UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

DIEGO VRAGUE NOBLE

The Impact of Social Context in Social Problem Solving

Thesis presented in partial fulfillment of the requirements for the degree of Master of Computer Science

Prof. Dr. Luís da Cunha Lamb
Advisor

Prof. Dr. Ricardo Matsumura Araújo
Coadvisor

Porto Alegre, April 2013

CIP – CATALOGING-IN-PUBLICATION

Noble, Diego Vrague

The Impact of Social Context in Social Problem Solving / Diego Vrague Noble. – Porto Alegre: PPGC da UFRGS, 2013.

80 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2013. Advisor: Luís da Cunha Lamb; Coadvisor: Ricardo Matsumura Araújo.

1. Computer Science; Artificial Intelligence; Social Computing. I. Lamb, Luís da Cunha. II. Araújo, Ricardo Matsumura. III. Título.

*"Essentially, all models are wrong,
but some are useful."*
— G. E. P. Box

*"The purpose of computing is insight,
not numbers."*
— R. W. Hamming

AGRADECIMENTOS

Agradeço profundamente aos meus orientadores, os professores Luís Lamb e Ricardo Araújo, pela oportunidade de realizar o mestrado. Aos meus pais, agradeço por tudo que me proporcionaram para que eu pudesse prosseguir com meus estudos. Agradeço aos amigos pelo constante apoio. Agradeço, em especial, aos amigos Daniel Palomino e Felipe Sampaio por estes dois anos de convívio. Agradeço à Universidade Federal do Rio Grande do Sul por ter proporcionado um ambiente de pesquisa de excelência. Finalmente, agradeço a todos aqueles que de alguma forma fizeram parte desta conquista.

CONTENTS

LIST OF ABBREVIATIONS AND ACRONYMS

APR	Acceptance Policy Rule.
DSR	Dynamic Search Range.
GA	Genetic Algorithm.
MNA	Memetic Networks Algorithm.
PSO	Particle Swarm Optimization.
SRP	Social Relative Performance.

LIST OF FIGURES

ABSTRACT

Our inability to perceive and understand all the factors that account for real-world phenomena forces us to rely on clues when reasoning and making decisions about the world. Clues can be internal such as our psychological state and our motivations; or external, such as the resources available, the physical environment, the social environment, etc. The social environment, or social context, encompasses the set of relationships and cultural settings by which we interact and function in a society. Much of our thinking is influenced by the social environment and we constantly change the way we solve problems in response to our social environment. Nevertheless, this human trait has not been thoughtfully investigated by current computational models of human social problem-solving, for these models have lacked the heterogeneity and self-adaptive behavior observed in humans. In this work, we address this issue by investigating the impact of social context in social problem solving by means of extensive numerical simulations using a modified social model. We show evidences that social context plays a key role in how the system behaves and performs. More precisely, we show that the centrality of an agent in the network is an unreliable predictor the agent's contribution when this agent can change its problem-solving strategy according to social context. Another finding is that social context information can be used to improve the convergence speed of the group to good solutions and that diversity in search strategies does not necessarily translates into diversity in solutions. We also determine that even if nodes perceive social context in same way, the way they react to it may lead to different outcomes along the search process. Together, these results contribute to the understanding that social context does indeed impact in social problem-solving. We conclude discussing the overall impact of this work and pointing future directions.

Keywords: Computer Science; Artificial Intelligence; Social Computing.

RESUMO

Nossa incapacidade em compreender todos os fatores responsáveis por fenômenos naturais faz com que tenhamos que recorrer a simplificações na representação e na explicação destes. Por sua vez, a forma com que representamos e pensamos a respeito destes fenômenos é influenciada por fatores de natureza interna, como o nosso estado psicológico, ou então de natureza externa, como o ambiente social. Dentre os fatores externos, o ambiente social, ou contexto social, é um dos que tem maior influência na forma que pensamos e agimos. Quando estamos em grupo, mudamos a todo instante a forma com que resolvemos problemas em resposta ao contexto que nos cerca. Entretanto, esta característica até então foi pouco explorada em modelos computacionais de resolução coletiva de problemas. Este trabalho investiga o impacto do contexto social na resolução coletiva de problemas. Nós apresentaremos evidências de que o contexto social tem um papel importante na forma com que o grupo e o indivíduos se comportam. Mais precisamente, nós mostraremos que a centralidade de um indivíduo na rede social nem sempre é um bom preditor de sua contribuição quando o mesmo pode adaptar sua estratégia de busca em resposta ao contexto. Além disso, mostraremos que a adaptação ao contexto social por parte dos indivíduos pode melhorar o desempenho coletivo, facilitando a convergência para soluções boas; e que a diversidade de estratégias de resolução do problema não leva necessariamente a uma diversidade de soluções na população; e que, mesmo que o contexto social seja percebido da mesma forma pelos indivíduos, a forma com que eles reagem pode levar a diferentes resultados. Todos estes resultados suportam a ideia de que o contexto social deve ser considerado em experimentos com resolução social de problemas. Por fim, concluímos o trabalho discutindo o impactso do mesmo e apontando novos problemas a serem investigados.

Palavras-chave: Ciência da Computação, Inteligência Artificial, Computação Social.

1 INTRODUCTION

A system capable of problem-solving is usually regarded as an intelligent system (NEWELL; SIMON, 1976). It is not surprising, though, that our intelligence is closely related with our skils in problems-solving. It is through such skills that we accomplish everyday trivial activities, such as deciding what is the fastest way to the airport, or more complicated activities such as analyzing the outcomes of a specific problem involving the expected outcomes of a group of people. The set of mental procedures we employ to find the best solution to problems is known as the problem-solving strategy, or heuristic (PEARL, 1984; WEISBERG, 2006). Different problems require different problem-solving strategies, therefore, there are strategies that are "well-practiced and virtually automatic, whereas others involve more on-line consideration and computation" (TAYLOR, 1998).

The best problem-solving strategy depends on the circumstances we face. For instance, the fastest path to the airport depends on the transportation methods available at the time. To decide which strategy is the best one, we rely on two kinds of clues: internal and external clues(FISKE; TAYLOR, 1991; SCHWARZ, 1998). Internal clues include our mood, motivation, expertise, personal goals; external clues include the physical environment, the physical resources available, the social environment, etc.

Social context is a particularly relevant external clue because much of our thinking is influenced by the social surrounding (LEVINE; RESNICK, 1993). It is difficult to imagine a purely cognitive situation where emotions, social meanings, social motivations, or any social artifacts make no difference at all. One explanation for the social cognition nature of our thinking is that it emerges from our internalization of social experiences during our socialization process (VALSINER; VEER, 1988). From a less theoretical and more pragmatical perspective, Mason and Watts 2012 have found that, as problem-solvers, we indeed change a lot our strategies when socially solving a problem.They found that humans performed generally better and copied considerably less solutions when compared to artificial counterparts. They attribute such result to the observation that all artificial agents in the population followed the same problem-solving strategy — in contrast with humans where the set of problem-solving strategies was highly heterogeneous — and that agents were not allowed to change their search behavior along the simulation — whereas the search strategies employed by humans varied a lot along the experiment. However, our inability to understand complex systems composed of distinct parts that can interact is limited (BARABÁSI; ALBERT, 1999) and, hence, it remains unclear whether social context accounts for the observed patterns in the experiment.

In this work, we address the relevance of social context in social problem-solving by measuring its impact in individual and collective performance. We do so by means of extensive numerical simulations using a modified social model. We provide evidences that social context is indeed relevant to social problem-solving. We show, for instance, that centrality measures of an agent in the network is an unreliable predictor the agent's contribution when this agent can change its problem-solving strategy according to the social context. Another finding is that social context information can be used to improve the convergence speed of the group to good solutions and that diversity in search strategies does not necessarily translates into diversity in solutions. In addition, we also determined that even if nodes perceive social context in same way, the way they react to it may lead to different outcomes along the search process.

We now detail the how we will tackle the problem.

1.1 General Methodology

We decided to investigate the impact of social context through an empirical approach instead of an analytical one. This is because agent-based models of problem-solving defy analytical approaches despite their triviality (POLI; KENNEDY; BLACKWELL, 2007). The stochastic processes that govern the individual behavior of the agents make these systems hardly predictable. In addition, these agents interact by exchanging messages that influence the outcomes of their actions and their peers actions in the future — the units that compose the system are heavily interdependent, yet autonomous.

We also decide to use the Memetic Networks model(ARAÚJO; LAMB, 2008, 2010) as our underlying framework, which attempts to mimic how humans solve problems when in group. The model of Memetic Networks specifies three generic rules that command the system during the problem-solving process. The model provides us flexibility to accommodate social context without violating any of its theoretical and philosophical underpinnings.

This dissertation is the aggregate product of each of the following steps.

1. *Situate* this work within a theoretical referential by providing the general and the specific context information.

2. *Formalize* the concept of social context within the chosen framework of the Memetic Networks.

3. *Develop* a method to measure social context in Memetic Networks.

4. *Devise* mechanisms that allow agents to react to social context.

5. *Compare* the results with the original model (control).

The Background chapter, chapter 2, covers the first step. The second, third, and fourth steps are covered in the Social Context chapter, chapter 3. The Experiment's chapter, chapter 4, presents the fifth; there we will discuss the experiments and their respective results. Finally, chapter 5 gives an overall discussion of this work and points further directions worthy of consideration.

We have performed parallel investigations along this dissertation. Even if these investigations shared with the present work the model of Memetic Networks as the basic framework, they sought to solve fundamentally different problems. Therefore, we opted to separate them from the main topic of this dissertation in an attempt to make its content more streamlined and terser. Nonetheless, we include and contextualize them in the appendix.

2 BACKGROUND

This chapter presents the relevant and essential background information to the unfamiliarized reader. We present the field of Social Computing, its perspectives and its current research issues. We then proceed to detail the research issue of agent-based social modeling and its methodological principles. Finally, we describe the general ideas behind the Memetic Networks model.

2.1 Social Computing

Social computing is a computing paradigm that involves fields such as sociology, social psychology, organization theory, and communication theory to analyze, model and understand social behavior phenomena. One social computing's main goal is to produce applications that support and improve interaction and communication of people in informational social systems (KING; LI; CHAN, 2009). Social computing can be precisely defined as the "computational facilitation of social studies and human social dynamics as well as the design and use of Information and Communication technologies that consider social context" (WANG et al., 2007).

This definition of social computing has two interdependent perspectives: one technological and one theoretical. The technological perspective of social computing seeks to build informational systems that consider social context and support effective communication among people. Therefore, findings in social computing can be directly applied in the development of any information system where people interact. One example is to create web services and tools, like blogs or social network services, that help people share and spread their ideas more easily and i in real-time to a large pool of readers.

The theoretical perspective on social computing aims to solve the research issues that stems from the application of technological domains and to understand theories about the social life. Recently, the theoretical branch has come to be known as Computational Social Science (LAZER et al., 2009) while the term Social Computing has become increasingly more associated with technological . Among the issues in Computational Social Science, there is the problem of representation of social information and social knowledge, the problem of analysis and prediction techniques for social systems, and the problem of modeling social behavior at both individual and collective levels using artificial agents.

In order to illustrate the interdependence, suppose that a store owner wants its recommending system to perform more effective advertisement by increasing its accuracy. One way to do so is to improve current knowledge discovery systems that are based solely on their client's attributes and their past interactions. If we assume the hypothesis that people who are friends tend to buy similar products, then, knowing who is friend of whom is valuable information to consider. The theoretical branch is concerned to prove or not such hypothesis while the technological branch seeks to solve design questions that may arise during the development of such system. We now proceed to further detail the theoretical branch, for this is closer to this work.

Social relations, institutional hierarchy, influence, roles, social context, among others, represent social information. From the individual's perspective, social knowledge represents the memory this individual has about exogenous entities as well as its endogenous cognitive variables that guide its goals and motivations. Both social information and social knowledge are basic descriptive information from which decisions can be taken in order to coordinate activities in a social system. Representing this information accurately at the right level of abstraction is a fundamental research issue. Social information about the structure of a social system is usually described using graphs. This formalism allows one to represent social ties as edges and social agents as nodes in the graph (EASLEY; KLEINBERG, 2010).

The analysis and prediction techniques for social systems includes statistic methods to analyse and predict costs and benefits associated with strategies, polices and decision-making methods. Some of these techniques include structural equations, cellular automata, Bayesian networks, hidden Markov models. Techniques from data-mining, machine-learning, and data visualization can be used to help to identify patterns and relationships within data. Social network analysis is focused on the relationships among social entities, and on the patterns and implications of these relationships (WASSERMAN; FAUST, 1994).

Agent-based social modeling is a theoretical approach that seeks to understand social life through the use of simplified formal models of group of individuals. It originated in computer science and artificial intelligence to study complex adaptive systems composed of a number of autonomous but interdependent entities (MACY; FLACHE, 2009). Such models are called agent-based because their basic units are the individuals. There are two aspects in agent-based social modeling: the macrosocial aspect, that seeks to generate or reproduce a pattern in the population from the aggregate behavior of all agents; and the microsocial aspect, that is concerned with methodologies to model agents and their interactions and the influence of the whole population in the individual. Since the research issue of agent-based social modeling is central in this dissertation, we will further explain it in the next section.

2.2 Agent-based Social Modeling

Methodological Individualism is a theory based on the principle that some macrosocial patterns can be explained as the outcome of individual decisions and actions (MACY; FLACHE, 2009). Examples of macrosocial patterns include herd behavior (BANERJEE, 1992), the evolution of cooperation(NOWAK, 2006), residential segregation (SCHELLING, 1978), homophily (MCPHERSON; SMITH-LOVIN; COOK, 2001), etc. The Methodological Individualism assumes that the individuals' attributes play no direct role in macrosocial outcomes if they are unable to make a decision and act in response to the environment. In this context, the Methodological Individualism theory can be seen as a reductionist approach to understand the social life.

To understand how some macrosocial pattern unfold from the interaction among individuals, researchers have resorted to computational models based in a population of artificial individuals, called agents (EPSTEIN, 2011). Agents are the basic units from agent-based models. They are computer programs whose behavior is formally encoded by rules. As so, they can perform computations to decide which course of action is the best one considering future outcomes — by solely measuring the utility of each available action — or previous outcomes — the agent takes the decision based in what actions proved successful in previous interactions. The dynamics of the model is given by the iteration of this process (MACY; FLACHE, 2009).

The goal of agent-based social modeling is to generate a complex population pattern from a simple model of local interactions among agents. Local interactions in Agent-based models must be the simplest and must hold as few as possible assumptions if one wants to explain what are the causal actions that build to the macrosocial patterns. Overly complex and arbitrary agent-based models can potentially hinder the research's ability to draw relevant and valuable cause-effect relations from the model, i.e. the model may look completely random through close inspection. In such cases, determining the underlying causal mechanism for the pattern is utterly difficult. However, highly complex and non-linear macrosocial patterns are hard to achieve from bottom-up. One classic example of agent-based model that retains simplicity and is able to build rich population pattern is the Schelling's residential segregation model (SCHELLING, 1978).

We will present the agent-based model of Memetic Networks in next section.

2.3 Memetic Networks

Earlier ideas on psychology and artificial intelligence tried to understand people 'as general problem solvers (NEWELL; SIMON, 1972). These ideas prompted the development of numerous cognitive models of how a person in isolation would solve problems. Social Computational Science shifted this focus from the individual to the collective level (LAZER et al., 2009; KEARNS, 2012). The Memetic Networks model is an example of an agent-based model resultant from this shift. It models how humans collectively solve problems (ARAÚJO; LAMB, 2008, 2010).

Memetic Networks is computational framework that determines guidelines to the development of a system whose objective is to search for the best solution to a given problem. One implements a Memetic Network to investigate the system itself under some scenarios or to devise a novel problem-solving system that solves a problem more efficient than current systems.

The model defines two entities: elements and rules. Both are conceptually problem invariant. Elements are: nodes, that represent the agents; edges, that represent social ties; and *memes*, that represent units of information. Rules state how these elements process, interact, and relate to each other. A *memenet* is what is called the final construct of a Memetic Network. It is when one assembles the Memetic Network's elements and then describes the rules by which the elements will interact. A memenet is composed of the following basic components:

Nodes — Nodes are the fundamental units of any memenet. They carry a complete or temporary solution to the problem at hand. This solution has a numerical value associated that describes how good it is, i.e. its objective quality. In addition, nodes are responsible for retrieving, aggregating and processing *memes*.

Meme — Memes are basic units of information (DAWKINS, 2006) and a solution in a memenet is a set of memes. The concept of what is "basic unit of information" depends on problem at hand. Links can be bi-directional or not.

Links — Links connects nodes so they can exchange memes. They are modeled as edges in a graph and they represent social ties among agents. If a link connects two nodes, than these nodes are said to be neighbors. Neighbors nodes can exchange memes during the problem-solving process. Links can by static — established at beginning of the search and kept until the end — or dynamic — at every iteration, or a group of iteration, they are updated according to some aspect of system.

The Memetic Network elements alone are not self-sufficient to perform search as it remains necessary to define how these elements will behave when assembled together. This is accomplished by three rules, which roughly capture how humans process information in a social context. These rules detail (1) how nodes choose to connect to each other, (2) how information is retrieved from the social network, and (3) how individual nodes contribute to a solution being sought. Each rule is detailed in what follows.

Connection Rule — Defines how the nodes should connect, i.e. how links are to be established. This rule can be performed once before the search takes place, characterizing a static network topology. Alternatively, the connection rule can be executed during the search process so the network topology can evolve together with the system. As in other socially-inspired models (e.g. Particle Swarm Optimizers), there is currently no conclusive evidence about what topology is generally better for specific types of problems (POLI; KENNEDY; BLACKWELL, 2007).

Aggregation Rule — This rule controls the interaction between connected nodes, that is, how nodes retrieve and aggregate information from their network neighbors. Typically, a node retrieve information [1] from better ranked neighbors only. After retrieving memes, a node must aggregate — i.e. combine — them to assemble a new solution. The aggregation rule must also describe algorithmically how these memes are to be combined. Aggregation is highly dependent on the network topology [2].

Appropriation Rule — In possession of the recently aggregated solution, a node may add some novelty to it. In this step, local information is added to the solution, allowing memenets to explore the search space, much in the same way mutation works in genetic algorithms. Appropriation can happen by simple random changes to the solution or by applying some deterministic local search to it (in a similar fashion to Genetic Algorithms (GOLDBERG, 1989) and Memetic Algorithms (MOSCATO, 1999) respectively).

Although the aggregation and the appropriation rules depend on the problem, the structure of a Memetic Network is the same across different problems. Algorithm 1 describes this structure of a generic memenet that uses a static network. The *stop condition* can be the number of iterations or a desired solution quality threshold and the $Problem_{size}$ input represents the size of the array, or equivalently, the dimension of the function's domain.

The memenet is first initialized with a population of nodes. These nodes are carry only a solution and have no links connecting them to other nodes. The second step is the execution of the connection rule, which establishes links among nodes and, thus, it gives

[1]Retrieval of information is considered noiseless. It remains an open question whether communication noise could be beneficial or not for population-based optimization problems.

[2]This feature is where socially-inspired optimization algorithms differ the most from Evolutionary Algorithms that have no explicit structured social network to regulate breeding.

Algorithm 1: Structure of Memetic Network using a static network.

```
Input  : $Population_{size}$, $Problem_{size}$
Output: $Solution_{best}$

// Initialize the population.
memenet ⟵ $\emptyset$
for $i \leftarrow 1$ to $Population_{size}$ do
    memenet ⟵ memenet ∪ NewNode ($Problem_{size}$)

// Establish links between nodes.
Connect (memenet)

// The search process itself.
repeat
    foreach $node \in$ memenet do
        Aggregate ($node$,$Problem_{size}$)
        Appropriate ($node$,$Problem_{size}$)

    // Keep track of the best solution found so far.
    UpdateBestSolution ($Solution_{best}$, memenet)
until stop condition
return $Solution_{best}$
```

every node the references about whose nodes are their neighbors. After these two initial steps, the algorithm iterates the aggregation and appropriation rules over all population. This is done for a limited number of turns while the algorithm keeps track of the best solution found so far.

In one wants to model a dynamic network topology, one need to place the connection procedure within the for loop. Naturally, this network feature incurs more overhead than a static network and adds complexity to whole system.

We now describe how to instantiate a memenet to the problem of function minimization. We need only to formalize the $aggregation$ and the $appropriation$ procedures.

2.4 Memetic Networks and the Problem of Function Minimization

Real-valued function minimization is a problem where one seeks the input that minimizes a given function. The input is a real-valued array and the output is numerical value. Each input's component represents a position within a single dimension of the search space. By definition, a given input has exactly one output which is quality of that solution. The lowest the values, the highest the quality.

To develop a memenet that performs minimization over real-valued function, one can associate an array with a complete solution and a meme to an array component. Therefore, an array of size n is a solution composed of n memes. After defining these concepts, we need to define the aggregation and the appropriation rule.

The aggregation rule retrieves and then combines memes from better ranked neighbors to create a new solution. This process is described in Algorithm 2.

Algorithm 2: The aggregation rule instantiated to function minimization.

```
Input: $node$,$Problem_{size}$
memepool $\longleftarrow [node_{solution}]$
foreach $neighbor \in node_{neighbors}$ do
    if $neighbor_{solution}$ isBetterThan $node_{solution}$ then
        memepool $\longleftarrow$ memepool $\cup$ $neighbor_{solution}$
if |memepool| $> 1$ then
    for $i \leftarrow 1$ to $Problem_{size}$ do
        $SolutionChosen \longleftarrow$ RandomlyChooseASolutionFrom (memepool)
        $node_{solution}[i] \longleftarrow SolutionChosen[i]$
```

The first step is to create the *memepool*: a data structure whose objective is to store better or equally ranked neighbors' solutions. It can be implemented as an array of arrays. The node's own solution is also include in the *memepool* too. The second step generates a new solution by selecting memes from a randomly chosen solutions in the $MemePool$. All solutions from the $MemePool$ have equal chances to be chosen.

Aggregation does not generate new memes because it only compose new combinations from already known solution components. To address aggregation's inability to add innovation, the memenet uses an appropriation rule by which nodes to modify their memes.The Algorithm 3 portrays random search that randomly replaces a meme by another valid meme.

Algorithm 3: The appropriation rule instantied to minimization.

```
Input  : $node$, $Problem_{size}$, $P_{appropriation}$
Output: $Solution$
for $i \leftarrow 1$ to $Problem_{size}$ do
    With probability $\frac{1}{Problem_{size}}$ :
        $Solution_i \longleftarrow$ RandomlyChooseMeme ()
return $Solution$
```

For each component of the solution, a valid meme is generated and attributed with probability $\frac{1}{Problem_{size}}$. As with mutation probability in genetic algorithms, the probability of changing a meme should be small enough so that convergence is not drastically disrupted. All valid memes have the same probability of being generated in this process.

3 SOCIAL CONTEXT

Social context encompasses "the social relationships, and cultural milieus within which defined groups of people function and interact" (BARNETT; CASPER, 2001). Culture and language, for instance, are two such artifacts that compose social context and that constantly influence the way we think about the social, physical and spiritual life (VALSINER; VEER, 1988). Therefore, social context is a particularly important factor in our problem-solving skills because much of what we think and do is influenced by it (LEVINE; RESNICK, 1993).

As explained in the introductory chapter, recent evidence pointed that humans frequently change their problem-solving strategies in the social setting and, currently, no social model capture this human behavior. According to (MASON; WATTS, 2012), a majority of social problem-solving models use homogeneous populations where the agents are unable to change their strategies; models that lack heterogeneity and the self-adaptability observed in real systems undermines attempts to generalize these unrealistic models. It is unknown, though, whether social context accounted for the observed behavior in these experiments.

Since we are concerned with the investigation of the impact of the social context in social problem-solving, we proceed to explain social context from the perspective of Memetic Networks, and elaborate on the various interpretations that social context can have within it.

3.1 Social Context in Memetic Networks

From a conceptual point of view, any information that is not part of the problem itself which a memetic node receives, processes, and sends is social context — in other words, anything that is not a meme and flows from one node to another can be used as social context information.

This section lists some kinds of information that can be candidates to be used as social context information. Here we refer to a problem-solving strategy as the combination of the search algorithms implemented within the aggregation, the appropriation, and the connection rules. For example, if two nodes implement the same aggregation and appropriation rules but a distinct connection rule, then these nodes do not have the same problem-solving strategy. Some possible interpretations for social context are:

- *Social Relative Performance* — A node can compare its current solution quality with its neighbors to have a hint on how well it is doing compared to them. After this comparative assessment, the same node can use this information to adapt its current problem-solving strategy;

- *Local Contribution* — A node can contribute to its neighborhood as long as it holds a better solution. The information of how many times a node contributed to its neighborhood can characterize a measure of local social contribution;
- *Global Contribution* — The information of how many times a node contributed to the best solution known by the whole system can be also taken as social context;
- *Neighbors' Strategy* — A node can determine which combination of rules led its neighbors to improve their respective solution quality. With this information, it can adapt its problem-solving strategy according to the social circumstances;
- *Neighbors' Centrality* — From a collective point of view, centrally positioned individuals tend to be more relevant to the whole than peripheral individuals (BONACICH, 1987; BURT, 1995). Therefore, the information about the network's position of neighbors can give a clue about their likelihood to hold good solutions.

