Exploiting Non-Dominance in Multi Agent Systems: An Artificial Immune Algorithm for Distributed and Complex Problem Solving Environments

Author/Contributor:
Efatmaneshnik, Mahmoud; Reidsema, Carl

Event details:
12th Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES08)
Melbourne, Australia

Publication Date:
2008

DOI:
https://doi.org/10.26190/unsworks/432

Exploiting Non-Dominance in Multi Agent Systems: An Artificial Immune Algorithm for Distributed and Complex Problem Solving Environments

Mahmoud Efatmaneshnik[1] and Carl Reidsema[1]

[1] School of Mechanical and Manufacturing Engineering,
The University of New South Wales
Sydney NSW 2052
Australia
Emails: mahmoud@student.unsw.edu.au
Reidsema@unsw.edu.au

Abstract: An ideal Multi Agent System is flat and has no dominant hierarchy. Multi agent computational and problem solving environments have been advocated for their ability to deliver overall solutions that are innovative and creative. There is however a significant threat to the coherence of Multi Agent Systems; chaos. This paper poses a new vision to the control and immunisation of the Multi Agent Systems against chaos. Employing a complexity measure of the problem and its lower and upper bounds, and monitoring the complexity of the problem solving agents' interactions, we propose the holistic control of the Multi Agent Systems that leads to immunisation of the system against chaos. The control however is not central and appears in the form of the agents' common knowledge and determines their tendency to proactively communicate.

Keywords: Multi Agent Systems, Immunity, Chaos, Complexity Measure, Parametric Problem Solving, Innovation

1. Introduction

It is acknowledged that the creativity in product development organisation is dependent on the amount of information exchange and communication between the design groups, individuals, and in general design agents [5] [24] [36]. It is also well studied that the successful completion of a complex product design project requires a substantial amount of creativity in order to integrate many subsystems of a complex product [2]. The integration is often the most critical part when dealing with complex systems. Top level supervision and intervention in the problem solving process is regarded as a bottleneck of information that deteriorates the ability of a problem solving system to deliver creativity and innovation under uncertain conditions [18] [33]. Non-dominance is a desirable and ideal attribute of systems that deal with complex problems and means flat and organic organisations and computational environments.

Multi Agent Systems are distributed systems that use the bottom up approach to problem solving, in which case the intervention of the centralized coordination between agents is minimal or totally eliminated. Each agent in a Multi Agent System behaves as an abstraction tool which has the characteristics of a self-contained problem solving system that is capable of autonomous, reactive, proactive as well as interactive behavior [30]. The solution in this case emerges as a whole and is the result of the synergetic effects. Synergy denotes a level of group performance that is above and beyond what could be achieved by the members of the group working independently [21]. Synergy in a Multi Agent System enables the integration of partial solutions of nonlinear and coupled problems.

One important threat to the flat distributed problem solving environments with no dominant centralized authority is chaos [38], which is the opposite of coherence. Coherence is a global property of the Multi Agent Systems that could be measured by the efficiency, quality, and consistency of a global solution as well as the ability of the system to degrade gracefully in the presence of local failures [38]. Coherency is about the ability of the Multi Agent System to cope with solutions integration. Several methods for increasing coherence have been studied, all of which relate to the individual agent's ability to reason about the following questions: who should I interact with? And when should I do it and why? Sophisticated individual agent reasoning can increase Multi Agent System coherence because each individual agent can reason about non-local effects of local actions, form expectations of the behavior of others, or explain and possibly repair conflicts and harmful interactions [38]. These issues are the focus of this paper.

Hierarchical architectures consist of semi-autonomous agents with a global control agent dictating goals/plans or actions to the other agents. In these systems control can be implemented in different ways: using a special control expert called a supervisor as in EXPORT [27], or a shared database as in SHARED [41]; or through multiple shared workspaces as in MATE [31]. In theory, a truly open Multi Agent System need not have any predefined global control. DIDE (Distributed Intelligent Design Environment) [32] and ANARCHY [29] are rare examples of such architectures. DIDE was based on cognitive agents and a message handling service called tool-talk server. Shen and Barthès [32] didn't present particular strategies for conflict resolution between

agents. ANARCHY was a working prototype of an asynchronous design environment and used a global design strategy based on simulated annealing [29]. Agents in ANARCHY were autonomous, and could have broadcast type communications.

An immune algorithm is a plan that determines how the components of the systems are going to interact to determine the system dynamics [39]. Dasgupta [8] examined various response and recognition mechanisms of immune system and suggested their usefulness in development of massively parallel adaptive decision support systems. Lau and Wong [22] presented a multi agent system that could imitate the properties and mechanisms of the human immune system. The agents in this artificial immune system could manipulate their capabilities to determine the appropriate response to various problems. Through this response manipulation, a non-deterministic and fully distributed system with agents that were able to adapt and accommodate to dynamic environment by independent decision-making and inter-agent communication was achieved [22]. Ghanea-Hercock [15] maintained a multi agent simulation model that could demonstrate self organizing group formation capability and collective immune response. He showed that the network of agents could survive in the face of continuous perturbations. Fyfe and Jain [13] presented a multi agent environment in which the agents could manipulate their intensions by using concepts suggested by artificial immune system to dynamically respond to challenges posed by the environment. Goel and Gangolly [16] presented a decision support for robust distributed systems security based on biological and immunological mechanism.

