject a billion-dollar merger operation. Such a system in a sense creates distinctions, as an indistinguishably small difference in initial states leads to macroscopically distinct outcomes.

The inverse of the amplifying effect of positive feedback is the dampening effect of negative feedback. Here any variation is counteracted or resisted, bringing the system back to its equilibrium state. As a result large causes (variations) may have little or no effect. For example, an entrenched culture in an organization can be very difficult to change, as new measures are actively or passively resisted, ignored or deflected. Such a

system destroys distinctions, as distinct causes lead to the same outcome.

Complex systems will typically exhibit a tangle of interconnected positive and negative feedback loops, where the effects of any change in a component cascade through an increasing number of connected components, in part feeding back, positively and/or negatively, into the initial component. If there is a variable time delay between these effects, it becomes in principle impossible to make predictions, because we do not know who will affect who first and thus whether an effect will be dampened before it has had the chance to get amplified or not (Gershenson, Broekaert, and Aerts, 2003). An example can be found in the stock exchange where stocks are bought and sold depending on their price, while the price is determined by how much is bought and sold. This intrinsic feedback loop has both negative and positive aspects. The law of supply and demand implies a negative feedback, since an increase in price normally reduces the demand, and this - after a variable delay - will reduce the price again. However, the parallel mechanism of speculation entails a positive feedback, as an increasing price makes buyers anticipate an even higher price in the future, thus enticing them to buy more of the stock now. The interaction between both nonlinear effects produces the chaotic movement of stock prices that are common in markets.

In the simpler situation where the delays are known (or can be neglected), it is sometimes possible to get at least a qualitative estimate of what can happen by identifying the signs (positive or negative) and the strengths of the different feedback loops in the network of influences. This method, which is often used to build computer simulations, is developed in the discipline of system dynamics (Sterman, 2000).

2.6 Adapting to Complexity

"Tendencies tend to change..."

Given the intrinsic unpredictability of complex systems, how can we design, build or generally deal with them? First we have to accept that we will never be able to control or predict their behavior completely. It is only natural that there will be surprises, errors and problems, as there have always been. However, we can always try to cope with unexpected events by *adapting* our actions to the new situation (Holland, 1975, 1995); if necessary, reconfiguring the system without destroying it. Different principles and methods for adaptation have been investigated in cybernetics (Heylighen and Joslyn, 2001), artificial intelligence (Russell

and Norvig, 1995), neural networks (Rumelhart et al., 1986), multi-agent systems (Wooldridge, 2002; Schweitzer, 2003), genetic algorithms (Mitchell, 1996), chaos control (Chen and Yu, 2003), and many other disciplines. Research is going on still, trying to design and build systems that are even more adaptive.

To adapt to any change, whether anticipated or not, it suffices to compensate for any perceived deviation of the actual situation from the desired course. This is the basic method of feedback control: correcting errors after the fact (Heylighen and Joslyn, 2001). If the reaction comes quickly enough, before the problem has had the chance to grow, feedback-based regulation can be extremely effective. The core innovation that engendered the field of cybernetics was the observation that it does not matter how complicated the system of factors and interactions that affect a variable that we wish to keep under control: as long as we have some means of counteracting the deviation, the underlying causality is irrelevant (Kelly, 1994). For example, it does not matter which complicated combination of social, political or technological changes causes an economy to overheat: in general, the central bank can regulate the rate of growth by increasing its interest rates.

Feedback control, however, still requires that we have a sufficiently broad repertoire of counteractions at our disposal (requisite variety), and that we know which action to execute in which circumstances (requisite knowledge) (Heylighen and Joslyn, 2001). The cybernetic *law of requisite variety* (Ashby, 1956) notes that the greater the variety of perturbations that the system may be subjected to, the larger the variety of actions it needs to remain in control. For example, the social brain hypothesis (Dunbar, 2003) states that the evolution of the complex human brain was promoted by the increasing complexity of its social environment. In other words, a complex (variable) brain is required to cope with a complex (variable) environment.

However, in order to react quickly and appropriately, it is good to have at least an expectation of what may happen and which reaction would be appropriate, i.e. to *anticipate* (Rosen, 1985). Expectations are subjective probabilities that we learn from experience: the more often circumstance B appears after circumstance A, or the more successful action B is in solving problem A, the stronger the association A $\rightarrow$ B becomes. The next time we encounter A (or a circumstance similar to A), we will be prepared, and more likely to react adequately. The simple ordering of options according to the probability that they would be relevant immensely decreases the complexity of decision-making (Kaelbling et al., 1996; Heylighen, 1994), since we would only need to pay attention to the most relevant circumstances. Thus, we do not need deterministic models or predictions: hav-

ing realistic default expectations with the possibility to correct for errors or exceptions after they have appeared works pretty well in practice. Moreover, we can tackle as yet unencountered combinations of circumstances by aggregating the recommendations made by different learned associations in proportion to their strength, e.g. using the method of spreading activation (Heylighen, 1999).

That is how our brains deal everyday with other complex systems: colleagues, children, pets, computers, etc. However, much to our dismay and frustration, most designed systems still lack this characteristic. For example, computers programmed according to rigid rules cannot recover on their own when something goes wrong (Heylighen and Gershenson, 2003).

To be able to adapt and anticipate, a system should also be *robust* (von Neumann, 1956; Jen, 2005). If a system is fragile, it will "break" (Ashby, 1947a) before it is able to counteract a perturbation. Thus, we can say that *a system is robust if it continues to function in the face of perturbations (Wagner, 2005)*. This can be achieved with modularity (Simon, 1996; Watson, 2002), degeneracy (Fernández and Solé, 2004), distributed robustness (Wagner, 2004), or redundancy (Gershenson, Kauffman, and Shmulevich, 2006).

Complex systems will combine adaptation, anticipation, and robustness to cope with their unpredictable environment. In the remaining chapters we will see how we can—as engineers—use self-organization to achieve this.

2.7 Conclusions

"Life is a constant adaptation"

We still do not understand complexity very well, and there is much to be done and explored in this direction. Our culture now is immersed and surrounded by complexity. But facing this complexity forces us to change our ways of thinking (Heylighen, 1991). This chapter argued how classical thinking, with its emphasis on analysis, predictability and objectivity, breaks down when confronted with complex systems. The core problem is that classical philosophy assumes invariant, conserved distinctions, whereas complex systems are entangled in such a way that their components and properties can no longer be separated or distinguished absolutely. Moreover, because of the inherent nonlinearity of the system, they tend to change in a chaotic, unpredictable way. At best, we can make

context-dependent distinctions and use them to build a partial model, useful for a particular purpose. But such model will never be able to capture all essential properties of the system, and a novel context will in general require a different model.

This chapter did not so much propose specific tools and techniques for dealing with complex systems, as this would require a much more extensive discussion. Moreover, introductions to and reviews of existing concepts are available elsewhere, e.g. Kelly (1994); Heylighen (1997, 2003b); Battram (2002). Instead we have brought forth a number of ideas that allow us to better understand and speak about complex systems that will be necessary in the following chapters. First, we must be aware that real complex systems are not completely predictable, even if we know how they function. We should be prepared to deal with the unexpected events that complexity most certainly will bring forth, by as quickly as possible correcting any deviation from our planned course of action. To achieve this kind of error-based regulation we should not try to predict or determine the behavior of a complex system, but to expect the most probable possibilities. This will make it easier for us to adapt when things go off-course. Because then we are ready to expect the unexpected.

In the next chapter, we discuss the concept of *self-organization*, which will be useful for designing and controlling complex systems able to cope with an unpredictable environment.

Chapter 3

Self-organization

This chapter[1] *presents a philosophical exploration of the conditions under which we can model a system as self-organizing. These involve the dynamics of entropy and the purpose, aspects, and description level chosen by an observer. We show how, changing the level or "graining" of description, the same system can appear self-organizing or self-disorganizing. We discuss ontological issues we face when studying self-organizing systems, and defend that self-organization is a way of observing systems, not an absolute class of systems. Aside from the philosophical debate, we present a practical notion of self-organization that will be useful in the next chapters for engineering self-organizing systems.*

[1] Based on Gershenson and Heylighen (2003); Gershenson (2006a).

3.1 Introduction

"It is as though a puzzle could be put together simply by shaking its pieces."
—Christian De Duve, Life Evolving, p. 22.

The term *self-organization* has been used in different areas with different meanings, as in cybernetics (von Foerster, 1960; Ashby, 1962; Heylighen and Joslyn, 2001), thermodynamics (Nicolis and Prigogine, 1977), biology (Camazine et al., 2003; Feltz et al., 2006), mathematics (Lendaris, 1964), computer science (Heylighen and Gershenson, 2003; Mamei et al., 2006; Kohonen, 2000), complexity (Schweitzer, 1997), information theory (Shalizi, 2001), evolution of language (de Boer, 1999; Steels, 2003), synergetics (Haken, 1981), and others (Skår and Coveney, 2003) (for a general overview, see (Heylighen, 2003b)). Many people use the term "self-organization", but it has no generally accepted meaning, as the abundance of definitions suggests. Also, proposing such a definition faces the philosophical problem of defining "self", the cybernetic problem of defining "system", and the universal problem of defining "organization". We will not attempt to propose yet another definition of self-organizing systems. Nevertheless, in order to try to understand these systems better, we will explore the following question: which are the conditions necessary to call a system "self-organizing"? We do so by combining insights from different contexts where self-organizing systems have been studied.

In the following section we explore the role of dynamics in self-organizing systems. We provide examples of systems that are self-organizing at one level but not at another one. In Section 3.3 we note the relevance of the observer for perceiving self-organization. We discuss some deeper conceptual problems for understanding self-organizing systems in Section 3.4. In Section 3.5 we present a practical notion to describe self-organizing systems that will be useful in the remainder of the book.

3.2 The Representation-Dependent Dynamics of Entropy

A property frequently used to characterize self-organization is an increase of order which is not imposed by an external agent (not excluding environmental interactions) (Heylighen, 2003b). The most common way to formalize the intuitive notion of "order" is to identify it with the negative of *entropy*. The second law of thermodynamics states that in an isolated

system, entropy can only increase, not decrease. Such systems evolve to their state of maximum entropy, or thermodynamic equilibrium. Therefore, physical self-organizing systems cannot be isolated: they require a constant input of matter or energy with low entropy, getting rid of the internally generated entropy through the output of heat ("dissipation"). This allows them to produce "dissipative structures" which maintain far from thermodynamic equilibrium (Nicolis and Prigogine, 1977). Life is a clear example of order far from thermodynamic equilibrium.

However, the thermodynamical concept of entropy as the dissipation of heat is not very useful if we want to understand information-based systems[2]. For that, we need the more general concept of *statistical entropy* (H) which is applicable to any system for which a state space can be defined. It expresses the degree of uncertainty we have about the state s of the system, in terms of the probability distribution $P(s)$.

$$H(P) = -\sum_{s \in S} P(s) \log P(s) \tag{3.1}$$

In this formulation, the second law of thermodynamics can be expressed as "every system tends to its most probable state" (Beer, 1966). This is in a sense a tautological law of nature, since the probabilities of the states are determined by us according to the tendencies of systems. At a molecular level, the most probable state of an *isolated* system is that of maximum entropy or thermodynamic equilibrium, where the molecules are distributed homogeneously, erasing any structure or differentiation. But does this apply as well to open complex systems embedded in dynamic environments?

We have to be aware that probabilities are relative to a level of observation, and that what is most probable at one level is not necessarily so at another. Moreover, a state is defined by an observer, being the conjunction of the values for all the variables or attributes that the observer considers relevant for the phenomenon being modeled. Therefore, we can have different degrees of order or "entropies" for different models or levels of observation of the same entity.

Let us illustrate this with the following, very simple example. Consider a system with four possible "microstates", a_1, a_2, b_1, and b_2, at the lowest, most detailed level of description. At the higher, more abstract level of description, we aggregate the microstates two by two, defining two macrostates: $A = \{a_1, a_2\}$ and $B = \{b_1, b_2\}$. This means that the system is

[2] Among other reasons—unlike matter and energy—information is not a conserved quantity, i.e. it can be created or destroyed

in macrostate A if it is either in microstate a_1 or in microstate a_2. The probabilities of the macrostates are simply the sum of the probabilities of their sub-states. Let us suppose that we start from an initial probability distribution P_1 so that $P(a_1) = P(b_1) = 0.1$ and $P(a_2) = P(b_2) = 0.4$. This implies $P(A) = P(B) = 0.5$. We can calculate the statistical entropy $H(P)$ using Equation 3.1, obtaining $H_l(P) \approx 1.72$ at the lower level, and $H_h(P) = 1$ at the higher level.

Now consider a second distribution P_2, $P(a_1) = P(a_2) = 0.2$ while $P(b_1) = P(b_2) = 0.3$. Therefore, $P(A) = 0.4$ and $P(B) = 0.6$. Now we have $H_l(P) \approx 1.97$ at the lower and $H_h(P) \approx 0.97$ at the higher level. Subtracting the initial H from the second, we have $\Delta H_l = H_l(P_2) - H_l(P_1) \approx 0.24$ at a lower level and $\Delta H_h = H_h(P_2) - H_h(P_1) \approx -0.029$ at the higher level. We have a change of distribution where entropy is *decreased* at the higher level ("self-organization"), and *increased* at the lower level ("self-disorganization"). To get the inverse change, we can just assume the final states to be the initial ones and vice versa. We would then have self-organization at the lower level and self-disorganization at the higher level.

This can be represented graphically in Figure 3.1, where tones of gray represent the probabilities of the states (darker color = lower value). The macrostates provide a coarse-grained (Hobbs, 1985) representation of the system, while the microstates provide a fine-grained one. We can visualize entropy as homogeneity of colors/probabilities. At the lower level, the distribution becomes more homogeneous, and entropy increases. At the higher level, the distribution becomes more differentiated.

Entropy not only depends on higher or lower levels of abstraction, but also on how we set the boundaries between states. Let us define alternative macrostates: A', which has a_1 and b_1 as sub-states, and B', with a_2 and b_2 as sub-states. Using the same values as above for the probabilities, we see that for the alternative macrostates the initial $H_{h'}(P) \approx 0.72$ and the final $H_{h'}(P) = 1$. So we have that $\Delta H_{h'} \approx 0.27$, which means that the statistical entropy increases in this macrorepresentation, while it decreases in the previous one. The system appears to be self-organizing or disorganizing, depending not only on the level at which we observe it, but also on *how* we do the "coarse-graining" of the system, that is to say, which variables we select to define the states.

The variables defined by the values (A, B), respectively (A', B'), represent two aspects of the same system, where the observer has focused on different, independent properties. For example, a particle's state includes both its position in space and its momentum or velocity. A subsystem is defined as a physical part of a system, limited to some of its components. Similarly, an *aspect system* can be defined as a *functional part of a system*,

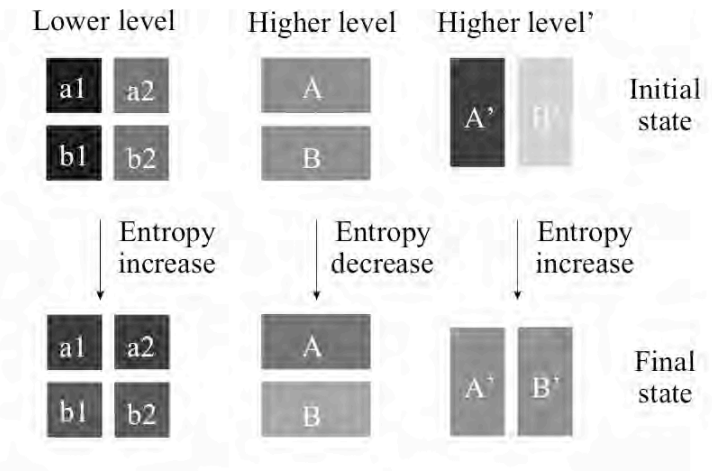


Figure 3.1: Entropy, seen as homogeneity, increases at lower level, while it might increase or decrease at the higher level, depending on how we divide the states.

limited to some of its properties or aspects (Ten Haaf et al., 2002, Ch. 3).

Let us illustrate this with swarming behavior. Groups of agents can be seen as subsystems of the swarm. The positions of all agents define one aspect system, while their velocities define another aspect system. Assume we start with non-moving agents scattered all over the simulated space. The position aspect is characterized by maximum entropy (agents can be anywhere in space), while the velocity aspect has minimum entropy (all have the same zero velocity). According to typical swarming rules, the agents will start to move with varying speeds towards the center of the swarm while mutually adjusting their velocities so as not to bump into each other. This means that their states become more concentrated in position space, but more diffuse in velocity space, until they converge to a stable configuration. In other words, while the swarm is in a transition phase, entropy decreases for the positions, while increasing for the velocities. Depending on the aspect we consider, the swarm self-organizes or self-disorganizes!

This example may appear too specific to support our argument. Let us therefore show that dynamical processes in general exhibit this kind of aspect-dependent or level-dependent behavior, either increasing or decreasing entropy. A dynamical system, where every state is mapped deterministically onto a single next state, can be seen as special case of a

Markov process, where a state s_i is mapped stochastically onto any number of other states s_j with a fixed transition probability $P(s_i \rightarrow s_j)$. To turn a deterministic dynamics into a stochastic one, it suffices to apply coarse-graining, aggregating a number of microstates into a single macrostate. Transitions from this macrostate can now go to any other macrostate that includes one of the microstates that were the initial destinations of its microstates.

It can be proved that the statistical entropy of a distribution cannot decrease if and only if the Markov process that describes the mapping from initial to final distribution is *doubly stochastic* (Koopman, 1978). This means that for the matrix of transition probabilities between states, the sum over a row and the sum over a column must be one. The sum over a row (probability of a given state ending in any state of the state space) is necessarily one, by definition of probability. However, the sum over a column is not a probability but can be seen as a measure of the "attractiveness" or "fitness" F of a state s_i:

$$F(s_i) = \sum_j P(s_j \rightarrow s_i) \tag{3.2}$$

High fitness ($F > 1$) of a state s means that on average more transitions enter s than leave s (the sum of all transition probabilities leaving s can never be >1). Thus, while the process runs, high fitness states become more probable, and low fitness states less probable, increasing their differentiation, as would be expected from a process undergoing self-organization.

A doubly stochastic process is defined by the requirement that $F(s_i) = 1, \forall s_i$. This corresponds to the entropy increasing processes studied in traditional thermodynamics. In a more general process, probability eventually concentrates in the high fitness states, decreasing overall entropy if the initial distribution is more homogeneous, increasing it if it is more "peaked". Therefore, a necessary and sufficient condition for a Markov process to allow self-organization is that it has a non-trivial fitness function, i.e. there exist states such that $F(s) \neq 1$. The sum of all probabilities must be 1. Therefore, if $\exists F(s) > 1 \Leftrightarrow \exists F(s') < 1$. This condition is equivalent to saying that the dynamics has a differential "preference" for certain states s over other states s'. Any dynamics that allows attractors has such an inbuilt preference for attractor states over basin states (Heylighen, 2003b). This is the most general case, and the one typically found in complex systems.

For any initial and final distribution of probabilities, such as the ones

in the examples we discussed, it is possible to determine a matrix of transition probabilities that maps the one onto the other. This matrix will in general not be doubly stochastic, and therefore allow self-organization as well as disorganization. Therefore, the same system, described in different aspects or levels of abstraction can be modeled as self-organizing in the one case, as self-disorganizing in another. Thus, it will depend partially on the observer, who decides on the granularity and aspects of the system to be observed, whether a system will be called self-organizing or not.

3.3 The Role of the Observer

"The question is not whether something is wrong with subjectivity. We are embedded in it, so we can only deal with it, or be blind and attempt to ignore it"

We have to be aware that even in mathematical and physical models of self-organizing systems, it is the *observer* who ascribes properties, aspects, states, and probabilities; and therefore entropy or order to the system. But organization is more than low entropy: it is structure that has a function or *purpose* (Heylighen and Gershenson, 2003). Stafford Beer (1966) noted a subtle but very important issue: what under some circumstances can be seen as organization, under others can be seen as disorder, depending on the purpose of the system. He illustrates this idea with the following example: When ice cream is taken from a freezer, and put at room temperature, we can say that the ice cream disorganizes, since it loses its purpose of having an icy consistency. But from a physical point of view, it becomes more ordered by achieving equilibrium with the room, as it had done with the freezer.[3] Again, the purpose of the system is not an objective property of the system, but something set by an *observer*.

W. Ross Ashby noted decades ago the importance of the role of the observer in relation to self-organizing systems:

> "A substantial part of the theory of organization will be concerned with *properties that are not intrinsic to the thing but are relational between observer and thing*" (Ashby, 1962, p. 258, emphasis in original).

Certainly there should be a correlate in the world to the observations. By generalizing the second law of thermodynamics, we can see that the

[3] Thermodynamic entropy can be seen as order or disorder in different situations (e.g. (Beer, 1966; Nicolis and Prigogine, 1977).

system through time will reach a more "probable" or "stable" configuration. We can say that it will reach an equilibrium or attractor[4]. The observer then needs to focus his/her viewpoint, in order to set the *purpose* of the system so that we can see the attractor as an "organized" state and to see it at the right *level* and *aspect*, and then self-organization will be observed. We can see that this is much more common than what intuition tells us. Not only lasers, magnets, Bénard rolls, ant colonies, or economies can be said to be self-organizing. Even an ideal gas can be said to be self-organizing, if we say (contrary to thermodynamics) that the equilibrium state where the gas homogeneously fills its container, is "ordered" (Beer, 1966; Nicolis and Prigogine, 1977). Most dynamical systems *can* be said to be self-organizing (Ashby, 1962). *Self-organization is a way of modeling systems, not a class of systems*. This does not mean that there is no self-organization independently of the observer, but rather that self-organization is everywhere.

Of course, not all systems are *usefully* described as self-organizing. Most natural systems can be easily fit into the class "self-organizing", unlike the simple mechanisms we find in physics textbooks. Most artificial systems are hard to see as self-organizing. Many are not dynamic, others involve only one element, and most of the rest follow sequences of rules that can be easily understood. Therefore there is no need to explain their functioning with the problematic concept of "self-organization".

We have said that any dynamical system, if observed "properly", can be seen as self-organizing. But if we set a different purpose or description level, then almost any dynamical system can be called self-disorganizing, i.e. if we decide to call the attractors of a system "disorganized". An economy will not be seen as self-organizing if we look only at a short timescale, or if we look at the scale of only one small business. An ant colony will not be self-organizing if we describe only the *global* behavior of the colony (e.g. as an *element* of an ecosystem), or if we only list the behaviors of individual ants. We have to remember that the description of self-organization is partially, but strongly, dependent on the observer.

3.4 Ontological Issues

"Objects do not depend on the concepts we have of them"

[4] In some chaotic systems, this can take practically infinite time. But as systems approach an attractor, we can say that they follow this principle. Also, the state space must be finite.

One of the most common problems when discussing self-organizing systems is the meaning of emergence. Self-organizing systems typically have higher level properties that cannot be observed at the level of the elements, and that can be seen as a product of their interactions (more than the sum of the parts). Some people call these properties *emergent*. The problem we face is ontological. According to classical thought, there is only one "true" description of reality. In this case, a system cannot be at the same time a set of elements and a whole with emergent properties. But by introducing the ontological distinction between "absolute being" and "relative being" (Gershenson, 2002b)(discussed in Section 2.4), we can clarify the issue.

We can observe any *abs*-being from an infinity of perspectives and describe an infinity of potential properties or aspects. Nevertheless, most *rel*-beings and contexts are similar, since they are inspired by the same *abs*-being seen by similar observers from a similar point of view. This enables us to share knowledge, but it is because of the different nuances in the different *rel*-beings and contexts that we fail to agree in every situation.

We can then say that the observation of a system at different abstraction levels or in different aspects is merely a difference in the perspective, and therefore the system *rel*-is different (only for the observers). But the system *abs*-is the same thing itself, independently of how we describe it. We can observe a cell as *rel*-being a bunch of molecules or as *rel*-being a living structure. But it *abs*-is both and even more. *Rel*-beings can be seen as different models or metaphors for describing the same thing. A change in the metaphor does not change the thing. If we define emergence as a process that requires a change of the model (Rosen, 1985) in order to better understand and predict the system (Shalizi, 2001), then it becomes clear that there is no magic. Any dynamical system *abs*-is self-organizing and self-disorganizing at the same time, in the sense that it can potentially *rel*-be both.

Another confusion may arise when people describe systems as the lower level *causing* change in the emergent properties of the same system. Vice versa, downward causation is the idea that higher level properties constrain or control components at the lower level (Campbell, 1974). Speaking about causality between abstraction levels is not accurate (Gershenson, 2002b), because actually they *abs*-are the same thing. What we could say is that when we observe certain conditions in the lower level, we can expect to observe certain properties at a higher level, and vice versa. There is correlation, but not actual causation.

This leads us to what is probably the most fundamental problem. If we can describe a system using different levels, aspects, or representations,

which is the one we should choose? As Prem (1993) suggests, the level should be the one where the prediction of the behavior of the system is easiest; in other words, where we need least information to make predictions[5] (Shalizi, 2001). Certainly, this will also depend on which aspects of a system we are interested in predicting. We can speculate that this prediction is possible because of regularities in the system at that particular level, and that this is what leads people to try to describe how the simple properties cause the not so simple ones, either upward or downward.

We should note that neither objective nor subjective descriptions alone are sufficient to describe self-organizing systems. This is because self-organization is described by an observer embedded within a context, but it needs to be contrasted with an objective system. We can call this approach a *contextual* description (Gershenson, 2002c; Edmonds, 2001), since the same description of an object can be accurate in some contexts and not in others.

3.5 Self-organization: A Practical Notion

"Natural processes should be judged different from mechanical ones because they are self-organizing" —Immanuel Kant

In the previous sections we saw that it is problematic to give a precise definition of a self-organizing system, since any dynamical system can be said to be self-organizing or not, depending partly on the observer (Gershenson and Heylighen, 2003; Ashby, 1962): If we decide to call a "preferred" state or set of states (i.e. attractor) of a system "organized", then the dynamics will lead to a self-organization of the system. We face similar circumstances with the definitions of cognition (Gershenson, 2004a), intelligence (Gershenson, 2001), life (Krakauer and Zanotto, 2006), and complexity (Gershenson, 2002b): there is no sharp boundary to distinguish systems that belong to the category and those which do not and they are partially determined by the observer describing the system and its purpose.

However, it is not necessary to enter into a philosophical debate on the theoretical aspects of self-organization to work with it, so a practical notion will suffice for the purposes of this book:

***Notion* 3.5.1** *A system* described *as self-organizing is one in which elements* interact *in order to achieve* dynamically *a global function or behavior. (Gershenson, 2006a)*

[5] This argument could be also followed to decide which "graining" to choose.

This function or behavior is not imposed by one single or a few elements, nor determined hierarchically. It is achieved *autonomously* as the elements interact with one another. These interactions produce feedbacks that regulate the system. All the examples of complex systems mentioned in Chapter 2 fulfill this notion of self-organization. More precisely, instead of searching for the necessary conditions for a self-organizing system—since any system can fulfill the most general conditions—the question can be formulated as follows: *when is it useful to describe a system as self-organizing?* This will be when the system or environment is very dynamic and/or unpredictable. If we want the system to solve a problem, it is useful to describe a complex system as self-organizing when the "solution" is not known beforehand and/or is changing constantly. Then, the solution is dynamically striven for by the elements of the system. In this way, systems can adapt quickly to unforeseen changes as elements interact locally. In theory, a centralized approach could also solve the problem, but *in practice* such an approach may require too much time to compute the solution and would not be able to keep the pace with the *unpredictable* changes in the system and its environment. This would occur when the system or its environment changes in less time than the one required to calculate a solution. For example, if it takes one week with a supercomputer to calculate the solution of a problem, and this problem changes once a year, a centralized approach can suffice. But if the problem changes every day, by the time a solution is found, i.e. after one week, this might not be the best one for the actual problem.

3.5.1 Artificial self-organizing systems

In engineering, *a self-organizing system would be one in which elements are designed to* dynamically *and* autonomously *solve a problem or perform a function at the system level*. In other words, the engineer will not build a system to perform a function explicitly, but elements will be engineered in such a way that their behaviour and interactions will lead to the system function. Thus, the elements need to divide, but also to integrate, the problem. This description will be *useful* when the function of the system is not easily *reducible* to the function of its elements. For example, the parts of a car are designed to perform a function at the system level: to drive. However, the parts of a (normal) car do not change their behavior in time, so it might be redundant to call a car self-organizing. On the other hand, a swarm of robots (Dorigo et al., 2004) will be conveniently described as self-organizing, since each element of the swarm can change its behavior

depending on the current situation. It should be noted that all engineered self-organizing systems are to a certain degree *autonomous*, since at least a part of their actual behavior will not be determined by a designer.

Certainly self-organization is not the only approach for designing and controlling systems, and in many cases it is not appropriate. But it can be very useful in complex systems where the observer cannot *a priori* conceive of all possible configurations, purposes, or problems that the system may be confronted with. Examples of these are organizations (corporations, governments, communities), traffic control, proteomics, distributed robotics, allocation of ecologic resources, self-assembling nanotubes, and complex software systems (Heylighen and Gershenson, 2003), such as the Internet.

3.5.2 Levels of abstraction

In order to understand self-organizing systems, two or more *levels of abstraction* (Gershenson, 2002b) should be considered: elements (lower level) organize in a system (higher level), which can in turn organize with other systems to form a larger system (even higher level). The understanding of the system's behavior will come from the relations observed between the descriptions at different levels. Note that the levels, and therefore also the terminology, can change according to the interests of the observer. For example, in some circumstances, it might be useful to refer to cells as elements (e.g. bacterial colonies); in others, as systems (e.g. genetic regulation); and in others still, as systems coordinating with other systems (e.g. morphogenesis).

3.5.3 Coping with the unknown

As we saw in Section 2.6, a complex system can cope with an unpredictable environment *autonomously* using different but closely related approaches:

- **Adaptation** (learning, evolution) (Holland, 1995). The system changes to cope with the change.

- **Anticipation** (cognition) (Rosen, 1985). The system predicts a change to cope with, and adjusts accordingly. This is a special case of adaptation, where the system does not require to experience a situation before responding to it.

- **Robustness** (von Neumann, 1956; Jen, 2005). A system is robust if it continues to function in the face of perturbations (Wagner, 2005). This can be achieved with modularity (Simon, 1996; Watson, 2002), degeneracy (Fernández and Solé, 2004), distributed robustness (Wagner, 2004), or redundancy (Gershenson, Kauffman, and Shmulevich, 2006). Modules can be useful to prevent the propagation of damage in a system. Degeneracy is the ability of elements that are structurally different to perform the same function. Distributed robustness occurs when the function of the system is performed by different components, so that they can compensate effects of perturbations. Redundancy is given when there are several copies of a type of element, so that if one fails, others can take its role.

Successful self-organizing systems will use combinations of these approaches to maintain their integrity in a changing and unexpected environment. Adaptation will enable the system to modify itself to "fit" better within the environment. Robustness will allow the system to withstand perturbations without losing its function or purpose, and thus giving it the possibility to adapt. Anticipation will prepare the system for changes before these occur, adapting the system without it being perturbed. Adaptation and anticipation are *active*, in the sense that they produce changes in the system. This is not achieved by robustness, which is rather *passive*. Using the cybernetics terminology (Heylighen and Joslyn, 2001), adaptation is analogous to feedback control, anticipation to feedforward control, and robustness to buffering. We can see that all of them should be taken into account while engineering self-organizing systems.

3.6 Conclusions

This chapter showed that self-organizing systems, rather than a *type* of systems, are a *perspective* for studying, understanding, designing, controlling, and building systems. This perspective has advantages and disadvantages, and there are systems that benefit from this approach, and others for which it is redundant. But even in the general case when the systems dynamics allows self-organization in the sense of entropy decrease, a crucial factor is the *observer*, who has to describe the process at an appropriate *level(s)* and *aspects*, and to define the *purpose* of the system. All these "make" the system to be self-organizing. In that sense, self-organization can be everywhere: it just needs to be observed.

Independently of the philosophical discussion, a practical notion of self-organization was introduced, a perspective which will be used in the rest of the book to design systems able to cope with an unpredictable environment.

Chapter 4

A General Methodology

This chapter[1] *presents a conceptual framework for speaking about self-organizing systems. The aim is to provide a methodology useful for designing and controlling systems developed to solve complex problems. Starting from the agent metaphor, a novel conceptual framework is presented. This provides formal ways of speaking about "satisfaction" of elements and systems. The main premise of the methodology claims that reducing the "friction" of* interactions *between elements of a system will result in a higher "satisfaction" of the system, i.e. better performance. Different ways in which this can be achieved are discussed.*

[1] Based on Gershenson (2006a).

4.1 Introduction

"Expect the unexpected"

Over the last half a century, much research in different areas has employed self-organizing systems to solve complex problems, e.g. Ashby (1956); Beer (1966); Bonabeau et al. (1999); Di Marzo Serugendo et al. (2004); Zambonelli and Rana (2005). Recently, particular methodologies using the concepts of self-organization have been proposed in different areas, such as software engineering (Wooldridge et al., 2000; Zambonelli et al., 2003), electrical engineering (Ramamoorthy et al., 1993), and collaborative support (Jones et al., 1994). However, there is as yet no general framework for constructing self-organizing systems. Different vocabularies are used in different areas, and with different goals. In this chapter, I develop a general methodology useful for designing and controlling *complex* systems (Bar-Yam, 1997). The proposed methodology, as with any methodology, does not provide ready-made solutions to problems. Rather, it provides a *conceptual framework*, a *language,* to assist the solution of problems. Also, many current problem solutions can be *described* as proposed. I am not suggesting new solutions, but an alternative way of thinking about them.

As an example, many standardization efforts have been advanced in recent years, such as ontologies required for the Semantic Web (Berners-Lee et al., 2001), or FIPA standards. I am not insinuating that standards are not necessary. Without them, engineering would be chaos. But as they are now, they cannot predict future requirements. They are developed with a static frame of mind. They are not adaptive. What this work suggests is a way of introducing the expectation of change into the development process. The intention of this is to be able to cope with the unexpected beforehand, in problem domains where this is desired.

In the next section, concepts necessary to implement the Methodology are introduced. The Methodology itself is presented in Section 4.3, followed by Discussion and Conclusions.

4.2 The Conceptual Framework

"Nothing is free of its own limits"

By "conceptual framework", I mean a set of concepts that should be useful to *describe* self-organizing systems. The usefulness of the concepts

presented in this section is shown in the case studies presented in the following chapters, but certainly this does not imply that the Methodology will be useful universally.

Elements of a complex system interact with each other. The actions of one element therefore affect other elements, directly or indirectly. For example, an animal can kill another animal directly, or indirectly cause its starvation by consuming its resources. These interactions can have negative, neutral, or positive effects on the system (Heylighen and Campbell, 1995).

Now, intuitively thinking, it may be that the "smoothing" of local interactions, i.e. the minimization of "interferences" or "friction" will lead to global improvement (Helbing and Vicsek, 1999). But is this always the case? To answer this question, the terminology of multi-agent systems (Maes, 1994; Wooldridge and Jennings, 1995; Wooldridge, 2002; Schweitzer, 2003) can be used. We can say that:

***Notion* 4.2.1** *An agent is a description of an entity that* acts *on its environment.*

Examples of this can be a trader acting on a market, a school of fish acting on a coral reef, or a computer acting on a network. Thus, every element, and every system, can be seen as agents with *goals* and behaviors aiming to reach those goals. The behavior of agents can affect (positively, negatively, or neutrally) the fulfillment of the goals of other agents, thereby establishing a relation. The *satisfaction* or fulfillment of the goals of an agent can be represented using a variable $\sigma \in [0, 1]$.[2] Relating this to the higher level, the satisfaction of a system σ_{sys} can be recursively represented as a function $f : \mathbb{R}^{2n+1} \rightarrow [0, 1]$ of the satisfaction σ_i of the n elements constituting it:

$$\sigma_{sys} = f\left(\sigma_1, \sigma_2, ..., \sigma_n, w_0, w_1, w_2, ..., w_n\right) \tag{4.1}$$

where w_0 is a bias and the other weights determine the importance given to each σ_i. If the system is homogeneous and the components have linear interactions, then f will be the weighted sum of σ_i, $w_i = \frac{1}{n} \forall i \neq 0$, $w_0 = 0$. Note that this would be very similar to the activation function used in many artificial neural networks (Rojas, 1996). For heterogeneous systems, f may be a nonlinear function. Nevertheless, the weights w_i's are determined

[2] In some cases, σ could be seen as a "fitness" (Heylighen and Campbell, 1995). However, in most genetic algorithms (Mitchell, 1996) a fitness function is imposed from the outside, whereas σ is a property of the agents, that can change with time.

tautologically by the importance of the σ_i of each element to the satisfaction of the system σ_{sys}. If several elements decrease σ_{sys} as they increase their σ_i, we would not consider them as part of the system. It is important to note that this is independent of the potential nonlinearity of f. An example can be seen with the immune system. It categorizes molecules and micro-organisms as akin or alien (Vaz and Varela, 1978). If they are considered as alien, they are attacked. Auto-immune diseases arise when this categorization is erroneous, and the immune system attacks vital elements of the organism. On the other hand, if pathogens are considered as part of the body, they are not attacked. Another example is provided by cancer. Carcinogenic cells can be seen as "rebel", and no longer part of the body, since their goals differ from the goal of the organism. Healthy cells are described easily as part of an organism. But when they turn carcinogenic, they can better be described as parasitic. The tautology is also useful because it gives a general mathematical representation for system satisfaction, which is independent of a particular system.

A reductionist approach would assume that maximizing the satisfaction of the elements of a system would also maximize the satisfaction of the system. However, this is not always the case, since some elements can "take advantage" of other elements, for example in the prisoner's dilemma (Axelrod, 1984). Thus, we need to concentrate *also* on the interactions of the elements.

If the model of a system considers more than two levels, then the σ of higher levels will be recursively determined by the σ's of lower levels. However, the f's most probably will be very different on each level.

Certainly, an important question remains: how do we determine the function f and the weights w_i's? To this question there is no complete answer. One option would be to approximate f numerically (De Wolf et al., 2005). An explicit f may be difficult to find, but an approximation can be very useful. Another method consists of *lesioning* the system[3]: removing or altering elements of the system, and observing the effect on σ_{sys}. Through analyzing the effects of different lesions, the function f can be reconstructed and the weights w_i's obtained. If a small change $\Delta\sigma_i$ in any σ_i produces a change $|\Delta\sigma_{sys}| \geq |\Delta\sigma_i|$, the system can be said to be *fragile*.