There could many other interpretations of social context. The next section details one of them, the simple Social Relative Performance.

3.2 Measuring Social Context

We define the Social Relative Performance (SRP) as the relationship between the number of neighbor nodes that are faring better against the total number of peers nodes of the focal node. The idea is to count how many neighbors are faring better than the focal node and divide this number by the total number of neighbors. The SRP ratio is given by Equation (3.2) where $Better$ is the set of better ranked neighbors of node n and $Total$ the set all neighbors of the focal node n.

$$SRP(n) = \frac{|Better_n|}{|Total_n|} \tag{3.1}$$

The best ranked nodes in the network will have an SRP of 0.0, and, the worst ranked nodes, an SRP of 1.0. In addition, we define a node a *leader* if it has SRP equals to 0.0. Figure 3.1 depicts the possible SRP values that a node can assume if it has 6 neighbors. SRP increases linearly as the number of better ranked neighbors increases.

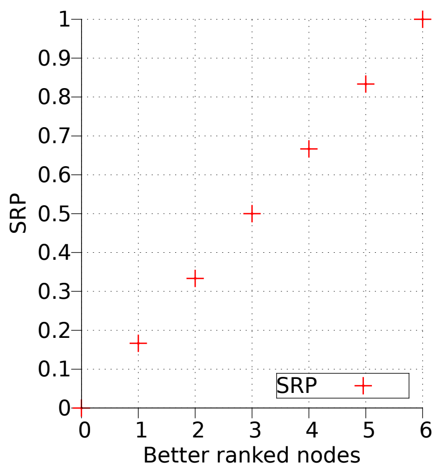


Figure 3.1: Possible SRP values for a node with 6 neighbors.

We opted to use this non-parametric model of social context because of its simplicity. Using an overly complex and arbitrary form can undermine our investigation in the sense that it would require a parameter exploration. We acknoledge the importance of the study of more sophisticated forms of social context but, at this point, we must restrain ourselves to the most simple forms. We still need to define how this information can be used to change its current problem-solving strategy. The next section elaborates on that.

3.3 Using Social Context Information

As with social context interpretations, there could be many mechanisms by which a node is able use SRP to change its current problem-solving strategy. There could be mechanisms within any rule: the connection rule, where links could settled and unsettled according to the social context; the aggregation rule, where the combination of solutions could consider social context; and the appropriation rule, which can use social context information as a parameter to adjust exploration and exploitation rates within of the algorithm implemented within it.

We will start by listing and explaining candidate mechanisms that can be implemented within the appropriation rule:

- *Dynamic Appropriation Probability* — In previous chapter, we implemented a random search algorithm inside appropriation(as described in algorithm 2). The algorithm there used a fixed probability to change memes ($\frac{1}{Problem_{size}}$. The mechanism of Dynamic Appropriation Probability basically uses the social context information to dynamically adjust this parameter. A node that lags behind its peers can increase this probability aiming at increasing its chance of finding better memes than the current ones — since these do not constitute a good solution;

- *Acceptance Policy Rule* — The Appropriation rule is a compulsory rule. Even if a node is the best among its peers, it must execute it at the risk of loosing good memes. Thus, one may ask what if a leader node (SRP of 0.0) could decide upon accepting the appropriated solution or not? It is plausible to accept higher solutions and reject lower quality solutions. In this case, social context is used to change the way the node decide which results must be accepted or not. The algorithm itself can stay the same;

- *Dynamic Search Range* — Instead of randomly replacing a meme with *any* meme during appropriation, it may be reasonable to attribute a higher probability to "closer" memes when the node is performing well (i.e. favor small changes to the solution when one is doing well compared to their peers). Alternatively, a node which is performing poorly compared to its peers may benefit from modifying its solution to a greater extent;

- *Algorithm Switch* — This mechanism tells the node to use a completely different algorithm within appropriation, and not just adjusting a parameter as function of social context. For example, a node may choose to perform as a random search or as guided local search method according to its social context.

The next two subsections will explain implementations of the Acceptance Policy Rule and the Dynamic Search Range mechanisms, respectively.

3.3.1 Acceptance Policy Rule

We believe that the Acceptance Policy Rule mechanism (APR) is the most simple form of self-adaptive mechanism that can respond to social context. The mechanism we propose grants to leader nodes ($SRP = 0.0$) the decision to accept or not the appropriated solution. The hypothesis behind this mechanism is that leaders could act as keepers of locally valuable pieces of information: good memes. Nodes that implement the APR mechanism are called APR nodes.

The inspiration for this mechanism comes from elitism selection schemes developed for GA(GOLDBERG, 1989). There, a set with better individuals from both the offspring and the parent generation moves directly to next generation without any change at all. This way, valuable genes are passed to generation to the next generation(THIERENS, 1998). The main difference among the elitist selection and the APR mechanism is that the former is global and the second is local. More specifically, the elitist selection mechanism originally considers all the population to select the set of better individuals: it uses only fitness information to rank all individuals in a group and only then selects the elitist subgroup. In addition, these individuals never perform any genetic operation as long as they remain at the elite group — and they can stay in that group indefinitely. Meanwhile, the APR mechanism is local in the sense if a leader can keep its solution even this leader happens to perform inefficiently when compared to the whole population: social ties and solution quality are combined to select what individuals are allowed to keep their solutions. APR nodes that are leaders still perform appropriation, though they do not aggregate memes because they know no better neighbors.

Algorithm 4 describes the APR mechanism within the appropriation rule. Note that the SRP is calculated beforehand as is passed as an input to the mechanism.

Algorithm 4: The APR mechanism within appropriation. Note that the SRP is calculated beforehand.

```
Input : $node$, $Problem_{size}$, $SRP_n$
$TmpSolution \longleftarrow \emptyset$
for $i \leftarrow 1$ to $Problem_{size}$ do
    With probability $frac1Problem_{size}$:
    $TmpSolution[i] \longleftarrow$ RandomlyChooseMeme ()
if $SRP_n$ > 0.0 or $TmpSolution$ isBetterThan $node_{solution}$ then
    $node_{solution} \longleftarrow TmpSolution$
```

3.3.2 Dynamic Search Range

The mechanism of Dynamic Search Range (DSR) uses social context to adapt how a node exploits local information during the search. Instead of randomly replacing a meme with *any* valid meme during appropriation, the DSR mechanism limits the search range according to the SRP parameter. If the appropriation rule implements a random search algorithm, then it is plausible to give higher probability to "closer" memes when the node is performing well (i.e. favor small changes to the solution when one is doing well compared to their peers). Alternatively, a node which is performing poorly compared to its peers may benefit from modifying its solution to a greater extent.

In order to instantiate DSR mechanism, we defined Δ to be the maximum *variation* allowed when modifying each component of the solution (i.e. when choosing the a new component for the solution, it must vary at most with $+\Delta$ and at least $-\Delta$). The Δ variation must be calculated as function of the SRP of a node. Therefore, we used a linear function model to calculate Δ value. This function uses the maximum and minimum valid input values that the function to be minimized supports and calculates the maximum interval possible. The resultant interval is then multiplied by the SRP and summed with a correction factor called α. The correction factor serves as the minimum amount of variation to be used when the node is a leader node ($SRP = 0.0$). Equation (3.2) describes this linear function where Max and Min represent the maximum and minimum input values allowed by the function, respectively.

$$\Delta = (|Max| + |Min|)/\,2 \times SRP + \alpha \tag{3.2}$$

In this equation, a node with an SRP equals to zero — i.e. the best ranked node in the neighborhood — will have an effective search range of $[-\alpha, +\alpha]$ while the worst node will have a search range of $[-(\alpha + (|Max| + |Min|)/2), +(\alpha + (|Max| + |Min|)/2)$. Nodes with intermediate performance will have linearly intermediate search ranges. Figure 3.2 describes Δ values as function of the SRP.

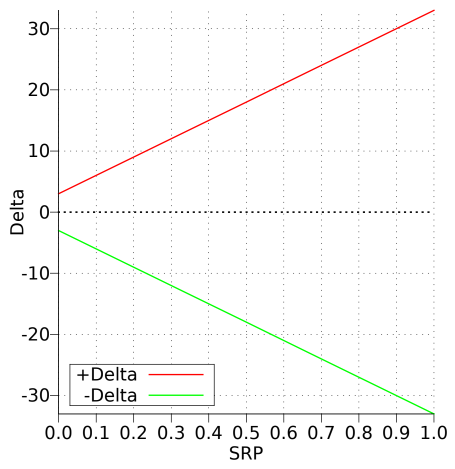


Figure 3.2: The linear function model that calculates Δ values using the SRP information. This example considers α as 3.0 and the valid maximum function value as 30.0 and minimum as -30.0

Algorithm 5 describes the APR mechanism within the appropriation rule.

Algorithm 5: The DSR mechanism within appropriation. Note that the SRP is calculated beforehand as is passed as an input to the mechanism.

```
Input: $node$, $Problem_{size}$, $SRP_n$
$\Delta \longleftarrow (|Max| + |Min|)/\,2 \times (SRP_n) + \alpha$
for $i \leftarrow 1$ to $Problem_{size}$ do
    With $\frac{1}{Problem_{size}}$ probability:
      $node_{solution}[i] \longleftarrow$ RandomlyChooseMemeBetween ( $Solution_i - \Delta$ ,
      $Solution_i + \Delta$ )
return $Solution$
```

This appropriation version requires a repair routine afterwards for it may generate solutions outside the valid search range. We proceed to use a simple routine which sets every meme that is smaller than Min to Min and Max to memes that are larger than Max.

4 EXPERIMENTS

In this chapter we describe the experiments themselves and their respective results. We performed three experiments to assess the impact of the social context in social problem solving:

The experiment 1 , described in section 4.3, seeks to understand the individual impact of how centrality and social context relate to each other;

The experiment 2 , in section 4.4, aims to investigate the collective aspect of how social context impacts on the speed that the system converges to the optimal solution (if it converges) and in the solution diversity in the population;

The experiment 3 , in section 4.5, explores the dynamics of a heterogeneous system composed with canonical, APR, and DSR nodes.

Each experiment has its own specific methodology, nevertheless, we will discuss some general points in the next section.

4.1 Experimental Methods

All experiments were developed using the same parametric computational framework. This problem-solving framework was described in *Python3* (version 3.2) code. We used the specialized numerical package *numpy* to manage the calculations, the statistical tools from the package *scipy* to handle statistical calculations, the *Networkx* package to model the network topologies used in experiments, and the *matplotlib* package and *gnuplot* program to plot the results. The pseudo-random number generator used across all experiments was standard python random number generator which uses "the Mersenne Twister as the core generator. [...] The Mersenne Twister is one of the most extensively tested random number generators in existence."(ROSSUM; DRAKE, 2012). The original random number generator can be found in Matsumoto et al. original work(MATSUMOTO; NISHIMURA, 1998).

The main method used to assess the impact of social context is to compare the canonical Memetic Networks model with the models composed entirely with Aceptance Policy Rule (APR) nodes and Dynamic Search Range (DSR) nodes. The canonical model is the model explained in the Background chapter, chapter 2. The α parameter used by the DSR was fixed at 0.0001.

The stochastic nature of the models we will investigate makes them harder to analyze than purely deterministic models. Randomization is used inside aggregation, appropriation, and initialization. It very likely results will be different across independent runs. To

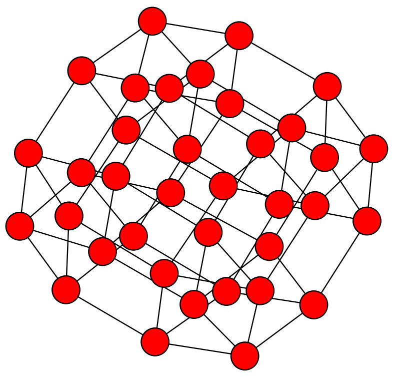


Figure 4.1: A periodic grid topology of 6×6 nodes.

tackle this problem, we will repeat the experiments 50 times and use the mean and confidence intervals to describe the data over all the independent executions (KRUSCHKE, 2012).

These nodes executed their search operations until the system reaches the stop criteria. In our experiments, the stop criteria used was number of iterations where each complete iteration counted after all nodes have executed their search operations. Alternatively to the number iterations, we could use a threshold value for the best known solution: if the systems find a solution that is best or equal to the threshold, than the system ceases the search. However, threshold values are usually arbitrary and their use may difficult future comparisons.

The grid topology used in the experiments 2 and 3 connects every node to other four other nodes: one above, one below, one at the right, and one at the left. Nodes at extremities are connected to the other side of the grid. This topology fix the number of neighbors of all nodes to four. Of course there could be other topologies more close to real social network such small-world (WATTS; STROGATZ, 1998) or those generated by the preferential attachment mechanism (BARABÁSI; ALBERT, 1999); however, by using the grid topology, we isolate the impact of centrality and facilitate the analysis of the results because there is no parameter in this topology. With illustrative purposes, we provide a three dimensional representation of a 6×6 grid topology Figure 4.1.

A compilation of traditional functions used in in the problem can be found in the work of Reynolds and Chung(REYNOLDS; CHUNG, 1997). We will use five functions that will be individually described in the next section. The domain dimension used was of 30.

4.2 Benchmark Functions

All functions can generalized to as many dimensions as one wants. However, the maximum and minimum limit values for components of a solution are fixed. The mathematical description of these functions is in table 4.1. The search range column specifies the minimum and maximum values for each component of the input. The n represents the number of dimensions in the general form. More detailed information can be found elsewhere (MENDES, 2004).

Table 4.1: The benchmark functions used. The n represents the number of dimensions.

Name	Definition	Search Range $[x_{min}, x_{max}]^n$
Ackley	$20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$	$[-30.0, 30.0]^n$
Griewank	$\frac{1}{4000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos(\frac{x_i - 100}{\sqrt{i}}) + 1$	$[-600.0, 600.0]^n$
Rastrigin	$\sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	$[-5.1, 5.1]^n$
Rosenbrock	$\sum_{i=1}^{n-1}\left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right)$	$[-30.0, 30.0]^n$
Sphere	$\sum_{i=1}^{n} x_i^2$	$[-100.0, 100.0]^n$

The mathematical description of these functions hides more intuitive information about their search spaces. To help with the visualization, we provide bidimensional representations — a landscape — and a verbose description of the five functions in what follows.

4.2.1 Ackley Function

The Ackley function is highly multi-modal function with the best solution $f(\mathbf{x}) = 0.0$ at the very center $\mathbf{x} = [0, .., 0]$ of the search space. The function is likely to trap a greedy myopic search process towards the center. Every component of $\mathbf{x}$ is defined between $[-30.0, 30.0]$. Figure 4.2 depicts its landscape representation.

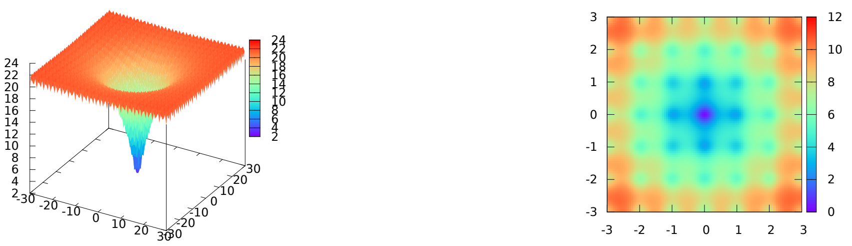


Figure 4.2: Ackley function landscape.

4.2.2 Griewank Function

The Griewank function is a multi-modal function with the largest search search space among all other functions. The best solution value is $f(\mathbf{x}) = 0.0$ at the center $\mathbf{x} = [0, .., 0]$ of the search space. Every component of $\mathbf{x}$ is defined between $[-600.0, 600.0]$. Figure 4.3 depicts its landscape representation.

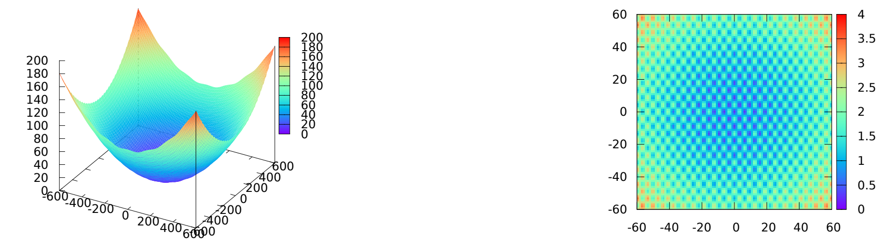


Figure 4.3: Griewank function landscape.

4.2.3 Rastrigin Function

The Rastrigin function is a multi-modal, with many local-minima distributed of the search space. The best solution value is $f(\mathbf{x}) = 0.0$ at the center $\mathbf{x} = [0, .., 0]$ of the search space. Every component of **x** is defined between $[-5.12, 5.12]$. Figure 4.4 depicts its landscape representation.

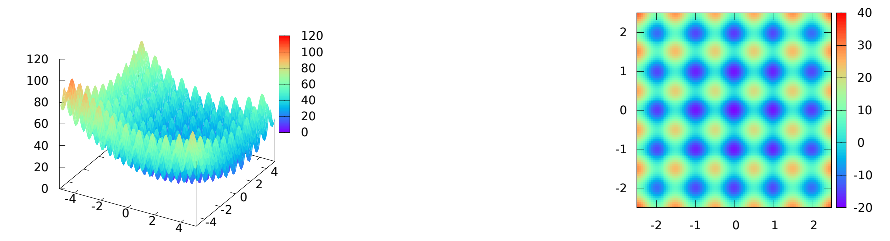


Figure 4.4: Rastrigin function landscape.

4.2.4 Rosenbrock Function

The Rosenbrock function is a unimodal function with a shape that resembles a banana. The gradient information may mislead algorithms. It is the only one function that has the global minimum at the center, in this case, the best solution value is $f(\mathbf{x}) = 0.0$ at $\mathbf{x} = [1, .., 1]$. **x** is defined between $[-30.0, 30.0]$. Figure 4.5 depicts its landscape representation.

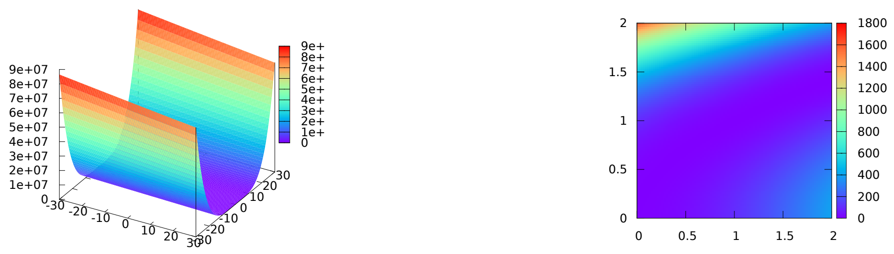


Figure 4.5: Rosenbrock function landscape.

4.2.5 Sphere Function

The Sphere function is the most simple function of the whole set. It is unimodal and the global optimum is at the very center of the search space, gradient information is reliable in this function. The best solution value is $f(\mathbf{x}) = 0.0$ at the center $\mathbf{x} = [0, .., 0]$ of the search space. Every component of $\mathbf{x}$ is defined between $[-100.0, 100.0]$. Figure 4.6 depicts its landscape representation.

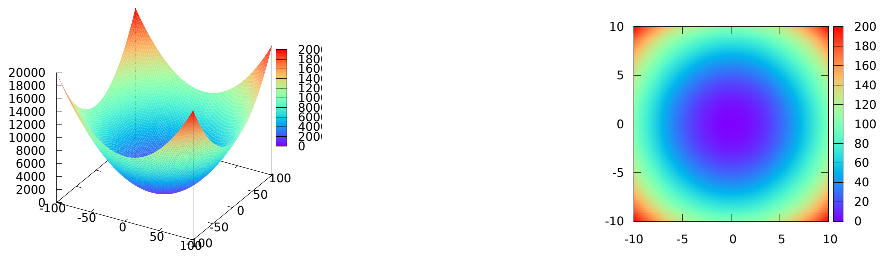


Figure 4.6: Sphere function landscape.

4.3 Experiment 1: Centrality and Social Context

We can characterize an individual by counting the number of connections that he has in the social network (BURT, 1995). In many domains that are modeled using graphs, nodes that have many connections, or hubs, tend to be more relevant than nodes have a few (BARABÁSI, 2003). For example, if one wants to shut down the Internet, or least render it useless, one needs to systematically aim at the most connected servers (ALBERT; JEONG; BARABáSI, 2000). In case of problem-solving, centrality has been associated with efficiency (FREEMAN, 1979).

In this experiment, we experiment the impact of social context in the relationship between centrality and efficiency. We proceed to characterize nodes by their betweenness centrality — a measure o centrality — and their respective global contribution. We call global contribution as a measure of how many times that node updated the known solution by the system. The Betweenness centrality measures how much information flows between nodes in the network by the way of the target node. In other words, it is the proportion of shortest paths between pairs of nodes that pass through the central node(FREEMAN, 1977). Equation 4.1 depicts how one calculates the betweenness centrality C of a node n: x, y and n are nodes and $||$ denotes the cardinality of the set.

$$C(n) = \sum_{x \neq n \neq y} \frac{|\text{shortest paths between nodes x and y that pass through n}|}{|\text{shortest paths between nodes x and y}|} \tag{4.1}$$

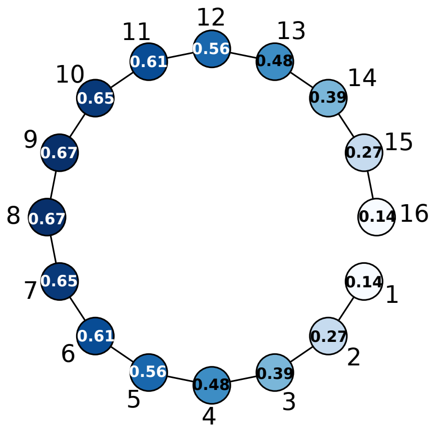
We compare the average result of the canonical Memetic Network model and a memetic network composed of APR nodes in the five functions and gather the global contribution after the 5000th iteration. Functionally, nodes from these two models have only the appropriation rule implemented differently.

The topologies used were the wheel and the path topology. We will explain these topologies and why they were chosen in what follows.

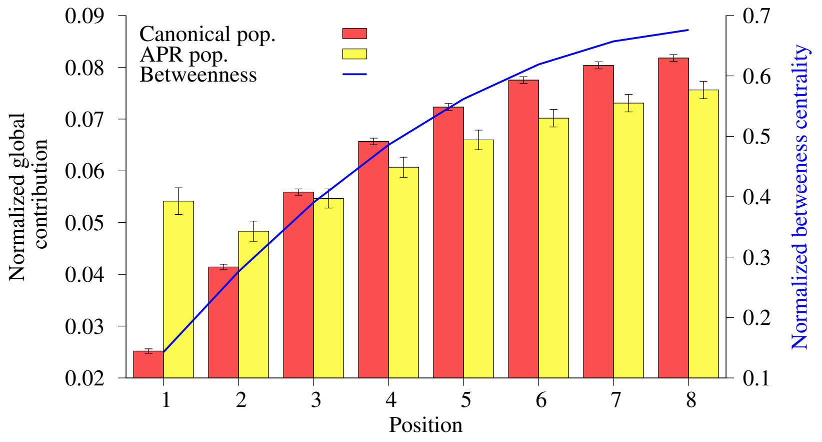
Path topology The path topology is built from a 2-regular graph also, but this time a single random edge is removed, opening the ring and leaving the network with two extremes far away from a center of the path. In this topology, centrality distributes itself as bell-shaped form: the lowest centrality values are in the extremes and centrality increases linearly as the node's position approached the middle of the path. The assumption tested here that nodes near the middle tend to contribute more(BONACICH, 1987; BURT, 1995) and the hypothesis is social context may spread the contribution more equally along the path;

Wheel topology The wheel topology is built from a 2-regular graph where one node is selected randomly to be connected to all other nodes, this way the graph has highly centralized node. This topology has a central and highly connected node that sees all other nodes while the rest population is connected to only to neighbors. The assumption is that the central node will contribute the most without social context information and the hypothesis is that social context could reduce its contribution.

The first experiment used the path graph with 16 nodes. This scenario considered the canonical memetic network model and the second scenario considered a population composed entirely of APR nodes. The results summarizes the results from the five previously described functions. A bidimensional representation of the topology including the positional labeling and the betweenness information is depicted in figure 4.7a. Figure 4.7b shows the results of the global contribution per position — due to symmetry property and in an attempt to facilitate visualization, we plotted only 8 nodes, where node 8th node represent the most central node. The values inside the nodes and their color represent the betweenness centrality of the node, the darker the node, the higher the centrality.



(a) Path topology.



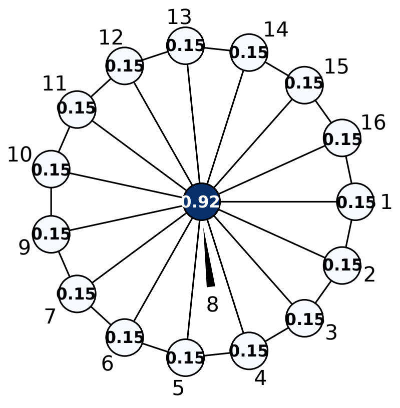
(b) Contribution results per position.