This paper proposes a computational algorithm that allows for top level managerial knowledge to be present and meaningful to the low level agents and can be incorporated into a decision support system. This knowledge is based on the matching between the complexity measure of the problem and that induced by cooperation and collaboration between agents at any instance of the design process. Exploiting this algorithm in a decision support system can ensure the coherency of the global behavior of the system agents and their immunity from chaos. This way this algorithm is an artificial immune algorithm or otherwise it might be considered as a class of artificial life. It should be noted that this algorithm is at conceptual level and has not yet been implemented.

2. Distributed Problem Solving

Chen and Li [4] referred to Concurrent Product Design taking place in the parametric design stage as Concurrent Parametric Design. Concurrent Parametric Design models a problem as sets of variables. These include the set of design variables (or inputs) and the set of design objectives (or outputs). The design variable sets include subsets of sizing variables, shape variables, topologies, configurations, and manufacturing variables such as process capabilities [28]. Each variable may be accompanied by a set of constraints. Design problem solving is the process of assigning values to these variables in accordance with the given design requirements, constraints, and optimisation criterion [42]. A design task in this view constitutes the determination of a single design variable, determining the value of which is the task of a design agent. These types of agents are known as single function agents [10]. An agent is a design participant that can be, in a broad sense, a human designer, computer or an algorithm, that is able to cope with distributed tasks as part of the whole design problem. In this design situation, agents may face uncertainties during the design process, especially when their design decisions are interrelated. Resolving this uncertainty is usually the task of mediators, facilitators or coordinators. Leenders et al. [24] showed that design system's creative performance will be negatively related to the presence of central supervisors (including brokers, mediators and facilitators) in the intra-team communication network. We argue that the mediators' role can be omitted if the low level knowledge of the problem is present at the low level design agents. We will elaborate on this further in section 3.2. However, in order to produce this knowledge one needs to resort to simulation based techniques.

2.1 A Bottom up Approach

In general in a problem solving environment the agents' actions can be planned or controlled by using three kinds of knowledge. The low level problem knowledge, the medium level knowledge of the problem solving process, and the high level organisational knowledge including high level goals and strategies. In product design the performance and operational requirements of the product are micro or low level parameters, whereas production costs, times and risks are macro or high level (and often emergent) properties of process. A system exhibits emergence when there are coherent properties at the macro-level (i.e. of the system as a whole) that dynamically arise from the interactions between the parts at the micro-level [40]. Emergent properties are meaningless and irrelevant at the local level. Many have argued that for complex problems bottom up approach must be used. This means that the local behavior of agents needs to lead to global emergent solutions. The question is which rules should the agents use? And, more importantly, what kind of knowledge should the rules be derived from? Design planning usually takes place in a top–down fashion by considering the high level organisational knowledge, particularly the structure of the organisation. If planning starts at the top, such models rarely reach the lowest levels of design activity, where individual design variables are determined based on other variables [3]. Determining these variables are the lowest level design activities, and a bottom-up, integrative analysis of these low-level activities can provide process structure insights [3]. Thus, in order to resolve the uncertainty and dealing with high level emergent properties of the process and organisational operations (that can become chaotic), the low level knowledge of the problem must be used to characterize the behavioral rules of agents. This is indeed a bottom up approach and is elaborated here.

The design structure matrix (DSM) is a well known knowledge representation and analysis tool for system modeling. A DSM displays the relationships between

components of a system in a compact, visual, and analytically advantageous format as a square matrix with identical row and column labels. DSMs are usually employed in modelling products, processes, and organisational architectures. Browning [3] argued that the three DSMs and the structures they model are tightly related, and in many real industrial cases they exhibit strong couplings. He presented the following definitions:

(i) Parameter-Based (or Low-Level Schedule) DSM**:** Used for modeling low-level relationships between design decisions and parameters, systems of equations, subroutine parameter exchanges which represents the product architecture.

(ii) Activity-Based or Schedule DSM: Used for modeling processes and activity networks based on activities and their information flow and other dependencies.

(iii) Organizational DSM: Used for modeling organization structures based on people and/or groups and their interactions.

Browning [3] emphasized that clearly, parameter-based DSMs have integrative applications. This characteristic of the parameter based DSM which represents the low level product knowledge makes it suitable to be utilized in the planning or determining the rule of behavior for low level agents involved in the engineering design of complex systems. This matrix is referred to as the self of the problem with the values of variables representing the non self [10].

2.3 Simulation Based Engineering

In order to have a priori knowledge of the problem at the upstream of the design process and early phases, simulation based techniques must be used. Simulation is the key to reconcile ambitious performance and operational requirements improvement with realistic development and production costs, times and risks for highly innovative industrial high-tech systems [12]. As creating high-fidelity simulation models are a complex activity that can be quite time-consuming [34], the Monte Carlo Simulation is suggested to establish the fitness landscape of the design problem [26]. A fitness landscape is a multi-dimensional data set, in which the number of dimensions is determined by the number of system variables. Every design variable is regarded as a random variable. Marczyk [25] has stressed that by means of Monte Carlo Simulation of design variables the fitness landscape of the design space is created enabling the verification of the global dependencies between low level design variables. We suggest the actualization of multidisciplinary parameter based DSM through Monte Carlo and Statistical Simulation in the early stage of the design process. In order to establish the correlation coefficients between different variables, global entropy based correlation coefficients have significant advantage over linear covariance based correlation coefficients. Entropy based correlations can capture both linear and nonlinear dependencies [10]. Table 1 show an example of a typical simulated parameter based DSM with normalized weights (all the weights are between zero and one).