What could then be done to maximize σ_{sys}? How can we relate the σ_i's and avoid conflicts between elements? This is not an obvious task, for it implies bounding the agents' behaviors that reduce other σ_i's, while preserving their functionality. Not only should the interference or friction

[3]This method has been used widely to detect functions in complex systems such as genetic regulatory networks and nervous systems, e.g. Beer (1990).

between elements be minimized, but the synergy (Haken, 1981; Corning, 2003) or "positive interference" should also be promoted. Dealing with complex systems, it is not feasible to tell each element what to do or how to do it, but their behaviors need to be constrained or modified so that their goals will be reached, blocking the goals of other elements as little as possible. These constraints can be called *mediators* (Michod, 2003; Heylighen, 2003a). They can be imposed from the top down, developed from the bottom up, be part of the environment, or be embedded as an *aspect* (Ten Haaf et al., 2002, Ch. 3) of the system. Mediators are determined by an *observer*, and can be internal or external to the system (depending on where the observer sets the boundaries of the system). An example can be found in city traffic: traffic lights, signals and rules mediate among drivers, trying to minimize their conflicts, which result from the competition for limited resources, i.e. space to drive through. The precise rules and conventions may change from country to country, e.g. side of the street to drive or behavior at intersections. Nevertheless, they are successful as long as everybody adheres to them. Another example of a mediator can be seen with crowd dynamics: columns near exits of crowded areas help mediate between people and facilitate their departure, reducing the probability of accidents caused by panicking crowds (Escobar and De La Rosa, 2003). The notion of mediator can be seen as a generalization of "slaving constraints" (Haken, 1988).

***Notion* 4.2.2** *A mediator arbitrates among the elements of a system, to minimize conflict, interferences and frictions; and to maximize cooperation and synergy.*

Therefore, the efficiency of the mediator can be measured directly using σ_{sys}. Individually, we can measure the "friction" $\phi_i \in [-1, 1]$ that agent i causes in the rest of the system, relating the change in satisfaction $\Delta\sigma_i$ of element i and the change in satisfaction of the system $\Delta\sigma_{sys}$:

$$\phi_i = \frac{-\Delta\sigma_i - \Delta\sigma_{sys}\,(n-1)}{n}. \tag{4.2}$$

Friction occurs when the increase of satisfaction of one element causes a decrease in the satisfaction of some other elements that is greater than the increase. Decreasing friction will lead to higher σ_{sys} because that is precisely how friction is defined in Equation 4.2, that is, as the difference in change of satisfaction of an element proportional to the change of satisfaction of the system. If $\phi_i > 0$, it is because there was a noticeable decrease in σ_{sys}, or a disproportional decrease in σ_i. If $\phi_i < 0$, then most probably it

is due to an increase of σ_{sys}, or at least a noticeable in increase in σ_i with a negligible cost to the system. Note that $\phi_i = 0$ does not imply an absence of conflict, since one agent can "get" the satisfaction proportionally to the "loss" of satisfaction of (an)other agent(s). Negative friction would imply *synergy*, e.g. when $\Delta\sigma_i \geq 0$ while other elements also increase their σ. The role of a *mediator* would be to maximize σ_{sys} by minimizing ϕ_i's. With this approach, friction can be seen as a type of *interaction* between elements. [4]

Thus, the problem can be put in a different way: how can we find / develop / evolve efficient mediators for a given system? One answer to this question is this Methodology. The answer will not be complete, since we cannot have in practice precise knowledge of f for large evolving complex systems. This is because the evolution of the system may change its own f (Kauffman, 2000), and the relationships among different σ_i's. Therefore, predictions cannot be complete. However, the Methodology proposes to follow steps to increase the understanding (and potentially the control) of the system and the relations between its elements. The goal is to identify conflicts and diminish them without creating new ones. This will increase the σ_i's and thus σ_{sys}. The precision of f is not so relevant if this is achieved.

It should be noted that the *timescale* chosen for measuring $\Delta\sigma_i$ is very important, since at short timescales the satisfaction can decrease, while on the long run it will increase. In other words, there can be a short term "sacrifice" to harvest a long term "reward". If the timescale is too small, a system might get stuck in a "local optimum", since all possible actions would decrease its satisfaction on the short term. But in some cases the long term benefit should be considered for maximization. This will also depend on the timescale at which the problem to be solved evolves, i.e. how fast the problem domain changes. A way of measuring the slow change of σ_i would be with its integral over time for a certain interval Δt:

$$\int_{t}^{t+\Delta t} \sigma_i dt. \tag{4.3}$$

Another way of dealing with local optima is to use neutral changes to explore alternative solutions (Kimura, 1983). Neutral changes are those that do not affect the performance (or "fitness") of a system (σ_{sys}), but they allow the exploration of alternative solutions and escaping local optima.

Before going into further detail, it is worth noting that this is not a reductionist approach. Smoothing out local interactions will not provide

[4] Using the terminology of game theory, interactions can be seen as games, satisfactions as payoffs, friction as negative-sums, and synergy as positive-sums.

straightforward clues as to what will occur at the higher level. Therefore, the system should be observed at both levels: making local and global changes, observing local and global behaviors, and analyzing how one affects the other.

Concurrently, the *dependence* $\epsilon \in [-1, 1]$ of an element to the system can be measured by calculating the difference of the satisfaction σ_i when the element interacts within the system and its satisfaction $\widetilde{\sigma}_i$ when the element is isolated.

$$\epsilon = \sigma_i - \widetilde{\sigma}_i. \tag{4.4}$$

In this way, full dependence is given when the satisfaction of the element within the system σ_i is maximal and its satisfaction $\widetilde{\sigma}_i$ is minimal when the element is isolated. A negative ϵ would imply that the element would be more satisfied on its own and is actually "enslaved" by the system. Now we can use the dependencies of the elements to a system to measure the *integration* $\tau \in [-1, 1]$ of a system, which can be seen also as a gradual measure of a meta-system transition (MST) (Turchin, 1977).

$$\tau = \frac{1}{n} \sum_{i=1}^{n} \epsilon_i. \tag{4.5}$$

A MST is a gradual process, but it will be complete when elements are not able to reach their goals on their own, i.e. $\widetilde{\sigma}_i \to 0$. Examples include cells in multi-cellular organisms and mitochondria in eukaryotes.

In an evolutionary process, natural (multilevel (Michod, 1997; Lenaerts, 2003)) selection will tend to increase τ because this implies higher satisfaction both for the system and its elements (systems with a negative τ are not viable). Relations and mediators that contribute to this process will be selected, since higher σ's imply more chances of survival and reproduction. Human designers and engineers also select relations and mediators that increase the σ's of elements and systems. Therefore, we can see that evolution will tend, in the long run, towards synergistic relationships (Corning, 2003), even if resources are scarce. Or, alternatively, returning to the tautology mentioned above, observers characterize systems that have evolved with a high τ. Still, we can see that this tautology will be useful to describe artificial systems that fulfill our expectations as designers by having a high τ.

In the next section, the steps suggested for developing a self-organizing system are presented, using the concepts described in this section.

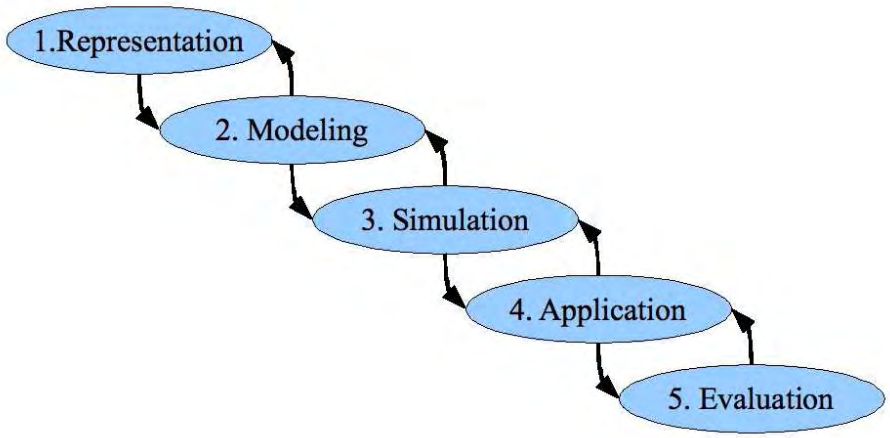


Figure 4.1: Diagram relating different stages of Methodology.

4.3 The Methodology

The proposed Methodology receives the requirements of a system, i.e. what the system should do, and enables the designer to produce a system that fulfills the requirements. The methodology includes the following steps: Representation, Modeling, Simulation, Application, and Evaluation, which will be exposed in the following subsections. Figure 4.1 presents these steps. These steps should not necessarily be followed one by one, since the stages merge with each other. There is also backtracking, when the designer needs to return to an earlier stage for reconsideration before finishing a cycle/iteration.

This methodology should not be seen as a recipe that provides ready-made solutions, but rather as a guideline to direct the search for them. The stages proposed are not new, and similar to those proposed by iterative and incremental development methodologies. Still, it should be noted that the active feedback between stages within each iteration can help in the design of systems ready to face uncertainties in complex problem domains. The novelty of the methodology lies in the *vocabulary* used to describe self-organizing systems.

4.3.1 Representation

The goal of this step is to develop a *specification* (which might be tentative) of the components of the system.

The designer should always remember the distinction between model and modeled. A model is an abstraction/description of a "real" system. Still, there can be several descriptions of the same system (Gershenson, 2002b; Gershenson and Heylighen, 2005), and we cannot say that one is better than another independently of a context.

There are many possible representations of a system. According to the *constraints* and *requirements*, which may be incomplete, the designer should choose an appropriate vocabulary (metaphors to speak about the system), abstraction levels, granularity, variables, and interactions that need to be taken into account. Certainly, these will also depend on the experience of the designer. The choice between different approaches can depend more on the expertise of the designer than on the benefits of the approaches.

Even when there is a wide diversity of possible systems, a general approach for developing a Representation can be abstracted. The designer should try to divide a system into elements by identifying semi-independent modules, with internal goals and dynamics, and with few interactions with their environment. Since interactions in a model will increase the complexity of the model, we should group "clusters" of interacting variables into elements, and then study a minimal number of interactions between elements. The first constraints that help us are space and time. It is useful to group variables that are close to each other (i.e. interacting constantly) and consider them as elements that relate to other elements in occasional interactions. Multiscale analysis (Bar-Yam, 2005) is a promising method for identifying levels and variables useful in a Representation. Since the proposed methodology considers elements as agents, another useful criterion for delimiting them is the identification of goals. These will be useful in the Modeling to measure the satisfaction σ of the elements. We can look at genes as an example: groups of nucleotides co-occur and interact with other groups and with proteins. Genes are identified by observing nucleotides that keep close together and act together to perform a function. The fulfillment of this function can be seen as a goal of the gene. Dividing the system into *modules* also divides the problem it needs to solve, so a complex task will be able to be processed in parallel by different modules. Certainly, the integration of the "solutions" given by each module arises as a new problem. Nevertheless, modularity in a system also increases its robustness and adaptability (Simon, 1996; Watson, 2002; Fernández and Solé, 2004).

The representation should consider at least two levels of abstraction, but if there are many variables and interactions in the system, more levels can be contemplated. Since elements and systems can be seen as agents,

we can refer to all of them as x-agents, where x denotes the level of abstraction relative to the simplest elements. For example, a three-layered abstraction would contemplate elements (0-agents) forming systems that are elements (subsystems, 1-agents) of a greater system (supersystem, 2-agents). If we are interested in modeling research institutes, 0-agents would be researchers, 1-agents would be research groups, and the research institute would be a 2-agent. A university containing several research institutes would be a 3-agent.. Each of these have goals and satisfactions (σ^x) that can be described and interrelated. For engineering purposes, the satisfaction of the highest level, i.e. the satisfaction of the system that is being designed, will be determined by the tasks expected from it. If these are fulfilled, then it can be said that the system is "satisfied". Thus, the designer should concentrate on engineering elements that will strive to reach this satisfaction.

If there are few elements or interactions in the Representation, most probably the system will be predictable and understandable, i.e. its state space can be exhaustively analyzed and there will be low complexity. The system might be better described using traditional approaches, since this Methodology might prove redundant. A large variety of elements and/or interactions might imply a complexity too high to be managed. If this is the case, the Representation should be revised before entering the Modeling stage.

4.3.2 Modeling

"Modeling, it should be clear, is an art form. It depends on the experience and taste of the modeler. In this it is much like cartooning, especially political cartooning. The modeler (cartoonist) must decide which features to make salient (exaggerate), and which features to eliminate (avoid), in order to answer the questions (make the political point)" —John Holland, Hidden Order, p. 11.

In science, models should ideally be as simple as possible, and predict as much as possible (Shalizi, 2001). Following Occam's razor, these models will provide a better understanding of a phenomenon than complicated models. Therefore, a good model requires a good Representation. The "elegance" of the model will depend very much on the metaphors we use to speak about the system. If the model turns out to be cumbersome, the Representation should be revised.

The Modeling should specify a Control mechanism that will ensure that the system does what it is required to do. Since we are interested in self-organizing systems, the Control will be *internal* and *distributed*. If the

problem is too complex, it can be divided into different subproblems. The Modeling should also consider different trade-offs for the system.

Control mechanism

*"We shouldn't see ourselves as 'controllers' of the world, but as 'actors' **in** the world"*

The Control mechanism can be seen as a *mediator* (Michod, 2003; Heylighen, 2003a), ensuring the proper interaction between elements of a system, and one that should produce the desired performance. However, one cannot have a strict control over a self-organizing system. Rather, the system should be *steered* (Wiener, 1948). In a sense, self-organizing systems are like teenagers: they cannot be tightly controlled since they have their own goals. We can only attempt to steer their actions, trying to keep their internal variables within certain boundaries, so that the systems (teenagers) do not "break" (in Ashby's (1947a) sense).

To develop a Control, the designer should find aspect systems, subsystems, or constraints that will prevent the negative interferences between elements (friction) and promote positive interferences (synergy). In other words, the designer should search for ways of minimizing frictions ϕ_i's that will result in maximization of the global satisfaction σ_{sys}. The performance of different mediators can be measured using equation (4.1)[5].

The Control mechanism should be *adaptive*. Since the system is dynamic and there are several interactions within the system and with its environment, the Control mechanism should be able to cope with the changes within and outside the system, in other words, be *robust*. An adaptive Control will be efficient in more contexts than a rigid one. In other words, the Control should be *active* in the search of solutions. A rigid Control will not be able to cope with the complexity of the system. There are several methods for developing an adaptive Control, e.g. Sastry and Bodson (1994). But these should be applied in a distributed way, in an attempt to reduce friction and promote synergy.

Different methods for reducing friction in a system can be identified. In the following cases, an agent A negatively affected by the behavior of an agent B will be considered[6]:

[5]This is similar to mechanism design (Dash et al., 2003). However, the present approach is intended for non-stationary problem domains, whereas mechanism design has been applied mainly in stationary problem domains where there are many agents with conflicting goals

[6]Even when equation 4.2 relates the satisfaction of an element to the satisfaction of the system, it can also be used for the relation between satisfactions of two elements.

- **Tolerance**. This can be seen as the acceptance of others and their goals. A can tolerate B by modifying itself to reduce the friction caused by B, and therefore increase σ_A. This can be done by moving to another location, finding more resources, or making internal changes. In an ecosystem, an animal can allow a new creature into its territory, or even give up its own territory, to prevent conflict.

- **Courtesy**. This would be the opposite case to Tolerance. B should modify its behavior not to reduce σ_A. In the same ecosystem, the new animal could opt to search for another territory to avoid conflict.

- **Compromise**. A combination of Courtesy and Tolerance: both agents A and B should modify their behaviors to reduce the friction. This is a good alternative when both elements cause friction to each other. This will be common when A and B are similar, as in homogeneous systems. Two animals could share the same territory. If resources are sufficient, tolerating each other is less problematic than fighting one another.

- **Imposition**. This could be seen as forced Courtesy. The behavior of B could be changed by force. The Control could achieve this by constraining B or imposing internal changes. In some primate societies, α-males "police" over the members of their group to prevent conflicts (Flack et al., 2006).

- **Eradication**. As a special case of Imposition, B can be eradicated. This certainly would decrease σ_B, but can be an alternative when either σ_B does not contribute much to σ_{sys}, or when the friction caused by B in the rest of the system is very high. Immune systems eliminate some cells that are not necessary for the organism.

- **Apoptosis**. B can "commit suicide". This would be a special case of Courtesy, where B would destroy itself for the sake of the system. "Programmed death" in cells occurs when they are no longer needed for an organism.

The difference between Compromise / Apoptosis and Imposition / Eradication is that in the former cases, change is triggered by the agent itself, whereas in the latter the change is imposed by a mediator. Tolerance and Compromise could be generated by an agent or by a mediator.

Different methods for reducing friction can be used for different problems. A good Control will select those in which other σ's are not reduced more than the gain in σ's. The choice of the method will also depend on

the importance of different elements for the system. Since more important elements contribute more to σ_{sys}, these elements can be given preference by the Control in some cases.

Different methods for increasing synergy can also be identified. These will consist of taking measures to increase σ_{sys}, even if some σ's are reduced:

- **Cooperation**. Two or more agents adapt their behavior for the benefit of the whole. This might or might not reduce some σ's. For example, group hunting enables animals to hunt for prey that a single individual would not be able to bring down.

- **Individualism**. An agent can choose to increase its σ if it increases σ_{sys}. A mediator should prevent increases in σ's if these reduce σ_{sys}, i.e. friction. For example, the success of a company is useful for the economy of a country, since it provides a source of employment and resources via taxes.

- **Altruism**. An agent can choose to sacrifice an increase of its σ or to reduce its σ in order to increase σ_{sys}. This would make sense only if the *relative* increase of σ_{sys} is greater than the decrease of the σ of the altruistic agent. A mediator should prevent wasted altruism. Philanthropists sacrify part of their fortunes for the benefit of society.

- **Exploitation**. This would be forced Altruism: an agent is driven to reduce its σ to increase σ_{sys}. Slavery and colonialism have profited imperialist nations, independently of the low satisfaction of the slaves / colonies.

A common way of reducing friction is to enable agents to learn via reinforcement (Kaelbling et al., 1996). With this method, agents tend to repeat actions that bring them satisfaction and avoid the ones that reduce it. Evolutionary approaches, such as genetic algorithms (Mitchell, 1996), can also reduce friction and promote synergy. However, these tend to be "blind", in the sense that variations are made randomly, and only their effects are evaluated. With the criteria presented above, the search for solutions can be guided with a certain aim. However, if the relationship between the satisfaction of the elements and the satisfaction of the system is too obscure, "blind" methods remain a good alternative.

An example of friction reduction can be seen in social systems across different cultures with social norms, which act as mediators between people. Religions have prevented—with different degrees

of success—friction between members of a community, by means of Tolerance, Courtesy, Compromise, Imposition, or Eradication, reaching a degree of social satisfaction that would be impossible without them (Wilson, 2003). In our globalized era, social norms have a similar effect. For example, in societies with high diversity it is considered immoral to discriminate minorities, preventing conflicts, minimizing potential friction, and increasing the satisfaction at the individual and social levels.

An example of synergy promotion can be given with ant colony optimization (Dorigo and Stützle, 2004). Every simulated ant tries to find the shortest path in a graph to a goal, with a probability of following pheromone trails left by other ants. The pheromones evaporate, so short paths are reinforced. We can say that the satisfaction of an ant is inversely proportional to the length of the path it traverses. There is no real friction between ants, since they do not compete for resources. But their pheromones promote their collective problem solving, since good solutions will be selected more often by other ants, reinforcing them and finally leading all the population to a near optimal solution, which is adaptable to changes in the graph. This promotes synergy by Individualism, since each ant searches to increase its own satisfaction, which linearly integrated for all ants leads to an overall system satisfaction (all ants satisfied means that the solution has been found). The mediator between the ants would be the pheromones, that promote short paths by evaporating more on longer paths, successfully integrating the efforts of single ants into a global solution.

A more complicated example can be seen with Google's PageRank (Brin and Page, 1998). To find the relevance of webpages, PageRank recursively assigns a value to each page depending on the PageRank of the webpages that have hyperlinks to it. The satisfaction of a webpage would be determined by its PageRank, assuming that all webpages would like to have the highest possible visibility. The satisfaction of the search engine would be determined by the users, i.e. if the engine returns useful results. Thus, there is a competition between webpages to have a higher PageRank. Still, Cooperation can arise e.g. with reciprocal links between webpages. Also, Altruism can be promoted by PageRank, since webpages will usually link to other pages that their authors deem useful. Nevertheless, this can enhance the value of the altruistic webpage, if it is found to be useful by other users. The key is that with PageRank as a mediator webpages cannot increase their own PageRank value, nor decrease the values of other webpages. Thus, the webpages are bounded, and cannot be individualistic nor generate friction. They only can be altruistic or cooperative. Nevertheless, a webmaster could try to cheat by generating several useless webpages that link to one that the webmaster would like

its PageRank increased. However, when Google detects such behavior it simply eradicates the cheating pages.

In general, the Control should explore different alternatives, trying to constantly increase σ_{sys} by minimizing friction and maximizing synergy. This is a constant process, since a self-organizing system is in a dynamic environment, producing "solutions" for the current situation. Note that a mediator will not necessarily maximize the satisfaction of the agents. However, it should try to do so for the system.

A mediator can also act on a system by bounding or promoting randomness, noise, and variability, taking it to the "*edge of chaos*" (Kauffman, 1993), where the stability and dynamics of the system are well balanced to explore good solutions without losing functionality. From the notion of complexity presented in Chapter 2 (Equation 2.1), we can see that adding more interactions or elements into a system will make it more complex. A system that is too complex is difficult to control, and a system not complex enough will not be able to perform a desired function. Limiting or promoting sources of complexity will enable us to have a system capable of performing its function.

Dividing the problem

If the system is to deal with many parameters, then it can be seen as a *cognitive* system (Gershenson, 2004a). It must "know" or "anticipate" what to do according to the current situation and previous history. Thus, the main problem, i.e. what the elements should do, could be divided into the problems of communication, cooperation, and coordination (Gershenson and Heylighen, 2004)[7].

For a system to self-organize, its elements need to *communicate*: they need to "understand" what other elements, or mediators, "want" to tell them. This is easy if the interactions are simple: sensors can give *meaning* to the behaviors of other elements. But as interactions become more complex, the *cognition* (Gershenson, 2004a) required by the elements should also be increased, since they need to process more information. New meanings can be learned (Steels, 1998; de Jong, 2000) to adapt to the changing conditions. These can be represented as "concepts" (Gärdenfors, 2000), or encoded, e.g. in the weights of a learning neural network (Rojas, 1996). The precise implementation and philosophical interpretations are not relevant for an engineer if the outcome is the desired one.

The problem of *cooperation* has been widely studied using game theory

[7] These subproblems are described in more detail in Chapter 7.

(Axelrod, 1984). There are several ways of promoting cooperation, especially if the system is designed. To mention only two of them: the use of tags (Riolo et al., 2001; Hales and Edmonds, 2003) and multiple levels of selection (Michod, 1997; Lenaerts, 2003) have proven to yield cooperative behavior. This will certainly reduce friction and therefore increase σ_{sys}.

Elements of a system should *coordinate* while reducing friction, not to obstruct each other. An important aspect of coordination is the *division of labor*. This can promote synergy, since different elements can specialize in what they are good at and *trust*[8] others to do what they are good at (Gaines, 1994; Di Marzo Serugendo, 2004). This process will yield a higher σ_{sys} compared to the case when every element is meant to perform all functions independently of how well each element performs each function. A good Control will promote division of labor by mediating a balance between *specialization* and *integration*: elements should devote more time doing what they are best at, but should still take into account the rest of the system. Another aspect of coordination is the *workflow* : if some tasks are prerequisites of other tasks, a mediator should synchronize the agents to minimize waiting times.

Trade-offs

A system needs to be able to cope with the complexity of its domain to achieve its goals. There are several trade-offs that can be identified to reach a balance and cope better with this complexity:

- **Complexity of Elements/Interactions**. The complexity of the system required to cope with the complexity of its domain can be tackled at one end of the spectrum by complex elements with few/simple interactions, and at the other by simple elements with several/complex interactions.
- **Quality/Quantity**. A system can consist at one extreme of a few complex elements, and at the other of several simple elements.
- **Economy/Redundancy**. Solving a problem with as few elements as possible is economical. But a minimal system is very fragile. Redundancy is one way of favoring the *robustness* of the system (von Neumann, 1966; Fernández and Solé, 2004; Wagner, 2004; Gershenson, Kauffman, and Shmulevich, 2006). Still, too much redundancy can also reduce the speed of adaptation and increase costs for maintaining the system.

[8] Trust is also important for communication and cooperation.

- **Homogeneity/Heterogeneity**. A homogeneous system will be easier to understand and control. A heterogeneous system will be able to cope with more complexity with less elements, and will be able to adapt more quickly to sudden changes. If there is a system of ten agents each able to solve ten tasks, a homogeneous system will be able to solve ten tasks robustly (if we do not consider combinations as new tasks). A fully heterogeneous system would be able to solve a hundred tasks, but it would be fragile if one agent failed. Heterogeneity also brings diversity, that can accelerate the speed of exploration, adaptation, and evolution, since different solutions can be sought in parallel. The diversity is also related to the amount of variety of perturbations that the system can cope with (Ashby, 1956).

- **System/Context**. The processing and storage of information can be carried out internally by the system, or the system can exploit its environment "throwing" complexity into it, i.e. allowing it to store or process information (Gershenson, Broekaert, and Aerts, 2003). For example, a navigating robot can use an internal map to guide itself though a labyrinth, or it can simply follow walls for exploration. In the first case the task is solved by the internal processing of the robot, while in the second the robot actively interacts with its environment, using the information provided by it.

- **Ability/Clarity**. A powerful system will solve a number of complex problems, but it will not be very useful if the functioning of the system cannot be understood. Designers should be able to understand the system in order to be able to control it (Schweitzer, 2003).

- **Generality/Particularity**. An abstract Modeling will enable the designer to apply the Modeling in different contexts. However, particular details should be considered to make the implementation feasible.

There are only very relative ways of measuring the above mentioned trade-offs. However, they should be kept in mind during the development of the system.

While developing a particular system, the trade-offs will become clearer once the Simulation is underway. They can then be reconsidered and the Modeling updated.

4.3.3 Simulation

The aim of this stage is to build computer simulation(s) that implement the model(s) developed in the Modeling stage, and test different scenarios and mediator strategies. This is a key stage, since the precise behaviors of a complex system cannot be easily deduced from the Modeling, i.e. they are not reducible. In other words, a model needs to "run" before it can be understood.

The Simulation development should proceed in stages: from abstract to particular. First, an abstract scenario should be used to test the main concepts developed during the Modeling. Only when these are tested and refined, should details be included in the Simulation. This is because particular details take time to develop, and there is no sense in investing before knowing whether the Modeling is on the right track. Details can also influence the result of the Simulation, so they should be put off until a time when the main mechanisms are understood.

The Simulation should compare the proposed solutions with traditional approaches. This is to be sure that applying self-organization in the system brings any benefit. Ideally, the designer should develop more than one Control to test in the simulation. A rock-scissors-paper situation could arise, where no Control is best in all situations (also considering classic controls). The designer can then adjust or combine the Controls, and then compare again in the Simulation.

Experiments conducted with the aid of the Simulation should go from simple to extensive. Simple experiments will show proof of concepts, and their results can be used to improve the Modeling. Once this is robust, extensive studies should be made to be certain of the performance of the system under different conditions.

Based on the Simulation results and insights, the Modeling and Representation can be improved. A Simulation should be mature before taking the implementation into the real world.

4.3.4 Application

The role of this stage is basically to use the developed and tested model(s) in a real system. If this is a software system, the transition will not be so difficult, since the software would have been already developed in the Simulation stage. On the other hand, the transition to a real system can expose artifacts of a naïve Simulation. A useful way to develop robust Simulations consists in adding some noise into the system (Jakobi, 1997).

Like this, the Simulation will already need to deal with unexpected situations generated by the noise.

Good theoretical solutions can be very difficult / expensive / impossible to implement (e.g. if they involve instantaneous access to information, mind reading, teleportation, etc.). The feasibility of the Application should be taken into account during the whole design process. In other words, the designer should have an *implementation bias* in all the Methodology stages. If the proposed system turned out to be too expensive or complicated, all the designer's efforts would be fruitless. If the system is developed for a client, there should be feedback between developers and clients during the whole process (Cotton, 1996) to avoid client dissatisfaction once the system is implemented. The *legacy* of previous systems should also be considered for the design (Valckenaers et al., 2003): if the current implementation is to be modified but not completely replaced, the designer is limited by the capabilities of the old system.

Constraints permitting, a pilot study should be made before engaging in a full Application, to detect incongruences and unexpected issues between the Simulation or Modeling stages and the Application. With the results of this pilot study, the Simulation, Modeling, and Representation can be fine-tuned.

4.3.5 Evaluation

Once the Application is underway, the performance of the new system should be measured and compared with the performances of the previous system(s).

Constraints permitting, efforts should be continued to try to improve the system, since the requirements it has to meet will certainly change with time (e.g. changes of demand, capacity, etc.). The system will be more adaptive if it does not consider its solution as the best once and for all, and is able to change itself according to its performance and the changing requirements.

4.3.6 Notes on the Methodology

- All returning arrows in the Figure 4.1 are given because it is not possible to predict the outcome of strategies before they have been tried out. All information and eventualities cannot be abstracted, nor emergent properties predicted before they have been observed. Emergent properties are *a posteriori*.

- The proposed Methodology will be useful for unpredictable and/or non-stationary problem domains, where all the possible system's situations cannot be considered beforehand. It could also be useful for creative tasks, where a self-organizing system can explore the design space in an alternative way. The Methodology in principle is applicable to describe any system, but it would be redundant in simple or stationary problem domains, i.e. with a fixed solution, where adaptation is not required.

- For certain systems, some σ's might be difficult to define or measure. Still, following the Methodology can aid in the definition of these variables after an initial Modeling or Simulation. As with the function f and the weights w_i's, the Methodology does not require a precise definition of satisfactions to start exploring solutions. These come in many cases *a posteriori*.

- Most methodologies in the literature apply to software systems, e.g. Jacobson et al. (1999); Jennings (2000). This one is more general, since it is domain independent. The principles presented are such that they can be applied to any domain for developing a functioning self-organizing system: any system can be modeled as a group of agents, with satisfactions depending on their goals. The question is *when* it is useful to use this Methodology. Only application of the Methodology will provide an answer to this question. It should be noted that similar approaches have been proposed in parallel. For example, Capera et al. (2003) propose the avoidance of "non-cooperative" situations. This Methodology considers cooperation as only one way to reduce friction or improve synergy, as discussed in Section 4.3.2. De Wolf and Holvoet (2005) also suggest the use of self-organization (and emergence), but their proposal is limited to software multi-agent systems . As the present work, they are part of the ongoing effort by the research community to understand self-organizing systems.

- The proposed Methodology is not quite a spiral design model (Boehm, 1988), because the last stage does not need to be reached to return to the first. That is, there is no need to deploy a working version (finish a cycle/iteration) before revisiting a previous stage, as for example in extreme programming. Rather, the Methodology is a "*backtracking* design model", where the designer can always return to previous stages (not necessarily the immediate previous one).

- If the system has multiple designers, these should agree on the goals of the system, in order to measure its satisfaction and performance. Still, in many systems, e.g. a country, different actors can suggest different goals, e.g. economy, health, or environment. This is indeed a problem, which is present independently of the methodology used to design such systems. Nevertheless, the present Methodology can aid in the exploration of possible outcomes, to help the stakeholders agree (or at least reach a compromise) on the system's goals.

- It is not necessary to understand a solution before testing it. In many cases understanding can come only after testing, i.e., the global behavior of the system is irreducible. Certainly, understanding the causes of a phenomenon will allow the modelers to have a greater control over it.

A detailed diagram of the different substeps of the Methodology can be appreciated in Figure 4.2.

4.4 Discussion

"Problems are unavoidable. We can only adapt to the ones we couldn't predict."

The backtracking between different steps in the Methodology is necessary because the behavior of the system cannot be predicted from the Modeling, i.e. it is not reducible. It might be possible to reason about all possible outcomes of simple systems, and then to implement the solution. But when complexity needs to be dealt with, a mutual feedback between experience and reasoning needs to be established, since reasoning alone cannot process all the information required to predict the behavior of a complex system (Edmonds, 2005).

For this same reason, it would be preferable for the Control to be distributed. Even when a central supercomputer could possibly solve a problem in real time, the information delay caused by data transmission and integration can reduce the efficiency of the system. Also, a distributed Control will be more robust, in as much as if a module malfunctions, the rest of the system can still provide reliable solutions. If a central Control fails, the whole system will stop working.

When the elements of a system are simple, bringing goals and satisfactions into the picture might complicate matters without having any benefit. This is a disadvantage of generality. Not that the methodology could

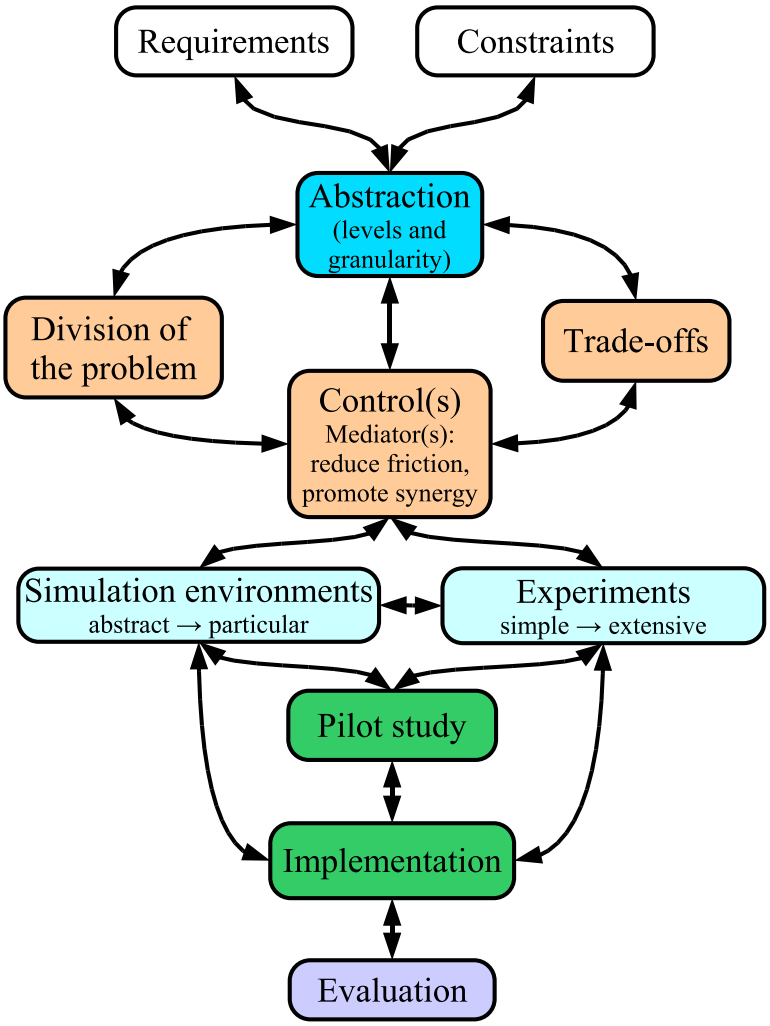


Figure 4.2: Detailed diagram of Methodology.

not be used, only that it would be redundant. The same would be for stationary or comprehensible problem domains, where an exact solution can be calculated and no adaptation is required. The proposed Methodology is aimed at being useful in complex, unpredictable problem domains.

The Simulation and Experiments are strictly necessary in the design of self-organizing systems (Edmonds, 2005). This is because their performance cannot be evaluated by purely formal methods (Edmonds and Bryson, 2004). Still, formal methods are necessary in the first stages of the Methodology. I am not suggesting a trial-and-error search. But since the behavior of a complex system in a complex environment cannot be predicted completely, the models need to be contrasted with experimentation before they can be validated. This Methodology suggests one possible path for finding solutions. The lack of predictability does not come only from chaotic processes. It might come also from new information generated by the interactions, so that the system behavior cannot be predicted from the behavior of the elements. Thus, one is forced to "let the system run and then see", as with cellular automata (Wolfram, 1986).

Now the reader might wonder whether the proposed Methodology is a *top-down* or a *bottom-up* approach. Well, it is both and neither, since (at least) higher and lower levels of abstraction need to be considered simultaneously. The approach tests different local behaviors, and observes local and global (and meso) performances, for local and global (and meso) requirements. Thus, the Methodology can be seen as a *multi-level* approach.

Since "conflicts" between agents need to be solved at more than one level, the Control strategies should be carefully chosen and tested. A suboptimal (Machol, 1965, pp. 1–8) situation as in the prisoner's dilemma (Axelrod, 1984) might easily arise, when the "best" solution on one level/timescale is not the best solution on another level/timescale (Heylighen, 1992).

Many frictions between agents are due to faulty communication, especially in social and political relations. If agents do not "know" the goals of others, it will be much more difficult to coordinate and increase σ_{sys}. For example, in a social system, knowing what people or corporations need to fulfill their goals is not so obvious. Still, with emerging technologies, social systems perform better in this respect. Already in the early 1970s, the project Cybersyn in Chile followed this path (Miller Medina, 2005): it kept a daily log of productions and requirements from all over the country (e.g. mines, factories, etc.), in order to distribute products where they were needed most; and as quickly as possible. Another step towards providing faster response to the needs of both individuals and social systems can be found in e-government (Layne and Lee, 2001). A company should also fol-

low these principles to be able to adapt as quickly as possible. It needs to develop "sensors" to perceive the satisfactions and conflicts of agents at different levels of abstraction, and needs to develop fast ways of adapting to emerging conflicts, as well as to changing economic environment. A tempting solution might be to develop a homogeneous system since, for example, homogeneous societies have fewer conflicts (Durkheim, 1893). This is because all the elements of a homogeneous system pursue the same goals. Thus, less diversity is easier to control. However, less diversity will be less able to adapt to sudden changes. Nevertheless, societies cannot be *made* homogeneous without generating conflicts since people are already diverse, and therefore already have a diversity of goals. The legacy (Valckenaers et al., 2003) of social systems gives less freedom to a designer, since diverse individual goals may be already present in the system. A social Control/mediator needs to satisfy these individual goals while trying to satisfy those of the social system.

4.5 Conclusions

"Philosophers get paid for posing interesting questions; scientists for answering them. Thus, one cannot live without the other..."

This chapter presented a conceptual framework and a general methodology for designing and controlling self-organizing systems. The Methodology proposes the exploration for proper Control mechanisms / mediators / constraints that will reduce frictions and promote synergy so that elements will dynamically reach a robust and efficient solution. The proposed Methodology is general not because it is the best in all contexts, but because it is applicable in a wide range of domains. Certainly, it is not the only way to *describe* self-organizing systems.