Figure 4.7: Global contribution of nodes in the path topology using 16 nodes. Global contribution measures how many times that node updated the known solution by the system. The 1st position represents an extreme node and the 8th position represents a middle node (the remaining positions have analogous results). The first result is centrality and global contribution are related and the second result is that the global contribution becomes more equally distributed when social context is available.

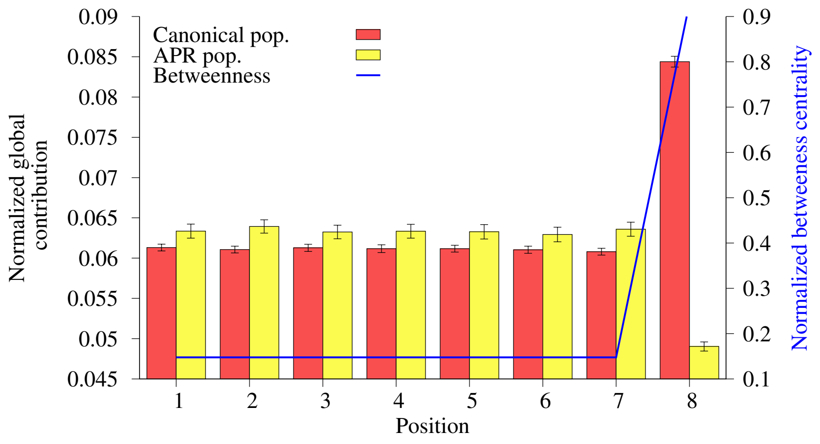
The main result from this experiment is that the node's betweenness centrality and its global contribution are related. The more central the node is, the more likely it its for that node to contribute. This result was expected and it gives additional evidence that centrality is a key property to predict how the contribution of individuals in collective problem solving(BONACICH, 1987; BURT, 1995). As we hypothesized, the second result is that the global contribution becomes more equally distributed when social context is available. We can observe that the extreme nodes in the APR model have contributed much more in comparison with the extreme nodes of canonical model. Our explanation for this result is that the extreme nodes have only one neighbor, thus, their chance to be a leader is higher than their peer node, which has two neighbors. Since leader nodes do not loose good solutions, they are more likely to contribute than non-leader solutions.

Another minor result is that the confidence intervals increased in size in the APR model, meaning that social context leaves the system more unpredictable. This result is expected since the social context increases the interdependence among the nodes, which, in turn, leaves the system more difficult to predict.

The second experiment used the wheel graph using 16 nodes. Again there were two scenarios, one that used the canonical memetic network and the other use an APR population. We plotted only 8 nodes, where the 8th position represents the central node. The results summarize the behavior observed in the five functions. The methodology was the same from the previous scenario.



(a) Wheel topology.



(b) Contribution results per position.

Figure 4.8: Global contribution of nodes in the wheel topology. The same pattern repeated in the wheel topology: centrality is related to global contribution and the whole population of nodes contributed more when there is social context. Without social context, the very central node contributed most of times and when social context was available, this node contribute less than the average node.

The wheel graph showed more evidently the same pattern that the path graph showed: network centrality is related to global contribution; the more central a node is, the more likely it is to contribute. Nevertheless, when social context information is available, an heterogeneous population makes centrality less important. The central node radically shifted from the most important node for global contribution to the least important node. We believe that the affluence of social context information destabilized the central node by making that node less prone to become a leader node.

Our conclusion from this experiment is that the centrality of a node can not be solely used to predict this node's contribution: we provided evidence that the ability to change and to adapt the search strategy in response to the social context can change the what has been known about the importance of centrality to predict the behavior of a node. Summarizing, centrality can be a good predictor *when* all nodes use the same search strategy.

4.4 Experiment 2: Social Context and Problem-solving Efficiency

A particularly interesting descriptive feature of a problem-solving system is how it improves its best known solution along a search. This convergence information is useful if one is to assess whether the system is prone to stagnation or not. Therefore, measuring the system's convergence characteristics is also a way to measure the impact of social context. This experiment deals with a comparison between convergence and distribution analysis of the canonical, APR and DSR models. We defined the stop criteria as 5000 iterations and used a population of 36 individuals. These indidividuals were arranged in a grid network topology as previosly stated. The first results from this experiments are depicted in Figure 5.3.

The canonical Memetic Network model stagnated too early when compared with the proposed methods. It was not able to minimize any test function after approximately the 250th iteration. Within the parameter interval chosen, the APR and DSR extensions did not stagnated. In all functions, social context allowed the system to converge to better solutions in the long run. Considering that the confidence intervals tell us that the mean of the distributions have 95% of probability to be within the interval and that the intervals were small, we assert that indeed social context improved the performance of the system. The hypothesis behind the APR and the DSR models lead to diversity in the search strategies employed by individuals in the population. When combined, these results add evidence that heterogeneity among the strategies used by problem-solvers benefits the whole system(HONG; PAGE, 2004).

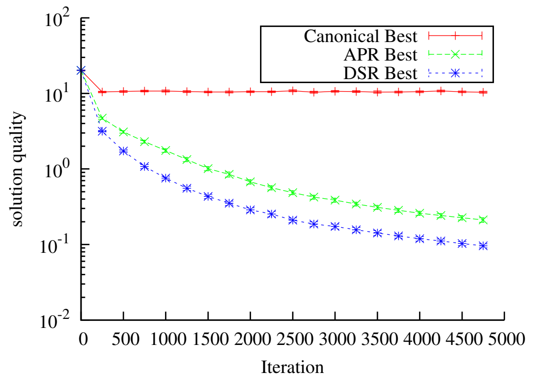
Although these results give us an idea of how the best solution held by a node within the population evolves over time, this information alone can not describe the dynamic of the whole population. Therefore, in order to understand how social context influences the entire population at different search moments, we decided to plot and analyze the complete population distribution at different iterations. We considered the 10th iteration as the initial stage and the 3000th iteration as the final one. We then extracted the entire population of 36 nodes from the 50 runs — amounting 1800 data points to visualize the population distribution through boxplots. A box is drawn around the region between the first and third quartiles, with a horizontal line at the median value. Whiskers extend from the box to any value that lies within 1.5 times the interquartile range.

Figure 4.10 shows the distribution at the initial stage.

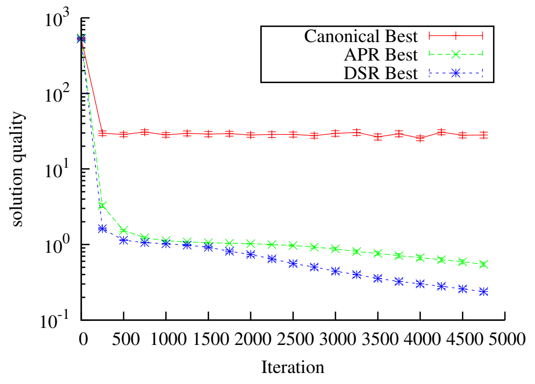
The main conclusion from these results is that at initial stages, there is hardly any differentiation among the models. This happens because ten iterations is a small value for us to observe the convergence of the model and also because nodes are initialized using a uniform random distribution across all search space. The DSR model slightly shifts the population downwards because its median individual is under all other medians.

Figure 4.11 depicts the distribution at the final stage.

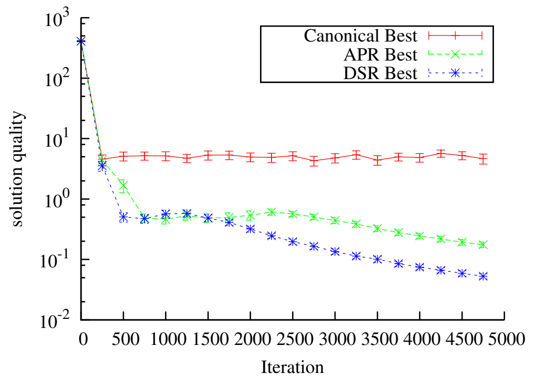
The DSR model is prone to reduce diversity, as one can observe from the data. Although the DSR model brought heterogeneity at the strategy level, it reduced diversity at the solution quality level when compared to the canonical and the APR model. An explanation for this fact is that nodes in the DSR model are too sensitive to their peers performance, which in turn makes them less independent of the results their neighbors possess. For instance, when a DSR node is performing well (its SRP is near or equal to 0.0), it will change its solution by a small factor, holding a new and likely similar solution. Since aggregation is performed towards the best performing neighbors, over time, the peers of that node will possess most of its memes and yet searching far away for better memes.



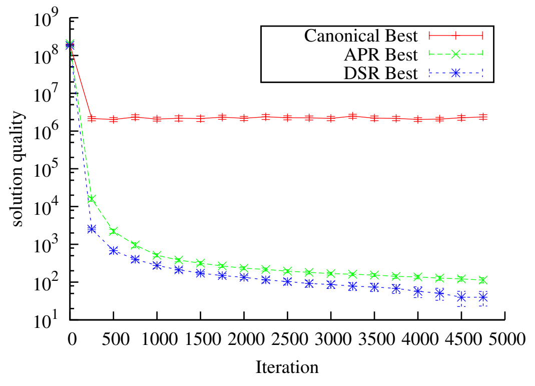
(a) Ackley function.



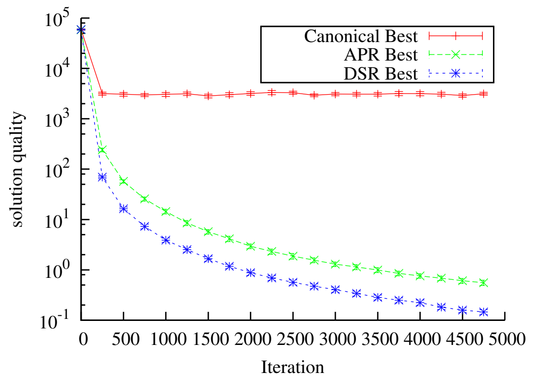
(b) Griewank function.



(c) Rastrigin function.

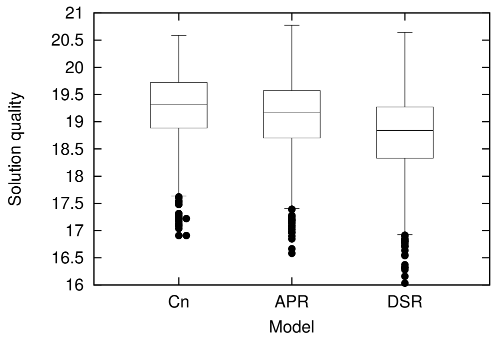


(d) Rosenbrock function.

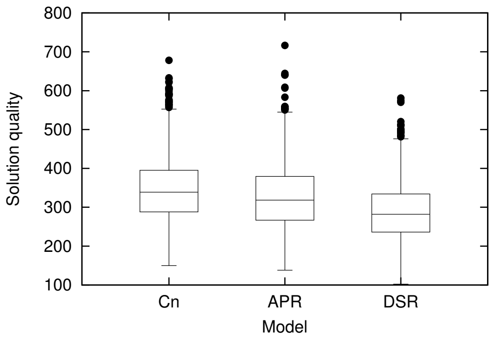


(e) Sphere function.

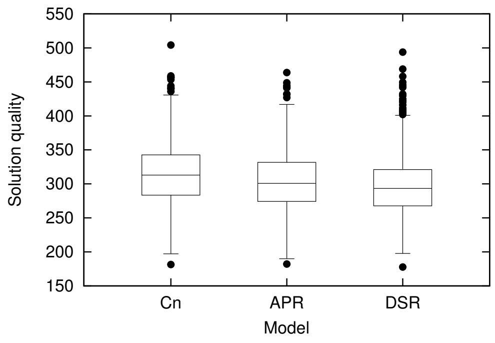
Figure 4.9: The evolution of the best solution held the population in the canonical Memetic Networks, the APR model and the DSR model.



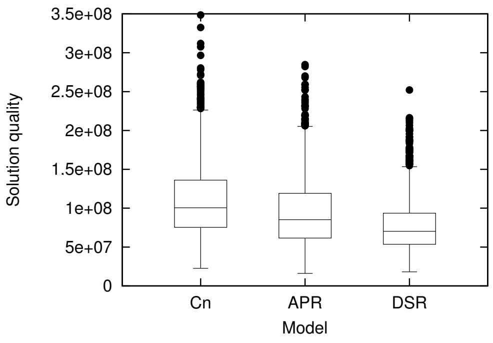
(a) Ackley function.



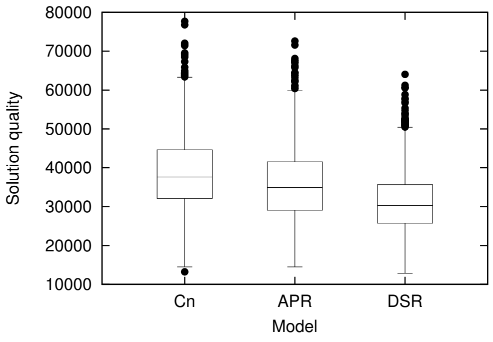
(b) Griewank function.



(c) Rastrigin function.

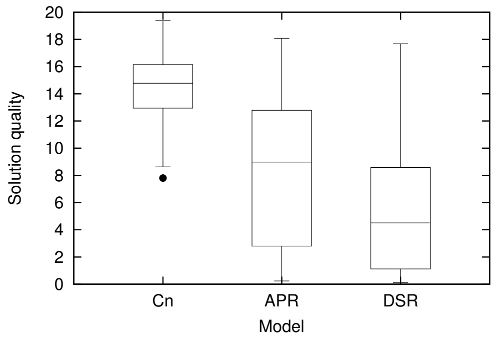


(d) Rosenbrock function.

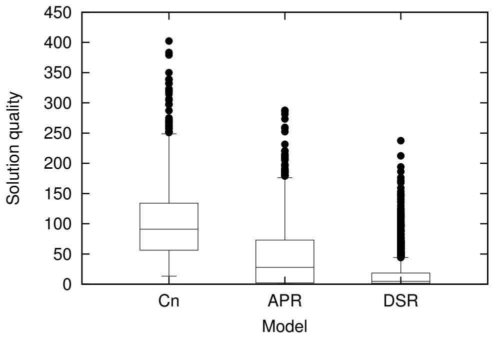


(e) Sphere function.

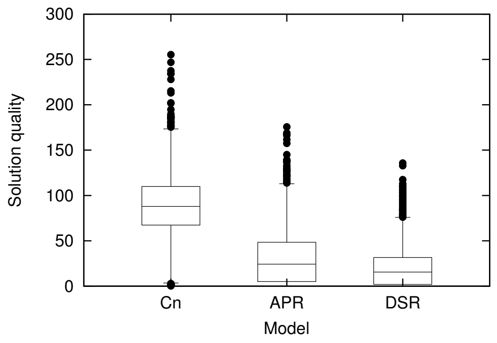
Figure 4.10: Boxplots describing the population distribution right after the 10th iteration for the three models. There is hardly any differentiation among the models at the initial stage.



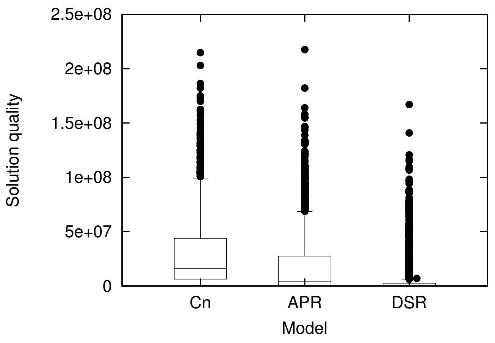
(a) Ackley function.



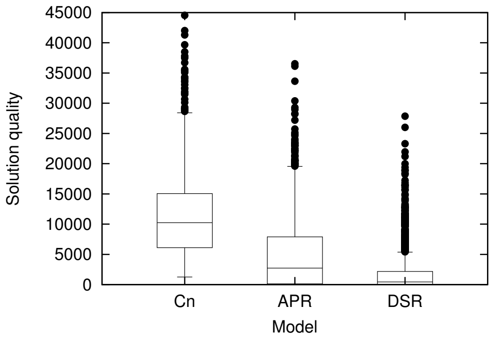
(b) Griewank function.



(c) Rastrigin function.



(d) Rosenbrock function.



(e) Sphere function.

Figure 4.11: Boxplots describing the population distribution right after the 3000th iteration for the three models. The DSR model eliminated solution diversity because individuals tend to have similar solutions. This pattern can be seem more clearly in functions Sphere, Rosenbrock and Griewank.

Other results are that the APR model induces similar solution diversity when compared to the canonical model — this result was expected since the APR model does diverge too much from the canonical model — and that the population distribution of the canonical model is the most predictable and homogeneous one — this result can be explained by the fact all that nodes in the canonical model follow the same rules rendering the population some homogeneity in solution and diversity when centrality is the same among nodes.

The overall conclusions from this experiment is that use of social context information can improve the convergence speed of the group to good solutions and that diversity in search strategies does not necessarily translates into diversity of solutions.

4.5 Experiment 3: Heterogeneous Populations and Social Context

The use of heterogeneous populations of problem-solvers is relatively new and presents itself as an alternative to hybridization. Hybridization typically combines search operators from traditional methods into one method. If the hybrid system is devised to perform search using a population of collaborative search processes, then this population is usually composed of the same methods, in other words, it is homogeneous. Scientists have began to study heterogeneous problem-solving systems recently(OCA et al., 2009; ENGELBRECHT, 2010) though these are still strictly related to the PSO model and they seek to contributed to the design of more efficient generalist heuristics.

In this experiment, we seek to understand the behavior of a heterogeneous population of social agents. So far, we have studied heterogeneous populations that changed their search strategy through a parameter the measured social context. In previous cases, the agents used the Social Relative Performance as the input parameters to the algorithms implemented within the appropriation rule. In other words, we want to know what happens when APR, DSR or canonical nodes are collaborating in the same population. This way, we brought heterogeneity to a new level.

We intuitively believe that, in a heterogeneous population, some strategies will become more adapted to solve the problem than the others. Thus, to test our hypothesis, we designed an experiment with an equally distributed population of individuals and counted how many times a the n^{th} position in the ranking. We used twelve canonical nodes, twelve APR nodes and twelve DSR nodes. We used this problem-solving strategy distribution because it does not make any assumptions about other than that strategies are equally distributed along the population. We used the grid network topology and sampled the ranking in the initial search stage (before any iteration, and after the 10th iteration), intermediate state(the 500th iteration) and final stage(10000th iteration). Results are depicted from Figure 4.12 to Figure4.14. The rank position number *1* represents the best position while the position number *36* represents the worst ranked node. The vertical axis represents the percentage of times that the respective ranking was occupied by a certain kind of node. Nodes were arranged randomly into the grid.

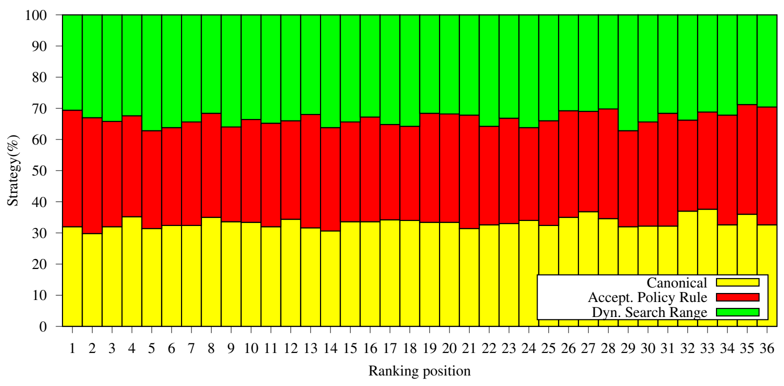
At the very beginning of the search process we can not see a clear predominance of an strategy over the other. We believe this happen because there is not much time for nodes to process any information at all, so they tend to distribute themselves equally across the ranking positions. However, after the tenth iteration, there is a slightly shift in the population's distribution: DSR nodes tended to occupy the very highest rankings while the canonical and APR nodes do not differ considerably. The fact nodes that nodes that can use social context to adapt how far they can see in the search space grants them initial advantage, we believe that DSR nodes tend do find good memes earlier than other nodes and these nodes have not been propagated yet.

Figure 4.13 show the ranking distribution after the 500th iteration, the intermediate stage:

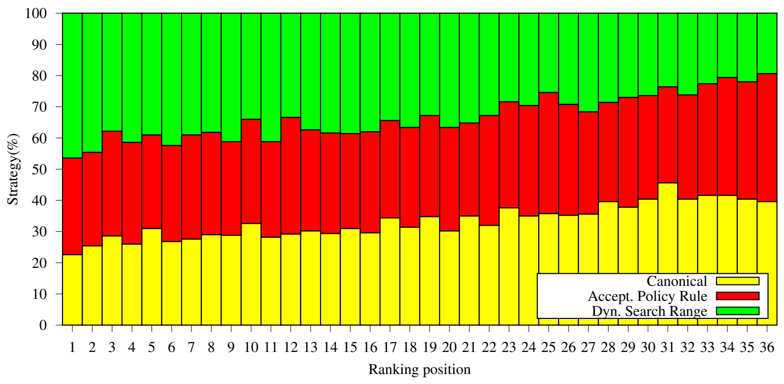
At intermediate stages, we see a modest reversal in the pattern: APR nodes start to differentiate from the canonical nodes by occupying better ranking positions. We attribute this to the fact that good memes have already started flowing in the social network, increasing the chance of APR nodes to possess good memes, and, consequently, good solutions. DSR nodes start to loose the likelihood of occupying the first ranks due to their inability to retain good memes.

Figure 4.14 presents the ranking distribution in the the final stage.

In the final stage, the ranking distribution remained the same. The DSR nodes occu-



(a) Initial ranking distribution.



(b) Ranking distribution after the 10th iteration.

Figure 4.12: In the initial stage, the population is more of less equally distributed with no clear pattern in distribution other than an uniform distribution. The rank position number *1* represents the best position while the position number *36* represents the worst ranked node. The vertical axis represents the percentage of times that the respective ranking was occupied by a certain kind of node.

pied the highest positions while APR nodes tended to be distributed at the best and worst ranking positions while DSR and canonical more or less equally at intermediate positions. Canonical nodes performed worst in the long run.

The main result from this experiment is that even if social context information is the same for all nodes, the way that those nodes react to it is relevant for us to predict their behavior along the search process.

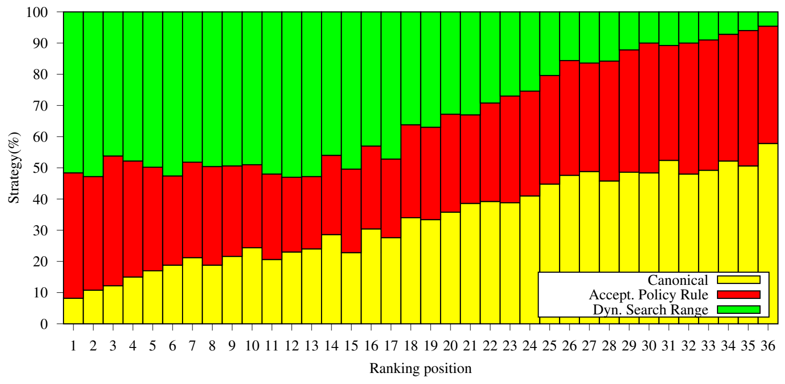


Figure 4.13: Ranking distribution after the 500th iteration. Int the intermediate stage, there is an emerging pattern in population distribution: APR nodes start to differentiate from the canonical nodes by occupying better ranking positions.

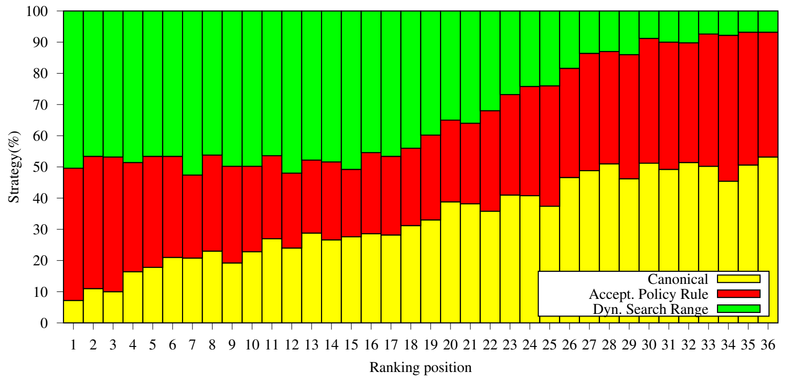


Figure 4.14: In the final stage, the population kept almost the same distribution from the intermediate stage. The DSR nodes occupied the highest positions while APR nodes tended to be distributed at the best and worst ranking positions while DSR and canonical were distributed more or less equally at intermediate positions. Canonical nodes performed worst in the long run.

4.6 General Discussion

We investigated the impact of social context in three cases and we found evidence that social context indeed impacts the system at the individual and at the collective level. To assess the individual impact, we measured how the expected contribution of a node is affected by social context; and to assess the collective impact, we measured the evolution of the whole system towards the best solution as well as the solution diversity within in the population. And we also studied which mechanisms tended to perform better than others in a heterogeneous population.

The first experiment confirmed the common belief that centrality is a relevant measure to characterize nodes in social system. In the case we investigated, we found that the betweenness centrality of a problem-solving node is directly related to its contribution. However, when nodes can adapt their search strategy according to their social surroundings, we found that the (1) expected contribution can be more uniformly distributed in population and that (2) highly centralized nodes can perform worse then nodes averagely centralized nodes.