Table 1 The parameter based DSM of a design problem with 10 variables

-	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
V1	0	0.76	0.45	0.16	0.22	0.77	0.12	0.01	0	0
V2	0.76	0	0.11	0.65	0.44	0.78	0	0	0	0.18
V3	0.45	0.11	0	0.64	0.11	0.31	0.02	0	0.15	0
V4	0.16	0.65	0.64	0	0.45	0.34	0	0	0	0
V5	0.22	0.44	0.11	0.45	0	0	0	0.01	0	0.01
V6	0.77	0.78	0.31	0.34	0	0	0	0	0	0
V7	0.12	0	0.02	0	0	0	0	0.2	0.7	0.1
V8	0.01	0	0	0	0.01	0	0.2	0	0.2	0.8
V9	0	0	0.15	0	0	0	0.7	0.2	0	0.9
V10	0	0.18	0	0	0.01	0	0.1	0.8	0.9	0

The outcome of a Monte Carlo Simulation can be fed into the Ontospace^{TM}[1] software that, in addition to the self map of the system, delivers the complexity of the map and its bounds.

2.4 A Complexity Measure of the Problem and its Bounds

Marczyk and Deshpande [26] stated that complexity is frequently confused with emergence; emergence of new structures and forms is the result of re-combination and spontaneous self-organisation of simpler systems to form higher order hierarchies, i.e. a result of complexity. We define complexity as the intensity of emergence in a system. If the complexity is too high the system becomes chaotic and uncontrollable and is likely to lose its structure. If the complexity is too low the system loses the intrinsic characteristics of the entity it was intended to describe, and fails to emerge as a spontaneous organization. Complexity materializes the system's self by the emergence of the self structure when the system's elements have sufficient interactions. Complexity is a "holistic" measure of the system that enables us to study the system as a "whole". Marczyk and Deshpande [26] proposed a comprehensive complexity metric that is embedded in Ontospace^{TM} software. The metric takes into account all significant aspects necessary for a sound and comprehensive complexity measure, namely structure, entropy and data granularity, or coarse-graining [26]. The metric allows one to relate complexity to fragility and to show how critical threshold complexity levels may be established for a given system. This software calculates three complexity measures for every self map (fig. 1):

(i) The complexity of the map which is a very specific measure reflecting the coupling, and size of the system. We will refer to the complexity of this map as self complexity.

(ii) The upper complexity bound to which the complexity of the system may be increased without exhibiting chaos.

[1] This is a first of its kind complexity management tool based on measure of complexity. See www.ontonix.com.

(iii) The lower complexity bound which the system with a complexity lower than that has lost its intrinsic characteristics and has failed to emerge as a spontaneous self.

Due to nondisclosure agreement with Ontonix s.r.l. and commercialization of Ontospace™ software the details of these measures are not revealed in this paper.

Dembski [9] explained that, a system performing a given basic function is irreducibly complex if it includes a set of well-matched, mutually interacting, non-arbitrarily individuated parts such that each part in the set is indispensable to maintaining the system's basic, and therefore original, function. The set of these indispensable parts is known as the irreducible core of the system. The lower complexity bound represents the irreducible complexity of the system that contains the intrinsic characteristics of the system.

There is a sufficient body of knowledge to sustain the belief that whenever dynamical systems undergo a catastrophe, the event is accompanied by a sudden jump in complexity [26]. This is also intuitive: a catastrophe implies loss of functionality, or organisation. The increase of entropy increases complexity (entropy is not necessarily adverse as it can help to increase fitness) but at a certain point, complexity reaches a peak beyond which even small increase of entropy inexorably cause the breakdown of structure [26]. After structural breakdown commences, an increase in entropy nearly always leads to loss of complexity (fitness) [26]. However, beyond the critical point, loss is inevitable, regardless of the dimensionality and/or density of the system. Therefore every closed system can only evolve/grow to a specific maximum value of complexity. This is known as the system's critical maximum complexity. Close to criticality, systems become vulnerable, fragile and difficult to manage. The difference between the current and critical complexity is a measure of the overall health of the system. The closer to criticality a system is, the less healthy and therefore more risky it becomes.

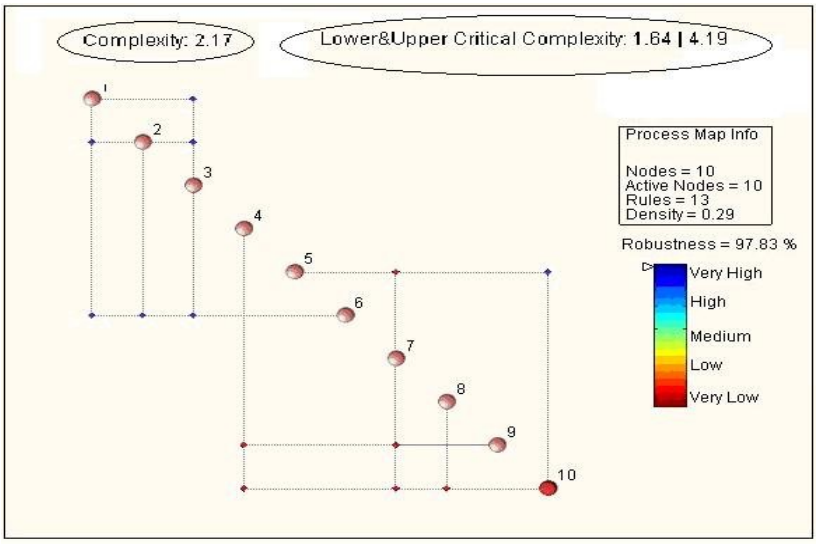

Fig. 1. The self map of the DSM in Table 1 and its three complexity measures calculated by Ontospace™.