Even if this work is aimed mainly at engineers, it is rather philosophical. The chapter presented no concrete results (encouraging practical results will be presented in the following chapters), but *ideas* that can be exploited to produce them. Certainly, these ideas have their roots in current practices, and many of them are not novel. Still, the aim of this work is not for novelty but for synthesis.

The Methodology strives to build artificial systems. Nevertheless, these could be used to understand natural systems using the synthetic method (Steels, 1993). [9] Therefore, the ideas presented here are potentially useful not only for engineering, but also for science.

[9] The synthetic method builds up theories of a system attempting to construct an ar-

The backtracking ideology is also applicable to this Methodology: it will be improved once applied, through learning from experience. This Methodology is not final, but evolving. The more this Methodology is used, and in a wider variety of areas, the more potentially useful its abstractions will be. For example, would it be a good strategy to minimize the standard deviation of σ's? This might possibly increase stability and reduce the probability of conflict, but this strategy, as any other, needs to be tested before it can be properly understood.

In the next chapters, this Methodology is applied to different domains to develop self-organizing systems with practical applications.

tificial system that exhibits the same capabilities of the natural system. This is different from the more traditional inductive method, which tests a theory of a system by matching predicted facts against observed facts (Steels, 1993).

Chapter 5

Self-organizing Traffic Lights

In this chapter[1]*, the Methodology presented in the previous chapter is applied to self-organizing traffic lights. A multi-agent Simulation is used to study three novel self-organizing methods, which are able to outperform traditional rigid and adaptive methods. Using simple rules and no direct communication, traffic lights are able to self-organize and adapt to changing traffic conditions, reducing waiting times, number of stopped cars, and increasing average speeds. A more realistic simulation was developed to model a Brussels avenue (Cools, 2006), where the self-organizing methods also outperform the current "green wave" method.*

[1] Based on Gershenson (2005); Cools et al. (2007).

5.1 Introduction

Anyone living in a populated area suffers from traffic congestion. Traffic is time, energy, and patience consuming. This has motivated people to regulate traffic flow in order to reduce the congestion. The idea is simple: if vehicles are allowed to go in any direction, there will be a high probability that one will obstruct another. To avoid this, rules have been introduced to *mediate* (Heylighen, 2003a) between the conflicting vehicles, by restricting or bounding their behavior. People have agreed on which side of the street they will drive (left or right); traffic lanes prevent cars from taking more space than necessary; traffic signals and codes prompt an appropriate behavior; and traffic lights regulate the crossing of intersections.

There is no solution to the traffic congestion problem when the car density saturates the streets, but there are many ways in which the car flow can be constrained in order to improve traffic. Traffic lights are not the only component to take into account, but they are an important factor. We can say that a traffic light system will be more efficient if, for a given car density, it increases the average speeds of vehicles. This is reflected in less time that cars will wait behind red lights, with the outcome of less energy spent and thus less pollution produced (assuming comparable acceleration rates, since a car will spend more energy accelerating and breaking constantly compared to one with a constant speed).

For decades, people have been using mathematical and computational methods that find appropriate periods and phases (i.e., cycles) of traffic lights, so that the variables considered will be optimized. This is good because certain synchronization is better than having no correlation of phases. However, many methods applied today do not consider the current state of the traffic. If cars are too slow for the expected average speed, this might result in the loss of the phases dictated by the traffic lights. If they go too fast, they will have to wait until the green light phase reaches every intersection. The optimizing methods are blind to "abnormal" situations, such as many vehicles arriving or leaving a certain place at the same time, such as a stadium after a match, a financial district before check-in time, or a university on a Friday evening. In most cases, traffic agents need to override the traffic lights and personally regulate the traffic. Nevertheless, traffic modeling has improved greatly our understanding of this complex phenomenon, especially during the past decade (Prigogine and Herman, 1971; Wolf et al., 1996; Schreckenberg and Wolf, 1998; Helbing et al., 2000; Helbing, 1997; Helbing and Huberman, 1998), suggesting different improvements to the traffic infrastructure.

I defend that traffic light control is not so much an optimization prob-

lem, but rather an adaptation problem, since traffic flows and densities change constantly. Optimization gives the best possible solution for a given configuration. For example, in wing design, optimization offers good solutions, since the aerodynamic demands for an airplane are constant, i.e. for a certain speed, weight, etc. But since the configuration is changing constantly in real traffic, it seems that we would do better with an adaptive mechanism than with a mechanism that is optimal sometimes, and creates havoc at other times. Indeed, modern "intelligent" advanced traffic management systems (ATMS) use learning methods to adapt phases of traffic lights, normally using a central computer (Federal Highway Administration, 1998; Hunt et al., 1981).[2] Another reason for preferring an adaptive method is that optimization can be computationally expensive. Trying to find all possible optimal solutions of a city is not feasible, since the configuration space is too huge, uncertain, and changes constantly.

In this Chapter, recent work on self-organizing traffic lights (Gershenson, 2005; Cools, 2006) will be used to illustrate the flow through the different steps of the Methodology. These traffic lights are called self-organizing because the global performance is given by the local rules followed by each traffic light: they are unaware of the state of other intersections and still manage to achieve global coordination. In the following sections, steps proposed by the Methodology are alternated with results from experiments carried out in computer simulations, followed by Discussion and Conclusions of the chapter.

5.2 Applying the Methodology I

Requirements. The goal is to develop a feasible and efficient traffic light control system.

Representation. The traffic light system can be modeled on two levels: the vehicle level and the city level. These are easy to identify because vehicles are objects that move through the city, establishing clear spatiotemporal distinctions. The goal of the vehicles is to flow as fast as possible, so their "satisfaction" σ can be measured in terms of their average speed and average waiting time at a red light. Cars will have a maximum σ if they go as fast as they are allowed, and do not stop at intersections. σ would be zero if a car stops indefinitely. The goal of the traffic light system on the

[2] A drawback of ATMS is their high cost and complexity that requires maintenance by specialists. There is yet no standard, and usually companies are hired to develop particular solutions for different cities. The private nature of ATMS also makes them difficult to study and compare with alternative methods.

city level is to enable vehicles to flow as fast as possible, while mediating their conflicts for space and time at intersections. This would minimize fuel consumption, noise, pollution, and stress in the population. The satisfaction of the city can be measured in terms of the average speeds and average waiting times of all vehicles (i.e. average of σ_i, $\forall i$), and with the average percentage of stationary cars. σ_{sys} will be maximum if all cars go as fast as possible, and are able to flow through the city without stopping. If a traffic jam occurs and all the vehicles stop indefinitely, then σ_{sys} would be minimal.

Modeling. Now the problem for the Control can be formulated: find a mechanism that will coordinate traffic lights so that these will mediate between vehicles to reduce their friction (i.e. try to prevent them from arriving at the same time at crossings).[3] This will maximize the satisfactions of the vehicles and of the city (σ_i's and σ_{sys}). Since all vehicles contribute equally to σ_{sys}, ideally the Control should minimize frictions via Compromise.

Simulation. Several traffic simulations use cellular automata to model traffic effectively (Faieta and Huberman, 1993; Biham et al., 1992; Nagel and Schreckenberg, 1992; Chowdhury and Schadschneider, 1999), since they are computationally cheap. However, the increase of computing power in the past few years has allowed the development of multi-agent simulations to create more realistic traffic simulations (Nagel, 2004; Wiering et al., 2004; Miramontes Hercog, 2004; Roozemond and Rogier, 2000). Thus, a simple simulation was developed in NetLogo (Wilensky, 1999), a multi-agent modeling environment. The "Gridlock" model (Wilensky and Stroup, 2002) was extended to implement different traffic control strategies. It consists of an abstract traffic grid with intersections between cyclic single-lane arteries of two types: vertical or horizontal (similar to the scenarios of Biham et al. (1992) and Brockfeld et al. (2001)). Cars only flow in a straight line, either eastbound or southbound. Each crossroad has traffic lights that allow traffic flow in only one of the intersecting arteries with a green light. Yellow or red lights stop the traffic. The light sequence for a given artery is green-yellow-red-green. Cars simply try to go at a maximum speed of 1 "patch" per timestep, but stop when a car or a red or yellow light is in front of them. Time is discrete, but not space. Cars use an acceleration of 0.099 to speed up or slow down. A "patch" is a square of the environment the size of a car.

[3] As it was mentioned in Section 1.3.2, the self-organizing traffic light controllers were developed before the Methodology. Still, the concept of friction is used here to understand better the mechanisms and to illustrate the Methodology.

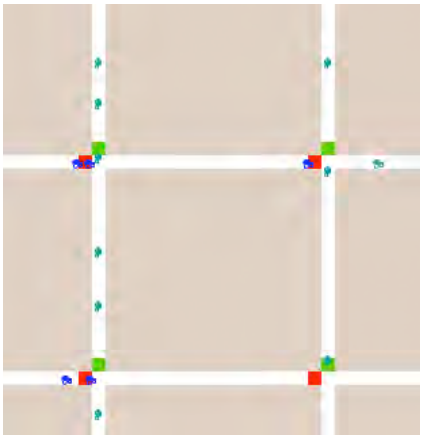


Figure 5.1: Screenshot of a part of the traffic grid. Green lights southbound, red lights eastbound.

A screenshot of a part of the environment can be seen in Figure 5.1. The user can change different parameters, such as the number of arteries or number of cars. Different statistics are shown: the number of stopped cars, the average speed of cars, and the average waiting times of cars. The reader is invited to test the simulation (source code included), with the aid of a Java-enabled Internet browser, at SOTL (2005).

At first, a tentative model was implemented. The idea was unsuccessful. However, after refining the model, an efficient method was discovered, named *sotl-request*.

Modeling. In the *sotl-request* method, each traffic light keeps a count κ_i of the number of cars times time steps ($c{*}ts$) approaching *only* the red light, independently of the status or speed of the cars (i.e. moving or stopped), from a distance ρ. κ_i can be seen as the integral of waiting/approaching cars over time. When κ_i reaches a threshold θ, the opposing green light turns yellow, and the following time step it turns red with $\kappa_i = 0$, while the red light which counted turns green. In this way, if there are more cars approaching or waiting before a red light, the light will turn green faster than if there are only few cars. This simple mechanism, described formally by Algorithm 1, achieves self-organization as follows: if there is a single or just a few cars, these will be made to stop for a longer period before a red light. This gives time for other cars to join them. As more cars join the group, cars will be made to wait shorter periods before a red light. Once there are enough cars, the red light will turn green even before the oncoming cars reach the intersection, thereby generating "green waves"

on demand. Having "platoons" or "convoys" of cars moving together improves traffic flow, compared to a homogeneous distribution of cars, since there are large empty areas between platoons, which can be used by crossing platoons with few interferences. The *sotl-request* method has no phase or internal clock. Traffic lights change only when the above conditions are met. If no cars are approaching a red light, the complementary light can remain green.

```
1  foreach (timestep) do
2      κ_i += cars_approachingRed in ρ;
3      if (κ_i ≥ θ) then
4          switchlight_i();
5          κ_i = 0
6      end
7  end
```

Algorithm 1: *Sotl-request* method.

Representation. It becomes clear now that it would be useful to consider traffic lights as agents as well. Their goal is to "get rid" of cars as quickly as possible. To do so, they should avoid having green lights on empty streets and red lights on streets with high traffic density. Since the satisfactions of the traffic lights and vehicles are complementary, they should interact via Cooperation to achieve synergy. Also, σ_{sys} could be formulated in terms of the satisfactions of traffic lights, vehicles, or both.

Modeling. Two classic methods were implemented to compare their performance with *sotl-request*: *marching* and *optim*.

Marching is a very simple method. All traffic lights "march in step": all green lights are either southbound or eastbound, synchronized in time. Intersections have a phase φ_i, which counts time steps. φ_i is reset to zero when the phase reaches a period value p. When $\varphi_i == 0$, red lights turn green, and yellow lights turn red. Green lights turn yellow one time step earlier, i.e. when $\varphi == p - 1$. A full cycle of an intersection consists of $2p$ time steps. "*Marching*" intersections are such that $\varphi_i == \varphi_j, \forall i, j$.

The *optim* method is implemented trying to set phases φ_i of traffic lights so that, as soon as a red light turns green, a car that was made to stop would find the following traffic lights green. In other words, a fixed solution is obtained so that *green waves* flow to the southeast. The simulation environment has a radius of r square patches, so that these can be identified with coordinates $(x_i, y_i), x_i, y_i \in [-r, r]$. Therefore, each artery consists of $2r + 1$ patches. In order to synchronize all the intersections, red lights

should turn green and yellow lights should turn red when

$$\varphi_i == round(\frac{2r + x_i - y_i}{4}) \tag{5.1}$$

and green lights should turn to yellow the previous time step. The period should be $p = r + 3$. The three is added as an extra margin for the reaction and acceleration times of cars (found to be best, for low densities, by trial and error).

These two methods are *non-adaptive*, in the sense that their behavior is dictated beforehand, and they do not consider the actual state of the traffic. Therefore, there cannot be Cooperation between vehicles and traffic lights, since the latter ones have fixed behaviors. On the other hand, traffic lights under the *sotl-request* method are sensitive to the current traffic condition, and can therefore respond to the needs of the incoming vehicles.

Simulation. Preliminary experiments showed that *sotl-request*, compared with the two traditional methods, achieves very good results for low traffic densities, but very poor results for high traffic densities. This is because depending on the value of θ, high traffic densities can cause the traffic lights to change instantly. This obstructs traffic flow. A new model was developed, taking this factor into account.

Modeling. The *sotl-phase* method takes *sotl-request* and only adds the following constraint: a traffic light will not be changed if the time passed since the last light change is less than a minimum phase, i.e. $\varphi_i < \varphi_{\min}$. Once $\varphi_i \geq \varphi_{\min}$, the lights will change if/when $\kappa_i \geq \theta$. This prevents the fast changing of lights[4]. *Sotl-phase* is described by Algorithm 2.

```
1  foreach (timestep) do
2      κ_i += cars_approachingRed in ρ;
3      if (φ_i ≥ φ_min) then
4          if (κ_i ≥ θ) then
5              switchlight_i();
6              κ_i = 0
7          end
8      end
9  end
```

Algorithm 2: *Sotl-phase* method.

Simulation. *Sotl-phase* performed a bit less effectively than *sotl-request* for very low traffic densities, but still much better than the classic methods.

[4] A similar method has been used successfully in the United Kingdom for some time, but for isolated intersections (Vincent and Young, 1986).

However, *sotl-phase* outperformed them also for high densities. An unexpected phenomenon was observed: for certain traffic densities, *sotl-phase* achieved *full synchronization*, in the sense that no car stopped. Therefore, speeds were maximal and there were no waiting times nor stopped cars. Thus, satisfaction was maximal for vehicles, traffic lights, and the city. This is an interesting but unrealistic situation, because full synchronization is achieved due to the toroidal topology of the simulation environment. The full synchronization is achieved because platoons are promoted by the traffic lights, and platoons can move faster through the city modulating traffic lights. If two platoons are approaching an intersection, *sotl-phase* will stop one of them, and allow the other to pass without stopping. The latter platoon keeps its phase as it goes around the torus, and the former adjusts its speed until it finds a phase that does not cause a conflict with another platoon.

Modeling. Understanding the behavior of the platoons, it can be seen that there is a favorable system/context trade-off. There is no need of direct communication between traffic lights, since information can actually be sent via platoons of vehicles. The traffic lights communicate *stigmergically* (Theraulaz and Bonabeau, 1999), i.e. via their environment, where the vehicles are conceptualized as the environment of traffic lights.

Another method was developed, *sotl-platoon*, adding two restrictions to *sotl-phase* for regulating the size of platoons. Before changing a red light to green, *sotl-platoon* checks if a platoon is not crossing through, not to break it. More precisely, a red light is not changed to green if on the crossing street there is at least one car approaching at ω patches from the intersection. This keeps platoons together by preventing the "tails" of platoons from being separated from the rest. For high densities, this restriction alone would cause havoc, since large platoons would block the traffic flow of intersecting streets. To avoid this, a second restriction is introduced. The first restriction is not taken into account if there are more than μ cars approaching the intersection. Like this, long platoons can be broken, and the restriction only comes into place if a platoon will soon be through an intersection. This is formally described by Algorithm 3.

The *cut-off* method was implemented, wanting to compare the self-organizing methods with a traditional traffic responsive method, that has proven to be better than rigid methods at single intersections (Fouladvand et al., 2004b). The idea of the *cut-off* method is simple: a traffic light will remain green until a queue of stopped waiting cars reaches a length of λ cars. At this moment, the green light turns yellow, and at the next time step, red, while the opposing light turns green. The advantage of this method is that it can adapt to changing traffic demands. The disadvantage

```
1  foreach (timestep) do
2      κ_i += cars_approachingRed in ρ;
3      if (φ_i ≥ φ_min) then
4          if not (0 < cars_approachingGreen in ω < μ) then
5              if (κ_i ≥ θ) then
6                  switchlight_i();
7                  κ_i = 0
8              end
9          end
10     end
11 end
```

Algorithm 3: *Sotl-platoon* method.

is that cars need to stop at an intersection before it changes. Recall that *sotl* methods keep a count of approaching cars, independently of their speed. Therefore, cars do not need to stop in order to change a traffic light.

To have an idea of the benefit of the different control methods, they were also compared with a non-correlated scheme *no-corr*: each traffic light is assigned a phase φ_i at random, and its value does not change during a simulation run. They all have the same period p. Thus, there is no correlation between different intersections.

5.3 Experiments: First Results

"Echando a perder se aprende"[5] *—Hispanic proverb*

Simulations were performed to obtain average statistics on the performance of the different control methods. These were namely speed,[6] percentage of stopped cars, and waiting time. The results shown in Figures 5.2 and 5.3 were obtained from runs of 10,000 time steps with random initial conditions in a grid of 10×10 arteries of $r = 80$ (therefore 3120 available patches, which represents the theoretical maximum carrying capacity[7]), with $p = 83$, $\theta = 41$, $\rho = 8$, $\varphi_{\min} = 20$, $\omega = 4$, $\mu = 3$, and $\lambda = 3$ (See Table 5.1 for a concise description of the parameters). These parameters were found to be the best for each method by a trial-and-error exploration of the parameter space. For each method one run was made varying the

[5] "One learns by spoiling"

[6] The cruise speed is 1 patch/time step, that is, the speed at which cars go without obstructions.

[7] This carrying capacity—which is reached when all the patches are occupied by a car—is theoretical, because in practice cars need space in front of them to move.

number of cars (shown on the x axis) from 20 to 2000, in steps of 20 (101 runs in total), with the same parameters. The results were extracted from NetLogo using a small Java program developed to automate simulation runs.[8] The curious reader is invited to repeat these and other experiments with the developed software laboratory SOTL (2005) and to analyze its source code.

Variable	Explanation	Used by
r	Radius (in patches) of simulation. Environment has patches of coordinates $(x, y) \in [-r, r]$.	Environment.
p	Period of fixed cycle methods.	*marching*, *optim*, and *no-corr*.
θ	Threshold of κ_i counter.	All *sotl* methods.
ρ	Distance (in patches) from red light at which cars are counted.	All *sotl* methods.
$\varphi_{\min}$	Minimum green phase.	*Sotl-phase* and *sotl-platoon*.
μ	Maximum size of platoon "tail".	*Sotl-platoon*.
ω	Distance (in patches) from green light at which "tails" are sensed.	*Sotl-platoon*.
λ	Number of cars waiting in queue to trigger light switch.	*Cut-off*.

Table 5.1: Parameters of NetLogo simulations.

We can see that the *marching* method is not very efficient for low traffic densities, that is, when there are roughly less than three cars between intersections. Since half of the arteries (all eastbound or all southbound) have red lights, this causes almost half of the cars to be stopped at any time, reducing the average speed of cars. On the other hand, its performance degrades slowly as the traffic densities reach certain levels, and performs the best for very high densities, that is, when more than eight cars are encountered between intersections, and traffic jam formation is probable. This is because it keeps a strict division of space occupied by cars, and interferences are less probable. Still, when there are too many cars, "deadlocks" are formed, that is, all cars are stopped.

[8] The precise figures were taken from the total averages for each variable (black lines in the NetLogo simulation plots).

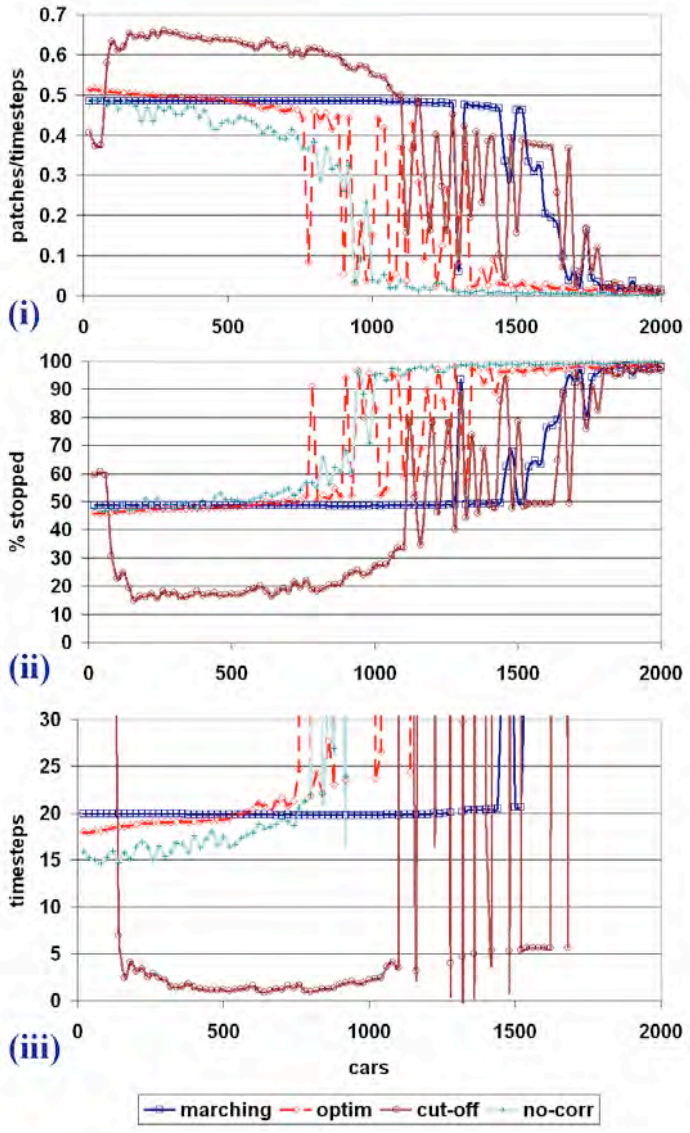


Figure 5.2: Results for standard methods as car density increases. (i) Average speeds of cars. (ii) Percentage of stopped cars. (iii) Average waiting times. Very high waiting times (out of graph) indicate deadlocks.

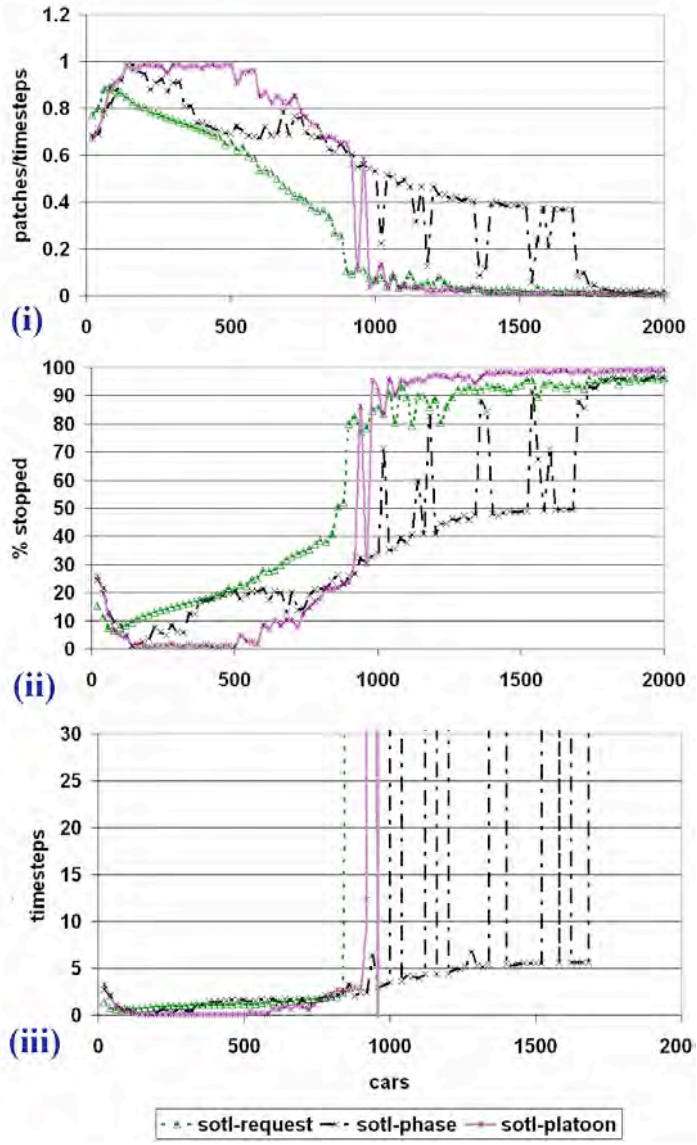


Figure 5.3: Results for self-organizing methods as car density increases. (i) Average speeds of cars. (ii) Percentage of stopped cars. (iii) Average waiting times. Very high waiting times (out of graph) indicate deadlocks.

For low densities, the *optim* method performs acceptably. However, for high densities cars can enter a deadlock much faster than with other methods. This is because cars waiting behind other cars at red lights do not reach green waves, reducing their speed and the speed of the cars which go behind them. Also, even when there will be some cars that do not stop, flowing through green waves, there will be an equivalent number of cars waiting to enter a green wave, losing the time gained by cars in green waves. Therefore, the performance cannot be much better than *marching*. Such a method could be useful only when the traffic flow distribution is highly skewed, e.g. 90% of cars flowing in one direction.

Sotl-request gives the best performance for low traffic densities because platoons can quickly change red lights into green, in most cases before actually reaching the intersections. Since the traffic density is low, this does not obstruct many cars approaching the intersection in the corresponding artery. However, for high densities this method is extremely inefficient, since there is a constant switching of lights due to the fact that θ is reached very fast. This reduces the speed of cars, since they stop on yellow lights, but also breaks platoons, so that the few cars that pass will have a higher probability of waiting more time at the next intersection.

Sotl-phase does not perform as good as *sotl-request* for low densities because in many cases cars need to wait behind red lights as κ_i reaches $\varphi_{\min}$, with no cars coming in the corresponding artery. The performance of *sotl* methods could be improved for low densities by reducing θ, since small platoons might need to wait too long at red lights. As the traffic density reaches a medium scale, platoons effectively exploit their size to accelerate their intersection crossing. With the considered parameters, in the region around 160 cars, and again at around 320, *sotl-phase* can achieve *full synchronization* in space, in the sense that no platoon has to stop, so all cars can go at a maximum speed. (In the graphs, the average speed reflects the average time it takes to achieve full synchronization, i.e., closer to one is faster.) This is not a realistic situation, because synchronization is achieved due to the toroidal topology of the simulation environment. Still, it is interesting to understand the process by which the full synchrony is reached. Platoons are formed, as described above, of observed sizes $3 \leq cars \leq 15$. One or two platoons flow per street. Remember that platoons can change red lights to green before they reach an intersection, if $\varphi_i \geq \varphi_{\min}$. If a platoon moving in an artery is obstructed, this will be because still $\varphi_i < \varphi_{\min}$, and because a platoon is crossing, or crossed the intersection recently in the complementary artery. The waiting of the platoon will change its phase compared to other flowing platoons. However, if no platoon crossed recently, a platoon will keep its phase relative to other

platoons. This induces platoons not to interfere with each other, until all of them go at maximum speed, i.e. with minimum friction. We can test that this condition is robust by resetting the traffic light periods and κ_i. Each reset can be seen in the spikes of the graphs shown in Figure 5.4. Nevertheless, the precise time at which full synchronization is reached can vary. For some initial conditions, full synchronization is not achieved, but it is approached nevertheless. One of this initial conditions arises when there are too many randomly assigned cars in one street, so that they cannot leave enough space for crossing platoons to cross without some car briefly stopping.

The phenomenon of full synchronization shows us how self-organizing traffic lights form platoons, which in turn modulate traffic lights. This feedback is such that it maximizes average speeds and minimizes waiting times and stopped cars (and thus friction) in a robust way. The self-organizing traffic lights are efficient without "knowing" beforehand the locations or densities of cars.

When there is a very high traffic density, *optim* and *sotl-request* reach deadlocks frequently, where all traffic is stopped. *Sotl-phase* behaves similar to *marching*, since traffic lights change as soon as $\kappa_i \geq \varphi_{\min}$, because in most cases $\kappa_i \geq \theta$ by then. This also reduces the sizes of platoons, which if very long can generate deadlocks. However, when the traffic density is too high, deadlocks will be inevitable, though *marching* generates less deadlocks than *sotl-phase*. This is because with the *marching* method whole arteries are either stopped or advancing. This reduces the probability of having a green light where cars cannot cross (e.g., due to a red light ahead, and a line of cars waiting to cross it), which would block the crossing artery at the next phase.[9]

Sotl-platoon manages to keep platoons together, achieving full synchronization commonly for a wide density range, more effectively than *sotl-phase*. This is because the restrictions of this method prevent platoons from leaving a few cars behind, with a small time cost for waiting vehicles. Still, this cost is much lower than breaking a platoon and waiting for separated vehicles to join back again. A platoon is divided only if $\mu = 3$, and a platoon of size three will manage to switch traffic lights without stopping for the simulation parameters used. However, for high traffic densities platoons aggregate too much, making traffic jams more probable. The *sotl-platoon* method fails when a platoon waiting to cross a street

[9]Deadlocks could be avoided by restricting all cars to cross intersections only if there is at least one free space after it. However, it is unrealistic to expect human drivers to behave in this way.

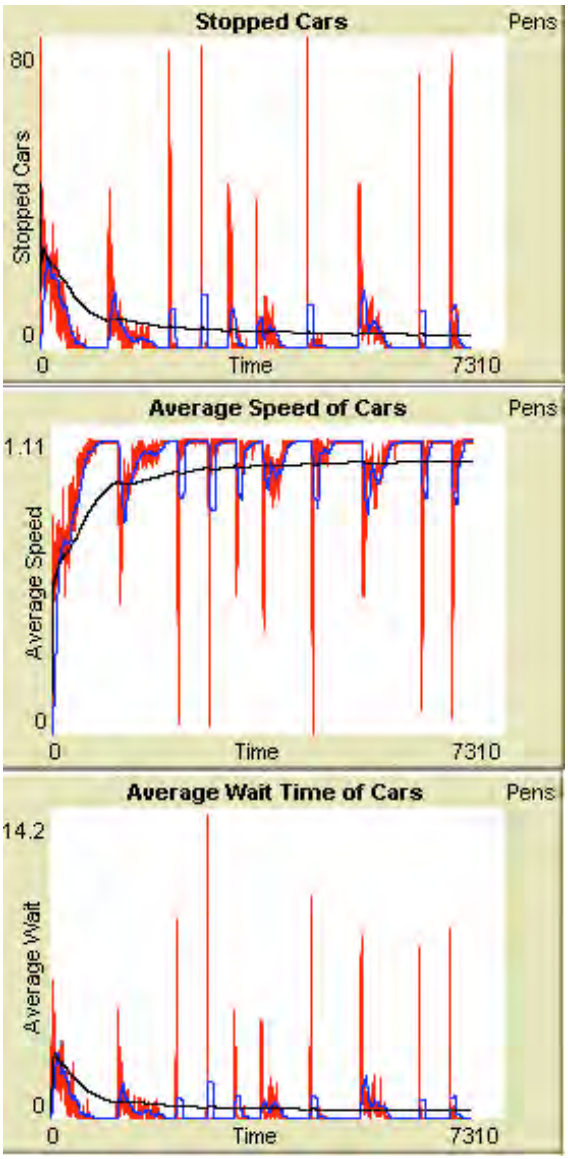


Figure 5.4: Resets of traffic lights as *sotl-phase* achieves full synchronization (80 cars in 5×5 grid, $r = 40$).

is long enough to reach the previous intersection, but not long enough to cut its tail. This will prevent waiting cars from advancing, until more cars join the long platoon. This failure could probably be avoided by introducing further restrictions in the method.

Cut-off performs better than the rigid methods, as it responds to the current traffic state (except for very low densities, when cars in streets may never reach the *cut-off* length λ). However, it is not as efficient as *sotl* methods, since cars need to stop before being able to switch a red light to green. Still, for high densities its performance is comparable to that of *sotl-phase*, performing better than the other two *sotl* methods.

With *no-corr*, we can observe that all the methods have an improvement over random phase assignation. Nevertheless, the difference between *no-corr* and rigid methods is less than the one between rigid and adaptive methods. This suggests that, for low traffic densities, adaptation is more important than "blind" correlation. For high traffic densities, the opposite seems to be the case. Still, *sotl* methods make their own correlation.

Tests with "faulty", i.e. non-correlated intersections, were performed. All methods are robust to failure of synchronization of individual traffic lights, and the global performance degrades gracefully as more traffic lights become faulty.

5.4 Applying the Methodology II

Simulation. With encouraging results, changes were made to the Simulation to make it more realistic. Thus, a scenario similar to the one of Faieta and Huberman (1993) was developed. Traffic flow in four directions was introduced, alternating streets. This is, arteries still consist of one lane, but the directions alternate: southbound-northbound on vertical roads, and eastbound-westbound on horizontal roads. Also, the possibility of having more cars flowing in particular directions was introduced. Peak hour traffic can be simulated like this, regulating the percentages of cars that will flow in different roads.[10].

An option to switch off the torus in the simulation was added. Cars that exit the simulation are removed from it. For creating new cars, gates are chosen randomly, with a probability proportional to the parameters that represent car percentages at vertical, eastbound, and southbound

[10] $\%\text{horizontal} = 100 - \%\text{vertical}$; $\%\text{westbound} = 100 - \%\text{eastbound}$; $\%\text{northbound} = 100 - \%\text{southbound}$.

roads. Then, at chosen gates, a car will be created with a probability

$$P_{\text{newc}} = 1 - \frac{c}{c_{\max}} \tag{5.2}$$

where c is the current number of cars, and $c_{\max}$ is the maximum number of cars. Note that without a torus, traffic jams are less probable, since new cars cannot be fed into the system until there is space, and also the probability of creating new cars reduces as their number approaches $c_{\max}$. Therefore, the actual number of cars will be less than $c_{\max}$.

Finally, a probability of turning at an intersection P_{turn} was included. Therefore, when a car reaches an intersection, it will have a probability P_{turn} of reducing its speed and turning in the direction of the crossing street. This can cause cars to leave platoons, which are more stable when $P_{turn} = 0$.

5.5 Experiments: Second Results

Similar sets of experiments as those presented in Section 5.3 were performed, with runs of 10,000 time steps with random initial conditions in a grid of 10×10 arteries of $r = 80$, with $p = 83$, $\theta = 41$, $\varphi_{\min} = 20$, $\rho = 8$, $\omega = 4$, $\mu = 3$, and $\lambda = 3$ (Again, see Table 5.1 for a concise description of the parameters). The percentage of cars in horizontal streets was the same as in vertical (50%), but of those, 60% in vertical roads were southbound (40% northbound) and 75% in horizontal streets were eastbound (25% westbound). $P_{\text{turn}} = 0.1$ was used. Since each street crosses 10 other streets, on average each car should turn more than once. Results of single runs, increasing the number of initial cars ($c_{\max}$ in Equation (5.2)) from 20 to 2000 in steps of 20, can be appreciated in Figures 5.5 and 5.6. It should be noted that the average number of cars is reduced as the initial density increases, since cars cannot enter the simulation until there is space for them. This reduces considerably the probability of deadlocks. A plot comparing the initial and average number of cars for the simulations can be seen in Figure 5.7.

In general terms, the improvements of the simulation did not alter the first results by much. *Marching* and *optim* are poor for low traffic densities, but the performance degrades smoothly as the density increases. There are almost no deadlocks because with high densities used in the simulation as initial states, more cars exit than enter until a "carrying capacity" of the city is reached. If this was a real city, there would be queues waiting to enter the city, which the statistics of these simulations do not consider.

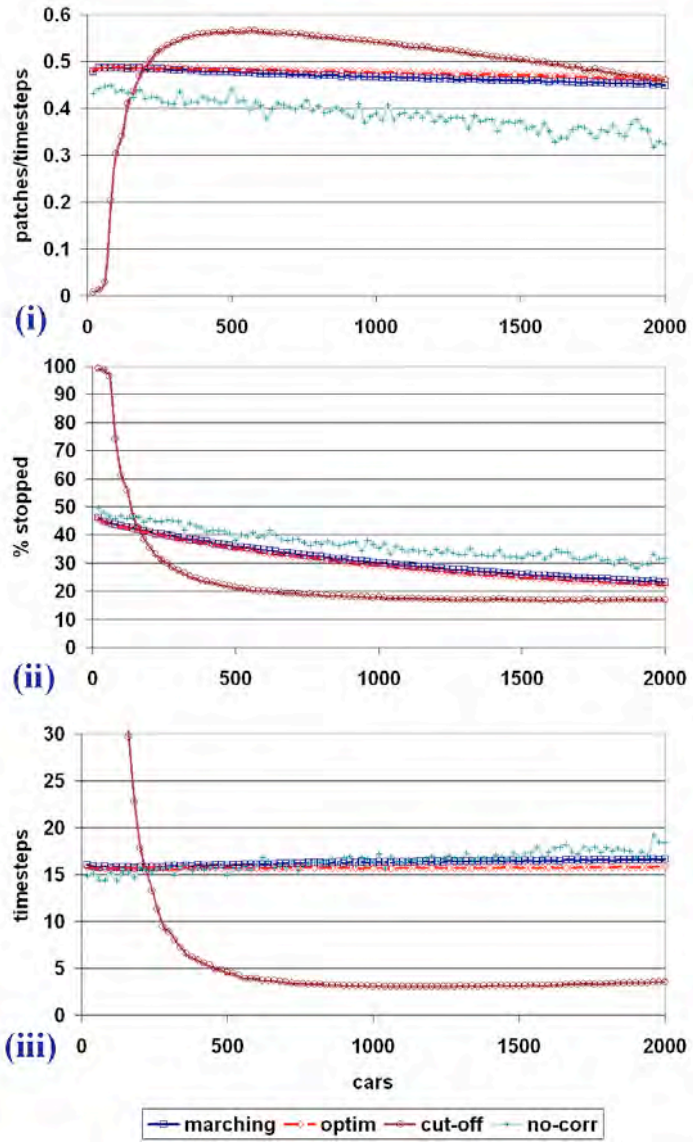


Figure 5.5: Results in four directions, turning, and without torus, for standard methods, as car density increases. (i) Average speeds of cars. (ii) Percentage of stopped cars. (iii) Average waiting times.