The second experiment showed the consequences of social context in the whole system. We found that both APR and DSR nodes improved the system's performance At the problem-solving by allowing them to converge faster to better solutions when compared to the canonical model. Another finding was that, although the DSR model improved when compared to the canonical model, it was prone to reduce solution diversity.

The third experiment investigated how a heterogeneous system composed of DSR, APR, and canonical nodes would behave. These nodes were equally distributed in the network, and we measure how many many times a given global ranking position was occupied by a kind a node. We found that DSR performed best and therefore occupied the first positions while canonical nodes occupied the last positions. The main result was that the way that nodes adapt their search strategies is relevant to predict their outcome in the search process.

We now proceed to the conclusions chapter, where we give and overview of this dissertation, focusing on the contribution of this investigation.

5 CONCLUSIONS

This dissertation presented an experimental investigation of the impact of social context in social problem-solving. We elaborated on the topic of social context, and then designed and performed experiments to assess what were the implications of social context at the individual and at the collective level. We devised a method to measure social context and designed two reactive mechanisms to social context.

By using the Memetic Networks model as the fundamental framework and framing the problem to real-valued function minimization, we defined social context as any information that a memetic node receives and sends that is not part of a solution to the problem, that is, everything that is not a meme. After elaborating on this definition and listing some potential interpretations, we opted for a simple yet expressive form of social context: the relationship between a node's number of neighbor nodes faring better against the total number of neighbors. We adapted this measure into a linear equation which result we called as the *Social Relative Performance* (SRP) of the node. The SRP of a node may range from 0.0 to 1.0 and is updated at each iteration. A SRP of 0.0 means that the node has no better faring neighbor while a SRP of 1.0 means that the node is the worst among his neighbors.

After defining social context in Memetic Networks, we elaborated on reactive mechanisms to social context. These mechanisms used the SRP metric to adapt the problem-solving strategy employed by nodes. The first and most simple mechanism was the *Acceptance Policy Rule* (APR), this mechanism defines that a node that has no better ranked neighbor can accept the result they appropriate in case this result improves their current solution or, otherwise, reject the result. The second mechanism, the *Dynamic Search Range* (DSR) employs the SRP to define the search range of the node. The mechanism is built on the assumption that a node with high SRP may not risk loosing potentially good memes during the appropriation step while a low SRP node may use a more risky strategy by searching more distantly for new memes. Both mechanisms when applied to the whole population generate the APR and the DSR models. We also give supportive evidence that social context information can be used to improve the convergence speed of the group to good solutions and that even if nodes perceive social context in same way, the way they react to it may lead to different outcomes along the search process.

Our main methodology was to compare the developed mechanisms with the original Memetic Network model. Among the experiments, we investigated the relationship between the betweenness centrality and the social context, finding that the centrality of a node is not always a good predictor of the node's contribution to the whole when this node can change its strategy according to the social context. In another experiment, we studied the impact of social context in the system convergence to good solutions and in solution diversity. We found that social context information can be used to improve the convergence speed of the group to good solutions and that diversity in search strategies does not necessarily translates into diversity in solutions. In the last experiment, we tested the impact of heterogeneity at the individual level by assembling a population of equally partitioned groups of canonical memetic nodes, APR and DSR nodes. We found that in the short run, the DSR nodes tended to occupy the highest social ranks among the entire population. From this experiment we conclude is that even if social context information is same for all nodes, the way that those nodes react to such is relevant for us to predict their behavior along the search process.

Yet, there remains open questions worthy of exploration:

1. Communication noise — What is the role of communication noise in a collective problem-solving system? What if aggregation does not copy the exactly same memes?

2. New network topologies — Investigate more complex networks topologies that resemble real social networks, such as clustered and dynamic small-worlds networks.

3. Strategy learning schemes — Devise methods that can learn from social context in order to improve its solution.

4. Explore social context in other models — Can we generalize our findings with Memetic Networks to the also socially-inspired Particle Swarm Optimization model?

Among the contributions of this dissertation, we cite: the development of the simple metric of social context, the Social Relative Performace (SRP); the APR and the DSR mechanisms whose objective was to allow agents to change their search strategies. Nevertheless, the main contribution of this dissertation is to provide strong evidences that social context impacts social problem-solving in a simulated setting. Through empirical experiments, we showed that the social environment of a problem solver must be considered in forthcoming experiments with social problem-solving experiments that use both human and artificial agents.

The parallel investigations during the master course lead other contributions, which were: the understanding that the Memetic Networks model is not suitable for low dimensionality search spaces; the investigation of the effects of heterogeneity in social-problem solving; and, the feasibility of using the DSR model to solve the problem of protein structure prediction. A summary of these can be found in the appendix.

APPENDIX

5.1 O Impacto do Contexto Social em Redes Meméticas

Sistemas capazes de resolver problemas por si só são usualmente considerados sistemas inteligentes (NEWELL; SIMON, 1976). Boa parte do esforço inicial dos pesquisadores de Inteligência Artificial se focou em tais sistemas e, sem surpresas, o principal sistema inteligente estudado nesta época fora o ser humano. O objetivo destes pesquisadores era de encontrar pistas sobre a forma com que humanos resolviam problemas a fim de que estas levassem ao desenvolvimento de sistemas computacionais robustos e genéricos o suficiente para resolver problemas diversos. A premissa de que humanos são exímios resolvedores de problemas em geral se mostrou falsa: um dos principais resultados do estudo foi a constatação de humanos são eficientes em algumas classes de problemas e péssimos em outras (RUSSELL; NORVIG, 2010). O campo de estudo responsável pelo desenvolvimento de sistemas que combinam e exploram as as vantagens humanas e de sistemas computacionais em um único sistema híbrido é atualmente conhecido como Computação Humana (AHN et al., 2008).

Recentemente, o estudo de sistemas inteligentes tem se focado no aspecto social destes sistemas, mas com um novo objetivo: facilitar a compreensão da dinâmica social humana (WANG et al., 2007; LAZER et al., 2009). Tal estudo é mais conhecido como Ciência Social Computacional e se apresenta como uma ramificação teórica da Área de Computação Social.

Quando resolvemos problemas em grupo, usualmente empregamos estratégias que mudam ao longo do tempo e que também dificilmente se repetem dentro do grupo (MASON; WATTS, 2012). Uma explicação para essa diversidade e dinamicidade de estratégias está na hipótese de que nós adaptamos nossas estratégias de acordo com indícios internos e externos (FISKE; TAYLOR, 1991; TAYLOR, 1998; SCHWARZ, 1998). Estes indícios são como variáveis internas e externas ao indivíduos que interferem na forma em que o problema é representado e na forma com que a solução é investigada. Entre os indícios de natureza interna, estão o nosso estado psicológico, nossas motivações, nossa disposição, etc. Já entre os de natureza externa, estão os recursos físicos disponíveis, o tempo, o ambiente social, entre outros.

Dentre os fatores externos, o ambiente social, ou contexto social, é um dos que tem maior influência na forma que pensamos e agimos (VALSINER; VEER, 1988; LEVINE; RESNICK, 1993; BURT, 1995). Entretanto, até então, os modelos de resolução social de problemas têm sido insuficientemente sofisticados e heterogêneos para capturar a diversidade observada em experimentos com seres humanos (MASON; WATTS, 2012): se faz necessário, portanto, que se investigue qual é o impacto do contexto social da resolução social de problemas.

Nesta dissertação, nós propomos a investigação do impacto do contexto social na resolução social de problemas. Para tal, investigaremos o conceito de contexto social dentro do modelo computacional das Redes Meméticas (ARAÚJO; LAMB, 2010). Este modelo foi escolhido por originalmente considerar a população como homogênea quanto às estratégias de resolução de problemas e por modelar computacionalmente a forma com que humanos resolvem problemas em grupos. Além disso, o modelo original servira como controle para que pudéssemos comparar um sistema que contempla o uso do contexto social e outro que não.

O contexto social pode ser definido como o conjunto de relações entre grupos de pessoas (BARNETT; CASPER, 2001). Dentro do modelo das Redes Meméticas, o contexto social pode ser interpretado como qualquer informação interna ao sistema que não faz parte da solução em si — qualquer informação que não é um meme. Dentre as várias interpretações possíveis para o contexto social dentro de redes meméticas, nós escolhemos a que nos pareceu a mais simples: o desempenho social relativo, ou “Social Relative Performance” (SRP). O SRP define o quão boa é uma solução quando comparada às soluções de vizinhos na rede social. O SRP decresce linearmente em relação à qualidade de respota do nó, quanto maior a qualidade da resposta quando comparada aos nodos vizinhos da rede, menor é o SRP.

Para que os indivíduos da rede considerassem o contexto social durante a resolução dos problemas, nós adaptamos o modelo original das Redes Meméticas para que ele acomodasse mecanismos de reação ao *Social Relative Performance* (SRP). A partir da definição do SRP, nós projetamos dois mecanismos reativos ao contexto: o *Acceptance Policy Rule* (APR) e o *Dynamic Search Range*(DSR). O primeiro mecanismo muda a regra de aceitação de novas resposta; o segundo, modifica a taxa de exploração e intensificação da busca em resposta o SRP. Ambos mecanismos foram comparados com o modelo original já que este não oferece suporta o contexto social — logo, não apresenta alguma heterogeinidade de estratégias na população.

Para que pudéssemos averiguar o impacto do contexto social na resolução coletiva de problemas, decidimos realizar dois experimentos principais que foram: identificar (1) a correlação entre centralidade e o contexto social e (2) o impacto do contexto social na eficência coletiva. O problema a ser resolvido em todos experimentos foi o problema de minimização de funções contínuas. Para todos experimentos, foram usadas cinco funções de avaliação: *Ackley*, *Griewank*, *Rastrigin*, *Rosenbrock* e *Rastrigin*. Estas funções são comumente usadas na investigação e verificação de algoritmos de otimização baseados em sistemas naturais e se distinguem em termos de número de mínimos locais e do grau de dificuldade. A minimização foi feita em 30 dimensões para todas as funções. Os experimentos também fora repetidos 50 vezes para que pudéssemos descrever o comportamento esperado do algortimo dentro de um intervalo de confiança de 95%.

Os resultados experimentais serão apresentados nas seções a seguir.

5.1.1 A Centralidade e o Contexto social

Podemos caracterizar um indivíduo em uma rede social pela sua posição na rede (**?**). Usualmente, indivíduos que ocupam posições centrais na rede são tidos como indíviuduos mais relevantes àquela rede. Logo, a centralidade de um indivíduo dentro de uma rede é uma informação importante. Uma forma de se medir o grau de centralidade de um indivíduo é através de sua centralidade de intermediação (do Inglês *betweenness centrality*). A centralidade de intermediação de um indivíduo é dada pelo número de caminhos mínimos entre quaisquer dois outros indivíduos que passam pelo indivíduo em questão.

Neste experimento, estamos interessados em medir o impacto do contexto social em indivíduos de diferentes posições sociais. Para tal, serão avaliados os resultados contribuição individual do indivíduo por posição após a quinta milésima iteração. A contribuição individual aqui representa o número de vezes que um indivíduo contribui com a melhor solução conhecida pelo sistema. Serão usadas duas topologias de rede com propriedades distintas: a primeira topologia é a topologia *path*, que é construída a partir de uma topologia em anel onde uma aresta é aleatoriamente selecionada e removida; a segunda topologia é a topologia *wheel*, onde a partir de uma topologia de anel, um indivíduo é escolhido aleatoriamente e conectado a todos os demais indivíduos. Na topologia path, a centralidade por intermédio cresce conforme o indivíduo se afasta das extremidades da rede. Na topologia wheel, o indivíduo escolhido para se conectar aos demais é o indivíduo mais central da rede enquanto os demais indivíduos possuem a mesma centralidade.

Ambas topologias foram instanciadas com 16 indivíduos. Foram testados e comparados dois grupos em cada topologia: um grupo de controle, que apresentava reação ao contexto social (a forma canônica das Redes Meméticas); e um grupo de teste com indivíduos que implementavam o mecanismo APR. A figura 5.1a mostra a topologia path usada onde os valores dentro de cada nó representam o valor de centralidade de intermédio normalizada e os valores fora dos nós representam um índice do nó. Estes índices são usados para indicar a contribuição do nó na figura 5.1b. Como a centralidade se distribui de forma simétrica entre os indíviduos ao longo da rede, apenas a metade dos indivíduos foi representada para facilitar a visualização.

Os dados suportam que (1) a centralidade e a contribuição de um indivíduos estão de fato bastante relacionadas e que (2) a adaptação ao contexto social faz com que a contribuição esperada seja distribuída de forma mais uniforme. Além disso, o contexto social fez com que a contribuição esperada de um indivíduo ficasse dentro de um intervalo maior.

A figura 5.2a mostra a topologia wheel e a figura 5.2b mostra a contribuição individual.

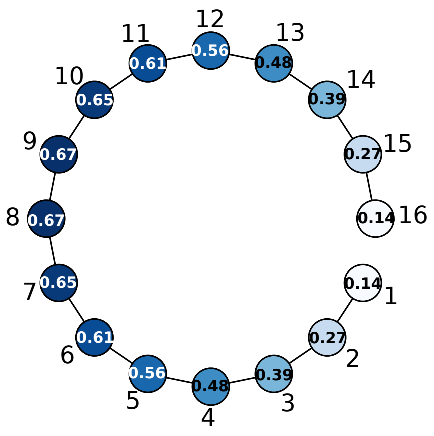
No grupo de controle, a centralidade está relacionada diretamente com a contribuição do indivíduo enquanto esta relação se inverte no grupo de teste, que considera o contexto social. A contribuição do nó central no modelo APR foi, inclusive, inferior a contribuição nós periféricos.

5.1.2 A eficiência coletiva e o contexto social

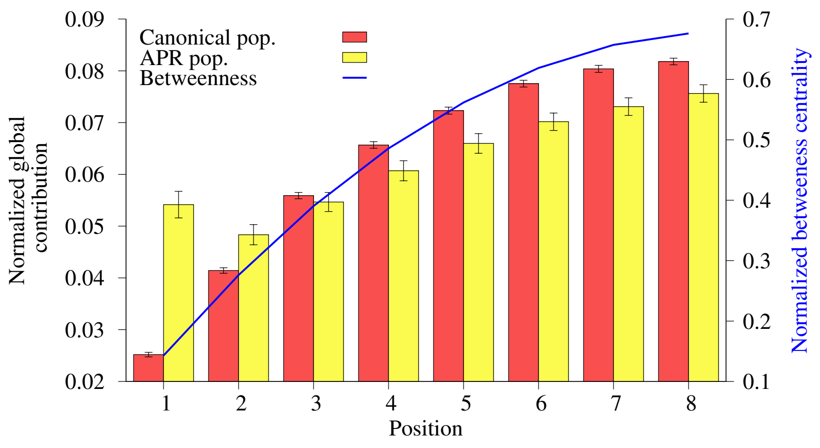
Uma característica descritiva de um sistema de resolução de problemas é a relação entre a qualidade da melhor solução conhecida pelo sistema e o como essa qualidade muda ao longo do tempo. Esta relação mostra o quão rápido o sistema convergiu para boas soluções. Além disso, esta relação aponta se o sistema tende a estagnar a busca pela solução do problema ou não. Portanto, medir a convergência do sistema podes nos dizer qual o impacto do contexto social no sistema, ou seja, no coletivo.

Para tal, elaboramos um experimento onde comparamos a convergência do sistema dos modelos APR e do modelo DSR. O critério de parada foi definido como 5000 iterações com uma população de 36 indivíduos. Este indivíduos foram posisionados aleatoriamente em um grid períodico de tamanho 6$\times$6. A jusfiticativa para a escolha desta topologia está no fato de que, nela, todos os indivíduos possuem a mesma centralidade o que nos permite isolar os efeitos da centralidade no desempenho coletivo e individual. Os resultados deste experimentos estão expostos na figura 5.3.

O modelo original das Redes Meméticas mostrou uma tendência à estagnação já nas



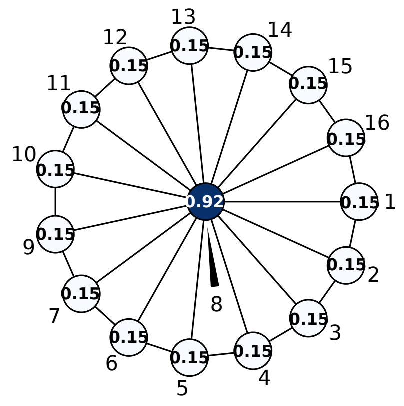
(a) Topologia *path*.



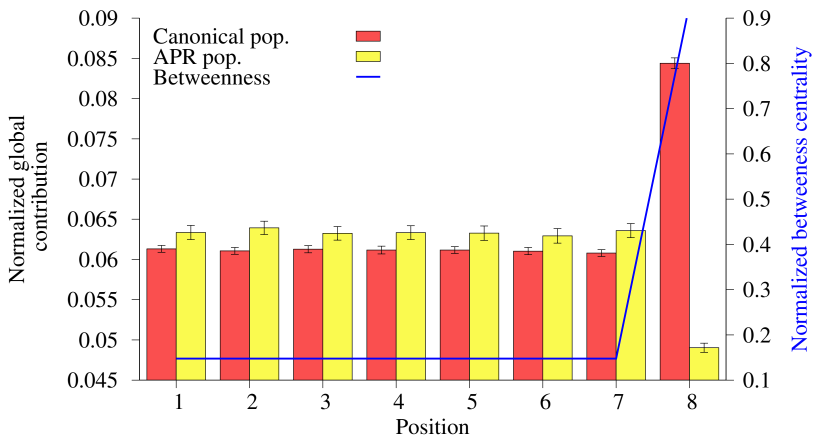
(b) Contribuição individual.

Figure 5.1: A contribuição individual média na topologia path. A contribuição mede o número de vezes que um indivíduo atualizou a melhor solução conhecida pelo sistema social. A primeira posição representa um indivíduo um uma extremidade da rede enquanto que a posição representa um indivíduo no centro da rede. As barras verticais representam um intervalo de confiança de 95%. Os dados suportam que (1) a centralidade e a contribuição de um indivíduos estão de fato bastante relacionadas e que (2) a adaptação ao contexto social faz com que a contribuição esperada seja distribuída de forma mais uniforme. Além disso, o contexto social fez com que a contribuição esperada de um indivíduo ficasse dentro de um intervalo maior.

primeiras iterações — após aproximadamente 250 iterações para todas funções. Considerando os parâmetros usados, os modelos APR e DSR não apresentaram a mesma tendência à estagnação. Para todas funções, o contexto social permitiu que o sistema encotrasse soluções melhores em relação ao modelo original.



(a) Topologia *wheel*.



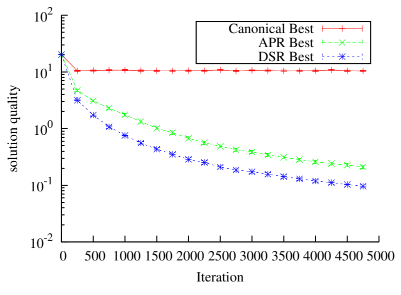
(b) Contribuição por posição.

Figure 5.2: Contribuição individual média na topologia wheel. A posição 8 representa o nó central e as barras verticais representam um intervalo de confiança de 95%. No grupo de controle, a centralidade está relacionada diretamente com a contribuição do indivíduo enquanto esta relação se inverte no grupo de teste, que considera o contexto social. No modelo APR, a contribuição do nó central foi inferior a contribuição nós periféricos.

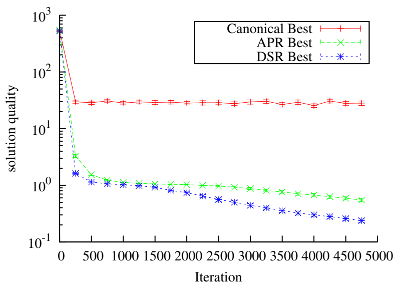
5.1.3 Conclusões

Este trabalho teve, como objetivo principal, a busca pela compreensão do impacto do contexto social sobre o indivíduo e sobre o coletivo em um sistema social de resolução de problemas. Para tal, adaptamos um modelo já existente para que ele acomodasse uma forma simples de contexto social, o *Social Relative Performance* (SRP). A partir da definição do SRP, nós projetamos dois mecanismos reativos ao contexto: o *Acceptance Policy Rule* (APR) e o *Dynamic Search Range*(DSR). O primeiro mecanismo muda a regra de aceitação de novas resposta; o segundo, modifica a taxa de exploração e intensificação da busca em resposta o SRP. Ambos mecanismos foram comparados com o modelo original já que este não oferece suporta o contexto social — logo, não apresenta alguma heterogeinidade de estratégias na população.

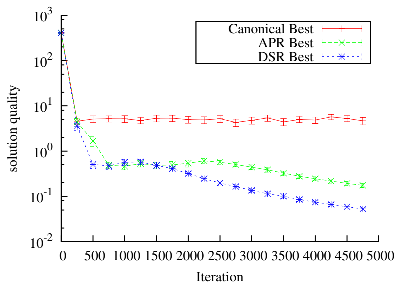
Através de três experimentos, avaliamos o impacto do contexto social no aspecto individual e coletivo. Mais precisamente, nós mostramos que a centralidade de um indivíduo na rede social nem sempre é um bom preditor de sua contribuição quando o mesmo pode adaptar sua estratégia de busca em resposta ao contexto. Além disso, mostramos que



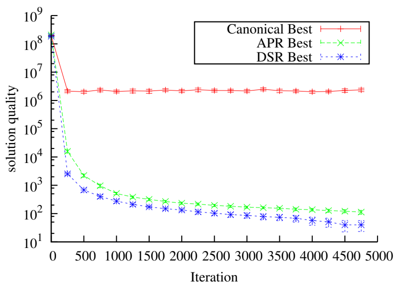
(a) Função Ackley.



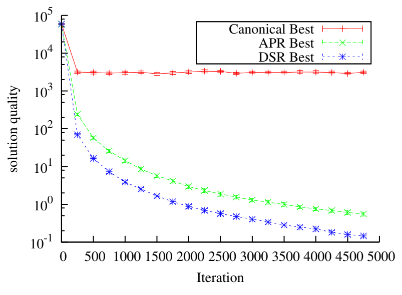
(b) Função Griewank.



(c) Função Rastrigin.



(d) Função Rosenbrock.



(e) Função Sphere.

Figure 5.3: Evolução da melhor solução conhecida pela pela população no modelo original das redes meméticas, no modelo APR e no modelo DSR. As barras verticais representam um intervalo de confiança de 95%.

a adaptação ao contexto social, por parte dos indivíduos, pode melhorar o desempenho coletivo, facilitando a convergência para soluções boas; que a diversidade de estratégias de resolução do problema não leva necessariamente a uma diversidade de soluções na população; e que, mesmo que o contexto social seja percebido da mesma forma pelos indivíduos, a forma com que eles reagem pode levar a resultados diferentes. Todos estes resultados contribuem para o entendimento de que o contexto social é relevante à resolução social de problemas e deve, portanto, ser considerado em experimentos relacionados.

5.2 Published Works

This chapter presents a list of published works that were written during the master course. These works represent checkpoints of the ongoing and parallel investigations carried during the master course. Table 5.1 describes the year, conference name, and some bibliometric information.

Table 5.1: List of Publications in Conference Proceedings.

Year	Title	Conference	Qualis	H-Index
2011	Two Novel Algorithms for High Quality Motion Estimation in High Definition Video Sequences	SIBGRAPI	B1	28
2012	Modeling Adaptative Social Behavior in Collective Problem Solving Algorithms	SASO	B3	15
2012	A memetic Network-based Approach to Search the 3-D Protein Conformational Space (**Best Poster Award**)	X-Meeting	B4	8
2013	Investigating a Socially Inspired Heterogeneous System of Problem-solving Agents (**accepted**)	AINA	A2	39

The first parallel investigation we pursued was to *determine the applicability* of the Memetic Networks model in the problem of motion estimation. Motion estimation is a numerical optimization problem performed in 2 dimensions whose objective is to reduce a kind of data redundancy found in most digital videos: the temporal redundancy. We instantiated an algorithm called MNA-ME to solve it and found that our algorithm performed poorly in terms of solution quality and number of calculations when compared to classical benchmark algorithms. We concluded that model of Memetic Networks does not perform well in the problem of motion estimation that it is likely perform as so in problems modeled in a few dimensions. The whole exploration was published on the Proceedings 24th Sibgrapi Conference on Graphics, Patterns and Images (SIBGRAPI-2011) (NOBLE et al., 2011).

The second parallel investigation studied *the effect of hybridization at the population level* of two social models of problem-solving: the Memetic Networks model and the Particle Swarm Optimization model. Hybridization is usually studied at the individual level, where a homogeneous population of individuals share search operators from two or more models. In this investigation, we studied the effects of hybridization at the population level, i.e. we placed individuals from different models into the same population. We found that the heterogeneous model outperformed the two original model in the Rastrigin and in the Ackley functions. Otherwise, it performed as well as the best performing homogeneous model. The results from this investigation were accepted for publication at the 27th IEEE International Conference on Advanced Information Networking and Applications (AINA-2013) (NOBLE; ARAÚJO; LAMB, 2013).