2.5 The Need for Radical Innovation

Integrated product development describes how tasks are interconnected and seeks to integrate the product process and organization (the network of the tasks) but does not provide adequate problem solving methodologies [28]. Choosing an integration scheme is critical in determining how efficient or how flexible the resulting problem solving architecture will turn out to be [28]. The product/problem architecture and organization structure relationship can affect an enterprise in several dimensions, including architectural innovation [3]. The product architecture has a large influence on the appropriate structure of the product development organization since organizational elements are typically assigned to develop various product components [3]. To incorporate this knowledge into the product development system, the organizational DSM can be derived directly from the simulated parameter based DSM. The requisite for this is that the organization does not have a predefined structure. A direct mapping can be used to force the organization structure to mirror the product architecture. This allows for predefined communication and information exchange channels, in a large and complex environment. It will also eliminate or reduce the threat of chaos but such system would be utterly ineffective in delivering radical innovation [17]. Creativity requires ad hoc communication in which the need to communicate often arises in an unplanned fashion, and is affected by the autonomy of the agents to develop their own communication patterns [24]. It is thus obvious that, a fixed organizational structure with established patterns of communication is not capable of delivering new complex structures (products).

Henderson and Clark [17] demonstrated that there are different kinds of innovation as depicted in fig. 2 where

innovation is classified along two dimensions. The horizontal dimension captures an innovation's impact on product's components, while the vertical dimension captures its impact on the linkages between core concepts and components. Architectural innovation changes only the relationships between modules but leaves the components, and the core design concepts that they embody, unchanged. Incremental innovation refines and extends an established design. Improvement occurs in individual components, but the underlying core design concepts, and the links between them, remain the same. Modular innovation on the other hand, changes only the core design concepts without changing the product's architecture. It can be said that radical innovation embodies both modular and architectural innovation. Radical innovation establishes a new dominant design and, hence, a new set of core design concepts that are linked together in a new architecture.

	Reinforced	Overturned
Unchanged	Incremental Innovation	Modular Innovation
Changed	Architectural Innovation	Radical Innovation

Fig. 2. Different types of innovation, from [17].

An organization's communication channels, both formal and informal are critical to achieving radical and architectural innovation [17]. The communication channels that are created between design agents will reflect the organization's knowledge of the critical interactions between product modules. Organization's communication channels will embody its architectural knowledge of the linkages between components that are critical to effective design [17]. They are the relationships around which the organization builds architectural knowledge.

Innovation processes in complex products and systems differ from those commonly found in mass produced goods [19]. The creation of complex products and systems often involves radical innovation [2], not only because they embody a wide variety of distinctive components and subsystems (modular innovation), skills and knowledge inputs but also because large numbers of different organizational units have to work together in a collaborative manner (architectural innovation). Here, the key capabilities are systems design, project management, systems engineering and integration [19]. Integration in complex system and product design is to make the solutions to sub problem compatible with each other and is possible through innovation [2]. The innovation that integrates the complex system must be radical innovation accompanied by and creativity that is an emergent property of the entire system rather than the property of the sub-solutions to the individual sub problems [36], [35]. A property that is only implicit, i.e. not represented explicitly, is said to be an emergent property if it can be made explicit and it is considered to play an important role in the introduction of new schemas [14]. The radical innovation and coherency in an engineered large scale system is emergent and obtained in a self organizing fashion in a multi agent environment. When designing self-organizing emergent Multi Agent Systems with emergent properties, a fundamental engineering issue is to achieve a macroscopic behavior that meets the requirements and emerges only from the behavior of locally interacting agents. Agent-oriented methodologies today are mainly focused on engineering the microscopic issues, i.e. the agents, their rules, how they interact, etc, without explicit support for engineering the required macroscopic behavior. As a consequence, the macroscopic behavior is achieved in an ad-hoc manner [41]. 'Emerging' properties and innovative organizational structures are required to coordinate between different designs agent actions.

3 Holistic Process Monitoring

When the inherent nature of a complex task is too large a better solution is to create an environment in which continuous innovation can occur [2]. This can be accomplished through process monitoring; Bayrak and Tanik [1] reported that improving the design process increases the product quality without increasing the design resources, and is possible by providing feedback to the designer to help him/her understand the nature of the design process. Therefore, the nature of the design becomes easier to analyze if there are metrics obtained from activity monitoring [1]. Since the design process of the complex systems by concurrent engineering is an emergent process [6], holistic metrics are required to monitor the design process. One such metric is the cognitive complexity of a process that is defined as the ability of a problem solver to flexibly adapt to a multidimensional problem space [23]. Cognitive complexity represents the degree to which a potentially multidimensional cognitive space is differentiated and integrated. A problem solver (a person, organization or a Multi Agent System) with higher cognitive complexity is more capable of having creative (and holistically correct) outcomes [23].