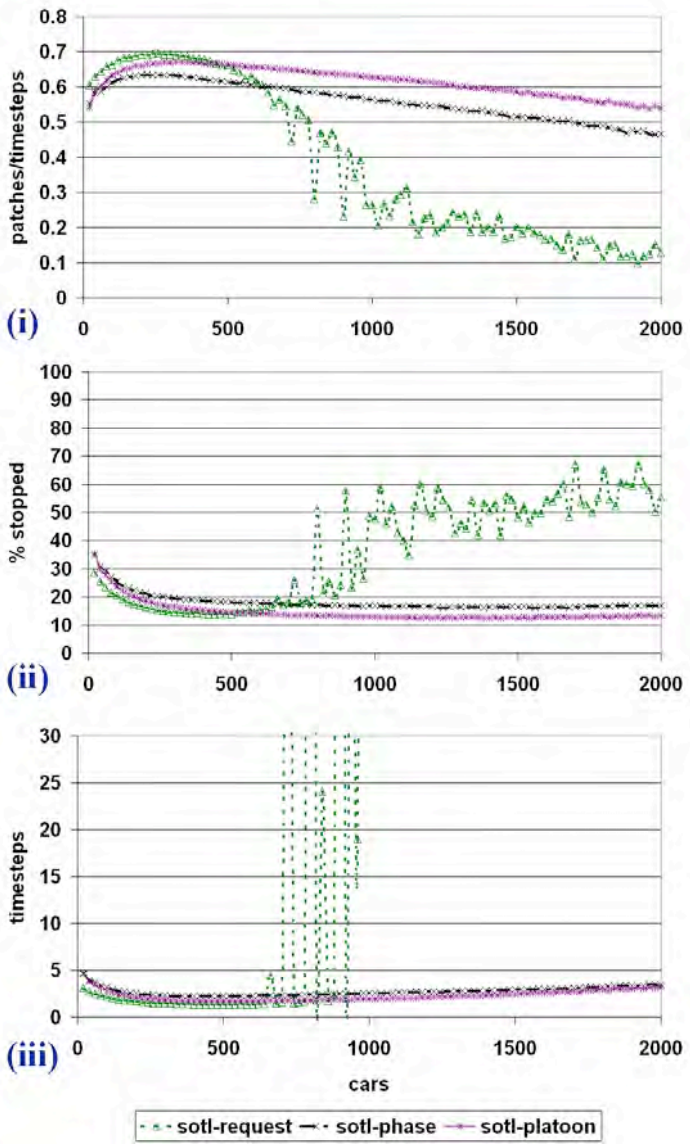


Figure 5.6: Results in four directions, turning, and without torus, for self-organizing methods, as car density increases. (i) Average speeds of cars. (ii) Percentage of stopped cars. (iii) Average waiting times.

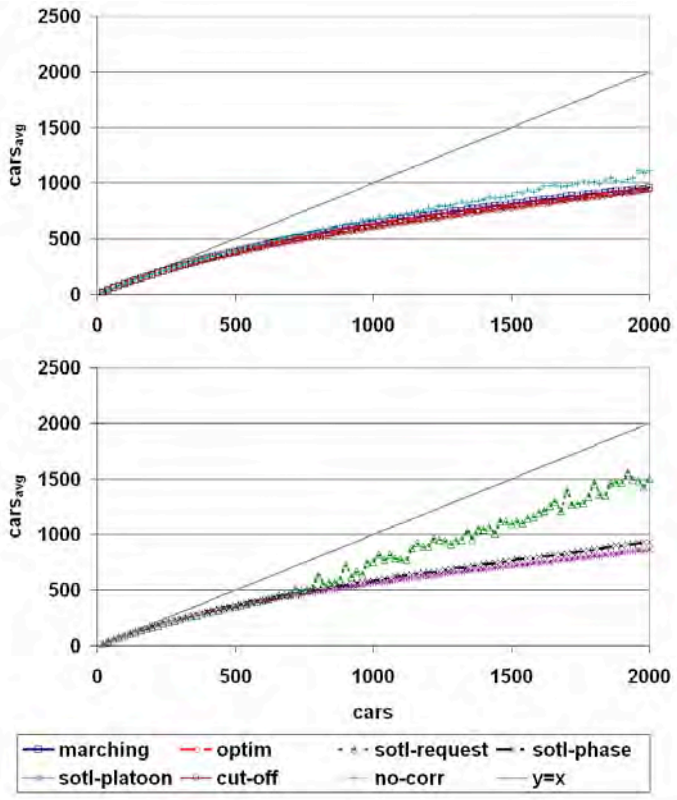


Figure 5.7: Comparison of initial and average number of cars for different methods without torus.

Sotl-request performs the best for low traffic densities, but worst for high densities, even worse than *no-corr*. This is because, as in the first results, dense platoons force the traffic lights into a constant switching, which reduces the performance.

The *sotl-phase* method avoids this problem with the restriction set by $\varphi_{\min}$. It still performs very good for low densities, and the average speed degrades slowly to a comparable performance with the non-adaptive methods. However, the percentage of stopped cars and the waiting times are much lower than the non-adaptive methods.

Sotl-platoon manages to keep platoons together, which enables them to leave the simulation faster. It gives on average 30% (up to 40%) more average speed, half the stopped cars, and seven times less average waiting times than non-responsive methods. Therefore, this method performs the best overall. It can adapt to different traffic densities, minimizing the conflicts between cars. It is not possible to achieve almost perfect performance, as it did for medium densities with a torus, since cars enter the simulation randomly. Still, this method is the one that manages to adapt as quickly as possible to the incoming traffic, effectively organizing vehicles into platoons that quickly leave the simulation, even when single vehicles might break apart from them (due to $P_{\text{turn}} > 0$).

The *cut-off* method again performs badly for very low densities. Still, afterwards it performs better than the non-adaptive methods, but not as good as *sotl-phase* or *sotl-platoon*.

Again, *no-corr* shows that all methods give an improvement over random phase assignment, except for *sotl-request* at high densities, where the method clearly breaks down.

The average number of cars, shown in Figure 5.7, can be taken as an indirect measure of the methods' performance: the faster the cars are able to leave the simulation, there will be less cars in it, thus more efficient traffic flow. An inverse correlation between the average number of cars and the average speeds can be observed. If the traffic lights can "get rid" of the incoming traffic as quickly as possible, it means that they are successfully mediating the conflicts between vehicles.

From observed simulations, the phenomenon of full synchronization is destroyed if there is no torus, or if $P_{\text{turn}} > 0$. However, it can still be achieved when the cars flow in four directions, when the number of horizontal arteries is different from the number of vertical arteries, or when the distance between arteries is not regular. It is easier to reach if there are less arteries—thus intersections—in the simulation. Also, if the length of horizontal and vertical arteries differs, that is, $r_x \neq r_y$, full synchronization is more difficult to obtain, since the periods of the platoons passing the same

traffic light depend on the length of the arteries. If these are proportional, for example, $r_x = 2r_y$, full synchronization can be achieved. Nevertheless, the *sotl-phase* and *sotl-platoon* methods achieve very good performance under any of these conditions.

5.6 Applying the Methodology III

The results mentioned above were presented at the cabinet of the Minister of Mobility and Public Works of the Brussels Region, Pascal Smet, in August 2005. The results were encouraging, but the simulation was still very abstract. The Brussels Region made available data from the Rue de la Loi/Wetstraat, a four-lane westwards one-way avenue in Brussels which gathers heavy traffic towards the center of the city. For his BSc thesis, my student Seung Bae Cools extended an open source traffic simulator to test the *sotl* methods in a more realistic simulation (Cools, 2006). His results are summarized below.

Simulation. Our simulator moreVTS (2006) (A More Realistic Vehicle Traffic Simulator) is the third of a series of open source projects building on the previous one, developed in Java. Green Light District (GLD (2001)) was developed by the Intelligent Systems Group at the University of Utrecht (Wiering et al., 2004). Then, GLD was improved by students in Argentina within the iAtracos project, which we used as a starting point for our simulator, which introduces realistic physics into the simulation. Among other things, acceleration was introduced, and the scale was modified so that one pixel represents one meter and one cycle represents one second.

The simulator allows the modeling of complex traffic configurations, allowing the user to create maps and then run simulations varying the densities and types of road users. Multiple-lane streets and intersections can be arranged, as well as spawn and destination frequencies of cars. For implementation details of moreVTS, please refer to Cools (2006).

The *sotl* methods were implemented in moreVTS. With data provided by the Brussels Capital Region, we were able to build a detailed simulation of the Wetstraat. We used the measured average traffic densities per hour on working days for 2004 (shown in Table 5.2) and the current "green wave" method—an adaptation of *optim*—which has a period of 90 seconds, with 65 seconds for the green phase on the Wetstraat, 19 for the green phase on side streets, and 6 for transitions. Like this, we were able to compare our self-organizing controllers with a standard one in a realistic setting. Figure 5.8 shows the simulation view of the Wetstraat and its

0	1	2	3	4	5	6	7
476	255	145	120	175	598	2933	5270
8	9	10	11	12	13	14	15
4141	4028	3543	3353	3118	3829	3828	3334
16	17	18	19	20	21	22	23
3318	3519	3581	3734	2387	1690	1419	1083

Table 5.2: Average vehicle count per hour at the beginning of the Wetstraat. Data kindly provided by the Brussels Capital Region.

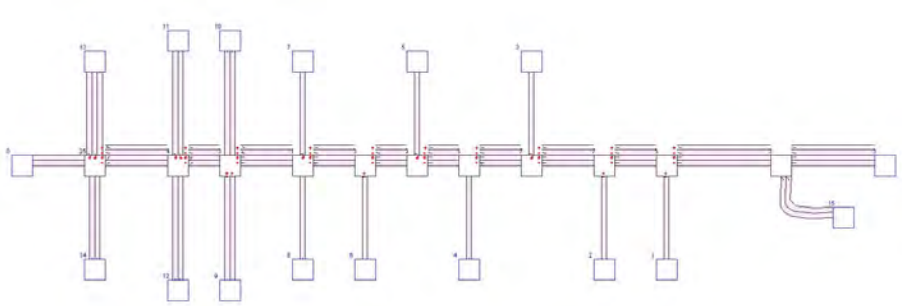


Figure 5.8: Simulation of the Wetstraat and intersecting streets. Cars flow westward on the Wetstraat. Red dots represent traffic lights for each incoming lane at intersections.

surrounding streets.

The data from Table 5.2 is for the cars entering the Wetstraat on the East (at the bridge over the Etterbeeksesteenweg), so the spawn rates for the two nodes in the simulation representing this were set according to these data. For the other nodes, the spawn and destination frequencies were set based on a field study we performed on May 2006, comparing the percentage of cars that flow through the Wetstraat and those that flow through side streets, enter, or leave the Wetstraat. These percentages were kept constant, so that when the density of cars entering the Wetstraat changed, all the other spawn rates changed in the same proportion. On average, for each five cars flowing through a side street, one hundred flow through the Wetstraat. This is not the case of the Kunststraat, a two way avenue at the West of the Wetstraat (second and third crossing streets from left to right on Fig. 5.8), where for 100 cars driving through the Wetstraat, about

40 turn right, 40 turn left, and only 20 go straight, while 20 more drive through the Kunststraat (about 10 in each direction). The precise spawn rates and destination frequencies are given in Cools (2006, pp. 55–57).

5.7 Experiments: Third Results

Since *sotl-platoon* was again the best of the *sotl* methods, only this method is discussed here. To measure the performance of the current green wave method and *sotl-platoon*, we used the average trip waiting times ($ATWT$). The travel waiting time for one car is the travel time minus the minimum possible travel time, i.e. travel distance divided by the maximum allowed speed, which for the Wetstraat simulation is about sixty seconds.

Several simulation runs were performed to find the best parameters for *sotl-platoon*. For each parameter and traffic density, five simulation runs representing one hour, i.e. 3600 cycles, were averaged. The results were robust and consistent, with *sotl-platoon* performing better than the green wave method for a wide range of parameters θ and $\varphi_{\min}$ (Cools, 2006). Only the best ones are shown in Figure 5.9, together with the results for the green wave method. The cruise speed used was 14 m/s, $\omega = 25$ and $\mu = 3$. Since some densities from Table 5.2 are very similar, we averaged and considered the same densities for 2:00, 3:00 and 4:00; 8:00 and 9:00; 10:00, 17:00 and 18:00; 11:00, 15:00 and 16:00; 13:00, 14:00 and 19:00; and 21:00 and 22:00.

As Figure 5.9 shows, there is considerable reduction in $ATWT$ using *sotl-platoon* instead of the current green wave method. The $ATWT$ for the densities at different hours using *sotl-platoon* were from 34% to 64% of the $ATWT$ for the green wave method, and on average for different densities 50%. Since the minimum travel time for the Wetstraat is about one minute, while the overall ATWT for the green wave method is also about one minute and for *sotl-platoon* about half, the improvement in the average total travel times would be of about 25%, i.e. cars under a green wave method would take 33% more time to reach their destination than those under *sotl-platoon*. This shows with a realistic simulation that *sotl-platoon* improves greatly traffic flow compared to the current green wave method, for all the studied traffic densities.

We should note that these results are consistent with those presented in Section 5.5, even when there are several differences in the traffic configurations. With a green wave method, such as *optim*, cars need to wait on average half their total travel times (less with low density, more with high density). On the other hand, with *sotl-platoon* cars need to wait on

average only one third of their total travel times, saving considerable time and fuel.

We have observed that there is a monotonic relation between the best θ and the traffic density (Cools, 2006). Exploring this relation better could allow us to set a variable θ depending on the current traffic density measured by the traffic lights. Still, since *sotl-platoon* performs very well for a broad range of parameters, it does not require their precise calculation. In other words, *sotl-platoon* is not sensitive to small changes in parameters, making it a robust method.

5.8 Applying the Methodology IV

Representation. If priority is to be given to certain vehicles (e.g. public transport, emergency), weights can be added to give more importance to some σ_i's. The σ_{sys} would still be the average of the individual σ_i's, only now weighted by the priority of vehicles.

A meso-level might be considered, where properties of platoons can be observed: their behaviors, performance, and satisfaction and the relationships of these with the vehicle and city levels could enhance the understanding of the self-organizing traffic lights and even improve them.

Application. The proposed system has not been implemented yet. Still, it is feasible to do so, since there is the sensor technology to implement the discussed methods in an affordable way, and the simulations have proved the benefit of the self-organizing methods.

Figure 5.10 shows one possible configuration for implementing *sotl-platoon* using simple proximity sensors, which can be installed under the pavement. More complicated sensors, such as cameras, could also be used. A sensor at a distance ρ from the traffic light would be used to start counting cars. A controller would know how many cars are approaching a red light by incrementing a counter c each time a car passes through the sensor at ρ, and decrementing c when a car crosses the intersection. Every certain time, say one second, c would be integrated in counter κ_i (see Algorithm 3). The same idea would be used to count how many cars are between ω and the intersection. If cars would enter or leave the street somewhere between a sensor and the intersection, e.g. a parking or small street, an additional sensor should be installed to adjust c.

Pedestrians could be taken into account considering them as cars approaching a red light. Buttons like the ones already available in the market can be used to include pedestrians in κ_i, resetting them after each change of light.

A pilot study should be made before applying widely self-organizing traffic lights, to fine tune different parameters and methods. External factors, e.g. pedestrians and cyclists, could also affect the performance of the system.

A mixed strategy between different methods could be considered, e.g. *sotl-platoon* for low and medium densities, and *sotl-phase* or *marching* for high densities. Another alternative worth exploring would be to vary θ according to the current traffic density, or to test different θ's for different directions for highly skewed traffic densities, such as in the Wetstraat. The relationship between θ with ρ and ω should also be studied.

Evaluation. If a city deploys a self-organizing traffic light system, it should be monitored and compared with previous systems. This will help to improve the system. If the system would be an affordable success, its implementation in other cities would be promoted, especially because the mechanisms and simulators are open to the community.

5.9 Discussion

"The environment is not best conceived solely as a problem domain to be negotiated. It is equally, and crucially, a resource to be factored in the solutions."
—Andy Clark

The *sotl* methods follow ideas of decentralized control similar to the ones used by Porche and Lafortune (1999, and references within), but with a much simpler implementation. There is no costly prediction of arrivals at intersections, and no need to establish communication between traffic lights to achieve coordination. They do not have fixed cycles.

The series of experiments performed show that *sotl* strategies are more efficient than traditional control methods. This is mainly because they are "sensitive" and adaptive to the changes in traffic. Therefore, they can cope better with variable traffic densities, noise, and unpredicted situations. Based on our results, we can say the following.

- The formation of platoons can be seen as a reduction of variety (Ashby, 1956, Ch. 11). It is much easier to regulate 10 groups of 10 cars than 100 cars independently.[11] Platoons make the traffic problem simpler. Oscillations in traffic will be reduced if cars interact as groups. We can also see this as a reduction of entropy: if cars are homogeneously spread on the street grid, at a particular

[11] This could be seen as "functional" modularity (Simon, 1996, pp. 188-195).

moment there is the same probability of finding a car on a particular block. This is a state of maximum entropy. However, if there are platoons, there will be many blocks without any car, and a few with several. This allows a more efficient distribution of resources, namely free space at intersections.[12] It is interesting to note that the *sotl* methods do not force vehicles into platoons, but induce them. This gives the system flexibility to adapt.

- We can say that the *sotl* methods try to "get rid" of cars as fast and fair as possible. This is because they give more importance to cars waiting for more time compared to recent arrivals, and also to larger groups of cars. This successfully minimizes the number of cars waiting at a red light and the time they will wait. The result is an increase in the average speeds. Also, the prompt "dissipation" of cars from intersections will prevent the formation of long queues, which can lead to traffic jams.

- Since cars share a common resource—space—they are in competition for that resource. Self-organizing traffic lights are synergistic (Haken, 1981), trying to mediate conflicts between cars. The formation of platoons minimizes friction between cars by Imposition, leaving free space around them. If cars are distributed homogeneously in a city, the probability of conflict increases.

- There is no direct communication among the self-organizing traffic lights. However, they "exploit" cars to stigmergically transmit information,[13] in a way similar to social insects exploiting their environment to coordinate. For traffic lights, car densities form their environment. Traffic lights respond to those densities. But cars also respond to the traffic light states. We could say that traffic lights and cars "co-control" each other, since cars switch traffic lights to green, and red traffic lights stop the cars.

5.9.1 Adaptation or optimization?

Optimization methods are very effective for problems where the domain is fairly static. This enables the possibility of searching in a defined space. But in problems where the domain changes constantly, such as traffic, an

[12] The formation of platoons has already been proposed for freeways, with good results (Sheikholeslam and Desoer, 1991).

[13] For an introduction to stigmergy, see (Theraulaz and Bonabeau, 1999).

adaptive method should be used, to cope with these changes and constantly approach solutions in an active way.

The problem of traffic lights is such that cars and traffic lights face different situations constantly, since they affect each other in their dynamics (i.e., traffic lights affect cars, cars affect cars). With *sotl* methods, cars affect traffic lights and traffic lights affect other traffic lights stigmergically via the cars. If the situation is unknown or unpredictable, it is better to use an adaptive, self-organizing strategy for traffic lights, since it is not computationally feasible to predict the system behavior.[14]

We can see an analogy with teaching: a teacher can tell exactly a student what to do (as an optimizer can tell a traffic light what to do). But this limits the student to the knowledge of the teacher. The teacher should allow space for innovation if some creativity is to be expected. In the same way, a designer can allow traffic lights to decide for themselves what to do in their current context. Stretching the metaphor, we could say that the self-organizing traffic lights are "gifted with creativity," in the sense that they find solutions to the traffic problem by themselves, without the need of the designer even understanding the solution. On the other hand, non-adaptive methods are "blind" to the changes in their environment, which can lead to a failure of their rigid solution.

We can deduce that methods which are based on phase cycles, and even adaptive cyclic systems (Hunt et al., 1981; Sims, 1979) (i.e., systems that try to coordinate phases with fixed durations) will not be able to adapt as responsively as methods that are adaptive and non-cyclic, since they are not bounded by fixed durations of green lights (Porche and Lafortune, 1999). Therefore, it seems that optimizing phases of traffic lights is not the best option, due to the unpredictable nature of traffic.

All traffic lights can be seen as mediators (Heylighen, 2003a) among cars. However, rigid methods do not take into account the current state of vehicles. They are more "autocratic." On the other hand, adaptive methods are regulated by the traffic flow itself. Traffic controls itself, mediated by "democratic" adaptive traffic lights.

5.9.2 Practicalities

There are many parallel approaches trying to improve traffic. We do not doubt that there are many interesting proposals that could improve traffic,

[14] This is because there is a high sensitivity to initial conditions in traffic, that is, chaos: if a car does not behave as expected by a non-adaptive control system, this can lead the state of the traffic far from the trajectory expected by the system.

for example, to calculate real-time trajectories of all cars in a city depending on their destination via GPS. However, there are the feasibility and economic aspects to take into account. Two positive points in favor of the self-organizing methods are the following. First, it would be very easy and cheap to implement them. There are already sensors on the market which could be deployed to regulate traffic lights in a way similar to *sotl-phase*. Sensors implementing the *sotl-platoon* method would not be too difficult to deploy, as shown in Figure 5.10. Second, *sotl* methods could be introduced gradually in a city, adapting seamlessly to the existing network. The system does not need to be implemented completely to start working and giving results. Moreover, there is no need for a central computer, expensive communication systems, or constant management and maintenance. The methods are robust, so they can resist incrementally the failure of intersections.

Self-organizing traffic lights would also improve incoming traffic to traffic light districts, for example, from freeways, since they adapt actively to the changing traffic flows. They can sense when more cars are coming from a certain direction, and regulate the traffic equitably.

Pedestrians could be included in a self-organizing scheme by considering them as cars approaching a red light. For example, a button could be used, as is now common, to inform the intersection, and this would contribute to the count κ_i. Certainly, only one pressing would be counted per cycle, to prevent its malicious use.

Vehicle priority could also be implemented, by simply including weights w_j associated to vehicles, so that the count κ_i of each intersection would reach the threshold θ counting $w_j c * ts$. However, this would require a more sophisticated sensing mechanism, although available with current technology for priority vehicle detection, e.g. RFID tags. Still, this would provide an adaptive solution for vehicle priority, which in some cities (e.g. buses in London) can cause chaos in the rest of the traffic lights network, since lights are kicked off phase or they are set according to the priority vehicles (e.g. trams in Brussels).

We should also note that traffic lights are not the best solution for all traffic situations. For example, roundabouts (Fouladvand et al., 2004a) are more effective in low speed, low density neighborhoods.

5.9.3 Environmental benefits

Even when cars consume more fuel when they are moving than when they are stopped, idle engines can produce large amounts of pollution. To il-

lustrate this, we calculated the potential environmental benefits of implementing *sotl-platoon* on the Wetstraat in Brussels, summarized in Table 5.3.

	Optim	SOTL	Difference
Day ATWT (s)	54.63	27.85	26.77
Day WT (hr)	908.61	463.28	445.32
Litres gas in WT	3089.26	1575.17	1514.09
l/year	1127579.85	574935.97	552643.88
CO2 tons/year	2706.19	1379.85	1326.35
NOx kg/year	9053.41	4616.2	4437.21
CO kg/year	136126.8	69409	66717.79
CxHx kg/year	18237.08	9298.81	8938.27

Table 5.3: Emissions by idling engines on Wetstraat, with information from the U.S. Environmental Protection Agency. See text for explanation.

Averaging 59877 cars per day, our simulations averaged about one minute of ATWT for the current method and half of that for *sotl-platoon*. Just a few seconds per car, but they amount to more than nine hundred hours a day, half for *sotl-platoon*. An idling engine spends on average 3.4 liters of gasoline per hour, so every day on the Wetstraat cars spend more than three thousand liters of gasoline per day and more than a million per year. Since *sotl-platoon* would consume only half of that, it would save more than a thousand tons of carbon dioxide, more than four tons of oxides of nitrogen, more than sixty six tons of carbon monoxide, and almost nine tons of hydrocarbons, every year. And this is taking into consideration only ten intersections. How many intersections are there in the world?

Implementing self-organizing traffic lights would certainly help countries fulfill emission reduction commitments under the Kyoto protocol, so their benefit goes beyond reducing travel times.

Moreover, saving half a million liters of fuel per year would imply also a saving of more than half a million euros per year, which would be less than the cost of implementing such a system on the Wetstraat.

5.9.4 Unattended issues

The only way of being sure that a self-organizing traffic light system would improve traffic is to implement it and find out. Still, the present results are encouraging to test our methods in more realistic situations.

A future direction worth pursuing would be a systematic exploration of the parameters θ, p, and $\varphi_{\min}$ values for different densities, as well as the exploration of different environmental parameters. A meta-adaptive method for regulating these parameters depending on the traffic densities would be desirable. Therefore, if a certain density is detected, proper parameter values could be used. It would also be interesting to compare our methods with others, for example, (Hunt et al., 1981; Sims, 1979), but many of these are not public, or very complicated to implement in a reasonable amount of time. Reinforcement learning methods (Wiering et al., 2004) will adapt to a particular flow density. However, in real traffic densities change constantly and unevenly. We should compare the speed of adaptation of these methods with the proposed self-organizing ones. We would also like to compare our methods with other distributed adaptive cyclic methods, e.g. Faieta and Huberman (1993); Ohira (1997) (*sotl* and *cut-off* are non-cyclic), to test if indeed phase cycles reduce the adaptability of traffic lights. It would also be interesting to compare *sotl-platoon* with the Dresden method (Helbing et al., 2005; Lämmer et al., 2006), which couples oscillators using self-organization, whereas *sotl-platoon* has no internal phases nor clocks. A comparison with methods inspired by game theory (Bazzan, 2005; Oliveira et al., 2005, 2006) is also desirable.

Another direction worth exploring would be to devise methods similar to the ones presented that promote "optimal" sizes of platoons for different situations. We would need to explore as well which platoon sizes yield less interference for different scenarios.

5.10 Conclusions

We have presented three self-organizing methods for traffic light control which outperform traditional methods due to the fact that they are "aware" of changes in their environment, and therefore are able to adapt to new situations. The methods are very simple: they give preference to cars that have been waiting longer, and to larger groups of cars. Still, they achieve self-organization by the probabilistic formation of car platoons. In turn, platoons affect the behavior of traffic lights, prompting them to turn green even before they have reached an intersection. Traffic lights coordinate stigmergically via platoons, and they minimize waiting times and maximize average speeds of cars. Under simplified circumstances, two methods can achieve robust full synchronization , in which cars do not stop at all.

From the presented results and the ones available in the literature

(Porche and Lafortune, 1999), we can see that the future lies in schemes that are distributed, non-cyclic, and self-organizing. In the far future, when autonomous driving becomes a reality, new methods could even make traffic lights obsolete (Gershenson, 1998a; Dresner and Stone, 2004), but for the time being, there is much to explore in traffic light research.

There are several directions in which our models could be improved, which at the present stage might be oversimplifying. However, the current results are very promising and encourage us to test self-organizing methods in real traffic environments, starting with pilot studies. However, we would not like to motivate even more the use of cars with an efficient traffic control, since this would increase even more traffic densities and pollution. Any city aiming at improving its traffic flow should promote in parallel alternative modes of transportation, such as cycling, walking, car pooling, or using public transport.

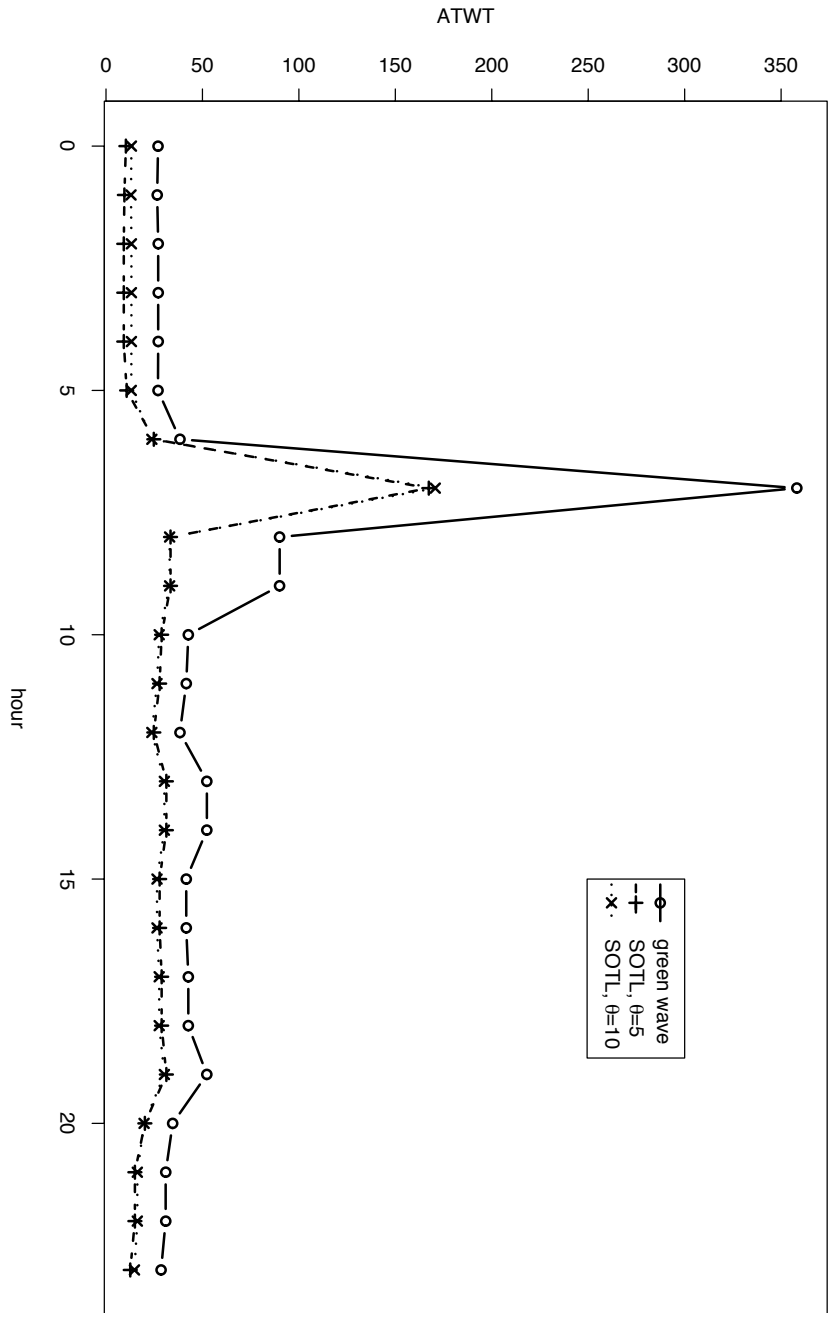


Figure 5.9: Average trip waiting times ($ATWT$) at different hours of the day, green wave controller and *sotl-platoon* controller with $\varphi_{min} = 5$ and $\theta = 5; 10$.

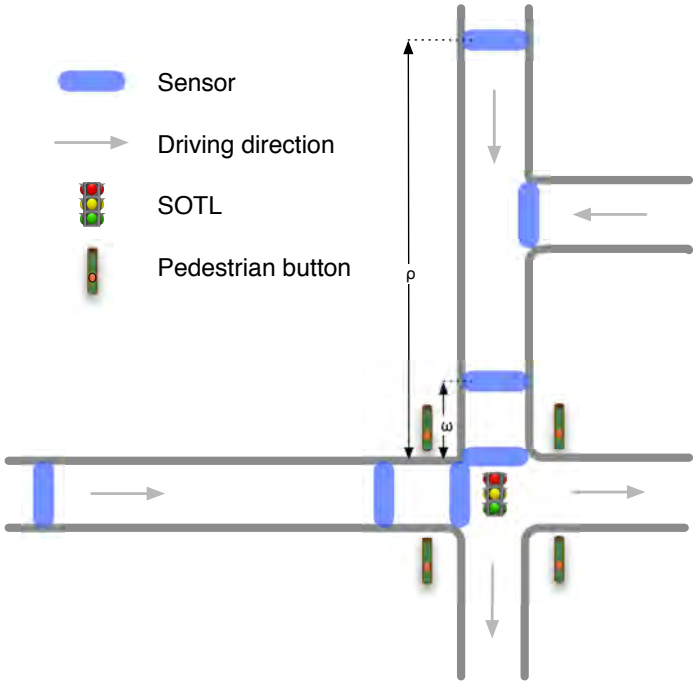


Figure 5.10: Potential implementation of *sotl-platoon*.

Chapter 6

Self-organizing Bureaucracies

The goal of this chapter[1] *is to encourage the use of self-organization as a method to improve the efficiency and adaptability of bureaucracies and similar social systems. Bureaucracies are described as networks of agents, where the main design principle is to reduce local "friction" to increase local and global "satisfaction". Following this principle, solutions are proposed for improving communication within bureaucracies, sensing public satisfaction, dynamic modification of hierarchies, and contextualization of procedures. Each of these reduces friction between agents (internal or external), increasing the efficiency of bureaucracies. "Random agent networks" (RANs)—novel computational models—are introduced to illustrate the benefits of self-organizing bureaucracies.*

[1] Based on Gershenson (2006b).

6.1 Introduction

"Hay que parar al que no quiera
que el pueblo gane esta pelea
Hay que juntar toda la ciencia
antes que acabe la paciencia"[2]
—Stafford Beer and Ángel Parra

Bureaucracies can be found in governments, corporations, and other social institutions. They have social goals and responsibilities that are achieved by a division of labor that is usually hierarchical. Examples of bureaucracies can be seen in the public and private sectors, e.g. tax collection systems, immigration services, and steering of educational and academic institutions. The efficiency of a bureaucracy is related to the fulfillment of its goals. Thus, it would be desirable to increase functional efficiency in bureaucracies. Ideally, such a system could be designed to reach maximal efficiency. In practice, as most people have experienced, this is far from being the case (Weber, 1968). Corruption, rigidity, and delays are just few examples of obstacles that hamper efficiency in bureaucracies. It would be naïve to aim for perfect bureaucracies, but certainly the efficiency of actual ones can be improved.

One approach would consist of *optimizing* the bureaucratic functionality, e.g. Hofacker and Vetschera (2001). This approach can provide good solutions if the function or goal of the organization does not change considerably, i.e. when a problem domain is *stationary*. However, the world is changing at accelerating rates. Changes cause the shifting of the optimum of a system. And in some cases, the behavior of the institution itself changes the optimum (Kauffman, 2000). Thus, a wiser approach would be to design bureaucracies that are able to *adapt* (Holland, 1975) to changing situations. Instead of attempting to predict all the functionality beforehand, an organization could adapt to the changing demands of its environment.

Cybernetics and systems theory proposed some of the first solutions in this direction already a few decades ago, e.g. Beer (1966). The Cybersyn project in Chile was even partially implemented, but was cut short by the 1973 military coup (Miller Medina, 2005).[3] However, this approach is still

[2] We have to stop those who don't want / the people to win this fight / We have to gather all science / before we run out of patience.

[3] One aspect of the Cybersyn system implemented a "nervous system" for the country, where every day information with the productions and demands from factories was telexed to a "brain" room where people decided which demands were more urgent and

not widely used in practice, probably because it requires alternative ways of thinking, as exposed in Chapter 2. It is always easier to solve problems for stationary domains than for dynamic environments.

Organization science has developed several concepts that are useful for improving the self-organization and adaptation of bureaucracies. Noting the cognitive limits of decision makers (March, 1978; Simon, 1982; Cyert and March, 1992) tells us that individuals will not be able to make perfect decisions. On one hand, the cognition necessary to solve complex tasks can be distributed (Hutchins, 1995; Weick and Roberts, 1993). On the other hand, organizations need to be able to adapt to unpredictable events (Carley, 1997). Organizational learning is one type of adaptation that has been widely studied (Levitt and March, 1988; March, 1991). Also, computational organization theory (Carley and Prietula, 1994) and agent based modeling, e.g. Epstein and Axtell (1996); Axelrod (2005), have aided in the understanding of the complexity inherent to organizations (Anderson et al., 1999; Anderson, 1999; Levinthal and Warglien, 1999; Lissack, 1999; Axelrod and Cohen, 2000)

Following this line of research, this chapter suggests methods that improve the efficiency of bureaucracies via self-organization[4]. Self-organization (Heylighen, 2003b) has been used as a principle in many domains such as computer science and robotics (Kohonen, 2000), the Internet (Bollen and Heylighen, 1996), and traffic light control (Gershenson, 2005), just to name a few. In organization science, self-organization has been studied as a phenomenon, e.g. Comfort (1994); Morgan (1996, p.233). The goal of this chapter is to use it conceptually as a *tool* to improve the efficiency of bureaucracies.

In the next section, the application of the Methodology to bureaucracies is discussed broadly. In the following sections, different aspects of bureaucracies and improvements using self-organization are proposed, namely in the areas of communication, sensors, hierarchies, and context. Afterwards, random agent networks (RANs) are defined to model bureaucracies. Computational experiments with RANs illustrate the benefits of self-organization for improving the performance of abstract bureaucracies. The chapter is aimed more at government bureaucracies, but in principle the ideas could be applied to business bureaucracies as well.

route the productions accordingly. Like this, the government was able to stock the country in spite of widespread strikes and scarcity.

[4]For an introduction to self-organization, see Section 3.5.

6.2 Designing Self-organizing Bureaucracies

"Everything is about balance. The problem is that one aspect seeking balance may unbalance other aspects"

Organizations can be modeled as systems of information processing agents (Radner, 1993; Van Zandt, 1998; DeCanio and Watkins, 1998). An agent is a description of an entity that *acts* on its environment. They could also be described as *cognitive* systems (Gershenson, 2004a). Thus, not only people can be described as agents, but also departments, ministries, and governments. Agents can have goals (Simon, 1964), that are described by an observer. Agents can be said to be cognitive because they need to "know" which actions to take to reach their goals. Following the conceptual framework proposed in Section 4.2, the "satisfaction" of the agents will be related to the achievement of their goals. Thus, a description of a bureaucracy can be made in terms of agents trying to fulfill goals to increase their satisfaction. The public can also be described as an agent or several agents, interacting (externally) with the bureaucracy.