The third parallel investigation *studied the feasibility of using the canonical Memetic and the DSR mechanism* in the problem of Protein Structure Prediction (PSP) bioinformatics. In this problem, the model must find the right angles between the chain of aminoacids that compose a protein. We modeled the problem the as a problem of minimization of real-functions and used, as objective function, a conformational energy function. We found preliminary evidence that the DSR mechanism improved over the canonical Memetic Networks in that domain. The results were reported at the 5th International Conference of the Brazilian Association for Bioinformatics and Computational Biology (X-meeting 2012) (NOBLE; DORN; LAMB, 2012).

We attached the three papers published or accepted for publication that were written during the master course.

Two Novel Algorithms for High Quality Motion Estimation in High Definition Video Sequences

Diego Noble*, Marcelo Porto**, Luciano Agostini**, Ricardo M. Araujo**, Luis C. Lamb*

*Institute of Informatics, Federal University Rio Grande do Sul
Porto Alegre, RS, Brazil
<dvnoble,lamb>@inf.ufrgs.br

**Technological Development Center, Federal University of Pelotas
Pelotas, RS, Brazil
<mporto,agostini,ricardo>@inf.ufpel.edu.br,

***Abstract*—In this paper, we propose two new algorithms for high quality motion estimation in high definition digital videos. Both algorithms are based on the use of random features that guarantee robustness to avoid dropping into a local-minimum. The first algorithm was developed from a simple two stage approach where a random stage is complemented by a greedy stage in a very simple fashion. The second algorithm is based on a more refined class of algorithms called Memetic Network Algorithms where each instance of the search may exchange information with its neighbour instances according to some rules that control the information flow. The proposed algorithms were implemented and tested exclusively with high definition sequences against well known fast algorithms like Diamond Search and Three Step Search. The results show that our algorithms can outperform other algorithms in quality yielding an increment in complexity that may be amortized if resources for a parallel execution are available. Additionally, we provide further evidence that fast algorithms do not perform well in high definition.**

***Keywords*-Motion Estimation; Video Coding; High Definition**

I. Introduction

Most modern video coding standards like H.264/AVC, for example, use block motion estimation (BME) together with motion compensation (MC) as a way to reduce or eliminate temporal redundancy in uncompressed video streams [1]. By doing so, BME and MC account for a large proportion in the reduction of the bit rate. However, this technique also represent the most computationally complex module of the coder [2]. In fact, motion estimation can consume up to 60% of the total encoding time of the H.264/AVC codec when just one reference frame is used [3].

To tackle this complexity, several fast algorithms have been proposed. Such algorithms rely on heuristics and metaheuristics to guide how the sampling of the solution space is conducted. Some well known representatives of this class of algorithms include: The Three Step Search algorithm (TSS) [4], one of the first fast ME algorithms proposed; Diamond Search (DS) [5], which uses two geometrically shaped search patterns, and Hexagon-Based Search (HEXBS) [6].

For applications where low bit rate is a key issue, e.g. cell phones, all these algorithms perform well provided that, in this context, fast and complex motion tends to be infrequently [7]. In this paper, we will present two novel algorithms for motion estimation in High Definition (HD) which were specifically designed to achieve high quality in this context while enjoying comparable computational complexity of fast algorithms previously developed. Furthermore, we will also show evidence that fast algorithms do not produce satisfactory results in face of HD digital video coding. Hence the importance to address inherent features of HD in the development of modern ME algorithms.

This paper is organised as follows: Section II gives an overview of the basics of motion estimation and related work. Section III will introduce in detail our two proposed algorithms. In Section IV, experimental results and their impact are described. Finally, we draw the conclusions and point out future research directions.

II. On Motion Estimation

An uncompressed digital video has many kinds of information redundancy. One of them is temporal redundancy, which is related to the spatial similarity that temporally correlated frames have. This superfluous information is an effect of the temporal sampling of a video where, commonly, 30 frames are condensed in a second so a human viewer can have a feeling of real-time movement. For a digital video to be more efficiently transmitted or stored, one needs to eliminate this information redundancy with within limits of what is acceptable in terms of image quality loss.

The ME/MC module of a block-based video encoder attempts to reduce the temporal redundancy by compensating the motion in a video through translating or warping the samples of the previously transmitted reference frame, which in its turn is likely to very similar to the actual frame yet to be encoded. The resulting motion-compensated predicted frame is then subtracted from the current frame to produce the residual frame [8], which is potentially a sparse matrix that can be more easily coded by the next stages of the coder. The more efficient the ME process is, the more efficient the compression becomes because less residual energy, or error, needs to be encoded and less information should be discarded in the quantization step for a fixed bit rate.

In a block-based video coding scheme, a frame is subdivided into non-overlapping blocks of pixels. These blocks may have sizes of $4 \times 4, 8 \times 8, 16 \times 16$ or any combinations of these. The size of the block used and the possibility of adopting multiple block sizes are completely dependent upon the video standard in question in these choices considerably affect the coding efficiency and complexity of the ME process [9].

The motion vector (MV) for a given block is achieved by conducting an algorithmic search which tries to minimize the value of a matching criterion. The most commonly used and one of the simplest criterion is the sum of absolute differences [2], [8], or SAD, which is calculated from the current block (which comes from the actual frame) and an equally sized reference block (which comes from the reference frame). Equation (1) depicts the SAD formula between two blocks of the same size where *R* represents the reference block and *C*, the current block.

$$SAD(R, A) = \sum_{i=0}^{N} \sum_{j=0}^{N} \left| R_{(i,j)} - C_{(i,j)} \right| \tag{1}$$

The spatial distance (measured in Cartesian coordinates) related to the best of all candidate reference and the current block is kept and it represents the MV. Regularly, the search for the MV is limited by a range in horizontal e vertical axis, called *search window*. This simplification significantly reduces the number of reference blocks to be analyzed. The search window may also be referenced as search range in the sense that a search window of $N \times N$ is equivalent to a search range of $(\pm N/2, \pm N/2)$. In this work, both concepts will be used interchangeably kept the right meaning.

The complexity of a Block Matching-based encoder largely depends upon motion estimation and the rate-constrained control [1]. However, the main goal of ME is to predict the actual frame as precisely as possible so less residual information needs to be processed and then transmitted. These two objectives are contradictory because usually the more precision is required, the more computationally demanding the search becomes. The trade off between these aspects depends on the application.

When precision (and thus objective quality) is the main objective, the Full Search (*FS*) algorithm is the best candidate since it evaluates all the possible reference blocks within the search window. However, due to the very high computational cost that this exhaustive search method demands, practical applications are hindered from using it. Generally, algorithms developed for ME try to maximize PSNR while minimizing the computational effort. It is useful to define the most used objective quality metric [10], the peak-signal-to-noise ratio (PSNR), shown in equation (2), in it *R* represents the reference frame and *C*, the current frame.

$$PSNR = 20 \cdot \log \left(\frac{255}{\sqrt{\frac{1}{N^2} \sum\limits_{i=0}^{N} \sum\limits_{j=0}^{N} \left(R_{(i,j)} - C_{(i,j)} \right)^2}} \right) \tag{2}$$

A. ME in High Definition

In our findings, we compared SAD maps so we could have a visual comprehension on how the solution space would seem like. These maps were built using the Full Search algorithm by plotting SAD values into a Cartesian coordinate system where each point of these maps represent the SAD value of that block. Figure 1 depicts one set of these maps for the same region of the same video in different resolutions. Note that the search window of each map remains proportionally the same and the center of the space solution is represented exactly at the center of a relative map where darker regions indicate better block matchings than brighter areas. The resolution is seen at the bottom of the respective caption.

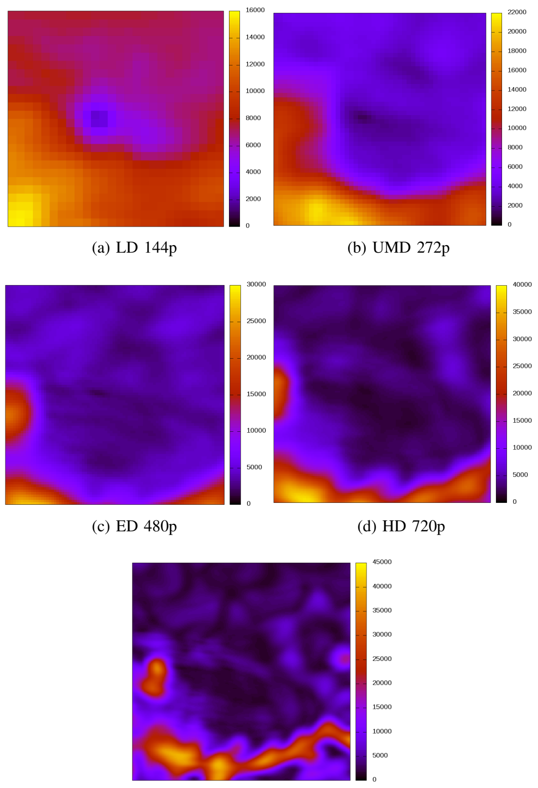
In Figure 1a, the global optima can be clearly seen, near the center. This pattern is somehow common for sequences in this resolution. In an optimization context, a greedy approach would yield very satisfactory results since this search process will converge to a global optima solution most of the time. This hypothesis partially explains why fast algorithms reach almost the same quality performance of full search for low definition sequences.

Nevertheless, for a larger resolution in Figure 1b, two regions can be seen in the map. Both, can potentially hold the global optimum and break the previous pattern. The ME in this map can be considered more difficult than in 1a since one needs to widen the search to better evaluate the solution space. This observation is also valid for an enhanced definition sequence in Figure 1c, where the map gets even rougher and one may not notice with certainty the difference between the global optima and local optima. Estimating motion in high definition gets even harder as can be seen in Figures 1d and 1e where a very complex and rough map with lots of valleys and hills that may intricate the problem even further.

Generally, as resolution grows the motion estimation also becomes more difficult in the sense that more search points may need to be evaluated. Considering the recently available consumer electronics devices such as HD digital video broadcasting [7], this is an evidence that ME in HD may become a central issue in the forthcoming years. To the best of our knowledge, there is a paucity of results about this issue, which requires further research.

B. Related Work

Several algorithms have been proposed to efficiently find motion vectors. All the previously mentioned algorithms could be seen as classical algorithms, because they have influenced many modern algorithms and techniques. They are generally considered fast algorithms because they do not search the



(a) LD 144p (b) UMD 272p

(c) ED 480p (d) HD 720p

(e) HD 1080p

Fig. 1: SAD maps for the block in different resolutions of a same sequence.

entire solution space. As a result, the motion vectors they find are not necessarily the optimum ones.

Three Step Search (TSS) was one of the first algorithms proposed to deal with this problem [4]. The TSS algorithm selects nine search points: one at the center and eight concentrically positioned points at the same distance. This pattern is repeated two times; at each step, the center of the eight points is the best evaluated block from the last step and the distance is divided by a factor of two. This algorithm was very successful and widely adopted in the early stages of video coding [11] due to its simplicity and regularity. The NTSS [12] algorithm is more recent improvement over it.

The Diamond Search Algorithm [5] is a well-known algorithm that led or influenced various other algorithms like the Hexagon Search algorithm (HS) [6] which uses a hexagon shaped geometry to execute the search. In its turn, HS has spawned Unsymmetrical-cross Multi-hexagon-grid Search (UMHexagonS) [3]. The work of [2] proposes the improvement of the UMHexagonS algorithm. In [13], the three-dimensional predict hexagon search (3DPHS) algorithm is proposed. This algorithm uses a rood-shaped search pattern at the fist two searching steps with a higher probability to get motion vectors and it can predict the object movement in horizontal and vertical direction. Most of these algorithms rely on using techniques to improve a common ancestor and in fact should be regarded as algorithmic improvements but not, up to a certain extent, fully original algorithms.

None of these algorithms is focused on high definition or is evaluated as such. In [14], the Dynamically Variable Step Search (DVSS) algorithm is proposed. However, qualitative results in terms of mean absolute differences (MAD) are not carried out in high definition sequences although the hardware architecture proposed is capable of processing 1080p video streams in real-time. In [15], two new random search algorithms are proposed, but their conclusions so far are highly prone to deviate from a HD real case scenario since the evaluation is conducted for only 20 frames of two non-HD sequences.

The need for developing algorithms for high definition ME has only been recently addressed. In [7] the Recursive Dynamically Variable Step Search (RDVSS) ME algorithm for real-time processing of HD video formats is proposed. This algorithm is an improvement over the DVSS algorithm and it dynamically determines the search patterns that will be used for each block based on the MVs of its spatial and temporal neighboring blocks. The algorithm qualitative evaluation is solely performed in high definition videos sequences. Although no quantitative evidence about why fast algorithms perform poorly on HD case studies, the results presented point the same tendency that our qualitative analysis.

III. The Proposed Algorithms

In this section, we will introduce our algorithms and explain each one accordingly.

A. The Random Search Algorithm RS4

Considering that motion estimation is a non-convex optimization problem, randomized techniques are a common tool in this kind of optimization [15]. The sole use of a greedy heuristic to guide the search would not suffice to accurately estimate all kinds of motion. This can be intuitively perceived by analyzing Figure 1e. Following this idea, it is possible to draw that combining these two approaches may have some advantage over a single particular approach. Howerver, we also assume that a completely random search is not efficient since it ignores some common and well-known patterns of motion estimation.

Discovering a new pattern that can be explored to improve the ME is a central issue in the design of motion estimation algorithms. For instance, it is commonly assumed that the matching error will decrease monotonically when approaching the global best point [3]. It is important to know that this assumption does not hold true for video sequences especially for those with large motion content [5]. These statements may seem contradictory but by joining these ideas with random search, we can derive that a previously applied random search step could open some solutions of the whole space and then select the best one for positioning the local step search. This local step search can be guided by a greedy heuristic and may use even a common geometrical search shape.

Assuming that high definition sequences particularly benefit from large search ranges [16], the randomization becomes a very attractive approach and it is based on these observations we propose a novel algorithm: the *Random Search 4* (RS4) algorithm.

The search in RS4 is done in Two Steps: the random step and the iterated local step. Before the execution of the random step, the central search point and its four neighbour points at distance 1 are evaluated. Thus, this step guarantees that the search is executed at the middle where for stationary takes the best candidate is likely to be. Then, with uniform probability N points within the search window are chosen to be evaluated (random step). The best candidate among the $5+N$ is set aside so that its position will be used as the center for the iterative local step. This step step iterates using the SDSP pattern [5] until the stop criteria is reached, that is the new evaluated block presents no advantage over the last best candidate.

B. The Memetic Network Algorithm MNA-ME

Memetic Networks is a class of algorithms that define a population which is able to communicate through a network [17]. This approach has the advantage of being flexible enough to adapt to many problems of optimization. Differently from other multi-agent based algorithms, the performance of a Memetic Network is strongly affected by the way the agents are connected, how they exchange information and how this information affects their current context. The Memetic Network model can be seen as a model of cultural evolution in the sense that a *meme* (that is a piece of information) may spread through a population in a fast manner when compared to a *gene* in a Genetic Algorithm (GA). An instance of a memetic network is created from three well-defined rules where each rule applies equally to all agents:

I *Connection Rule*: tells how the agents should connect, that is, to whom one agent should connect to. Distinct rules will generate distinct topologies where some topologies are more adequate to particular problems than other. This rule definition allows one to borrow theorems from network science.

II *Aggregation Rule*: this rule is responsible to manage how one agent's meme (or solution) should influence their respectively connected agents.

III *Appropriation Rule*: tells how the agent should manipulate the meme it has. For instance, it could be any metaheuristic used in a local iterated search algorithm.

As one may realize, this model is extremely flexible in the sense that rules are defined from a very abstract context. Moreover, this model is extremely powerful because it could unify different metaheuristics and techniques into a single and hybrid instance.

The memetic network model has some advantages over a purely GA approach. One of them is that an instance of this model may converge faster than a GA to a good solution and this is particularly interesting in the case of the ME problem especially for real-time video coders. This advantage and the structural simplicity of a memetic network form together the main motivation to the use of an instance of this model focused on the HD ME problem. We thus propose a new memetic network-based algorithm called MNA-ME.

In the MNA-ME, one agent always start in the center and the others should start in random positions inside the search window. Given that, the three rules are defined as follows:

I *Connection Rule*: each agent should necessarily connect to the central agent and with the agent who holds the best current solution. This way, each agent should know where is the best ranked agent, that is the best block matching achieved so far. This rule is represented by a matrix which is updated at each iteration.

II *Aggregation Rule*: the agent which has incoming connections spatially "attracts" the connected agent by a factor named *aggressiveness* denoted by α. The higher this factor is, the stronger will be its influence.

III *Appropriation Rule*: the agent, after exchanging information, performs a full search in the range $(\pm 1, \pm 1)$ and changes its current location to better solution if one is found.

It is useful to further explain the *aggressiveness* parameter which denotes the factor by which an agent should change its positions according to connections made so far. This parameter may vary from 0.0 to 1.0 but is fixed throughout the computation. The value 0.0 denotes no influence at all, while the value 1.0 replicates the best agent rendering itself irrelevant. A value of 0.5 puts the connected agents at the middle of the original distance between them.

IV. Experimental Results

As our objective with both algorithms is to achieve high quality in high definition, the test sequence set should be composed exclusively of HD sequences. These sequences are freely available at [18] and their resolution is 1920x1080 pixels progressive (1080p). Motion estimation and motion compensation were executed solely on luminance samples and the PSNR was obtained comparing the original frame and the motion compensated frame that is, the output of the MC (the residual frame was discarded). This decision is justified in the sense that this work is not tied to a single video coding standard, but rather to a conceptual point of view. Adopting this principle is useful in the sense that it contributes to the generalization of our contribution.[1]

Our main complexity metric is the number of evaluated search points (ESP). Since the processing time grows linearly with the number of evaluated blocks, the time length of each simulation was also discarded. The block size used was fixed

[1]The algorithms RS4 and MNA-ME were implemented in C so they could, in principle, be evaluated in terms of complexity and objective quality.

in 4×4 pixels because more vectors would be generated and consequently a more precise motion estimation process would be carried out. Using a small block size is not a problem at all since a bottom-up variable block size algorithm [19], [20] can be used to achieve larger block sizes that in turn may reduce the number of vectors to be encoded. Thus improving the coding efficiency of the coder as a whole. No SAD sampling technique was used in our experiments and only the first two hundreds frames of each sequence were considered (approximately 8 seconds).

Since the RS4 and the MNA-ME are essentially stochastic algorithms, their results may change from distinct executions for the same input. This feature renders the need of running the execution many times, so the expected behaviour can be better know. For our experiments, both algorithms were executed ten times for each sequence for a given set of parameters and the mean absolute deviation rendered itself not significant, being less then 0.02 in average for PSNR. Figure 2 presents the results of the RS4 algorithm for each sequence video test set where each line represents a different size of the search window used. A search window of 16×16 means that the range of search is from -8 to $+8$ and so on. The number of search points chosen for the random step was 16 since this value is approximately the upper bound for the DS algorithm with 4 iterations.

The first result to be noticed is that sequences like *pedestrian area* and *riverbed* can be better coded using a bigger search range. These sequences are motion intensive and the 64×64 search window presents considerable gains over the 32×32 search window of 1.46db and 1db respectively. These values are considerable high considering the logarithm scale of the PSNR metric. On the other hand, the RS4 algorithm shows better efficiency with a small search window on sequences with low motion information, namely *station2* and *sunflower*. For the sequences *tractor* and *rush hour*, the 32×32 search window yielded the best results.

The MNA-ME algorithm was tested with the fixed connection rule described in Section III and three agents only to keep complexity under an acceptable limit. However, the aggressiveness parameter was evaluated in the range $[0, 1]$ with 0.2 incremental steps. This evaluation would allow us to better understand how much a single node may interfere on its neighbor nodes in different space solutions. For example, a rough space solution search may be privileged by a higher aggressiveness since less time would be spent in potentially low quality solutions. In Figure 3, the results of this study are presented.

All results in Figure 3 were evaluated for a window size of 256x256. The decision to use such a relatively large search area is to make the motion estimation difficult especially for the random step so a real hard test would be endured by the MNA-ME algorithm. According to the results, there is a correlation between aggressiveness and the objective quality which holds true for all sequences except *sunflower* and *station2*. The optimum value for this parameter considering the test set was 0.2, which is a relatively small value.

For a more wider comparison with our algorithms, one should carry out evaluations of classical algorithms even if their focus is not HD. Thus, for completeness we considered both fast classical algorithms and one exhaustive search. The former with relatively large fixed search window range and the later with different search window ranges. This data is available in Table I where the average results of each algorithm for the test sequences are introduced. The table contains the quality metric (PSNR in db) and the complexity metric (ESP) for a given search range. The algorithms implemented and evaluated were: the Diamond Search (DS) algorithm, the Three Step Search algorithm (TSS), the 4 Step Search algorithm (4SS) and the Full Search algorithm.

TABLE I: Comparative of the proposed with the classic algorithms.

Algorithm	**PSNR**	**ESP** $(\times 10^9)$	**Search Range**
RS4	**34.43**	**0.723**	**(±8,±8)**
RS4	**34,87**	**0.716**	**(±16,±16)**
RS4	**34,82**	**0.719**	**(±32,±32)**
RS4*	**35,51**	**-**	**-**
MNA-ME	**33,27**	**0.983**	**(±128,±128)**
TSS	32.77	0.697	(±64,±64)
4SS	34.7	0.928	(±64,±64)
DS	33.46	0.637	(±64,±64)
FS	35.08	7.429	(±8,±8)
FS	38.33	26.542	(±16,±16)
FS	40.21	106.168	(±32,±32)
FS	41.31	424.673	(±64,±64)

Except for the range $(\pm 8, \pm 8)$, the RS4 algorithm presented PSNR gains over the DS, TSS and 4SS algorithms. In the case of RS4 algorithm in the search range of $(\pm 16, \pm 16)$ gains of 2.1db, 1.41db and 0.17db in PSNR were achieved over TSS, DS and 4SS respectively. Considering the ESP, the RS4's complexity is very close to the TSS and is just 11% higher. It should be noted that a fixed N value was used and that this ESP is closely related with it. As it can be seen, the RS4 algorithm better evaluates a search window when compared to other fast classical algorithms. This in turn reflects in less burden to memory buses since small data chunks are needed. Compared to the smallest FS, the RS4 achieves in average 0.65db less in PSNR but does approximately 10 times less block operations. Since the FS explores all possible candidate blocks, its ESP grows exponentially as the search range grows. However, the PSNR seems saturated at search ranges larger than $(\pm 64, \pm 64)$. The RS4* row presents the average result of the best search range among $(\pm 8, \pm 8)$, $(\pm 16, \pm 16)$ and $(\pm 32, \pm 32)$, that is the range maximize PSNR for a given sequence and its relevance is to show that a adaptive search range technique may improve considerably the quality results for a fixed N value. Furthermore, it is important to note that the results in this case were better than the Full Search algorithm.

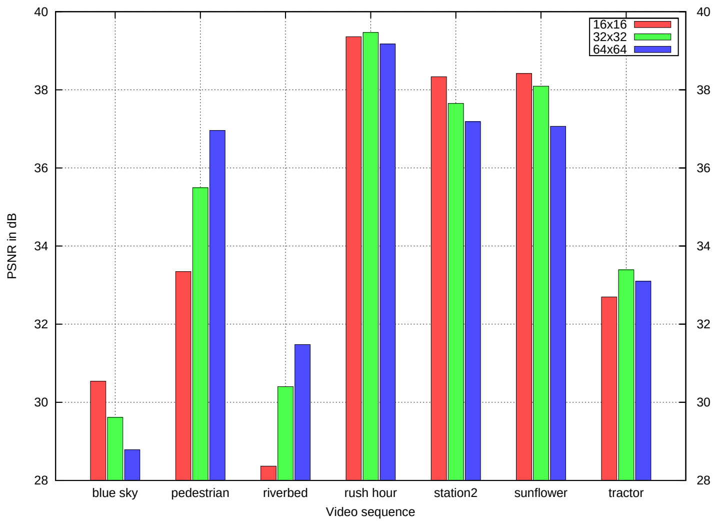


Fig. 2: The effect of search window size in RS4 PSNR results.

The MNA-ME algorithm did poorly in quality and complexity since it did not outperform the DS and the 4SS, achieving quality gains only over the TSS algorithm. These results present evidence that our current approach may not be the optimal one. In particular, the failure to achieve a good quality on HD ME merits further investigation about the problem, instance of the model and the model itself. This may also suggest that a fundamentally different approach is needed, combining a more straightforward approach in the design of the algorithm. One hypothesis is that distinct agents may leave too soon their current local optimal in favour of other agent information. This in turn could be a trap since the former agent could possible reach a better solution give time for it. Moreover, the agent that provided the information could be stuck in a local minimal.

A possible solution would be allowing to have agents that do not share information neither allow others to see their solution since this approach would allow some agents to better explore its neighbour blocks. Another solution would be adopting an information exchange strategy similar to a simulated annealing where one may locally explore its close solution at the beginning and as time advances they become more willing to influence of other agents. This approach may guarantee better local space solution evaluating we believe.

Comparing the results of new algorithms would be *per se* a relatively important contribution but this is not the focus of this work. For simplicity, we assume that the improvements they present over the original classical algorithms are essentially dependent of the performance of the root algorithm for a specific video sequence.

