Fuzzy Decision Making in a Globalization Process

Pablo C. Rovarini Díaz
Universidad Tecnológica Nacional, FRT Departamento de Sistemas
Tucumán, Argentina, 4000
provarini@gmail.com

and

María de las Mercedes Cerviño
Universidad Tecnológica Nacional, FRT Departamento de Ciencias Básicas
Tucumán, Argentina, 4000
mercervino@gmail.com

Proyecto de Investigación UTILIZACION DE REDES SOCIOECONÓMICAS EN TOMA DE DECISIONES

Abstract

In a social context, Decision Making Support Systems use an aggregation mechanism to combine the input of individual characteristics as a way of reaching collective decision. In this paper we present our ideas about how these systems can achieve such goal in huge social nets (millions of agents). With this goal in mind, we have conjugated the power of Fuzzy Logic, Evolutionary Algorithms and Swarm Algorithms in a framework for obtaining a Collective Decision Making, in order to use simultaneously agent's specialization and connectivity. This framework incorporates different algorithms for the aggregation step.

Key Words: Social nets, Collective Decision Making, fuzzy logic, evolution, swarm.

Resumen

Los sistemas de soporte a la Toma de Decisiones en contextos sociales deben tomar a su cargo la agregación de características individuales para general soluciones colectivas. En este trabajo presentamos nuestras ideas de cómo se puede lograr tal objetivo en redes sociales grandes (millones de agentes). Con esta finalidad, hemos conjugado la potencia computacional de la Lógica Fuzzy y Algoritmos Swarm, conformando un marco referencial para la obtención de Toma de Decisiones Colectivas, considerando la especialización y grado de conectividad de cada agente, permitiendo además el uso de diferentes algoritmos para la agregación.

Palabras claves: Redes Sociales, Toma de Decisiones Colectivas, lógica fuzzy, evolución, swarm.

1. Introduction

George Modelsky makes clear the importance of globalization:

"Globalization is sometimes described as the defining feature of our current era"

It regards a diachronic process, and therefore a historic process. Its understanding requires blasts from the past. Our intention is to present the process of globalization as part of an evolutionary process, if we acknowledge evolution as a succession of trail-error which result in a progressive accumulation of knowledge.
An evolutionary focus on globalization must have the following features:

- Holistic, taking into account as unit of the analysis agents, particularly humans beings, and generations as the unity of evolutionary time.
- To take a global change obeying just a few simple laws that act over a set of learning process, nested and synchronized, made of successive iterations from a Darwinian algorithm (variation, selection, cooperation, and amplification).
- Multimensional, resulting in Multidisciplinarity.
- Possible to be contrasted with evidence of real facts, in a way that allows us to see the process of globalization as an integral set-about of the humankind, the latter being composed of sub processes and the making up of new way of thinking.

Technologically, this process leads to ephemeralization, term coined by R.B. Fuller. It refers to the ability of people to use technological advances to continuously do more with less. Fuller's vision was that ephemeralization will result in ever-increasing standards of living for an ever-growing population despite finite resources. Fuller saw ephemeralization as an inevitable trend in human development
Let us consider a society made up by agents, starting from global connectivity which increases in time, and which in turn increases the interaction between them. In consequence, increased possibilities of conflicts may happen. In order to decrease them, it is important to create institutions acting as mediators, who will have the need of collective decision making. Mediators must take decisions that will satisfy, at least in a way, every agent involved. For example, the stirmergy or unconscious collaboration between agents uses the shared environment - such as the Internet - so as to allow the agent to learn (in a self-organized way).

2. Social Networks

Social networks are used to show the existing relationships among agents of a population. We are going to representate a Social Network by means of a multidimensional graph with heterogeneous branches and nodes. Social networks ontology defines the type of nodes and branches (seen as relations). We demand that the graph must be directed, tagged, and with associated weight. Formally G = < N, R >, where N represent nodes, and R its branches, in a way that $R \subseteq N \times N$ and if $r \in R$, then $r_{k,h}^{trust} \in \mathbb{R}$ connects the node a_k with the node a_h according to a pre-defined semantics δ, with $\delta \in \Sigma^*$, where Σ is an alphabet and Σ* the Kleene star over Σ. In this paper, we will consider the set N divided into four disjoint sets of nodes. Then we will see a G node as a compound structure formed by:

- $A \subseteq N$: set of all agents. Generic element a_i
- $D \subseteq N$: set of all domains. Generic element d_i
- $P \subseteq N$: set of all problems. Generic element p_i
- $S \subseteq N$: set of all solutions for problems in P. Generic element s_i

With:

$$N = A \cup D \cup P \cup S$$
$$A \cap D \cap P \cap S = \emptyset$$

Each node $a_i \in A$ is associated with a set of domains used to grade its social relationship and problems. If $a_i(D)$ represents the associated domains with a_i:

$$D = \cup_{\forall i} \; a_i(D)$$
$$a_i(D) \cap a_j(D) = \emptyset \quad ; \quad i \neq j$$

Every problem p_i has an associated set of solutions (eventually empty) p_i (S) such that:

$$S = \cup_{\forall i}\ p_i\ (S)$$
$$p_i\ (S)\ \cap\ p_j\ (S) = \emptyset\ \ ;\ \ i \neq j$$

2.1 Trust-Based Social Networks

Be a trust-based social network (TBSN) a graph representing the inter-agents relationships in a social space. It is based on a semantic relation δ = *trust*, which refers to a trust relation about decision making. If a_k lacks expertise and believes that another agent a_h will make a better decision then a branch is created from node a_k to node a_h with a label $r_{k,h}^{trust}$, as it is shown in figure 1 (hence, there´s the need of a directed graph, which doesn´t have to have a symmetric relationship). Marko Rodriguez [4] proposes the use of conditional probabilities so as to find $r_{k,h}^{trust}$ quantitatively, considering only one domain (contextual independent):

$$r_{k,h}^{trust} = p\ (a_h is\ good \mid a_k\ trust\ a_h)$$

The above equation demonstrates that a_k believes in the ability of a_h to making a good decision based on previous knowledge about a_h behavior. If the agents involved are consider multidimensional, with different believed and abilities over several domains, we lose the domain in which a_k trusts in a_h. It is necessary to face the problem by assigning labels of trust according to the domains. Therefore, Rodriguez makes use of a more complex conditional probability:

$$r_{k,h}^{trust} = p\ (a_h is\ good\ in\ domain\ d_l \mid a_k\ trust\ a_h\ in\ domain\ d_l)$$

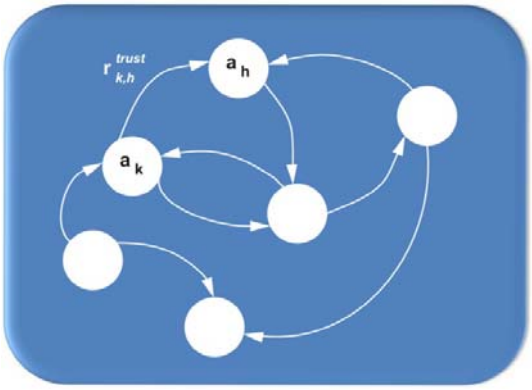

Fig. 1 – TBSN with trust relation

They allow us to work with two models of TBSN:

- Simple Domain Models (SDM): the level of trust has only a single context.
- Multiple Domain Models (MDM): the level of trust is sensitive to the context.

SDM do not require the construction of domains, and it presents only one graph which indicates every quantitative trust values of D. MDM allow an agent to specify a problem and its domain d. If #D is the cardinality of D, then the graph will have #D separate parts. A general scheme is shown in figure 2.
Our proposal differs in many aspects. The first real fact to take into account is that the trust level of an agent in another is, in a good deal, *subjective*, using previous knowledge extracted of a DB (about the historic behavior, colleague's references, management or governmental reports, or somebody you are familiar with, etc.).
Zadeh [8] suggested that the purpose of the Fuzzy Set Theory was the representation of how the human mind perceives and manipulates the information, if is adequate to take as a premise the fact that human beings frequently use fuzzy concepts when they perceive the surrounding environment and when they are thinking about the search of solutions to their problems. While this is happening some linguistic variables may appear, which in turn may have modifiers such as *good, high, similar, very, enough, close, near*, etc. We accept that one of the most representative characteristics of the human thought is the one of achieving a synthesis of the information in levels of fuzzy sets to describe linguistically particular situations. The ideas that impregnate the fuzzy sets and their membership values provide us with an excellent model for complex and very difficult concepts, which are very hard to cover with other

representations, being completely efficient in representing the necessary knowledge to achieve accurate and effective decisions. The dynamic processes in our brain are very complex, which make the process of accurate decision making, using an heterogeneous mixture of modifiers and its transformation to numeric values, results epistemologically impossible unless we treat it by means of appropriate tools, as fuzzy logic.