However, the satisfaction of one agent (e.g. a clerk) can be in conflict with the satisfaction of another agent (e.g. the minister). As the main premise of the Methodology states, it can be argued that decreasing the "*friction*" or interference of agents at one level (e.g. personal level), i.e. how one agent decreases the satisfaction of another agent, will result in an increase of satisfaction at the higher level (e.g. ministry level) (Gershenson, 2006a; Helbing and Vicsek, 1999). This is a (useful) tautology because the goals of agents are described by observers according to the desired function of the system. Notice that this is different from implying that maximizing the satisfaction of agents at one level will always lead to an increase of satisfaction at the higher level. The key difference lies in the *mediation* of conflicting goals to increase satisfaction. Similarly, we can speak about "negative friction", or *synergy* (Haken, 1981), where the behavior of one agent increases the satisfaction of another agent. Certainly, not only friction should be minimized, but also synergy maximized. Different ways in which this can be achieved are discussed in detail in Section 4.3.2. Within practical limits, friction reduction and synergy promotion will be always useful, since all actors, internal or external, will benefit.[5] For example, the easier it is to pay taxes, the more people will be motivated to pay them. This benefits the state (more votes on next election, more money

[5]Note that high costs cause friction, so self-organization should always take cost into account, i.e. not to promote changes with a cost higher than their benefit.

collected) and taxpayers (less time lost). Certainly, as in most social systems, a problem will remain when it comes to measuring satisfaction. This will be discussed in Section 6.4.

The goals of a firm can be easily related to its profits (Van Zandt, 1998). However, the goals of a bureaucracy are related to its particular function. This function can be co-determined by the state, by the public, and by the bureaucracy itself. Thus, there is as yet no general way to measure the performance of a bureaucracy. Efficiency could be a way of evaluating a bureaucracy, but there is the same measurement problem with efficiency: it will differ according to the particular bureaucracy. Still, a lack of explicit descriptions of function, efficiency, or satisfaction are not a limitation for speaking about the goals of a bureaucracy. It should just be considered that these can change depending on the behavior of the bureaucracy itself.

In order to adapt to unpredictable changes, bureaucracies require a certain flexibility. Changes should be made, but the function needs to be preserved. Robustness is required (Jen, 2005), so that adaptive changes do not prevent the bureaucracy from reaching its goals. The main idea to guide changes is the following: First, detect how each agent affects satisfaction of others. Then, implement changes to minimize friction and promote synergy. This can be achieved by reinforcement: behaviors that have proven themselves inefficient should be avoided, and beneficial ones should be promoted. This can be seen as a particular case of *learning* (Levitt and March, 1988; March, 1991).

Before implementing radical changes that might lead to unexpected outcomes, computer simulations should be used (Gershenson, 2002d; Axelrod, 2005). These will be useful for detecting possible flaws in the changes planned, or simply to improve them. Moreover, the changes themselves can be explored with the aid of computer simulations, since it is not obvious in every case what should be done, as the complexity of bureaucracies often exceeds our predictive capabilities. Simulations are important because the behavior of highly complex systems cannot be predicted beforehand, but it should be first observed and then explained. Moreover, simulations allow observation without potential risks.

The changes could be introduced gradually and with a certain redundancy to compare the benefits and disadvantages of the new methods with the previous ones, with the possibility to return to previous situations.

In the following sections diverse aspects of bureaucracies are explored, suggesting different possible improvements within each domain. Each improvement would contribute to the performance of a bureaucracy in a different manner, so their application could be considered combined or in isolation.

6.3 The Role of Communication

"All human relationships are based on misunderstandings"

Communication between agents can be classified in two categories: synchronous and asynchronous (Desanctis and Monge, 1999).

Synchronous communication occurs when the agents involved in the process are responding at the same time. There is immediate feedback between speaker and listener, so that a dialog can be established continuously. Examples of this are verbal communication, telephone conversations, video conferencing, IRC (Internet relay chat), and VoIP (Voice over Internet Protocol, e.g. Skype). The advantage of this mode of communication is that dialogs can be resolved without interruption. The disadvantage is that all participants need to coordinate to participate in the process.

Asynchronous communication occurs when the agents involved do not participate simultaneously in the process. There is a delayed feedback between agents, so that dialogs are interrupted depending on the length of the transmission delay. Examples include post, telegraph, telex, fax, e-mail, and instant messaging, and SMS. The trade-offs of this mode of communication complement those of synchronous communication: on the positive side asynchronous communication allows exchanges without coordination required, but on the negative side the communication can be delayed. Technological development has reduced transmission delay, enabling asynchronous communication to depend only on the constraints of the agents. Moreover, asynchronous communication allows for certain decision time, whereas in synchronous communication most responses should be immediate, reducing the possibility of digesting information properly.

Technology seems to lead to a convergence of synchronous and asynchronous communication into *semisynchronous* communication, where media can be used synchronously or asynchronously depending on the circumstances. This is the case with e-mail, instant messaging, IRC, and SMS: instant replies can lead to a conversation, but the information persists in case one of the parties is not able to respond immediately.

In a bureaucracy, different agents need to communicate to satisfy the goals of the system. Thus, communication delays can be seen as a type of friction between agents. The faster the communication takes place, the better it will be for the system. Thus, synchronous communication might be preferred to enable quick responses. However, this would imply a great coordination effort, since agents usually perform other activities apart from communicating. It could be quite possible that an agent would

be too busy with other matters to have a synchronous exchange. Then it seems that asynchronous communication would be preferred, since one agent can send a message and keep on working on other matters while the response arrives. Then the question would be: how can asynchronous communication be facilitated?

As mentioned above, one great improvement is given by technology. Being able to send documents electronically instead of by post reduces the delay of message transmission from the scale of days to the scale of seconds. Certainly, organizations have exploited this opportunity, and worries about security have been solved with digital signatures. Still, it is a common practice in several bureaucracies to handle paper documents, even when they must be sent across continents, as it is the case e.g. with the Mexican foreign services, having a transmission delay of weeks instead of seconds.

However, the adoption of electronic means of communication can do much more than reducing the transmission delay of messages. Analyzing the times when a message is sent and when it is replied to can provide very useful information, namely that of response delay (see Figure 6.1). This can be used to detect bottlenecks: If one agent (individual or department) takes too long in replying to requests, the work of other agents might be delayed as well, as in a production chain. Resources could then be reassigned in real time to overcome the bottleneck, giving priority to the agent with the response delay. In fact, we can say that a delay in response causes friction to other agents, since they need the feedback to reach their goals. Thus, a bureaucracy could self-organize by modifying in real time its own structure, once it is known where friction is coming from. Solutions can vary depending on the precise nature of the delay: assign more individuals to a department, replace individual(s), or reorganize departments. Like this, efficiency of the bureaucracy would be improved. It would be self-organizing, because the changes are dictated by the behavior of the bureaucracy itself. The changes would imply *learning* in the organization from its experience, while enabling it to adapt constantly to changes of its environment.

The response delay would depend on several factors: decision delay (the time it takes the agent to perform a task), delay from previous tasks (the time it takes an agent to start making a decision), and delay from other responses (the time it takes other agents to respond to the agent's requirements) (see Figure 6.2). In other words, the time it takes an agent to respond depends not only on the time necessary to perform a task (decision delay), but also the time it takes to finish previous requests (delay from previous tasks), and the time it needs to wait for other agents to com-

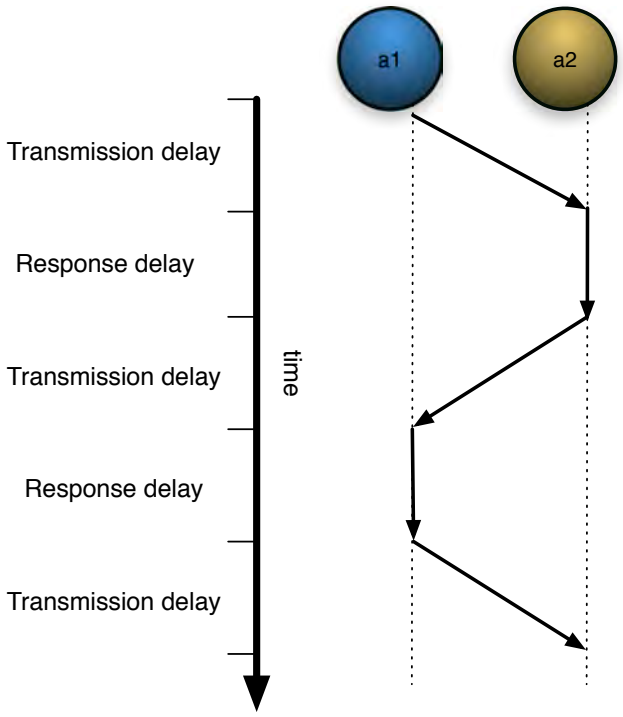


Figure 6.1: Asynchronous communication. Technology has reduced transmission delays, and can help to detect and decrease response delays.

plete the task (delay from other responses). Each of these delays should be taken into account while modifying the bureaucracy.

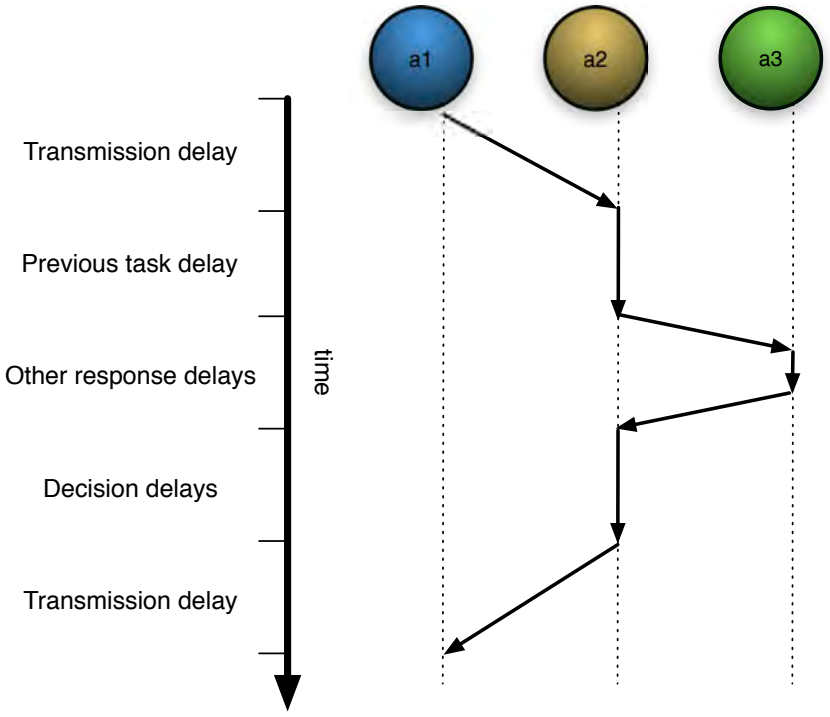


Figure 6.2: Response delay can be decomposed in previous task delay, other response delay, and decision delay.

A benefit of electronic logs is that they can be used to provide accountability of decisions, as was the case for the Enron e-mail data set[6]. The workload of individual agents or departments could also be measured by the number of requests they send or receive, considering the decision delays required by each request. Finally, a visualization of the interactions within the bureaucracy (who communicates with who) could provide insights to improve its design, for example detecting redundant agents or interactions, or creating "shortcuts" (cf. Bollen and Heylighen, 1996) between agents that communicate frequently via other agents.

[6]See http://www.cs.cmu.edu/~enron/

6.3.1 Decision Delays

Technology has also aided in the reduction of decision delays, which also lead to friction. Electronic databases provide instant information, while in a physical repository a clerk has to search an archive for documents. The role of the clerk is taken by software in an electronic database. Similarly, monotonous decisions can be taken by computer systems near instantly, reducing decision delays. An example of this can be seen with bank credit evaluation (Hand and Henley, 1997), where a computer system can give instant decisions on whether to give credit to an applicant or not. Similar methods could be used to make instant decisions to judge e.g. visa applicants or prospective students. Turning decisions into computer systems will reduce decision delays, thus reducing friction, improving communication, and increasing the bureaucracy efficiency. This is precisely one of the directions agent technology is taking (Luck et al., 2005), using notions of negotiation, trust, and reputation to facilitate the coordination of electronic decision makers. Also research and technology applied to e-government (Layne and Lee, 2001) and computer aided decision making (Turban and Aronson, 1997; Stahl, 2006) will improve the performance of bureaucracies.

Such a hybrid scenario, where humans and software agents interact in an organization, has been described with the term "cognitive stigmergy" (Ricci et al., 2006). Stigmergy is used to describe systems, such as insect colonies, that exploit their environment to communicate and coordinate (Theraulaz and Bonabeau, 1999; Werfel and Nagpal, 2006). In a similar way, computer systems can be used as an environment to facilitate the communication, coordination, and decision of agents.

6.4 The Role of Sensors

"Sometimes our intentions are not responsible for our possibilities"
—Nadia Gershenson

In the previous section, reduction of friction within the bureaucracy was discussed. In this section, reduction of friction between the bureaucracy and its environment, namely the public, will be discussed.

Much research has been made in decision making, e.g. Simon (1976). It is clear that without proper sensors there will not be enough information to make proper decisions. Still, even with simple sensors, a system can obtain much information from its surroundings. An example can be seen with blind people who perceive their environment with a walking stick,

sensing by pressure only one point in space. Integrating information in time, they are able to obtain relevant information to navigate through complex areas. Nevertheless, complex sensors can reduce the complexity of a decision making process, by "digesting" relevant information[7]. Therefore, bureaucracies should aim at developing fine sensors to perceive and digest information relevant to their goals. In any case, without proper sensors, no self-organization nor adaptation can take place.

One element that facilitates the sensing process is public participation, since people themselves digest and feed the information to the bureaucracy. However, many people are reluctant to participate in such processes, since they do not see any benefit for it, while it takes some of their time. An alternative would be to reconstruct public opinion from a limited set of the population, as polls have been doing, and novel methods could improve, e.g. Rodriguez and Steinbock (2004). Here, only improvement of sensors that do not require public participation will be discussed. Still, a simple feedback e.g. rating satisfaction between 1 and 5 could be given by the public without much effort, sensing not only detecting friction, but also evaluating whether implemented changes had any visible effect.

In order to measure the efficiency of a bureaucracy, sensors should be used. A popular variable related to this efficiency is public satisfaction: if the public is happy with the services provided by the bureaucracy, then its efficiency can be assumed. Polls have been used to measure public satisfaction, but demand a certain effort from public and resources to design and analyze them. Also, they cannot measure all possible mishaps. Moreover, it takes several days to obtain results with them.

Thus, bureaucracies should develop sensors for public satisfaction that do not require public participation. This could be done measuring the public attention delay, which would be the sum of the waiting delay (how much time a person needs to queue) and the procedure delay (how much time a person needs to interact with the bureaucracy). Another indicator would be the frequency of interaction, namely how many times the same person needs to interact with the bureaucracy. These delays can be considered as friction between the bureaucracy and the public, and should be minimized. Both the public and the bureaucracy will be satisfied if they need to interact with each other as few as possible (low interaction frequency), and each of these interactions takes as little time as possible (low public attention delay). Like this, the precise places where bottlenecks arise, and for which cases, can be detected, and measures can be taken. For example, if a procedure for a special type of license takes consistently

[7] This is the System/Context tradeoff discussed in Section 4.3.2.

more time than others, this procedure should be revised and adapted.

One could argue that bureaucracies do not need to care for the public, since they can be considered as monopolies. But the tendency towards improving bureaucratic services refutes this argument. For example, e-government practically eliminates the waiting delays. It is beneficial for political parties in office to improve bureaucratic services to increase public satisfaction, and thus get more votes in the next election. Natural selection will give better chances of survival to political parties that genuinely satisfy public demands. Certainly, this can only happen in countries with a certain degree of political diversity. Otherwise, the state would indeed be a monopoly.

6.5 The Role of Hierarchies

Hierarchies are certainly useful for organizations (Helbing et al., 2006). These imply a ranking where agents are subordinates of other agents in a pyramid-like structure, i.e. less agents on the top and more on the bottom. A problem might arise when these are too rigid and changes are necessary for adaptation. Moreover, when several aspects should be dealt by a bureaucracy, it might be that one agent should be above another in some aspect (e.g. logistics), whereas in a different aspect the opposite might be the case (e.g. legal advice).

Ashby's law of requisite variety (Ashby, 1956) tells us that a system needs to have proportional variety of actions to respond to the variety of perturbations from its environment (the word variety here could be substituted for the word complexity). A hierarchy could also be necessary for coping with environmental complexity (Aulin, 1979). Multiscale analysis (Bar-Yam, 2005) is a formal tool that can be used to determine when a hierarchy is required. Basically, if the complexity of an environment cannot be coped with by individual agents, these need to aggregate and coordinate to cope collectively. The organizational relations between agents lead naturally to hierarchies, in the sense that some agents will tell other agents what to do.

To visualize hierarchies, bureaucracies can be represented as networks (Strogatz, 2001; Newman, 2003), where each node represents an agent (at a particular scale), and edges represent interactions between agents. Certainly, a network representing a bureaucracy will not be homogeneous, since different roles are taken by different agents. A hierarchical bureaucracy can also be represented as a network (Figure 6.3a). As the complexity (variety) demanded by the bureaucracy's environment increases, the

diversity of roles and interactions also augments. A solution would be to increase the number of agents, but this would lead to longer communication delays. A better alternative would be to increase the interaction types between the existing agents, to avoid the introduction of new actors while coping with the required complexity. These new interactions might change the strictly hierarchical nature of the bureaucracy. However, even when the bureaucracy might be highly distributed, a certain hierarchy will always be found, simply because the network is not random nor homogeneous, i.e. there will always be agents with more weight in the network's function than others (Figure 6.3b). In other words, the high nodes can delegate part of the control of the system to their subordinates to reduce the necessity of information flows and their respective delays.

In a system where too many agents need to interact at once, such as the European Union with its current twenty five members, the complexity of the interactions may be too difficult to manage. *Modularity* can help in coping with the complexity (Simon, 1996). Following the EU example, it will be less complicated if some decisions are made (locally) e.g. by five groups of five countries, and then these five groups discuss a final decision, than having all members discussing at once (globally). This is because the decision delays of each agent add up, since agents (in theory) need to listen to other agents before making a decision. More agents interacting imply more potential friction. In the modular case, discussions can go in parallel, so it would take five decision delays for the first round, and five for the second round, ten in total. In the plain case, it would take twenty five decision delays to have a discussion. Certainly, too many modules would also create delays. A balance should be sought where agents can make decisions and interact as efficiently as possible. What is important is to note that modularity in a network also implies a certain hierarchy (Figure 6.3c). The size of modules will also be limited by the cognitive abilities of the agents (Miller, 1956; Dunbar, 1993), [8] Extrapolating these ideas, we can say that other types of agent should also keep the number of interactions low. From Equation 2.1, we can see that more interactions imply a higher complexity of the system, but in many cases higher complexity is required to cope with an increasingly complex environment.

A desirable property of bureaucratic networks will be that they have a "small world" topology (Watts and Strogatz, 1998). This means that most interactions between agents will not need many intermediaries. This is important for information transfer, again, to reduce communication de-

[8] If there is an external cognitive enhancer, such as a collaborative environment, maybe this size could be increased.

lays. Simply ensuring that messages do not need to pass through several agents before reaching their destination will result in a small world effect, because like this the agents that need to be connected will be connected. If the same message needs to pass from agent A to agent B, to finally reach agent C, it might be worthwhile to simplify and do a shortcut from A to C (Figure 6.3d). This same idea was proposed by Bollen and Heylighen (1996) to improve website navigation by dynamically creating direct links between pages that users reached via other pages.

In this scenario, a bureaucratic hierarchy is dynamic and changing when necessary, adapting to changes of its environment directed by friction reduction. Again, these changes can be said to be self-organizing, because the restructuring comes from within the institution, directed by its own dynamics.

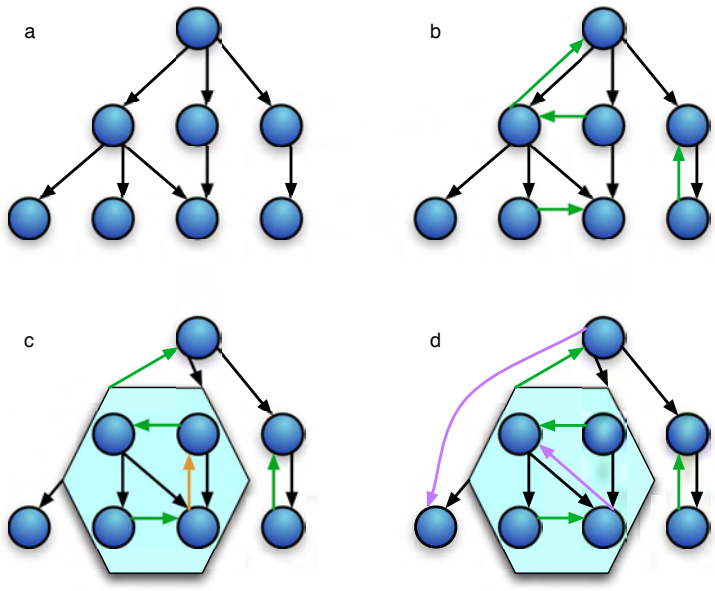


Figure 6.3: Hierarchy represented as a network. Arrows indicate dependencies. a) Strict hierarchical network. b) As interactions and dependencies increase, strict hierarchy is broken, but still far from homogeneous. c) Modules can be created when too many interactions cause delays. d) "Shortcuts" can be made to avoid intermediaries. Links can also be removed or modified, as the network adapts.

6.6 The Role of Context

In order to cope with its own complexity, any organization will try to simplify procedures. Abstracting from several instances, details can be omitted, and a uniform approach can be taken to deal with new instances, internal and external. For example, the public tends to be treated uniformly. This makes sense in cases when there are not many differences between individuals, e.g. to apply for a passport: every citizen has a name, address, etc. However, when a uniform approach is used for cases where there is diversity in the public, difficulties may arise. A single template cannot predict beforehand all cases, and usually makes simple cases complicated. An example can be seen with certain tax declarations, that include very specific sections that only few people must fill in, but are delivered to everybody, even if they just need to sign and declare that they had no income. And changes in the taxation policies can make it complicated enough to pay somebody to fill in a declaration for us. Rather than including all possible cases in a single form, a more reasonable approach would be to *contextualize* the situation, providing individual solutions for specific cases. Electronic media make this feasible, by generating instant options depending on the current circumstances.

Contextualizing interactions will reduce frictions, internal and external, because both agents involved in the interaction would benefit if delays are reduced by removing considerations that do not apply for the current situation. Certainly, too much contextualization can be counterproductive, since agents need to learn how to deal with each case. If every case requires new decisions, then expertise will not be able to improve the performance of agents.

What could be done is to categorize contexts into commonly occurring categories, by using one of many well-known techniques for automatic classification or clustering. Returning to the tax declaration example, people who filled in similar parts of the form can be automatically classified into a contextual category, such as pensioners or unemployed. Like this, a system can find automatically which contexts are common, and what measures should be taken only for those contexts. Since this would be a continuous process, new contextual categories can arise and old ones may disappear. The advantage is that these changes are led by the usage of the bureaucracy itself, satisfying its demands.

6.7 A Toy Model: Random Agent Networks

To have a better feeling of the usefulness of the ideas described so far, especially in Section 6.3, a simple computational model can be used to measure the performance of abstract bureaucracies, represented as "random agent networks" (RANs). This model, partly inspired by random Boolean networks (Kauffman, 1969, 1993; Wuensche, 1998; Aldana-González et al., 2003; Gershenson, 2004b), tries to make as few assumptions as possible about the structures of bureaucracies.

A RAN consists of N nodes. A node represents an agent, which could represent a person, a department, or a ministry. Each agent i solves a task. To do so, it sends requests to K_i other agents, which can be called "dependencies" or connections. The dependencies of every agent are chosen randomly at the beginning of a simulation, not to assume any organization. Once the agent receives a response from all its dependencies, the task is complete. However, the dependencies might receive several requests from several agents. Thus, they store requests in a queue, which they attend in a first-come, first-served basis[9]. Time is also abstracted, so agents take one time step to send requests (transmission delay), one time step to answer one request from the queue (decision delay), and one step to integrate the responses and complete a task (decision delay). Once a task is complete, agents start a new task. Agents respond to requests from their queue only when they are expecting responses from their own dependencies, i.e. they are not busy performing a task.

The performance of the network can be measured by the number of tasks it is able to complete. Thus, the time "wasted" by agents while waiting for responses from their dependencies (response delay) and having an empty queue should be minimized.

Now, there are many possible ways of randomly assigning dependencies to agents. The simplest would be **homogeneous**, where each agent has exactly K dependencies chosen randomly. Following a **normal** probability distribution, every agent will have on average K dependencies, so some agents will have more and some will have less. A more natural distribution would be **scale-free** (Aldana, 2003), where few agents have many dependencies, and most agents few[10]. Intuitively, a special topology where every agent receives the same number of tasks should obtain the

[9] For simplicity, in the model dependencies do not propagate: requests from queues are answered in one time step. In real bureaucracies, some of these requests might require further requests to further dependencies.

[10] More precisely, the number of dependencies for each agent is generated with the probability distribution $P(x) = (\gamma - 1)x^{-\gamma}$ (Gershenson, 2004b)

best performance, so that workloads are distributed equitably, not allowing request queues to grow for particular agents. This would imply that all agents would receive the same number of requests, so that idling time and queue lengths would be minimized, and the whole network would be coordinated. A (non random) topology where every agent connects to K neighbors, similar to cellular automata, fulfills these requirements. This topology can be called **symmetric**.

The RAN model was implemented in a public software laboratory written in the Java programming language. The reader can use the software laboratory and download the source code via the website http://rans.sourceforge.net

As an initial state, all agents send requests to their dependencies. Afterwards, agents are updated sequentially each time step, i.e. there are N updates per time step. The behavior of the network converges to a periodic or quasi-periodic pattern, i.e. an attractor. Interesting parameters to observe are the response delays (how long it takes an agent to complete a task, which is determined by how quickly its dependencies are able to process its requests) and queue lengths (how many requests an agent has yet to process). An example of the dynamics of these parameters can be seen in Figure 6.4.

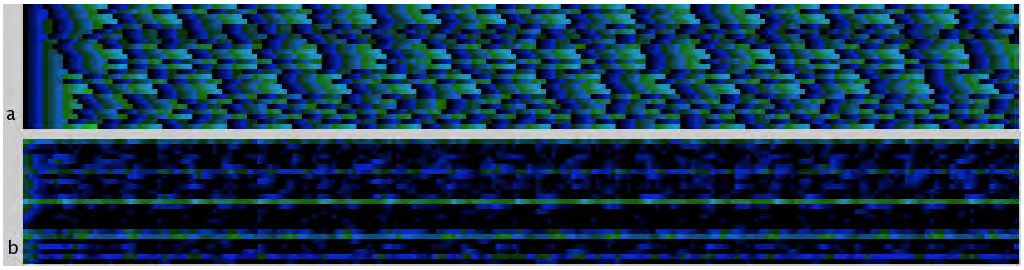


Figure 6.4: Dynamics of a random agent network of $N = 25$, $K = 5$ with homogeneous topology for 200 time steps. a) Response delays. b) Queue lengths. Lighter colors indicate higher values. The initial state is the leftmost column, and the subsequent columns show the temporal evolution of the RAN.

Except for the symmetric topology, for any values of N and K, the task queues seem to converge typically to a power law distribution: there are few long ones (bottlenecks), and many short ones. Still, this remains to be studied thoroughly.

6.7.1 Using self-organization to improve performance

If we see the satisfaction of agents in terms of the tasks they are able to complete, the satisfaction will be lower when the response delay is higher. Agents with longer queues cause more friction than others, because they cause a high response delay on the agents that are dependent on them, i.e. they have a high previous task delay. Thus, a natural way of reducing friction would be by restructuring the network in such a way as to reduce the longest queues.

A very simple criterion achieves this. To restructure a RAN, the agent with the maximum queue average (A) is detected. Then, the agent with a maximum response delay (B) that has as a dependent the agent with the longest queue changes its dependency to the agent with the shortest queue (C). In a real bureaucracy, electronic logs could be used to obtain these measures. This mechanism is illustrated in Figure 6.5. In many cases, the agent with highest response delay (A) has the agent with longest queue (B) as one of its dependencies. This is natural, since B will be able to complete its task only when A reaches the request after processing its queue. It is obvious that changing the dependency of B from A to C will reduce the response delay. What is not obvious is the precise effect that this will have on the global performance, i.e. the impact of the change.

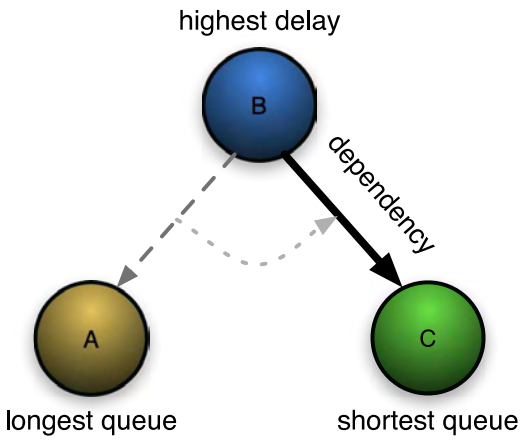


Figure 6.5: RAN Self-organization mechanism: The agent with highest delay (B) restructures its dependency from the agent with longest queue (A) to the one with shortest (C).

6.7.2 Simulation Results

To compare the scale-free topology with the others, it was normalized to have a total number of dependencies in the network very close to $N * K$.[11] The networks with normal topology also were checked to have a comparable number of dependencies, since networks with less dependencies are able to process more tasks.

Simulation runs were performed for different parameter values. For a network size of $N = 15$, for each topology, 1000 RANs were created, and their response rate (average tasks completed per time step) was plotted as the self-organization was iteratively applied once every 1000 time steps. For $K = 1$ (Figure 6.6), even without self-organization the scale free topology performs even better than the symmetric one. This is because there is usually only one or few nodes with lots of dependencies, and many nodes with few or no dependencies. The latter ones are able to complete their tasks quickly, since they have little or no interference from the delays of other nodes, and this enables them to respond quickly to the demands of the former ones. Note that there is already a certain hierarchy and modularity inherent in this configuration. As self-organization restructures the RANs, the non-symmetric topologies increase their performance, and after few reorganizations, the homogeneous and normal topologies also perform better than the symmetric. These two have initially bad performance because randomly some nodes are dependencies of more than one node, while others are dependency of none. This creates queues that affect all the nodes that share busy dependencies. By changing the dependency to idle agents, the idle agents are still able to complete their own tasks, while serving as dependencies of other nodes, and reducing the overall friction in the network. Still, self-organization is not able to improve the performance of the symmetric networks, which seem to be trapped in a local optimum.

For $K = 2$ (Figure 6.7), before self-organization, the symmetric topology performs the best. After few self-organizations, the homogeneous topology achieves the same performance, while the normal and scale free surpass it. This is because of the same reason explained above: if some nodes have several dependencies, while these dependencies have few dependencies themselves, overall they are able to process more tasks than if every node has the same number of dependencies: the nodes with few dependencies are able to process their own tasks quickly, and to respond

[11] More precisely, $\gamma = 2.48$ was used, since networks with this value have similar properties to $K = 2$ (Aldana, 2003). Then, the probability was multiplied by $K/2$, to normalize.

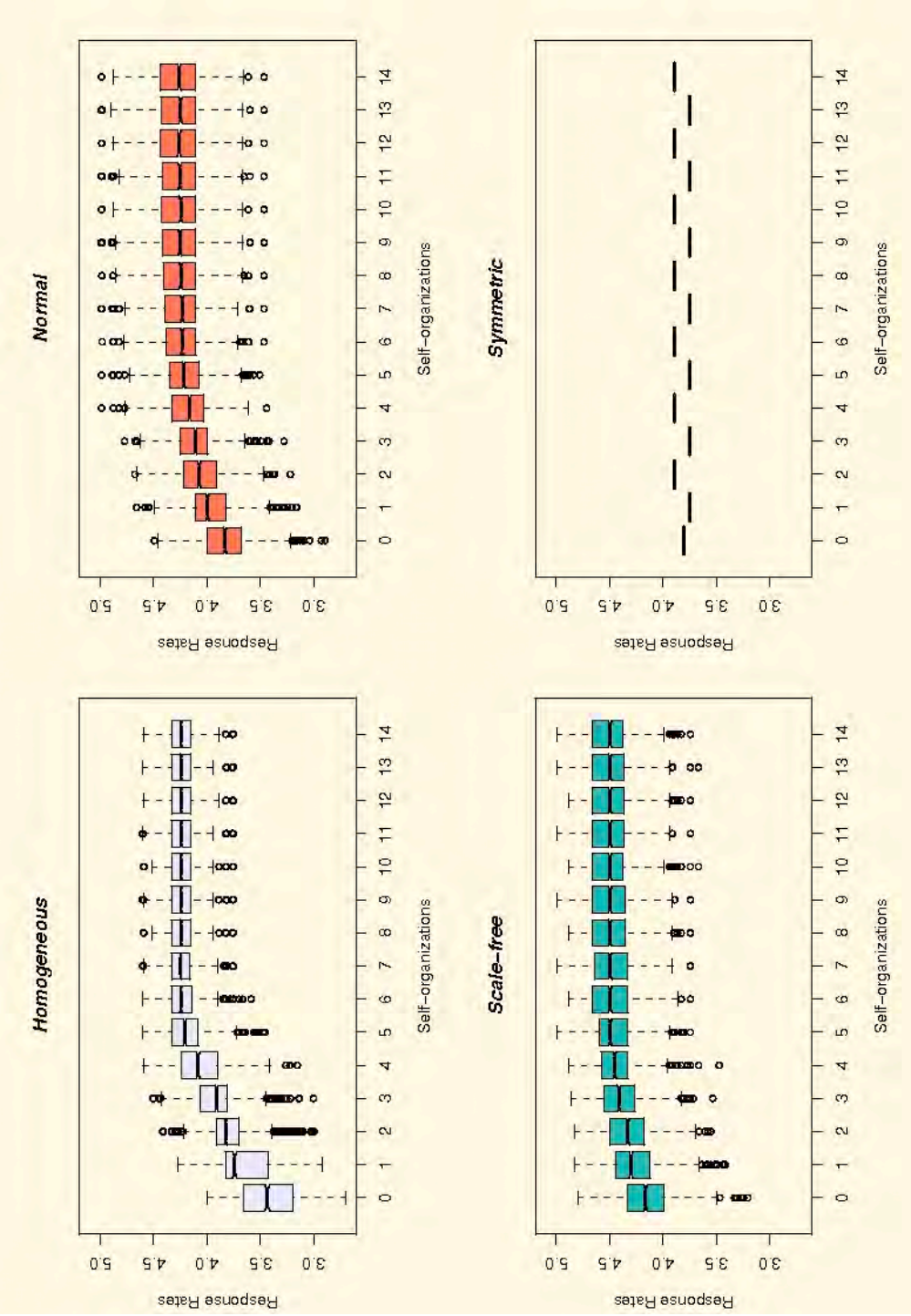


Figure 6.6: Results for $N = 15$, $K = 1$.

to the agents with many dependencies promptly, creating certain hierarchy and modularity at the same time. A similar case is seen when $K = 5$ (Figure 6.8).

As K increases, it becomes more difficult to benefit from having several dependencies, since these will also have several dependencies. A high degree of connectivity, unrealistic but considered for completeness, also implies less hierarchy and less modularity. Thus, initially all topologies perform worse than symmetric, but through self-organization, they tend to reach a similar performance. This can be seen for the extreme case when $K = 15$ (Figure 6.9).

Scale-free topologies already imply a hierarchy, since few agents have several dependencies. Actually, these "central" nodes dictate the rhythm of the network, simply because they can quickly propagate changes to most of the network via their dependencies, whereas most agents with few dependencies cannot.

The same behavior as described above can be seen for larger networks ($N = 100$), for $K = 1$ (Figure 6.10), $K = 2$ (Figure 6.11), $K = 5$ (Figure 6.12), and $K = N$ (Figure 6.13) (for this last case, only 25 networks were generated).

6.7.3 RAN Discussion

Many open questions remain in this abstract model. However, the main goal was to illustrate the benefits of self-organization. The simulation results showed that only a few modifications of the network topology are required to increase performance to near optimal levels. For higher values of K, more modifications are required, simply because there are more dependencies in the network. Still, an interesting outcome of this model is that only a small fraction of the total number of dependencies needs to be reconfigured to enable a *random* network to achieve a very good performance. Note, however, that the cost of the changes is not taken into account by the present model.

Self-organization does not ensure optimality, but adaptability. For example, if the changes in demand of a bureaucracy change the decision time for a task, this can reconfigure itself to accommodate the change robustly. This can be useful for automatic detection of malfunctions and initial response to them: if an agent "breaks down", its queue would grow, but the agents that have it as a dependency could rearrange their connectivity towards agents working properly. Notice that the presented model assumes that all agents have equal decision and transmission delays. How-

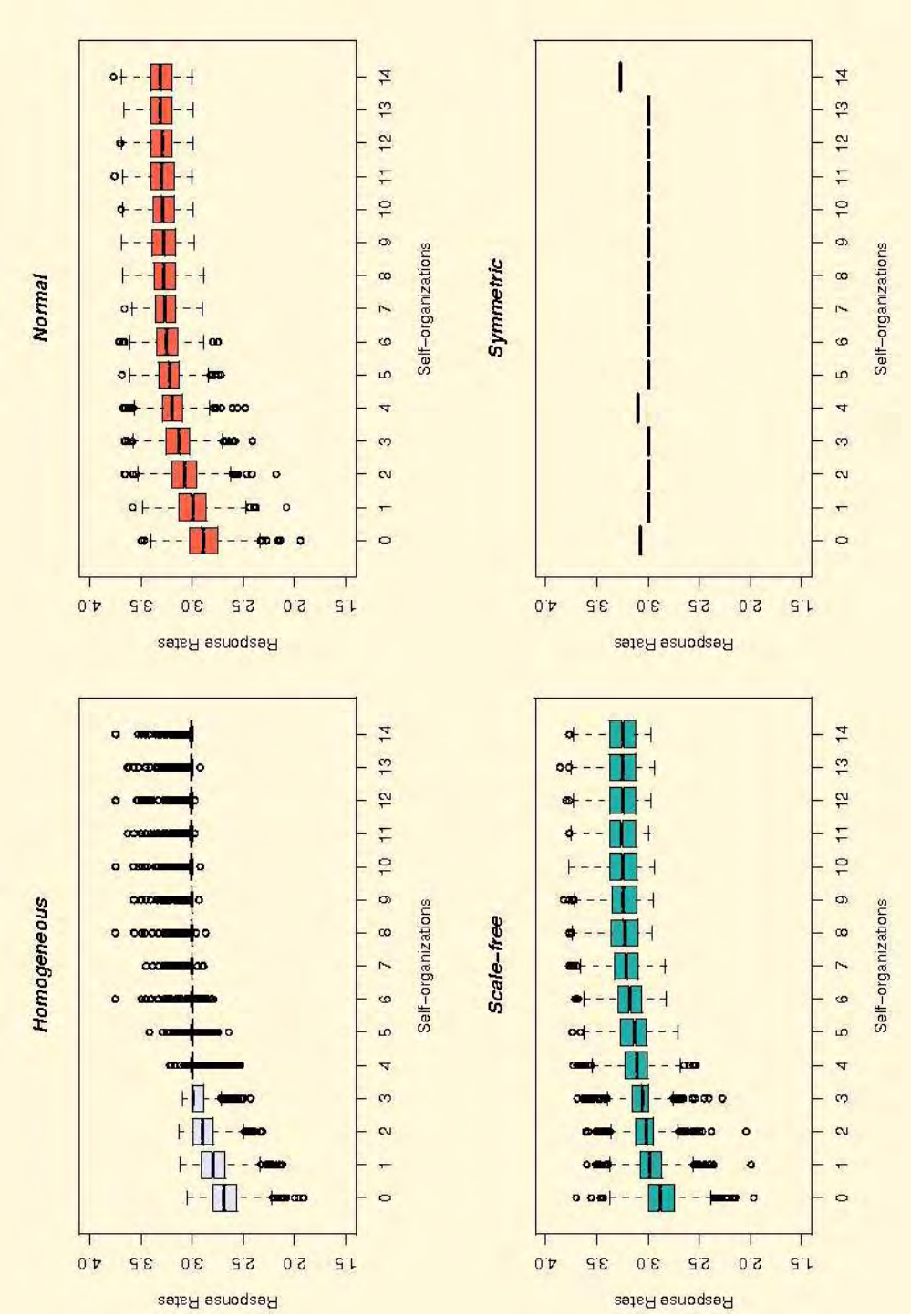


Figure 6.7: Results for $N = 15$, $K = 2$.