We suggest measuring the cognitive complexity of a Multi Agent Design System as a function of the information exchange between the design agents. The main complication here is the way in which the information exchange is measured. Kan and Gero [20] suggested the use of entropy based measures for evaluation of information content of a design agent's interactions. We suggest using a fuzzy method by simply asking the design participants to tag qualitative and quantitative information content of their interactions with a single fuzzy variable, e.g. high, low, and medium, etc. These can then be defuzzified, which is the process of producing a quantifiable result in fuzzy logic, according to a simple fuzzy rule, a simple example of which is shown in fig 3. Note that the information mentioned here is both qualitative and quantitative

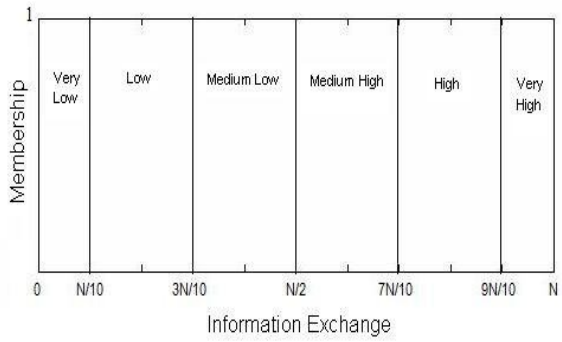

Fig. 3. A simple fuzzification scheme. N is the maximum amount of information exchange in parameter based DSM.

A main point is that the inclination of the design agents for more collaboration by the means of information exchange does not necessarily lead to more overall cognitive complexity of the design system. This is to say unnecessary information exchange may lower the overall cognitive complexity. By testing a sample of 44 new product development organizations, Leenders et al. [24] showed that the performances of innovation networks have an inversely U-shape relationship to frequency of cooperation. One can, thus, conclude that cognitive complexity of the whole design system must be upper and lower bounded for having effective and efficient innovation networks. Chiva-Gomez [5] also favoured balanced participation of design players in design decision making process, against increasing information flow between the design players to a maximum.

Bar-Yam [2] has stated that in order to solve a problem, the problem solver needs to have (cognitive) complexity more than or equal to the problem complexity, which is a version of the Ashby's law of requisite variety [2]. This is to say the cognitive complexity of the process must be equal or more than the minimum (irreducible) complexity of the problem.

$$C_C \geq C_{min} \quad (1)$$

It is obvious that the cognitive complexity of the process needs not be more than the maximum complexity of the problem since more than required information exchange can lead to creativity blocks [24] which can be termed a chaotic situation. Thus an upper bound for the cognitive complexity of the process is the maximum complexity of the problem:

$$C_C \leq C_{max} \quad (2)$$

Innovation in a multi agent environment is the result of communication between social agents that happens in a self organizing fashion when the Multi Agent System finds itself on the so-called edge of chaos [37]. When the cognitive complexity of the process is in between the minimum and maximum complexity of the problem, the design system might be on the edge of chaos but certainly not chaotic. Besides for collaborative Multi Agent Systems with cognitive complexity less than the minimum complexity the design process is certainly away from the edge of chaos, thus the design systems does not have enough functionality to deliver radical innovation in an optimal and efficient manner. For systems that are excessively persistent in collaboration and cooperation the cognitive complexity may reduce and thus chaos appear. Chaos makes the design process fragile and susceptible to uncertainties (such as the indetermination of a design variable) and renders the design system ineffective.

Fig 4. shows that design system's overall cognitive complexity increase only to a certain threshold by the tendency of the design agents for exchanging design information. In order to ensure the health of the design process it is necessary to ensure that the overall cognitive complexity stays away from and below the maximum complexity and above the minimum. This way the minimum and maximum complexity that are obtained by using the initial Monte Carlo Simulation of the complex product (low level product knowledge) are used to monitor the efficiency and effectiveness and health of the complex product design process. This is the holistic monitoring of the design process. The process monitoring here serves the purposes of meeting the design objectives (quality, cost, and lead time) by immunizing the design system against chaos and lack of effectiveness. This immunization enables the design system to integrate the complex system and product through utilization of radical innovation. Note the holistic process monitoring does not violate the agents' autonomy.

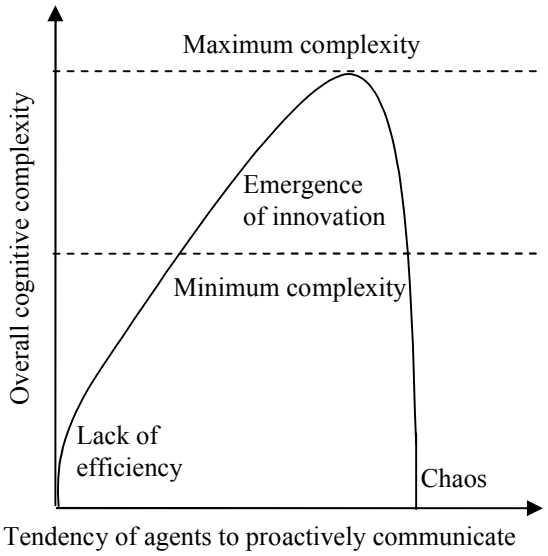

Fig 4. Design process functionality versus process complexity.

3.1 Artificial Immune Algorithm

According to Cohen [7] the immune system is a computational strategy to carry out the functions of protecting and maintaining the body. Cohen's maintenance

role of the immune system requires it to provide three properties:

(i) Recognition: to determine what is right and wrong.
(ii) Cognition: to interpret the input signals, evaluate them, and make decisions.
(iii) Action: to carry out the decisions.

These properties are provided via a cognitive strategy in which self-organization of the immune system is used to make decisions [39]. The stages correspond to the holistic control of the system, which is to immunize or ensure the realization of self-organization, by using a complexity measure:

(i) Recognition: recognizing the lower and upper complexity bounds.
(ii) Cognition: evaluating the instantaneous cognitive complexity of system of agents at every design instance.
(iii) Action: maintaining this complexity in between the bounds at all times.