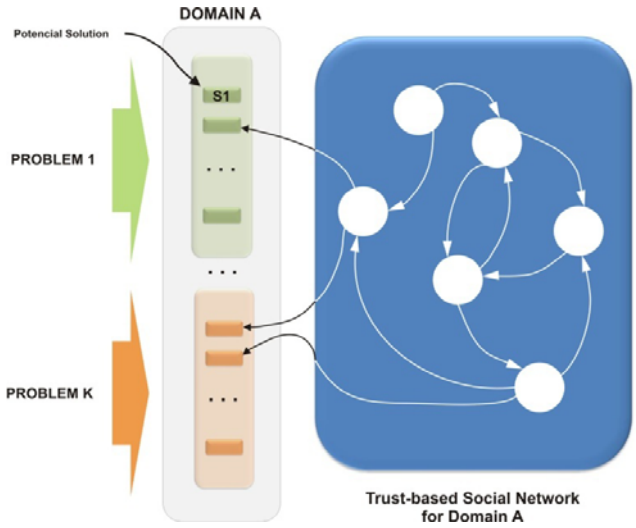

Fig. 2 – General schema (quadripartite model)

We are going to describe a methodology that allows obtaining crisp values for the involved grades of trust, from a restriction over the number of the characteristics that an agent uses when he wishes to make an evaluation of trust in other agent. We will take only two: 1) information about the success achieved by others agents in previous decision making, and 2) individual appreciation of the degree of expertise. This restriction is easily dropped by extending the number of conditions to a value which is arbitrarily large without making any changes in the methodology.
Let us supposed that agent a_k wants to find his level of trust with respect to agent a_h by calling it $r_{k,h}^{trust}$. In order to do this, it starts from M previous decision making examples carried out by a_h and taken from a DB, with table structure as shown in Table 1[1]. First, we are going to consider a SDM model, so we are not going to distinguish the context (all problems belong to the same domain).

PROBLEM	a_h DECISION	**parameter1**	**parameter2**	**. . .**	**parameterM**
p_1	s^h_1	s_{11}	s_{12}		s_{1M}
p_2	s^h_2	s_{21}	s_{22}		s_{2M}
. . .	. . .	. . .	. . .		. . .
p_M	s^h_M	s_{M1}	s_{M2}		s_{MM}

Table 1

In a Fuzzy Logic type 1, we will propose as membership functions those shown in the figure 3, assigning to the horizontal axis a normalized variable. The next step is evaluate each one of the parameters by means of the selection of one of the five membership functions (seeing them as reference, but possible to change depending on the particularities of the problem at hand, as much in number as in shape), followed by an aggregation process using them. All the rules in this process are of unconditional type, and then the aggregation is an extremely simple construction process, based on a search for minimum values [6]. The figure 4 shown an example of this process, with M = 3 for simplicity.

[1] The knowledge of an appreciation of believe over a solution results from the study of previous data, by applying to the problem a decision making generated by a_h, taking into account as many measurable parameters as we need (we are going to show our ideas to automized this process in a forthcoming paper).

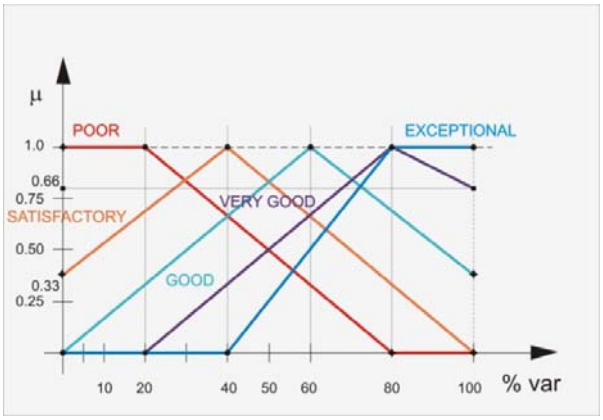

Fig. 3 – Membership functions

The evaluation of parameters in this example, as we can see in the previous figure, is:

Parameter1 = Satisfactory
Parameter2 = Good
Parameter3 = Poor

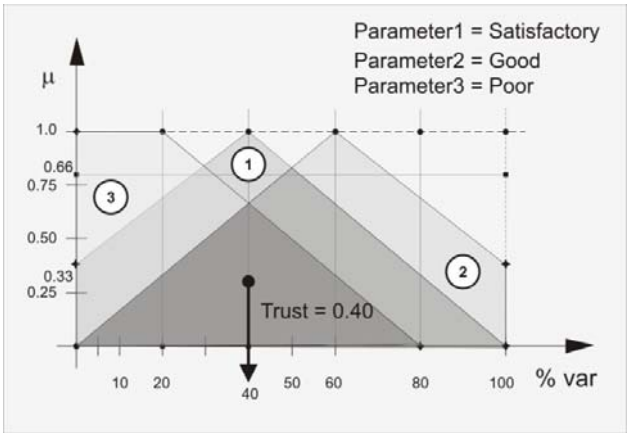

Fig. 4 – Aggregation process

We show in detail how to proceed if we want to use the aggregation process, by means of an algorithm called Agent Profile (AP) [6]. This process concludes in a fuzzy region, being necessary a defuzzification step, using procedures such as COG, for obtaining a crisp value (the value 0.40 in figure 5). If we repeat these process N times, it gives us all the values that we need to find the level of trust between any pair of agents as a consequence of previous decisions making.
As it is reasonable to think, not all the problems have the same importance. We need to introduce a parameter η, $0 \leq \eta \leq 10$, to evaluate the relative importance among them, and trough the function:

$$r_{k,h}^{trust} = COG\ [\eta_1 Trust\ (1), \dots\ , \eta_N Trust\ (M)]$$

we can obtain a total trust level. Under our assumptions, to complete the analysis of obtaining the grade of total trust between two agents, it is necessary to add individual appreciation of the degree of expertise of a_h from a_k. The simplest way is to request to a_k an evaluation restricted by the linguistic variables defined in figure 4. Let us suppose that a_h is only a satisfactory decision maker for a_k point of view. Then, it is still necessary to establish two characteristics: vector position and module in the SATISFACTORY area. With regard to the first characteristic, and taking off other parameters, we locate the vector in the center. For the second, we can take the same size that the value obtained in the figure 5 (a coarse approximation), or to incorporate another parameters of the type of η to achieve a more specific trust about previous decision making or trust based on others knowledge. The result is shown in figure 6.

Now, let us consider the MDM model. In it, an agent a_k is allowed to specify the context on which should determine a level of belief about another agent a_h as a decision maker. The domains in a MDM are disjoints, then we can consider all of them as being in a parallel process without interferences. The problem for a single domain has already been solved, and our obstacle in MDM resides in obtaining an integral level of trust of a_k over a_h (starting with an appropriately categorization of decisions in M domains). The solution is simple, and it is enough with applying similar concepts to those used in the determination of a *total unique level of trust*. For this determination, it is useful to add a domain column in Table 1, with values depending on the problem type. Our proposal in this case is based on a split of files of the DB in as many parts as domains they are considered, applying to each part the method previously developed.
If we introduce the *similarity* concept [3, 5] between domains, we can obtain good decision making with a great economy in the computational resources.