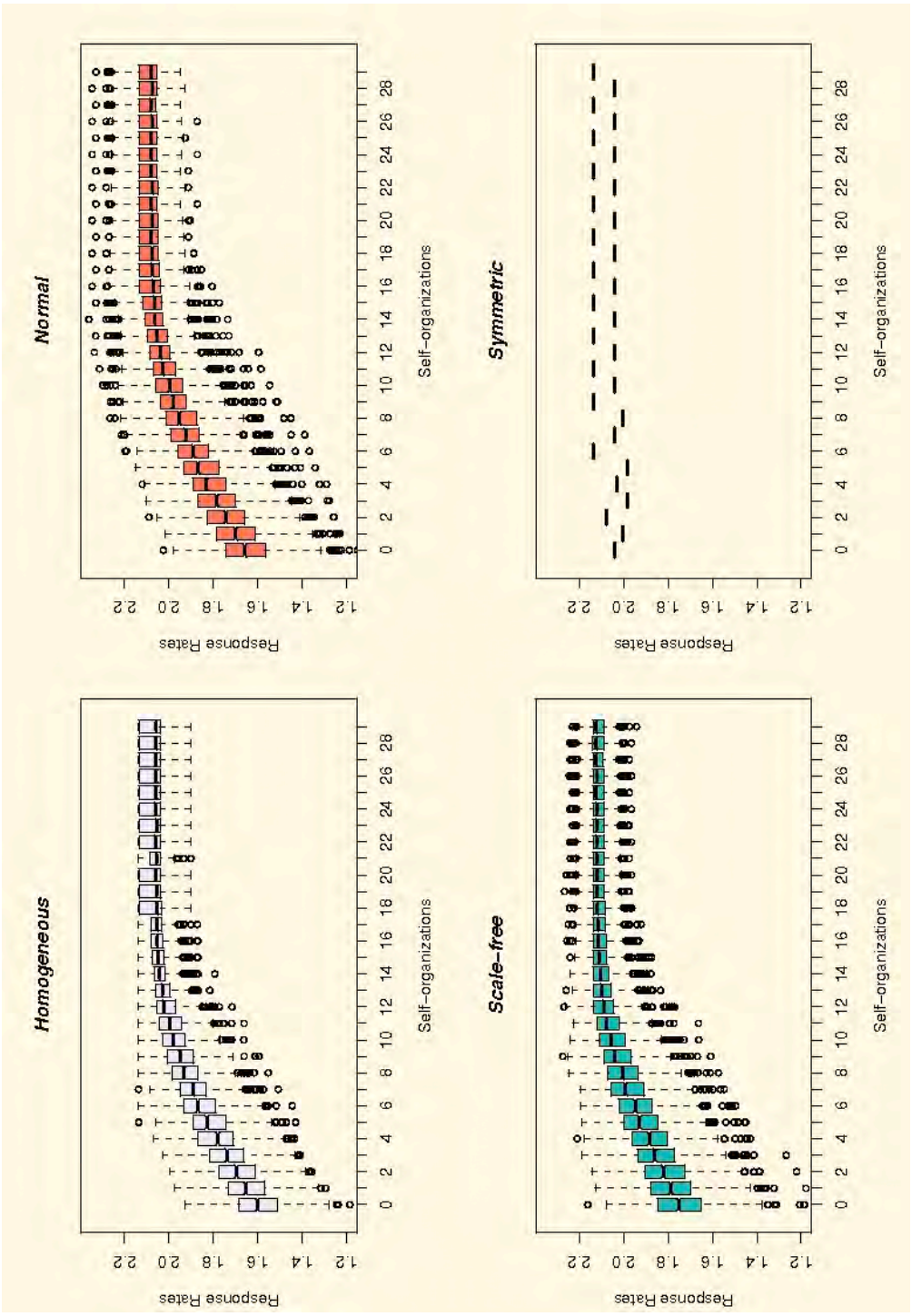


Figure 6.8: Results for $N = 15$, $K = 5$.

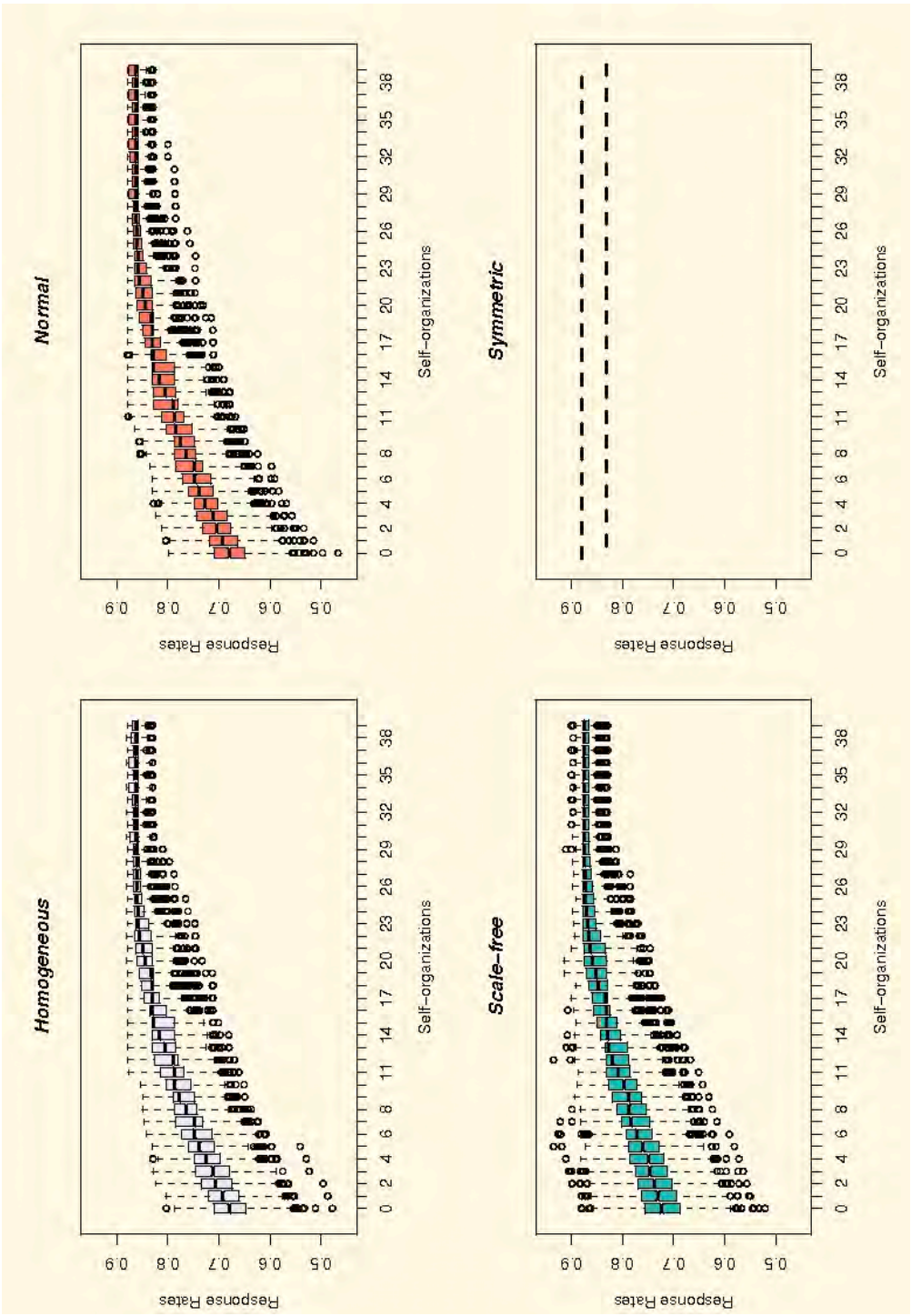


Figure 6.9: Results for $N = 15$, $K = 15$.

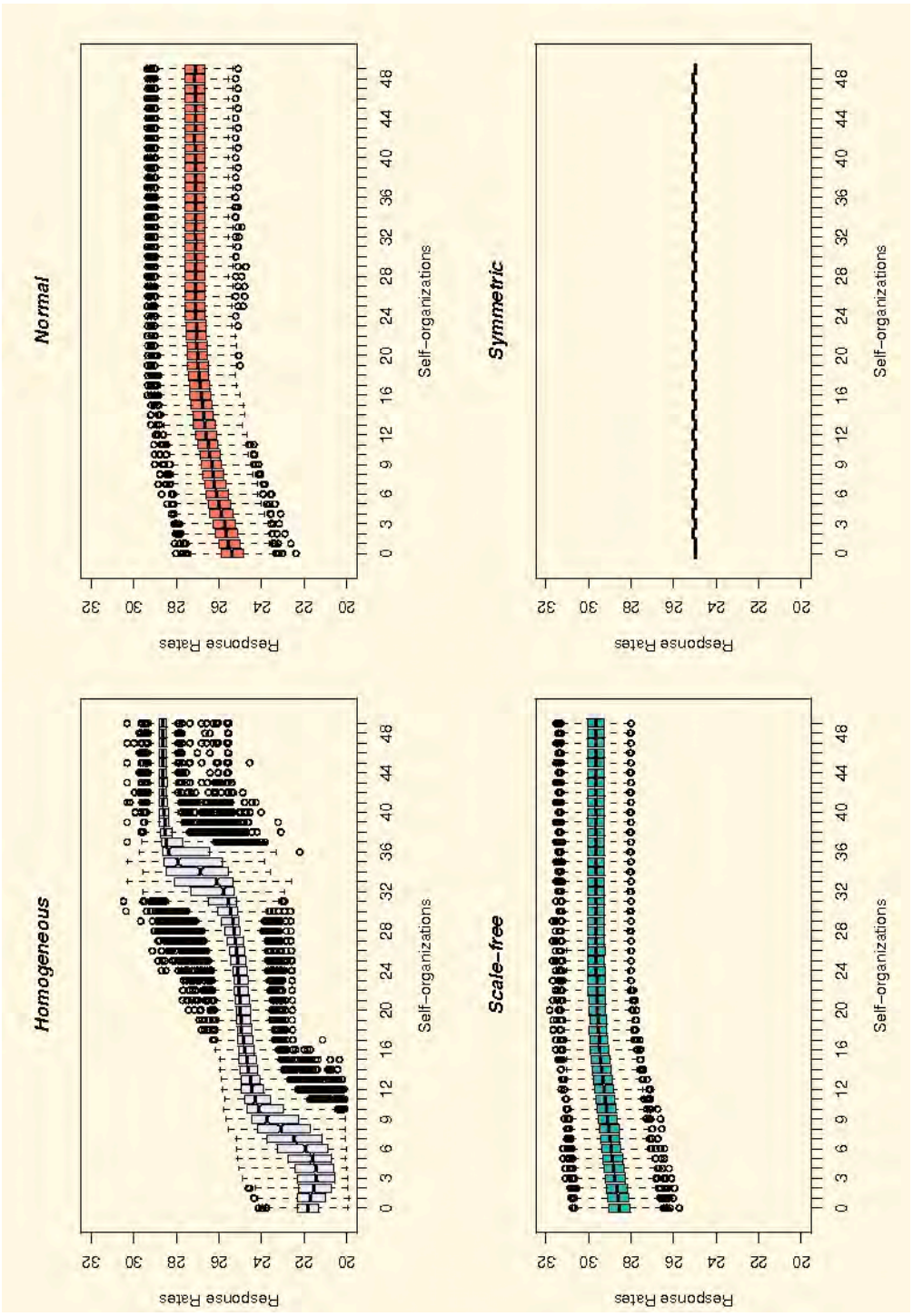


Figure 6.10: Results for $N = 100$, $K = 1$.

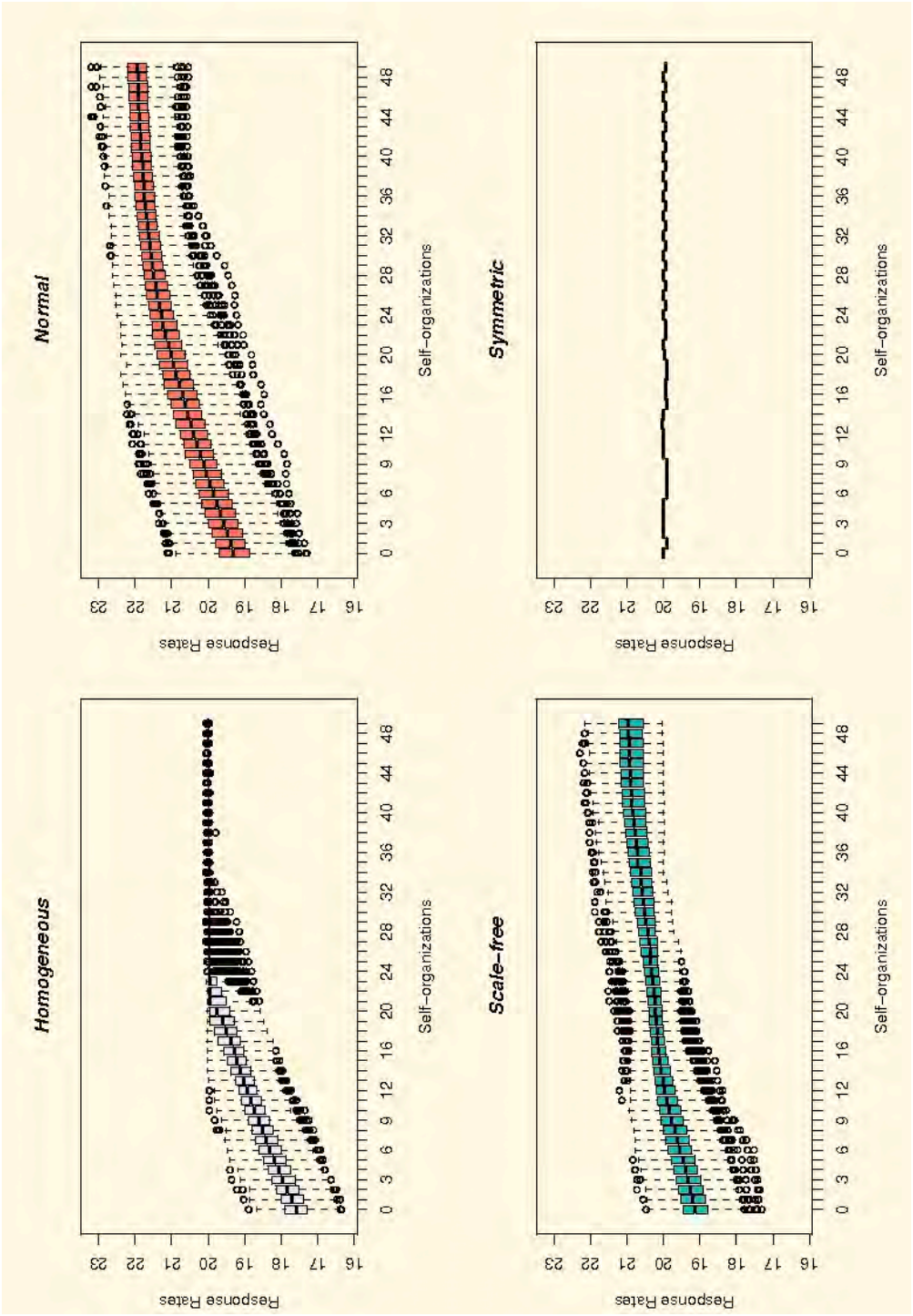


Figure 6.11: Results for $N = 100$, $K = 2$.

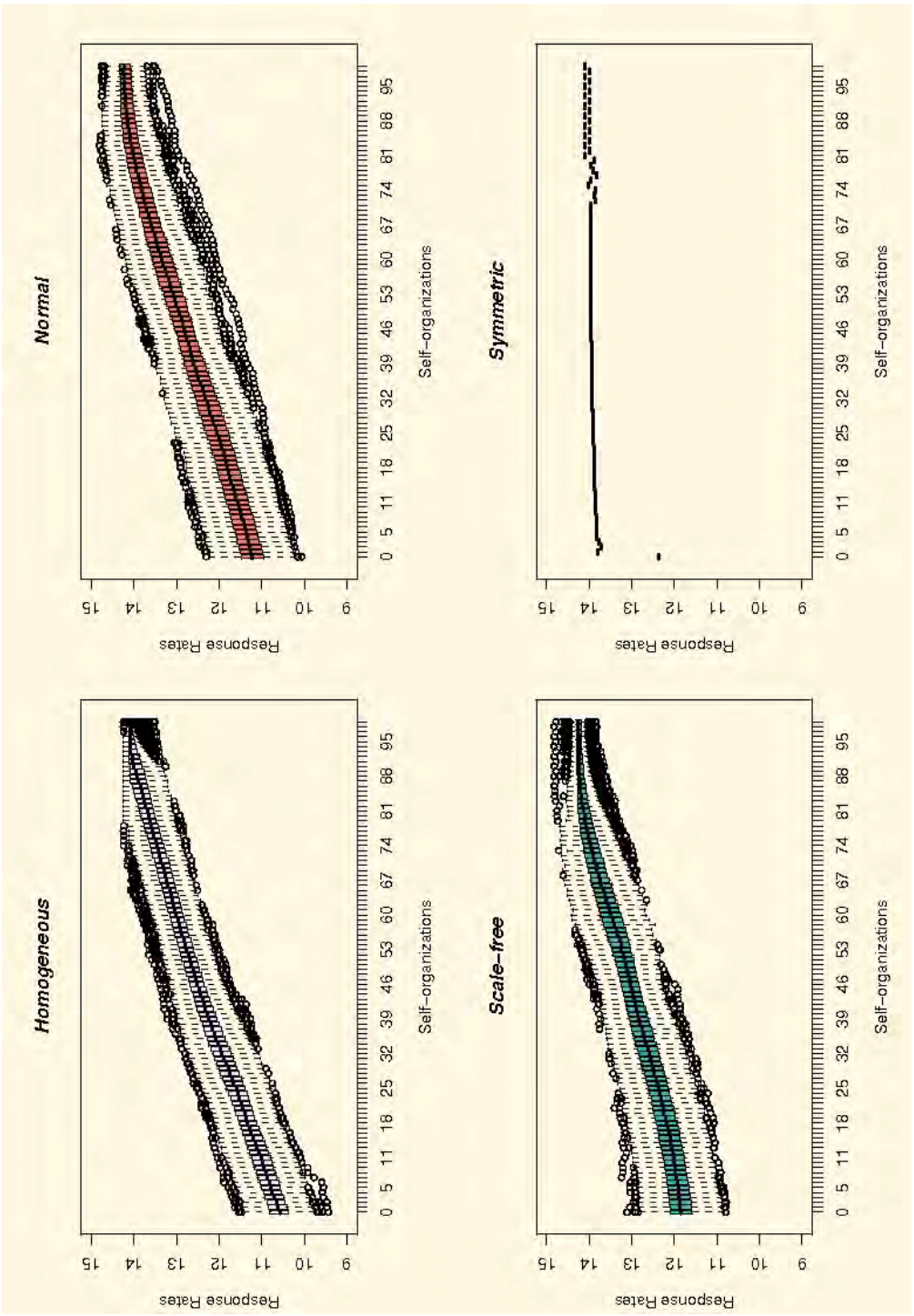


Figure 6.12: Results for $N = 100$, $K = 5$.

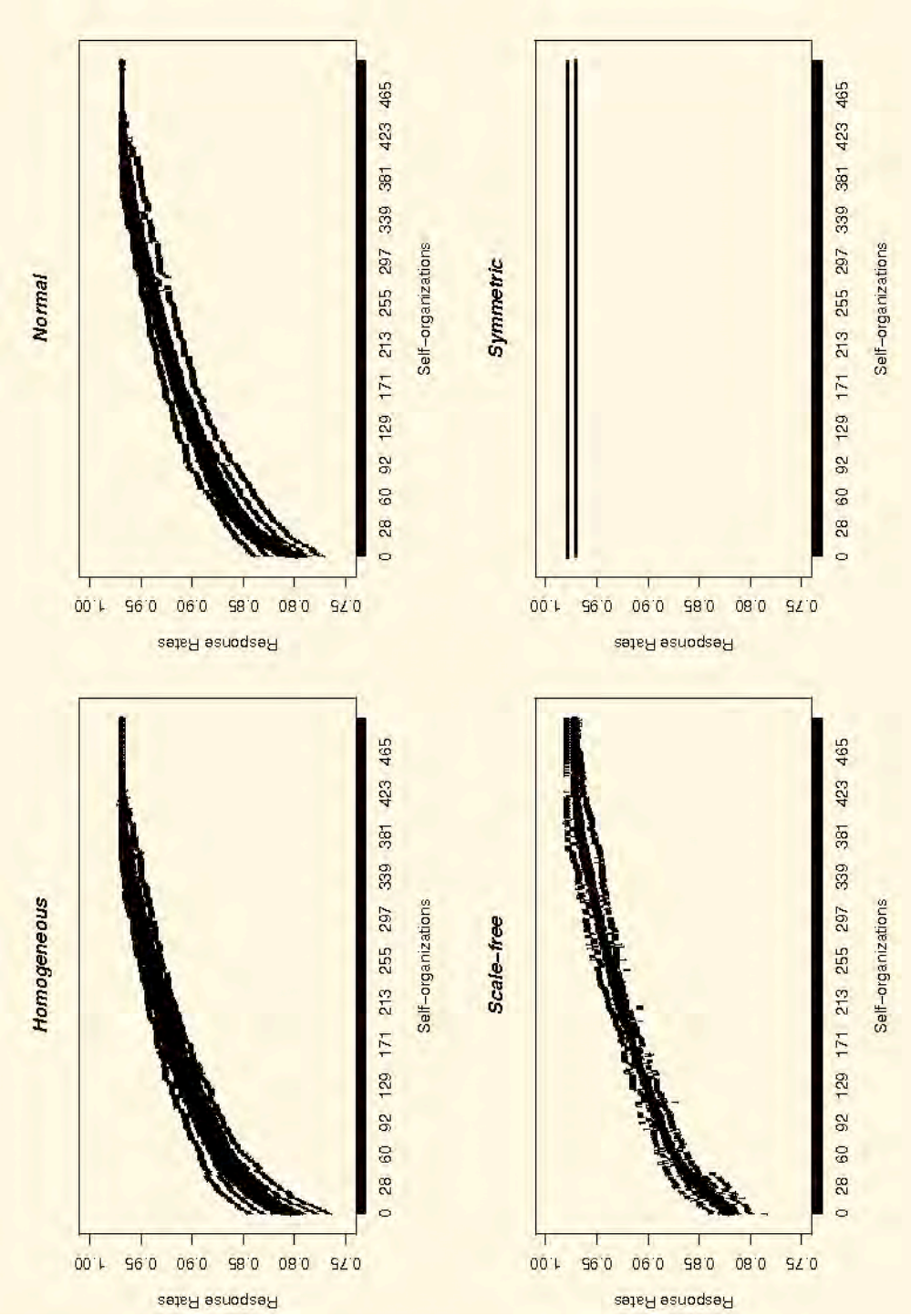


Figure 6.13: Results for $N = 100$, $K = 100$.

ever, weights could be used to model diversity in the delays of agents. Also, if in a real bureaucracy some dependencies cannot be changed, self-organization will find its way with the available flexible dependencies.

One remaining question is: when to stop self-organizing? In principle, self-organization can continue without degrading the performance of the network, while ensuring its adaptability. However, for the simple case presented here, there is no need of adaptation, so sooner or later the self-organization will take the RAN to a previously visited configuration. Some changes actually decrease slightly the performance, but the long run tendency is towards the highest possible performance for a particular network. If it is known what is the desired maximum performance, then that can be a criterion to stop the self-organization, but in some cases it might not be obvious to know beforehand the maximum performance.

RAN-like models might also be useful to study organizational robustness, i.e. how well an organization can respond to node failure, using sensitivity analysis (Gershenson, 2004b). For example, redundancy of nodes can be useful to ensure functionality of key or problematic tasks (Gershenson, Kauffman, and Shmulevich, 2006).

6.8 Conclusions

"Every social regime creates problems"
—Kenneth Arrow

This chapter presented suggestions to use self-organizing techniques to improve the efficiency of different aspects of bureaucracies. All the improvements mentioned decrease different delays within a bureaucracy, reducing frictions and leading to efficient adaptability. This is because increasing speeds of reaction and decision will allow a bureaucracy to adapt quickly to unexpected changes, while preserving its functionality. In consequence, the "satisfactions" of agents, whether internal or external, will be increased.

It should be noted that even when self-organization can suggest changes to improve efficiency, the human factor needs to be taken also into account, since it is natural to have a resistance to changes. Because of this, any actual implementation of the ideas presented here needs to be developed together with the members of the bureaucracy, to explore which changes are viable and which ones are not.

Standards and digital signatures certainly could be used to comply with the formalities of bureaucracies. Using electronic media is not an im-

pediment for this. The adoption of these media is already underway, and it might bring in more benefits than just decreasing transmission delays. They can effectively support different types of self-organization within bureaucracies. The real value of self-organization will only be appreciated once it is applied in these organizations. But the ideas presented here are encouraging enough to try. Similar approaches could also be useful for other types of organizations: if frictions are reduced, satisfactions will increase.

The model presented, random agent networks, is very abstract indeed. It would be interesting to study more realistic versions of the model, e.g. considering costs of changing the topology, non-homogeneous delays, and domain expertise of agents. The *stability* of RANs should also be studied. Perhaps, as with random Boolean networks, there would be an "edge of chaos", where a balance is achieved between stability and change. If so, different parameters could be used to shift this "edge of chaos", to modulate the performance of the RAN. Furthermore, different updating schemes (Gershenson, 2002a, 2004c) could also enhance our understanding of the model.

Chapter 7

Self-organizing Artifacts

In this chapter[1] *we discuss which properties common-use artifacts should have to collaborate without human intervention. We conceive how devices, such as mobile phones, PDAs, and home appliances, could be seamlessly integrated to provide an "ambient intelligence" that responds to the user's desires without requiring explicit programming or commands. While the hardware and software technology to build such systems already exists, as yet there is no standard protocol that can learn new meanings. We propose the first steps in the development of such a protocol, which would need to be adaptive, extensible, and open to the community, while promoting self-organization. We argue that devices, interacting through "game-like" moves, can learn to agree about how to communicate, with whom to cooperate, and how to delegate and coordinate specialized tasks. Thus, they may evolve a distributed cognition or collective intelligence capable of tackling complex tasks.*

[1] Based on Gershenson and Heylighen (2004).

7.1 A Scenario

"Our brains make the world smart so that we can be dumb in peace!"
—Andy Clark

Imagine the near future: you download a recipe, or get it by SMS on your mobile, or your friend beamed it into your PDA. You ask your device (computer, mobile, or PDA), to check your stock with your "intelligent kitchen assistant" (IKA), which, with the aid of RFID tags, keeps in real time track of your products, their expiry date, etc. The IKA would then be able to send to your PDA or ask your printer to produce a shopping list of the products that are missing in your stock and needed for the recipe. Or even better, the IKA could send an order to your supermarket, so that your shopping would be delivered at your home, or be ready for pickup. The technology to achieve this vision is already at hand. However, there are many steps that should be taken before it can be materialized.

The diversity and capabilities of devices we use at home, school, or work, are increasing constantly. The functions of different devices often overlap (e.g. a portable computer and a mobile phone have agendas; a radio-clock and a PDA have alarms), but most often we cannot combine their capabilities automatically (e.g. the PDA cannot tell the radio to set its alarm for the early Tuesday's appointment), and users need to repeat the same tasks for different devices (e.g. setting up an address book in different devices). Moreover, using the functionality of some devices in combination with others would be convenient (e.g. if my computer has an Intelligent User Interface, I would like to use it to ask for coffee, without the need of having speech recognition in the coffee machine: The computer should be able to ask the coffee machine for cappuccino).

Could we build devices so that they would *automatically coordinate*, combining their functions, and possibly producing new, "emergent" ones? The technology to achieve this already exists. What we lack is a proper design methodology, able to tackle the problems posed by autonomously communicating artifacts in a constantly changing technosphere. In this chapter we try to delineate the requirements that such a design paradigm should fulfill. The scenario we imagine considers a nearby future where technological artifacts *self-organize*, in the sense that they are able to communicate and perform desirable tasks with minimal human intervention.

This vision is closely related to the concept of "Ambient Intelligence" (AmI)(ISTAG, 2001), which envisages a future where people are surrounded by "smart" and "sensitive" devices. AmI would be the result of the integration of three technologies: Ubiquitous Computing

(Weiser, 1997), Ubiquitous Communication, and Intelligent User Friendly Interfaces. The first one conceives of a seamless integration of computation processes taking place in the variety of artifacts that surround us, being part of "The Grid", the network that would allow anyone anywhere to access the required computing power. This chapter focuses on the aspect of Ubiquitous Communication that attempts to obtain seamless information exchange between devices. Intelligent User Friendly Interfaces should enable an intuitive, effortless interaction between users and devices.

With current approaches, this scenario would be possible, since we have the technology, but extremely expensive, since people would need to buy from the same producer all of their devices. We can see a similar case in the area of Home Automation: the technology is available on the market, but it is not possible to buy today ventilation for a house, and in five years integrate the system with a new fire detector. An engineer needs to integrate them manually, so that the ventilation system could take the appropriate measures if smoke is detected, simply because the ventilation system was not designed to receive such signals. These limitations increase the price and restrict the market of devices for Home Automation, since complete solutions should be bought in order to have full coordination and functionality between devices. People would be more willing to invest in Home Automation if they could have the possibility of acquiring it progressively.

7.2 Requirements for self-organizing artifacts

We see self-organization as a paradigm for designing, controlling, and understanding systems (Gershenson and Heylighen, 2003). As seen in Chapter 3, a key characteristic of a self-organizing system is that structure and function of the system "emerge" from interactions between the elements. The purpose should not be explicitly designed, programmed, or controlled. The components should *interact* autonomously with each other and with the environment, mutually adapting by reducing "friction" to reach an intrinsically "preferable" or "fit" configuration (attractor), thus defining an emergent purpose for the system (Heylighen and Gershenson, 2003). By "self-organizing artifacts" we mean a setup where different devices, with different fabrications and functionalities, and moving in and out of different configurations, can communicate and integrate information to produce novel functionalities that the devices by themselves could not achieve. The performance of a system composed by such devices can

be measured with user satisfaction, as in the previous chapter.

A first requirement is cross-platform compatibility. This is already achieved for programming with Java, and for documents with XML. Another requirement is wireless communication, which is offered by technologies such as IR, Bluetooth and WiFi. Near Field Communications (NFC) is a newly envisioned standard, proposed by a consortium headed by Sony, Nokia, and Philips, which would allow information to be transmitted between devices that come in close spatial proximity ("touching").

Even with such a standard, the problem remains that the user generally would need to specifically request such communication between devices (e.g. "transfer this file from here to there"). Ideally, the devices would *know* what we want them to do and how to do it. User Interfaces already help us to tell them our wishes. Still, one device cannot tell *another* device what we want, especially if they are produced by different manufacturers. This is a general problem of communication between artifacts: they can recognize standard messages, but they do not "know" what the messages *mean*. To avoid endless debates, we can say that the meaning of a message is determined by its *use* (Wittgenstein, 1999): if a device has received a message, and does "the right thing" (for the user), then it has "understood" the meaning of the message. Thus, the user's satisfaction is the ultimate measure of the effectiveness of the artifacts' performance, i.e. σ_{sys} in Equation 4.1.

Another issue is how to deal with changes in technology. We do not want to reconfigure every artifact each time a new device arrives. Moreover, we want the old devices to be able at least to cope with the functionality of new ones. New devices should configure themselves as automatically as possible. Older ones may require user intervention at first (as they cannot know beforehand which functions will be required), but they should be able to cope with new technology being added to the network. The overall system must be *adaptive*, *extensible*, and *open*.

An adaptive system can cope with unexpected changes in its environment, as exemplified by the constantly changing technology. Having flexibility built into our systems is desirable: they should at least be able to tolerate events they were not designed for without breaking down, but preferably try to find adapted solutions, or at least ask assistance from the user. For example, home appliances have a limited set of functions. To have them self-organize (e.g. the alarm clock coordinating with the microwave oven, and the oven with the kettle), their functions could be easily programmed to respond to unknown messages. If a new device arrives, and an old one does not know what to do when it receives a message, it can check what the user wants, thus learning how to respond appropri-

ately. The possibility to add more devices to an existing configuration may be called *extensibility*.

Suppose that a company develops adaptable and extensible devices that interact seamlessly with each other. This would still leave the problem that customers cannot add devices from other companies, as these would follow their own standards, thus creating compatibility problems. We believe that the solution is to have *open* technologies, in the spirit of GNU. Open means that everyone has free access to their specifications, not necessarily their source code. The advantage is that they can develop much faster, meeting the requirements of more people, because they are developed by a global community that can try out many more approaches than any single company. Still, a company can benefit in promoting an open technology, since this would provide them with free publicity while everyone is using their technology (e.g. Sun's Java).

Open technology can respond to the needs of the largest variety of people, while allowing problems and errors to be detected and corrected more easily. Another advantage is that it allows people to get updates developed by other users for free. For example, if I program my "old" toaster to integrate with my new mobile phone, it costs me nothing to make the program available on the Internet to anyone else who might need it. Thus, updates, extensions, and specialized applications can flow much more quickly from a global community than from a private company. The growing success of software such as Firefox and OpenOffice prove these benefits. Still, companies would profit from this approach, since people would be more willing to buy new devices as integrating them into their existing, open setup, will be easier.

7.3 Achieving self-organization

As it was described in Section 4.3.2, we can divide the problem of self-organizing integration into three subproblems, that will help reduce friction and promote synergy:

1. Devices should learn to *communicate* with each other, even when they have no a priori shared understanding of what a particular message or function means.

2. Devices should learn which other devices they can trust to *cooperate*, avoiding unreliable ones.

3. Devices should develop an efficient *division of labor* and workflow, so

that each performs that part of the overall task that it is most competent at, at the right moment, while delegating the remaining functions to the others.

These issues are all part of what is known as *collective intelligence* (Heylighen, 1999) or *distributed cognition* (Hutchins, 1995): a complex problem cannot be tackled by a single device or agent, but must be solved by them working together, in an efficiently coordinated, yet spatially distributed, system, where information flows from the one agent to the other according to well-adapted rules. Until now, distributed cognition has been studied mostly in existing systems, such as human organizations (Hutchins, 1995) or animal "swarms" (Bonabeau et al., 1999), that have evolved over many generations to develop workable rules. Having the rules self-organize from scratch is a much bigger challenge, which has been addressed in distributed AI and multi-agent simulations of social systems (de Boer, 1999; de Jong, 2000; Belpaeme, 2001; Steels, 2003). Based on these explorations, we propose a number of general mechanisms that could probably tackle the three subproblems.

7.4 Learning to communicate

To communicate effectively, different agents must use the same concepts or categories. To achieve effective coordination, agents must reach a shared understanding of a concept, so that they agree about which situations and actions belong to that category, and which do not. A group of agents negotiating such a consensus will self-organize (De Vylder and Tuyls, 2006), so that a globally shared categorization emerges out of local interactions between agents.

Such self-organization has been shown in different simulations of the evolution of language (Hutchins and Hazelhurst, 1995; Steels, 1998, 2003; de Boer, 1999; de Jong, 2000; Wiesman et al., 2002). Here, interacting software agents or robots try to develop a shared lexicon, so that they interpret the same expressions, symbols, or "words" in the same way. In these simulations agents interact according to a protocol called a "language *game*". There are many varieties of such games, but the general principle is that two agents "meet" in virtual space, which means that through their sensors they experience the same situation at the same time. Then they try to achieve a consensus on how to designate one of the components of their shared experience by each in turn performing elementary *moves*.

In a typical move, the first agent produces an "utterance" referring to

a phenomenon that belongs to one of its inbuilt or previously learned categories, and the second one finds the best fitting category for that phenomenon in its knowledge base. The second agent then indicates a phenomenon belonging to that same category. If this phenomenon also belongs to the same category for the first agent, both categorizations are reinforced, otherwise they are reduced in strength. In the next move of the "game", another phenomenon is indicated, which may or may not belong to the category. The corresponding categorization is strengthened or weakened depending on the degree of agreement. After a number of moves the game is stopped, each agent maintaining the mutually adjusted categories. Each agent in turn is coupled to another agent in the system, to play a new game using different phenomena. It has been recently demonstrated (De Vylder and Tuyls, 2006) that after some games a stable and coherent system of categories will be shared by all agents. A good example of such a set-up can be found in Belpaeme's (2001) simulation of the origin of shared color categories.

If for some reason devices are not able to communicate, they should be able to notify the user, and ask for the correct interpretation of the message. This is relatively easy, since devices have a limited functionality. It would be possible to "teach" a device what to do if it receives a particular message, and the device should "learn" the meaning of the message.