Upon the finalized decision on the value of each design variable, and in general, upon the arrival of any new information in the design system, the simulation must be updated. This will allow for new estimation of the problem complexity and from that, the new estimation of the new collective cognitive complexity bounds. In this way the presented algorithm responds to changes that are made in the problem state. The immune algorithm is therefore evolutionary and adaptive.

3.2 Agents Structure

The single function agents [10] that can focus their attention at one design variable at a time need to be equipped with the right tools in order to be able to benefit from the immune algorithm. Single function agents perform the design task of determining the values of the design variables, one at a time. In artificial intelligence, an intelligent agent observes and acts upon an environment, as a rational agent: an entity that is capable of perception, action and goal directed behavior. The internal architecture of an agent is essentially the description of its modules and how they work together [33]: agent architectures in various agent based systems (including agent based concurrent design and manufacturing systems) range from the very simple (a single function control unit with a single input and output) to very complex human like models.

Four different agent architectures have been discussed in the literature: reactive agents (also known as behavior based or situated agent architectures), deliberative agents (also called cognitive agents, intentional agents, or goal directed agents), collaborative agents (also called social agents or interacting agents), and hybrid agents [33]. Reactive agents are passive in their interactions with other agents. They do not have an internal model of the world and respond solely to external stimuli. They respond to the present state of the environment in which they are situated. They do not take history into account or plan for the future [38]. Through simple interactions with other agents, complex global behavior can emerge. In reactive systems, the relationship between individual behaviors, environment, and overall behavior is not understandable. However, the advantage of reactive agent architecture is simplicity.

Deliberative agents use internal symbolic knowledge of the real world and environment to infer actions in the real world. They proactively interact with other agents based on their sets of Beliefs, Desires and Intentions. These agents perform sophisticated reasoning to understand the global effects of their local actions [38]. Consequently, they have difficulties when applied to large complex systems due to the potentially large symbolic knowledge representations required [38]. Shen et al. [33] identified collaborative agents as a distinct class of agents that work together to solve problems; communication in between them leads to synergetic cooperation, and emergent solutions.

Hybrid architectures are neither purely deliberative nor purely reactive [38]. Hybrid agents usually have three layers: at the lowest level in the hierarchy, there is typically a reactive layer, which makes decisions about what to do on the basis of raw sensor input. This layer contains the self knowledge that is the knowledge of the agent about itself including physical state, location, and skills, etc [33]. The middle layer, typically abstracts away from raw sensor input and deals with a knowledge-level view of the agent's environment, often making use of symbolic representations [38]. This layer contains two types of knowledge: domain knowledge and common sense knowledge. The domain knowledge is the description of the working projects (problems to be solved), partial states of the current problem, hypothesis developed and the intermediate results [33]. The uppermost level of the architecture handles the social aspects of the environment [38]. This layer contains the social knowledge and is in charge of coordination with other agents.

The hybrid agents are suitable for consuming the immune algorithm. The common sense knowledge in the middle layer of the agents should contain the status of the collective cognitive complexity of the system relative to the complexity bounds of the problem. This knowledge can be produced by a centralized data base such as a blackboard data base that stores the current status of the problem, and the map of the information exchanges taking place between agents at any time. On the basis of the common sense knowledge the agent decides whether to be more proactive (if the cognitive complexity of the agents system is less than the minimum complexity of the problem) or reactive (if the complexity is close to the maximum complexity). Fig 5. shows that the amount of pro-activity as the common sense knowledge of a hybrid agent should have inverse relation to the distance of instantaneous cognitive complexity from maximum complexity. It also should have direct relation to the distance of instantaneous cognitive complexity of the design system and the minimum complexity of the problem.

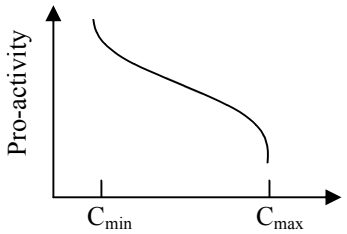

Fig. 5. The common sense knowledge of the agents.

3.3 An Example

Consider the problem with simulated parameter based DSM in Table 1. From this table and fig 1. the following prosperities can be clarified:

$$C_{min}=1.64 \quad (3)$$

$$C_{max}=4.19 \quad (4)$$

$$N=0.9 \quad (5)$$

Where N is the maximum amount of information exchange, C_{min} is the minimum complexity and C_{max} is the maximum complexity. Each of the variables in Table 1 is assigned to a design agent to determine its value. After a while the design agents are asked to report the amount of their information exchanges and interactions with each other. Consider Table 2 as the reported or observed fuzzy team based DSM at an instance of the design process.

Table 2 The fuzzy monitored (reported) team based DSM

-	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
A1	-	Low	-	Low	Low	High	Low	VL	-	-
A2	H	-	L	MH	ML	H	-	-	-	L
A3	ML	L	-	MH	L	ML	VL	-	L	-
A4	L	MH	MH	-	ML	L	-	-	-	-
A5	L	ML	L	ML	-	-	-	VL	-	VL
A6	H	H	ML	ML	-	-	-	-	-	-
A7	L	-	VL	-	-	-	-	VH	VL	L
A8	VL	-	-	-	VL	-	L	-	L	H
A9	-	-	L	-	-	-	H	L	-	H
A10	-	L	-	-	VL	-	L	H	H	-

Table 3 will be resulted by defuzzifying the observed team based DSM according to the fuzzy rule in fig 3.