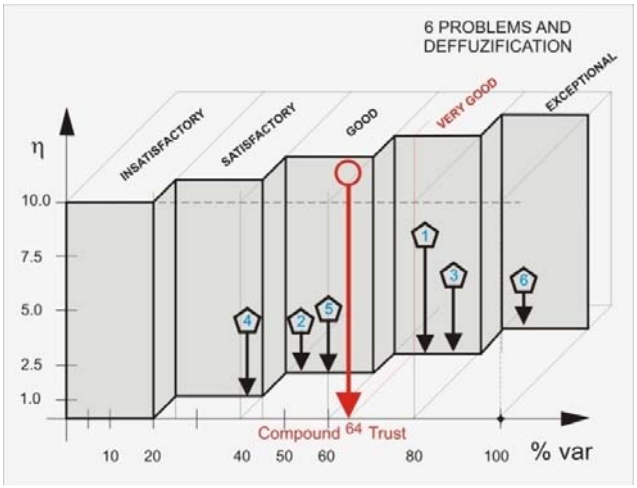

Fig. 5 – Compound Trust Evaluation

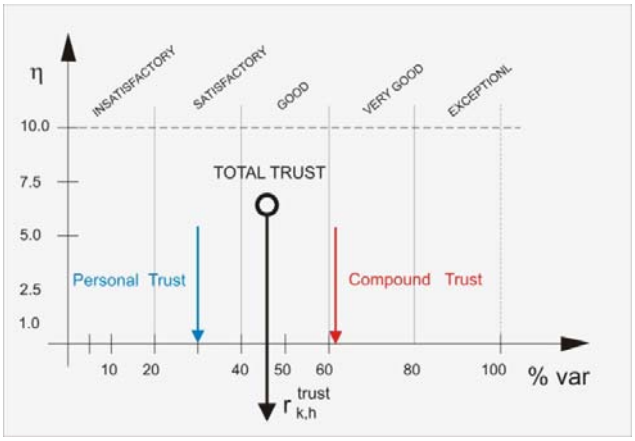

Fig. 6 – Total Trust Evaluation

3. Vote-Based Decision Making

It is important for our purposes to divide the initial population of agents A into Active Agents (AA) and Generating Agents (GA). The GA set contains those agents who offer decision making proposals. In other words, they contribute with solutions to problems. The agents in AA set, on the other hand, participate only by choosing the preferred solutions to them by means of a *voting* process. Let us notice the similarity linking this separation and the ideas behind Cultural Algorithms (CA) [2, 4] have - characteristic used by our group in an investigation currently under development. Anyhow, the dynamic process of the CA indicates that our static separation of the population is an artificial feature in complex environment. This situation is easily shown by a set of agents, taken as solutions providers to certain problems - belonging to GA - but some or eventually all of them transformed in members of AA over other problems. In order to avoid handling too many details in this paper, we will consider two statements:

- ❖ All agents in A will emit one vote
- ❖ The separation of the initial population remains invariant in time

The semantics of this relation between agents and solutions to problems is given by the relation δ = *vote*. If we increase the graph of a trust-based social net with the nodes of problems in P and the nodes of solutions in S (as shown in figure 2), we are confirming the existence of a branch $r^{vote}_{ak,pl\ (sq)}$ between the node a_k and the node s_q, if s_q is a solution provided by a_k to the problem p_l. According to Rodriguez [4] we can find it by means of a conditional probability:

$$r^{vote}_{ak,pl\ (sq)} = p\left(s_q\ is\ a\ good\ solution\ for\ problem\ p_l \mid a_k\ knowledge\ of\ p_l\right)$$

saying that the agent a_k, knowing the problem p_l, considers s_q as a good solution for him. If we calculate this probability for all the possible solutions s_q to a certain problem p_l, it allows us to create a listing of priorities (priority ranking) for the agent a_k. Under the first previous assumption, each agent creates a weighted ranking that an agent provides for the solution set to a particular problem, as their subjective evaluation of the relative optimality of the solutions for the problem. These are called the individual solution rankings. In order to move from an individual solution ranking to a collective solution ranking, an aggregation algorithm is required. Given a trust-based social network and a vote-based decision network, we will use a family of algorithms for aggregating individual votes into a collective solution ranking. All of these algorithms are implemented under the parameterized grammar-based particle SWARM framework [2]. In a particle swarm, a particle is considered an *atom* of decision making influence. They spread over the network, in a stochastic manner; which allows calculating a collective solution ranking for a particular problem. Since particles are discrete, indivisible entities, the diffusion of particles through a network requires a sufficient initial distribution to expose the underlying network topology. The more particles initially supplied to the network, the more accurate the collective ranking.
Let us assign a certain number of particles to each node in the graph. A particle, initially assigned to a_k, follows the network of credibility TBSN and the voting network branches, until they reach solutions nodes in a stochastic way.
At the end of the particle propagation algorithm, when all particles have either been destroyed or have reached a solution node, the distribution of particles over the solution set of the problem determines the collective's solution ranking. Frequently, we apply methods related to political systems to find that collective ranking.

3.1 Direct Democracy

It is based in the idea: one agent → one vote. When we use Direct Democracy (DD) the TBSN is *not* used to calculate the collective decision. If the agent does not vote, then the individual does not participate. If an agent does not participate, then it does not influence the collective solution ranking (the agent doesn't belong to GA or AA). In order to implement this algorithm within the particle swarm framework, each individual is supplied with n particles, which can only traverse a voter branch $r^{vote}_{ak,pl,(sq)}$. If the agent has not voted, then the particle destroys itself. After one step, all particles are either destroyed or are at a particular solution node to problem s_l. The distribution of particles over the solution nodes represents the direct democracy collective solution ranking.