V. Conclusions and Future Work

In this paper, we introduced two novel algorithms for high quality motion estimation for high definition coding. The first algorithm, the RS4, relies in an initial random sampling that is further refined by a greedy search step. Although very simple, the RS4 algorithm presented good results with the advantage that it could be better tuned in real time by adjusting the N parameter dynamically according to the video input. This feature has not been implemented yet, but it has the potential to reduce the number of search points calculated for low motion sequences and then spend this saved computations when needed and thus providing a better solution at all. Additionally, dynamically adjusting the search window and using a motion prediction vector algorithm may help the RS4 algorithm to achieve a better trade off between complexity and quality.

The other algorithm was based on a population-based stochastic optimization class of algorithms in which individuals exchange information through the underlying network. This algorithm was based on a memetic network algorithm. The rules were appropriately defined for the ME application and the resulting instance was called MNA-ME. This instance was tested in for high definitions sequences and our results implied that the choice of the rules (especially the connection rule) may not be the optimum one since the results were somewhow limited in terms of quality and complexity. Perhaps, this empirical study also revealed more fundamental issues related to the MNA class. Although addressing and solving these issues with the model is not the focus of this work, it is worth to mention that one still lacks an efficient methodology to instantiate an optimization algorithm of this class for a

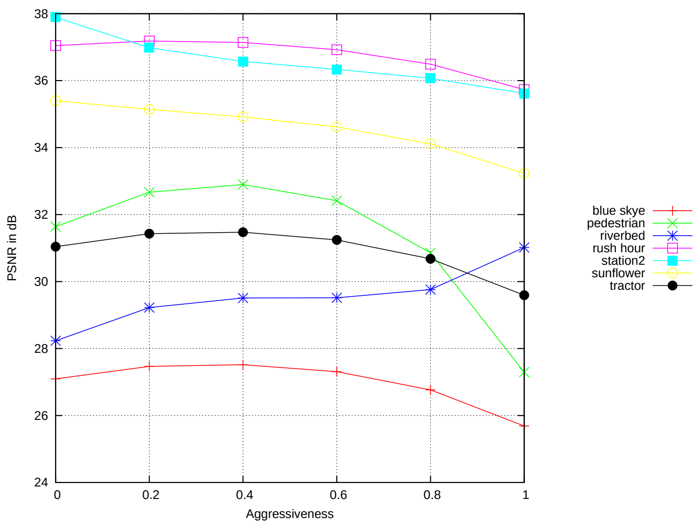


Fig. 3: Variation of the aggressiveness for the MNA-ME algorithm.

specific problem. This issue is completely understandable in the sense that MNA is a relatively new general model for solving problems.

Another contribution of this work is that we provided further evidence that fast algorithms do not perform well for high definition video sequences and that all the work related to this field should consider addressing this point by executing the simulations for HD contents also. This point is consistent with the ever increasing demand for high definition content to the final user. Moreover, we believe that it is important to bring this point into consideration.

As future work, we pretend to improve both algorithms and implement them inside a real coder so that bit rate data could be obtained from simulations. For instance, new topologies should be evaluated for a new MNA-ME instance like small-worlds, hierarchical and sparsely-connected topologies, etc. We also pretend to experiment SAD subsampling techniques with our algorithms since they do not affect the quality in a significant way [21]. Besides, a hardware implementation in an FPGA device for RS4 algorithm is currently being planned.

Acknowledgment

This work was partly supported by CNPq-Brazil.

References

[1] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with h.264/avc: Tools, performance and complexity"," *IEEE Circuits and Systems Mag.*, 2004.

[2] T. Toivonen and J. Heikkila, "Improved unsymmetric-cross multi-hexagon-grid search algorithm for fast block motion estimation," in *Proceedings of the IEEE International Conference on Image Processing*, oct. 2006, pp. 2369–2372.

[3] Z. Chen, J. Xu, Y. He, and J. Zheng, "Fast integer-pel and fractional-pel motion estimation for h.264/avc," *Journal of Visual Communication and Image Representation*, vol. 17, no. 2, pp. 264–290, 2006.

[4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," *Nature*, 1981.

[5] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Transactions on Image Processing*, vol. 9, no. 2, pp. 287–290, feb 2000.

[6] C. Zhu, X. Lin, and L.-P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 5, pp. 349–355, may 2002.

[7] O. Tasdizen and I. Hamzaoglu, "Recursive dynamically variable step search motion estimation algorithm for high definition video," in *Proceedings of the 20th International Conference on Pattern Recognition (ICPR)*, aug. 2010, pp. 2354–2357.

[8] I. Richardson, *Video codec design: developing image and video compression systems*. Wiley, 2002.

[9] C. Cai, H. Zeng, and S. K. Mitra, "Fast motion estimation for h.264," *Image Communication*, vol. 24, no. 8, pp. 630–636, 2009.

[10] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, april 2004.

[11] L.-M. Po and W.-C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 313–317, jun 1996.

[12] R. Li, B. Zeng, and M. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438–442, aug 1994.

[13] T.-H. Tsai and Y.-N. Pan, "A novel 3-d predict hexagon search algorithm for fast block motion estimation on h.264 video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 12, pp. 1542 –1549, dec. 2006.

[14] O. Tasdizen, A. Akin, H. Kukner, and I. Hamzaoglu, "Dynamically variable step search motion estimation algorithm and a dynamically reconfigurable hardware for its implementation," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 3, pp. 1645–1653, august 2009.

[15] S. Boltz and F. Nielsen, "Randomized motion estimation," in *Proceedings of the IEEE International Conference on Image Processing*, sept. 2010, pp. 781–784.

[16] J. Vanne, E. Aho, K. Kuusilinna, and T. Hamalainen, "A configurable motion estimation architecture for block-matching algorithms," *IEEE*

Transactions on Circuits and Systems for Video Technology, vol. 19, no. 4, pp. 466 –477, april 2009.

[17] R. M. Araujo and L. C. Lamb, “Memetic networks: Analyzing the effects of network properties in multi-agent performance,” in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI-08)*, 2008, pp. 3–8.

[18] Xiph.org Test Media Repository, 2011. [Online]. Available: http://media.xiph.org/video/derf/

[19] I. Rhee, G. Martin, S. Muthukrishnan, and R. Packwood, “Quadtree-structured variable-size block-matching motion estimation with minimal error,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 10, no. 1, pp. 42–50, feb 2000.

[20] R. S. Dornelles, F. M. Sampaio, and L. V. Agostini, “Variable block size motion estimation architecture with a fast bottom-up decision mode and an integrated motion compensation targeting the h.264/avc video coding standard,” in *Proceedings of the 23rd symposium on Integrated circuits and system design*. New York, NY, USA: ACM, 2010, pp. 186–191.

[21] F. Moschetti, M. Kunt, and E. Debes, “A statistical adaptive block-matching motion estimation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 5, pp. 417–431, may 2003.

Modeling Adaptative Social Behavior in Collective Problem Solving Algorithms

Diego Noble and Luís Lamb
Federal University of Rio Grande do Sul
Institute of Informatics
Porto Alegre, Brazil
$\{dvnoble, lamb\}$*@inf.ufrgs.br*

Ricardo Araújo
Federal University of Pelotas
Centro de Desenvolvimento Tecnológico, CDTec
Pelotas, Brazil
ricardo@inf.ufpel.edu.br

***Abstract*—Collective problem solving can lead to the development of new methods and algorithms that can potentially contribute to novel Artificial Intelligence applications and tools. Socially-inspired optimization algorithms are a class of algorithms that aim at conducting a search over a large solution space using mechanisms similar to how humans solve problems in a social context. Several such algorithms exist in the literature, including adaptations of classical ones, such as Genetic Algorithms. These models, however, do not take into account a fundamental concept in human social systems: the individual ability to adapt problem-solving strategies as a function of the social context. In this paper, we propose and investigate an extension inside a socially-inspired model of collective problem solving which allows one to model agents with such adaptability. This extension is based on the concept of humans as "motivated tacticians" and it dictates how agents are to adapt their search heuristics according to their respective social context. We show how this rule can speed up the system's convergence to good solutions and improve the search space exploration. The results contribute towards the design of socially inspired computational systems for collective problem-solving.**

***Keywords*-Computational Intelligence; Swarm Intelligence; Optimization**

I. Introduction

A central issue in Artificial Intelligence is how to efficiently perform search for solutions in a very large search space [1]. Several heuristics exist that try and reduce the need for computational resources over exhaustive search methods. These heuristics are developed with a "strategy first" approach which renders them as black box generalist methods. Analytical results showed that no such black box algorithm can outperform every algorithm across all domains [2]. This important result points that the development of an algorithm must be bound to its application, incorporating as much information as possible about the domain. Yet, there the study of black box optimization algorithms is a strong research area and its main motivation is to devise algorithms which could be used as last choice when there is little to no information about the problem to be solved. Complying with this motivation, is the need to better understand search mechanisms and their limitations.

A somewhat recent trend in the design of search heuristics is to take inspiration in social systems, by using parallel computing to represent many simultaneous search that somehow exchange information. Such systems can be composed of e.g. social insects [3] or simpler "inanimate" particles [4]. Human social systems have recently gained attention and some heuristics try and capture some aspects of human social networks [5]. The motivation for such heuristics range from understanding how these social systems work in the real world to constructing algorithms that can perform well in certain scenarios.

These socially-inspired models typically employ agents that are simple rule-based automatons, unable to adapt to any information discovered during the search. To do so, we modified an instance of the Memetic Network metamodel [5] to allows agents to adapt their search heuristic based on the social context information.

We provide preliminary experimental results on the effects of introducing such concept in several optimization scenarios composed of minimizing multidimensional real-valued functions. The results provide evidence that allowing social adaptation can be beneficial to the search as a whole.

The paper is organized as follows. Section II provides the necessary background to understand Memetic Networks and provides a standard instance of the model for real-valued function minimization. Section III introduces the extension of the model by which agents are allowed to adapt their search behavior. Section IV presents the experimental setup. Section V presents and analyzes the results. Section VI concludes the paper and point out directions for further research.

II. The Memetic Network Model

Memetic Networks is a recent metamodel inspired by how humans collectively solve problems [5]. It basically determines guidelines for the development of a system whose task is to search for the best solution to a given problem. The development of such system may be driven solely by the search task. However, one may also want to investigate the effects of modeling real social features into artificial social systems by objectively measuring the performance of the system in the search.

The metamodel defines simple elements such as nodes, memes and links and also defines rules that state how these constructs relate to each other. The concepts behind elements and rules are problem invariant, however their implementation is not: one needs to encode information about the problem's domain in order to build an effective problem solving system. In this sense, the Memetic Networks metamodel is a not a purely blackbox box method for optimization, but rather a guideline for structured design of algorithms with embedded information about the problem domain.

A *memenet* is the final construct when one assembles the Memetic Network's elements and then describes the rules by which the elements will interact. The memenet's main objective is to achieve better solutions in less iterations than purely independent and isolated nodes would achieve. To do so, it allows nodes to, a nodes in a memenet are allowed to exchange information.

The computing mechanism behind Memetic Networks is represented by nodes. Each node contains a complete solution to the problem at hand. This solution can be (i) evaluated using a global evaluation function, and (ii) compared or ranked against other solutions. Nodes are responsible for retrieving, aggregating and processing *memes*, which are the minimal meaningful pieces that compose complete solutions. The meme concept is partially based in ideas from Dawnkin's book, *The Selfish Gene* [6]. Links connect nodes and ultimately determine a supporting "social" network. If there is a link between two nodes, then these nodes are allowed to exchanging memes in both directions.

The Memetic Network elements alone are not self-sufficient to perform search, it remains necessary to define how these elements will behave when assembled together. This is accomplished by three rules, which roughly capture how humans process information in a social context [5]. These rules detail how nodes choose to connect to each other, how information is retrieved from the social network and how individual nodes contribute to a solution being sought. Each rule is detailed in what follows.

Connection Rule — Defines how the nodes should connect, that is, how links are to be established. This rule can be performed once before the search takes place, characterizing a static network topology. Alternatively, the connection rule can be executed during the search process so the network topology can evolve together with the system. As in other socially-inspired models [1], there is currently no conclusive evidence about what topology is generally better for specific types of problems [7].

Aggregation Rule — This rule controls the interaction between connected nodes, that is, how nodes retrieve and aggregate information from their network neighbors. Typically, a node retrieve information [2] from better ranked neighbors only. After retrieving memes, a node must aggregate — i.e. combine — them to assemble a new solution. The aggregation rule must also describe algorithimically how these memes are to be combined. Aggregation is highly dependent on the network topology [3].

Appropriation Rule — In possession of the recently aggregated solution, a node may add some novelty to it. In this step, local information is added to the solution, allowing memenets to explore the search space, much in the same way mutation works in genetic algorithms. Appropriation can happen by simple random changes to the solution or by applying some deterministic local search to it (in a similar fashion to Genetic Algorithms [8] and Memetic Algorithms [9] respectively).

Although the aggregation and the appropriation rules depend on the problem at hand, the structure of a memetic network algorithm is the same across different problems. Algorithm 1 describes this structure if one is to model a static — the connection rule is performed once only — fully connected network. The *stop condition* can be the number of iterations or a desired solution quality threshold.

Now that the structure of the metamodel is described, we need to instantiate the model to the problem of minimization over multidimensional real functions. This is done in the next subsection.

A. Memetic Networks for Function Minimization

In this subsection, we will define the elements and the rules to build a memenet capable of performing minimization over multidimensional real functions. A solution for this problem can be coded as a real-valued array where each component represents a position inside a single dimension of the search space. Every solution has an evaluation and the goal is to find the solution that minimizes this evaluation function.

Having defined the problem, it is possible to build a straightforward memenet by associating an array to a node and a meme to an array component. So if we are working with a function in a n-dimensional space, a node will store an array of size n where each component of this array will be considered a meme. Links in our memenet will be bi-directional and static, these features will keep the model essentially simple while still in accordance with our objectives. For comparative purposes, every node will also carry the function output from the last function evaluation. This value is called *cost* for this is a minimization problem.

The aggregation rule retrieves and then combines memes from better ranked neighbors to create a new solution. This process is described in Algorithm 2. A node first creates a $MemePool$. The purpose of this structure is to store solutions that came from neighbors with lower scores and

[1]e.g. Particle Swarm Optimizers

[2]Retrieval of information is considered noiseless. It remains an open question whether communication noise could be benefical or not for population-based optimization problems.

[3]This feature is where socially-inspired optimization algorithms differ the most from Evolutionary Algorithms.

Algorithm 1: Pseudocode for the Memetic Network Algorithm using a static fully connected network.

```
Input  : $Population_{size}$, $Problem_{size}$
Output : $Solution_{best}$
memenet ⟵ InitializePopulation ($Population_{size}$,$Problem_{size}$)
foreach $node \in$ memenet do
    $node_{\text{neighbors}}$ ⟵ All nodes from memenet
repeat
    EvaluatePopulation (memenet)
    foreach $node \in$ memenet do
        betterNeighbors ⟵ RetrieveBetterRankedNeighbors ($node_{\text{neighbors}}$,$node_{score}$)
        $Solution_{Aggregated}$ ⟵ Aggregate (betterNeighbors,$Problem_{size}$)
        $Solution_{Appropriated}$ ⟵ Appropriate ($Solution_{Aggregated}$, $Problem_{size}$)
        $node_{Solution}$ ⟵ $Solution_{Appropriated}$
    $Solution_{best}$ ⟵ RetrieveBestSolutionFrom (memenet)
until stop condition
return $Solution_{best}$
```

also the nodes own solution. Each component of the solution is replaced by a respective meme of the $MemePool$ list, that is a meme from the same position of the available arrays. All solutions from $MemePool$ have equal chances to be chosen.

Algorithm 2: The aggregation rule instantied to minimization.

```
Input  : BetterNeighbors, $Problem_{size}$
Output : $Solution$
MemePool ⟵ RetrieveSolutions (BetterNeighbors)
Append node's solution to MemePool
$Solution$ ⟵ $\emptyset$
if CountSolutions (MemePool) $> 1$ then
    for $i \leftarrow 1$ to $Problem_{size}$ do
        $SolutionChosen$ ⟵
        RandomlyChooseASolutionFrom (MemePool)
        $Solution_i$ ⟵ $SolutionChosen_i$
else // When the node has no better
     neighbor, its solution is kept.
    $Solution$ ⟵ $CurrentNodeSolution$
return $Solution$
```

Aggregation does not generate new memes because it only compose new combinations from already known solution components. To address aggregation's inability to add innovation, the memetic network employs a mechanism which allows nodes to modify their memes. A stochastic simple mechanism is portrayed in Algorithm 3. For each component of the solution, a valid meme is generated and attributed with probability $P_{appropriation}$. As with mutation probability in genetic algorithms, the p value should be small enough so that convergence is not drastically disrupted. All valid memes have the same probability of being generated in this process.

Algorithm 3: The appropriation rule instantied to minimization.

```
Input  : $Solution$, $Problem_{size}$, $P_{appropriation}$
Output : $Solution$
for $i \leftarrow 1$ to $Problem_{size}$ do
    With $P_{appropriation}$ probability:
        $Solution_i$ ⟵ RandomlyChooseMeme ()
return $Solution$
```

III. Modeling Adaptative Social Behavior

In the last section, we described an appropriation method for the problem of function minimization. By that method, all nodes in the population follow the same fixed and unconditional set of steps, insensible to social context. This feature contradicts the evidence social psycologists have that in real social systems "individual behavior is strongly influenced by the environment, especially the social environment" [10]–[12].

We hypothesize that if an agent is capable of infer its social context, then adapting its search behavior may improve not only its results but also the collective performance in minimization. We further qualify our hypothesis by specifying that nodes which are faring well (good social context) must exploit more their good solution Our investigation then aims at the analysis of the effects of the influence the social environment has in socially sensible agents.

In the context of Memetic Networks, we could define socially sensible agents as agents who know how its social relative performance. Based on its social relative performance, an agent is capable of adapt its search behavior then during the appropriation step. This is justified because in this step, no memes are exchanged anymore and the node is to improve its solution by itself

To test our hypothesis, we must decide (1) which clues will be used to guide appropriation and (2) in what ways appropriation will implement different search behaviors. There are many answers to each these questions though we are interested in the simplest ones. Regarding the first question, a node could be allowed to count how many nodes are faring better than it. Regarding the second, a node may dynamically manipulate the search space range. Hitherto, we need more formal and concise description of the previous answers.

We define a *Social Relative Performance* (SRP) ratio to determine how well a node fairs in social context. The idea is to allow agents to infer social context by counting how many neighbors are faring better than them. The SRP ratio is given by Equation (2) where B is the set of better ranked neighbors of node n and T the set all neighbors of n. This is a utterly simple property a node have and can be easily computed during the search.

$$SRP(n) = \frac{|B|}{|T|} \tag{1}$$

Therefore, the best ranked node in the network will have an SRP of 0.0 and the worst node an SRP of 1.0.

We propose adapting a node's search heuristic by controlling how much it changes the solution as a function of its SRP. Instead of randomly replacing a meme with *any* meme during appropriation, it may be reasonable to attribute a higher probability to "closer" memes when the node is performing well (i.e. favor small changes to the solution when one is doing well compared to their peers). Alternatively, a node which is performing poorly compared to its peers may benefit from modifying its solution to a greater extent. A similar approach in Genetic algorithms (GA) is to use Gaussian mutation operators [13]. Our contribution is new in the sense that it regulates such adjustment through the provided social context, something which essentially GAs lack.

In order to instantiate the above reasoning, we define Δ to be the maximum variation allowed when modifying each component of the solution (i.e. when choosing the a new component for the solution, it must vary at most with Δ). The simplest choice to relate SRP with search heuristic is to adapt the range according a linear function. Equation (2) describes this function, where Max and Min represent the maximum and minimum values allowed by the representation of the problem, respectively. The constant α defines the minimum amount of variation possible is bounded by the machine representation.

$$\Delta = (|Max| + |Min|)/\,2 \times SRP + \alpha \tag{2}$$

In this equation, a node with an SRP equals to zero — i.e. the best ranked node in the neighborhood — will have an effective search range of $[-\alpha, +\alpha]$ while the worst node will have a search range of $[-(\alpha + (|Max| + |Min|)/2), +(\alpha + (|Max| + |Min|)/2)$. Nodes with intermediate performance will have linearly intermediate search ranges. We call this appropriation algorithm *R** in reference to range adaptation and we describe it in Algorithm 4.

Algorithm 4: The R* appropriation rule instantiated to minimization.

```
Input  : $Solution$, $Problem_{size}$, $P_{appropriation}$,
         $node_{Neighbors}$
Output : $Solution$
$Better \longleftarrow$ CountBetter ($node_{Neighbors}$)
$Total \longleftarrow$ Count ($node_{Neighbors}$)
$SRP \longleftarrow Better/Total$
$\Delta \longleftarrow (|Max| + |Min|)/\,2 \times (SRP) + \alpha$
for $i \leftarrow 1$ to $Problem_{size}$ do
    With $P_{appropriation}$ probability:
      $Solution_i \longleftarrow$ RandomlyChooseMemeBetween (
      $Solution_i - \Delta$ , $Solution_i + \Delta$ )
return $Solution$
```

The R* appropriation requires a repair routine afterwards for its may generate solutions outside the search range. We use a simple routine which sets every meme that is smaller than Min to Min and Max to memes that are larger than Max.

IV. Experimental Setup

Five benchmark functions were used to test our hypothesis: Ackley, Griewank, Rastrigin, Rosenbrock, and Sphere functions. They are described in in Table I as is their respective search ranges and the values used in initialization.

The unsymmetrical range during initialization was purposeful since uniform distribution inside the search space may favor or prejudice search operators [14]. With the exception of the Rosenbrock Function, all functions have their global minimum at the very center of the search space. The global minimum value is 0.0 for all functions. The Rosenbrock and the Sphere function are unimodal while the remaining three functions are multimodal. These functions were evaluated in 30 dimensions.

The float precision used was of eight decimal digits. We used 1000 iterations as the *stop condition* for all functions. All links were static and they were built under no specific assumptions[4]. The experiments were carried using a population of 36 nodes arranged in a 6×6 Von Neumann network topology. In this topology each node has four distinct neighbors. Fig. 1 depicts this topology in a three dimensional representation.

In each experiment, the original memenet was compared against the memenet extended with the R* appropriation and the canonical Particle Swarm Optmization. The R* appropriation used $\alpha \leftarrow (|Max| + |Min|)\,/2000$. We performed 100 independent trials for each experiment and provide statistical results of these experiments.

[4]One could assume that spatial position or initial score could be used to establish links between related nodes.

Table I
BENCHMARK FUNCTIONS

Name	Definition	Initialization Range $[x_{min}, x_{max}]^n$	Search Range $[x_{min}, x_{max}]^n$
Ackley	$20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)}$	$[15.0, 30.0]^n$	$[-30.0, 30.0]^n$
Griewank	$\frac{1}{4000}\sum_{i=n}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos(\frac{x_i - 100}{\sqrt{i}}) + 1$	$[300.0, 600.0]^n$	$[-600.0, 600.0]^n$
Rastrigin	$\sum_{i=n}^{n}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	$[2.5, 5.1]^n$	$[-5.1, 5.1]^n$
Rosenbrock	$\sum_{i=1}^{n-1}\left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right)$	$[15.0, 30.0]^n$	$[-30.0, 30.0]^n$
Sphere	$\sum_{i=1}^{n} x_i^2$	$[50.0, 100.0]^n$	$[-100.0, 100.0]^n$

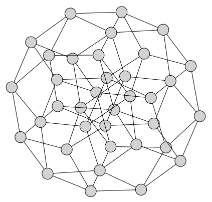


Figure 1. The 6×6 three dimensional representation of the Von Neumann network topology used in the experiments.

V. RESULTS

Fig. 2 depicts the convergence of the memenet with R* appropriation, the memenet wit the conventional appropriation and the canonical PSO [7], [15]. In it, the 10 worst and best trials were removed and the results were averaged over the 80 remaining trials — this was done to guarantee more stable results. The vertical lines represent the maximum and the minimum results.

The convergence of the R* appropriation in the Ackley and Griewank functions was slightly faster then the conventional appropriation. The R* appropriation also kept improving the result until the stop criteria in the former. Until the 300^{th} iteration the R* appropriation was competitive with the PSO. The R* appropriation outperformed the PSO algorithm the Rastrigin fuction, a highly multimodal function. In this case, the memenet presented the largest variation of minimum and maximum values of quality for the best solutions. This behavior was not observed in the other functions where the convergence of the R* appropriation was remarkably stable. The result of R* appropriation in the Rosenbrock function was competitive with the PSO result. In the Sphere function, the R* appropriation's result was better than the convetional appropriation while worse than the PSO. This last result was expected due to the PSO highly exploitation rate.

VI. CONCLUSIONS

In this article, we have investigated the metaphor of "motivated tacticians" in order to derive mechanisms of social adaptation. Our motivation was to contribute towards the design of socially inspired computational systems with intelligent methods. We hypothesized that through a social adaptation mechanism, a collective problem-solving system could converge faster to better solutions than a system without it. We described a mechanism — the R* appropriation — within a model inspired by human social problem-solving and this mechanism operated as a function of the social context. We tested this new mechanism in five benchmark functions and two different network topologies.