Research has been done in multi-agent systems where agents negotiate their protocols (Reed et al., 2001; Dastani et al., 2001), which could be extended for a setup of self-organizing artifacts. However, agent communication standards, such as FIPA, still do not contemplate adaptation to new meanings. Nevertheless, there is promising research going on in this direction.

7.5 Learning to cooperate

"Winning or losing does not matter as much as what you learn from it"

Integrated devices should not only communicate, but cooperate. Cooperation may seem self-evident in preprogrammed systems, where the components are explicitly designed to respond appropriately to requests made by other components. However, this is no longer the case in open, extensible configurations.

For example, a person at the airport would like her PDA to collaborate with the devices present at the airport, so that it can automatically warn her when and where she has to go, or tell her which facilities are available

in the airport lounge. Yet not all devices at the airport may be ready to help a PDA, e.g. because of security restrictions, because they are proprietary and reserved for paying customers, or because they simply do not care about personal wishes. Moreover, devices may be ready to share certain types of services but not others, e.g. telling users when the flight is scheduled to depart, but not how many passengers will be on it. As another example, devices may not only be uncooperative, but malevolent, in the sense that they try to manipulate other devices in a way detrimental to their user. Such devices may be programmed, e.g. by fraudsters, spies, or terrorists.

There exists an extensive literature on the evolution of cooperation between initially "selfish" agents, inspired by the seminal work of Axelrod (1984) that compared different strategies for playing a repeated "Prisoners' Dilemma" game. However, this game does not seem directly applicable to information exchanging devices. Moreover, the chief result, while sensible, may seem trivial: the most effective strategy to achieve robust cooperation appears to be *tit for tat*, i.e. cooperate with agents that reciprocate the cooperation, stop cooperating with those that do not. More recent, tag-based models, e.g. Riolo et al. (2001); Hales and Edmonds (2003) start from a simpler situation than the Prisoners' Dilemma, in which one agent "donates" a service to another one, at a small cost to the donor but a larger benefit to the recipient. The main idea is that agents are identified by "tags", and that they cooperate with those agents whose tags are similar to their own. The rationale is that agents with the same type of tag belong to the same social group, "family" or "culture", following the same rules, so that they can be trusted to reciprocate.

For artifacts, a tag may include such markers as brand, model, and protocols understood. This would show that a device is capable and willing to lend particular services to another one, thus obviating the need for a repeated, "tit-for-tat-like" interaction probing the willingness to reciprocate. Yet extensible environments should allow the addition of very dissimilar devices, made by different companies using different standards and functionalities. Therefore, we propose a different approach, combining some advantages of tags and tit-for-tat strategies.

Consider a game with the following moves: an agent makes a request and the other agent either cooperates (donates) or "defects". Agents learn from these interactions in the following manner: if the result is positive (cooperation), the agent will get more "trust" in the other agent's cooperativeness. Thus, the probability increases that it will make further requests to that agent in the future, or react positively to the other's requests. Vice-versa, a negative result (i.e. friction) will lead to more "distrust" and a

reduced probability to make or accept requests to/from this agent, reducing friction on the long run via Tolerance. Still, to recognize this agent, it has to take its clue from the tag, which is usually not unique to that agent. This means that a later interaction may be initiated with a different agent that carries a similar tag, but that is not necessarily willing to cooperate to the same extent. We may assume that if the first few interactions with agents having similar tags all generate positive (or negative) results, the agent will develop a default propensity to react positively (or negatively) always to agents characterized by that type of markers. A similar mechanism is effectively used in eBay, where buyers rate sellers based on their purchase experience. Like this, good sellers get good ratings, and vice versa. Buyers will tend to interact more with sellers which received good ratings, even if they never interacted with them before. This not only promotes effective interactions, but also motivates sellers to give a good service, otherwise it will be more difficult for them to sell items in the future.

We expect that in this way the initially undirected interactions will produce a differentiation in clusters of similarly marked agents that cooperate with each other (e.g. all devices belonging to the same user or organization), but that are reluctant to interact with members of other groups (e.g. devices belonging to rival organizations). The tags and their association thus develop the function of a *mediator* (Heylighen, 2003a) that increases the probability of positive interactions (i.e. synergy) and reducing the probability of negative ones (i.e. friction) by creating a division between "friends" (in-group) and "strangers" or "foes" (out-group). Note, however, that there is no assumption that an agent only cooperates with agents bearing the same tag as itself: by default it cooperates with anyone having a tag similar to the one of agents that were cooperative in the past. This means that there can be groups with which everyone cooperates (e.g. "public" devices), but also that specific types of "symbiosis" can develop in which one group systematically seeks out members of a different group to cooperate with because of their complementary capabilities. This brings us to the more complex issue of the division of labor.

7.6 Learning to coordinate

After having ascertained that our devices can communicate and cooperate, we still need to make sure that the functions they perform satisfy the user. This desired functionality can be viewed as a complex of tasks that need to be executed. The tasks are mutually dependent in the sense that a certain task (e.g. locating a file) has to be accomplished before subsequent

tasks (e.g. downloading and playing the file) can be initiated. Each agent can either execute a task itself, or delegate it to another agent. Initially, we may assume that all agents that have a certain inbuilt functionality (e.g. playing a sound file) are equally competent at performing that type of task. However, in practice the satisfaction of the user can vary. For example, a recording is likely to be played with a higher sound quality by a surround sound system than by a PDA or television. By default, devices can use certain preprogrammed rules-of-thumb to decide who takes precedence (e.g. newer or more specialized devices are preferred to older, less specialized ones). Yet in an open environment there is no guarantee that such simple heuristics will produce the best result. Again, we may tackle this problem through individual learning coupled with collective self-organization.

Assume that the user regularly expresses his/her overall satisfaction with the ambient intelligence environment (e.g. explicitly by clicking on a scale from one to ten, or implicitly by facial or physiological cues that express happiness/unhappiness). This score can be used as a feedback signal to the network of devices, allowing it to reinforce the more successful rules, while weakening the less effective ones. We will assume that the agent who delegated a task will increase its trust in the competence of the agent that successfully performed that task, and thus increase its probability to delegate a similar task to the same agent in the future. Otherwise, it will reduce its trust. As demonstrated by the simulation of Gaines (1994), this assumption is sufficient to evolve a self-reinforcing division of labor where tasks are delegated to the most "expert" agents.

However, when the tasks are mutually dependent, selecting the right specialist to carry out a task is not sufficient: First the preparatory tasks have to be done by the right agents, in the right order. When the agents do not know a priori what the right order is, they can randomly attempt to execute or delegate a task, and, if this fails, pick out another task. Eventually they will find a task they can execute, either because it requires no preparation, or because a preparatory task has already been accomplished by another agent. Each completed task enables the accomplishment of a series of directly dependent tasks. In this way the overall problem will eventually be solved. In each problem cycle, agents will learn better when to take on which task by themselves, or when to delegate it to a specific agent.

We expect that this learned organization will eventually stabilize into a system of efficient, coordinated actions, adapted to the task structure, just as language games converge to a shared vocabulary (De Vylder and Tuyls, 2006) . When new devices are added to the system, system and device should mutually adapt, producing a new organization. While no single

agent knows how to tackle the entire problem, the knowledge has been "distributed" across the system. The "tags" that identify agents, and the learned associations between a tag and the competence for a particular task, play the role of a mediator (Heylighen, 2003a), delegating tasks to the right agents and coordinating their interactions so that the problem is tackled as efficiently as possible.

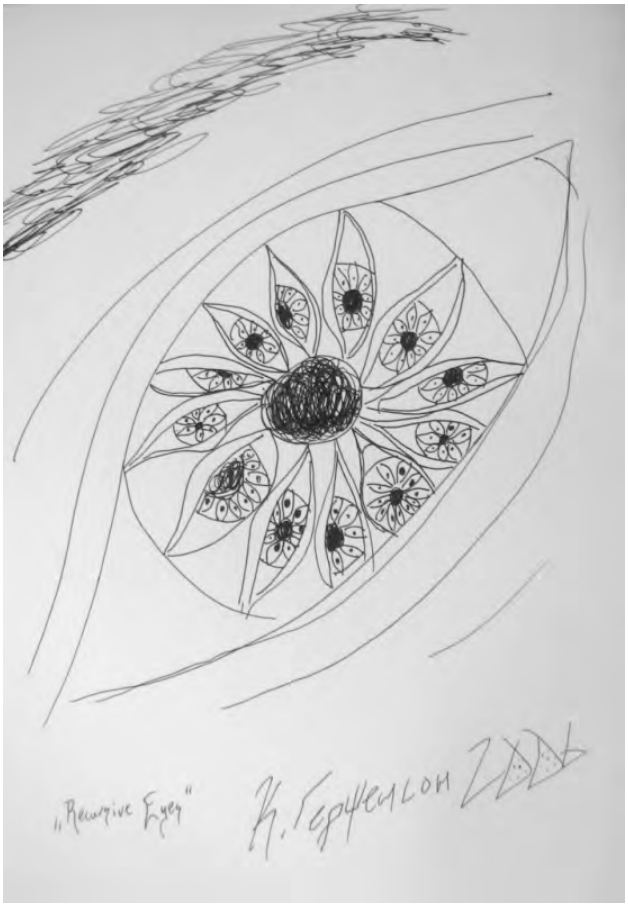
7.7 Conclusions

We cannot keep on adding functions to personal computers. They serve as text editors, game consoles, televisions, home cinemas, radios, agendas, music players, gateway to the Internet, etc. Such general devices will never produce the same quality as specialized appliances. Our PCs are like ducks: they can swim, but not as well as fish; fly, but not as well as hawks; and walk, but not as well as cats. Rather than integrate so many functions in a single device, it seems preferable to entrust them to an ever expanding network of specialized devices that is kept coordinated through an ongoing process of self-organization. We have described a number of general requirements and approaches that may enable our artifacts to learn the most effective way of cooperation.

In our overall scenario, we have assumed that standard functions and interaction rules are preprogrammed by a global community (the human in the loop) to handle the most common, default situations, but that the system is moreover ready to extend its own capabilities, adapting to newly encountered tasks, situations, or devices. This ability to adapt should be already present in the interaction rules. The adaptation may be achieved through the self-organization of the system of agents, using recurrent, "game-like" interactions, in which the agents learn what messages mean and who they can trust to perform which task. Most of this can happen outside of, or in parallel with, their normal "work", using idle processing power to explore many different communication and collaboration configurations. Thus, we can imagine that our future, intelligent devices, like young animals or children, will learn to become more skillful by exploring, "playing games" with each other, and practicing uncommon routines, so as to be prepared whenever the need for this kind of coordinated action appears.

Chapter 8

Conclusions



"Recursive Eyes"
Carlos Gershenson
2006.
Gel on paper.

8.1 Achievements

"(...) our brains are only minuscule fragments of the universe, much too small to hold all the facts of the world but not too idle to speculate about them"
—Valentino Braitenberg, Vehicles, p. 1.

The central contribution of this book was the Methodology to design and control self-organizing systems, presented in Chapter 4. Its conceptual framework allows the description of any engineered system as a collection of agents, where the goals of the system are determined by the purpose set by the designer. As part of the new insights provided, the "satisfaction" of the system was put forward as a measure of the degree at which the goals of the system have been reached. As a design principle, we should *not* try to increase the satisfaction of the elements of the system, since this can lead to suboptimal situations (Machol, 1965; Heylighen, 1992). The Methodology proposes to use as a design principle the reduction of the friction between elements, which will lead to an increase of the system satisfaction. Different ways in which this can be done were identified, to facilitate the design and implementation of *mediators* (Michod, 2003; Heylighen, 2003a) that will steer the system through a dynamic problem domain.

The Methodology was exemplified with three case studies: traffic lights, bureaucracies, and artifacts in an Ambient Intelligence (AmI) scenario (Chapters 5, 6, and 7). The simulations developed to test the self-organizing traffic lights showed that these can improve considerably the traffic flow in cities, as the traffic lights adapt to the current traffic situation. They achieve this with very simple methods, which promote the organization of cars into platoons. This organization is then exploited by both cars and traffic lights to reduce potential "friction": obtaining green lights quickly and leaving space between platoons that other platoons can use to cross an intersection. In the case of self-organizing bureaucracies, different ways in which friction can be reduced were identified, namely by reducing delays, leading to better performance and satisfaction. A simple computational model—random agent networks—was introduced to illustrate quantitatively the benefits of self-organization in bureaucracies. The self-organizing artifacts was the least developed case study, i.e. without developing a simulation. Nevertheless, requirements were identified to allow the communication, cooperation, and coordination of AmI artifacts. These requirements would reduce potential friction during artifact interactions, improving the efficiency of AmI systems.

In the first chapters (2 and 3), conceptual and philosophical issues related to complexity and self-organization were discussed, proposing practical notions. The problem with the concepts of complexity and self-organization is that we cannot have a "crisp" inclusive-exclusive definition, where systems are complex or not, self-organizing or not. We can only describe complexity and self-organization with *gradual* notions. There are different degrees of complexity in any system, and the notion proposed here attempted to clarify and quantify the differences of complexity between systems. As for self-organization, it was proposed that instead of characterizing *a class* of systems, it offers *a way of describing* systems, since all dynamical systems with an attractor can be said to be self-organizing, if we decide to call their attractor an organized state (Gershenson and Heylighen, 2003; Ashby, 1962). Still, the concept of self-organizing system, where elements interact to achieve dynamically a global function or behaviour, can be useful to design and engineer systems which have non-stationary problem domains.

This research is still ongoing, so it is difficult to measure its potential impact. Nevertheless, the current results are encouraging enough—with theoretical and practical consequences—to keep on developing self-organizing systems to solve complex problems, and continue refining the Methodology.

8.1.1 Limitations

The main limitation of the Methodology is also its main virtue: its generality. It has several advantages to have a general Methodology, since it can be applied to any domain. The drawback lies in the fact that more shall be done to apply it to a particular problem (remember the Generality/Particularity trade-off described in Section 4.3.2). Thus, it will not necessarily be the best in all problem domains. In other words, the Methodology is general *in theory*, since it can be used to describe any self-organizing system. However, *in practice*, particular approaches should be more efficient for an established domain. Still, the aim of the Methodology is to be useful as a starting point in domains without particular methods.

This Methodology will never be "finished", for experience gained in applying it will be used to improve it. Further formalizations and explorations need to be made before it can be widely applied, but the work presented here and the results obtained within the case studies are encouraging enough to continue this research.

The theoretical concepts covered in the first chapters can also be im-

proved. A working notion of self-organization was given, but this could be formalized as it was done with the notion of complexity (Equation 2.1). A similar formula should be developed to compare the organization of two systems or two configurations of a system. This would also provide a better understanding of the concept of "organization".

The case studies only showed partially the Methodology, as none of them has been implemented in a real situation. Having covered in this thesis several different domains, there was no time to go too deeply into any of them. The most developed one, self-organizing traffic lights, may start a pilot study using few intersections in Brussels this year. The solutions proposed for improving communication flows within self-organizing bureaucracies are very general, and it would be desirable to apply them to a real organization. Also, random agent networks are a very abstract model. This should be further explored, refined and applied to a more particular problem domain. This is also the case for the self-organizing artifacts: solutions were proposed, but we will know how good they are only after they are tested in real systems.

8.2 Future Work

There are many different directions I would like to explore:

- A software platform for agent based modeling using the Methodology would enable designers and engineers to develop and test self-organizing systems in a visual and accessible fashion. This would make the feedback between Modeling and Simulation much more fluid, increasing the speed of development and by its use refining the Methodology.

- The case studies are still being developed. Currently we are studying the possibility of implementing a pilot project in Brussels of self-organizing traffic lights. Random agent networks should be further explored, potentially suggesting additional benefits to bureaucracies and other organizations. Different updating schemes (Gershenson, 2002a, 2004c) should also be studied. The suggestions for self-organizing artifacts should be tested, and promoted within the Ambient Intelligence and Autonomic Communications communities, with feedback from the current developments in those fields.

- New case studies where the Methodology can be useful are already envisaged. One of them would be "self-organizing democracies", ex-

tending the work of Rodriguez and Steinbock (2004); Rodriguez et al. (2007). This proposes the use of a social network-based method to effectively represent decision-making at the society level with the participation of only few individuals. In other words, only few people would need to vote to obtain a meaningful representation of the will of the people. To introduce self-organization, reinforcement learning could be used to adjust the social network based on the voting history, i.e. reinforcing links to people with similar votes (synergy) and weaken links to people with different preferences (friction). In theory, this would increase the performance of the social decision-making, having a better representation of the society. Another case study would be "self-organizing laws", where laws, regulations and norms would not be stationary but adaptive according to the current situation. This does not mean that laws would become unpredictable, but that their effective application would depend on the actual circumstances. Such an approach seems promising also for sustainable development, as Paciotti et al. (2005) have shown.

- Early in my PhD, I explored self-organizing traffic control (Gershenson, 2003), trying to use flocking algorithms to promote platoon formation in freeways. The results were somehow unexpected, since "selfish" agents (trying to go as fast as possible) performed best, as every car tried to pass other cars, increasing the global speed; while the self-organizing (flocking) methods induced oscillations that hampered global performance. It would be useful to revise the simulation and the methods, since this is another potential area where self-organization can be applied (Treiber and Helbing, 2001).

- The conceptual framework proposed in Section 4.2 could be potentially useful for game theory, e.g. to study different means by which cooperation can evolve and be maintained (Axelrod, 1984) or the evolution of social norms (Pacheco et al., 2006). Its relationship with mechanism design (Dash et al., 2003) should also be explored.

- There are many open philosophical questions (can they ever be closed?): there is still no consensus on the notions of complexity or self-organization. Do we need a definition we all agree on, or do we need to agree on not having a strict definition? As we saw, different notions are more appropriate for different contexts, so it seems that there will be no general agreement. Still, there seem to be common features across contexts that may help us understand better these concepts.

8.3 Philosophical Implications

"Knowledge brings more questions than answers"

8.3.1 Objectivity or Subjectivity? Contextuality!

As we saw in Chapters 2 and 3, the observer is essential in determining features of a system such as complexity, self-organization, cognition, intelligence, life, and consciousness. Does this mean that we are doomed to subjectivity? No, since all of these properties are applied to an external abs-being, i.e. independent of the observer, and can be scientifically contrasted (Popper, 2002). Like this, we can hope to find pragmatically when it is more *useful* to attribute certain properties to certain systems, but always within a certain *context* (Gershenson, 2002c; Edmonds, 2001). This is because the "usefulness" of describing an abs-being with a particular rel-being may change across contexts. This idea was already proposed, among others, by constructivism (von Glasersfeld, 1984; Riegler, 2005) and second-order cybernetics (Heylighen and Joslyn, 2001), but seems to be lacking in most sciences still, as researchers keep on trying to find a purely objective and absolutely true nature of reality. We can only approach reality as observers, so we cannot ignore our inherent subjectivity. But a feedback between reasoning and experience will be able to help us discern which models are more appropriate for different circumstances.

The conflict between objectivity and subjectivity goes back to the debate between rationalism and empiricism, which can be traced to the opposition between the teachings of Parmenides and Heraclitus. But actually, when they spoke about the being, the former referred to the abs-being (static, unique, absolute), while the latter to rel-beings (dynamic, multiple, relative). Which one was right? Neither and both, since they each refer to different rel-beings appropriate for different contexts. What we gain with contextuality is the ability to switch between approaches as different contexts demand, since we understand that in practice a "true" model or description of the world cannot be reached.

This has implications not only for science and philosophy, but also for society in general. Contextuality gives us the possibility of avoiding social friction by Tolerance, Courtesy, and Compromise. With a purely "objective" worldview (Aerts et al., 1994), one cannot accept that more than one "truth" may exist, leading to fanaticism and orthodoxy, e.g. Nazism, terrorism. A purely "subjective" worldview would not be able to discern which ideas are useful for society, since it gives equal value to any of them, e.g. certain non-academic postmodern worldviews. However, a "contex-

tual" worldview is able to find rel-beings valid for specific contexts (contrasted with experience (Popper, 2002)) and to accept different rel-beings in different contexts. Like this, it not only tolerates, but even interacts with them[1]. Only a contextual worldview will be able to reduce friction and promote synergy to increase social satisfaction.

8.3.2 The Benefits of Self-organization

"We can only see a short distance ahead, but we can see plenty there that needs to be done" —Alan M. Turing

The case studies presented in this book tried to show the benefits of modeling complex systems as self-organizing.

The self-organizing traffic lights (Chapter 5) are able to coordinate dynamically according to actual traffic densities, adapting effectively to changes in the traffic densities and flows. Moreover, they are robust and distributed, having several benefits above traditional "blind" methods apart from considerably reducing travel times.

The ideas and simulations presented for self-organizing bureaucracies (Chapter 6) showed that using the Methodology in organizations may enable them to improve constantly their performance, adapting to changing demands.

Finally, the ideas presented to achieve the self-organization of artifacts (Chapter 7) argued that it is possible to design adaptive technologies that will be able to cope with changes of specifications, which will certainly occur in our information-centered world. This would enable devices to learn by themselves new meanings and ways of interaction, potentially producing novel functionalities as they coordinate.

As we can see, one of the main benefits of an engineered self-organizing system is that of *adaptation*. This would be redundant for a stationary problem domain. However, most complex systems have non-stationary problem domains, where solutions change constantly. A self-organizing system will be able to seek by itself new solutions, having more potential and robustness than a traditional approach.

Any system is liable to make mistakes (and *will* make them in an unpredictable environment). But a good system will *learn* from its mistakes. This is the basis for adaptation. It is pointless to attempt to build a "perfect" system, since it is not possible to predict future interactions with its environment. What should be done is to build systems that can adapt to

[1] Paradoxes are not an impediment for this (Gershenson, 1998b, 1999).

their unexpected future and are robust enough not to be destroyed in the attempt. Self-organization provides one way to achieve this, but there is still much to be done to harness its full potential.

BIBLIOGRAPHY

AERTS, D., APOSTEL, L., MOOR, B. D., HELLEMANS, S., MAEX, E., BELLE, H. V., AND VAN DER VEKEN, J. (1994). *World views, From Fragmentation towards Integration*. VUB-PRESS. 144

ALDANA, M. (2003). Boolean dynamics of networks with scale-free topology. *Physica D* **185** (1): 45–66. URL http://dx.doi.org/10.1016/S0167-2789(03)00174-X. 112, 115

ALDANA-GONZÁLEZ, M., COPPERSMITH, S., AND KADANOFF, L. P. (2003). Boolean dynamics with random couplings. In Perspectives and Problems in Nonlinear Science. A Celebratory Volume in Honor of Lawrence Sirovich, E. Kaplan, J. E. Marsden, and K. R. Sreenivasan, (Eds.). *Springer Appl. Math. Sci. Ser.*. URL http://www.fis.unam.mx/%7Emax/PAPERS/nkreview.pdf. 112

ANDERSON, P. (1999). Complexity theory and organization science. *Organization Science* **10** (3) (May-June): 216–232. 99

ANDERSON, P., MEYER, A., EISENHARDT, K., CARLEY, K., AND PETTIGREW, A. (1999). Introduction to the special issue: Applications of complexity theory to organization science. *Organization Science* **10** (3) (May-June): 233–236. 99

ANDERSON, P. W. (1972). More is different. *Science* **177**: 393–396. 12

ASHBY, W. R. (1947a). The nervous system as physical machine: With special reference to the origin of adaptive behavior. *Mind* **56** (221) (January): 44–59. 2, 21, 47

ASHBY, W. R. (1947b). Principles of the self-organizing dynamic system. *Journal of General Psychology* **37**: 125–128. 2

ASHBY, W. R. (1956). *An Introduction to Cybernetics*. Chapman & Hall, London. URL http://pcp.vub.ac.be/ASHBBOOK.html. 20, 38, 53, 88, 108

ASHBY, W. R. (1962). Principles of the self-organizing system. In *Principles of Self-Organization*, H. V. Foerster and G. W. Zopf, Jr., (Eds.). Pergamon, Oxford, pp. 255–278. 24, 29, 30, 32, 141

AULIN, A. Y. (1979). The law of requisite hierarchy. *Kybernetes* **8**: 259–266. 108

AXELROD, R. (2005). Advancing the art of simulation in the social sciences. In *Handbook of Research on Nature Inspired Computing for Economy and Management*, J.-P. Rennard, (Ed.). Idea Group. URL http://tinyurl.com/ymdk9n. 99, 101

AXELROD, R. AND COHEN, M. D. (2000). *Harnessing Complexity: Organizational Implications of a Scientific Frontier*. Free Press, New York. 99

AXELROD, R. M. (1984). *The Evolution of Cooperation*. Basic Books, New York. 40, 52, 59, 134, 143

BAR-YAM, Y. (1997). *Dynamics of Complex Systems*. Studies in Nonlinearity. Westview Press. URL http://www.necsi.org/publications/dcs/. 2, 10, 38

BAR-YAM, Y. (2005). About engineering complex systems: Multiscale analysis and evolutionary engineering. In *Engineering Self Organising Sytems: Methodologies and Applications*, S. Brueckner, G. Serugendo-Di Marzo, A. Karageorgos, and R. Nagpal, (Eds.). Lecture Notes in Artificial Intelligence, vol. 3464. Springer, 16–31. URL http://necsi.org/projects/yaneer/ESOA04.pdf. 45, 108

BATTRAM, A. (2002). *Navigating Complexity: The Essential Guide to Complexity Theory in Business and Management*. Spiro Press. 22

BAZZAN, A. L. C. (2005). A distributed approach for coordination of traffic signal agents. *Autonomous Agents and Multiagent Systems* **10** (1) (March): 131–164. URL http://tinyurl.com/2vld8y. 93

BEER, R. D. (1990). *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology*. Academic Press. 40

BEER, S. (1966). *Decision and Control: The Meaning of Operational Research and Management Cybernetics*. John Wiley and Sons, New York. 17, 25, 29, 30, 38, 98

BELPAEME, T. (2001). Reaching coherent colour categories through communication. In *Proc. 13th Belgium-Netherlands Conference on AI*, B. K. et al., (Ed.). pp. 41–48. 132, 133

BERLEKAMP, E. R., CONWAY, J. H., AND GUY, R. K. (1982). *Winning Ways for Your Mathematical Plays*. Vol. 2: Games in Particular. Academic Press, London. URL http://tinyurl.com/yeplyq. 12

BERNERS-LEE, T., HENDLER, J., AND LASSILA, O. (2001). The semantic web. *Scientific American*. URL http://tinyurl.com/i59p. 38

BIHAM, O., MIDDLETON, A. A., AND LEVINE, D. (1992). Self-organization and a dynamical transition in traffic-flow models. *Physical Review A* **46**: R6124–R6127. URL http://dx.doi.org/10.1103/PhysRevA.46.R6124. 66

BOEHM, B. W. (1988). A spiral model of software development and enhancement. *Computer* **21** (5): 61–72. URL http://dx.doi.org/10.1109/2.59. 56

BOLLEN, J. AND HEYLIGHEN, F. (1996). Algorithms for the self-organisation of distributed, multi-user networks. possible application to the future world wide web. In *Cybernetics and Systems '96*, R. Trappl, (Ed.). Vol. 911-916. Austrian Society for Cybernetics. URL http://pcp.vub.ac.be/Papers/AlgorithmsWeb.pdf. 99, 105, 110

BONABEAU, E., DORIGO, M., AND THERAULAZ, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. Santa Fe Institute Studies in the Sciences of Complexity. Oxford University Press, New York. 38, 132

BRIN, S. AND PAGE, L. (1998). The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web*. pp. 107–117. URL http://infolab.stanford.edu/~backrub/google.html. 50

BROCKFELD, E., BARLOVIC, R., SCHADSCHNEIDER, A., AND SCHRECKENBERG, M. (2001). Optimizing traffic lights in a cellular automaton model for city traffic. *Physical Review E* **64**: 056132. URL http://dx.doi.org/10.1103/PhysRevE.64.056132. 66

CAMAZINE, S., DENEUBOURG, J.-L., FRANKS, N. R., SNEYD, J., THERAULAZ, G., AND BONABEAU, E. (2003). *Self-Organization in Biological Systems*. Princeton University Press. URL http://www.pupress.princeton.edu/titles/7104.html. 24

CAMPBELL, D. T. (1974). 'Downward causation' in hierarchically organized biological systems. In *Studies in the Philosophy of Biology*, F. J. Ayala and T. Dobzhansky, (Eds.). Macmillan, 179–186. 31

CAPERA, D., GEORGÉ, J.-P., GLEIZES, M.-P., AND GLIZE, P. (2003). The AMAS theory for complex problem solving based on self-organizing cooperative agents. In *1st International Workshop on Theory and Practice of Open Computational Systems TAPOCS 2003 at IEEE 12th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises WETICE 2003*. p. 383. URL http://tinyurl.com/y88ok3. 56

CARLEY, K. AND PRIETULA, M. J., Eds. (1994). *Computational Organizational Theory*. Lawrence Erlbaum, Hillsdale, NJ. 99

CARLEY, K. M. (1997). Organizational adaptation. *Annals of Operations Research* **75**: 25–47. 99

CHEN, G. AND YU, X. H., Eds. (2003). *Chaos Control: Theory and Applications*. Lecture Notes in Control and Information Sciences, vol. 292. Springer Verlag. 20

CHOWDHURY, D. AND SCHADSCHNEIDER, A. (1999). Self-organization of traffic jams in cities: Effects of stochastic dynamics and signal periods. *Physical Review E* **59**: R1311–R1314. URL http://dx.doi.org/10.1103/PhysRevE.59.R1311. 66

COMFORT, L. K. (1994). Self-organization in complex systems. *Journal of Public Administration Research and Theory: J-PART* **4** (3) (July): 393–410. 99

COOLS, S. B. (2006). A realistic simulation for self-organizing traffic lights. Unpublished BSc Thesis, Vrije Universiteit Brussel. URL http://cogprints.org/5229/. 6, 63, 65, 84, 86, 87

COOLS, S. B., GERSHENSON, C., AND D'HOOGHE, B. (2007). Self-organizing traffic lights: A realistic simulation. In *Self-Organization: Applied Multi-Agent Systems*, M. Prokopenko, (Ed.). Springer. URL http://uk.arxiv.org/abs/nlin.AO/0610040. 6, 63

CORNING, P. A. (2003). *Nature's Magic: Synergy in Evolution and the Fate of Humankind*. Cambridge University Press. URL http://www.complexsystems.org/magic.html. 41, 43

COTTON, T. (1996). Evolutionary fusion: A customer-oriented incremental lifecycle for Fusion. *Hewlett Packard Journal* **47** (4). URL http://www.hpl.hp.com/hpjournal/96aug/aug96a3.htm. 55

CYERT, R. M. AND MARCH, J. G. (1992). *A Behavioral Theory of the Firm*, 2nd ed. Blackwell Publishing. 99

DASH, R. K., JENNINGS, N. R., AND PARKES, D. C. (2003). Computational-mechanism design: A call to arms. *IEEE Intelligent Systems*: 40–47. Special Issue on Agents and Markets. URL http://tinyurl.com/39te6f. 47, 143

DASTANI, M., HULSTIJN, J., AND DER TORRE, L. V. (2001). Negotiation protocols and dialogue games. In *International Conference on Autonomous Agents*. ACM, pp. 180–181. 133

DE BOER, B. (1999). Self-organisation in vowel systems. Ph.D. thesis, Vrije Universiteit Brussel. URL http://tinyurl.com/tfp47. 24, 132

DE JONG, E. D. (2000). Autonomous formation of concepts and communication. Ph.D. thesis, Vrije Universiteit Brussel. URL http://www.cs.uu.nl/%7Edejong/thesis/. 4, 51, 132

DE VYLDER, B. AND TUYLS, K. (2006). How to reach linguistic consensus: A proof of convergence for the naming game. *Journal of Theoretical Biology* **242** (4) (October): 818–831. URL http://dx.doi.org/10.1016/j.jtbi.2006.05.024. 132, 133, 136

DE WOLF, T. AND HOLVOET, T. (2005). Towards a methodolgy for engineering self-organising emergent systems. In *Self-Organization and Autonomic Informatics (I)*, H. Czap, R. Unland, C. Branki, and H. Tianfield, (Eds.). Frontiers in Artificial Intelligence and Applications, vol. 135. IOS Press, 18–34. URL http://tinyurl.com/y3d8pe. 56

DE WOLF, T., SAMAEY, G., AND HOLVOET, T. (2005). Engineering self-organising emergent systems with simulation-based scientific analysis. In *Proceedings of the International Workshop on Engineering Self-Organising Applications*. Utrecht, The Netherlands,. URL http://tinyurl.com/y2vmau. 40

DECANIO, S. J. AND WATKINS, W. E. (1998). Information processing and organizational structure. *Journal of Economic Behavior and Organization* **36** (3): 275–294. URL http://dx.doi.org/10.1016/S0167-2681(98)00096-1. 100

DESANCTIS, G. AND MONGE, P. (1999). Introduction to the special issue: Communication processes for virtual organizations. *Organization Science* **10** (6) (November): 1047–7039. 102

DI MARZO SERUGENDO, G. (2004). Trust as an interaction mechanism for self-organising systems. In *International Conference on Complex Systems (ICCS'04)*, Y. Bar-Yam, (Ed.). 52

DI MARZO SERUGENDO, G., KARAGEORGOS, A., RANA, O. F., AND ZAMBONELLI, F., Eds. (2004). *Engineering Self-Organising Systems, Nature-Inspired Approaches to Software Engineering*. Lecture Notes in Computer Science, vol. 2977. Springer. Revised and extended papers presented at the Engineering Self-Organising Applications Workshop, ESOA 2003, held at AAMAS 2003 in Melbourne, Australia, in July 2003 and selected invited papers from leading researchers in self-organisation. 38

DORIGO, M. AND STÜTZLE, T. (2004). *Ant Colony Optimization*. MIT Press. 50

DORIGO, M., TRIANNI, V., ŞAHIN, E., GROSS, R., LABELLA, T. H., BALDASSARRE, G., NOLFI, S., DENEUBOURG, J.-L., MONDADA, F., FLOREANO, D., AND GAMBARDELLA, L. (2004). Evolving self-organizing behaviors for a swarm-bot. *Autonomous Robots* **17** (2-3): 223–245. URL http://www.swarm-bots.org. 33

DRESNER, K. AND STONE, P. (2004). Multiagent traffic management: A reservation-based intersection control mechanism. In *The Third International Joint Conference on Autonomous Agents and Multiagent Systems*. URL http://tinyurl.com/w67yz. 94

DUNBAR, R. I. M. (1993). Coevolution of neocortical size, group size and language in humans. *Behavioral and Brain Sciences* **16** (4): 681–735. URL http://tinyurl.com/3ybeet. 109

DUNBAR, R. I. M. (2003). The social brain: mind, language and society in evolutionary perspective. *Ann. Rev. Anthrop* **32**: 163–181. URL http://tinyurl.com/yc24o5. 20

DURKHEIM, É. (1893). *The Division of Labor in Society*. The Free Press, New York. Translated by George Simpson (1984). 60

EDMONDS, B. (1999). What is complexity?: the philosophy of complexity per se with application to some examples in evolution. In *The Evolution of Complexity*, F. Heylighen, J. Bollen, and A. Riegler, (Eds.). Kluwer, Dordrecht, 1–18. URL http://bruce.edmonds.name/evolcomp/. 12

EDMONDS, B. (2000). Complexity and scientific modelling. *Foundations of Science* **5**: 379–390. URL http://cfpm.org/cpmrep23.html. 12

EDMONDS, B. (2001). What if all truth is context-dependent? Tech. Rep. 01-77, CPM. URL http://cfpm.org/cpmrep77.html. 32, 144

EDMONDS, B. (2005). Using the experimental method to produce reliable self-organised systems. In *Engineering Self Organising Sytems: Methodologies and Applications*, S. Brueckner, G. Serugendo-Di Marzo, A. Karageorgos, and R. Nagpal, (Eds.). Lecture Notes in Artificial Intelligence, vol. 3464. Springer, pp. 84–99. URL http://cfpm.org/cpmrep131.html. 57, 59

EDMONDS, B. AND BRYSON, J. (2004). The insufficiency of formal design methods—the necessity of an experimental approach for the understanding and control of complex mas. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents & Multi Agent Systems (AAMAS'04)*, N. Jennings, C. Sierra, L. Sonenberg, and M. Tambe, (Eds.). ACM Press, New York, pp. 938–945. URL http://cfpm.org/cpmrep128.html. 59

EPSTEIN, J. M. AND AXTELL, R. L. (1996). *Growing Artificial Societies: Social Science from the Bottom Up*. Brookings Institution Press MIT Press. URL http://www.brookings.org/press/books/artifsoc.htm. 99

ESCOBAR, R. AND DE LA ROSA, A. (2003). Architectural design for the survival optimization of panicking fleeing victims. In *Advances in Artificial Life, 7th European Conference, ECAL 2003 LNAI 2801*, W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, (Eds.). Springer-Verlag, pp. 97–106. 41

FAIETA, B. AND HUBERMAN, B. A. (1993). Firefly: A synchronization strategy for urban traffic control. Tech. Rep. SSL-42, Xerox PARC, Palo Alto. 66, 78, 93

FEDERAL HIGHWAY ADMINISTRATION. (1998). *Traffic Control Systems Handbook*. U.S. Department of Transportation. 65

FELTZ, B., CROMMELINCK, M., AND GOUJON, P., Eds. (2006). *Self-organization and Emergence in Life Sciences*. Synthese Library, vol. 331. Springer. 24

FERNÁNDEZ, P. AND SOLÉ, R. (2004). The role of computation in complex regulatory networks. In *Power Laws, Scale-Free Networks and Genome Biology*, E. V. Koonin, Y. I. Wolf, and G. P. Karev, (Eds.). Landes Bioscience. URL http://arxiv.org/abs/q-bio.MN/0311012. 21, 35, 45, 52

FLACK, J. C., GIRVAN, M., DE WAAL, F. B. M., AND KRAKAUER, D. C. (2006). Policing stabilizes construction of social niches in primates. *Nature* **439**: 426–429. URL http://tinyurl.com/y2eo2t. 48

FOULADVAND, M. E., SADJADI, Z., AND SHAEBANI, M. R. (2004a). Characteristics of vehicular traffic flow at a roundabout. *Physical Review E* **70**: 046132. URL http://dx.doi.org/10.1103/PhysRevE.70.046132. 91

FOULADVAND, M. E., SADJADI, Z., AND SHAEBANI, M. R. (2004b). Optimized traffic flow at a single intersection: Traffic responsive signalization. *J. Phys. A: Math. Gen.* **37**: 561–576. URL http://dx.doi.org/10.1088/0305-4470/37/3/002. 70

GAINES, B. R. (1994). The collective stance in modeling expertise in individuals and organizations. *Int. J. Expert Systems* **71**: 22–51. 52, 136