Table 3 The defuzzified monitored team based DSM

-	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
A1	0	0.18	0	0.18	0.18	0.72	0.18	0.04	0	0
A2	0.72	0	0.18	0.56	0.36	0.72	0	0	0	0.18
A3	0.36	0.18	0	0.56	0.18	0.36	0.04	0	0.18	0
A4	0.18	0.56	0.56	0	0.36	0.18	0	0	0	0
A5	0.18	0.36	0.18	0.36	0	0	0	0.04	0	0.04
A6	0.72	0.72	0.36	0.36	0	0	0	0	0	0
A7	0.18	0	0.04	0	0	0	0	0.9	0.04	0.18
A8	0.04	0	0	0	0.04	0	0.18	0	0.18	0.72
A9	0	0	0.18	0	0	0	0.72	0.18	0	0.72
A10	0	0.18	0	0	0.04	0	0.18	0.72	0.72	0

Eq. (6) shows the complexity of the defuzzified team based DSM (Table 3) which is in between the minimum and maximum complexity bounds (1.64, and 4.19).

$$C_C=1.9191 \quad (6)$$

This information is sent to design agents. By using their common sense knowledge the agents will know that they are allowed to communicate more actively and increase the cognitive complexity of the system.

4. Conclusion

An immune algorithm for the control of Multi Agent Systems was presented (fig 6.). This algorithm is suitable for design of complex products such as commercial aircrafts, cars and other advanced multi technological products that require the cooperation of hundreds of thousands of designers. The algorithm allows for the emergence of innovation because it eliminates the need for centralized intervention in the problem solving process. The algorithm however yields control that takes place in a bottom up fashion: by simply changing the intentions of the agents for communication through the manipulation of their communication tendency. The algorithm employs low level knowledge of the problem and on this basis decides on the degree of agents' communications complexity. This complexity (known as cognitive complexity of the design system) would be announced to all agents regularly along with the lower and upper complexity bounds estimated from the problem map or parameter based DSM. The agents would then be obliged to adjust their communication tendency based on the distance of the instantaneous cognitive complexity of the design system and the measured minimum and maximum complexity. For emergence of innovation and also coherency in the design system the cognitive complexity must be in between the two bounds. If the cognitive complexity is more than the upper bound chaos may appear and the agents must reduce their communication tendency. If it is lower than the lower bound, there would not be enough functionality in the design system to cope with the problem and the agents must increase their communication tendency.

The paper did not give any notes on the implementation of the algorithm in a Decision Support Systems. However, a blackboard data base can be used to store the design variables, and determine their interdependency via simulation. Parts of the blackboard can be dedicated to agents' announcement of their communications and information exchanges.

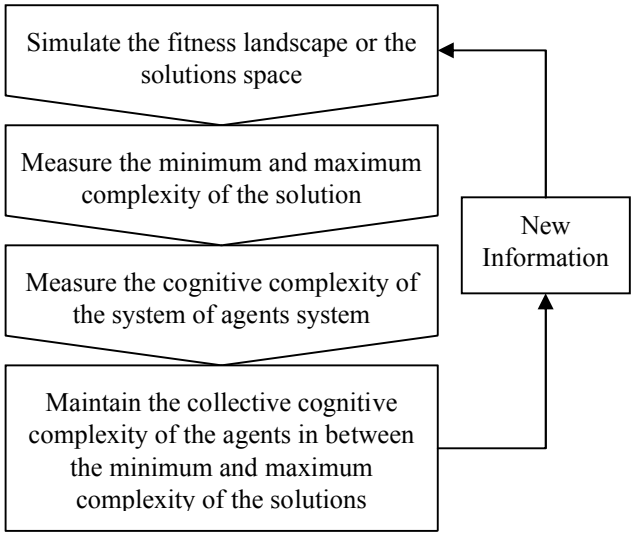

Fig. 6. An Immune algorithm for design of complex systems

References

[1] C Bayrak, MM Tanik, A Process Oriented Monitoring Framework. Systems Integration, Vol. 8, 1998, pp. 53-82

[2] Y Bar-Yam Making Things Work: Solving Complex Problems in a Complex World, NECSI Knowledge Press, 2004, pp. 90-91

[3] TR Browning, Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions. IEEE Transactions on Engineering Management, Vol. 48, 2001, pp. 292-306

[4] L Chen, S Li, Concurrent Parametric Design Using a Multifunctional Team Approach. In Design Engineering Technical Conferences DETC'01, Pittsburgh, Pennsylvania, 2001

[5] R Chiva-Gomez, Repercussions of complex adaptive systems on product design management. Technovation, Vol. 24, 2004, pp. 707-711

[6] A Cisse, S Ndiaye, J Link-Pezet, Process Oriented Cooperative Work: an Emergent Framework. In IEEE Symposium and Workshop on Engineering of Computer Based Systems, Friedrichshafen, Germany, 1996, pp. 342-347

[7] I Cohen, Real and Artificial Immune Systems: Computing the State of the Body. Imm Rev, Vol. 7, 2007, pp. 569-574

[8] D Dasgupta, An Artificial Immune System as a Multi-Agent Decision Support System. In Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC), vol. 4, 1998, pp. 3816-3820, San Diego, California

[9] W Dembski, No Free Lunch: Why Specified Complexity Cannot Be Purchased without Intelligence, Rowman & Littlefield Publishers, Inc., 2002

[10] B V Dunskus, Single Function Agents and Their Negotiation Behavior in Expert Systems. Report paper, Worcester Polytechnic Institute, Worcester, MA, 1994

[11] M Efatmaneshnik, C Reidsema, Immunity as a Design Decision Making Paradigm for Complex Systems: a Robustness Approach. Cybernetics and Systems, Vol. 38, No. 8, 2007, pp. 759-780