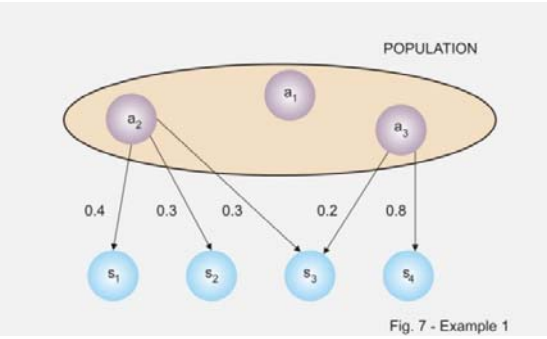

Fig. 7 - Example 1

Let us take a simple example to show how DD work. Figure 7 shows a population formed by three agents a_1, a_2 and a_3, each of them starting with 1000 particles, and four solutions s_1, s_2, s_3 and s_4 of a particular problem, with levels:

$$r^{vote}_{2,p(1)} \rightarrow weight = 0.4$$
$$r^{vote}_{2,p(2)} \rightarrow weight = 0.3$$
$$r^{vote}_{2,p(3)} \rightarrow weight = 0.3$$
$$r^{vote}_{3,p(3)} \rightarrow weight = 0.2$$

$$r_{3,p(4)}^{vote} \rightarrow weight = 0.8$$

After the first step, because particle diffusion is a stochastic process biased by edge weights, we will have accumulated approximately 400 particles in s_1, 300 particles in s_2, 500 particles in s_3 and 800 in s_4. The particles given to a_1 destroyed because a_1 not voted on a solution. The normalized distribution over the solution set is:

$s_1 \rightarrow 400$	normalized: (400x100) / 2000 =	20%
$s_2 \rightarrow 300$		15%
$s_3 \rightarrow 300 + 200$		25%
$s_4 \rightarrow 800$		40%

This normalized distribution is the Direct Democracy collective solution ranking (winner: s_4).

3.2 Dynamic Distributed Democracy (DDD)

The Dynamically Distributed Democracy (DDD) was developed to handle fluctuating levels of participation in a ad-hoc way, to ensure that every individual can influence the collective solution ranking even if it is only through a proxy representative. This algorithm has been shown to be an accurate manner to model the collective's perspective as voter participation decay. In DDD, a particle, if it is unable to take a vote edge to a particular solution, it uses the TBSN to move to a proxy representative. If that representative has voted, then the particle moves to one of the representative's chosen solutions. If the representative has not voted, the particle traverses a trust edge to move to yet another representative. This iterative process continues until a solution to the problem is found.
For example: since the agent a_1 does not vote, and it follows the social network until a_2, since $r_{1,2}^{trust}$ exists. The 1000 particles of a_1 go to a_2, in such a way that now the node gives, in total, 2000 particles when voting, as shown in figure 8. Since the agent a1 doesn't vote, it follows $r_{1,2}^{trust}$ of TBSN delivering 1000 particles to a2. After the migration process:

$s_1 \rightarrow 600$	normalized: (600x100) / 3000 =	20%
$s_2 \rightarrow 1000$		33.3%
$s_3 \rightarrow 400 + 300$		23.3%
$s_4 \rightarrow 700$		23.3%

Now the solution is s_2. This method is very handy in such cases of reduced participation, when ad hoc representative structures emerge to simulate full participation.

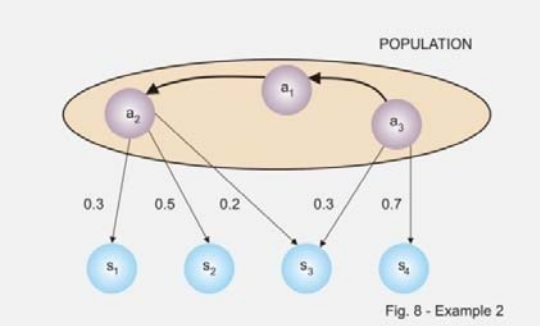

Fig. 8 - Example 2

3.3 Proxy vote

The initial particles distribution is under the influence of the fan-in (node input degree in a graph) of the nodes in A. When the agents of A increase their belief in a_k as good decision maker, this is reflected in an increasing of fan-in and that in turn in a bigger particles initial assignment. This method is very useful in domains where it is known with enough accuracy the level of each agent's experience. In cases of reduced participation, Proxy ad hoc representative structures emerge to simulate full participation (a very nice extension of DDD). For example: in figure 9 we see:

Fan-in		Assignation
$a_1 \rightarrow 1$		$1 \rightarrow 2000$
$a_2 \rightarrow 1$	$\Rightarrow$	$2 \rightarrow 2000$

$a_3 \rightarrow 0$ $3 \rightarrow 1000$

and following the previous example:

$s_1 \rightarrow 1200$	normalized: (1200x100) / 5000 =	24%
$s_2 \rightarrow 1200$		24%
$s_3 \rightarrow 1600 + 500$		42%
$s_4 \rightarrow 500$		10%

Now the solution is s_3.

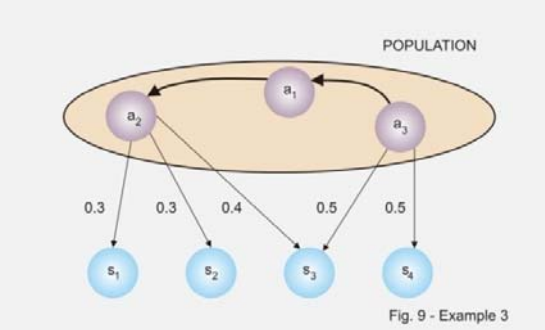

Fig. 9 - Example 3

4 Conclusions

In this paper we outline a methodology of *Collective Decision Making*, posit the use of Fuzzy Logic as an aggregation mechanism to combine the input of individual characteristics as a way of reaching collective decision. In a companion paper [7] we present an interesting technique as the results of lifting restrictions imposed on a social net model, developing from an extended model new forms of finding significant values for Collective Decision Making, using the Kohonen approach, providing a robust and resilient method of classifying collections of data. Presently, our group goal is near to complete a first step of research in CDM, hoping that in a reasonable amount of time, a prototype of social software - useful to our purposes - will be available. We have also left open two interesting problems: 1) estimate *a priori* the number of measurable parameters in a DB and 2) how to use a Cultural Algorithm as a very efficient and dynamic approach to achieve a partition over an initial population.

References

[1] Bonabeau Eric, Dorigo Marco and Theraulaz Guy. Swarm Intelligence From Natural to Artificial Systems. Oxford University Press, 1999.
[2] Kennedy J. and Eberhart R.C. Particle Swarm Optimization. Proceedings IEEE International Conference on Neural Nets, Perth, Australia, IEEE Service Center, 12-13. 1995
[3] Reynolds, R and Sverdlik, W. Problem Solving Using Cultural Algorithms. *Proceedings of 1th. IEEE World Congress on Computational Intelligence*, Orlando, Florida, 1994, pp. 1004-1008.
[4] Rodriguez, Marko et. al. Smartocracy: Social Networks for Collective Decision Making. 40th Hawaii International Conference on Systems Science (HICSS-40 2007), 3-6 January 2007, Waikoloa, Big Island, HI, USA.
[5] Rovarini, Pablo and Cerviño, M. de las Mercedes. Algoritmos Culturales, Lógica Fuzzy y Sistemas Sociales I. *Revista Ciencia, Tecnología y Medio Ambiente*, Año VII, Número 7– ISSN 1667 – 457X, FRT, UTN, 2008, pp. 42-53.
[6] Rovarini, Pablo and Cerviño, M.de las Mercedes. Evaluación Fuzzy de Tests en Humanidades (I). Ponencia aceptada en Urbi et Orbi, Congreso sobre *Ciencias, tecnologías y culturas. Diálogo entre las disciplinas del conocimiento. Mirando al futuro de América Latina y el Caribe*, Universidad de Santiago de Chile-USACH, noviembre 2008.
[7] Rovarini, Pablo and Cerviño, M. de las Mercedes. Globalización: Necesidad de Tomas de Decisiones Colectivas II. En proceso de revisión, presentado a *Revista Ciencia, Tecnología y Medio Ambiente*, ISSN 1667 – 457X, FRT, UTN, 2009.
[8] Zadeh, Lotfi A. Fuzzy Sets. *Information and Control*, Vol. 8, No. 3, 1965, pp. 338-353.