GÄRDENFORS, P. (2000). *Conceptual Spaces: The Geometry of Thought*. Bradford Books. MIT Press. URL http://mitpress.mit.edu/0262572192. 51

GERSHENSON, C. (1998a). Control de tráfico con agentes: CRASH. In *Memorias XI Congreso Nacional ANIEI*. Xalapa, México. URL http://tinyurl.com/ybgwk8. 94

GERSHENSON, C. (1998b). Lógica multidimensional: Un modelo de lógica paraconsistente. In *Memorias XI Congreso Nacional ANIEI*. Xalapa, México, pp. 132–141. URL http://tinyurl.com/y9hb4e. 11, 145

GERSHENSON, C. (1999). Modelling emotions with multidimensional logic. In *Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society (NAFIPS '99)*. IEEE Press, New York City, NY, pp. 42–46. URL http://tinyurl.com/yek3ms. 11, 145

GERSHENSON, C. (2001). Artificial societies of intelligent agents. Unpublished BEng Thesis. URL http://cogprints.org/1477/. 32

GERSHENSON, C. (2002a). Classification of random Boolean networks. In *Artificial Life VIII: Proceedings of the Eight International Conference on Artificial Life*, R. K. Standish, M. A. Bedau, and H. A. Abbass, (Eds.). MIT Press, pp. 1–8. URL http://alife8.alife.org/proceedings/sub67.pdf. 126, 142

GERSHENSON, C. (2002b). Complex philosophy. In *Proceedings of the 1st Biennial Seminar on Philosophical, Methodological & Epistemological Implications of Complexity Theory*. La Habana, Cuba. URL http://uk.arXiv.org/abs/nlin.AO/0109001. 6, 9, 10, 13, 15, 31, 32, 34, 45, 169

GERSHENSON, C. (2002c). Contextuality: A philosophical paradigm, with applications to philosophy of cognitive science. POCS Essay, COGS, University of Sussex. URL http://cogprints.org/2621/. 32, 144, 169

GERSHENSON, C. (2002d). Philosophical ideas on the simulation of social behaviour. *Journal of Artificial Societies and Social Simulation* **5** (3). URL http://jasss.soc.surrey.ac.uk/5/3/8.html. 101

GERSHENSON, C. (2003). Self-organizing traffic control: First results. Unpublished. URL http://uk.arxiv.org/abs/nlin.AO/0309039. 143

GERSHENSON, C. (2004a). Cognitive paradigms: Which one is the best? *Cognitive Systems Research* **5** (2) (June): 135–156. URL http://dx.doi.org/10.1016/j.cogsys.2003.10.002. 16, 32, 51, 100

GERSHENSON, C. (2004b). Introduction to random Boolean networks. In *Workshop and Tutorial Proceedings, Ninth International Conference on the Simulation and Synthesis of Living Systems (ALife IX)*, M. Bedau, P. Husbands, T. Hutton, S. Kumar, and H. Suzuki, (Eds.). Boston, MA, pp. 160–173. URL http://uk.arxiv.org/abs/nlin.AO/0408006. 13, 112, 125

GERSHENSON, C. (2004c). Updating schemes in random Boolean networks: Do they really matter? In *Artificial Life IX Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems*, J. Pollack, M. Bedau, P. Husbands, T. Ikegami, and R. A. Watson, (Eds.). MIT Press, pp. 238–243. URL http://uk.arxiv.org/abs/nlin.AO/0402006. 126, 142

GERSHENSON, C. (2005). Self-organizing traffic lights. *Complex Systems* **16** (1): 29–53. URL http://uk.arxiv.org/abs/nlin.AO/0411066. 6, 63, 65, 99

GERSHENSON, C. (2006a). A general methodology for designing self-organizing systems. Tech. Rep. 2005-05, ECCO. URL http://uk.arxiv.org/abs/nlin.AO/0505009. 6, 9, 23, 32, 37, 100, 170

GERSHENSON, C. (2006b). Towards self-organizing bureaucracies. Tech. Rep. 2006-03, ECCO. URL http://uk.arxiv.org/abs/nlin.AO/0603045. 6, 97

GERSHENSON, C. (2007a). Design and control of self-organizing systems. Ph.D. thesis, Vrije Universiteit Brussel, Brussels, Belgium. URL http://cogprints.org/5442/. 6

GERSHENSON, C. (2007b). Towards a general methodology for designing self-organizing systems. In *Complexity, Science and Society*, J. Bogg and R. Geyer, (Eds.). Radcliffe Publishing, Oxford. 6

GERSHENSON, C., AERTS, D., AND EDMONDS, B., Eds. (2007). *Philosophy and Complexity*. Worldviews, Science and Us. World Scientific, Singapore. URL http://www.worldscibooks.com/chaos/6372.html. 10

GERSHENSON, C., BROEKAERT, J., AND AERTS, D. (2003). Contextual random Boolean networks. In *Advances in Artificial Life, 7th European Conference, ECAL 2003 LNAI 2801*, W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, (Eds.). Springer-Verlag, pp. 615–624. URL http://uk.arxiv.org/abs/nlin.AO/0303021. 19, 53

GERSHENSON, C. AND HEYLIGHEN, F. (2003). When can we call a system self-organizing? In *Advances in Artificial Life, 7th European Conference, ECAL 2003 LNAI 2801*, W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, (Eds.). Springer, Berlin, pp. 606–614. URL http://uk.arxiv.org/abs/nlin.AO/0303020. 6, 23, 32, 129, 141

GERSHENSON, C. AND HEYLIGHEN, F. (2004). Protocol requirements for self-organizing artifacts: Towards an ambient intelligence. In *Proceedings of International Conference on Complex Systems ICCS2004*, Y. Bar-Yam, (Ed.). Boston, MA. Also AI-Lab Memo 04-04. URL http://uk.arxiv.org/abs/nlin.AO/0404004. 6, 51, 127

GERSHENSON, C. AND HEYLIGHEN, F. (2005). How can we think the complex? In *Managing Organizational Complexity: Philosophy, Theory and Application*, K. Richardson, (Ed.). Information Age Publishing, Chapter 3, pp. 47–61. URL http://uk.arxiv.org/abs/nlin.AO/0402023. 6, 9, 12, 45

GERSHENSON, C., KAUFFMAN, S. A., AND SHMULEVICH, I. (2006). The role of redundancy in the robustness of random boolean networks. In *Artificial Life X, Proceedings of the Tenth International Conference on the Simulation and Synthesis*

of Living Systems., L. M. Rocha, L. S. Yaeger, M. A. Bedau, D. Floreano, R. L. Goldstone, and A. Vespignani, (Eds.). MIT Press, pp. 35–42. URL http://uk.arxiv.org/abs/nlin.AO/0511018. 21, 35, 52, 125

GLD. (2001). Green Light District. URL http://tinyurl.com/yzxnqk. 84

HAKEN, H. (1981). Synergetics and the problem of selforganization. In *Self-Organizing Systems: An Interdisciplinary Approach*, G. Roth and H. Schwegler, (Eds.). Campus Verlag, New York, pp. 9–13. 24, 41, 89, 100

HAKEN, H. (1988). *Information and Self-organization: A Macroscopic Approach to Complex Systems*. Springer-Verlag, Berlin. 41

HALES, D. AND EDMONDS, B. (2003). Evolving social rationality for MAS using "tags". In *Proceedings of the 2nd International Conference on Autonomous Agents and Multiagent Systems*, J. S. Rosenschein, T. Sandholm, M. Wooldridge, and M. Yokoo, (Eds.). ACM Press, pp. 497–503. 52, 134

HAND, D. J. AND HENLEY, W. E. (1997). Statistical classification methods in consumer credit scoring: A review. *Journal of the Royal Statistical Society. Series A* **160** (3): 523–541. 106

HELBING, D. (1997). *Verkehrsdynamik*. Springer, Berlin. 64

HELBING, D., AMMOSER, H., AND KÜHNERT, C. (2006). Information flows in hierarchical networks and the capability of organizations to successfully respond to failures, crises, and disasters. *Physica A* **363** (1) (April): 141–150. URL http://dx.doi.org/10.1016/j.physa.2006.01.055. 108

HELBING, D., HERRMANN, H. J., SCHRECKENBERG, M., AND WOLF, D. E., Eds. (2000). *Traffic and Granular Flow '99: Social, Traffic, and Granular Dynamics*. Springer, Berlin. 64

HELBING, D. AND HUBERMAN, B. A. (1998). Coherent moving states in highway traffic. *Nature* **396**: 738–740. 64

HELBING, D., LÄMMER, S., AND LEBACQUE, J.-P. (2005). Self-organized control of irregular or perturbed network traffic. In *Optimal Control and Dynamic Games*, C. Deissenberg and R. F. Hartl, (Eds.). Springer, Dordrecht, 239–274. URL http://arxiv.org/abs/physics/0511018. 93

HELBING, D. AND VICSEK, T. (1999). Optimal self-organization. *New Journal of Physics* **1**: 13.1–13.17. URL http://tinyurl.com/yxwftj. 39, 100

HEYLIGHEN, F. (1989). Causality as distinction conservation: a theory of predictability, reversibility and time order. *Cybernetics and Systems* **20**: 361–384. 11

HEYLIGHEN, F. (1990a). Classical and non-classical representations in physics i. *Cybernetics and Systems* **21**: 423–444. 10, 14

HEYLIGHEN, F. (1990b). *Representation and Change. A Metarepresentational Framework for the Foundations of Physical and Cognitive Science*. Communication and Cognition, Gent. 11, 17

HEYLIGHEN, F. (1991). Modelling emergence. *World Futures: the Journal of General Evolution* **31**: 89–104. 21

HEYLIGHEN, F. (1992). Evolution, selfishness and cooperation. *Journal of Ideas* **2** (4): 70–76. 59, 140

HEYLIGHEN, F. (1994). Fitness as default: the evolutionary basis for cognitive complexity reduction. In *Cybernetics and Systems '94*, R. Trappl, (Ed.). World Science, Singapore, 1595–1602. 20

HEYLIGHEN, F. (1997). Publications on complex, evolving systems: A citation-based survey. *Complexity* **2** (5): 31–36. 22

HEYLIGHEN, F. (1999). Collective intelligence and its implementation on the web. *Computational and Mathematical Theory of Organizations* **5** (3): 253–280. 21, 132

HEYLIGHEN, F. (2003a). Mediator evolution: A general scenario for the origin of dynamical hierarchies. Tech. rep., Principia Cybernetica. URL http://pcp.vub.ac.be/Papers/MediatorEvolution.pdf. 41, 47, 64, 90, 135, 137, 140

HEYLIGHEN, F. (2003b). The science of self-organization and adaptivity. In *The Encyclopedia of Life Support Systems*, L. D. Kiel, (Ed.). EOLSS Publishers, Oxford. URL http://pcp.vub.ac.be/Papers/EOLSS-Self-Organiz.pdf. 2, 22, 24, 28, 99

HEYLIGHEN, F. AND CAMPBELL, D. T. (1995). Selection of organization at the social level: Obstacles and facilitators of metasystem transitions. *World Futures: the Journal of General Evolution* **45**: 181–212. URL http://pcp.vub.ac.be/Papers/SocialMST.pdf. 39

HEYLIGHEN, F., CILLIERS, P., AND GERSHENSON, C. (2007). Complexity and philosophy. In *Complexity, Science and Society*, J. Bogg and R. Geyer, (Eds.). Radcliffe Publishing, Oxford. URL http://uk.arxiv.org/abs/cs.CC/0604072. 10

HEYLIGHEN, F. AND GERSHENSON, C. (2003). The meaning of self-organization in computing. *IEEE Intelligent Systems*: 72–75. URL http://pcp.vub.ac.be/Papers/IEEE.Self-organization.pdf. 21, 24, 29, 34, 129

HEYLIGHEN, F. AND JOSLYN, C. (2001). Cybernetics and second order cybernetics. In *Encyclopedia of Physical Science and Technology*, 3rd ed., R. A. Meyers, (Ed.). Vol. 4. Academic Press, New York, 155–170. 19, 20, 24, 35, 144

HOBBS, J. R. (1985). Granularity. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*. Vol. 1. pp. 432–435. 26

HOFACKER, I. AND VETSCHERA, R. (2001). Algorithmical approaches to business process design. *Computers & Operations Research* **28** (13) (November): 1253–1275. URL http://dx.doi.org/10.1016/S0305-0548(00)00038-1. 98

HOLLAND, J. H. (1975). *Adaptation in natural and artificial systems*. The University of Michigan Press. 19, 98

HOLLAND, J. H. (1995). *Hidder Order: How Adaptation Builds Complexity*. Helix books. Addison-Wesley. 19, 34

HUNT, P. B., ROBERTSON, D. I., BRETHERTON, R. D., AND WINTON, R. I. (1981). SCOOT-a traffic responsive method of coordinating signals. Tech. rep., TRRL. 65, 90, 93

HUTCHINS, E. (1995). *Cognition in the Wild*. MIT Press. 99, 132

HUTCHINS, E. AND HAZELHURST, B. (1995). How to invent a lexicon: The development of shared symbols in interaction. In *Artificial Societies*, N. Gilbert and R. Conte, (Eds.). UCL Press. 132

ISTAG. (2001). Scenarios for ambient intelligence in 2010. Tech. rep., ISTAG. 128

JACOBSON, I., BOOCH, G., AND RUMBAUGH, J. (1999). *The Unified Software Development Process*. Addison-Wesley Object Technology Series. Addison-Wesley Longman Publishing Co., Inc., Boston, MA. 56

JAKOBI, N. (1997). Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behavior* **6** (2): 325–368. 54

JEN, E., Ed. (2005). *Robust Design: A Repertoire of Biological, Ecological, and Engineering Case Studies*. Santa Fe Institute Studies on the Sciences of Complexity. Oxford University Press. URL http://tinyurl.com/swtlz. 21, 35, 101

JENNINGS, N. R. (2000). On agent-based software engineering. *Artificial Intelligence* **117** (2): 277–296. URL http://dx.doi.org/10.1016/S0004-3702(99)00107-1. 56

JONES, P. M., CONTRACTOR, N., O'KEEFE, B., AND LU, S. C.-Y. (1994). Competence models and self-organizing systems: Towards intelligent, evolvable, collaborative support. In *Systems, Man, and Cybernetics, 1994. 'Humans, Information*

and Technology'., 1994 IEEE International Conference on. Vol. 1. pp. 367–372. URL http://dx.doi.org/10.1109/ICSMC.1994.399866. 38

KAELBLING, L. P., LITTMAN, M. L., AND MOORE, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* **4**: 237–285. URL http://arxiv.org/abs/cs.AI/9605103. 20, 49

KAUFFMAN, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology* **22**: 437–467. 13, 112

KAUFFMAN, S. A. (1993). *The Origins of Order*. Oxford University Press. 13, 51, 112

KAUFFMAN, S. A. (2000). *Investigations*. Oxford University Press. 42, 98

KELLY, K. (1994). *Out of Control*. Addison-Wesley, New York. 20, 22

KIMURA, M. (1983). *The Neutral Theory of Molecular Evolution*. Cambridge University Press, Cambridge. 42

KOHONEN, T. (2000). *Self-Organizing Maps*, 3rd ed. Springer. 24, 99

KOOPMAN, B. (1978). Entropy increase and group symmetry. In *Maximum Entropy Formalism*, M. Tribus and I. Levine, (Eds.). MIT Press. 28

KRAKAUER, D. C. AND ZANOTTO, P. M. A. (2006). Viral individuality and limitations of the life concept. In *Protocells: Bridging Nonliving and Living Matter*, S. Rasmussen., M. A. Bedau, L. Chen, D. Deamer, D. C. Krakauer, N. Packard, and D. P. Stadler, (Eds.). MIT Press. 32

LÄMMER, S., KORI, H., PETERS, K., AND HELBING, D. (2006). Decentralised control of material or traffic flows in networks using phase-synchronisation. *Physica A* **363** (1) (April): 39–47. URL http://dx.doi.org/10.1016/j.physa.2006.01.047. 93

LAYNE, K. AND LEE, J. (2001). Developing fully functional E-government: A four stage model. *Government Information Quarterly* **18**: 122–136. URL http://dx.doi.org/10.1016/S0740-624X(01)00066-1. 59, 106

LENAERTS, T. (2003). Different levels of selection in artificial evolutionary systems: Analysis and simulation of selection dynamics. Ph.D. thesis, Vrije Universiteit Brussel. 43, 52

LENDARIS, G. G. (1964). On the definition of self-organizing systems. *Proceedings of the IEEE* **52** (3) (March): 324–325. URL http://tinyurl.com/23zlnb. 24

LEVINTHAL, D. A. AND WARGLIEN, M. (1999). Landscape design: Desiging for local action in complex worlds. *Organization Science* **10** (3) (May). 99

LEVITT, B. AND MARCH, J. G. (1988). Organizational learning. *Annual Review of Sociology* **14**: 319–338. URL http://dx.doi.org/10.1146/annurev.so.14.080188.001535. 99, 101

LISSACK, M. R. (1999). Complexity: the science, its vocabulary, and its relation to organizations. *Emergence* **1** (1). 99

LUCK, M., MCBURNEY, P., SHEHORY, O., WILLMOTT, S., AND THE AGENTLINK COMMUNITY. (2005). *Agent Technology: Computing as Interaction. A Roadmap for Agent Based Computing*. University of Southampton. URL http://www.agentlink.org/roadmap/. 106

MACHOL, R. E., Ed. (1965). *System engineering handbook*. McGraw-Hill. 59, 140

MAES, P. (1994). Modeling adaptive autonomous agents. *Artificial Life* **1** (1&2): 135–162. URL http://tinyurl.com/yhggsa. 39

MAMEI, M., MENEZES, R., TOLKSDORF, R., AND ZAMBONELLI, F. (2006). Case studies for self-organization in computer science. *Journal of Systems Architecture* **52** (8-9) (August-September): 443–460. URL http://dx.doi.org/10.1016/j.sysarc.2006.02.002. 24

MARCH, J. G. (1978). Bounded rationality, ambiguity, and the engineering of choice. *Bell Journal of Economics* **9** (2) (Autumn): 587–608. 99

MARCH, J. G. (1991). Exploration and exploitation in organizational learning. *Organization Science* **2** (1): 71–87. 99, 101

MICHOD, R. E. (1997). Cooperation and conflict in the evolution of individuality. i. multi-level selection of the organism. *American Naturalist* **149**: 607–645. URL http://tinyurl.com/y95rj3. 43, 52

MICHOD, R. E. (2003). Cooperation and conflict mediation during the origin of multicellularity. In *Genetic and Cultural Evolution of Cooperation*, P. Hammerstein, (Ed.). MIT Press, Cambridge, MA, Chapter 16, pp. 261–307. URL http://tinyurl.com/y76639. 41, 47, 140

MILLER, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review* **63**: 81–97. URL http://www.well.com/~smalin/miller.html. 109

MILLER MEDINA, E. (2005). The state machine: Politics, ideology, and computation in Chile, 1964-1973. Ph.D. thesis, MIT. 59, 98

MIRAMONTES HERCOG, L. (2004). Co-evolutionary agent self-organization for city traffic congestion modeling. In *Genetic and Evolutionary Computation - GECCO 2004: Genetic and Evolutionary Computation Conference, Seattle, WA,*

USA, June 26-30, 2004. Proceedings, Part II. Springer-Verlag, pp. 993–1004. URL http://dx.doi.org/10.1007/b98645. 66

MITCHELL, M. (1996). *An Introduction to Genetic Algorithms*. MIT Press. 20, 39, 49

MOREVTS. (2006). A more realistic vehicle traffic simulator. URL https://sourceforge.net/projects/morevts/. 84

MORGAN, G. (1996). *Images of Organization*, 2nd ed. SAGE Publications. 99

MORIN, E. (2006). Restricted complexity, general complexity. In *Philosophy and Complexity*, C. Gershenson, D. Aerts, and B. Edmonds, (Eds.). Worldviews, Science and Us. World Scientific. Translated from French by Carlos Gershenson. 10, 11

NAGEL, K. (2004). *Multi-Agent Transportation Simulation*. Book in progress. URL http://www.sim.inf.ethz.ch/papers/book/book.pdf. 66

NAGEL, K. AND SCHRECKENBERG, M. (1992). A cellular automaton modell for freeway traffic. *Journal of Physics I France* **2**: 2221–2229. 66

NEWMAN, M. E. J. (2003). The structure and function of complex networks. *SIAM Review* **45**: 167–256. URL http://arxiv.org/abs/cond-mat/0303516. 108

NICOLIS, G. AND PRIGOGINE, I. (1977). *Self-Organization in Non-Equilibrium Systems: From Dissipative Structures to Order Through Fluctuations*. Wiley. 24, 25, 29, 30

OHIRA, T. (1997). Autonomous traffic signal control model with neural network analogy. In *Proceedings of InterSymp'97: 9th International Conference on Systems Research, Informatics and Cybernetics*. Baden-Baden, Germany. SCSL-TR-97-004. URL http://arxiv.org/abs/adap-org/9704005. 93

OLIVEIRA, D., BAZZAN, A. L. C., AND LESSER, V. (2005). Using cooperative mediation to coordinate traffic lights: a case study. In *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS)*. New York, IEEE Computer Society, pp. 463–470. URL http://tinyurl.com/2vhot5. 93

OLIVEIRA, D., BAZZAN, A. L. C., SILVA, B. C., BASSO, E. W., NUNES, L., ROSSETTI, R. J. F., OLIVEIRA, E. C., SILVA, R., AND LAMB, L. C. (2006). Reinforcement learning based control of traffic lights in non-stationary environments: a case study in a microscopic simulator. In *Proceedings of the 4th European Workshop on Multi-Agent Systems (EUMAS06)*, B. Dunin-Keplicz, A. Omicini, and J. Padget, (Eds.). pp. 31–42. URL http://tinyurl.com/2stm9z. 93

PACHECO, J. M., SANTOS, F. C., AND CHALUB, F. A. C. C. (2006). Stern-judging: A simple, successful norm which promotes cooperation under indirect reciprocity. *PLoS Computational Biology* **2** (12): e178. URL http://dx.doi.org/10.1371/journal.pcbi.0020178. 143

PACIOTTI, B., HADLEY, C., HOLMES, C., AND BORGERHOFF MULDER, M. (2005). Grass-roots justice in Tanzania. *American Scientist* **93** (1): 58. URL http://dx.doi.org/10.1511/2005.1.58. 143

POPPER, K. R. (2002). *The Logic of Scientific Discovery (Routledge Classics)*. Routledge. URL http://tinyurl.com/yzmok3. 144, 145

PORCHE, I. AND LAFORTUNE, S. (1999). Adaptive look-ahead optimization of traffic signals. *Journal of Intelligent Transportation Systems* **4** (3): 209–254. URL http://tinyurl.com/yx55wp. 88, 90, 94

PREM, E. (1993). Understanding self-organization: What can the speaking lion tell us? Tech. Rep. TR-93-14, Oesterreichisches Forschungsinstitut fuer Artificial Intelligence, Wien. 32

PRIGOGINE, I. AND HERMAN, R. (1971). *Kinetic Theory of Vehicular Traffic*. Elsevier, New York. 64

RADNER, R. (1993). The organization of decentralized information processing. *Econometrica* **61** (5) (September): 1109–1146. 100

RAMAMOORTHY, P., ZHANG, S., FUBAO, C., AND RAMACHANDRAN, D. (1993). A new paradigm for the design of nonlinear dynamical systems and self-organizing systems. In *Intelligent Control, 1993., Proceedings of the 1993 IEEE International Symposium on*. pp. 571–576. URL http://dx.doi.org/10.1109/ISIC.1993.397634. 38

REED, C., NORMAN, T. J., AND JENNINGS, N. R. (2001). Negotiating the semantics of agent communication languages. *Computational Intelligence* **18** (2): 229–252. 133

RICCI, A., OMICINI, A., VIROLI, M., GARDELLI, L., AND OLIVA, E. (2006). Cognitive stigmergy: A framework based on agents and artifacts. In *Third International Workshop on Environments for Multiagent Systems (E4MAS 06)*. URL http://tinyurl.com/y46a4b. 106

RIEGLER, A. (2005). Editorial. the constructivist challenge. *Constructivist Foundations* **1** (1): 1–8. URL http://tinyurl.com/2ycs4p. 144

RIOLO, R., COHEN, M. D., AND AXELROD, R. M. (2001). Evolution of cooperation without reciprocity. *Nature* **414**: 441–443. 52, 134

RODRIGUEZ, M. A. AND STEINBOCK, D. (2004). Societal-scale decision making using social networks. In *North American Association for Computational Social and Organizational Science Conference Proceedings*. URL http://tinyurl.com/y9y948. 107, 143

RODRIGUEZ, M. A., STEINBOCK, D. J., WATKINS, J. H., GERSHENSON, C., BOLLEN, J., GREY, V., AND DEGRAF, B. (2007). Smartocracy: Social networks for collective decision making. In *Hawaii International Conference on Systems Science (HICSS)*. IEEE Computer Society. URL http://tinyurl.com/ybojp8. 143

ROJAS, R. (1996). *Neural Networks: A Systematic Introduction*. Springer, Berlin. 39, 51

ROOZEMOND, D. A. AND ROGIER, J. L. H. (2000). Agent controlled traffic lights. In *ESIT 2000; European Symposium on Intelligent Techniques*. 66

ROSEN, R. (1985). *Anticipatory Systems: Philosophical, Mathematical and Methodological Foundations*. Pergamon Press. URL http://tinyurl.com/y98zsx. 20, 31, 34

RUMELHART, D. E., MCCLELLAND, J. L., AND THE PDP RESEARCH GROUP, Eds. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. MIT Press. 20

RUSSELL, S. J. AND NORVIG, P. (1995). *Artificial Intelligence: A Modern Approach*. Prentice Hall. 19

SASTRY, S. AND BODSON, M. (1989-1994). *Adaptive Control: Stability, Convergence, and Robustness*. Prentice-Hall. URL http://www.ece.utah.edu/%7Ebodson/acscr/. 47

SCHRECKENBERG, M. AND WOLF, D. E., Eds. (1998). *Traffic and Granular Flow '97*. Springer, Singapore. 64

SCHWEITZER, F., Ed. (1997). *Self-Organization of Complex Structures: From Individual to Collective Dynamics*. Gordon and Breach. URL http://tinyurl.com/vbxyp. 24

SCHWEITZER, F. (2003). *Brownian Agents and Active Particles. Collective Dynamics in the Natural and Social Sciences*. Springer Series in Synergetics. Springer, Berlin. 20, 39, 53

SHALIZI, C. R. (2001). Causal architecture, complexity and self-organization in time series and cellular automata. Ph.D. thesis, University of Wisconsin at Madison. URL http://tinyurl.com/v3lho. 24, 31, 32, 46

SHEIKHOLESLAM, S. AND DESOER, C. A. (1991). Combined longitudinal and lateral control of a platoon of vehicles: A system level study. Tech. Rep. 1991-09-01, California PATH. URL http://tinyurl.com/yyo54y. 89

SIMON, H. A. (1964). On the concept of organizational goal. *Administrative Science Quarterly* **9** (1). 100

SIMON, H. A. (1976). *Administrative behavior: a study of decision-making processes in administrative organization*, 3rd ed. Free Press. 106

SIMON, H. A. (1982). *Models of bounded rationality*. MIT Press. 99

SIMON, H. A. (1996). *The Sciences of the Artificial*, 3rd ed. MIT Press. 12, 21, 35, 45, 88, 109

SIMS, A. G. (1979). SCATS: The Sydney co-ordinated adaptive system. In *Proceeding of the Engineering Foundation Conference on Research Priorities in Computer Control of Urban Traffic Systems*. 90, 93

SKÅR, J. AND COVENEY, P. V., Eds. (2003). *Self-Organization: The Quest for the Origin and Evolution of Structure*. Phil. Trans. R. Soc. Lond. A 361(1807). Proceedings of the 2002 Nobel Symposium on self-organization. 24

SOTL. (2004-2005). URL http://tinyurl.com/t7qbn. 3, 67, 72

STAHL, G. (2006). *Group Cognition*. MIT Press. 106

STEELS, L. (1993). Building agents out of autonomous behavior systems. In *The Artificial Life Route to Artificial Intelligence: Building Embodied Situated Agents*, L. Steels and R. A. Brooks, (Eds.). Lawrence Erlbaum. 60, 61

STEELS, L. (1998). Synthesising the origins of language and meaning using co-evolution, self-organisation and level formation. In *Approaches to the Evolution of Language*, J. R. Hurford, M. Studdert-Kennedy, and C. Knight, (Eds.). Cambridge University Press, pp. 384–404. 51, 132

STEELS, L. (2003). Evolving grounded communication for robots. *Trends in Cognitive Science* **7** (7): 308–312. URL http://tinyurl.com/y2u735. 24, 132

STERMAN, J. D. (2000). *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Irwin/McGraw-Hill. 19

STROGATZ, S. H. (2001). Exploring complex networks. *Nature* **410**: 268–276. URL http://tinyurl.com/waaeu. 108

TEN HAAF, W., BIKKER, H., AND ADRIAANSE, D. J. (2002). *Fundamentals of Business Engineering and Management, A Systems Approach to People and Organisations*. Delft University Press. 27, 41

THERAULAZ, G. AND BONABEAU, E. (1999). A brief history of stimergy. *Artificial Life* **5** (2) (Spring): 97–116. URL http://dx.doi.org/10.1162/106454699568700. 70, 89, 106

TREIBER, M. AND HELBING, D. (2001). Microsimulations of freeway traffic including control measures. *Automatisierungstechnik* **49**: 478–484. URL http://arxiv.org/abs/cond-mat/0210096. 143

TURBAN, E. AND ARONSON, J. (1997). *Decision Support Systems and Intelligent Systems*, 5th ed. Prentice Hall. 106

TURCHIN, V. (1977). *The Phenomenon of Science. A Cybernetic Approach to Human Evolution*. Columbia University Press, New York. URL http://pcp.vub.ac.be/POSBOOK.html. 43

VALCKENAERS, P., VAN BRUSSEL, H., HADELI, BOCHMANN, O., SAINT GERMAIN, B., AND ZAMFIRESCU, C. (2003). On the design of emergent systems: An investigation of integration and interoperability issues. *Engineering Applications of Artificial Intelligence* **16** (4): 377–393. URL http://dx.doi.org/10.1016/S0952-1976(03)00080-0. 55, 60

VAN ZANDT, T. (1998). Organizations that process information with an endogenous number of agents. In *Organizations with Incomplete Information*, M. Majumdar, (Ed.). Cambridge University Press, Cambridge, Chapter 7, pp. 239–305. 100, 101

VAZ, N. M. AND VARELA, F. J. (1978). Self and non-sense: An organism-centered approach to immunology. *Medical Hypothesis* **4** (3): 231–267. URL http://dx.doi.org/10.1016/0306-9877(78)90005-1. 40

VINCENT, R. A. AND YOUNG, C. P. (1986). Self optimising traffic signal control using microprocessors - the TRRL MOVA strategy for isolated intersections. *Traffic Engineering and Control* **27** (7-8) (July/August): 385–387. 69

VON FOERSTER, H. (1960). On self-organizing systems and their environments. In *Self-Organizing Systems*, M. C. Yovitts and S. Cameron, (Eds.). Pergamon, New York, pp. 31–50. 24

VON GLASERSFELD, E. (1984). An introduction to radical constructivism. In *The Invented Reality*, P. Watzlawick, (Ed.). Norton, New York. URL http://tinyurl.com/2fadgg. 144

VON NEUMANN, J. (1956). Probabilistic logics and the synthesis of reliable organisms from unreliable components. In *Automata Studies*, C. Shannon and J. McCarthy, (Eds.). Princeton University Press, Princeton. 21, 35

VON NEUMANN, J. (1966). *The Theory of Self-Reproducing Automata*. University of Illinois Press. Edited by A. W. Burks. 13, 52

WAGNER, A. (2004). Distributed robustness versus redundancy as causes of mutational robustness. Tech. Rep. 04-06-018, Santa Fe Institute. URL http://tinyurl.com/yx57xj. 21, 35, 52

WAGNER, A. (2005). *Robustness and Evolvability in Living Systems*. Princeton University Press, Princeton, NJ. URL http://www.pupress.princeton.edu/titles/8002.html. 21, 35, 170

WATSON, R. A. (2002). Compositional evolution: Interdisciplinary investigations in evolvability, modularity, and symbiosis. Ph.D. thesis, Brandeis University. 21, 35, 45

WATTS, D. J. AND STROGATZ, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature* **393**: 440–442. URL http://dx.doi.org/10.1038/30918. 109

WEBER, M. (1968). *Economy and Society*. University of California Press, Berkeley. 98

WEICK, K. E. AND ROBERTS, K. H. (1993). Collective mind in organizations: Heedful interrelating on flight decks. *Administrative Science Quarterly* **38** (3) (September): 357–381. 99

WEISER, M. (1997). Some computer science problems in ubiquitous computing. *Communications of the ACM*. 129

WERFEL, J. AND NAGPAL, R. (2006). Extended stigmergy in collective construction. *IEEE Intelligent Systems* **21** (2): 20–28. URL http://hebb.mit.edu/people/jkwerfel/ieeeis06.pdf. 106

WIENER, N. (1948). *Cybernetics; or, Control and Communication in the Animal and the Machine.* Wiley and Sons, New York. 2, 47

WIERING, M., VREEKEN, J., VEENEN, J. V., AND KOOPMAN, A. (2004). Simulation and optimization of traffic in a city. In *IEEE Intelligent Vehicles Symposium (IV'04)*. IEEE, pp. 453–458. URL http://tinyurl.com/yfdrrd. 66, 84, 93

WIESMAN, F., ROOS, N., AND VOGT, P. (2002). Automatic ontology mapping for agent communication. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*. ACM Press, New York, NY, USA, pp. 563–564. URL http://dx.doi.org/10.1145/544862.544876. 132

WILENSKY, U. (1999). NetLogo. URL http://ccl.northwestern.edu/netlogo. 66

WILENSKY, U. AND STROUP, W. (2002). NetLogo HubNet Gridlock model. URL http://tinyurl.com/y8624f. 66

WILSON, D. S. (2003). *Darwin's Cathedral: Evolution, Religion, and the Nature of Society*. The University of Chicago Press. 50

WITTGENSTEIN, L. (1999). *Philosophical Investigations*, 3rd ed. Prentice Hall. 130

WOLF, D. E., SCHRECKENBERG, M., AND BACHEM, A., Eds. (1996). *Traffic and Granular Flow '95*. World Scientific, Singapore. 64

WOLFRAM, S. (1986). *Theory and Application of Cellular Automata*. World Scientific. 13, 59

WOOLDRIDGE, M. (2002). *An Introduction to MultiAgent Systems*. John Wiley and Sons, Chichester, England. URL http://tinyurl.com/y2apyt. 20, 39

WOOLDRIDGE, M. AND JENNINGS, N. R. . (1995). Intelligent agents: Theory and practice. *The Knowledge Engineering Review* **10** (2): 115–152. URL http://tinyurl.com/y2bkbj. 39

WOOLDRIDGE, M., JENNINGS, N. R., AND KINNY, D. (2000). The Gaia methodology for agent-oriented analysis and design. *Journal of Autonomous Agents and Multi-Agent Systems* **3** (3): 285–312. URL http://tinyurl.com/yxkknw. 38

WUENSCHE, A. (1998). Discrete dynamical networks and their attractor basins. In *Complex Systems '98*, R. Standish, B. Henry, S. Watt, R. Marks, R. Stocker, D. Green, S. Keen, and T. Bossomaier, (Eds.). University of New South Wales, Sydney, Australia, pp. 3–21. URL http://tinyurl.com/y6xh35. 112

ZAMBONELLI, F., JENNINGS, N. R., AND WOOLDRIDGE, M. (2003). Developing multiagent systems: The Gaia methodology. *ACM Trans on Software Engineering and Methodology* **12** (3): 317–370. URL http://tinyurl.com/y5yh44. 38

ZAMBONELLI, F. AND RANA, O. F. (2005). Self-organization in distributed systems engineering: Introduction to the special issue. *Systems, Man and Cybernetics, Part A, IEEE Transactions on* **35** (3) (May): 313–315. URL http://dx.doi.org/10.1109/TSMCA.2006.846372. 38

GLOSSARY

abs-being Absolute being, independent of the observer. The modeled, 15

adaptation A change in an agent or system as a response to a state of its environment that will help the agent or system to fulfill its goals, 19

agent An agent is a description of an entity that *acts* on its environment, 39

anticipation A change in an agent or system as a response to a *future* state of its environment that will help the agent or system to fulfil its goals, 20

behavior A description an observer makes of the changes in a system with respect to an environment with which the system interacts (**?**, p. 163), 19

complexity The complexity of a system scales with the number of its elements, the number of interactions between them, the complexities of the elements, and the complexities of the interactions (Gershenson, 2002b), 13

context Set of circumstances and conditions which surround and determine an idea, theory, proposition, or concept. These circumstances and conditions can be spatial, temporal, situational, personal, social, cultural, ecological, etc. (Gershenson, 2002c), 144

control Regulation, steering, or ruling of a system, 2

design Creation or planning of a system, 2

emergent properties The properties of a system that are not present at the lower level but are a product of the interactions of elements, 12

environment Set of external variables, elements, agents, or systems that *interact* with an element, agent, or system. Note that every element, agent, or system may have a different environment, 21

friction Measure describing how one agent or system decreases the *satisfaction* of other agents or systems, 41

goal Purpose of an agent or system. In engineered systems, goals are determined by the designer, 39

hierarchy A ranking organization where one agent subordinates other agents, 108

information Anything that an agent can perceive or sense, 25

interaction A relation between two or more variables, elements, systems, or agents, 2

mediator A mediator arbitrates among the elements of a system, to minimize conflict, interferences and frictions; and to maximize cooperation and synergy, 41

model An abstract representation of a phenomenon, made by an observer within a specific context, 10

non-stationary problem domain It is given when the phase space of a dynamical system or the state space of a discrete dynamical system *does* change in time, 2

rel-being Relative being, dependent on observers and contexts. The model, 15

robustness A system is robust if it continues to function in the face of perturbations (Wagner, 2005), 21

satisfaction Measure describing the degree to which the *goals* of an agent have been reached, 39

self-organizing system A system *described* as self-organizing is one in which elements *interact* in order to achieve *dynamically* a global function or behavior (Gershenson, 2006a), 32

stationary problem domain It is given when the phase space of a dynamical system or the state space of a discrete dynamical system does *not* change in time, 2

synergy Measure describing how one agent or system increases the *satisfaction* of other agents or systems. Negative friction., 41

system A collection of interacting elements (components, parts). Note that elements can be systems themselves, 2

INDEX

“Als ik kan”
—Jan Van Eyck