[12] A Formica, J Marczyk, Strategic Multiscale A New Frontier for R&D and Engineering, Ontonix, Turin, 2007, pp. 56

[13] C Fyfe, L Jain, Teams of intelligent agents which learn using artificial immune systems. Journal of Network and Computer Applications, vol. 29, 2006, pp.147–159

[14] J S Gero, Creativity, emergence and evolution in design. Knowledge-Based Systems, Vol. 9, 1996, pp. 435-448

[15] R Ghanea-Hercock, (2007) Survival in cyberspace. Information Security, vol. 12, 2007, pp. 200–208

[16] S Goel, J Gangolly, On decision support for distributed systems protection: A perspective based on the human immune response system and epidemiology International. Journal of Information Management, vol. 27, 2007, pp. 266–278

[17] R M Henderson, K B Clark, Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms. Administrative Science Quarterly, Vol. 35, 1990

[18] P Hinds and C McGarth, Structures that work: social structure, work structure and coordination ease in geographically distributed teams. In proceedings of the 20th anniversary conference on Computer supported cooperative work, Banff, Alberta, Canada, 2006, Nov 04-08: pp. 343-352

[19] M Hobday, H Rush, J Tidd, Innovation in complex products and system. Research Policy, Vol. 29, 2000, pp. 793-804

[20] J Kan, J Gero, Can entropy represent design richness in team designing? In proceedings of 10th International Conference on Computer Aided Architectural Design Research in Asia CAADRIA'05, Bhatt A (ed.), New Delhi, 2005, pp. 451-457

[21] J R Larson, Deep Diversity and Strong Synergy: Modeling the Impact of Variability in Members' Problem-Solving Strategies on Group Problem-Solving Performance. Small Group Research, Vol. 38, 2007, pp. 413-436

[22] H Lau, V Wong, Immunologic Responses Manipulation of AIS Agents. Lecture Notes in Computer Science, vol. 3239, 2004, pp. 65-79
[23] J Lee, D P Truex, Cognitive Complexity and Methodical Training: Enhancing or Suppressing Creativity. In proceedings of 33rd Hawaii International Conference on System Sciences, 2000
[24] R Leenders, J Kratzer, J Hollander et al., Virtuality, Communication, and New Product team Creativity: a Social Network Perspective. Engineering and Technology Management, Vol. 20, No. 1, 2003, pp. 69-92
[25] J Marczyk, Principles of Simulation Based Computer Aided Engineering. FIM Publications, Barcelona, 1999, pp. 64
[26] J Marczyk, B Deshpande, Measuring and Tracking Complexity in Science. In 6th International Conference on Complex Systems, Minai A, Braha D, Bar-Yam Y (eds.) Boston, MA, 2006
[27] E Monceyron, J-P Barthes, Architecture for ICAD Systems: an Example from Harbor Design. Sience et Techniques de la Conception, Vol. 1, 1992, pp. 49-68
[28] B Prasad, Concurrent Engineering Fundamentals, Volume II: Integrated Product Development, Prentice Hall, 1996
[29] R W Quadrel, R F Woodbury, S J Fenves et al., Controlling asynchronous team design environments by simulated annealing. Research in Engineering Design, Vol. 5, 1993, pp. 88-104
[30] C Reidsema, E Szczerbicki, A Blackboard Database Model of the Design Planning Process in Concurrent Engineering. Cybernetics and Systems, Vol. 32, 2001, pp. 755-774
[31] M Saad, M L Maher, Shared understanding in computer-supported collaborative design. Computer-Aided Design, Vol. 28, 1996, pp. 183-192
[32] W Shen, J-P Barthès, An Experimental Multi-Agent Environment for Engineering Design. International Journal of Cooperative Information Systems, Vol. 5, 1996, 131-151
[33] W Shen, D H Norrie, J-P Barthès, Multi-Agent Systems for Concurrent Intelligent Design and Manufacturing, CRC Press, 2001
[34] R Sinha, V C Liang, C J Paredis et al., Modeling and Simulation Methods for Design of Engineering Systems, Computing and Information Science in Engineering, Vol. 1, 2001, pp. 84-91
[35] R Sosa, J Gero, a Computational Study of Creativity in Design. AIEDAM, Vol. 19, 2005, pp. 229-244
[36] R Sosa, J Gero, Diffusion of Creative Design: Gate keeping Effects. International Journal of Architectural Computing, Vol. 2, 2004, pp. 518-531
[37] R D Stacey, the Science of Complexity: An Alternative Perspective for Strategic Change Processes. Strategic Management Journal, Vol. 16, 1995, pp. 477-495
[38] K Sycara, Multiagent Systems, In AI Magazine Vol. 19, No. 2, 1998, pp.79-93
[39] J Timmis, P Andrews, N Owens et al., an Interdisciplinary Perspective on Artificial Immune Systems. Evolutionary Intelligence, Vol. 1, 2008, pp. 5-26
[40] Wolf TD, Holvoet T () Towards a Methodology for Engineering Self-Organising Emergent Systems. In Self-Organization and Autonomic Informatics, Glasgow, UK, 2005, pp. 18-34
[41] A Wong, D Sriram, SHARED: An information model for cooperative product development. Research in Engineering Design, Vol. 5, 1993, pp. 21-39
[42] Zdrahal Z, Motta E, Case-Based Problem Solving Methods for Parametric Design Tasks. In Proceedings of the third workshop on Advances in Case-Based Reasoning, , Springer-Verlag, London, UK, 1996, pp. 473-486