The results of the R* appropriation gave supporting evidence to our hypothesis that a swarm of motivated tacticians sensible to social context can in fact improve the system's convergence speed. Improvements were achieved in terms of both convergence rate and solution quality.

As future work, we plan to extend the SRP ratio to take into account the neighbor's scores as well as the simple count of how many are doing better. There is also the problem of investigating noise in form of small random perturbations during the retrieval of memes by a node. We are also interested to observe the how motivated tacticians influence the system in combinatorial optimization problems.

ACKNOWLEDGMENTS.

This work was supported by CNPq, Brazil.

REFERENCES

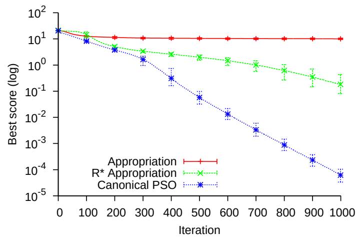
[1] S. Russell and P. Norvig, *Artificial intelligence: a modern approach*, ser. Prentice Hall series in artificial intelligence. Prentice Hall, 2010.

[2] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67 –82, apr 1997.

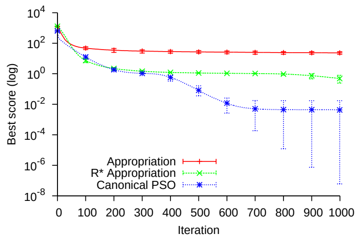
[3] M. Dorigo and T. Stützle, *Ant Colony Optimization.* MIT Press, 2004.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, nov 1995, pp. 1942–1948.

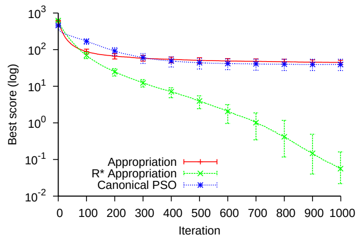
[5] R. M. Araújo and L. C. Lamb, "Memetic networks: analyzing the effects of network properties in multi-agent performance," in *Proceedings of the 23rd national conference on Artificial intelligence.* Chicago, Illinois: AAAI Press, jul 2008, pp. 3–8.



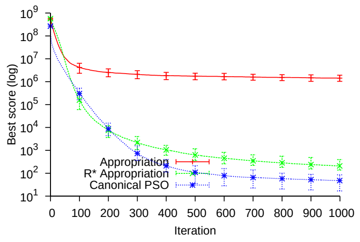
(a) Ackley Function



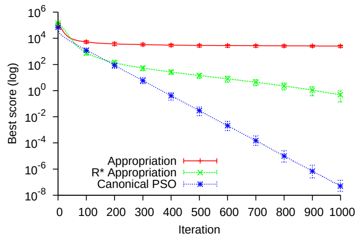
(b) Griewank Function



(c) Rastrigin Function



(d) Rosenbrock Function



(e) Sphere Function

Figure 2. Progression of the best score averaged over the 80 trials for all functions. The 10 worst and best trials were removed. Vertical lines represent maximum and minimum results. The R* appropriation showed superior results to the conventional appropriation. The R* appropriation with memenet outperformed canonical PSO in the Rastrigin function.

[6] R. Dawkins, *The Selfish Gene*. Oxford University Press, 2006.

[7] R. Poli, J. Kennedy, and T. Blackwell, “Particle swarm optimization,” *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, Aug. 2007.

[8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[9] P. Moscato, “Memetic algorithms: A short introduction,” in *New Ideas in Optimization*. McGraw-Hill, 1999.

[10] S. E. Taylor, *The social being in social psychology*, ser. The Handbook of Social Psychology. McGraw-Hill, 1998.

[11] N. Schwarz, “Warmer and more social: Recent developments in cognitive social psychology,” *Annual Review of Sociology*, vol. 24, pp. 239–264, 1998.

[12] S. T. Fiske and S. E. Taylor, *Social Cognition*, 2nd ed. McGraw-Hill, 1991.

[13] R. Hinterding, Z. Michalewicz, and T. Peachey, “Self-adaptive genetic algorithm for numeric functions,” in *Parallel Problem Solving from Nature âĂŤ PPSN IV*, ser. Lecture Notes in Computer Science, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Springer Berlin, 1996, vol. 1141, pp. 420–429.

[14] D. B. Fogel and H.-G. Beyer, “A note on the empirical evaluation of intermediate recombination,” *Evolutionary Computation*, vol. 3, no. 4, pp. 491–495, Dec. 1995.

[15] R. Eberhart and Y. Shi, “Comparing inertia weights and constriction factors in particle swarm optimization,” in *Proceedings of the Congress on Evolutionary Computation (CEC)*, vol. 1, 2000, pp. 84–88.

Investigating a Socially Inspired Heterogeneous System of Problem-solving Agents

Diego V. Noble and Luís C. Lamb
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, Brazil
$\{dvnoble, lamb\}$@inf.ufrgs.br

Ricardo M. Araújo
Centro de Desenvolvimento Tecnológico, CDTec
Federal University of Pelotas
Pelotas, Brazil
ricardo@inf.ufpel.edu.br

***Abstract*—Social interactions have recently been used as an inspiration for novel agent-based problem-solving models. Particle Swarm Optimization and Memetic Networks are two such algorithms. Although they draw inspiration from different real-world social systems, they both rely on the concept of a social network to regulate the internal information flow in a structured way. In this paper, we systematically investigate how a heterogeneous population composed of individuals from these two models behave as the system seeks the solution to the benchmark problems. We report on extensive numerical simulations, showing that this heterogeneous model is able to converge faster in two highly multimodal scenarios while being otherwise statistically equivalent to the original homogeneous models. Our results provide supportive evidence for the hypothesis that higher diversity in populations of problem-solvers can be beneficial and also adds a new dimension to previous heterogeneous problem-solving models.**

***Index Terms*—Computational Intelligence; Swarm Intelligence; Problem-solving; Optimization**

I. Introduction

Many real-world problems can be translated into optimization tasks, which can be challenging to be solved by traditional methods due to their dynamic, multimodal, non-continuous and non-linear features. When the problem domain is novel and little to nothing is known about it, general algorithms, i.e. black-box methods, can be used as first-line choices. Several such algorithms try to mimic how similar problems are solved in nature. This is the case of e.g. Genetic Algorithms [1], [2] and Artificial Immune Systems [3], [4]. Such approaches have two benefits: allowing for automatically and efficiently finding good solutions to certain problems and providing insights on the workings of the natural system.

Social systems have also been used to draw inspiration for novel algorithms. For example, Ant Colony Optimization mimics social insects, Particle Swarm Optimization [5], [6] (PSO) models bird flocks and fish schools foraging for food. More recently, attempts have been made to build algorithms based on *human* social systems, either by extending existing models [7]–[9] or by the proposal of novel models, such as Memetic Networks [10]. These socially inspired models differ on the general philosophical approach and in the specification of the individuals' behaviors and interactions. However, they share at least a common feature which is the underlying social network that regulates the information flow between the individuals.

In this paper, we investigate the benefits of creating a heterogeneous model composed of two socially inspired models: Memetic Networks and PSO. In particular, we provide a way for the two types of individuals to communicate seamlessly when solving the same problem and test this new model on several benchmark optimization tasks. We report that the heterogeneous model we developed outperforms the homogeneous models in highly multimodal scenarios.

The paper is organized as follows. Section II introduces relevant previous work on heterogeneous problem-solving systems. Section III briefly describes the PSO and the Memetic Networks model, respectively. Section IV presents the heterogeneous model we propose. Section V describes the experimental setup and section VI presents and elaborates on the results. Section VII concludes the paper and points out directions for further research.

II. Background and Motivation

Since no problem-solving algorithm is superior to others under any situation [11], researchers from the field of computational intelligence have resorted to hybridization as a strategy that "capitalizes on the respective strengths of the components of the hybrid computational intelligent system, and eliminate weaknesses of individual components" [12]. And hybrid algorithm typically combine local and global search operators from different heuristic models into a single model [13]–[18]. If the hybrid algorithm is devised to perform search using a population of collaborative search processes, then this population is usually composed of the same methods, in other words: it is homogeneous.

The literature about heterogeneous problem-solvers is scarce in comparison to hybrid homogeneous models. Thus, the purpose of this section is to present and integrate the most relevant works on heterogeneous populations of problem-solvers and describes our motivation to study them under a social bias.

Mühlenbein et al. [19] proposed one of the first heterogeneous models of problem-solving for minimization of real functions. In their work, a Parallel Genetic Algorithm was adapted to support mixed subpopulations. Such subpopulations seek to locate local minima of the function to be minimized. If a subpopulation does not progress after a fixed number of generations, a hill-climbing strategy is used then. Subpopulations can exchange information about the position of good local minima with other neighboring subpopulations also. Their algorithm presented superior results in *highly multimodal* scenarios. A rather similar approach was done by Alba et al. [20] whose work addressed the parallelization of the Gradual Distributed Real-Coded Genetic Algorithm. Their algorithm used a set of eight subpopulations organized in a cube topology having two faces for promoting exploration and exploitation. Though essentially heterogeneous, both works were restricted to Genetic Algorithms family.

Engelbrecht [21] proposed heterogeneous models where particles were allowed to use one search behavior available from a pool of five distinct PSO algorithms. He called the approach as Heterogeneous Particle Swarm Optimization (HPSO) and reported improvements over some homogeneous approaches. Montes de Oca et al. [22] carried experiments with a dual particle heterogeneous population. They explored the intra-swarm interaction among particles with different configurations and the effects of such interactions on the algorithms performance. They tested two forms of heterogeneity: update rule heterogeneity and model of influence heterogeneity. Their results points that heterogeneous swarms perform better than the worst homogeneous swarm in some cases and can outperform the best homogeneous model in other cases. A related work showed that heterogeneous Particle Swarm Optimization models scale better in the problem size than homogeneous PSO models [23]. These models are strictly associated with the Particle Swarm Optimization family though.

The *Lifecycle* [24] model presents a more broader approach for problem-solving with a heterogeneous population because it is not tied to a specific family of models. The Lifecycle's population is composed of individuals that have different search heuristics. An individual can perform search using operators from Genetic Algorithms, Particles Swarm Optimization or hill-climber models at a time. These individuals can change their category during the search according to a stagnation criteria. It is, however, not clear whether individuals from distinct categories are allowed to communicate seamlessly. For example, it is not clear whether a genetic algorithm individual can perform a crossover operation using a particle and a hill climber. The model was tested with the problem of function minimization using five benchmark real functions and the authors report improvements over the individual algorithms.

In another previous work, social scientists designed a heterogeneous system to better understand the role of diversity within the group. They report that a random group of intelligent problem-solvers outperforms a group of the best problem-solvers [25]. Although their experiments used the maximization of random functions as the problem [1], we believe this result is not only relevant from sociological point of view, but also from a computational point of view since a heterogeneous framework for problem-solving could help us to better exploit the vast number of problem-solving models developed over the years. Additionally, in a recent work Mason and Watts [27] found mismatching results from a problem-solving system composed of real human subjects and artificial problem-solving systems, warning that current artificial sociological systems based on a population of problem-solvers are insufficiently sophisticated and heterogeneous to reflect real human responses to changing circumstances.

[1]It has been argued that no problem in real world is random [26]

The study of heterogeneous problem-solving system is embodied with a higher level of difficulty then homogeneous systems for all those models need to collaborate together. Thus, communication plays a key role for the success of the collective. The main problems that arise in the investigation of such systems are: (1) the design of a communication medium that does not interfere with a model search process while allowing different models to exchange information and cooperate, this medium ought to be simple in order to avoid communication overhead; and (2) the complexity inherent of a heterogeneous system must be kept bearable to analysis. To tackle such problems, a sound approach would be the using of problem-solving systems where effective communication is as important as the individual performance, such problem-solving systems are called socially inspired systems of problem-solving. As our knowledge about heterogeneous problem-solving systems increases, more robust and efficient problem-solvers as well as more realistic systems can be built to test and predict real social phenomena. In light of these observations, we selected two socially inspired systems to start the endeavor of studying heterogeneous problem-solving systems: the Particle Swarm Optimization algorithm [5] (PSO) and the Memetic Networks [10] algorithms.

The next section will describe both PSO and Memetic Networks models before we introduce the initial heterogeneous model we propose.

III. Socially Inspired Models of Problem-solving

A. Particle Swarm Optimization

The model of Particle Swarm Optimization had its inception in the work of Heppner and Grenander [28]. In this work, the authors investigated simple social analogues of a population of interacting automatons which were modeled as flocks of birds searching for food. Kennedy and Eberhart [5] evolved their ideas intending to produce computational intelligence from the interactions of such automatons [6]. The resulting model was then used with relative success to solve optimization problems. In 2008, Poli [29] analyzed over 1100 publications concerning improvements and applications of the PSO model. He contributed to the understanding that the PSO model is a mature and quite successful model of social problem-solving.

The PSO model basically defines how particles will interact collectively and the individual effect of such interaction. At the collective level, particles are allowed to interact if there is a communication channel between. This level is usually modeled using undirected graphs, where particles are represented by nodes and channels by edges of the graph. The most common interaction between particles is through copy where a particle retrieves and then slightly changes the position [2] of the neighbor node which is faring the best. At the individual level, particles are concerned with their current and previous positions. Nonetheless, the objective of all particles is to find the position which minimizes — or maximizes — the output of the target function.

The collective and the individual influence interplay in the internal state of the particle characterizes a motion in search space of valid positions. This motion is described by a velocity vector and it is the mechanism by which a particle update its position. Previous studies recognized the importance of restraining the particle's velocity in some way [6]. Thus, Kennedy [30] developed the constriction factor mechanism. Equations (1) and (2) depict the process of velocity and position update considering these constriction factors.

$$\begin{aligned}&\vec{v}_{i+1} \leftarrow 0.7298(\vec{v}_i + U(0, 2.05)\\&\otimes(\vec{p}_i - \vec{x}_i) + U(0, 2.05)\\&\otimes(\vec{p}_g - \vec{x}_i))\end{aligned} \tag{1}$$

$$\vec{x}_{i+1} \leftarrow \vec{x}_i + v_{i+1} \tag{2}$$

where v_i and x_i represent the current velocity and position respectively of a particle, $U(0, 2.05)$ represents a random vector with components uniformly distributed between $[0, 2.05]$, $\otimes$ is the component-wise multiplication, p_g is the best previous position know by the best ranked neighbor — the collective influence, and p_i is the best previous position know by the own particle — the individual influence.

Yet, improvement was achieved when the particle's speed was limited in each dimension by the maximum range of such [31]. The PSO with constriction factors and with speed limits is considered the canonical version of PSO [6].

B. Memetic Networks

Memetic Networks is a recent metamodel inspired by how humans collectively solve problems [32]. It basically determines guidelines for the development of a system whose task is to search for the best solution to a given problem. The development of such system may be driven solely by the search task. However, one may also want to investigate the effects of modeling real social features into artificial social systems by objectively measuring the performance of the system along the search process.

The metamodel defines simple elements such as nodes, memes and links and also defines rules that state how these constructs relate to each other. The concepts behind elements and rules are problem invariant; however, their implementation is not: one needs to encode information about the problem's domain in order to build an effective problem-solving system. In this sense, the Memetic Networks metamodel is not a purely blackbox method for optimization, but rather a design guideline for the development of algorithms with embedded information about the problem domain.

[2]Or a candidate solution to the optimization problem in other words.

A *memenet* is the final construct when one assembles the Memetic Network's elements and then describes the rules by which the elements will interact. The memenet's main objective is to achieve better solutions in less iterations than independent and isolated nodes would achieve. To do so, nodes in a memenet are allowed to exchange information.

The computing mechanism behind Memetic Networks is represented by nodes. Each node contains a complete solution to the problem at hand. This solution can be (i) evaluated using a global evaluation function, and (ii) compared or ranked against other solutions. Nodes are responsible for retrieving, aggregating and processing *memes*, which are the minimal meaningful pieces that compose complete solutions. The meme concept is partially based in ideas from Dawkin's book, *The Selfish Gene* [33]. Links connect nodes and ultimately determine a supporting "social" network. If there is a link between two nodes, then these nodes are allowed to exchange memes.

The Memetic Network elements alone are not self-sufficient to perform search as it remains necessary to define how these elements will behave when assembled together. This is accomplished by three rules, which roughly capture how humans process information in a social context [32]. These rules detail how nodes choose to connect to each other, how information is retrieved from the social network and how individual nodes contribute to a solution being sought. Each rule is detailed in what follows.

Connection Rule — Defines how the nodes should connect, i.e. how links are to be established. This rule can be performed once before the search takes place, characterizing a static network topology. Alternatively, the connection rule can be executed during the search process so the network topology can evolve together with the system. As in other socially-inspired models (e.g. Particle Swarm Optimizers), there is currently no conclusive evidence about what topology is generally better for specific types of problems [6].

Aggregation Rule — This rule controls the interaction between connected nodes, that is, how nodes retrieve and aggregate information from their network neighbors. Typically, a node retrieve information [3] from better ranked neighbors only. After retrieving memes, a node must aggregate — i.e. combine — them to assemble a new solution. The aggregation rule must also describe algorithmically how these memes are to be combined. Aggregation is highly dependent on the network topology [4].

Appropriation Rule — In possession of the recently aggregated solution, a node may add some novelty to it. In this step, local information is added to the solution, allowing memenets to explore the search space, much in the same way mutation works in genetic algorithms. Appropriation can happen by simple random changes to the solution or by applying some

[3]Retrieval of information is considered noiseless. It remains an open question whether communication noise could be beneficial or not for population-based optimization problems.

[4]This feature is where socially-inspired optimization algorithms differ the most from Evolutionary Algorithms.

deterministic local search to it (in a similar fashion to Genetic Algorithms [1] and Memetic Algorithms [34] respectively).

Although the aggregation and the appropriation rules depend on the problem at hand, the structure of a Memetic Network is the same across different problems.

IV. The Socially Inspired Heterogeneous Model of Problem-solving

In this section, we will address the design decisions relative to the heterogeneous problem-solving system we propose and instantiate the model to the problem of real function minimization. Here we are concerned with the solution which has the least cost according to an objective function. A solution is encoded as an array of real numbers and for every valid array, there is an output from the function which we will be cost.

The initial design decision was select the right models to combine into the same population. The decision of choosing socially inspired models of problem-solving was addressed in section II but here will elaborate more on it. The PSO and Memetic Networks models rely on the concept of social network to regulate the internal information flow in a structured way. Such structure allow a node to retrieve and send information to as many sources as possible. Additionally, socially inspired models need a few modifications from their canonical form to work in collaboration, since communication and information processing are conceptually separated. This contrasts with the canonical GA, for instance, which communication is done indirectly trough crossover operations using two individuals.

In the heterogeneous system, the communication between a particle and a memetic node should be as seamlessly as possible in order to any avoid communication overhead that translating routines may impose. To facilitate such feature, all agents must encode the solution in the same way. For example, if memetic nodes encode their solutions as arrays of real values, particles shall encode their solutions as arrays of values as well. To solve the problem, we developed a simple interface that must be followed by all nodes, particles and memetic nodes, which is the following: a node must have a routine (1) to update its internal state by performing its respective search operations, (2) evaluate its current internal state, and (3) store the internal state if it leads to a better solution then any state. Algorithm 1 describes the structure of the proposed model including references to routines for population initialization and social network formation [5]. The *stop condition* can be a maximum number of iterations, running time, or a desired solution quality threshold.

The next decision is about the communication between particles and memetic nodes and vice versa. Each particle retrieves a solution from its best ranked neighbor while a memetic node retrieves multiple solutions from the set of better ranked neighbors. However, if a memetic node is the best ranked neighbor of a particle, it must provide its previous best position to the particle. Therefore, a memetic node must store its previous best position even if this was originally not sought in the Memetic Networks model. If a memetic node has a one or more particles in its neighboord, then it can retrieve their current solution directly.

[5]The network routine depicted is done only once, characterizing a static network

Algorithm 1: Structure of our heterogeneous system. Note that the $build\_network()$ sub procedure is performed once only and that the *stop condition* can be the number of iterations or a desired solution quality threshold.

```
Input : $Population_{size}$, $stopcondiction$, $Problem_{size}$
Output: $Solution_{best}$
Population ⟵ InitializePopulation ( $Population_{size}$, $Problem_{size}$ )
InitializeNetwork (Population)
repeat
    foreach $node$ ∈ Population do
        UpdateState ($node$)
        EvaluateState ($node$)
        if $node'scurrent_{score} < node'sbest_{score\_yet}$ then
            // Store new solution as best
               solution found so far by node.
until $stopcondition$
$Solution_{best}$ ⟵ RetrieveBestSolutionFrom (Population)
return $Solution_{best}$
```

It remains necessary to implement the search operators from both particles and memetic nodes trough the $UpdateState$ routine for the problem of function minimization. We will start by the particle $UpdateState$ routine which is depicted in Algorithm 2.

Algorithm 2: Update state procedure for a particle.

```
Input: $node$
$BestNeighbor$ ⟵ RetrieveBestNeighbor ($node$)
UpdateVelocity ($node$,$BestNeighbor$)
UpdatePosition ($node$)
```

Our implementation follows the canonical PSO form stated in Poli's review work [6]; more details can be found in the original work [31].

The $UpdateState$ routine of a memetic node is depicted in Algorithm 3.

Algorithm 3: Update state procedure for a memetic node.

```
Input: $node$
BetterNeighbors ⟵ RetrieveBetterNeighbors ($node$)
$aggregated_{solution}$ ⟵ Aggregate (BetterNeighbors)
Appropriate ( $aggregated_{solution}$)
```

The aggregation procedure retrieves and then combines memes from a set of solutions from better ranked neighbors to create a new solution. For real-function minimization this process is described in Algorithm 4 and work as follows: the memetic node generates an array of arrays called $MemePool$. The purpose of this structure is to store solutions that came from neighbors with better scores and also the node's own solution. Each component of the solution is replaced by a respective meme of the $MemePool$ list, that is a meme from the same position of the available arrays. All solutions from $MemePool$ have equal chances to be chosen.

Algorithm 4: The aggregation rule instantiated to minimization.

```
Input  : BetterNeighbors, $Problem_{size}$
Output: $Solution$
MemePool $\longleftarrow$ RetrieveSolutions (BetterNeighbors)
Append node's solution to MemePool
$Solution \longleftarrow \emptyset$
if CountSolutions (MemePool) $> 1$ then
    for $i \leftarrow 1$ to $Problem_{size}$ do
        $SolutionChosen \longleftarrow$
        RandomlyChooseASolutionFrom (MemePool)
        $Solution_i \longleftarrow SolutionChosen_i$
else // When the node has no better neighbor,
    its solution is kept.
    $Solution \longleftarrow CurrentNodeSolution$
return $Solution$
```

The appropriation procedure is the mechanism which allows nodes to exploit their current solution, that is perform small modifications to it. For real-function minimization a simple mechanism is portrayed in Algorithm 5. For each component of the solution, a valid meme is generated randomly by a uniform distribution and attributed with probability p. As with mutation probability in genetic algorithms, the p value should be small enough so that convergence is not drastically disrupted.

Algorithm 5: The appropriation rule instantied to minimization.

```
Input  : $Solution$, $Problem_{size}$, $P_{appropriation}$
Output: $Solution$
for $i \leftarrow 1$ to $Problem_{size}$ do
    With $P_{appropriation}$ probability:
        $Solution_i \longleftarrow$ RandomlyChooseMeme ()
return $Solution$
```

V. Experimental Setup

This section provides the methodology used to validate the model. For the purpose of this paper, we used five benchmark functions for the problem of real-function minimization. These functions were: Ackley (f_1), Griewank (f_2), Rastrigin (f_3), Rosenbrock (f_4), and Sphere (f_5). Three functions are multimodal, f_1 to f_3 (Ackley, Griewank and Rastrigin) while two functions, f_4 and f_5, are unimodal. Functions f_2 and f_4 are not separable. Solutions are represented by arrays of floating point numbers. The number of dimensions of the solutions, i.e. the size of the arrays, was set to 100. The float precision used was of eight decimal digits for both solution components and cost value.

Since four functions have the global minimum at the very center of the search space [6], all particles and nodes were initialized following a uniform distribution inside a limited search range to avoid any advantage a model may have over the other [35]. The specific information about these functions is described in Table I.

The total population size was set to 30, distributed as half particles and half memetic nodes. The distribution of the population remained static along 10,000 iterations that were used as the *stop condition*.

The network topology chosen for all experiments was a 6×5 "*Von Neumann*" neighborhood. In this setting, the population is arranged in a rectangular matrix where each node is connected to one individual above, one below, and one at both sides, wrapping the edges of the matrix [36]. Previous works found supporting evidence that this topology is better suited to the kind of problem we are tackling [9], [36], [37]. Moreover, this topology is balanced in terms of density and average path length when compared to more traditional network topologies — like the fully connected network or the ring network. Figure 1 provides a three dimensional representation of the topology used. All links between nodes or particles were static and were built under no specific assumptions[7].

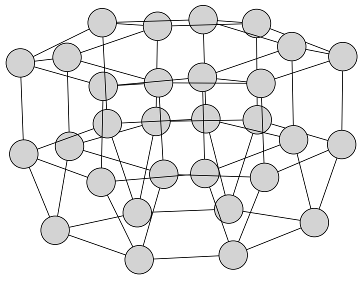


Fig. 1. The 6×5 periodic grid graph used as the network topology in our experiments.

In each experiment, the heterogeneous model was compared against the original PSO and Memetic Network models. Also, an improved version of PSO algorithm, the Fully Informed Particle Swarm (FIPS) algorithm [9] was included with comparative purposes. The FIPS algorithm improves over the original PSO by gathering information of all neighbors the particle have and not just the best one. Each experiment was repeated 50 times and the results averaged. The p probability used by memetic nodes in appropriation was 0.02.

[6]The exception is the Rosenbrock Function.

[7]One could assume that spatial position or initial score could be used to establish links between related nodes.

TABLE I
BENCHMARK FUNCTIONS

Name	Definition	Initialization Range $[x_{min}, x_{max}]^n$	Search Range $[x_{min}, x_{max}]^n$
Ackley	$20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$	$[15.0, 30.0]^n$	$[-30.0, 30.0]^n$
Griewank	$\frac{1}{4000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos(\frac{x_i-100}{\sqrt{i}}) + 1$	$[300.0, 600.0]^n$	$[-600.0, 600.0]^n$
Rastrigin	$\sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	$[2.5, 5.1]^n$	$[-5.1, 5.1]^n$
Rosenbrock	$\sum_{i=1}^{n-1}\left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right)$	$[15.0, 30.0]^n$	$[-30.0, 30.0]^n$
Sphere	$\sum_{i=1}^{n} x_i^2$	$[50.0, 100.0]^n$	$[-100.0, 100.0]^n$

VI. EMPIRICAL RESULTS

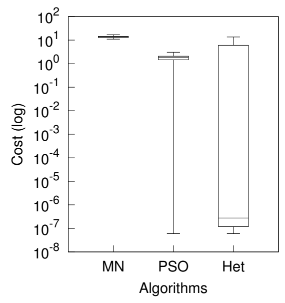
Figure 2 presents the evolution of the best solution found by all algorithms averaged over the 50 independent trials. Our heterogeneous model is represented by *Hybrid*, the original memenet by *MN*, the original PSO by *PSO* and the Fully Informed Particle Swarm by *FIPS*.

The heterogeneous model with the population distribution of half particles and half memetic nodes presented no statistically difference from the best performing model in the Griewank, Rosenbrock and Sphere functions. In such cases, the PSO was the best performing model that composed the heterogeneous models. This result is positive because if we suppose that one is to solve a novel problem that no exploitable structure is known at moment, then a heterogeneous model can perform sometimes as better than the worst performer or as good as the best performing model. The insight of which model performs best is a valuable one, for it allows the designer to concentrate the efforts into a single model. Moreover, such result confirms previous findings by Montes de Oca et al. [22]. The extended version of PSO, the FIPS algorithm, presented slightly advantage in cited functions.

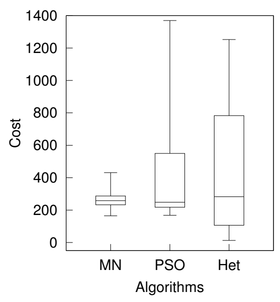
In the Ackley and Rastrigin functions, both highly multimodal, the heterogeneous model presented a remarkable advantage. We performed unpaired t-tests comparing all the 50 last generations of the heterogeneous model against PSO, Memetic Networks and FIPS, finding significant results ($p = 0$,$t < 0$). The hypothesis for such result is the heterogeneous model presents a higher solution diversity in these functions. A higher solution diversity translates into a more exploratory and less exploitative search behavior, which in turns increases the likelihood of finding a good solution. To test this hypothesis, we extracted the 30 individuals from the last generation of all the 50 runs and analyzed the distribution of the population according to their solution quality. Figure 3 presents the boxplots of this distribution.

As hypothesized, the heterogeneous model presented the highest solution diversity in the tested functions: Ackley and Rastrigin. This result was expected because the two sorts of individuals [8] that compose our heterogeneous model can be seen as two driving forces that guide the search to different regions. It is likely that particles may find a specific region of the search space more promising than memetic nodes as well as the opposite. From this lack of agreement, a favorable search pattern may have arisen in these two functions. Nevertheless, we are further investigating the effect and developing a visualization tools to analyze the population dynamics across all generations. Another result was that the pure PSO model eventually found the global optimum in the Ackley function — this can be seen as the lower whisker mark. However, the expected result is much above it. The pure Memetic Networks model achieved similar results in most runs, this fact is depicted by the equally distributed quartiles from the median.

[8] Represented by their search heuristics.

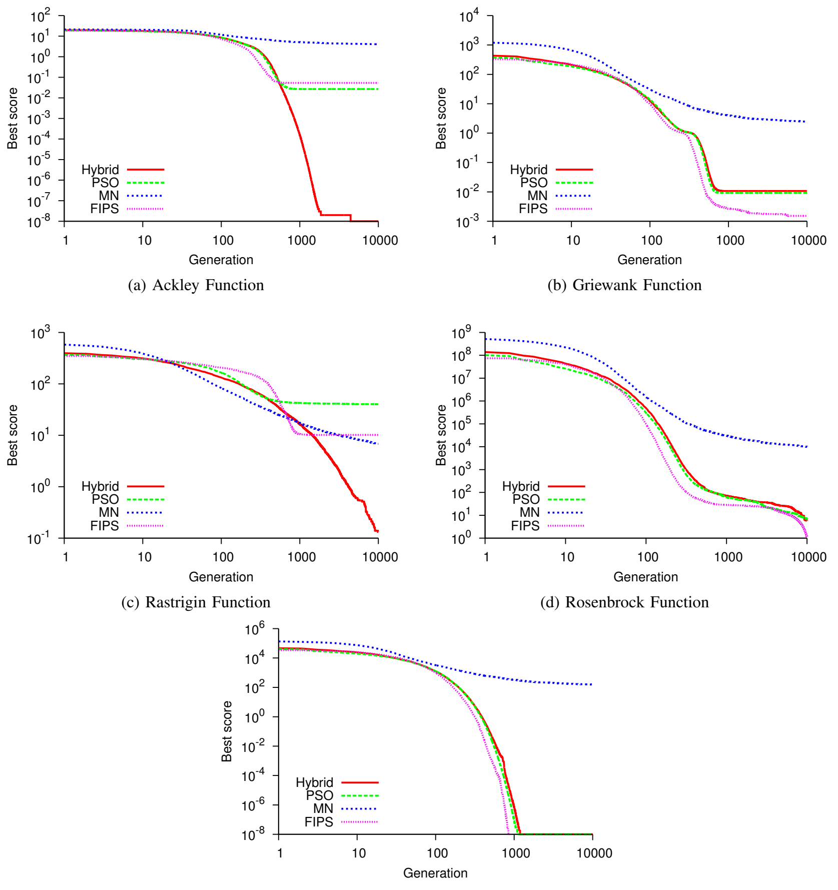


(a) Ackley Function



(b) Rastrigin Function

Fig. 3. Solution quality distribution extracted the 30 individuals from the last generation of all the 50 runs. The heterogeneous model presented the highest solution diversity among all as we hypothesized.



(a) Ackley Function

(b) Griewank Function

(c) Rastrigin Function

(d) Rosenbrock Function

(e) Sphere Function

Fig. 2. The evolution of the best solution found by all algorithms averaged over the 50 independent trials. Our heterogeneous model with the population distribution of half particles and half memetic nodes presents better results for the Ackley and Rastrigin Functions after one thousand iterations. In the other cases, the heterogeneous model was equivalent to PSO and FIPS. Both axis are in logarithmic scale.

VII. Conclusions

In this paper, we have proposed an initial heterogeneous problem-solving system based on socially inspired models. More specifically, we were concerned with the investigation of the benefits of a heterogeneous problem-solving system composed of two socially inspired models: Particle Swarm Optimization and Memetic Networks. Our approach did not change how individuals behave, as it is commonly tackled by hybridization mechanisms, rather it allowed different individuals, with different search heuristics, to communicate and collaborate to solve the problem at hand — which we framed as minimization of five real valued functions.

These individuals were connected by links in Von Neumann network and they were allowed to communicate information about their solution through a communication medium designed for that. Our results showed that at one hand the heterogeneous system was able to outperform the Memetic Network model in all scenarios and considerably outperform the PSO model in two of the most difficult (highly multimodal) scenarios. On the other hand, in simpler scenarios (mostly unimodal functions), the heterogeneous model was able to perform equally well as the homogeneous models. We further investigated why our heterogeneous model performed well in highly multimodal scenarios and elaborated the hypothesis that heterogeneous models increase diversity of solutions. From the analysis of distribution of solution quality we were able to confirm our hypothesis. Together, these results are evidence that heterogeneous systems are promising problem-solving tools but their inner working mechanisms must still be better understood.

As future work, we intend to devise meta-heuristics capable of inferring which search behavior is the best for a given agent or set of agents. To do so, we need to investigate mechanisms that can detect problem features and switch the agent search behavior accordingly. Such endeavor explicits the need for a statistical framework that would allow one to assess which kind of agent contributed the most and when it contributed along the search. We are also interested to assess the network topology influence in this algorithm and investigate the other forms of heterogeneity, as solution encoding, for example.

Acknowledgments.

This work is partly funded by the Federal Agencies CNPq and CAPES/Brazil.

References

[1] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[2] M. Mitchell, *An Introduction to Genetic Algorithms*, ser. Complex Adaptive Systems. Mit Press, 1998.

[3] J. D. Farmer, N. H. Packard, and A. S. Perelson, "The immune system, adaptation, and machine learning," *Physica D*, vol. 2, no. 1-3, pp. 187–204, oct 1986.

[4] S. Forrest, S. A. Hofmeyr, and A. Somayaji, "Computer immunology," *Communications of the ACM*, vol. 40, no. 10, pp. 88–96, oct 1997.

[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[6] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, Aug. 2007.

[7] P. Moscato, "A gentle introduction to memetic algorithms," in *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2003, pp. 105–144.

[8] R. G. Reynolds, "An introduction to cultural algorithms," in *Proceedings of the 3rd Annual Conference on Evolutionary Programming*. World Scientic, 1994, pp. 131–139.

[9] R. Mendes, J. Kennedy, and J. Neves, "Watch thy neighbor or how the swarm can learn from its environment," in *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706)*. IEEE, 2003, pp. 88–94.

[10] R. M. Araujo and L. C. Lamb, "Memetic networks: analyzing the effects of network properties in multi-agent performance," in *Proceedings of the 23rd International conference on Artificial intelligence*, ser. AAAI'08, vol. 1. AAAI Press, 2008, pp. 3–8.

[11] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67 –82, apr 1997.

[12] A. Engelbrecht, *Computational Intelligence: An Introduction*. John Wiley & Sons, 2007.

[13] C.-J. Lin, C.-H. Chen, and C.-T. Lin, "A Hybrid of Cooperative Particle Swarm Optimization and Cultural Algorithm for Neural Fuzzy Networks and Its Prediction Applications," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 39, no. 1, pp. 55–68, Jan. 2009.

[14] S. H. Ling, H. H. C. Iu, F. H. F. Leung, and K. Y. Chan, "Improved Hybrid Particle Swarm Optimized Wavelet Neural Network for Modeling the Development of Fluid Dispensing for Electronic Packaging," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 9, pp. 3447–3460, Sep. 2008.

[15] S. Naka, T. Genji, T. Yura, and Y. Fukuyama, "A hybrid particle swarm optimization for distribution state estimation," *IEEE Transactions on Power Systems*, vol. 18, no. 1, pp. 60–68, Feb. 2003.

[16] J.-H. Seo, C.-H. Im, C.-G. Heo, J.-K. Kim, H.-K. Jung, and C.-G. Lee, "Multimodal function optimization based on particle swarm optimization," *IEEE Transactions on Magnetics*, vol. 42, no. 4, pp. 1095–1098, Apr. 2006.

[17] T. Ting, K. Wong, and C. Chung, "Hybrid constrained genetic algorithm/particle swarm optimisation load flow algorithm," *IET Generation, Transmission & Distribution*, vol. 2, no. 6, p. 800, 2008.

[18] Y. Wang and L. Li, "Heterogeneous Redundancy Allocation for Series-Parallel Multi-State Systems Using Hybrid Particle Swarm Optimization and Local Search," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 42, no. 2, pp. 464–474, Mar. 2012.

[19] H. Mühlenbein, M. Schomisch, and J. Born, "The parallel genetic algorithm as function optimizer," *Parallel Computing*, vol. 17, no. 6-7, pp. 619–632, Sep. 1991.

[20] E. Alba, F. Luna, A. Nebro, and J. Troya, "Parallel heterogeneous genetic algorithms for continuous optimization," *Parallel Computing*, vol. 30, no. 5-6, pp. 699–719, May 2004.

[21] A. Engelbrecht, "Heterogeneous Particle Swarm Optimization," in *Swarm Intelligence*, ser. Lecture Notes in Computer Science, M. Dorigo, M. Birattari, G. Di Caro, R. Doursat, A. Engelbrecht, D. Floreano, L. Gambardella, R. Groß, E. Sahin, H. Sayama, and T. Stützle, Eds. Springer Berlin / Heidelberg, 2010, pp. 191–202.

[22] M. A. Montes de Oca, J. Pena, T. Stutzle, C. Pinciroli, and M. Dorigo, "Heterogeneous particle swarm optimizers," in *2009 IEEE Congress on Evolutionary Computation*. IEEE, May 2009, pp. 698–705.

[23] A. P. Engelbrecht, "Scalability of a heterogeneous particle swarm optimizer," in *2011 IEEE Symposium on Swarm Intelligence*. IEEE, Apr. 2011, pp. 1–8.

[24] T. Krink and M. Løvbjerg, "The lifecycle model: Combining particle swarm optimisation, genetic algorithms and hillclimbers," in *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, ser. PPSN VII. London, UK, UK: Springer-Verlag, 2002, pp. 621–630.

[25] L. Hong and S. E. Page, "Groups of diverse problem solvers can outperform groups of high-ability problem solvers." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 46, pp. 16 385–16 389, Nov. 2004.

[26] R. Mendes, J. Kennedy, and J. Neves, "The Fully Informed Particle Swarm: Simpler, Maybe Better," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, Jun. 2004.

[27] W. Mason and D. J. Watts, "Collaborative learning in networks." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 109, no. 3, pp. 764–9, Jan. 2012.

[28] F. Heppner and U. Grenander, "A stochastic nonlinear model for coordinated bird flocks," in *The Ubiquity of Chaos*, S. Krasner, Ed. AAAS Publications, 1990, ch. 19, pp. 233–238.

[29] R. Poli, "Analysis of the publications on the applications of particle swarm optimisation," *Journal of Artificial Evolution and Applications*, vol. 2008, pp. 1–10, Jan. 2008.

[30] J. Kennedy, "The behavior of particles," in *Proceedings of the 7th International Conference on Evolutionary Programming VII*, ser. EP '98. London, UK, UK: Springer-Verlag, 1998, pp. 581–589.

[31] R. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC)*, vol. 1, 2000, pp. 84–88.

[32] R. M. Araujo and L. C. Lamb, "On the Effects of Network Structure in Population-Based Optimization," in *20th IEEE International Conference on Tools with Artificial Intelligence*. Washington, DC, USA: IEEE, Nov. 2008, pp. 268–271.

[33] R. Dawkins, *The Selfish Gene*. Oxford University Press, 2006.

[34] P. Moscato, "Memetic algorithms: A short introduction," in *New Ideas in Optimization*. McGraw-Hill, 1999.

[35] P. J. Angeline, "Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences," in *Proceedings of the 7th International Conference on Evolutionary Programming*, ser. EP '98. London, UK: Springer-Verlag, 1998, pp. 601–610.

[36] J. Kennedy and R. Mendes, "Neighborhood topologies in fully informed and best-of-neighborhood particle swarms," *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 36, no. 4, pp. 515–519, Jul. 2006.

[37] ——, "Population structure and particle swarm performance," in *Proceedings of the Congress on Evolutionary Computation*, vol. 2. IEEE, 2002, pp. 1671–1676.

REFERENCES

AHN, L. von et al. reCAPTCHA: human-based character recognition via web security measures. **Science**, [S.l.], v.321, n.5895, p.1465–1468, 2008.

ALBERT, R.; JEONG, H.; BARABáSI, A.-L. Error and attack tolerance of complex networks. **Nature**, [S.l.], v.406, n.6794, p.378–382, 2000.

ARAÚJO, R. M.; LAMB, L. C. Memetic networks: analyzing the effects of network properties in multi-agent performance. In: INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE. **Anais...** AAAI Press, 2008. p.3–8. (AAAI'08, v.1).

ARAÚJO, R. M.; LAMB, L. C. **Memetic networks** : problem-solving with social network models. 2010. Tese (Doutorado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul, Instituto de Informática, Programa de Pós Graduação em Computação, Porto Alegre.

BANERJEE, A. V. A Simple Model of Herd Behavior. **The Quarterly Journal of Economics**, [S.l.], v.107, n.3, p.797–817, 1992.

BARABÁSI, A. **Linked**: how everything is connected to everything else and what it means for business, science, and everyday life. [S.l.]: Plume, 2003. (Plume book).

BARABÁSI, A.-L.; ALBERT, R. Emergence of Scaling in Random Networks. **Science**, [S.l.], v.286, n.5439, p.509–512, 1999.

BARNETT, E.; CASPER, M. A definition of "social environment". **American journal of public health**, [S.l.], v.91, n.3, p.465, Mar. 2001.

BONACICH, P. Power and Centrality: a family of measures. **American Journal of Sociology**, [S.l.], v.92, n.5, p.1170–1182, 1987.

BURT, R. **Structural Holes**: the social structure of competition. [S.l.]: Harvard University Press, 1995.

DAWKINS, R. **The Selfish Gene**. [S.l.]: Oxford University Press, 2006.

EASLEY, D.; KLEINBERG, J. **Networks, Crowds, and Markets**: reasoning about a highly connected world. [S.l.]: Cambridge University Press, 2010.

ENGELBRECHT, A. Heterogeneous Particle Swarm Optimization. In: DORIGO, M. et al. (Ed.). **Swarm Intelligence**. [S.l.]: Springer Berlin / Heidelberg, 2010. p.191–202. (Lecture Notes in Computer Science).

EPSTEIN, J. **Generative Social Science**: studies in agent-based computational modeling. [S.l.]: Princeton University Press, 2011. (Princeton Studies in Complexity).

FISKE, S. T.; TAYLOR, S. E. **Social Cognition**. 2.ed. [S.l.]: McGraw-Hill, 1991.

FREEMAN, L. C. A Set of Measures of Centrality Based on Betweenness. **Sociometry**, [S.l.], v.40, n.1, p.35–41, Mar. 1977.

FREEMAN, L. C. Centrality in social networks: conceptual clarification. **Social Networks**, [S.l.], v.1, n.3, p.215–239, 1979.

GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization and Machine Learning**. 1st.ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

HONG, L.; PAGE, S. E. Groups of diverse problem solvers can outperform groups of high-ability problem solvers. **Proceedings of the National Academy of Sciences of the United States of America**, [S.l.], v.101, n.46, p.16385–9, Nov. 2004.

KEARNS, M. Experiments in social computation. **Communications of the ACM**, [S.l.], v.55, n.10, p.56–67, Oct. 2012.

KING, I.; LI, J.; CHAN, K. T. A brief survey of computational approaches in Social Computing. In: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS. **Anais...** IEEE, 2009. p.1625–1632.

KRUSCHKE, J. K. Bayesian Estimation Supersedes the t Test. **Journal of experimental psychology. General**, [S.l.], July 2012.

LAZER, D. et al. Computational social science. **Science**, [S.l.], v.323, n.5915, p.721–3, Feb. 2009.

LEVINE, J.; RESNICK, L. SOCIAL FOUNDATIONS OF COGNITION. **ANNUAL REVIEW OF PSYCHOLOGY**, 4139 EL CAMINO WAY, PO BOX 10139, PALO ALTO, CA 94303-0139, v.44, p.585–612, 1993.

HEDSTRÖM, P.; BEARMAN, P. (Ed.). **Social Dynamics from the Bottom up**: agent-based models of social interaction. [S.l.]: Oxford University Press, 2009. (Oxford Handbooks in Political Science & International Relations Series).

MASON, W.; WATTS, D. J. Collaborative learning in networks. **Proceedings of the National Academy of Sciences of the United States of America**, [S.l.], v.109, n.3, p.764–9, Jan. 2012.

MATSUMOTO, M.; NISHIMURA, T. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. **ACM Trans. Model. Comput. Simul.**, New York, NY, USA, v.8, n.1, p.3–30, Jan. 1998.

MCPHERSON, M.; SMITH-LOVIN, L.; COOK, J. M. Birds of a Feather: homophily in social networks. **Annual Review of Sociology**, [S.l.], v.27, p.415–444, 2001.

MENDES, R. **Population Topologies and Their Influence in Particle Swarm Performance**. 2004. Tese (Doutorado em Ciência da Computação) — Universidade do Minho, Escola de Engenharia, Minho, Portugal.

MOSCATO, P. Memetic Algorithms: a short introduction. In: **New Ideas in Optimization**. [S.l.]: McGraw-Hill, 1999.

NEWELL, A.; SIMON, H. A. Computer science as empirical inquiry: symbols and search. **Commun. ACM**, New York, NY, USA, v.19, n.3, p.113–126, Mar. 1976.

NEWELL, A.; SIMON, H. **Human problem solving**. [S.l.]: Prentice-Hall, 1972.

NOBLE, D.; ARAÚJO, R.; LAMB, L. Investigating a Socially Inspired Heterogeneous System of Problem Solving Agents. In: IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS, Barcelona, Spain. **Anais...** [S.l.: s.n.], 2013.

NOBLE, D.; DORN, M.; LAMB, L. A Memetic Network-based Approach To Search The 3-d Protein Conformational Space. In: INTERNATIONAL CONFERENCE OF THE BRAZILIAN ASSOCIATION FOR BIOINFORMATICS AND COMPUTATIONAL BIOLOGY. **Anais...** [S.l.: s.n.], 2012.

NOBLE, D. et al. Two Novel Algorithms for High Quality Motion Estimation in High Definition Video Sequences. In: GRAPHICS, PATTERNS AND IMAGES (SIBGRAPI), 2011 24TH SIBGRAPI CONFERENCE ON. **Anais...** [S.l.: s.n.], 2011. p.197 –204.

NOWAK, M. A. Five rules for the evolution of cooperation. **Science (New York, N.Y.)**, [S.l.], v.314, n.5805, p.1560–3, Dec. 2006.

OCA, M. A. M. et al. Heterogeneous particle swarm optimizers. In: IEEE CONGRESS ON EVOLUTIONARY COMPUTATION, 2009. **Anais...** IEEE, 2009. p.698–705.

PEARL, J. **Heuristics**: intelligent search strategies for computer problem solving. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1984.

POLI, R.; KENNEDY, J.; BLACKWELL, T. Particle swarm optimization. **Swarm Intelligence**, [S.l.], v.1, n.1, p.33–57, Aug. 2007.

REYNOLDS, R.; CHUNG, C. Knowledge-based self-adaptation in evolutionary programming using cultural algorithms. In: IEEE INTERNATIONAL CONFERENCE ON EVOLUTIONARY COMPUTATION. **Anais...** IEEE, 1997. p.71–76.

ROSSUM, G. v.; DRAKE, F. L. **The Python Language Reference**. 2012.

RUSSELL, S.; NORVIG, P. **Artificial intelligence**: a modern approach. [S.l.]: Pearson Education/Prentice Hall, 2010. (Prentice Hall series in artificial intelligence).

SCHELLING, T. **Micromotives and Macrobehavior**. [S.l.]: Norton, 1978. (Fels Lectures on Public Policy Analysis).

SCHWARZ, N. Warmer and More Social: recent developments in cognitive social psychology. **Annual Review of Sociology**, [S.l.], v.24, p.239–264, 1998.

GILBERT, D.; FISKE, S.; LINDZEY, G. (Ed.). **The social being in social psychology**. [S.l.]: McGraw-Hill, 1998. (The Handbook of Social Psychology).

THIERENS, D. Selection Schemes, Elitist Recombination, and Selection Intensity. In: INTERNATIONAL CONFERENCE ON GENETIC ALGORITHMS. **Anais...** Morgan Kaufmann, 1998. p.152–159.

VALSINER, J.; VEER, R. On the Social Nature of Human Cognition: an analysis of the shared intellectual roots of george herbert mead and lev vygotsky. **Journal for the Theory of Social Behaviour**, [S.l.], v.18, n.1, p.117–136, Mar. 1988.

WANG, F.-Y. et al. Social Computing: from social informatics to social intelligence. **IEEE Intelligent Systems**, [S.l.], v.22, n.2, p.79–83, Mar. 2007.

WASSERMAN, S.; FAUST, K. **Social Network Analysis**: methods and applications. [S.l.]: Cambridge University Press, 1994. (Structural Analysis in the Social Sciences).

WATTS, D. J.; STROGATZ, S. H. Collective dynamics of 'small-world' networks. **Nature**, [S.l.], v.393, n.6684, p.440–442, June 1998.

WEISBERG, R. **Creativity**: understanding innovation in problem solving, science, invention, and the arts. [S.l.]: Wiley, 2006.